



Aprendizaje Automático y Minería de Datos

Minería de datos
Limpieza y calidad

Motivación

- Los datos son los que son, pero todos buscamos **información correcta y completa**
 - La calidad hace referencia a esas *incorrecciones* e *incompletitudes* observadas en la exploración
- Tras haber detectado los problemas, trataremos de reducirlos al máximo mediante un tratamiento de *limpieza*



Calidad

- La calidad del modelo dependerá de la calidad de los datos con lo que lo creamos
 - Principio GIGO (*Garbage In, Garbage Out*)
 - Lo que ofrece el propio Kaggle puede considerarse una primera auditoría de calidad de los datos
- Aspectos de la calidad
 - Completitud
 - Exactitud y validez
 - Consistencia

```
import pandas as pd

# Inspección rápida de la "salud" del dataset
df = pd.read_csv("dataset.csv")

# Resumen general del conjunto de datos
print(df.info())
```

Carga y estructura

- Detectar **errores estructurales de origen**
 - Desalineación entre cabecera y datos
 - Columnas fantasma o fallas al analizar delimitadores
 - Sesgos de recolección
- En Pandas una cosa son los datos, cuya primera posición es la (0, 0), y otra sus etiquetas
 - **index** devuelve las etiquetas de las filas
 - **columns** devuelve las etiquetas de las columnas

```
# Carga usando los valores de la columna 'id_juego' como etiquetas de filas
# y los de la primera fila (header=0) como etiquetas de columnas.
# Si no se indicase nada, Pandas por defecto crea etiquetas numéricas de 0 a N-1.
df = pd.read_csv("dataset.csv", index_col="id_juego", header=0)

# Eliminar ciertas columnas, por inútiles o redundantes.
# errors="ignore" evita que salta un error si alguna de las columnas no existe.
df = df.drop(columns=["uuid_registro", "columna_vacia"], errors="ignore")
```

Tipos de datos, formato y duplicados

- Discrepancia en los tipos

PARSING

- Números *leídos* como cadenas de texto
- Fechas *leídas* como cadenas de texto
- Muestras idénticas o con variaciones tipográficas

```
# 1. Parseo de cadenas y limpieza de texto/símbolos
df["precio"] = df["precio"].astype(str).str.replace("$", "").str.replace(",", "").astype(float)

# 2. Convertir a tipo datetime y extracción de partes temporales (.dt)
df["fecha_lanza"] = pd.to_datetime(df["fecha_lanza"])
df["anio_lanza"] = df["fecha_lanza"].dt.year

# 3. Eliminar duplicados explícitos
df = df.drop_duplicates()
```

Distribución estadística y asimetría

- Recuerda siempre que **correlación no implica causalidad**
- En datos con sesgo masivo o distribuciones de cola larga, la **mediana** y el **rango** INTERQUARTILE RANGE (IQR) **intercuartílico** son más representativos que la **media** y la **desviación típica**
 - Ej. Ventas en catálogos de juegos
- **Spearman** evalúa relaciones monotónicas entre variables (aumentan o disminuyen juntas), siendo útil ante valores atípicos o distribuciones *no* normales frente a **Pearson**

Ejemplo

```
import matplotlib.pyplot as plt

# Resumen estadístico
print(df.describe())

# Mediana e IQR (robusto ante asimetría)
mediana = df["ventas"].median()
iqr = df["ventas"].quantile(0.75) - df["ventas"].quantile(0.25)

# Matriz de correlación de Spearman
correlacion_spearman = df.corr(method="spearman", numeric_only=True)

# Visualización rápida de la asimetría
df["ventas"].hist(bins=30)
plt.title("Distribución de Ventas (Asimétrica)")
plt.xlabel("Ventas")
plt.show()
```

Gestión de valores ausentes

MISSING VALUES

- Tipología de las causas

- Ausencia totalmente “fortuita” o aleatoria MISSING COMPLETELY AT RANDOM (MCAR)
 - Ej. Fallo puntual del micrófono
- Ausencia dependiente de *otra* característica MISSING AT RANDOM (MAR)
 - Ej. Ciertas plataformas no incorporan micrófono
- Ausencia dependiente del propio valor MISSING NOT AT RANDOM (MNAR)
 - Ej. El micrófono no registra nada si susurras

- Estrategias

- Eliminación, solo si el volumen de ausentes es bajo y tenemos datos de sobra
- Imputación, darle un valor razonable
 - Imputación simple: media o mediana (numéricas), moda o una constante (categóricas)

Ejemplo

```
import numpy as np

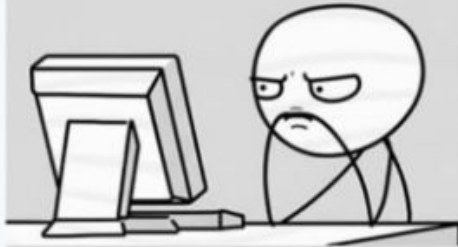
# Detectar nulos y porcentajes por columna
print(df.isnull().mean() * 100)

# Estrategia 1: Eliminar filas con nulos (si son muy pocos)
df_limpio = df.dropna(subset=["variable_critica"])

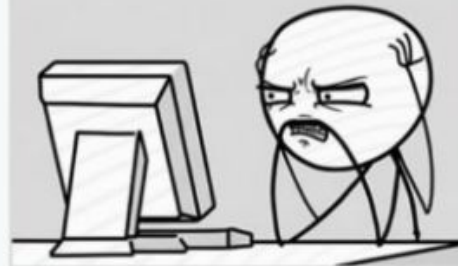
# Estrategia 2: Imputación simple con Pandas
df["edad_mediana"] = df["edad"].fillna(df["edad"].median())
df["categoria_moda"] = df["categoria"].fillna(df["categoria"].mode()[0])
df["categoria_constante"] = df["categoria"].fillna("Desconocido")
```

Gestión de valores ausentes

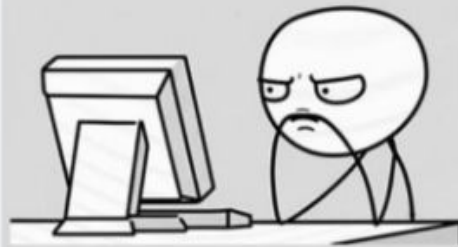
Imports the dataset



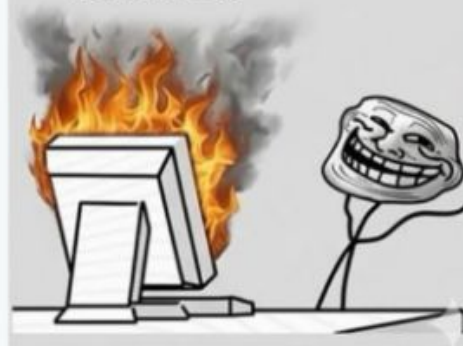
12 missing values found



Cleans the dataset



326 missing values detected



Detección de valores atípicos

- Distinguir entre errores de medición/telemetría (a eliminar/corregir) y valores extremos reales
- **Detección visual**: Uso de Boxplots o diagramas de dispersión para detectarlos
- **Detección basada en el rango intercuartílico (IQR)**: Definir como atípico cualquier valor fuera de un rango como: $[Q1 - 1.5 * IQR, Q3 + 1.5 * IQR]$

Ejemplo

```
# Cálculo del rango IQR para filtrar outliers
Q1 = df["precio"].quantile(0.25)
Q3 = df["precio"].quantile(0.75)
IQR = Q3 - Q1

limite_inferior = Q1 - 1.5 * IQR
limite_superior = Q3 + 1.5 * IQR

# Identificar outliers
outliers = df[(df["precio"] < limite_inferior) | (df["precio"] > limite_superior)]

# Filtrar o recortar (Clipping) dentro del rango permitido
df["precio_limpio"] = df["precio"].clip(lower=limite_inferior, upper=limite_superior)
```

Codificar datos

- Trabajar con variables *categorías* es complicado y conviene codificarlas para trabajar con valores *numéricos*

0 = COLD y 1 = HOT

- **Codificación Una-Activa** Una técnica habitual es crear tantas variables binarias (0 ó 1) como valores tenga la variable categórica, dejando *sólo una activa* cuando se quiere representar dicho valor

ONE-HOT
ENCODING
/ DUMMY
VARIABLES

```
import pandas as pd

# Supongamos que tenemos una serie o columna de Pandas
df = pd.DataFrame({'provincias': ['Madrid', 'Barcelona', 'Sevilla', 'Madrid']})

# Codificación One-Hot con Pandas
provincias_encoded = pd.get_dummies(df['provincias'], drop_first=True, dtype=int)

print(provincias_encoded)
```

```
# Aplicar get_dummies indicando qué columna transformar
df_transformado = pd.get_dummies(df, columns=['provincias'], drop_first=True, dtype=int)

print(df_transformado)
```

Codificar datos

- La codificación Una-Activa también puede hacerse con la herramienta **SciKit-Learn**
- Ej. Codificando provincias
 - Ej. Tenemos X provincias y vamos a crear X-1 variables binarias (la primera provincia se representa con todas las variables inactivas)
 - Se las pasamos al codificador en forma de matriz con tantas filas como hagan falta y sólo 1 columna, y este “hace su magia”, cuyo resultado podemos ver en array

Escalado de características

FEATURE SCALING

- Todos los datos tienen que estar en la misma escala, especialmente para luego trabajar con ciertos algoritmos de AA
 - Escalado Puntuación Z (Estandarización) Z-SCORE SCALING / STANDARDIZATION
 - Escalado Min-Max (Normalización) MIN-MAX SCALING / NORMALIZATION
 - Valores entre 0 y 1
 - Normalización vectorial VECTOR NORMALIZATION
- Esta transformación se calcula *sólo para los datos de entrenamiento* y luego se aplica “a ciegas” sobre los datos de validación y prueba final

Ejemplo

Training set: $x^{(1)}, x^{(2)}, \dots, x^{(m)}$

$$x' = \frac{x - \mu}{\sigma}$$

$\mu_j = \frac{1}{m} \sum_{i=1}^m x_j^{(i)}$
Replace each $x_j^{(i)}$ with $x_j - \mu_j$

$$x_j^{(i)} \leftarrow \frac{x_j^{(i)} - \mu_j}{s_j}$$

1. Estandarización (Z-Score)

```
media = df["edad"].mean()  
desviacion = df["edad"].std()  
df["edad_zscore"] = (df["edad"] - media) / desviacion
```

2. Normalización Min-Max

```
v_min = df["precio"].min()  
v_max = df["precio"].max()  
df["precio_minmax"] = (df["precio"] - v_min) / (v_max - v_min)
```

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

Participación

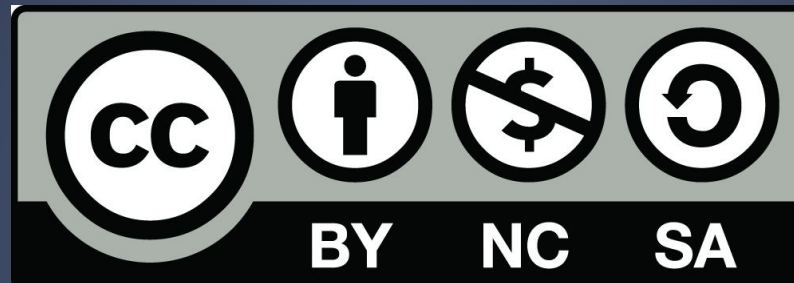
- ¿Qué hacer si tengo **valores atípicos** a la derecha en una variable numérica?
 - A. Imputarlos con la moda
 - B. Imputarlos con la media
 - C. Imputarlos con la mediana
 - D. Ignorarlos por estar a la derecha
 - E. Sustituirlos por el equivalente en variable categórica



Problemas habituales en la práctica

- **Errores estructurales**, cabecera y datos no tienen las mismas columnas o no siguen el mismo orden
- **Tipos inadecuados**, fechas, rangos o colecciones expresadas con cadenas de texto, y múltiples valores en un sólo campo
- **Valores ausentes sin codificar**, poniendo 0's o cadenas de texto vacías
- **Características cuasi-constantes**, llenas de nulos o de un mismo valor categórico
- **Sesgo extremo**, suele dominar el 0
- **Demasiadas variables cardinales únicas**

Críticas, dudas, sugerencias...



* Excepto el contenido multimedia de terceros autores

Federico Peinado (2026)

www.federicopeinado.es

