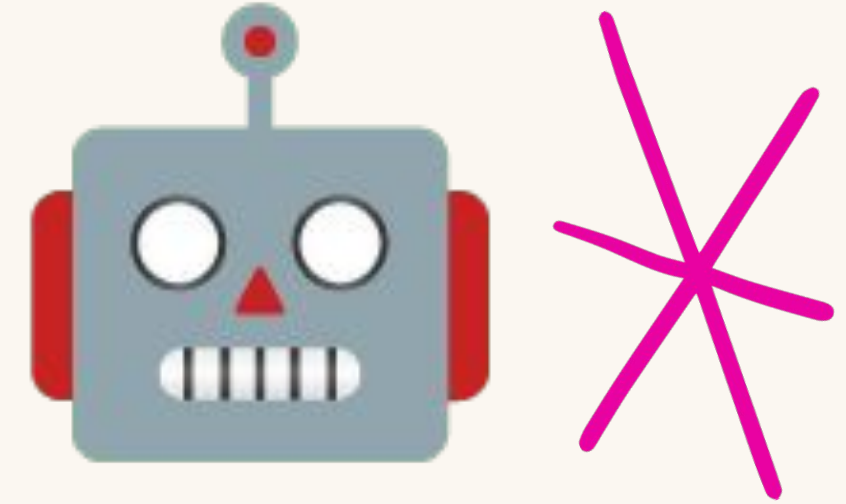


Desbloqueando el Poder de los Datos



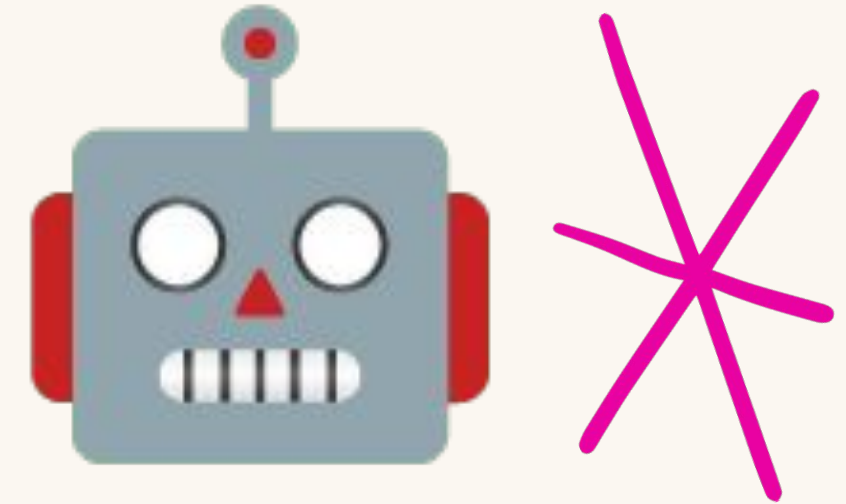
Inteligencia Artificial & Ciencia de Datos para todos

Descanzo. Regresamos a las: 8:05 a.m.

¿Te gustaría comenzar el día con alguna canción en específico?

Coméntala en el chat 🎵💬

Desbloqueando el Poder de los Datos



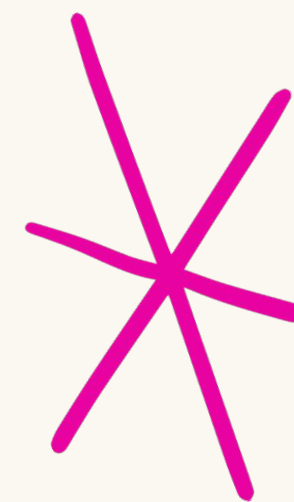
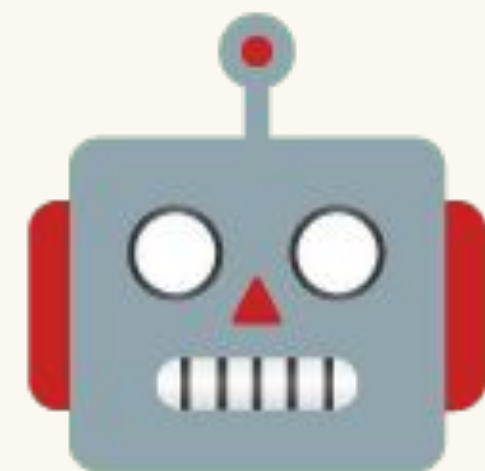
Inteligencia Artificial & Ciencia de Datos para todos

Comenzamos a las 7:05 a.m. en punto.

¿Te gustaría comenzar el día con alguna canción en específico?

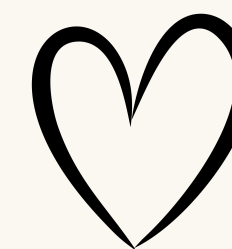
Coméntala en el chat 🎵💬

Desbloqueando el Poder de los Datos



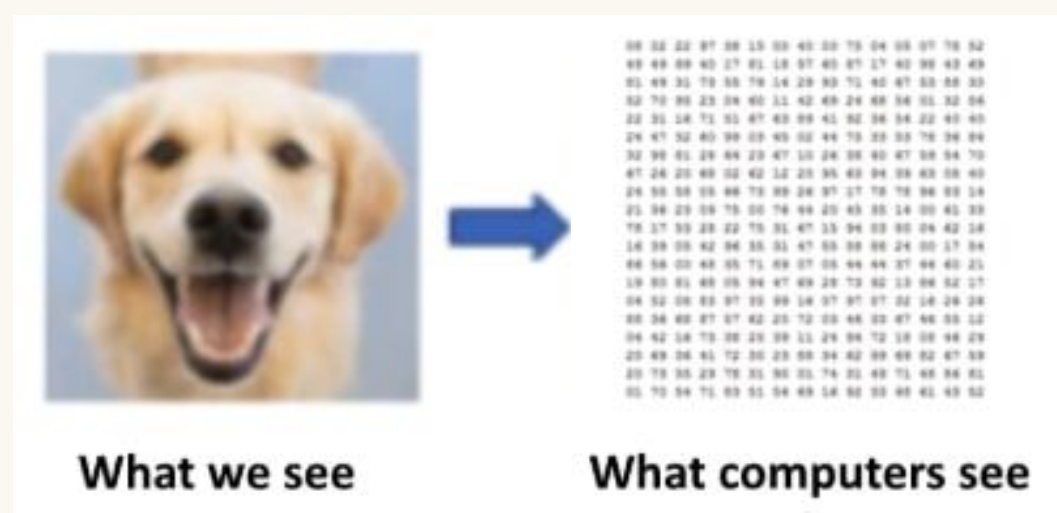
Inteligencia Artificial & Ciencia de Datos para todos





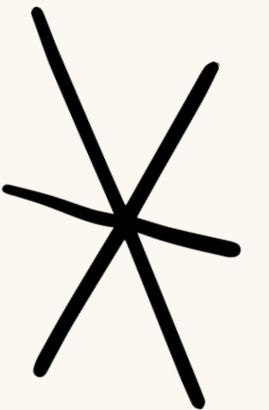
Ingeniería de características

Septiembre 24, 2024





1. Resumen rápido del flujo de trabajo en un proyecto de ciencia de datos
2. Conjunto de entrenamiento y prueba
3. Ingeniería de características
4. `.fit` vs `.transform` vs `fit_transform`
5. Técnicas de Ingeniería de Características
 - Codificación de variables categóricas
 - Escalado de variables numéricas
 - Valores faltantes
 - Cómo manipular texto



Partes de un modelo de Machine Learning

In [4]:

```
import seaborn as sns
df = sns.load_dataset('iris')
df.head()
```

Out[4]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Partes de un modelo de Machine Learning

Entradas del Modelo

Son las variables o datos de entrada que se utilizan para hacer predicciones

También conocidas como:

- Input
- Características (Features)
- Atributos
- Predictores
- Entradas
- Variables independientes
- Dimensiones
- X
- Probablemente más...

In [4]:

```
import seaborn as sns
df = sns.load_dataset('iris')
df.head()
```

Out [4]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Partes de un modelo de Machine Learning

Salidas del Modelo

Son los valores o resultados que el modelo intenta predecir a partir de los datos de entrada

También conocidas como:

- Output
- Objetivo
- Respuesta
- Target
- Salida
- Variable dependiente
- Etiquetas
- Y
- Probablemente más...

In [4]:

```
import seaborn as sns
df = sns.load_dataset('iris')
df.head()
```

Out [4]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Partes de un modelo de Machine Learning

Fila de datos (Input + Output)

Cada fila representa una observación o un caso específico dentro del conjunto de datos

También conocida como:

- Observación
- Punto de datos
- Registro
- Fila
- Probablemente más...

In [4]:

```
import seaborn as sns
df = sns.load_dataset('iris')
df.head()
```

Out [4]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Partes de un modelo de Machine Learning

Etiquetas (en el contexto del aprendizaje supervisado)
Son los valores de las variables objetivo que el modelo intenta predecir

En este caso específico
las etiquetas son:

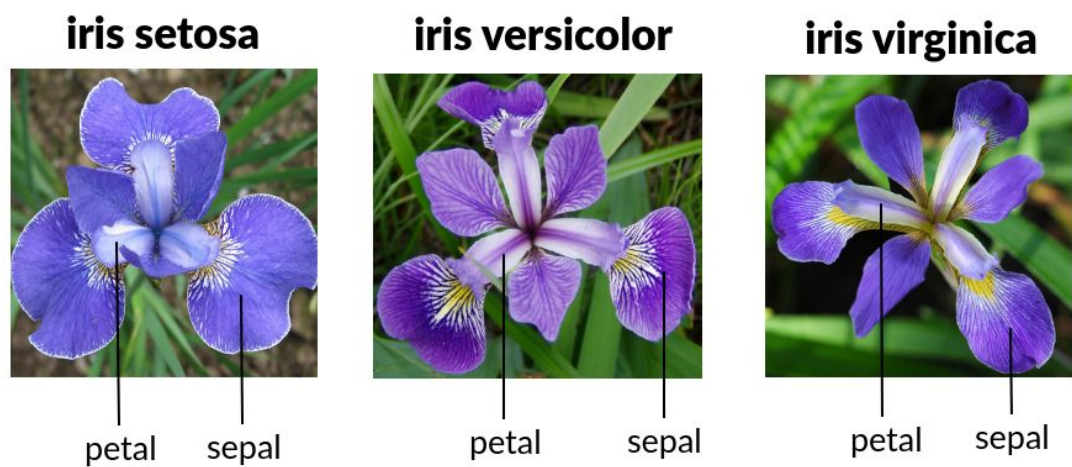
- Setosa
- Versicolor
- Virginica

In [4]:

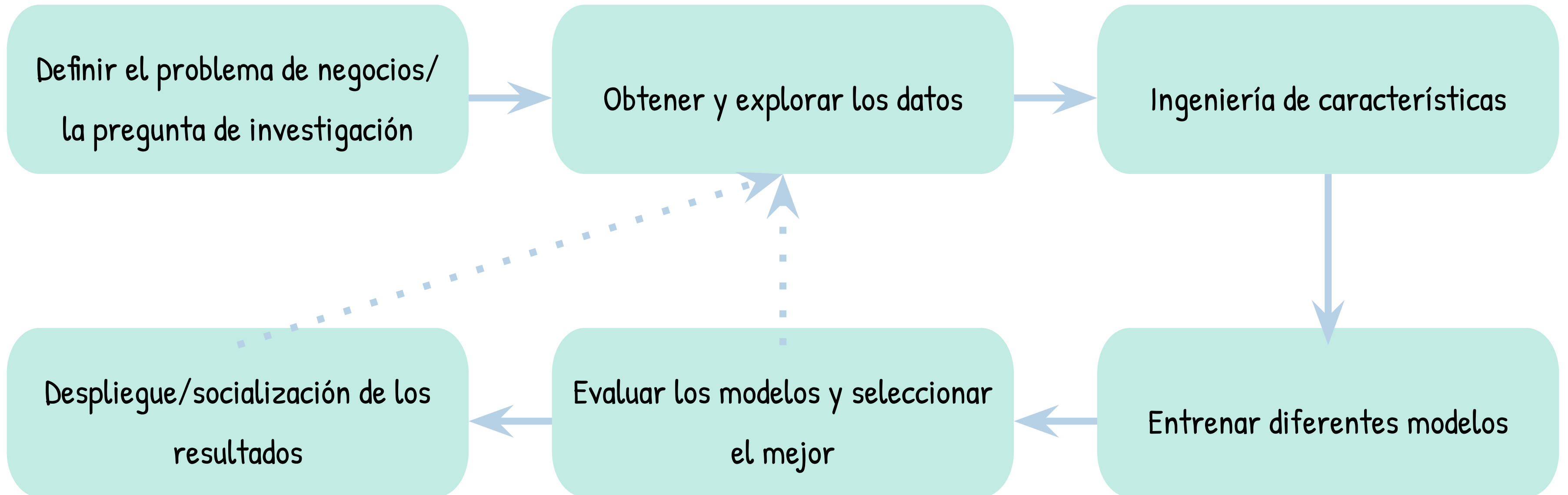
```
import seaborn as sns
df = sns.load_dataset('iris')
df.head()
```

Out [4]:

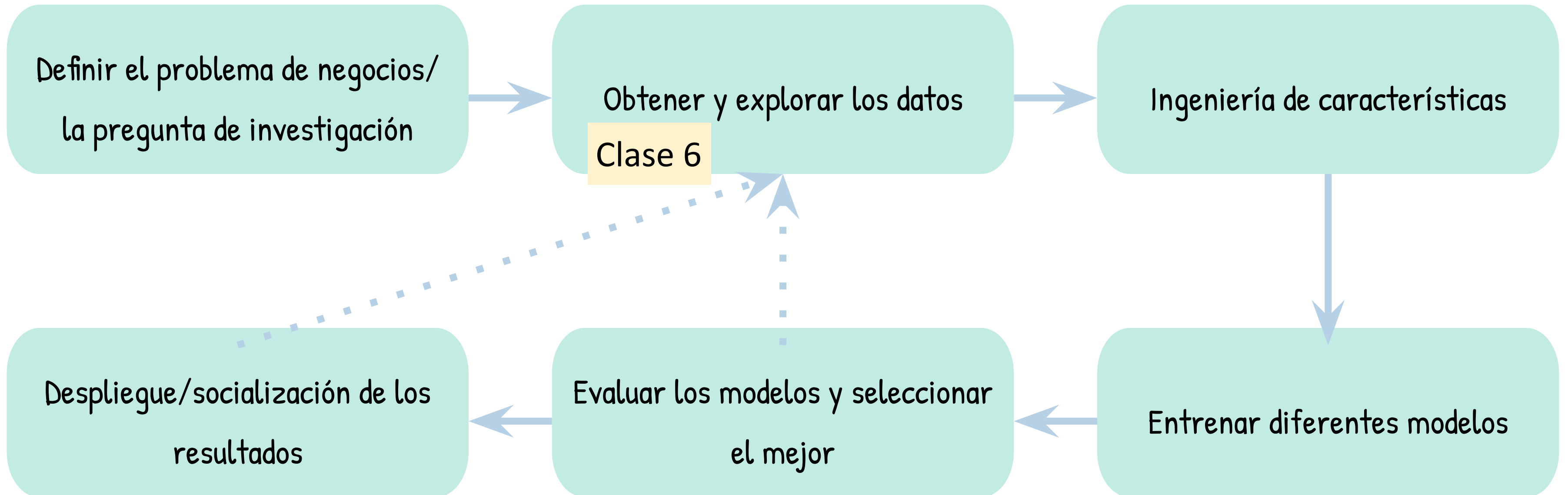
	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa



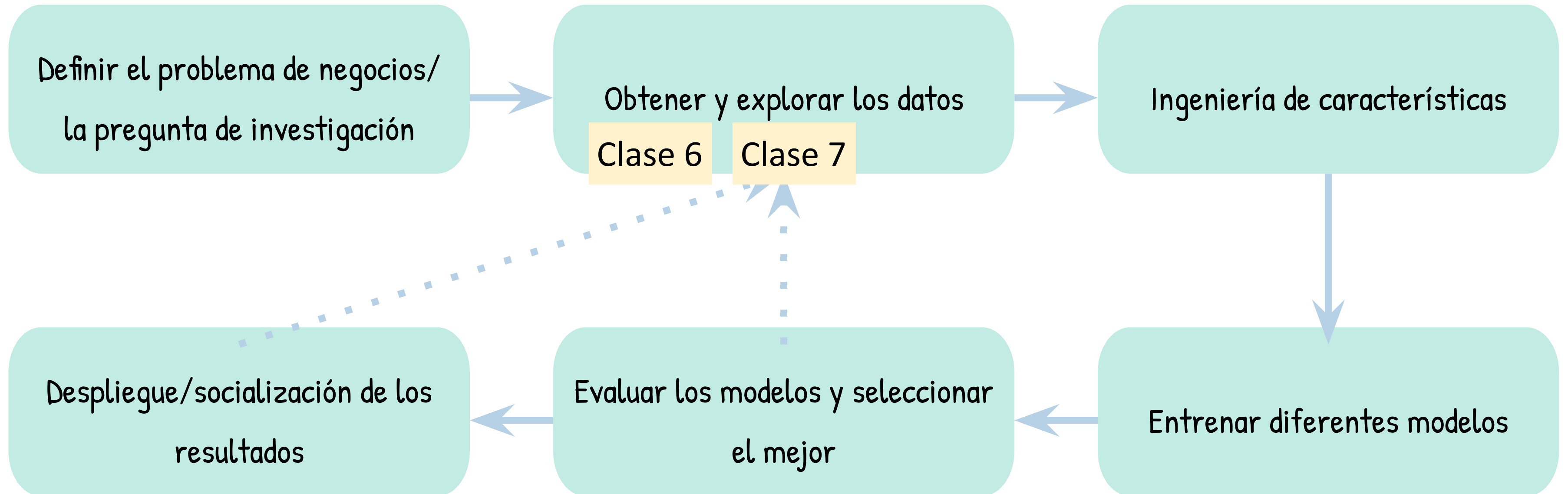
Pasos en un proyecto de Machine Learning



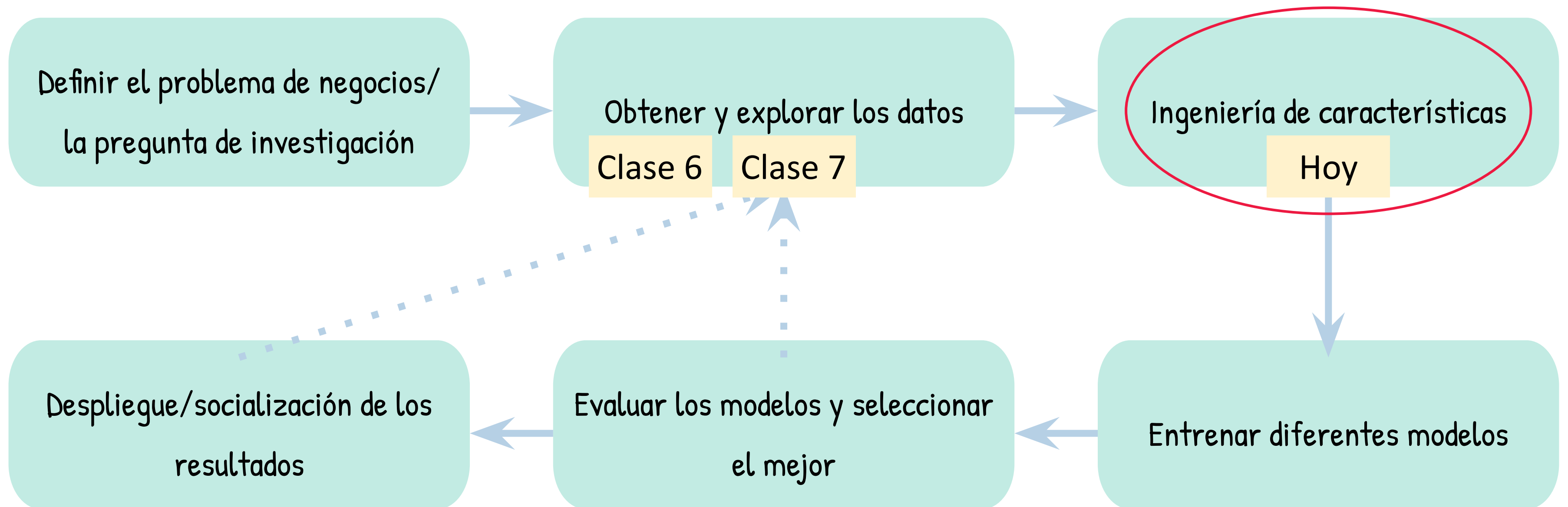
Pasos en un proyecto de Machine Learning



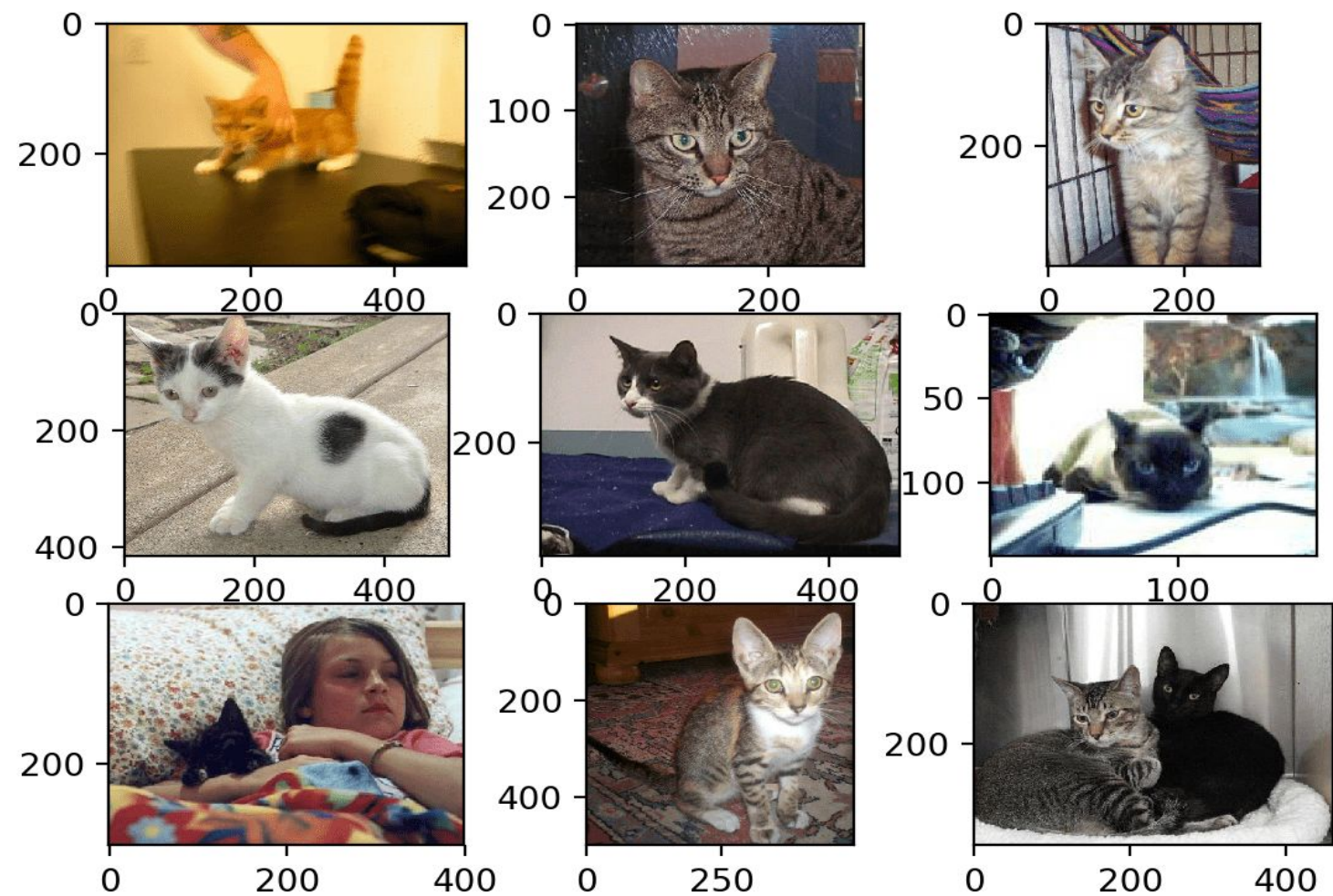
Pasos en un proyecto de Machine Learning



Pasos en un proyecto de Machine Learning



¿Qué es machine learning? (Aprendizaje automático)



El **machine learning** es la misma idea.

En lugar de programar una computadora con un conjunto específico de instrucciones explícitas para realizar una tarea, le proporcionamos a la computadora una gran cantidad de datos y dejamos que aprenda **generalizar** a partir de esos datos.

Al igual que un niño, cuanto más ejemplos tenga la computadora para aprender, ¡mejor será en esa tarea!

¿Cómo podemos comprobar si nuestro modelo funciona?

¿Cómo podemos comprobar si nuestro modelo funciona?

- En Machine Learning, se utiliza el conjunto de datos disponible para construir un modelo que pueda **generalizar** a nuevos datos.
- Para asegurar que el modelo aprenda correctamente y no simplemente memorice el conjunto de datos, dividimos los datos en dos partes:
 - **Conjunto de Entrenamiento** (training set):
 - El modelo "aprende" las relaciones entre las características (features) y la variable objetivo (label) con estos datos.
 - **Conjunto de Prueba** (test set):
 - Este subconjunto de datos **NO se utiliza** durante el entrenamiento. En su lugar, se usa **después** de que el modelo ha sido entrenado para **evaluar** su rendimiento.
 - La idea es simular cómo se comportará el modelo con datos que nunca ha visto antes, para asegurarnos de que no ha simplemente memorizado.

Métodos para dividir los datos en entrenamiento y prueba

- Para la mayoría de casos, la manera más fácil y efectiva para dividir los datos en conjuntos de entrenamiento y prueba es a través de una división aleatoria. Por lo general se dividen el 80% de los datos para entrenamiento y el 20% prueba.
- También existe la división estratificada (según la proporción de los datos), la división para series temporales (según el tiempo de los datos) y otros métodos de división que se pueden aplicar si el problema lo america.

División aleatoria

```
from sklearn.model_selection import train_test_split

# Definir las características (X) y la variable objetivo (y)
X = df.drop(columns=['Survived']) # Aquí estamos eliminando la columna 'survived', que es el objetivo
y = df['Survived'] # Esta es nuestra variable objetivo

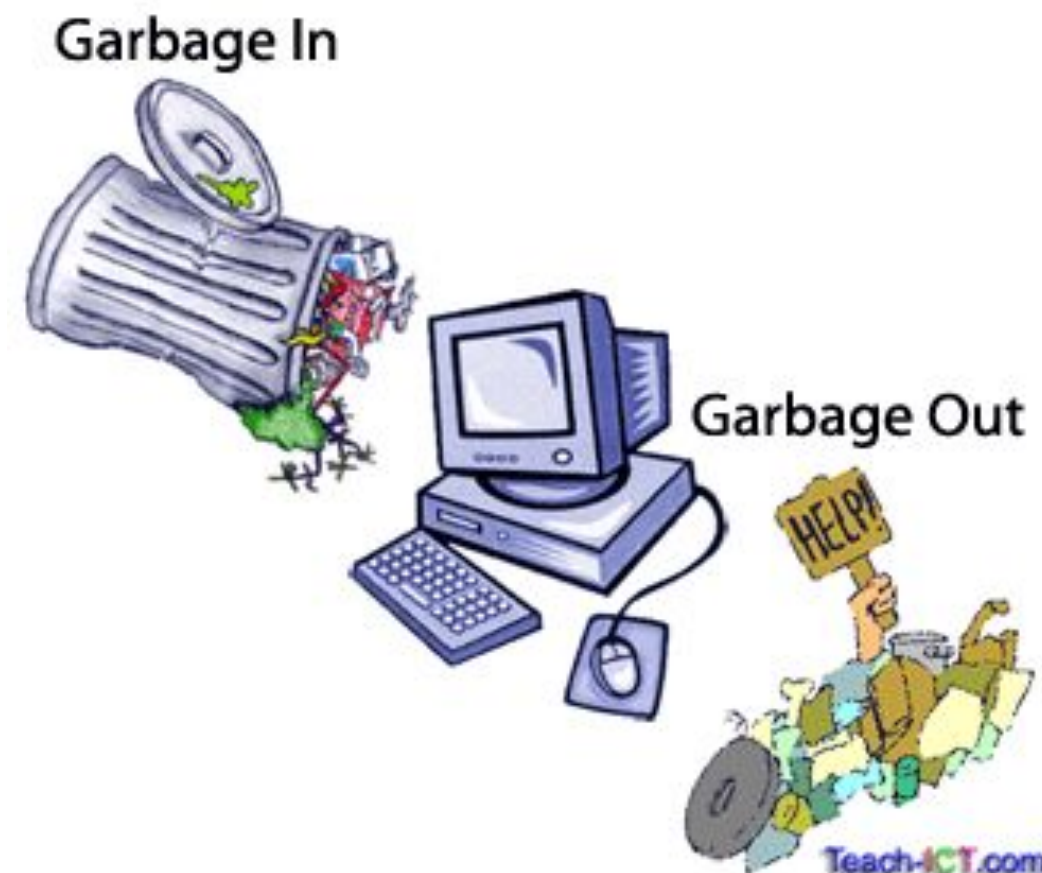
# Dividir el conjunto de datos en entrenamiento (train) y prueba(test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```



Scikit-learn (sklearn) es una librería de Python que facilita la implementación de algoritmos de Machine Learning para construir modelos y analizar datos de forma sencilla.

Obtener datos para un proyecto de Machine Learning

- La adquisición de datos es el proceso de identificar, recopilar y extraer información útil de diversas fuentes para su uso en proyectos de ciencia de datos y aprendizaje automático
- La **calidad**, relevancia y variedad de los datos obtenidos influyen directamente en la efectividad, precisión y desempeño del modelo, haciendo que los datos sean un componente esencial para el éxito del proyecto.



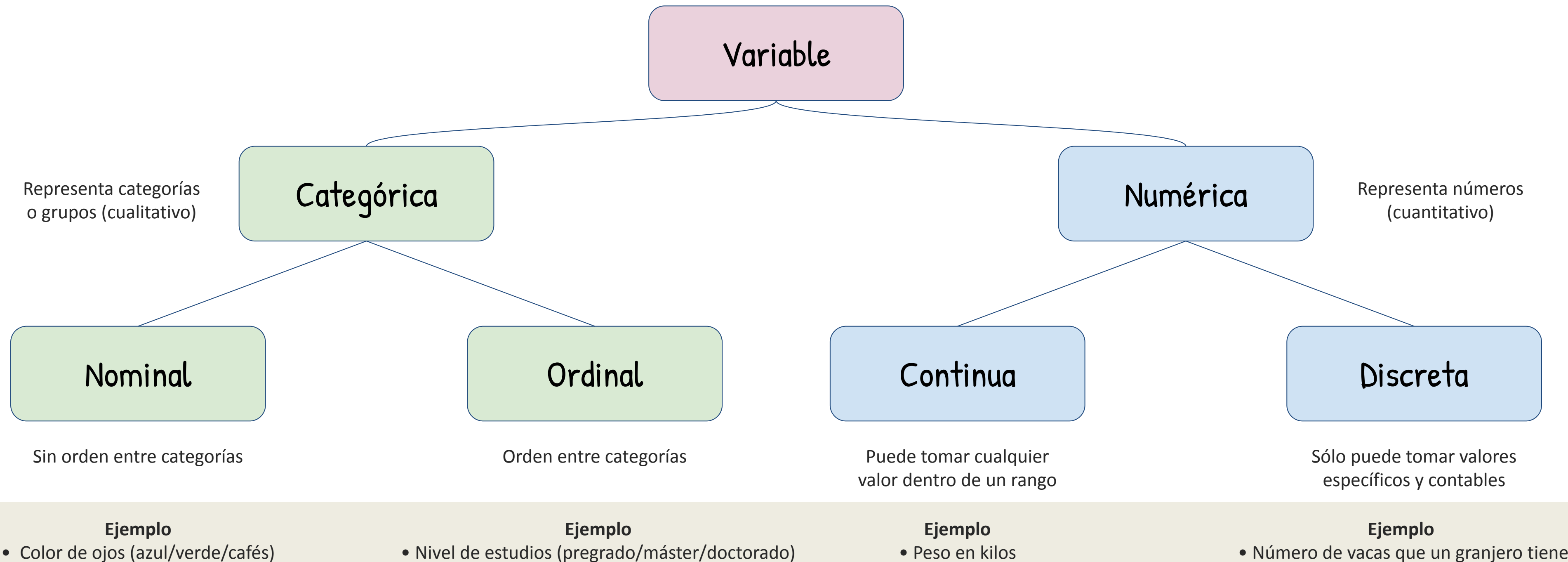


Lo que nosotros vemos

08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 77 91 08 08 02 22 97
49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00 49 49 99 40
81 49 31 73 55 79 14 29 93 71 40 67 53 88 30 03 49 13 36 65 81 49 31 73
52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91 52 70 95 23
22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80 22 31 16 71
24 47 32 60 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50 24 47 32 60
32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70 32 98 81 28
67 26 20 68 02 62 12 20 95 63 94 39 63 08 40 91 66 49 94 21 67 26 20 68
24 55 58 05 66 73 99 26 97 17 78 78 96 83 14 88 34 89 63 72 24 55 58 05
21 36 23 09 75 00 76 44 20 45 35 14 00 61 33 97 34 31 33 95 21 36 23 09
78 17 53 28 22 75 31 67 15 94 03 80 04 62 16 14 09 53 56 92 78 17 53 28
16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57 16 39 05 42
86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58 86 56 00 48
19 80 81 68 05 94 47 69 28 73 92 13 86 52 17 77 04 89 55 40 19 80 81 68
04 52 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66 04 52 08 83
88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 53 69 88 36 68 87
04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36 04 42 16 73
20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16 20 69 36 41
20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 57 05 54 20 73 35 29
01 70 54 71 83 51 54 69 16 92 33 48 61 43 52 01 89 19 67 48 01 70 54 71

Lo que los computadores ven

Una variable puede ser categórica ó numérica



Ingeniería de características (Feature Engineering)

- Es el proceso de tomar los datos crudos y transformarlos en características (features) que un modelo de Machine Learning pueda usar de manera efectiva.
- Las características son las entradas que el modelo utiliza para hacer predicciones, por lo que crear buenas características puede mejorar el rendimiento del modelo significativamente.

In [4]:

```
import seaborn as sns
df = sns.load_dataset('iris')
df.head()
```

Out [4]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

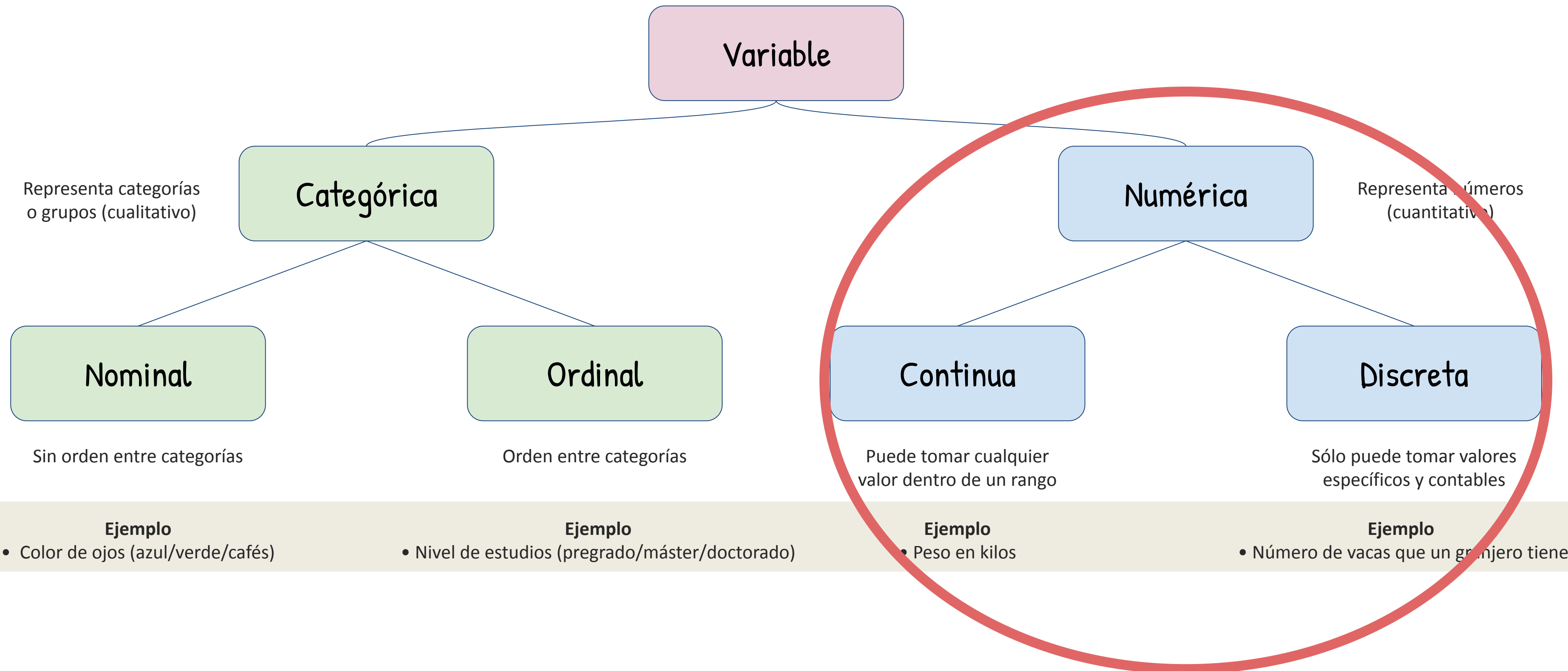
Variable
objetivo

Características

Ingeniería de características (Feature Engineering)

- Ejemplo:
 - Supongamos que tienes una columna que contiene fecha y hora: “06/06/2016:08:30:10”
 - Puedes crear nuevas columnas tal como:
 - Día de la semana (lunes, martes, etc.)
 - Mes del año
 - El año o el trimestre
 - Si es un día laboral o fin de semana
 - Esto es útil porque ayuda al modelo a captar patrones relacionados con el tiempo que no podrían ser evidentes en la fecha cruda

Una variable puede ser categórica ó numérica



Ingeniería de características para variables numéricas

- Con pocas excepciones, los algoritmos de machine learning no funcionan bien cuando los valores numéricos de entrada tienen escalas muy diferentes.
 - Por ejemplo: Al predecir el valor de una casa, número de cuartos y metros cuadrados
- Las técnicas más comunes son:
 - **MinMax**
 - También conocido como normalización.
 - Los valores se desplazan y se reescalan de tal manera que terminan dentro de un rango de 0 a 1.
 - Se calcula restando el valor mínimo y dividiendo por la diferencia entre el valor mínimo y el máximo.
 - **Estandarización**
 - Se calcula restando el valor medio y luego se divide el resultado por la desviación estándar.
 - A diferencia de MinMax scaling, no restringe los valores a un rango específico.

Ejemplo MinMax

Ejemplo: $X = [2, 10, 15]$

$$\text{valor escalado} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

- Para 2:

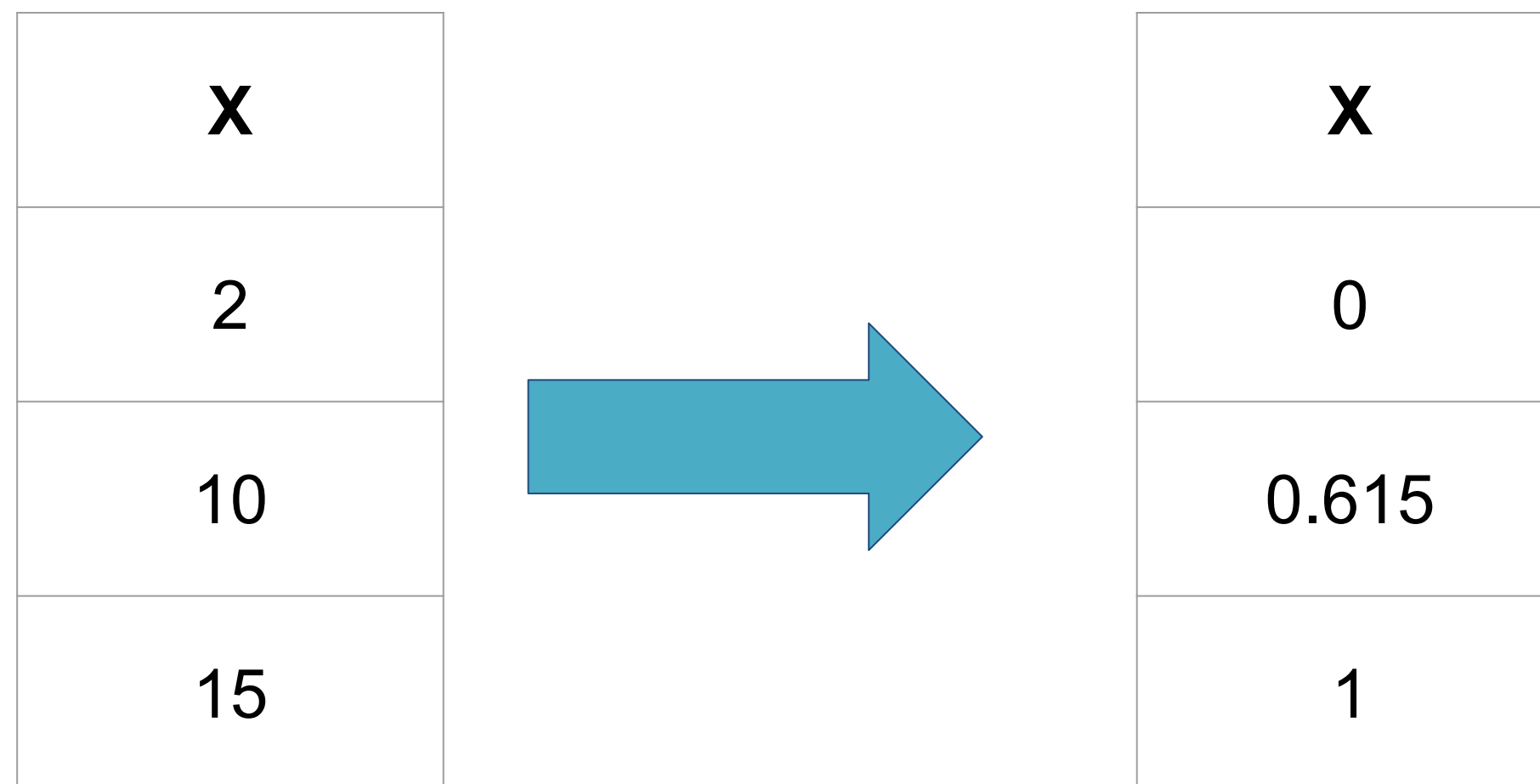
$$\frac{2 - 2}{15 - 2} = \frac{0}{13} = 0$$

- Para 10:

$$\frac{10 - 2}{15 - 2} = \frac{8}{13} \approx 0.615$$

- Para 15:

$$\frac{15 - 2}{15 - 2} = \frac{13}{13} = 1$$



Ejemplo MinMax

Ejemplo: $X = [2, 10, 15]$

$$\text{valor escalado} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

- Para 2:

$$\frac{2 - 2}{15 - 2} = \frac{0}{13} = 0$$

- Para 10:

$$\frac{10 - 2}{15 - 2} = \frac{8}{13} \approx 0.615$$

- Para 15:

$$\frac{15 - 2}{15 - 2} = \frac{13}{13} = 1$$

Ejemplo Estandarización

Ejemplo: X = [2, 10, 15]

valor estandarizado = $\frac{X - \mu}{\sigma}$

Donde:

- μ (media) es el promedio de los valores.
- σ (desviación estándar) mide la dispersión de los datos.

• Para 2:

$$\frac{2 - 9}{5.387} = \frac{-7}{5.387} \approx -1.299$$

• Para 10:

$$\frac{10 - 9}{5.387} = \frac{1}{5.387} \approx 0.186$$

• Para 15:

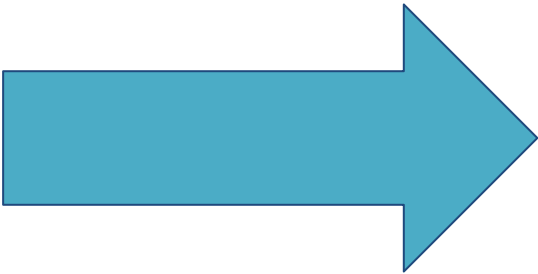
$$\frac{15 - 9}{5.387} = \frac{6}{5.387} \approx 1.114$$

• Media (μ):

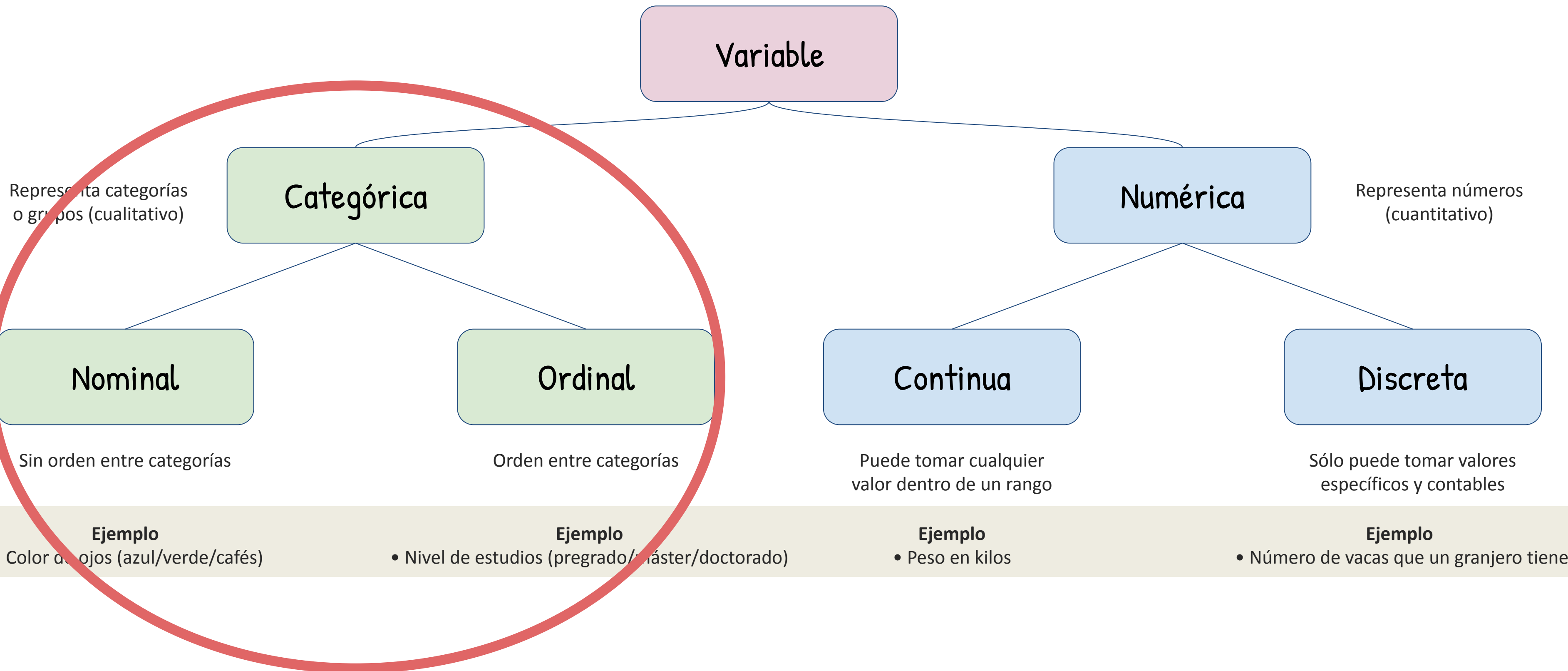
$$\mu = \frac{2 + 10 + 15}{3} = \frac{27}{3} = 9$$

• Desviación estándar (σ):

$$\sigma = \sqrt{\frac{(2 - 9)^2 + (10 - 9)^2 + (15 - 9)^2}{3}} = \sqrt{\frac{49 + 1 + 36}{3}} = \sqrt{\frac{86}{3}} \approx 5.387$$

X		X
2		-1.299
10		0.186
15		1.114

Una variable puede ser categórica ó numérica

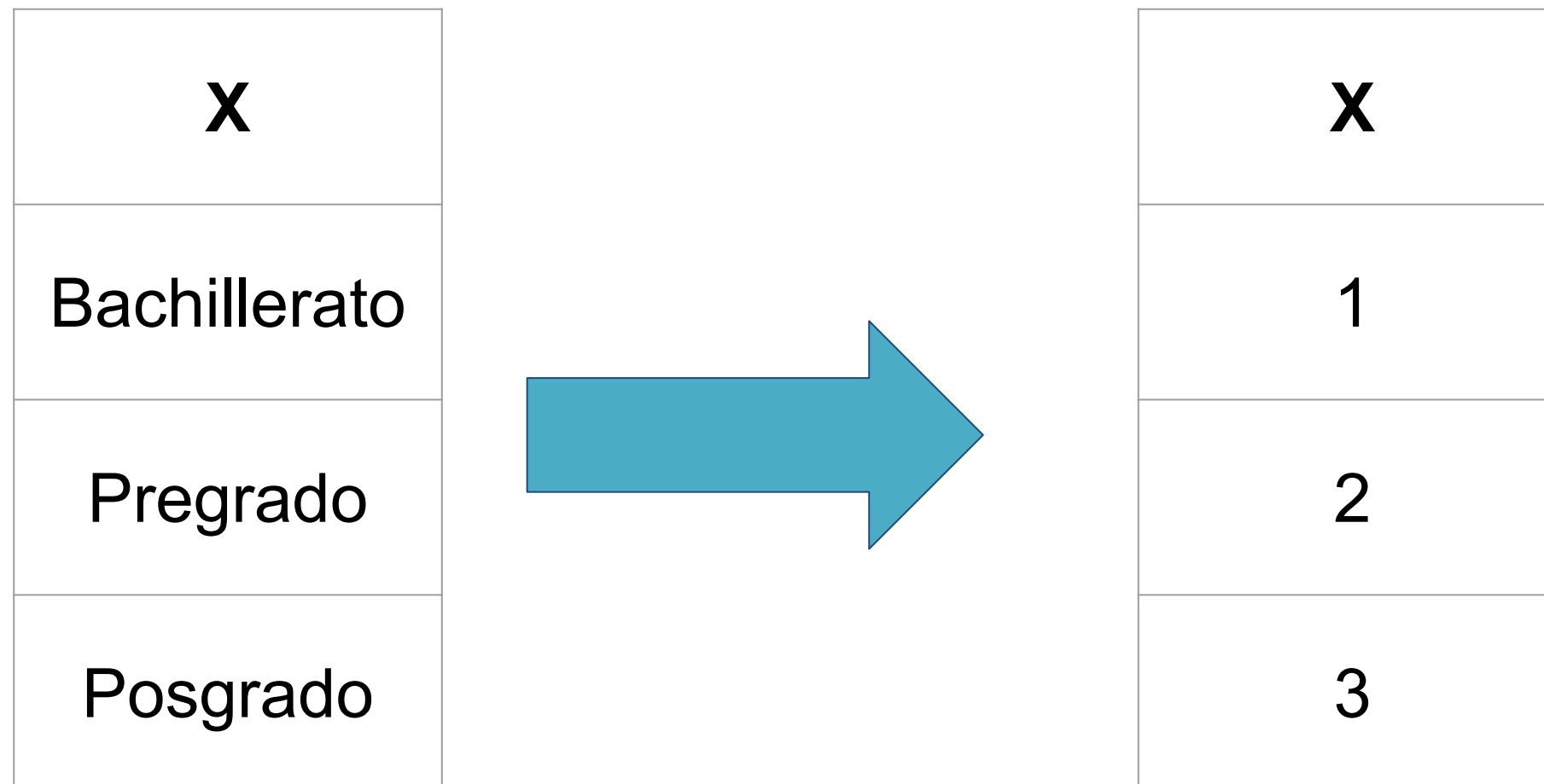


Ingeniería de características para variables categóricas

- El objetivo es convertir variables categóricas en una representación numérica
- Las técnicas más comunes son:
 - **Codificación One-Hot**
 - Ideal para variables nominales
 - Cada columna representa si la variable pertenece o no a esa categoría
 - **Codificación Ordinal**
 - Ideal para variables ordinales
 - El resultado es sólo una columna

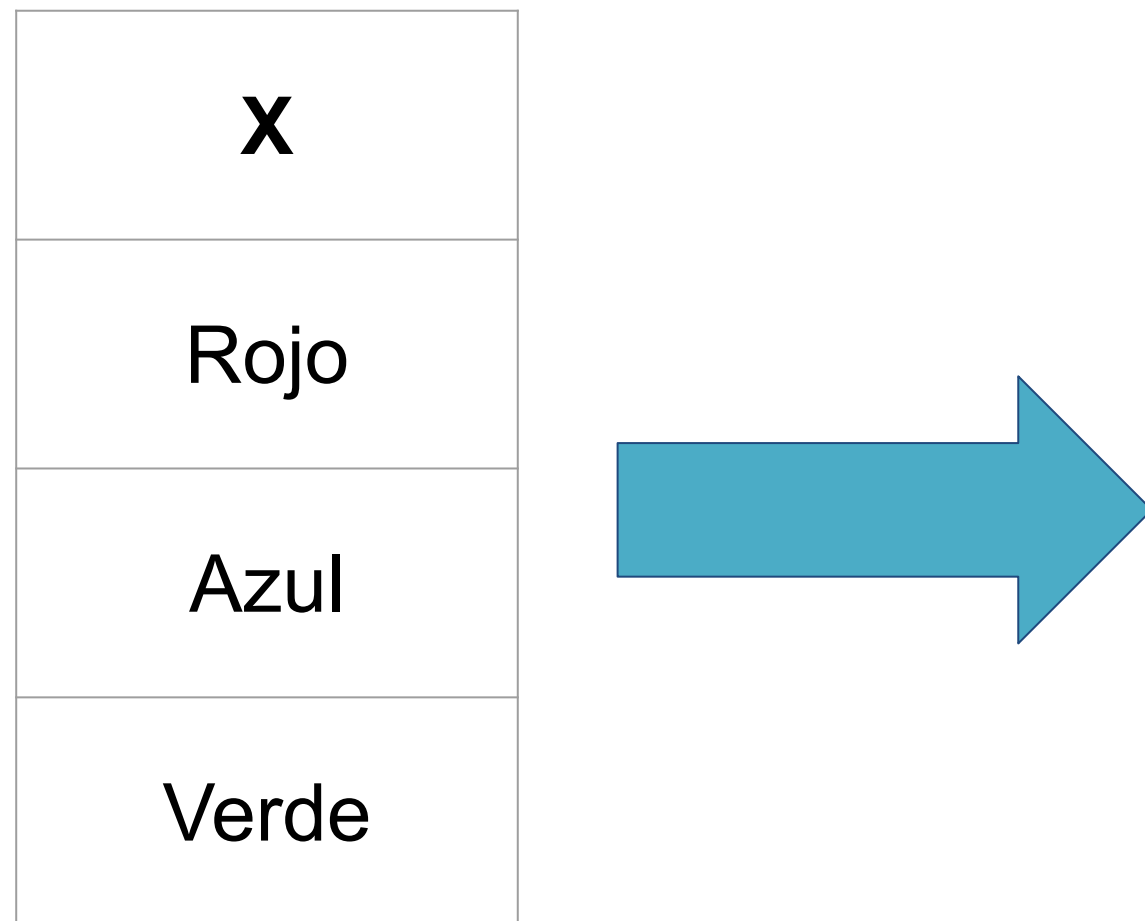
Ejemplo Codificación Ordinal

Ejemplo: $X = [\text{"Bachillerato"}, \text{"Pregrado"}, \text{"Posgrado"}]$



Ejemplo Codificación One-Hot

Ejemplo: $X = [\text{"Rojo"}, \text{"Azul"}, \text{"Verde"}]$



Ejemplo Codificación One-Hot

Ejemplo: $X = [\text{"Rojo"}, \text{"Azul"}, \text{"Verde"}]$

X
Rojo
Azul
Verde



X_rojo	X_azul	X_verde
1	0	0
0	1	0
0	0	1

Datos estructurados vs datos no estructurados

Ventas			
Producto	Potencia	Unidades	Ganancias
Bicicletas	Eléctrica	476	\$751.604
Bicicletas	Manual	302	\$581.350
Motonetas	Eléctrica	387	\$427.248
Motonetas	Manual	309	\$48.513
Patinetas	Eléctrica	251	\$135.791
Bicicletas	Eléctrica	354	\$558.966
Bicicletas	Manual	219	\$336.165
Motonetas	Eléctrica	312	\$583.128
Motonetas	Manual	419	\$396.793

- Los **datos estructurados** están altamente organizados y son fácilmente legibles por máquinas. Normalmente se almacenan en formatos tabulares, como hojas de cálculo (CSV, Excel) o bases de datos relacionales (SQL).

Cada observación está en un fila y sus características en columnas predefinidas, lo que facilita su procesamiento y análisis.



- Los **datos no estructurados** no siguen un formato o estructura específica, lo que los hace más difíciles de organizar y analizar. Este tipo de datos incluye texto libre, imágenes, videos, audios y otros formatos multimedia.

Debido a su naturaleza, los datos no estructurados a menudo requieren técnicas avanzadas, como procesamiento de lenguaje natural (NLP) o redes neuronales convolucionales (CNN).

Ingeniería de características para textos

- La mayoría de modelos trabajan con números, por lo que el texto debe transformarse a un formato numérico para que los modelos puedan procesarlo.
- Para poder capturar el significado de un texto, no es tan fácil como simplemente asignar números. Para esto usamos la codificación de textos.
- Las técnicas más comunes son:
 - TF-IDF
 - Word2Vec
 - BERT
 - GPT

Ingeniería de características para textos

- La mayoría de modelos trabajan con números, por lo que el texto debe transformarse a un formato numérico para que los modelos puedan procesarlo.
- Para poder capturar el significado de un texto, no es tan fácil como simplemente asignar números. Para esto usamos la codificación de textos.
- Las técnicas más comunes son:
 - **TF-IDF**
Frecuencia de términos - Frecuencia inversa de documentos
Resalta palabras importantes en un documento y reduce el peso de palabras comunes (como "y" o "el"), para que los algoritmos se enfoquen en lo relevante.
 - Word2Vec
 - BERT
 - GPT

Procesamiento de Lenguaje Natural

https://www.youtube.com/watch?v=srQTRZzBnoo&list=PLJw3ZK6gs8y2AQu_5AxCE7DDas3rg1y8N

CoSIAM:

<https://cosiam.net/index.php/mmc-cosiam-2024/>

Más ingeniería de características...

- Todo es válido. La mayoría de estas decisiones son guiadas por el conocimiento del sector y tu criterio.

En código... la ingeniería de características se hace DESPUÉS de separar los datos en entrenamiento y prueba

```
from sklearn.model_selection import train_test_split
```

```
# Definir las características (X) y la variable objetivo (y)
```

```
X = df.drop(columns=['Survived']) # Aquí estamos eliminando la columna 'survived', que es el objetivo
```

```
y = df['Survived'] # Esta es nuestra variable objetivo
```

```
# Dividir el conjunto de datos en entrenamiento (train) y prueba(test)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

Métodos comunes en Machine Learning

- **.fit()**: Aprende los parámetros (en el caso del MinMax el mínimo y el máximo) del conjunto de datos que se le pasa. No transforma los datos todavía.
- **.transform()**: Usa los parámetros aprendidos durante el **.fit()** para transformar los datos. No aprende nada nuevo.
- **.fit_transform()**: Combina ambos pasos, primero ajusta el scaler con **.fit()** y luego transforma los datos con **.transform()** en un solo paso.

Ejemplo

$X = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$

- Dividimos este conjunto de datos aleatoriamente en entrenamiento (train) y prueba (test)

$X_{\text{train}} = [1, 2, 3, 6, 7, 8]$

$X_{\text{test}} = [4, 5, 9, 10]$

Ejemplo

$X = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$

- Dividimos este conjunto de datos aleatoriamente en entrenamiento (train) y prueba (test)

`X_train = [1, 2, 3, 6, 7, 8]`

`X_test = [4, 5, 9, 10]`

1. Primer paso: Ajustamos el escalador en el conjunto de entrenamiento (train), donde se calculan el mínimo y el máximo de los valores solo en el conjunto de entrenamiento:

`MinMax.fit(X_train)`

Minimo = 1

Máximo = 8

Ejemplo

$X = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$

- Dividimos este conjunto de datos aleatoriamente en entrenamiento (train) y prueba (test)

$X_{\text{train}} = [2, 3, 4, 6, 7, 8]$

$X_{\text{test}} = [1, 5, 9, 10]$

1. Segundo paso: Aplicamos la transformación en el conjunto de entrenamiento

`MinMax.transform(X_train)`

- Para $X_{\text{train}} = 1$:

$$\frac{1 - 1}{8 - 1} = \frac{0}{7} = 0$$

- Para $X_{\text{train}} = 2$:

$$\frac{2 - 1}{8 - 1} = \frac{1}{7} \approx 0.143$$

- Para $X_{\text{train}} = 3$:

$$\frac{3 - 1}{8 - 1} = \frac{2}{7} \approx 0.286$$

- Para $X_{\text{train}} = 6$:

$$\frac{6 - 1}{8 - 1} = \frac{5}{7} \approx 0.714$$

- Para $X_{\text{train}} = 7$:

$$\frac{7 - 1}{8 - 1} = \frac{6}{7} \approx 0.857$$

- Para $X_{\text{train}} = 8$:

$$\frac{8 - 1}{8 - 1} = \frac{7}{7} = 1$$

Ejemplo

$X = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$

- Dividimos este conjunto de datos aleatoriamente en entrenamiento (train) y prueba (test)

$X_{\text{train}} = [2, 3, 4, 6, 7, 8]$

$X_{\text{test}} = [1, 5, 9, 10]$

1. Segundo paso: Aplicamos la transformación en el conjunto de prueba

`MinMax.transform(X_test)`

- Para $X_{\text{test}} = 4$:

$$\frac{4 - 1}{8 - 1} = \frac{3}{7} \approx 0.429$$

- Para $X_{\text{test}} = 5$:

$$\frac{5 - 1}{8 - 1} = \frac{4}{7} \approx 0.571$$

- Para $X_{\text{test}} = 9$:

$$\frac{9 - 1}{8 - 1} = \frac{8}{7} \approx 1.143$$

- Para $X_{\text{test}} = 10$:

$$\frac{10 - 1}{8 - 1} = \frac{9}{7} \approx 1.286$$



Notebook de hoy

https://colab.research.google.com/drive/1j7V0oLEx_LOBJrKONLSmr5EVZtZxIdax?usp=sharing





Taller # 8

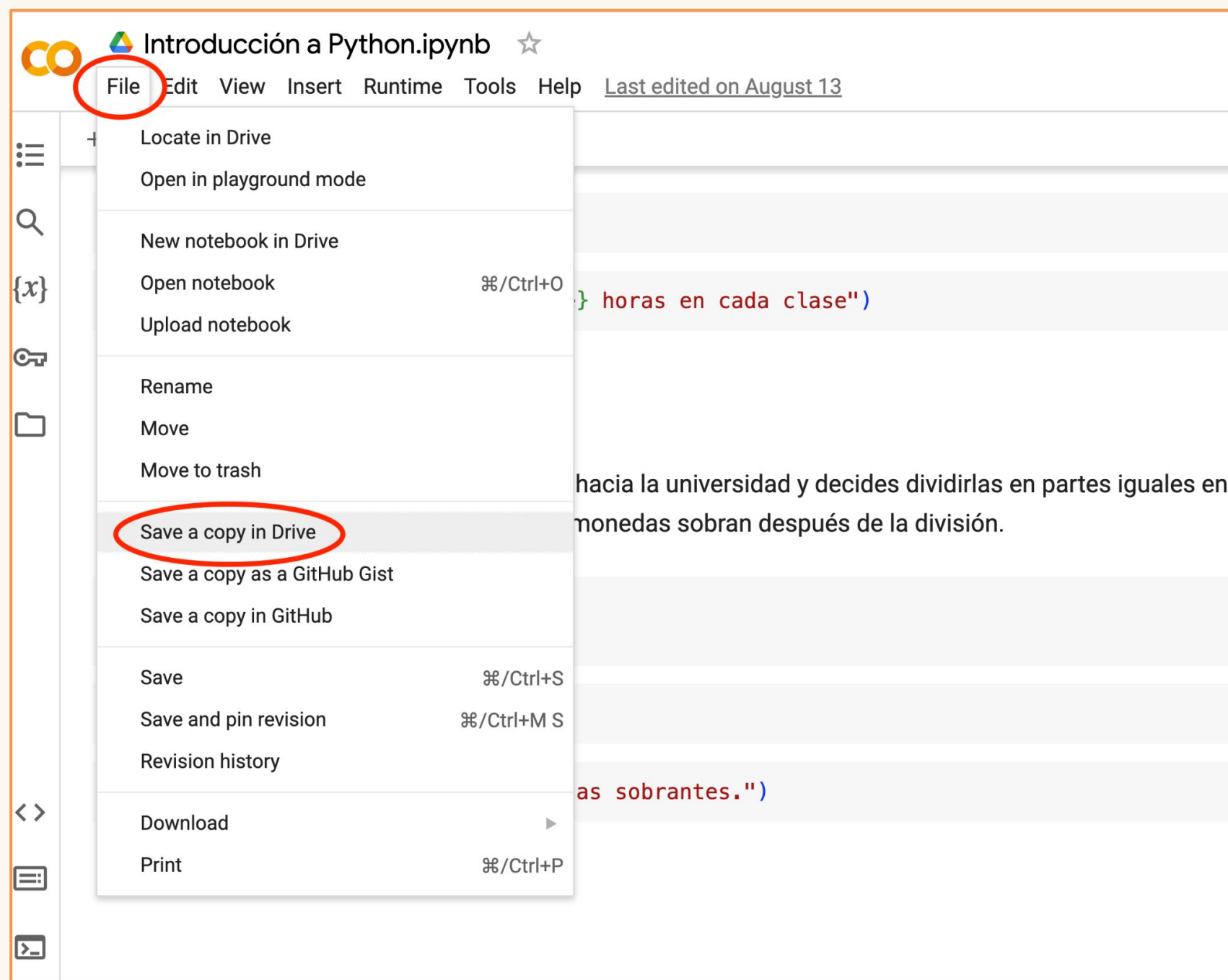
Ingeniería de características

Realizar ingeniería de características a un conjunto de datos de su libre elección
(no olvidar separar en conjunto de datos de entrenamiento y prueba)

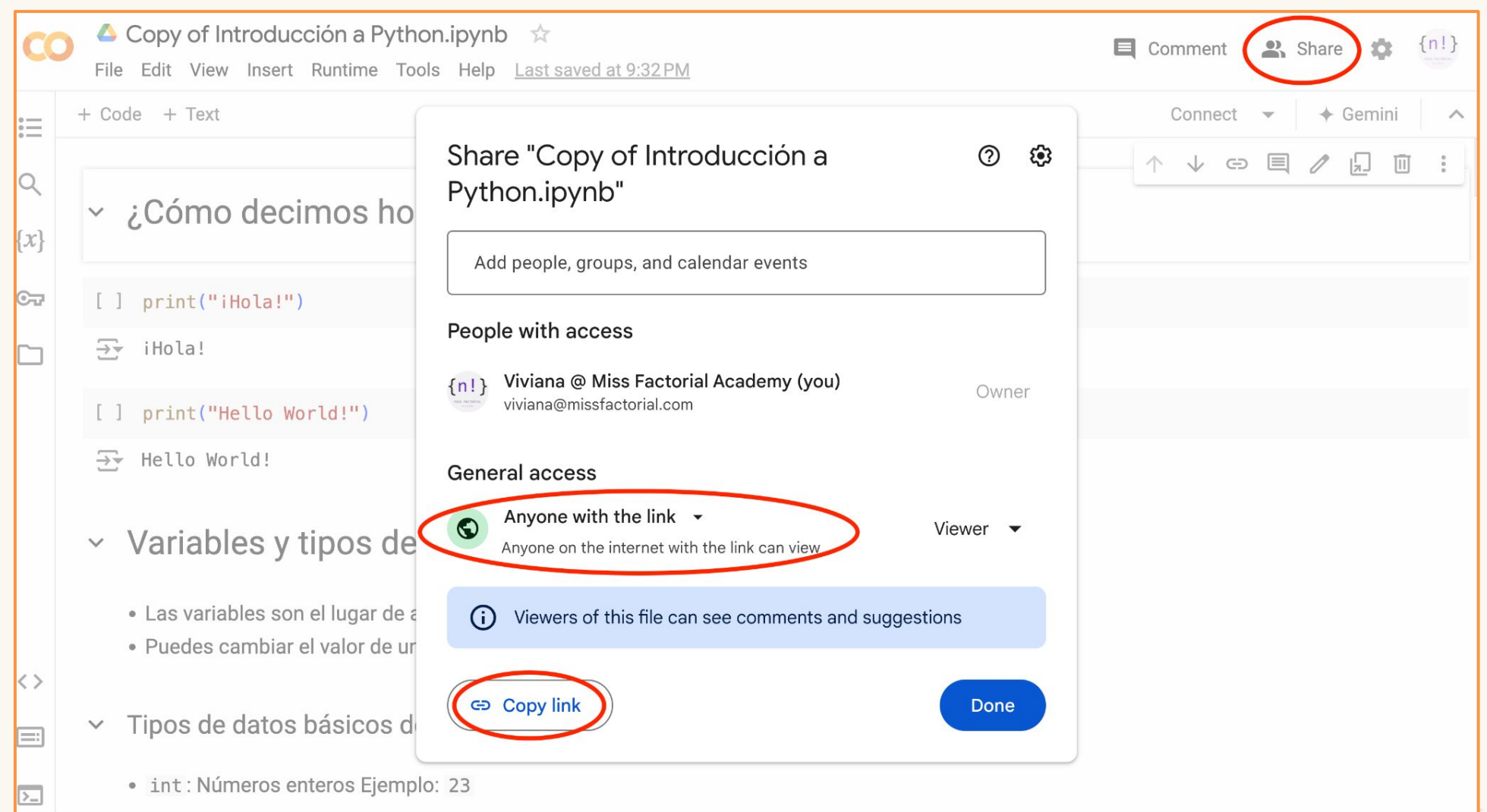
Fecha de entrega: Septiembre 30, 2024

Para enviar los talleres de código

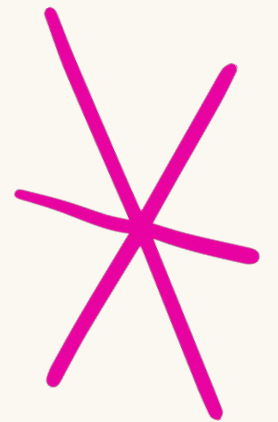
- ❑ Hacer click en **archivo** → **guardar copia en mi Drive** para que les quede una copia en su cuenta, de lo contrario, los resultados no serán guardados.
- ❑ En la copia creada, hacer click en **compartir**, asegurarse que el enlace sea visible a **cualquier persona**, copiar el enlace y enviarlo.



vroberta@unicomfaucauca.edu.co



¡Gracias!



¿Dudas? Email de la profe:

vroberta@unicomfauca.edu.co

Página web del curso con toda la info:

<https://github.com/vivianamarquez/unicomfauca-ai-2024>