Fundamentos del Análisis Programático de Datos

Pensamiento algorítmico y exploración inicial

26 de agosto de 2025

Objetivos de la clase

- Comprender por qué programar para analizar datos
- Conocer los principios de datos ordenados (*tidy data*)
- Desarrollar pensamiento algorítmico y pseudocódigos
- Identificar las etapas iniciales del análisis de datos

¿Por qué programar para analizar datos?

Limitaciones de las herramientas tradicionales

- **Reproducibilidad**: Imposible documentar cada click en Excel
- **Escalabilidad**: Manejo limitado de grandes volúmenes
- **Automatización**: Procesos manuales propensos a errores
- **Colaboración**: Dificultad para compartir y versionar

Ventajas de la programación

- **Documentación completa** del proceso analítico
- **Reproducibilidad** total de los resultados
- **Automatización** de reportes y análisis recurrentes
- **Control de versiones** y trabajo colaborativo
- **Flexibilidad** para manejar datos complejos

Casos de uso en la práctica

Reportes automáticos

- Dashboard mensual de ventas
- Informes regulatorios
- Análisis de performance

Análisis complejos

- Modelos estadísticos avanzados
- Machine learning
- Simulaciones

Procesamiento a gran escala

- Millones de transacciones
- Múltiples fuentes de datos
- Análisis en tiempo real

Integración de sistemas

- APIs y bases de datos
- Pipelines de datos
- ETL automatizados

Datos ordenados (*Tidy Data*)

Principios de datos ordenados

Los tres principios fundamentales:

1. **Cada variable** forma una columna
2. **Cada observación** forma una fila
3. **Cada valor** ocupa una celda

¿Por qué importa?

- Facilita el análisis y la visualización
- Permite usar herramientas estándar de manera consistente
- Reduce la complejidad del código
- Mejora la comprensión de los datos

Ejemplo: datos desordenados

Empresa	Q1_2024	Q2_2024	Q3_2024	Q4_2024
ABC	100	120	110	130
XYZ	80	90	95	100

Problemas:

- Los trimestres están en columnas (deberían ser valores)
- Dificulta calcular promedios por trimestre
- Complica la graficación temporal

Ejemplo: datos ordenados

Empresa	Trimestre	Ventas
ABC	Q1_2024	100
ABC	Q2_2024	120
ABC	Q3_2024	110
ABC	Q4_2024	130
XYZ	Q1_2024	80
XYZ	Q2_2024	90
XYZ	Q3_2024	95
XYZ	Q4_2024	100

Ventajas:

- Cada variable en su columna
- Fácil agrupación y análisis
- Compatible con herramientas de visualización

Ejercicio: ¿cómo ordenarías estos datos?

Datos de empleados por sucursal

Empleado	Sucursal_Norte_Ventas	Sucursal_Norte_Clientes	Sucursal_Sur_Ventas	Sucursal_Sur_Clientes
García	45000	120	0	0
López	0	0	38000	95
Martín	52000	140	28000	75
Silva	0	0	41000	88

Pregunta

¿Qué problemas identificás en esta estructura? ¿Cómo la transformarías para que sea *tidy*?

Respuesta: datos ordenados

Empleado	Sucursal	Metrica	Valor
García	Norte	Ventas	45000
García	Norte	Clientes	120
López	Sur	Ventas	38000
López	Sur	Clientes	95
Martín	Norte	Ventas	52000
Martín	Norte	Clientes	140
Martín	Sur	Ventas	28000
Martín	Sur	Clientes	75
Silva	Sur	Ventas	41000
Silva	Sur	Clientes	88

Cambios realizados:

- **Sucursal** pasó de estar en nombres de columnas a ser una variable
- **Métrica** (Ventas/Clientes) también se convirtió en variable
- Se eliminaron los ceros (observaciones inexistentes)
- Ahora cada fila es una observación única

Convenciones de nomenclatura tidy

Estilo snake_case (recomendado)

Principios:

- Todo en minúsculas
- Palabras separadas por guión bajo (_)
- Nombres descriptivos y claros
- Sin espacios ni caracteres especiales

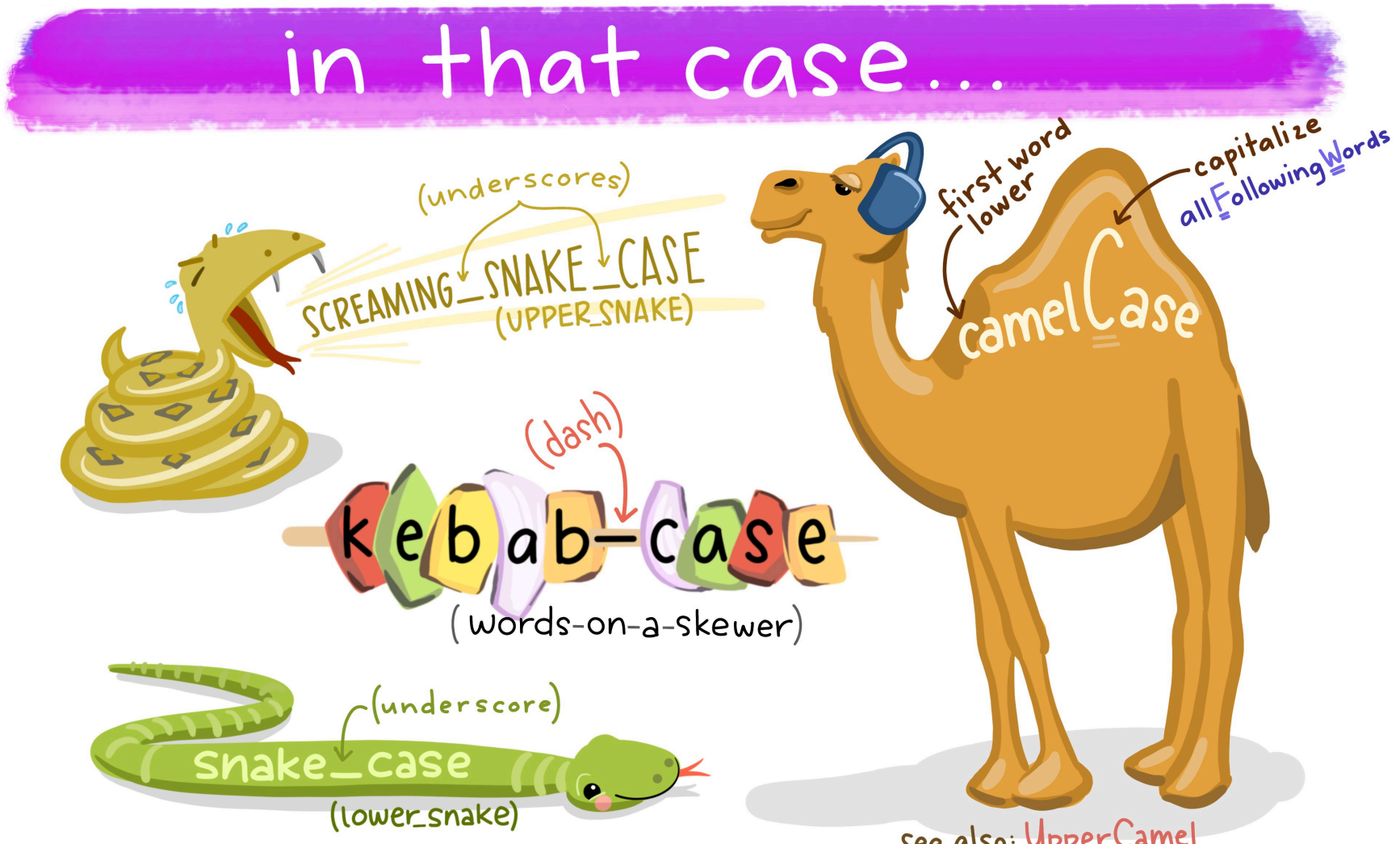
Ejemplos prácticos

❌ Evitar:

```
NombreCliente
fechaNacimiento
Ventas-Totales
ID CLIENTE
región_país
```

✅ Usar:

```
nombre_cliente
fecha_nacimiento
ventas_totales
id_cliente
region_pais
```


in that case...
(underscores)
SCREAMING_SNAKE_CASE
(UPPER_SNAKE)
first word lower
capitalize all Following Words
camelCase
(dash)
kebab-case
(words-on-a-skewer)
(underscore)
snake_case
(lower_snake)
see also: UpperCamel

Aplicando nomenclatura a nuestros datos

Nombres de variables consistentes

Datos de ventas (versión tidy):

vendedor	ventas_mensuales	comision_calculada	fecha_venta
garcía	45000	900	2024-08-01

Datos de inventario (versión tidy):

codigo_producto	stock_actual	precio_unitario	estado_stock
a001	5	100	critico

Ventajas:

- Fácil lectura y escritura
- Compatible con tidyverse/pandas
- Evita errores de tipeo
- Facilita autocompletado en IDEs

Metodología de resolución de problemas

1. Entender el problema

- ¿Qué resultado necesitamos obtener?
- ¿Qué datos tenemos disponibles?
- ¿Qué restricciones o condiciones existen?

2. Descomponer en pasos

- Dividir el problema en tareas más simples
- Identificar la secuencia lógica
- Considerar casos especiales

3. Identificar elementos clave

- **Input**: datos de entrada
- **Proceso**: transformaciones necesarias
- **Output**: resultado esperado

¿Qué es un pseudocódigo?

Definición

Descripción paso a paso de un algoritmo en lenguaje natural, pero con estructura lógica

Características

- Independiente del lenguaje de programación
- Enfoque en la lógica, no en la sintaxis
- Facilita la planificación antes de codificar
- Herramienta de comunicación entre analistas

Ejemplo simple

```
PARA calcular promedio de ventas por región:
1. CARGAR datos de ventas
2. AGRUPAR por región
3. CALCULAR promedio de cada grupo
4. MOSTRAR resultados
```

Ejemplo 1: análisis de satisfacción

Problema

Calcular el porcentaje de clientes satisfechos (rating ≥ 4) por sucursal

Pseudocódigo

```
INICIO
1. CARGAR datos de encuestas
2. FILTRAR solo ratings válidos (1-5)
3. CREAR variable satisfecho:
   SI rating ⩾ 4 ENTONCES 1 SINO 0
4. AGRUPAR por sucursal
5. CALCULAR porcentaje de satisfechos
6. ORDENAR de mayor a menor
7. EXPORTAR tabla final
FIN
```

Datos de entrada

Cliente	Sucursal	Rating
001	Norte	5
002	Norte	3
003	Sur	4
004	Centro	5
005	Norte	2

Resultado esperado

Sucursal	% Satisfechos
Centro	100%
Sur	100%
Norte	33%

Ejercicio 2: cálculo de comisiones

Problema

Los vendedores cobran comisión según sus ventas:

- **2%** si ventas < $50,000
- **3%** si ventas ≥ $50,000

Datos disponibles

Vendedor	Ventas
García	45000
López	65000
Martín	52000
Silva	38000

Consigna

Calculá la comisión de cada vendedor y ordenar los resultados por ventas (de mayor a menor)

Solución: cálculo de comisiones

Pseudocódigo

```
INICIO
1. CARGAR datos de ventas por vendedor
2. PARA cada vendedor:
   SI ventas ⩾ 50000 ENTONCES
     comision = ventas * 0.03
   SINO
     comision = ventas * 0.02
3. CREAR columna comision
4. CALCULAR total comisiones
5. ORDENAR por ventas desc
FIN
```

Resultado esperado

Vendedor	Ventas	Comisión
López	65000	1950
Martín	52000	1560
García	45000	900
Silva	38000	760

Total comisiones: $5,170

Ejercicio 3: control de inventario

Problema

La empresa necesita:

- Identificar productos con **stock crítico** (< 10 unidades)
- Calcular el **valor total del inventario** por producto
- Generar alertas para productos críticos

Datos disponibles

Producto	Stock	Precio
A001	5	100
A002	25	50
A003	8	200
A004	15	75

Consigna

¿Cómo estructurarías el proceso para obtener un reporte de productos críticos con sus valores?

Solución: control de inventario

Pseudocódigo

```
INICIO
1. CARGAR datos de inventario
2. CREAR variable valor_total:
   stock * precio_unitario
3. CREAR variable stock_critico:
   SI stock < 10 ENTONCES "Crítico"
   SINO "Normal"
4. FILTRAR productos críticos
5. CALCULAR valor total inventario
6. MOSTRAR alertas por categoría
FIN
```

Resultado: productos críticos

Producto	Stock	Estado	Valor
A001	5	Crítico	500
A003	8	Crítico	1600

Resumen general

- **Total inventario**: $4,000
- **Productos críticos**: 2 de 4
- **Valor en riesgo**: $2,100

Primeros pasos en análisis de datos

El flujo típico de análisis

`Importar → Explorar → Limpiar → Transformar → Analizar → Comunicar`

1. **Importar**: cargar los datos al entorno de trabajo
2. **Explorar**: entender estructura y contenido
3. **Limpiar**: corregir errores y inconsistencias
4. **Transformar**: crear variables y reestructurar
5. **Analizar**: aplicar métodos estadísticos
6. **Comunicar**: presentar resultados

Exploración inicial: preguntas clave

Sobre la estructura

- ¿Cuántas filas y columnas tengo?
- ¿Qué tipo de variables hay?
- ¿Los nombres de columnas son claros?

Sobre el contenido

- ¿Hay valores faltantes?
- ¿Existen datos atípicos o errores evidentes?
- ¿Los rangos de valores son razonables?

Sobre la calidad

- ¿Los datos están completos?
- ¿La información es consistente?
- ¿Hay duplicados?

Funciones de exploración inicial

En R

```r
head(datos)      # Primeras filas
tail(datos)      # Últimas filas
str(datos)       # Estructura
summary(datos)   # Resumen estadístico
dim(datos)       # Dimensiones
names(datos)     # Nombres de columnas
```

En Python (pandas)

```python
datos.head()         # Primeras filas
datos.info()         # Información general
datos.describe()     # Estadísticas descriptivas
datos.shape          # Dimensiones
datos.columns        # Nombres de columnas
```

Funciones principales: de Excel a programación

Operaciones que ya conocés de Excel

En Excel hacés:

- **Filtros**: Data > Filtro > Seleccionar valores
- **Columnas**: Seleccionar columnas A, C, E
- **Fórmulas**: `=PROMEDIO(A:A)` por grupo
- **Tabla dinámica**: Arrastrar campos a filas/columnas
- **Ordenar**: Data > Ordenar por columna
- **Nueva columna**: `=SI(A1>100,"Alto","Bajo")`

En programación es similar:

- **Filtrar**: `datos %>% filter(ventas > 1000)`
- **Seleccionar**: `datos %>% select(cliente, ventas)`
- **Resumir**: `datos %>% group_by(region) %>% summarise(promedio = mean(ventas))`
- **Agrupar**: `datos %>% group_by(vendedor)`
- **Ordenar**: `datos %>% arrange(desc(ventas))`
- **Crear**: `datos %>% mutate(categoria = ifelse(ventas > 100, "Alto", "Bajo"))`

Comparación práctica: análisis de ventas

En Excel (pasos manuales):

1. **Filtrar** datos → Click en filtro, seleccionar criterios
2. **Seleccionar columnas** → Click y arrastrar para elegir rangos
3. **Crear columna nueva** → Escribir fórmula `=SI(C2>1000,"Grande","Chica")`
4. **Tabla dinámica** → Insertar > Tabla dinámica > Arrastrar campos
5. **Copiar resultados** → Ctrl+C, Ctrl+V a otra hoja
6. **Repetir todo** → Si llegan datos nuevos, empezar de nuevo

En programación (código reproducible):

```
resultado ← datos %>%
  filter(fecha ⩾ "2024-01-01") %>%          # Filtrar
  select(cliente, region, ventas) %>%       # Seleccionar
  mutate(categoria = ifelse(ventas > 1000,  # Crear nueva variable
                            "Grande", "Chica")) %>%
  group_by(region, categoria) %>%           # Agrupar
  summarise(total = sum(ventas),            # Resumir
            promedio = mean(ventas)) %>%
  arrange(desc(total))                      # Ordenar
```

Ventajas de la programación vs Excel

Excel

✅ **Ventajas:**

- Visual e intuitivo
- Rápido para análisis simples
- Familiar para la mayoría

❌ **Limitaciones:**

- Proceso no documentado
- Errores al copiar fórmulas
- Límite de 1M filas
- Difícil automatización
- Pérdida de pasos intermedios

Programación

✅ **Ventajas:**

- Proceso completamente documentado
- Reproducible con datos nuevos
- Sin límites de tamaño
- Automatización total
- Control de versiones
- Menos propenso a errores

❌ **Desafíos:**

- Curva de aprendizaje inicial
- Menos visual al principio

Las 6 operaciones fundamentales

1. **Seleccionar columnas**

Excel: Click en columnas A, C, E
Programación: `select(cliente, ventas, region)`

2. **Filtrar filas**

Excel: Data > Filtro > Criterios
Programación: `filter(ventas > 1000, region == "Norte")`

3. **Crear nuevas variables**

Excel: `=C2*0.1` en columna nueva
Programación: `mutate(comision = ventas * 0.1)`

Las 6 operaciones fundamentales (cont.)

4. **Agrupar por categorías**

Excel: Tabla dinámica con campo en "Filas"
Programación: `group_by(vendedor, region)`

5. **Resumir/Calcular estadísticas**

Excel: Función en tabla dinámica (SUMA, PROMEDIO)
Programación: `summarise(total = sum(ventas), promedio = mean(ventas))`

6. **Ordenar resultados**

Excel: Data > Ordenar por columna
Programación: `arrange(desc(ventas))` o `arrange(cliente)`

En la próxima clase veremos la implementación práctica de estas operaciones

Síntesis

Puntos clave de la clase

Programación vs. herramientas tradicionales

- Reproducibilidad, escalabilidad y automatización

Datos ordenados (*tidy data*)

- Una estructura estándar que facilita el análisis

Pensamiento algorítmico

- Descomponer problemas en pasos lógicos
- Pseudocódigos como herramienta de planificación

Exploración inicial

- Primer paso fundamental: conocer nuestros datos
- Funciones básicas para entender estructura y calidad

Introducción práctica a tidyverse/pandas

Implementación de las funciones principales

¿Preguntas?