

Módulo 1. Machine learning y sus aplicaciones

Introducción

En este módulo, nos adentraremos en los conceptos iniciales relacionados al *machine learning* como disciplina y sus diversos usos en la vida cotidiana, que relacionan, incluso, nuestro día a día con el aprendizaje de máquinas.

Junto con ello, lograremos entender y diferenciar los tipos de aprendizaje: supervisado y no supervisado, para, posteriormente, aplicar procesamientos de set de datos bien preparados y ajustados para operar sobre diversos algoritmos de *machine learning*.

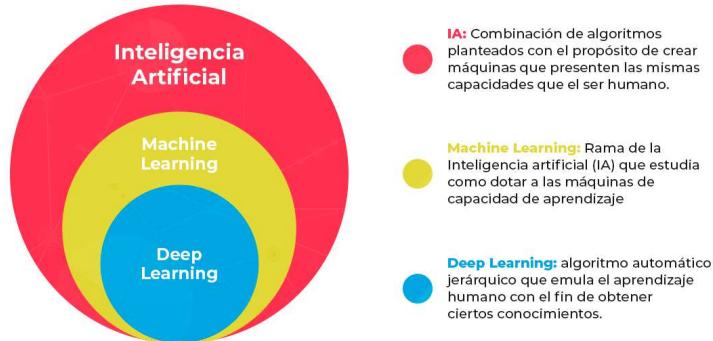
Video de inmersión

Unidad 1: Machine learning y sus aplicaciones

Tema 1. ¿Qué es *machine learning*?

El *machine learning*, también conocido como aprendizaje automático, es una subcategoría inserta dentro de lo que es la inteligencia artificial. Permite que las computadoras o máquinas aprendan patrones o conductas sin la necesidad de programar, obligatoriamente, paso a paso para obtener resultados y predicciones de un problema.

Figura 1. *Machine learning*



Fuente: [Imagen sin título sobre machine learning]. (s. f.). <https://bit.ly/3M82Yzb>

En palabras de José Luis Espinoza, *data scientist* de BBVA México, el *machine learning* puede ser considerado como: “un maestro del reconocimiento de patrones. Es capaz de convertir una muestra de datos en un programa informático capaz de extraer inferencias de nuevos conjuntos

de datos para los que no ha sido entrenado previamente” (BBVA, 8 de noviembre de 2019, <https://www.bbva.com/es/machine-learning-que-es-y-como-funciona/>).

Desde una mirada más interconectada, también es interesante la definición de Drew Conway, citado en *Machine learning para todos*, quien define al *machine learning* como “la unión de habilidades con ordenadores y las matemáticas y estadística” (2018, <https://machinelearningparatodos.com/que-es-el-aprendizaje-automatico-o-machine-learning/>). El diagrama anterior nos permite entender en qué ubicación podemos encontrar el *machine learning* y su cruce con la ciencia de datos.

Este conocimiento de conductas se vuelve una habilidad indispensable para solucionar problemas de la vida cotidiana que tengan como resultado clasificar personas, gustos, estilos de música, problemas de salud, etc.

Historia del machine learning

Antes de comenzar con esta historia, debemos tener muy en claro que el *machine learning* no es un tema de futuro, debido a que gran parte de las decisiones basadas en grandes volúmenes de datos se realizan mediante diversas técnicas de aprendizaje automático que son, prácticamente, una obligación empresarial hoy en día.

Demos un vistazo a distintas piezas de un *puzzle* que han ayudado a que el aprendizaje de máquinas sea tan fundamental:

- **1812 - Teorema de Bayes:**

Definición de probabilidades de que ocurra un evento basándose en el conocimiento de un hito anterior relacionado con el evento que ya ocurrió. Por ejemplo, la probabilidad de que llueva un jueves, dadas las características de una lluvia ocurrida el día anterior.

- **1940 - Bases de la programación informática:**

Se establecen las bases de máquinas que permiten seguir una serie de acciones secuenciadas en un computador.

Sobre estos descubrimientos, 10 años más tarde, Alan Turing, matemático considerado como uno de los padres de la ciencia de la computación y la informática moderna, se cuestionó la posibilidad de que las máquinas puedan pensar mediante inteligencia artificial y que puedan tener la capacidad de resolver tareas humanas, como escribir o reconocer imágenes a diferentes volúmenes y escalas.

- **1950 – Primeras investigaciones de máquinas inteligentes en relación con redes neuronales:**

Se crean las primeras neuronas artificiales basadas en computación, que tienen como finalidad transmitir información entre ellas. Para lograr esto, se creó un programa informático que tenía como desafío lograr sacar una bolita de un laberinto, sin tener alguna programación previa para poder lograrlo, sino que aprendía, a partir de ejemplos de salidas que se proporcionaban con anterioridad.

- **1950 – IBM crea el primer programa exitoso del juego de mesa de damas**
- **1967 – K vecinos más próximos**

Se desarrolla el algoritmo *Nearest Neighbors* que es considerado el nacimiento de los algoritmos de clasificación supervisada, la cual consta del reconocimiento de patrones más cercanos a una vecindad de datos.

- **2003 – Estudios de sistemas de ficheros distribuidos Google File System (GFS):**

Se presenta un nuevo paradigma de procesamiento distribuido al que se llamará *Map & Reduce*.

- **Actualidad - Asistentes personales y estudios**

Hoy en día, el *machine learning* está presente en distintos campos de aplicación, desde la toma de decisiones estratégicas de empresas, que permiten predecir datos de fuga de clientes, segmentación de mercados mediante criterios puntuales; reconocimiento de rutas más cortas de viajes; uso de asistentes personales mediante Siri (Apple), Alexa (Amazon); así como en el campo de la medicina permite predecir enfermedades mediante datos e imágenes.

Actividades

La empresa Máquina Artificial, dedicada a la importación de computadores para el cono sur de la región, está pensando en utilizar machine learning para predecir datos de los posibles clientes que puedan dejar de comprar productos en el corto plazo. Con esta información, la empresa puede tener un gran apoyo de datos que permita:

No contactar más a esos clientes

Saber cuántos computadores van a sobrar a la empresa

Ayudar a la toma de decisiones estratégicas de la empresa

Justificación

Tema 2. Ejemplos y algoritmos de aplicación

En relación con el *machine learning*, existen infinidades de aplicaciones y situaciones en las cuales podemos encontrarlo. Resulta indispensable para la toma de decisiones en distintos tipos de negocios e industrias, ya sea desde invertir más en publicidad dentro de una empresa, hasta ayudar en la decisión de realizar una operación a un paciente que podría tener una enfermedad.

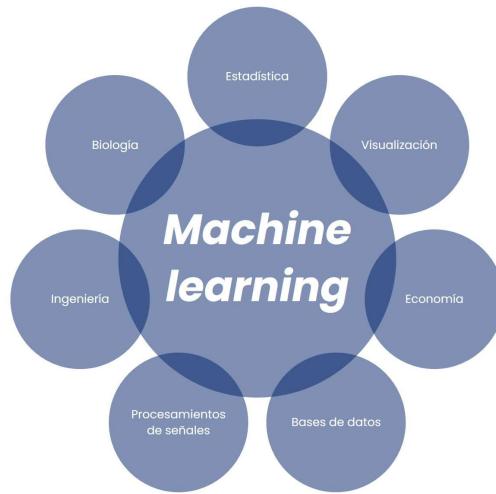
En la actualidad, el *machine learning* ronda nuestro día a día cuando realizamos una búsqueda en Google, para ello, utilizamos un motor de búsqueda o, tal vez, cuando usamos el reconocimiento de voz en celulares, controles remotos de *smart TV*, detección de rostro en celulares (*face ID*), etc.

Otras aplicaciones más amplias de su uso son:

- Uso de antivirus.
- Genética. En la clasificación de secuencias de ADN.
- Determinación de la ruta más corta de transporte (*Waze, Google Maps*).
- Predicción del clima de una zona determinada.
- Comprensión de textos (lectura y transcripción de voz).
- Vehículos autónomos y robots.
- Análisis de imágenes de alta calidad.
- Análisis de comportamiento de consumo y productividad. Para la identificación de clientes potenciales y, en general, en sectores de telecomunicaciones, banca y seguros.

Como podemos ver, el *machine learning* es un mundo que nos facilita la vida y la experiencia de distintos servicios (como usuarios). También, nos permite analizar grandes volúmenes de datos para tomar decisiones complejas que permiten avanzar y crear valor dentro de un sinnúmero de empresas, aplicaciones y todo lo que nos podamos imaginar.

Figura 2. Machine learning



Fuente: elaboración propia.

Actividades

Determina si la siguiente afirmación es verdadera o falsa:

Chic Terrex comentaba a sus compañeros de trabajo que, en el mundo educativo, también se puede utilizar machine learning, ya que nos permitiría predecir si un alumno puede desertar de su carrera de estudio por las zapatillas que usa, la camisa que utiliza o el tipo de corte de cabello que tiene.

Verdadero

Falso

Justificación

Tema 3. Aprendizaje supervisado

El aprendizaje supervisado nos ayuda a resolver problemas que son conocidos, ya que utiliza conjuntos de datos que se han usado anteriormente para hacer aprender a una máquina. Para ello, es necesario tener información previamente etiquetada que permita entrenar un algoritmo que realice tareas específicas.

Veamos esto desde un ejemplo de nuestra vida cotidiana, para así lograr entender el funcionamiento del aprendizaje en un algoritmo.

Cuando éramos muy pequeños, nuestros padres nos entregaban información previamente etiquetada y nosotros realizábamos el trabajo de *machine learning* en nuestra mente.

Por ejemplo, salíamos a una plaza y nos encontrábamos con un perro o un gato.

Ante ello, nuestros padres nos decían "ese es un perro", "ese es un gato".

Con esa información, podíamos hacer una relación de las características de cada uno de los animales y así asignarles una etiqueta. Y, en caso de que no tuviésemos claridad sobre si el animal que veíamos era un perro o un gato, nuestros padres nos decían:

"No, esto no es un perro. Esto es un gato".

De esta misma forma, el aprendizaje supervisado trabaja bajo modelos que permiten predecir resultados con datos conocidos, por ejemplo:

Un proceso de aprendizaje supervisado podría consistir en clasificar vehículos de dos y cuatro ruedas a partir de sus imágenes.

Los datos tendrían que estar correctamente etiquetados para identificar si un vehículo es de dos o cuatro ruedas.

El aprendizaje supervisado permite que los algoritmos 'aprendan' de datos históricos de entrenamiento y los apliquen a entradas desconocidas para obtener la salida correcta.
(TIBC, s. f., <https://www.tibco.com/es/reference-center/what-is-supervised-learning>).

Otros ejemplos:

Clasificaciones binarias (0 o 1):

Clasificación de etiquetas con datos de entrada en uno de dos grupos posibles. A menudo, una de las clases indica un estado "normal/deseado" (valor 1) y la otra indica un estado "anormal/no deseado" (valor 0). Algunas de las aplicaciones prácticas de la clasificación binaria más clásicas incluyen (TIBC, s. f.):

- **Fraude bancario:**

La detección de fraude bancario por parte de entidades financieras (fraude / no fraude).

- **Deserción estudiantil:**

Deserción estudiantil en una carrera técnica (alumno deserta / alumno no deserta).

- **Detección de spam**

El algoritmo recibe ejemplos de correos electrónicos que están etiquetados como "spam" o "no spam" durante la fase de aprendizaje supervisado. Posteriormente, cuando el algoritmo recibe

una nueva entrada de correo electrónico, predice si el correo corresponde a un "spam" o "no spam". (TIBC, s. f., <https://www.tibco.com/es/reference-center/what-is-supervised-learning>).

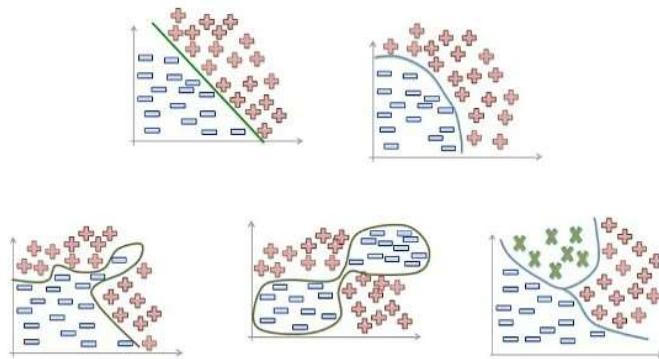
- **Predicción de fuga de clientes**

El algoritmo utiliza un conjunto de datos de entrenamiento de clientes que previamente cancelaron la suscripción de un servicio. Según el entrenamiento, el algoritmo predice si un nuevo cliente finalizará la suscripción o no, en función de los parámetros de entrada.

- **Predicción de conversión**

El algoritmo se entrena con los datos del comprador y si compró el artículo o no. Luego, basándose en esta capacitación, el algoritmo predice si un nuevo cliente realizará una compra o no. (TIBC, s. f., <https://www.tibco.com/es/reference-center/what-is-supervised-learning>).

Figura 3. Ejemplo gráfico de clasificación de datos mediante el uso de etiquetas en machine learning



Fuente: elaboración propia.

Consideraciones del aprendizaje supervisado:

- En esta modalidad de *machine learning*, los algoritmos aprenden de los datos introducidos por una persona [o de un set de datos con resultados conocidos].
- Se necesita la intervención humana para etiquetar, clasificar e introducir los datos en el algoritmo.
- El algoritmo genera datos de salida esperados [etiquetas esperadas], ya que en la entrada han sido clasificados por alguien. (BiSmart, s. f., <https://blog.bismart.com/diferencias-machine-learning-supervisado-no-supervisado>).
- Si las etiquetas no han sido clasificadas de manera correcta, por consecuencia, la máquina aprenderá de manera errónea y entregará resultados que no se condicen con lo esperado.

Actividades

Según lo aprendido sobre aprendizaje supervisado, determina cuál de las siguientes opciones no correspondería a un modelo que permita predecir información supervisada:

Fraude bancario

Detección de spam

Deserción estudiantil

Gustos de verduras verdes en niños de 3 años

Justificación

Tema 4. Aprendizaje no supervisado

El aprendizaje no supervisado nos ayuda a resolver problemas que no son conocidos, ya que utiliza conjuntos de datos que no se han utilizado anteriormente para hacer aprender a una máquina. En otras palabras, este tendrá que aprender a encontrar similitudes de información mediante agrupaciones y relaciones que son más grandes, por ejemplo, dividir un set de 1000 datos en 50 agrupaciones que puedan tener semejanza en sus patrones.

Veamos esto desde un ejemplo de nuestra vida cotidiana, para así lograr entender el funcionamiento del aprendizaje en un algoritmo.

Cuando éramos niños, comíamos arvejas y, quizá, no nos gustaban. Posteriormente, comíamos pepino y puede que no nos gustara tampoco o, incluso, algún otro tipo de verduras de color verde tampoco nos gustaban.

Para ello, nuestros padres no nos decían que las verduras que tienen color verde no nos gustarían en su totalidad, sino, más bien, esto depende de nuestro aprendizaje y gustos.

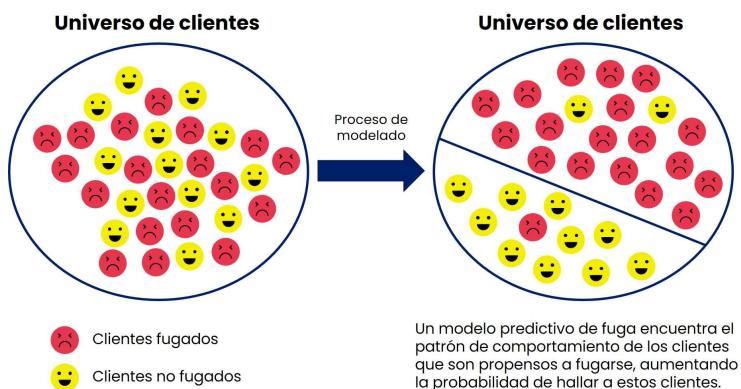
En este caso, si nuestro paladar nos decía “todas aquellas verduras que tengan color verde no te van a gustar, el “no te van a gustar” es nuestra etiqueta.

De esta manera, los algoritmos o modelos intentan darles sentido a los datos por sí mismos, mediante la búsqueda de patrones y características que sean semejantes a la observación de los datos.

Clasificaciones clásicas que surgen en el aprendizaje no supervisado nos pueden ayudar a responder preguntas del tipo:

- ¿Están surgiendo nuevos grupos masivos de fraude?
- ¿Están surgiendo nuevos patrones de compra para la promoción de moda de otoño-invierno?

Figura 4. Proceso de modelado



Fuente: elaboración propia.

En la vida cotidiana, el *data scientist* o analista de datos puede realizar agrupaciones de aprendizaje no supervisado para encontrar respuestas a problemas de áreas como:

- **Marketing:**

Encontrar grupos de clientes con un comportamiento similar, dada una gran base de datos de clientes que contiene sus propiedades y registros de compras anteriores.

- **Biología:**

Clasificación de plantas y animales dadas sus características.

- **Estudios de terremotos:**

Aglomeración de epicentros de terremotos observados para identificar zonas peligrosas.

Desventajas del aprendizaje no supervisado

Si bien ambos modelos de aprendizaje —supervisado y no supervisado— son altamente utilizados, el aprendizaje no supervisado tiene ciertas desventajas respecto a la precisión de la información que uno quisiera obtener, a continuación, se detallan algunas consideraciones:

- No se puede obtener información precisa con respecto a la clasificación de datos. La salida de datos utilizados en el aprendizaje no supervisado es no etiquetada y no se conoce.
- [Hay menor precisión de los resultados, esto] se debe a que los datos de entrada no son conocidos y no están etiquetados de antemano por la gente. Esto significa que la máquina requiere hacer esto por sí misma.
- El usuario necesita dedicar tiempo a interpretar y etiquetar las clases que siguen esa clasificación. (Aprende IA, s. f., <https://aprendeia.com/aprendizaje-no-supervisado-machine-learning/>).

Para lo último, se necesita cierta experticia para entender los fenómenos que están ocurriendo tras un negocio.

Por último, la clave para obtener una ventaja competitiva en el mercado específico está en el uso eficaz de los datos. Los algoritmos de aprendizaje no supervisado te ayudan a segmentar los datos para estudiar las preferencias de tu público objetivo o ver cómo reacciona un virus específico a un antibiótico específico. (Aprende IA, s. f., <https://aprendeia.com/aprendizaje-no-supervisado-machine-learning/>).

Con esta información ya podemos entender de qué manera se clasifican las bases del *machine learning* y comprender la gran utilidad que nos da su existencia.

Como usuarios, veremos que nuestra mente funciona como el *machine learning* y las máquinas nos facilitan la vida y nos entregan comodidad para realizar distintas funciones que son parte de nuestro día a día, gracias al aprendizaje de datos que a esta se le otorga.

Actividades

Según lo aprendido sobre aprendizaje no supervisado, determina cuál de las siguientes opciones correspondería a un modelo que permita predecir información de este tipo:

Clasificación de plantas y animales, dadas sus características

Detección de fraudes bancarios

Detección de spam

Deserción estudiantil

Justificación

Unidad 2: Preparando datos para su análisis

Tema 1. Predicción de datos

A menudo, encontraremos que se habla del concepto de la predicción de datos igual que como se habla de análisis predictivo.

El análisis predictivo es el proceso de utilizar información para realizar predicciones basadas en datos. En este proceso, se hace uso de los datos junto con técnicas analíticas, estadísticas y de aprendizaje automático a fin de crear un modelo para predecir eventos futuros. (Math Works, s. f., <https://es.mathworks.com/discovery/predictive-analytics.html#:~:text=>).

En este contexto, el análisis predictivo tiene un espacio muy ganado en el contexto del *big data* y su amplia gama de información en diversas industrias, como hemos visto anteriormente.

Considerando que cada vez existe mayor tendencia en los negocios a tomar decisiones basadas en datos, lo que se conoce como cultura *data driven*, en esta instancia, comenzaremos a responder a preguntas que surgen luego de entender lo que es el aprendizaje supervisado y no supervisado:

¿Qué tan difícil es crear un modelo predictivo? ¿Cómo puedo crear un algoritmo que sea capaz de aprender la información que le quiero entregar?

Para ello, aprenderemos nuevos conceptos que nos ayudarán a estructurar nuestro set de datos para el posterior análisis predictivo.

Conjunto de datos (data set):

Es la materia prima del sistema de predicción. Es el histórico de datos que se usa para entrenar al sistema que detecta los patrones. El conjunto de datos se compone de instancias; y las instancias, de factores, características o propiedades. (González, s. f., <https://cleverdata.io/conceptos-basicos-machine-learning/>).

Entrenamiento y validación:

Para entender estos conceptos de *machine learning* y saber si está funcionando bien, supongamos el siguiente ejemplo:

Tenemos un set de 100. 000 datos. Este se deberá dividir en dos partes.

Muestra de entrenamiento: estos son los datos con los que se construye el modelo.

Muestra de validación: es una porción de datos que se usa para validar el modelo (prevenir sobreajuste o subajuste).

Estos se dividen de manera aleatoria, considerando un único conjunto de datos de la siguiente manera.

Figura 5. Muestra de entrenamiento



Fuente: elaboración propia.

A modo estándar, ambas muestras, en general, están compuestas por un 70-30, donde el 70 % de los registros corresponde a la muestra de entrenamiento y el 30 % corresponde a validación.

Por lo cual, para nuestro *data sets* de 10 000 datos deberíamos considerar:

70 000 datos aleatorios que serán parte de la muestra de entrenamiento, mientras que los 30 000 restantes serán parte de la validación del modelo.

En algunos casos, puede usarse una proporción de 80-20 o, incluso, 90-10. El objetivo es evaluar la capacidad predictiva del modelo en un segmento distinto al de entrenamiento.

La finalidad de las muestras de entrenamiento y validación es poder testear bajo distintos escenarios el aprendizaje del algoritmo, en otras palabras:

Con la muestra de entrenamiento (70 000 datos) se le enseña al algoritmo diversos patrones de comportamiento de los datos, los cuales, luego de ser aprendidos, se validan en la muestra de validación, para revisar si el comportamiento es semejante.

En el caso de ser así, quiere decir que el modelo es capaz de aprender las propiedades del conjunto de datos de manera generalizada.

Actividades

Dentro de la predicción de datos, el análisis de los datos cumple un rol fundamental para generar buenos resultados. Para ello, se necesita dividir los datos en dos partes y así poder dar conocimiento a los algoritmos. En este aspecto, ¿cómo se le llama a esta división de datos?

Microfase 1/Microfase 2

Macrofase 1/Macrofase 2

Aprendizaje supervisado/aprendizaje no supervisado

Entrenamiento/Validación

Tema 2. Preprocesamiento de *data sets*

Para esta sección, nos será más fácil entender el preprocesamiento de datos mediante el uso de un *data set* de diabetes, que podemos descargar desde Kaggle, en el siguiente *link*:

<https://www.kaggle.com/datasets/mathchi/diabetes-data-set>

Una vez descargado el archivo, procederemos a revisar el procesamiento de los datos:

Paso 0:

Recuerda que, para poder hacer uso y procesamiento, debes contar con procesadores de datos, ya sea mediante: Google Colab, Jupyter u otro lector de Notebooks de Python.

Paso 1:

Instalación de librerías clásicas para el manejo y manipulación de datos: Pandas, NumPy y Matplotlib.

Figura 6. Instalación de librerías

```

1 # Paquetes numéricos
2 import numpy as np
3
4 # DataFrames/Procesamiento
5 import pandas as pd
6
7 # Gráficas
8 import matplotlib.pyplot as plt

```

Fuente: elaboración propia.

Paso 2: carga y visualización de datos

Recuerda que la librería de Panda te servirá para que tengas preparados tus futuros modelos de *machine learning*.

Métodos de lectura: `read_csv()` para leer y cargar el archivo csv del *data set* en un *DataFrame*.

Método de previsualización: `head()` para ver las 5 primeras filas del contenido del *DataFrame*.

Figura 7. Carga y visualización de datos

```

1 # Leer y cargar el CSV
2 data = pd.read_csv("diabetes.csv")
3
4 # Ver qué hay en el DataFrame
5 data.head()

   Pregnancies Glucose BloodPressure SkinThickness Insulin BMI DiabetesPedigreeFunction Age Outcome
0          6      148           72            35       0  33.6           0.627    50        1
1          1       85            66            29       0  26.6           0.351    31        0
2          8      183           64            0       0  23.3           0.672    32        1
3          1       89            66            23       0  28.1           0.167    21        0
4          0      137           40            35       1  43.1           2.288    33        1

```

Fuente: elaboración propia.

Paso 3: entender los datos – tipos

Podemos entender los tipos de datos de nuestro modelo utilizando **dtypes** para identificar cada columna.

Figura 8. Utilización de dtypes

```

1 data.dtypes

```

Pregnancies	int64
Glucose	int64
BloodPressure	int64
SkinThickness	int64
Insulin	int64
BMI	float64
DiabetesPedigreeFunction	float64
Age	int64
Outcome	int64
dtype: object	

Fuente: elaboración propia.

En este caso:

int64 representa que la variable es de carácter entera. Los enteros sirven cuando se trata de etiquetar algo o se tienen grupos discretos. Por ejemplo, alguien puede tener 1 o 2 hijos, pero no 1.5 hijos.

También sirven para representar una categoría de mapeo. Por ejemplo, imaginamos que tenemos 3 ciudades, donde "1" significa New York City, "2" significa San Francisco y "3" significa Philadelphia.

float64 representa valores numéricos de punto flotante. Estos sirven para representar valores continuos, por ejemplo, BMI o insulina, donde decimales/fracciones son permitidas.

Paso 4: entender los datos y valores faltantes

Antes de entrar al análisis preliminar, revisamos que no haya ningún valor faltante. SueLEN indicarse como *NaN* o *Not a Number*.

Usamos el método **dropna()** para revisar si existen filas que no deban ser contadas.

dropna() removerá la fila si una sola columna individual tiene un *feature* faltante. De forma alternativa, podemos especificar si este es el acercamiento indicado o si queremos eliminar solo filas donde falte toda la información de la columna.

Revisemos si existen datos faltantes en nuestro *dataframe*:

data.isna() permite mirar el *dataframe* y categorizar como valor true donde existen datos NA.

Figura 9. Data.insa

1 data.isna()									
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
...
763	False	False	False	False	False	False	False	False	False
764	False	False	False	False	False	False	False	False	False
765	False	False	False	False	False	False	False	False	False
766	False	False	False	False	False	False	False	False	False
767	False	False	False	False	False	False	False	False	False

Fuente: elaboración propia.

En este caso, tenemos una previsualización de la BBDD, pero no sabemos qué pasa con los datos centrales, por lo cual, utilizaremos una suma de valores *false* y *true* para determinar si hay datos faltantes.

Figura 10. Data.insa

1 data.isna().sum()	
Pregnancies	0
Glucose	0
BloodPressure	0
SkinThickness	0
Insulin	0
BMI	0
DiabetesPedigreeFunction	0
Age	0
Outcome	0
dtype:	int64

Fuente: elaboración propia.

De esta forma, podemos corroborar que nuestro *data set* no cuenta con datos faltantes.

Paso 5: representar el patrón de los datos

Dado nuestro *data set*, queremos ser capaces de decir si alguien tiene o no diabetes, y que ello sea indicado por la columna *outcome*.

Donde “1” indica que el paciente tiene diabetes y “0” sugiere que no la tiene. Queremos aprender si las condiciones preexistentes (*age*, *blood-pressure*, *insulin*, *BMI*) pueden contribuir a la diabetes. Para ello, usaremos histogramas y gráficas de dispersión para ver cómo se comportan los grupos/población de nuestros datos. Es bueno hacer esto antes de construir un modelo.

Creación de histograma

Primero, revisamos las distribuciones de frecuencia. Para esto, utilizaremos un histograma.

Un histograma usa ***bins*** o límites de un valor numérico. Contamos cuántas filas caen dentro de los límites superior e inferior de cada *bin* y esto crea la distribución.

Pandas puede automatizar la creación de histogramas para no contar desde cero.

Podemos darle el número de *bins* o esquinas de los *bins* (el criterio para límites inferior y superior).

Dado que queremos saber si la persona tiene o no diabetes, podemos preguntar si nuestros datos están bien distribuidos en valores de 1 o 0 para el *outcome*.

Necesitamos clases igualmente balanceadas para hacer predicciones decentes.

Revisamos, con la gráfica de barras, el resultado del siguiente código.

Figuras 11. Creación de histograma

```

# Ajustar el tamaño de fuente.
plt.rcParams['font.size'] = 15

# Crear una figura y ajusta su tamaño.
f = plt.figure(figsize=(8,4))

# Crear un subplot o subrama - al ser una sola figura es 1,1,1
ax = f.add_subplot(1,1,1)

# Gráfica tus datos usando 'hist'. Pasa el objeto 'ax' a Pandas. Agrega un borde negro con un grosor de 2.
data["Outcome"].hist(ax=ax, edgecolor='black', linewidth=2)

# Establece los límites en el eje x.
ax.set_xlim([-0.5, 1.5])

# Establece la frecuencia de tick. Tenemos 0 y 1 que corresponden a Sí y No respectivamente.
ax.set_xticks([0, 1])

# Etiquetar xtick labels.
ax.set_xticklabels(["N", "Y"])

# Crea el título.
ax.set_title("Diabetes Y/N?")

# Establece la etiqueta del eje X.
ax.set_xlabel("Answer")

```

Fuente: elaboración propia.

Figuras 12. Creación de histograma

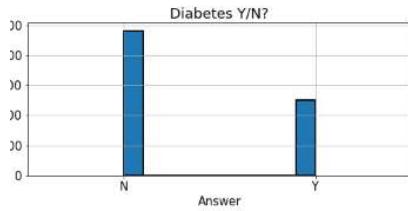
```

# Establece la etiqueta del eje Y.
ax.set_ylabel("Count")

# Establece los límites superior/inferior del eje y.
ax.set ylim([0, 510])

# Míce que las cosas sean bonitas, no es necesario, pero se ajusta al tamaño de la figura.
f.tight_layout()

```



Fuente: elaboración propia.

De lo anterior, podemos ver que tenemos más "No" que "Sí", lo que significa que hay más personas sin diabetes que con diabetes, en este conjunto de muestra.

Consideremos lo siguiente: dentro del aprendizaje supervisado que revisamos en temas anteriores, hablábamos de etiquetar un dato con una variable de salida. En este caso, esta variable *outcome* es la etiqueta de salida en el *data set*, ya que, dentro de todos los patrones de cada paciente, es la que entrega la resolución o respuesta que buscamos a los datos: si el paciente tiene diabetes o no.

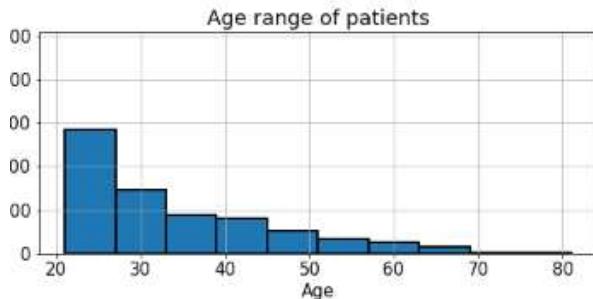
Es importante utilizar los histogramas en distintas variables que nos ayuden a entender el modelo de datos, por ejemplo. También lo realizaremos con la variable **edad** de los pacientes.

Figura 13. Código

```

f = plt.figure(figsize=(8,4))
ax = f.add_subplot(1,1,1)
data["Age"].hist(ax=ax, edgecolor='black', linewidth=2)
ax.set_title("Age range of patients")
ax.set_xlim([0, 510])
ax.set_xlabel("Age")
ax.set_ylabel("Count")
f.tight_layout()

```



Fuente: elaboración propia.

Si resolvimos lo anterior, podremos darnos cuenta de que la mayoría de nuestros pacientes parecen tener entre 20 y 40 años.

Actividades

Respecto a la función `dtypes`, esta permite:

Dividir una muestra en entrenamiento y validación

Contar datos para saber la dimensión de filas y columnas

Cargar la librería Panda

Identificar el tipo de datos que tiene el data set

Justificación

Tema 3. Imputación de variables

La imputación de variables es una parte del proceso de *data sets* que siempre la encontraremos en la gran mayoría de los conjuntos de datos que trabajemos.

Estos datos representan aquellos en los que no se almacena ningún valor de datos en una observación. Es por ello que es de vital importancia identificarlos y trabajarlos para su imputación, pero ¿por qué deberíamos imputar datos?

Esto se debe a que, en *machine learning*, debemos intentar trabajar con los valores lo más limpios y completos posible, para un posterior análisis y entrenamiento de los modelos, por eso,

es muy importante poder tener un buen preprocesamiento e imputación de las variables, previo al ingreso de los datos al modelo.

Revisemos una imputación de datos con el uso de un *data set*.

Figura 14. Revisión de una imputación de datos con el uso de un *data set*

The screenshot shows a Jupyter Notebook cell with the following code:

```
[9] 1 import numpy as np
2 import pandas as pd
3 data = pd.DataFrame({'id': [1, 2, 3, 4, 5, 6, 7, 8, 9],
4                      'area': [1, 2, 3, 1, 2, 3, 1, 2, 3],
5                      'age': [32, 30, np.nan, 23, 27, 44, 67, 23, np.nan],
6                      'amount': [102, 121, 343, np.nan, 121, np.nan, 155, 149, 221]})
```

Below the code, the variable `data` is displayed as a Pandas DataFrame:

	id	area	age	amount
0	1	1	32.0	102.0
1	2	2	30.0	121.0
2	3	3	NaN	343.0
3	4	1	23.0	NaN
4	5	2	27.0	121.0
5	6	3	44.0	NaN
6	7	1	67.0	155.0
7	8	2	23.0	149.0
8	9	3	NaN	221.0

Fuente: elaboración propia.

Como podemos ver, tanto la variable “age” y “mount” contienen valores NaN, para los cuales tenemos las siguientes opciones.

Para poder imputar los valores nulos, podemos utilizar el siguiente código.

Figura 15. Imputación de valores nulos

The screenshot shows a Jupyter Notebook cell with the following code:

```
1 data.dropna()
```

Below the code, the variable `data` is displayed as a Pandas DataFrame, showing only the rows without NaN values:

	id	area	age	amount
0	1	1	32.0	102.0
1	2	2	30.0	121.0
4	5	2	27.0	121.0
6	7	1	67.0	155.0
7	8	2	23.0	149.0

Fuente: elaboración propia.

Este código reducirá la cantidad de datos de la muestra, dejará solo aquellos que son valores numéricos y eliminará las filas en las que se encuentra algún dato faltante.

Este tipo de imputación es una de las tantas que se pueden realizar.

También, podemos tener otras opciones, como reemplazar los valores NaN por el promedio de la variable que queremos completar. Esto se realiza con la finalidad de no perder datos que sean

parte de nuestro modelo.

Observamos cómo podemos hacerlo con el siguiente código.

Figuras 16, 17 y 18. Reemplazo de los valores NaN

```
1 import numpy as np
2 import pandas as pd
3 data = pd.DataFrame({'id': [1, 2, 3, 4, 5, 6, 7, 8, 9],
4                      'area': [1, 2, 3, 1, 2, 3, 1, 2, 3],
5                      'age': [32, 30, np.nan, 23, 27, 44, 67, 23, np.nan],
6                      'amount': [102, 121, 343, np.nan, 121, np.nan, 155, 149, 221]})
```

```
1 data
```

	id	area	age	amount
0	1	1	32.0	102.0
1	2	2	30.0	121.0
2	3	3	NaN	343.0
3	4	1	23.0	NaN
4	5	2	27.0	121.0
5	6	3	44.0	NaN
6	7	1	67.0	155.0
7	8	2	23.0	149.0
8	9	3	NaN	221.0

definimos un nuevo objeto que solo contenga los valores sin NaN

```
[18] 1 data1 = data.dropna()
```

Aplicamos el promedio a los datos de edad del objeto "data1"

```
[19] 1 mean = data1['age'].mean()
```

```
[20] 1 mean
```

35.8

Finalmente reemplazamos los datos de Edad en el data set original

```
[21] 1 data['age'] = data['age'].fillna(mean)
```

Visualizamos el data set completo

[22] 1 data

	id	area	age	amount
0	1	1	32.0	102.0
1	2	2	30.0	121.0
2	3	3	35.8	343.0
3	4	1	23.0	NaN
4	5	2	27.0	121.0
5	6	3	44.0	NaN
6	7	1	67.0	155.0
7	8	2	23.0	149.0
8	9	3	35.8	221.0

Fuente: elaboración propia.

Posteriormente, podemos realizar el mismo procedimiento para completar los datos de la columna amount, ¿te animas a hacerlo tú?

Actividades

Analizar la siguiente afirmación:

Una de las importancias de la imputación de datos es poder eliminar datos faltantes, sin tener la necesidad de reemplazarlos, pues, en la práctica, no hay problema en perder datos en vez de realizar su reemplazo. ¿Es correcta esta afirmación?

Verdadero

Falso

Justificación

Tema 4. Ajuste y sobreajuste de datos

El sobreajuste de datos es uno de los conceptos principales de *machine learning*. Se le denomina sobreajuste cuando un modelo de entrenamiento aprende información que está muy cargada a una etiqueta de datos, lo que tiene como consecuencia que los datos se generalicen de sobremanera y no se logren resultados realistas dentro de la muestra de validación.

Recordemos que uno de los principales objetivos que tiene el *machine learning* es obtener patrones de datos para entrenar que puedan estar siempre disponibles para predecir o inferir de manera correcta los posibles nuevos datos que vayamos a sumar dentro de nuestro *data set*.

En otras palabras, es clave tener patrones generales que puedan extrapolarse a la nueva información para la toma de decisiones.

Pensemos en el caso de nuestra memoria humana, el sobreajuste se produciría si aprendiéramos muchas cosas de memoria sin entender el trasfondo de lo que estamos aprendiendo.

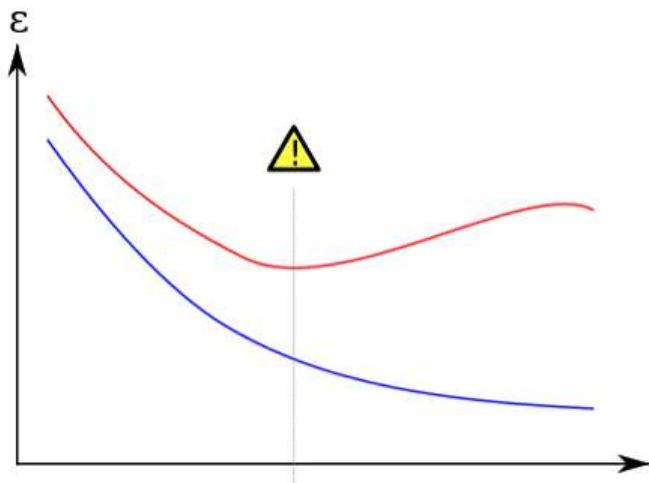
Pero ¿cuándo se produce el sobreajuste?

Esto puede ocurrir cuando el *data set* se entrena demasiado o con datos anómalos, lo que hace que el algoritmo aprenda patrones que no son generales. Aprende características específicas, pero no los patrones generales, el concepto. Los modelos más complejos tienden a sobreajustarse más que los modelos más simples. Además, ante un mismo modelo, a menor cantidad de datos, es más posible que ese modelo se sobreajuste.

¿Cómo evaluarlo?

“Existen varios métodos para evaluar cuándo un modelo se está sobreajustando. Uno de los métodos más conocidos es por medio de las gráficas de errores de entrenamiento y de testeo” (*Machine learning para todos*, 2020, <https://machinelearningparatodos.com/que-es-el-sobreajuste-u-overfitting-y-por-que-debemos-evitarlo/>).

Figura 19. Gráfico



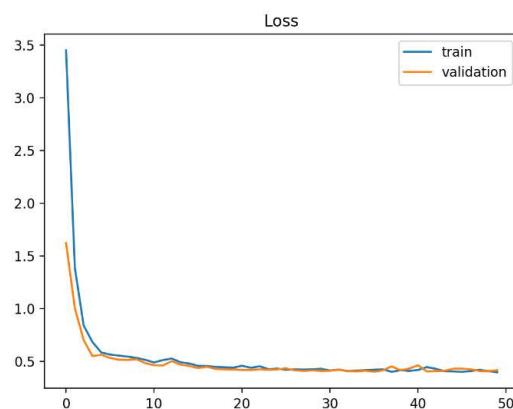
Fuente: Machine learning para todos, 2020, <https://bit.ly/3rzlt6k>

En la siguiente gráfica, consideramos que la línea roja son errores en la muestra de testeos, mientras que la línea azul muestra el error de entrenamiento.

A medida que más datos se tienen de entrenamiento para que el algoritmo aprenda, el error tiende a bajar, pero si a esto lo aplicamos a los datos de testeos, el error tiende a subir, a medida que los datos van acumulando información.

El caso ideal es que ambas líneas, **entrenamiento y test**, estén lo más cerca posible.

Figura 20. Gráfico



Fuente: elaboración propia.

Actividades

Según lo aprendido hasta el momento, ¿qué definición representaría de mejor forma el concepto de sobreajuste de un modelo de datos?

Es cuando un modelo de entrenamiento aprende información que está muy cargada a una etiqueta de datos.

Es cuando un modelo de validación aprende información que está muy cargada a una etiqueta de datos.

Es cuando un modelo de entrenamiento no cuenta con etiquetas de datos.

Es cuando el modelo de validación no elimina los datos faltantes.

Justificación

Video de habilidades

Glosario

Referencias

Aprende IA. (s. f.). ¿Qué es el aprendizaje no supervisado? <https://aprendeia.com/aprendizaje-no-supervisado-machine-learning/>

BBVA. (8 de noviembre de 2019). *Machine learning: ¿qué es y cómo funciona?* <https://www.bbva.com/es/machine-learning-que-es-y-como-funciona/>

BiSmart. (s. f.). Diferencias entre *machine learning* supervisado y no supervisado. <https://blog.bismart.com/diferencias-machine-learning-supervisado-no-supervisado>

González, A. (s. f.). Conceptos básicos de machine learning. <https://cleverdata.io/conceptos-basicos-machine-learning/>

[Imagen sin título sobre *machine learning*]. (s. f.). <https://www.masterdatascienceucm.com/que-es-machine-learning/>

López Briega, R. (s. f.). *Libro online de IAAR*. <https://iaarbook.github.io/autor/>

Machine learning para todos. (2020). ¿Qué es el sobreajuste u overfitting y por qué debemos evitarlo? <https://machinelearningparatodos.com/que-es-el-sobreajuste-u-overfitting-y-por-que-debemos-evitarlo/>

Math Works. (s. f.). Análisis predictivo. <https://es.mathworks.com/discovery/predictive-analytics.html#:~:text=>

Mitchell, T. (1997). *Machine learning*. McGraw-Hill Education.

TIBC. (s. f.). ¿Qué es el aprendizaje supervisado? <https://www.tibco.com/es/reference-center/what-is-supervised-learning>