



CAPÍTULO II

TIPOS DE DATOS E INGENIERÍA DE CARACTERÍSTICAS

Después del Capítulo 1, en el cual sentamos las bases de este libro, empezamos ahora a abordar los tipos de datos que lo componen. Sin embargo, antes de entrar en los detalles propios de los datos estructurados, de series de tiempo y de audio sintético, es necesario revisar algunos conceptos transversales que acompañan cualquier trabajo investigativo en ciencia de datos.

En este capítulo estudiaremos cómo se clasifican los datos según su forma (estructurados, semiestructurados y no estructurados) y según la escala en la que se encuentran (nominal, ordinal, de intervalo o de razón), pues estas decisiones determinan qué operaciones son válidas y qué transformaciones son pertinentes. Presentaremos también los fundamentos de la ingeniería de características: mejorar variables existentes, construir nuevas, seleccionar las más relevantes, extraer representaciones y permitir que ciertos modelos aprendan las suyas propias.

Estas ideas serán el punto de partida para los análisis, retos y proyectos que desarrollaremos a lo largo del libro. Comprender la naturaleza del dato no es un paso accesorio: es la base para diseñar experimentos más claros, tomar decisiones mejor informadas y avanzar con rigor en cada uno de los tres universos de datos que exploraremos.

2.1. Tipos de datos según su estructura

Cuando iniciamos un proyecto de ciencia de datos, una de las primeras preguntas que debemos responder es: **¿qué forma tienen los datos con los que vamos a trabajar?** La estructura del dato condiciona la forma en que podemos explorarlo, las transformaciones que serán posibles y el tipo de modelos que podrán utilizarse con mayor efectividad. Por eso, antes de cualquier análisis, es fundamental reconocer en qué categoría se encuentra la información disponible.

2.1.1. Datos estructurados

Los datos estructurados son, en general, los más sencillos de manipular, porque están organizados en formato tabular: **filas** que representan observaciones y **columnas** que corresponden a variables o características. En muchos casos, una de estas columnas corresponde a la **variable objetivo o salida**, es decir, aquella que deseamos predecir o explicar.

Este tipo de datos requiere poco espacio de almacenamiento y se encuentra con frecuencia en archivos .csv, bases de datos relacionales o sistemas transaccionales.

Para ilustrarlo, supongamos que contamos con información sobre los *precios de viviendas* de un barrio, tomada de anuncios publicados en un buscador de inmuebles. De cada anuncio podemos extraer variables como:

- tamaño del apartamento (m²),
- antigüedad del edificio,
- número de habitaciones y baños,
- piso en el que se ubica,
- cantidad de parqueaderos,
- presencia de depósito, terraza o gimnasio, entre otros.

Cada una de estas características se convierte en una columna del dataset, y serán las entradas de un modelo de predicción. La columna del *precio* será la salida a estimar. Con datos históricos, un modelo de regresión puede

aprender patrones y predecir el valor de otros inmuebles del mismo sector, útil tanto para vendedores que desean fijar un precio justo como para compradores que quieren evaluar si un apartamento está sobrevalorado, subvalorado o dentro del rango esperado.

**Tabla 2. Ejemplo de dataset estructurado:
precio de vivienda (en millones de pesos).**

M2	Años	Hab.	Baños	Piso	Parq.	Depósito	Terraza	Gimnasio	Precio
100	2	3	3	8	1	0	1	1	500
60	2	2	2	6	1	0	1	1	320
80	2	2	3	7	1	0	1	1	400
100	15	3	3	4	1	0	1	0	400
...	...	...	...	...	...	...	...	...	...

Con este tipo de datos, un modelo podría identificar, por ejemplo, que la antigüedad del inmueble tiene un impacto mayor en el precio que el número de habitaciones, o que un apartamento pequeño pero nuevo puede tener un valor similar a uno más grande pero más antiguo.

En este libro, trabajaremos con dos grandes categorías de datos estructurados:

- datos tabulares básicos, y
- series de tiempo, una forma especial de datos estructurados donde el orden temporal es esencial.

El primer tipo se aborda en el Capítulo 3, mientras que el segundo se desarrolla en el Capítulo 4.

2.1.2. Datos semiestructurados

Los datos semiestructurados no siguen un formato tabular rígido, pero sí cuentan con una organización interna que facilita su interpretación. Suelen venir en formatos como *JSON*, *XML* o en archivos de *logs*, donde la información se representa mediante pares clave vs. valor, jerarquías o etiquetas.

Aunque no pueden analizarse directamente como un dataset tabular, pueden transformarse mediante procesos de limpieza e *ingeniería de características*, o convertirse en tablas para análisis posteriores. En investigación es común enriquecer estos datos para integrarlos en pipelines más complejos.

Ejemplos frecuentes:

- respuestas de APIs en JSON,
- datos provenientes de sistemas web,
- archivos XML con metadatos,

2.1.3. Datos no estructurados

Los datos no estructurados carecen de un formato predefinido o un esquema fijo. Aquí encontramos información almacenada en formatos libres: texto, imágenes, audio o video. Para extraer valor de estos datos se requieren técnicas especializadas como procesamiento de lenguaje natural, visión por computador o análisis de audio.

En este libro trabajaremos especialmente con un subtipo muy particular: audio sintético, un dato no estructurado que, al transformarse en representaciones espectrales (como Mel, log-Mel o CQT), puede analizarse mediante métodos clásicos y modelos profundos.

Ejemplos comunes de datos no estructurados:

- documentos de texto,
- fotografías y videos,
- grabaciones de voz,
- comentarios en redes sociales.

La Figura 3 resumen las diferencias entre datos estructurados y datos no estructurados.



Figura 3. Figura comparativa datos estructurados vs. datos no estructurados.

2.2. Niveles de los datos

Una vez definidos los tipos de datos, vamos ahora a revisar los niveles de los datos, que típicamente encontramos en datos estructurados y semiestructurados. Estos son: nivel nominal, nivel ordinal, nivel intervalo, y escala de razón.

2.2.1. Nivel Nominal

Corresponde a variables o *features* que no poseen ningún tipo de jerarquía entre sí. Supongamos, por ejemplo, que contamos con la información de las localidades de una ciudad como Bogotá (Colombia). En ese caso, podríamos encontrar nombres como Usaquén, Chapinero, Barrios Unidos y Teusaquillo.

Cada una de estas localidades debe transformarse en un valor numérico que pueda ser interpretado por un modelo de aprendizaje de máquina durante una tarea de predicción.

Para este tipo de variables, la transformación más adecuada es *one-hot encoding*, en la cual se crean tantas columnas nuevas como categorías tenga la variable. En nuestro ejemplo de cuatro localidades, se generarían cuatro columnas, y para cada registro solo se activaría una de ellas, así:

LOCALIDAD		USAQUÉN	CHAPINERO	BARRIOS UNIDOS	TEUSAQUILLO
USAQUÉN	One-hot-encoding →	1	0	0	0
CHAPINERO		0	1	0	0
BARRIOS UNIDOS		0	0	1	0
TEUSAQUILLO		0	0	0	1

Figura 4. Ejemplo transformación de datos tipo nominal

La ventaja de esta transformación es que evita introducir sesgos derivados de asignar números consecutivos a cada categoría (por ejemplo, Usaqué = 1, Chapinero = 2, etc.), lo cual podría interpretarse erróneamente como un orden o una distancia entre ellas. En cambio, con *one-hot encoding*, cada categoría existe únicamente en su propia columna.

No obstante, cuando una variable tiene un número muy elevado de categorías, esta transformación puede incrementar de manera significativa la cantidad de columnas del dataset, aumentando la complejidad del modelo.

2.2.2. Nivel Ordinal

A diferencia del nivel anterior, en este nivel sí existe un sentido de jerarquía. Se utiliza, por ejemplo, cuando estamos asignando una valoración cualitativa al nivel de satisfacción de un servicio.

Supongamos que queremos calificar qué tan satisfechos estamos con la clase de la electiva de Data Engineering. Las opciones son: muy contento, contento, neutral, insatisfecho, muy insatisfecho. En este caso, es evidente que “muy contento” es mejor que “muy insatisfecho”, por lo que podemos asignar una escala cuantitativa con jerarquía, así:

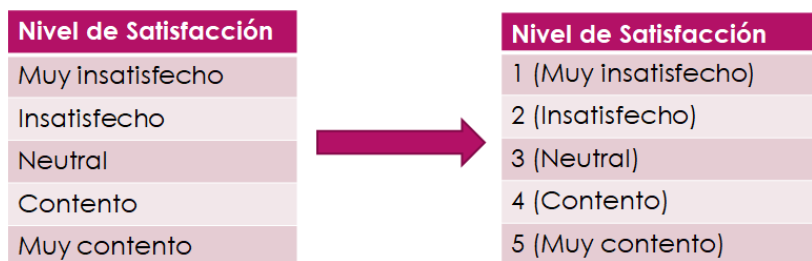


Figura 5. Ejemplo transformación de datos tipo ordinal

2.2.3. Nivel Intervalo

En este nivel existe un sentido claro de orden, y los valores están expresados en una escala numérica continua (enteros o flotantes). Sin embargo, el cero no representa ausencia absoluta de la magnitud medida.

Un ejemplo típico es la temperatura. Sabemos que en determinada ciudad los valores pueden oscilar entre 10 °C y 22 °C. En este tipo de variables es posible calcular promedios, medianas, mínimos, máximos o desviaciones estándar sin problema.

2.2.4. Escala de razón

A diferencia del nivel intervalo, en este caso sí existe un cero absoluto real, lo que permite interpretar las proporciones de manera significativa. Ejemplos de este tipo incluyen: moneda, edad, altura y peso.

Esto permite afirmaciones como:

- una persona pesa el doble que otra,
- un adulto mide el doble que un niño, o
- alguien dispone de la tercera parte del presupuesto de otra persona.

Cuando los datos pertenecen a esta categoría, es válido aplicar operaciones como la media aritmética, geométrica o armónica, pues estas respetan la proporcionalidad de las magnitudes.

Supongamos que contamos con dos columnas: Satisfacción del usuario (CSAT) y Índice de promotores (NPS). La primera se mide en una escala de 0–5 y la segunda en una escala de 0–10. Nuestro objetivo es construir una nueva columna que represente la calificación global de cada departamento.

Tabla 3. Ejemplo de columnas tipo escala de razón.

Departamento	Satisfacción Usuario (CSAT)	Índice de promotores (NPS)
A	4.0	7.5
B	3.0	9.0

Media aritmética

$$A = (4.0 + 7.5) / 2 = 5.75$$

$$B = (3.0 + 9.0) / 2 = 6.00$$

Media geométrica

$$A = \sqrt{(4.0 \times 7.5)} = \sqrt{30} \approx 5.48$$

$$B = \sqrt{(3.0 \times 9.0)} = \sqrt{27} \approx 5.20$$

Media armónica

$$A = 2 / (1/4.0 + 1/7.5) = 2 / 0.3833 \approx 5.22$$

$$B = 2 / (1/3.0 + 1/9.0) = 2 / 0.4444 \approx 4.50$$

Con la media aritmética, la calificación más alta la obtiene el Departamento B. Sin embargo, con la media geométrica y la media armónica, la nota más alta corresponde al Departamento A. La mayor diferencia se observa con la media armónica, ya que esta penaliza los casos en los que los valores presentan una mayor dispersión entre sí.

2.3. Ingeniería de Características

La ingeniería de características (*Feature Engineering*, FE) es una de las etapas más determinantes en un proyecto de ciencia de datos. Incluso el mejor modelo no puede compensar datos mal representados. Por eso, antes de entrenar cualquier algoritmo, debemos transformar, depurar y reformular las variables para que capturen mejor los patrones relevantes en los datos.

En esta sección revisamos cinco enfoques fundamentales: mejoramiento, construcción, selección, extracción y aprendizaje automático de características. Cada técnica tiene un propósito distinto, pero todas buscan lo mismo: mejorar la calidad de la representación de los datos.



Figura 6. Consolidados tipos de Feature Engineering.

A continuación, se explicará cada una de ellas.

2.3.1. Mejoramiento de Características (Feature Improvement)

El mejoramiento consiste en optimizar las variables existentes para que sean más estables, interpretables y útiles.

Tareas típicas:

- Imputación de valores faltantes, usando:
 - Moda para variables nominales/ordinales.
 - Mediana para datos numéricos asimétricos.
 - Media, en distribuciones simétricas.
- Estandarización, llevando los datos a media 0 y desviación estándar 1.
- Normalización, ajustando valores a un rango como [0, 1].
- Tratamiento de *outliers*, por winsorización o transformación logarítmica.

Para ilustrar esta técnica de FE, supongamos que contamos con un dataset de precios de vivienda acompañado del área en metros cuadrados. En una de las filas, falta el valor del área. En términos generales, tenemos dos opciones: eliminar la fila con datos faltantes o imputar el valor con una medida representativa, como la mediana.

Partimos del siguiente ejemplo:

Tabla 4. Ejemplo precio de vivienda, antes de imputación de datos.

Área	Precio
60	350
NaN	420
48	395

El valor faltante del área puede imputarse con la mediana (54 m²), quedando así:

Tabla 5. Ejemplo precio de vivienda, después de imputación de datos.

Área	Precio
60	350
54	420
48	395

Esto no crea nuevas variables, pero mejora la calidad de las ya existentes.

2.3.2. Construcción de Características (Feature Construction)

La construcción de características consiste en crear nuevas variables a partir de la información disponible, o integrar datos externos que aumenten la capacidad predictiva del modelo.

Estrategias comunes:

- Combinar variables: razones, diferencias, promedios, proporciones.
- Codificar variables categóricas (one-hot encoding, ordinal encoding, etc.).
- Cruzar características (interacciones entre variables).
- Integrar nuevas fuentes de datos (datos geográficos, climatológicos, demográficos, etc.).

Recordando el nivel nominal, visto en la sección anterior, podemos crear nuevas características a partir de una variable de tipo nominal, con una columna por cada opción tenga esa característica. Así, tendremos tantas nuevas columnas como opciones tiene esa característica.

2.3.3. Selección de Características

En *datasets* con gran cantidad de variables, no todas aportan información relevante. Algunas incluso pueden introducir ruido o distorsiones en el modelo. Por ejemplo, en el caso de predicción del precio de un apartamento, podríamos tener las siguientes características

Área, número habitaciones, número baños, total parqueaderos, total depósito, gimnasio, terraza, piso dentro del edificio, color de las paredes, tipo de baldosa, tipo de cortinas, ...

Es evidente que variables como **el color de las paredes** no son determinantes para predecir el precio del inmueble. Dos apartamentos con colores similares pueden tener precios completamente distintos debido a

características estructurales. Por lo tanto, esta variable debería eliminarse antes del modelamiento.

Dentro de las motivaciones principales para realizar la selección de características, es:

- Evitar la maldición de la dimensionalidad.
- Eliminar características redundantes o altamente correlacionadas.
- Aumentar la interpretabilidad del modelo.
- Mejorar el rendimiento y la estabilidad del modelo.

Técnicas frecuentes:

- Filtro: pruebas estadísticas, correlaciones, chi-cuadrado.
- Wrapper: RFE (eliminación recursiva).
- Embedded: árboles de decisión, regularización L1 (Lasso).

2.3.4. Extracción de Características

La extracción aplica técnicas matemáticas o estadísticas para transformar un conjunto de variables en otro más informativo o más compacto. Aquí no creamos variables manualmente; las derivamos mediante transformaciones.

Ejemplos de técnicas:

- PCA (reduce dimensionalidad identificando componentes principales).
- SVD (útil en sistemas de recomendación).
- TF-IDF o Bag-of-Words para texto.
- Espectrogramas para audio.

Este último caso será utilizado en el capítulo final del libro, donde transformaremos señales de audio en representaciones visuales de su energía espectral.

2.3.5. Aprendizaje Automático de Características

En esta técnica, un modelo, generalmente redes profundas, aprende de manera automática las representaciones internas o latentes de los datos, sin necesidad de especificarlas explícitamente. Es fundamental en el procesamiento de datos no estructurados.

Modelos frecuentes:

- Autoencoders → crean representaciones comprimidas.
- CNNs → extraen patrones espaciales (imágenes).
- Transformers → extraen relaciones contextuales (texto).
- GANs → aprenden distribuciones complejas.

Retomando el proyecto *FakeVoiceFinder*, dentro de la CNN las capas convolucionales y de *pooling* aprenden patrones relevantes sin ser instruidas explícitamente. Estas capas detectan irregularidades de timbre, texturas acústicas y artefactos característicos de voces sintéticas. Con estas representaciones, el bloque final del modelo puede discernir si un audio es natural o generado artificialmente.

