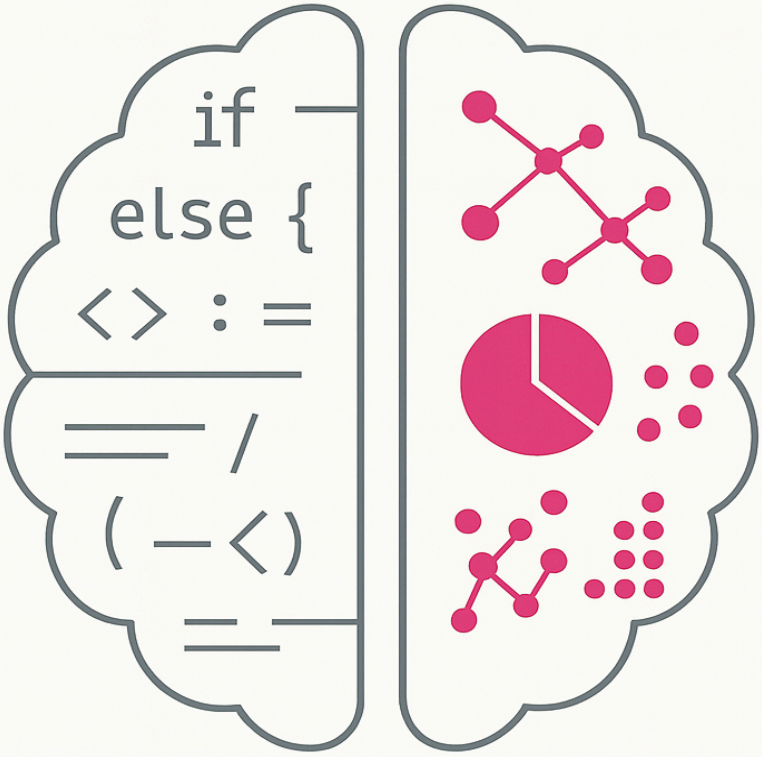
EL SESGO DEL DESARROLLADOR

Por qué Analizar ≠ Programar



Regina N. Molares

El Sesgo del Desarrollador

Por qué Analizar ≠ Programar

Regina N. Molares

DATA REGINA
FORMACIÓN PARA MIPYMES

EL SESGO DEL DESARROLLADOR

La imagen de la cubierta fue generada con inteligencia artificial. Edición final: Regina N. Molares.

Iª edición

Regina N. Molares

El Sesgo del Desarrollador

Por qué Analizar ≠ Programar

DATA REGINA
FORMACIÓN PARA MIPYMES

Dedicatoria

A quienes creyeron en mi trabajo.

A quienes alguna vez formaron parte de mi equipo,

por su apoyo cuando todo era cuesta arriba.

A mis alumnos,

mi inspiración, mi motor, y mi razón de ser.

A mi síndrome del impostor,

ese eterno rival que me empuja siempre más lejos.

A todos los demás: gracias por el combustible.

ÍNDICE

PARTE III — CAMBIO DE MENTALIDAD

9. Pensar con datos

El dato como disparador, no como respuesta. El análisis como conversación, no como pipeline.

10. Cuando los modelos fallan

El fracaso de la predicción como puerta de entrada al análisis genuino.

11. Datos sin usuarios no valen nada

Stakeholders, audiencias, decisiones. La dimensión humana del análisis.

12. Más allá del código: formar criterio

¿Qué significa tomar buenas decisiones con datos? Criterio, contexto y responsabilidad.

EPÍLOGO

Lo que no podemos automatizar, tal vez valga más

Una invitación a sostener la ambigüedad, explorar lo irrepetible, y formar pensamiento analítico más allá de la herramienta.

NOTA FINAL — VERSIÓN 1.0

PRÓLOGO

No fue una decisión. Fue una emergencia.

Me dejaron sola en una capacitación interna que ya estaba condenada antes de empezar. El programa no tenía sentido. Nadie quería estar ahí. Yo tampoco, si soy sincera. El grupo era hostil, el contenido era ajeno. Lo pensé: fingir mi propia muerte. Mi único recurso era improvisar. Así que eso hice.

Rediseñé todo el curso. Empecé a prestar atención a lo que sí necesitaban. No a lo que decía el plan. Descubrí que lo que más usaban no era Python ni SQL, sino planillas de cálculo. Yo venía de otro mundo. Mi formación era técnica. Pensaba en código, en estructuras, en scripts. Pero esa lógica no alcanzaba. Tuve que cambiar el chip.

Aprendí a mirar los datos al mismo tiempo que los enseñaba. Me encontré explicando cosas que todavía no terminaba de entender. Y algo raro pasó: funcionó. El grupo

empezó a responder, a engancharse, a traer sus propios ejemplos. Años después, todavía me escriben alguno de esos alumnos.

Casi en paralelo, me ofrecieron dar clases de desarrollo full stack. Ese sí era mi territorio. Python, HTML, bases de datos. Estaba todo armado. Pero algo había cambiado. La vieja seguridad técnica ya no me alcanzaba. Empecé a formarme en serio en análisis de datos. Le entré por la puerta chica —Google Sheets— y terminé diseñando planes de estudio, coordinando programas, entrenando equipos.

Y ya no hubo vuelta atrás.

Los datos me habían ganado. Porque no solo implican una nueva herramienta. Exigen otra manera de pensar. Y una vez que se aprende a ver de ese modo, no se puede desaprender.

Este libro es, en parte, un intento de explicarlo. Y en parte, una advertencia: **cuidado con lo que el código nos impide ver.**

Spoiler: *vamos a abarcar varios temas, pero la idea siempre es la misma.*

SOBRE LA AUTORA

Regina N. Molares es formadora y diseñadora de contenidos especializada en Ciencias de Datos, con una trayectoria consolidada en la capacitación técnica de adultos y en el desarrollo de materiales educativos aplicados.

Inició su carrera como desarrolladora Python, pero su recorrido profesional tomó un giro definitivo al involucrarse en capacitaciones donde el análisis de datos era una necesidad concreta. Esta experiencia fundacional —en la que debió adaptar y rediseñar contenidos en tiempo real, aprendiendo a la par de sus estudiantes— marcó el inicio de una nueva etapa, centrada en transformar la mirada técnica tradicional hacia una comprensión más profunda y crítica del análisis de datos.

Desde entonces, ha coordinado programas formativos en Ciencia de Datos, Business Intelligence y full stack Python

en instituciones públicas y privadas, incluyendo el Programa Talento Tech y Codo a Codo del Gobierno de la Ciudad de Buenos Aires. Actualmente se desempeña como contenidista, capacitadora y consultora pedagógica, con foco en diseñar experiencias de aprendizaje que combinen claridad conceptual, aplicación práctica y pensamiento analítico riguroso.

Cuenta con múltiples certificaciones internacionales en análisis y ciencia de datos (Google, IBM), y su formación abarca programación, estadística, visualización, pedagogía y metodologías ágiles.

“El sesgo del desarrollador” es el resultado de ese recorrido: una invitación a repensar lo que se da por sabido, y a analizar los datos más allá del código.

LO QUE SABEMOS NOS LIMITA

El conocimiento expande posibilidades, pero también puede reducirlas.

Cuando una formación es muy sólida en lo técnico, tiende a volverse totalitaria: anula matices, subordina otros saberes y no tolera disidencias. Con el tiempo, se convierte en un filtro que deja pasar solo lo que encaja en sus reglas. Todo lo demás parece inútil, fuera de lugar. Cuanto más cómodos estamos programando, más difícil se nos hace aceptar que no todo tiene solución en código. El análisis no se ejecuta, se piensa. Y los datos no se entienden con atajos: necesitan contexto y criterio. Eso no viene en ningún paquete instalable.

Este libro nace de una sospecha: que parte de lo que creemos saber sobre los datos nos está impidiendo analizarlos bien. No porque sea falso, sino porque está incompleto. Porque aprendimos a ver con herramientas

que sirven para operar, pero no para entender, interpretar y traducir.

Tener un *background* como desarrolladores no es un obstáculo. Al contrario: es una ventaja. Pero si queremos hacer análisis de datos, también puede ser un sesgo.
Este libro intenta hackear ese sesgo.
No con teoría abstracta ni con fórmulas nuevas, sino cuestionando lo que ya damos por hecho.

PARTE 1

Código, confianza y ceguera

1. EL SESGO DEL DESARROLLADOR

Todos empezamos creyendo que si el código corre, el análisis está completo. Que una tabla con *emojis* ya es un *dashboard*. Que si no hay errores de ejecución, entonces los datos están limpios. Que todo lo que no se puede automatizar es tiempo perdido.

A lo largo de mi experiencia como formadora, me encontré con muchos perfiles técnicos —yo incluida— que aprendimos a pensar en términos de sintaxis, eficiencia y control del flujo. Desde esa lógica, todo lo que no encaja en una estructura clara o no puede expresarse con una condición booleana empieza a parecer innecesario. *"Es demasiada teoría",* decimos, cuando se nos invita a pensar en los *stakeholders*. *"Mostrame qué fórmula pongo y listo",* cuando la consigna exige criterio, no solo ejecución.

Algunos borramos *outliers* sin pensarlo, como si fueran basura que molesta en el camino. Otros rellenamos los

huecos con ceros o promedios, solo para que el código no se queje. Visualizamos sin explorar. Resolvemos sin preguntar. Abrimos Visual Studio Code como si fuera una bola de cristal y pasamos del CSV al gráfico creyendo que eso es analizar. De alguna manera lo intuimos: no estamos entendiendo el dato, solo armando un pipeline que no extrae valor, solo tira información al fondo del océano, donde nadie vuelve a buscarla.

Y cuando aparece la propuesta de analizar un problema antes de resolverlo, asoma el fastidio: ***"¿Para qué sirve esto si no lo voy a programar?"***

Ese es el sesgo del desarrollador: la idea de que lo importante ocurre solo en el código. Que los datos son un insumo técnico, no un lenguaje. Que si algo no es procesable, no es relevante. No es un sesgo que nace de la ignorancia, sino del entrenamiento. Y por eso cuesta tanto verlo. Porque funciona como una herencia mental que seguimos aplicando, incluso cuando el problema cambió.

Este capítulo es una invitación a ponerlo sobre la mesa. No para demonizarlo, sino para entenderlo y domesticarlo. Mientras sigamos analizando con la lógica de desarrollo, vamos a seguir escribiendo buen código… pero llegando a malos análisis.

2. ANALIZAR NO ES PROGRAMAR

Entrar al mundo de los datos desde el desarrollo es como cruzar a la *Twilight Zone:* un universo paralelo. Se apuntala en la tecnología, pero está repleto de incertidumbre. *"Decime si está bien o mal." "¿No hay una fórmula para eso?" "Pero... ¿cuál es la respuesta correcta?".* La única respuesta honesta que les puedo dar, al principio, es: *“No sé”.*

Y si eso les incomoda… bienvenidos al análisis.

Quien viene del mundo del desarrollo busca certezas. Resultado esperado. Salida definida. `Assert()`. En programación, si algo está mal, falla. Punto. En análisis de datos, en cambio, todo parece funcionar bárbaro: el código corre, la tabla se llena de datos, los gráficos aparecen. Pero la interpretación puede estar equivocada. Y estamos al horno, porque el código no avisa. No hay error de sintaxis que lo delate.

El error en programación se ve, grita, rompe el build. El error en análisis se esconde, sonríe, se disfraza de dato correcto… y recién aparece cuando ya tomaste una mala decisión.

Analizar no es replicar. No es ejecutar. No es encontrar una solución universal. Analizar implica leer el contexto, preguntarse qué hay detrás de lo que se ve, dudar del formato y la fuente, tomar decisiones que no están escritas en ninguna documentación. Y para más diversión, la respuesta puede variar según quién la escuche.

Un buen análisis no siempre tiene una respuesta clara. A veces, el mejor resultado que puede darnos es una pregunta más incómoda… esa que nadie quería hacer, pero que cambia todo el camino del proyecto.

Y ahí es donde el sesgo técnico vuelve a asomar. Si no hay una función específica, si no hay una solución única, si no hay una métrica perfecta... entonces aparece el fastidio. Como si el análisis fuera menos profesional por no poder garantizar certeza.

Pero la incertidumbre no es un defecto. Es una parte estructural del análisis.

El desarrollo busca optimizar. El análisis busca entender, aunque eso implique no optimizar. Entender no siempre es optimizar: a veces es ralentizar, desarmar, volver a mirar. No porque el dato esté roto, sino porque la pregunta es frágil.

Analizar no es programar. Y en análisis, el *input* no son sólo los datos. También necesitamos preguntas y saber quién las hace.

3. NO TODO DATO ES ESTRUCTURABLE

Hay datos que no entran en un *DataFrame,* pero eso no significa que dejen de ser datos.

A veces llegan columnas con respuestas que parecen hechas por alguien que solo quería ver el mundo arder.
Respuestas como: *“Depende”*, *“Lo hago cuando puedo”*, *“No sé”*, *“Eso no aplica en mi caso”*.
Y la pregunta automática salta: *“¿Y esto cómo lo normalizo?”*.
O peor: *“Esto no sirve, está mal cargado”*.

Pero… ¿estás seguro?

Ese impulso de descartar lo que no se puede tabular no es inocente.
Viene de una manera de pensar muy técnica, muy estructurada, muy obsesionada con la limpieza.
Si algo no entra en `key : value`, molesta. Si no se puede tipificar, incomoda. Si no tiene una *primary key* para

sostenerlo, desconcierta. Si no lo puedo poner en un *DataFrame*, no lo uso.

Aprendimos a tratar los datos como si fueran variables bien tipadas. Y cuando algo no encaja, lo sentimos como un bug que hay que borrar rápido… aunque tal vez sea la única pista valiosa de toda la tabla.

En análisis, muchas veces, lo que no encaja es lo más interesante.

La trampa del formato limpio es que nos hace creer que la tabla está diciendo la verdad, cuando en realidad **solo está diciendo lo que el formato le permitió contar.**

No todo lo sucio es error. No todo lo ambiguo es descartable. Y no todo lo que incomoda hay que limpiar (hablamos de datos, no se entusiasmen).

Hay formularios donde las opciones no abarcan todas las posibilidades.
Hay encuestas que condensan vidas en una casilla de texto libre.
Hay respuestas que no respetan el formato porque el formato nunca fue pensado para ellas.

Ejemplo clásico: campo “Ocupación”. Y ahí te encontrás con joyitas como estas:

- *“Buscando changas.”*
- *“Cuido a mi suegra.”*
- *“No sé si cuenta, pero vendo ropa por IG.”*

Eso no entra en Empleado / Desempleado / Independiente. Pero sí dice algo. Dice mucho. Es incómodo, pero es real.

¿Esto lo limpiarías? ¿O lo mirarías mejor?

Primero, vos, desarrollador pura sangre: **respirá hondo.**

Después, imaginá que te llega esta tabla. Estás haciendo análisis para un programa de capacitación laboral.

ID	Edad	Ocupación	Ingresos	Observaciones personales
1	28	Empleado	180000	Todo bien
2		Laburo a veces, depende	No fijo	Cuido a mi nena. A veces hago fletes.
3	44	—	Ninguno	Me echaron en pandemia. Vivo con mi vieja.
4	31	Freelance / Vendo cosas por IG	Variable	Estoy viendo si consigo algo más estable.
5	37	Cosecha (no siempre)	Según semana	Cuando hay laburo, cobro. Si no, no.
6	25		$	¿Tengo que poner esto? No me quedó claro.

(La tabla continúa en la página siguiente)

(Continuación)

ID	Edad	Ocupación	Ingresos	Observaciones personales
7	29	Empleado registrado	220000	Nada para agregar
8	33	Cuido a mi suegra	Sin ingresos	No tengo tiempo para trabajar afuera.

Data simulada. Cualquier parecido con datos reales… es la vida misma

¿Qué harías con estas filas?

¿Qué campos eliminarías por “inconsistentes”?

¿Qué intentarías tipificar?

¿Qué transformarías antes de analizar?

¿Qué perderías si limpiás demasiado rápido?

Versión limpiada (mal)

Lo que verías si alguien decidiera "estructurar" esta tabla sin criterio analítico, priorizando formato por sobre significado:

ID	Edad	Ocupación	Ingresos	Observaciones
1	28	Empleado	180000	—
2	—	Desempleado	—	—
3	44	Desempleado	0	—
4	31	Independiente	Variable	—
5	37	Temporario	Variable	—
6	25	—	—	—
7	29	Empleado	220000	—
8	33	Tareas de cuidado	0	—

¿Qué se perdió?

- Se borraron **las frases espontáneas** que hablaban de inestabilidad, informalidad, sobrecarga doméstica, ambigüedad y desconexión con los formularios.
- Se forzaron ocupaciones a categorías **no representativas**, inventando rótulos para lo que no encajaba (como “Temporario” o “Tareas de cuidado”).
- Los campos vacíos fueron **rellenados con ceros o guiones**, como si el problema fuera solo “faltante de datos”.
- **La voz de las personas desapareció.**

Así queda una tabla cuando se estructura sin criterio: limpia, ordenada… y muda.

Se ven columnas completas, filas consistentes, etiquetas estándar. Pero ya no hay historias. Ya no hay señales de lo que está pasando en esas vidas.

Y si el objetivo era entender, no ejecutar… entonces, esta tabla limpia **no sirve.**

Lo que se perdió entre comillas, tildes y ceros, lo vamos a recuperar. Aguantá hasta el capítulo 6.

En desarrollo, los datos son insumo y la estructura es una necesidad.
En análisis, los datos son lenguaje. La estructura le hace perder riqueza y significados.

A veces, para que hablen, hay que romper el molde y leerlos como son: silvestres, desprolijos, incómodos. Porque a veces, la verdad viene sin formato.

No todo puede estructurarse sin pérdida.
No todo lo valioso tiene estructura.
No todo lo que no encaja es error.

Hay patrones que no entran en un diccionario, pero nos cuentan una historia.

Eso también cuenta. Eso también es un dato.

4. EL ALGORITMO NO DECIDE: DECIDIMOS NOSOTROS

Muchos desarrolladores creemos que un condicional es solo una bifurcación: un `if`, un `else`, y a otra cosa.

Y sí, en programación, un condicional evalúa una condición y decide qué bloque de código se ejecuta. Controla el flujo. Es simple y predecible.

Pero en análisis, los condicionales no controlan el sistema, construyen una forma de **interpretar la realidad.**

Cuando creamos una nueva columna a partir de un `if`, no estamos solo etiquetando.
Estamos diciendo: *esto va en este grupo*, *esto queda afuera*, *esto es de un tipo*, *esto es de otro*.

“Agrupé por edad porque era lo más fácil.”
“Puse sí/no para actividad.”

“Lo segmenté como venía en el dataset.”

Todos hicimos eso alguna vez. Pero hacerlo mal no es solo poco riguroso. Puede llevarnos a caer en sesgos sin darnos cuenta.

No por mala intención. Es por falta de criterio.

Situación	Condición mal planteada	¿Qué problema genera?
Edad	`edad > 30` → “adulto”	Arbitrario. Junta a una persona de 31 con alguien de 65. No tiene sentido demográfico ni laboral.
Actividad	`actividad == "sí"`	¿Qué es “sí”? ¿Trabajo formal? ¿Emprendimiento? ¿Changas? ¿Cuidado no pago?
Nivel educativo	`nivel == "Secundario"` → “Bajo”	Mezcla personas que terminaron, no terminaron, o hicieron parte de terciario. Puede ser ofensivo o simplemente inexacto.
Ingreso	`ingreso > 150000` → “Alto”	Define “alto” en términos absolutos sin contexto. No es comparable entre regiones o estructuras familiares.

Si planteamos mal la condición, lo que estamos segmentando no es solo el dato: es nuestra propia comprensión.

Esta tabla va a volver en el Capítulo 9. Para ese entonces, esperamos que el `if` tenga algo más de cabeza.

"Agrupé por edad porque era lo más fácil."

`if edad > 30 → "adulto"`. Listo. Segmentación hecha.

¿Por qué elegimos 30, y no 35, 40 o un corte por percentiles? ¿Sabés a quién dejaste afuera? ¿Qué patrón estás fabricando?

¿Qué tiene en común el de 30 con el de 60?
¿Quiénes quedan juntos y quiénes quedan separados?
¿Esa división tiene sentido o es apenas un corte cómodo?

"Faltaba la columna para agrupar, así que puse sí/no."

Binarios como "sí/no", "completo/incompleto" o "arriba/abajo" pueden parecer soluciones limpias, pero eliminan matices importantes.

¿Qué hacés con los que no entran?
¿Dónde va alguien que hace changas?
¿Quién define qué es "ocupado"?

En desarrollo, un condicional **controla el sistema**.
En análisis, **construye una mirada sobre los datos**.

Cada vez que escribimos una nueva columna con un `if`, estamos diciendo:

"Esto entra, esto no."
"Esto es de un tipo, esto de otro."

Eso no es inocente. No es neutro. **Es una decisión y debemos ser conscientes de por qué la tomamos.** Como toda decisión, puede ser acertada, torpe o directamente mala.

No se trata de entrar en debates ideológicos. Se trata de no armar categorías flojas, que reducen la complejidad del dato sin aportar análisis real.

Cuando automatizamos sin pensar, reproducimos estructuras que quizás no elegimos.
Después, cuando el análisis da resultados incoherentes o pobres, nos preguntamos qué pasó.

Lo que pasó es que decidimos sin darnos cuenta de que estábamos decidiendo.

El código no interpreta. Ejecuta.
Un condicional sin criterio no es solo un sesgo: es una decisión disfrazada de neutralidad. Puede distorsionar por completo la lectura del fenómeno.

PARTE 2

Ver de otro modo

5. EL TEXTO TAMBIÉN ES DATO

(Y no es solo el texto libre)

“El campo 'observaciones' no dice nada.”

“Esa columna está vacía.”

“Esto está mal cargado.”

Así, una y otra vez, descartamos piezas del rompecabezas. Porque no están completas, porque no son claras, o porque nos incomodan.

Los datos no necesitan estar bien escritos para hablar. Lo que necesitan es que alguien deje de limpiarlos a ciegas y se tome el trabajo de escucharlos.

Cuando analizamos, no trabajamos con un archivo. Trabajamos con un fragmento del mundo.

Ahora vamos a darnos un gustito por un segundo. Recordemos la definición de “entidad” en UML: *“es un objeto o concepto del mundo real que se representa en un*

modelo. Es una abstracción que representa un conjunto de objetos con características y comportamientos similares.”

Fin del recreo: en la práctica no es tan limpito y prolijo. Ese fragmento del mundo, esa abstracción, llega filtrada por decisiones humanas. No es neutra.
Lo que se incluyó, lo que se repitió y lo que quedó en blanco.
La forma en que están armadas las opciones, el orden de las filas, la manera en que se cargó la misma respuesta mil veces.

Todo eso también es lenguaje, aunque no esté en formato texto. Todo eso también narra.

Una base no es solo una tabla. Es una narrativa rota: una historia a medias, con huecos, repeticiones y silencios que también dicen algo.
Alguien la pensó, alguien la completó, alguien la procesó.
Y en el medio, algo se perdió, algo se agregó, algo se distorsionó.
Eso es parte del dato. Y muchas veces, es lo más revelador.

¿Quién eligió esas categorías?
¿Quién decidió que había que llenar esa columna?
¿Por qué la mitad está vacía y la otra mitad parece escrita a desgano?

¿Por qué todos los registros empiezan con el mismo nombre?
¿Es un error? ¿Una plantilla predefinida? ¿Un operador que copia y pega? ¿O una pista?

Veamos este infierno mal estructurado ¿Habrá algún un patrón narrativo si se la sabe leer con otros ojos?

¿Qué pasa si dejamos de mirar “estructura” y empezamos a mirar intención, repetición, omisión y contexto?

ID	Fecha ingreso	Motivo	Derivación	Operador	Código
1	1/3/2023	—	Sí	MARIA	210
2	1/3/2023	—	Sí	MARIA	210
3	1/3/2023	—	Sí	MARIA	210
4	1/3/2023	—	Sí	MARIA	210
5	2/3/2023	X	No	CARLOS	198
6	3/3/2023	Y	No	CARLOS	198
7	3/3/2023	Z	No	—	—
8	5/3/2023	—	Sí	MARIA	210
9	5/3/2023	—	Sí	MARIA	210
10	6/3/2023	W	No	—	198

Ejemplo: Registros de atención inicial en un centro comunitario

Cómo piensa el desarrollador:

“Motivo está incompleto. Hay que eliminar esas filas.”

“El operador no figura en algunas. Se completa con el valor más frecuente.”

“El código 210 se repite demasiado. Lo agrupo como ‘default’.”

Listo. Tabla limpia.

Cómo lee la tabla un analista:

- Del 01 al 04: mismos datos, mismo operador, mismo código, sin motivo informado. → *¿Carga por lote? ¿Derivación automática? ¿Alguien validando sin entrevistar?*
- El 05 y 06 tienen motivos distintos, otro operador, y no derivan. → *¿Protocolos distintos? ¿Cambio de turno?*
- Fila 07 está incompleta: no hay operador ni código. → *¿Atención fallida? ¿Caso que se cortó a mitad?*
- Fila 10 tiene código, pero no operador. → *¿Carga posterior? ¿Relleno por backoffice?*

Lo que nos dice la tabla:

- MARIA carga muchas atenciones con datos idénticos. *¿Sigue protocolo? ¿Evita entrevistas? ¿Carga masivamente desde papel?*
- CARLOS registra motivos individualizados. *¿Mayor precisión? ¿Distinto tipo de atención?*
- Donde no hay operador ni motivo, puede haber tensión institucional, mala carga o casos abandonados.

Los silencios, las repeticiones y las omisiones no son errores. Son pistas.

La tabla puede parecer mal cargada, pero si la miramos bien, es la historia de cómo se atiende en ese lugar.

No está dicha en texto libre. No hace falta.

Está escrita en repeticiones sospechosas, en campos vacíos, en días que no cuadran. Está escrita como se vive la realidad.

Entonces:

El dato también es texto cuando alguien pone una tilde donde no va.
Cuando alguien escribe TODO EN MAYÚSCULAS.

Cuando el usuario repite *“ninguna”, “ninguna”, “ninguna”* como si estuviera harto.
Cuando usa comillas para marcar distancia.
Cuando completa algo que nadie le pidió.
O cuando deja en blanco justo lo más importante porque no confía.

También cuando hay un campo de texto libre. El silencio habla.
Y ese no es el único lugar donde se esconden las historias. A veces están en la forma, no en el contenido. En la excepción, no en el promedio. En lo que no se carga, no en lo que aparece completo.

Ver de otro modo es dejar de preguntarnos solo “qué dice este dato”... y empezar a mirar cómo lo está diciendo

Qué quiso hacer quien lo completó.
Qué esperaba quien lo diseñó.
Qué decisiones ya fueron tomadas antes de que empecemos a analizar.

Un dataset no solo se ejecuta ni se visualiza: también se lee. Y cuando aprendemos a leerlo, descubrimos verdades que ningún gráfico puede dibujar.

Errores repetidos, omisiones sistemáticas, duplicaciones sospechosas… todo eso puede hablarnos de un protocolo

mal diseñado, un ambiente de trabajo tenso, una comunicación deficiente. Todo eso también son *insights*.

Y si no aprendemos a leer eso, estamos desperdiciando gran parte del potencial de los datasets.

6. EL RUIDO NO ES BASURA

“Eliminé los valores extremos porque rompían el promedio.”

“Saqué los nulos para que el modelo no tire error.”

“Descarté los casos que no coincidían.”

Y ahí se fue por el caño el hallazgo, el conflicto, la excepción que explica el resto.

Limpiar no siempre es mejorar

Limpiar es un gesto técnico. Interpretar es un gesto analítico.

El secreto no está en sólo limpiar, porque limpiar sin entender qué estás sacando, es suicida.

Los valores atípicos no molestan el análisis. Molestan la comodidad del que quiere cerrar el archivo rápido. **Y la comodidad, en datos, es la enemiga mortal del criterio**.

“Esto rompe el promedio.”

Perfecto. Quizás eso es lo único que vale la pena estudiar.

Ejemplo de escenario concreto:

Una base de ingresos por hogar.

- El 97% está entre 150.000 y 350.000 pesos.
- Un registro marca 3.900.000.
- Otro: 0.

El técnico dice: *"son outliers, los saco"***.** *Coffee break*.

El analista pregunta:

- ¿El de 0 vive de otra persona? ¿Está subregistrado?
- ¿El de 3.9 millones es evasor? ¿Es error de carga? ¿Es real?

El técnico normaliza. El analista interpreta.

Peligro: Desaparecer el problema

En muchos ámbitos, el *outlier* no es ruido: es **el síntoma**.

- El valor extremo en un informe de seguridad.
- El salto inesperado en el registro de temperatura.
- El ingreso 10 veces superior en una licitación.

Limpiarlos es como apagar la alarma de incendio porque el sonido molesta. La señal no desaparece, solo la ignoramos hasta que es tarde.

Resolución de la problemática del Capítulo 3

En el capítulo 3 vimos cómo una tabla puede ser despojada de toda su riqueza en nombre de la “estructura”.

Hay otra forma de trabajar con esos datos.

Veamos qué aparece cuando no la limpiamos, sino que la desmenuzamos.

ID	Edad	Formalidad laboral	Estabilidad	Tipo de actividad	Comentario interpretado
1	28	Formal	Alta	Relación de dependencia	Sin observaciones relevantes
2	—	Informal	Baja	Changa / tareas varias	Sostiene económicamente a su hija con trabajos ocasionales
3	44	Ninguna	Muy baja	Desocupado	Expulsado del mercado formal, dependencia familiar
4	31	Independiente	Media	Emprendimiento informal	Busca mayor estabilidad laboral
5	37	Temporaria	Baja	Trabajo rural estacional	Ingreso atado a condiciones externas

(La tabla continúa en la página siguiente)

(Continuación)

ID	Edad	Formalidad laboral	Estabilidad	Tipo de actividad	Comentario interpretado
6	25	No declarada	Muy baja	No determinada	Muestra inseguridad frente al formulario
7	29	Formal	Alta	Relación de dependencia	Sin observaciones relevantes
8	33	Ninguna	Muy baja	Tareas de cuidado	Carga no remunerada le impide acceder al mercado laboral

Reconstrucción interpretativa basada en los mismos registros.

Esta es la misma tabla, pero interpretada con un criterio analítico. No es que los datos estuvieran mal, que fueran inútiles porque no respondían a un criterio. Es que no alcanzaba con sólo procesarlos. Había que leerlos.

Cambiando el mindset

- *"Esto es un outlier, lo descarté."* → Adiós al fraude, al riesgo, al escándalo que ibas a destapar.
- *"Eliminé los nulos para que no me tire error."* → ¿Y si ese nulo era porque el campo no debía existir?
- *"Estos datos están mal, no coinciden con el resto."* → Genial. **Preguntate por qué.**

¿Y si esos datos “raros” no son un error, sino la única grieta por donde se ve el problema real?

¿Hay ámbitos donde los datos “raros” son nuestra materia prima?

Solo imaginemos que trabajamos en alguno de estos escenarios: detección de fraudes, criminalística, control aéreo, de calidad, aduana, salud, educación… ¿seguimos enumerando o ya está prendiendo la idea? En todos encontramos “casos raros” que nos disparan alguna alarma. Y **la alarma dispara la acción**. Ahí está la importancia del dato atípico.

Y acá rematamos:

El análisis no se basa en la mayoría. La mayoría explica lo común. Nos da las estadísticas.
Pero si querés entender lo importante, si querés descubrir fallos, mejorar procesos… mirá lo que rompe la curva.
Lo que queda afuera.
Lo que no se puede predecir.
Eso no es basura.
Eso no es basura. Es la **realidad** filtrándose a pesar de los filtros. Y si la sacamos, no limpiamos el dato: borramos nuestra capacidad de entender.

7. CONSULTAR NO ES COMPRENDER

Consultar no es comprender

“Ya hice el SELECT, ¿qué más querés?”
“Le pedí a sistemas que me pasen el Excel.”
“Con la tabla dinámica nos arreglamos.”
Perfecto. Tenemos datos.

¿Y ahora qué?

Es muy fácil confundir acceso con comprensión.
Saber ejecutar una query no es entender lo que devuelve. Y si está mal planteada, no entendemos ni lo que estamos pidiendo.
Obtener una tabla no es lo mismo que saber qué está contando.
Y tampoco ver qué está dejando afuera.

SQL no es análisis. Es apenas el principio.

Saber escribir queries no garantiza que sepamos qué preguntar.
Un SELECT puede ser impecable en lo técnico, pero inútil en lo analítico.

¿Te devolvió muchos datos?
¿Te respondió lo que necesitabas?
¿O simplemente ejecutaste lo que podías pedir, no lo que tenías que entender?

El error del "pipeline automático"

Consulta → Tabla → Gráfico → Informe → Chau análisis

Esa secuencia no analiza nada: sólo simula. Da la ilusión de que el trabajo está hecho, cuando en realidad no empezó.

Agrupar, contar, sacar promedios…
Sin contexto, eso solo produce resúmenes planos.
Y muchas veces, resúmenes que deforman lo que está pasando.

"Promedio de ingresos por categoría."
¿Comparaste las cantidades de registros en cada grupo?
¿Chequeaste los nulos?
¿Revisaste los rangos extremos?
¿Sabés qué quedó afuera de esa agrupación?

Pedir no es conseguir

“Le pedí a sistemas que me pasen el Excel.”

“Con eso ya hacemos los gráficos.”

¿Y qué pediste exactamente?
¿Todos los casos? ¿Sólo los activos? ¿Los últimos seis meses?
¿Qué pasa si lo que te mandaron ya viniese filtrado, truncado, o agregado?
La consulta que recibimos ya es una interpretación.
Y si no la cuestionamos, estamos analizando un recorte sin saberlo.

Lo que no se pregunta, no aparece

Los datos no te corrigen.
No levantan la mano para avisarte que estás consultando mal.
No dicen: *“che, eso que estás pidiendo no tiene sentido”.*
Devuelven lo que les pedís. Aunque no tenga lógica. Aunque no tenga valor. Aunque no sirva.

El análisis empieza cuando cuestionamos lo que estamos viendo.
Cuando no damos por sentada la estructura de la tabla.

Cuando volvemos a mirar la consulta para entender qué asume, qué omite, y qué impone.

Un ejemplo rápido

Querés saber cuántas personas asistieron a una capacitación.
Consultás:

```
SELECT COUNT(*) FROM inscriptos WHERE asistencia = 'sí';
```

Perfecto. Te da 238.

¿Y si te dijera que había 600 inscriptos?
¿Y si “asistencia” estaba vacía para los que no fueron?
¿Y si alguien escribió “Sí” con tilde?
¿Y si hubo tres fechas distintas y tu query sólo toma una?

El dato es correcto. Pero el análisis está incompleto.

Lo que devuelve tu consulta depende de lo que sabés preguntar

Contexto de consulta: Supongamos que estamos trabajando con una base de capacitaciones. La tabla original contiene:

- `id_participante`
- `fecha_asistencia`
- `nombre_capacitacion`

- `modalidad` (presencial / virtual)
- `asistencia` (sí / no / vacío / “Sí” con tilde, etc.)

“¿Cómo estamos de asistencias?”, tira el coordinador de los cursos. Y ahí va, nuestro primer intento:

```
SELECT nombre_capacitacion, COUNT(*) AS
total_asistencias
FROM capacitaciones
WHERE asistencia = 'si'
GROUP BY nombre_capacitacion;
```

El resultado:

Nombre del curso	Total de asistencias
Excel Nivel 1	75
Python Básico	103
Comunicación Asertiva	29

¿Qué tiene de malo?

- No contempla registros con “Sí” con tilde, ni con valores vacíos o variantes ("Si", "si").
- No filtra por fecha ni por modalidad.
- No compara contra la cantidad de inscriptos → no sabemos el % de asistencia.
- Cuenta asistencias “como están”, sin limpieza ni validación.

- Omite inconsistencias: por ejemplo, capacitaciones con 0 asistentes no aparecen.

Vamos por el segundo intento, y esta vez, estamos más avispados en manejo de datos:

```
SELECT
  nombre_capacitacion,
  COUNT(*) AS total_inscriptos,
  SUM(CASE
      WHEN LOWER(TRIM(asistencia)) IN ('si', 'sí')
THEN 1
      ELSE 0
  END) AS asistieron,
  ROUND(100.0 * SUM(CASE
      WHEN LOWER(TRIM(asistencia)) IN ('si', 'sí')
THEN 1
      ELSE 0
  END) / COUNT(*), 1) AS porcentaje_asistencia
FROM capacitaciones
WHERE fecha_asistencia BETWEEN '2024-06-01' AND
'2024-06-30'
GROUP BY nombre_capacitacion;
```

Resultado:

nombre_capacitacion	total_inscriptos	asistieron	porcentaje_asistencia
Excel Nivel 1	150	75	50
Python Básico	150	103	68.7
Comunicación Asertiva	42	29	69

¿Qué mejora?

- Limpia y normaliza valores de texto antes de contarlos.
- Compara contra el total de inscriptos.
- Delimita el rango temporal.
- Devuelve un análisis más útil para evaluación de impacto, no solo para conteo.

La diferencia entre consultar y analizar no está en la sintaxis.

Está en el criterio con el que decidimos qué mirar, qué comparar y qué dejar afuera.

Consultar sin criterio es generar confusión

En datos nada es inocente. Las queries tampoco. Los bias están en todas partes.

Reflejan lo que esperás encontrar, no lo que necesariamente pasa.
Y muchas veces, ese sesgo se arrastra al resultado final, que "parece estar bien", pero no cuenta nada.

"Ya tengo los datos, ahora solo hay que graficarlos."
Error.
Si no sabés qué estás mirando, el gráfico va a decorar tu ignorancia.

El dato no habla solo

Podés tener una tabla perfecta y seguir sin tener información.
Podés correr una query brillante… y obtener un resumen inútil. La consulta es apenas la puerta de entrada.

Consultar es pedir.

Analizar es saber qué pedir, por qué pedirlo y qué hacer con eso cuando llega.

8. LOS GRÁFICOS NO SON DECORACIÓN

Los gráficos no son decoración

“Metí un gráfico porque sobraba espacio.”
“Usé pie chart porque es más lindo.”
“No importa el eje, lo importante es que se vea claro.”
“Le puse colores para que parezca más profesional.”

Pongamos orden.
Un gráfico no embellece datos. Un gráfico los interpreta.
Y como toda interpretación, cuenta algo. A veces bien. A veces mal. A veces lo que alguien quiere que digamos.

Un gráfico no muestra verdades: muestra las decisiones que tomamos sobre qué ver, qué esconder y cómo queremos que otros lo lean.

Por eso:

- Un gráfico no es un resumen visual.
- Un gráfico es una narración con intención.

Y como toda narración, puede ser potente, imprecisa, irrelevante… o directamente manipuladora.

La visualización no es neutra

Ya lo dije antes: en datos nada es inocente. Y eso se cumple también con los gráficos, y de manera contundente. Porque los gráficos son nuestro canal de comunicación.

No existe el gráfico objetivo.

Todo gráfico elige:

- Qué datos mostrar (y cuáles no).
- Qué escala usar (y qué comprimir).
- Qué categorías comparar (y cuáles agrupar).
- Qué eje destacar (y cuál ignorar).
- Qué forma tomar (y cómo distribuir el espacio).

Elegir un gráfico no es un paso técnico.
Es una toma de posición sobre lo que importa y lo que debe ser visto (u ocultado).

El gráfico no habla por sí mismo. Decime qué datos usaste. Decime qué cortaste. Decime qué decidiste dejar afuera.”

¿Y si me guío por lo que se ve más claro?

Claridad no es lo mismo que buen criterio.
Un gráfico claro puede estar completamente mal planteado.

¿Lo ves bien? Genial.
¿Lo entendés bien? Esa es otra historia.

Errores que vemos todo el tiempo

No hace falta manipular para graficar mal. A veces basta con no pensar qué estamos mostrando. Ejemplos:

- Usar mapas de calor sin justificación, donde un gráfico de burbujas o barras segmentadas habría contado mejor la historia.
- Tablas decoradas con emojis, que sólo distraen y dificultan la lectura.
- Burbujas sin eje claro ni escala explícita.
- Pie charts con más de 6 categorías, o con valores demasiado cercanos: ilegibles e innecesarios.
- Gráficos con ejes invertidos, escalas cortadas, o comparaciones sin contexto.
- Visualizaciones que repiten lo mismo que dice la tabla, pero con colores.

- Paletas de colores absurdas (como usar rojo para cosas buenas y verde para cosas malas).
- Elementos prescindibles que elevan la carga cognitiva, diluyen la contundencia del mensaje o distraen del foco de atención.

No son detalles de “estética”. Cuando graficamos mal, no solo ensuciamos la pantalla: desinformamos a todo color.

Esos errores no son técnicos: son conceptuales.

Una misma tabla, dos relatos completamente distintos

Contexto: evaluación de desempeño de cursos internos en una empresa. Nos presentan la siguiente tabla:

Curso	Cohorte	Total Inscritos	Aprobados
Curso A	Cohorte 1	120	118
Curso A	Cohorte 2	180	179
Curso B	Cohorte 1	220	70
Curso B	Cohorte 2	190	189
Curso C	Cohorte 1	100	99
Curso C	Cohorte 2	140	138

Gráfico 1 – Pie chart decorativo

¡Sorpresa! No está tan mal la tasa de aprobados.

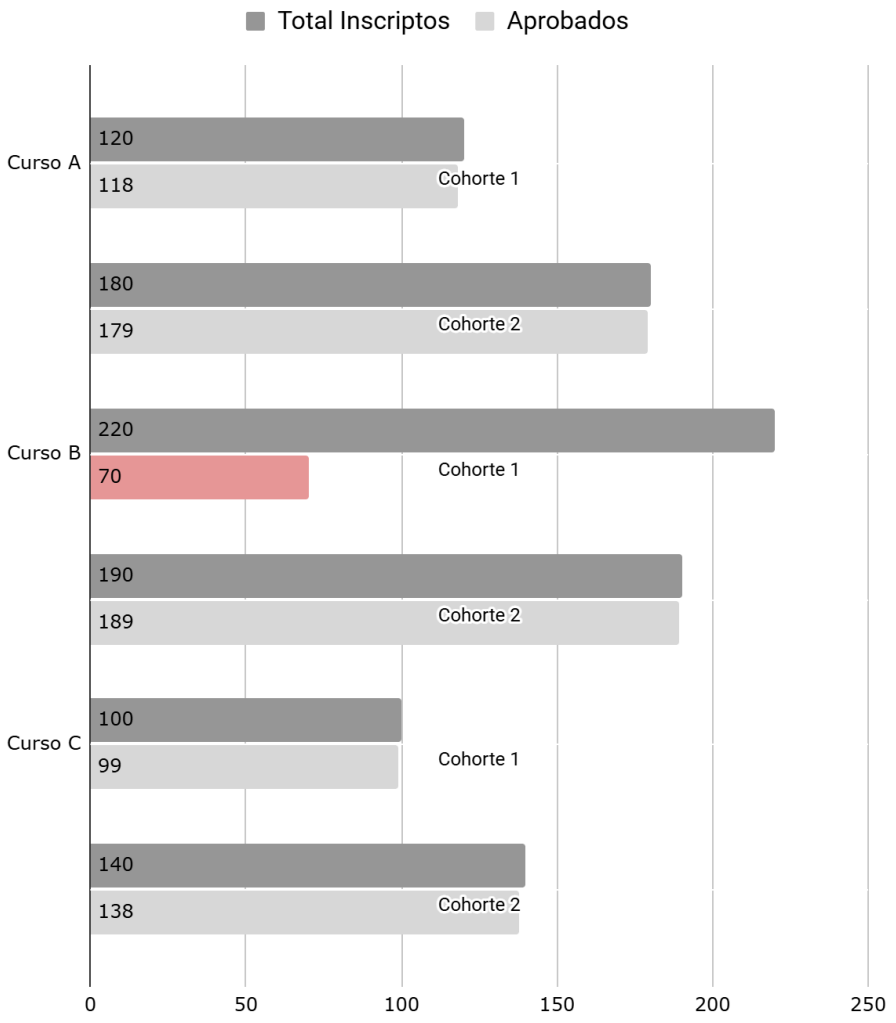
Colorcito. Efectos. Rótulos. Fin.

Problemas:

- Agrupa 3 cursos distintos.
- No hay ejes.
- No hay proporciones ni escalas.

No hay forma de analizar nada.

Gráfico 2 – Barras con segmentación por curso y cohorte



Mejoras:

Se muestran tasas de aprobación por curso y por tanda (cohorte 1 / cohorte 2). Se ve que:

- Una cohorte tiene más inscriptos… y una tasa de aprobación alarmantemente baja.
- El resto de los cursos tiene una tasa de aprobación casi perfecta.

Ese gráfico cuenta una historia.
No adorna. No es *"bonito"*. Incluso hasta se ve más antipático. Pero muestra lo importante. Dónde hay un fallo. Dónde necesitamos activar.
Es útil. Esa utilidad no está en el color ni en las formas: está en cómo estructura la mirada.

El buen gráfico molesta

Sí. Molesta.

Porque revela patrones que no querías ver.
Porque muestra que algo no funciona.
Porque evidencia lo que se quiso disimular con números bonitos.

Por eso, a veces, caemos en el error de mostrar no lo que es más útil… sino lo más cómodo.

Un buen gráfico no es el que decora, es el que incomoda. Porque nos obliga a mirar lo que está mal, lo que nadie quería exponer, y nos deja sin la excusa de ignorarlo.

No es diseño. Es lectura.

Elegir el gráfico correcto no es cuestión de estética.
Es saber qué estamos mirando y por qué.
Es poder defender por qué lo mostramos así y no de otra forma.
Y es, sobre todo, tener claro qué está viendo —y cómo lo va a interpretar— quien lo recibe.

PARTE 3

Cambio de mentalidad

9. PENSAR CON DATOS

Pensar con datos

“¿Qué dice el dato?”
“Tirame un insight.”
“Ahí tenés los datos, ahora sacá una conclusión.”

Y ahí es donde solemos meter la pata: creemos que el análisis es un trámite.
Un *pipeline* donde entra un CSV y sale —supuestamente— una verdad absoluta.

Pero no. Pensar con datos no es eso. No es esperar respuestas.
Es hacerse preguntas. Es construirlas. Reformularlas. Abandonarlas y volver a empezar.

El dato no “dice” nada por sí mismo. No es un oráculo, no tiene voz. Si no lo interrogamos, lo único que devuelve es lo que está ahí. Lo que fue cargado. Lo que alguien definió que

valía la pena registrar.
Y si eso no se interroga, no hay análisis. Hay resumen.

Pensar con datos no es automatizar

Podemos tener dashboards actualizados, modelos de predicción, informes bien armados y todavía no entender qué está pasando.

Pensar con datos no es delegar el análisis al sistema. Es leer lo que el sistema no puede resolver: las ambigüedades, las contradicciones, los casos frontera.

Es un proceso engorroso y repetitivo *(Sí, lo sé. Como los conceptos que les estoy machacando mil veces en este libro)*.

Pensar con datos es preguntarnos:

- ¿Qué patrón estoy asumiendo?
- ¿Qué categoría estoy dando por válida?
- ¿Qué historias quedan afuera con este recorte?

El dato no es la respuesta: es el disparador

A veces se revela un patrón y cambia todo.
A veces lo único que tenemos es una duda nueva, pero más precisa.
Un buen análisis no siempre dispara una acción.
A veces deja más preguntas que certezas. Y eso no es un

fallo.

Eso es empezar a pensar con datos.

Dos formas de mirar un mismo dato

Enfoque mecánico	Enfoque analítico
“Promedio general: 6,3” → “Aprobó.”	“Promedio general: 6,3” → “¿Aprobó todo o arrastra materias?”
“75% de asistencia” → “Listo, cumple.”	“75% de asistencia” → “¿Por qué faltó? ¿Cuándo? ¿En qué contexto?”
“50.000 usuarios más” → “Estamos creciendo.”	“50.000 usuarios más” → “¿Qué hicieron? ¿Repitieron? ¿Se quedaron?”

Pensar con datos no es ejecutar

Es detenerse a interpretar.

Eso no lo hace un script. Lo hacemos nosotros.

En capítulos anteriores vimos cómo una mala segmentación, una limpieza automática, o una consulta sin criterio, pueden distorsionar el análisis.

Ahora es momento de afilar la cabeza.

Dejar de esperar que los datos nos “digan algo” y empezar a pensar qué queremos ver, qué necesitamos

Para que quede claro el cambio de mentalidad que necesitamos, pongamos en contraste las dos formas más comunes de encarar el trabajo con datos:

Automatizar sin pensar	Pensar con datos
“El modelo ya está armado.”	“¿Qué supuestos tiene este modelo?”
“Saqué los extremos, tiraban el promedio abajo.”	“¿Qué explica ese valor extremo?”
“Agrupé por categoría y conté.”	“¿Qué relación hay entre esas categorías? ¿Faltan algunas?”
“La tabla me dio esto.”	“¿Qué datos no están en esa tabla?”
“Filtré y graficé. Ya está.”	“¿Qué pasa si cambio la forma de visualizarlo?”
“Así venía en el dataset.”	“¿Quién definió esa estructura? ¿Por qué?”
“Pido los datos, saco la conclusión.”	“Exploro, formulo hipótesis, y a veces no hay una única respuesta.”
“Uso código limpio.”	“Uso pensamiento riguroso.”

Del automatismo al criterio: automatizar sin pensar vs. pensar con datos

No hay pasos a seguir para pensar con datos.
Se trata de elegir preguntas, tensionar supuestos y sostener el análisis aun cuando no hay una respuesta clara.
Lo más valioso de trabajar con datos no está en lo que sabemos ejecutar, sino en lo que nos atrevemos a cuestionar. Los datos por sí solos no piensan ni concluyen nada: si no los interrogamos, solo son números sin sentido.

10. CUANDO LOS MODELOS FALLAN

“El modelo no sirve, no predice bien.”
“Tuve underfitting, lo tengo que tirar.”
“¿Cuál es la accuracy buena?”
“No da bien porque hay ruido.”

Y entonces... ¿lo descartamos?

Eso es lo que suele pasar. Si no acierta, si no “da bien”, lo consideramos fallido. Pero en muchos casos, el fallo no es el problema: **es el síntoma**.

Un modelo que no predice bien puede estar revelando que no entendemos del todo el fenómeno. Que falta una variable clave. Que las categorías están mal pensadas. Que hay excepciones que no son errores, sino indicios.

Ese “mal resultado” puede ser lo más valioso de todo el análisis. A esta altura ya sabemos que a los de datos nos gusta el bardo. Y que donde otros ven un fallo, nosotros vemos una pista.

¿Qué es un modelo?

Para no perder a nadie:
Un modelo es una simplificación del mundo. Una estructura matemática que intenta representar relaciones entre variables para predecir o clasificar algo.

El modelo no "entiende". Solo generaliza.

Conceptos clave que vale tener a mano

- ***Overfitting:*** el modelo memoriza los datos, no los comprende. Tiene un rendimiento excelente en el conjunto de entrenamiento, pero falla con datos nuevos.
- ***Underfitting:*** el modelo no capta patrones significativos. Su rendimiento es bajo incluso en el conjunto de entrenamiento.
- ***Accuracy:*** porcentaje de aciertos sobre el total. No siempre refleja si el modelo es útil (por ejemplo, si hay clases desbalanceadas).
- **Ruido:** variabilidad aleatoria, datos erráticos o no explicados. Pero atención: no todo lo que parece ruido lo es.

La trampa del modelo que “funciona”

Un modelo puede tener una accuracy del 92% y no servir para nada.
¿Por qué? Porque puede estar acertando sobre lo irrelevante, o ignorando por completo el fenómeno clave.

- *¿Predice bien?* Sí.
- *¿Explica algo útil?* No necesariamente.

Eso es lo que nos pasa cuando priorizamos el rendimiento técnico por encima del criterio analítico.
Medimos, afinamos, optimizamos… y muchas veces seguimos sin entender nada. El modelo luce brillante, pero no explica el fenómeno.

Cuando el modelo falla, empieza el análisis

Pensar con datos también es saber frenar cuando un modelo falla, no para arreglarlo, sino para interrogarlo.

Contexto: Queremos predecir si una persona va a abandonar un curso antes de terminarlo.

Tenemos un dataset con 1000 personas:

Resultado real	Cantidad
Completó el curso	950
Abandonó el curso	50

MODELO A – “El perfecto inútil”

- Predice que todas las personas van a completar el curso.
- Accuracy: 95%
- ¿Sirve? No.

¿Por qué?

Porque ignora por completo el fenómeno que queremos entender: el abandono.

El 95% de aciertos se debe a que **la clase mayoritaria domina la predicción,** no a que el modelo entienda nada.

MODELO B – “El que incomoda, pero revela”

- Detecta correctamente 30 de los 50 casos de abandono.
- Se equivoca en algunos positivos.
- Accuracy: 91%
- Ahora sí podemos estudiar:
    - Qué variables aparecen en los casos que sí abandona (situación laboral, conectividad).

    - Qué condiciones se repiten (¿falta de tiempo?, ¿desinterés?).
    - Qué trayectorias podrían predecir deserción.

Este contraste muestra que una predicción “correcta” puede ser analíticamente vacía, mientras que un modelo “incómodo” puede ser la puerta al verdadero entendimiento.

Cuando falla, nos tenemos que preguntar

- ¿Por qué no predice bien?
- ¿Qué casos le cuestan?
- ¿Qué patrón rompe la curva?
- ¿Qué variables están ausentes?
- ¿Dónde está el sesgo?

El modelo que falla **nos obliga a pensar.**

No hay modelo perfecto. Pero hay errores reveladores

Un análisis valioso no siempre genera una predicción confiable. A veces detecta una inconsistencia, una falla estructural, o una dinámica que nadie había registrado.

Y ahí es cuando se pone interesante.

Un modelo no entiende. Solo pone a prueba una intuición. Si falla, no es basura: es un **punto de partida** para pensar mejor.

Un modelo puede fallar… y a veces, eso es lo mejor que nos puede pasar. Porque mientras otros se apuran a mostrar una predicción que “da bien”, nosotros estamos viendo el fenómeno real que no encaja. Y ahí es donde empieza el verdadero análisis.

11. DATOS SIN USUARIOS NO VALEN NADA

“Ya le pasé los datos, que hagan lo que quieran.”
“Ese número está en la tabla, si no lo vieron no es mi culpa.”
“Yo solo entrego el dashboard.”
“Lo hice técnico para que no me pregunten.”

¿Te suenan? A mí también. Y cada vez que las escuchamos, sentimos lo mismo: ahí se rompió el puente entre los datos y las decisiones.

Los datos no existen en el vacío. No son neutrales. No son autosuficientes.
Todo dato importa porque alguien lo necesita para algo.
Si no hay nadie del otro lado, el análisis se vuelve un monólogo.

El dato tiene su razón de ser

Los datos tienen origen, tienen contexto, y sobre todo: tienen destino.
No analizamos para tener razón. Analizamos para aportar a un proceso que incluye a otros: usuarios, equipos, decisores, actores que van a hacer algo (o no) con lo que producimos.

Si ignoramos esa dimensión humana, el análisis puede ser impecable en lo técnico… y estar muerto en lo único que importa: su capacidad de cambiar algo. No hay valor estratégico.

Un dataset, muchas historias

Un mismo conjunto de datos puede contarnos cosas muy distintas según quién lo mire y qué necesite.
No es lo mismo un informe para un responsable técnico que para un director de programa.
No es lo mismo hablar con recursos humanos que con el equipo de campo.
No todos entienden lo mismo y no todos necesitan lo mismo.
Y ahí está la clave: saber leer el dato en función de su interlocutor.

- Porque si hablamos de retención, ¿de qué hablamos?
- ¿Del porcentaje mensual de permanencia?
- ¿De la evolución por cohortes?
- ¿De la experiencia percibida por los usuarios que se quedan?
- ¿De los que se van, pero vuelven?

Lo que mostramos depende de para quién lo hacemos.
Y si no sabemos quién está del otro lado, entonces estamos adivinando. O peor aún: hablando solos.

Si nadie puede usarlo, no sirve

Un análisis que no se entiende no es sofisticado: es inútil.
No es más pro porque tenga jergas. No es más valioso porque sea complejo.
Si no permite que alguien actúe, decida o comprenda algo mejor… no cumplió su función.

Mostrar datos no es lo mismo que comunicar información.
Un gráfico vistoso no reemplaza una conversación.
No basta con decir “esto está en la tabla”. Hay que saber qué necesita esa persona para poder decidir mejor.

¿Qué necesitamos cambiar?

- Salir del modo *"output"*: dejar de pensar que entregar el archivo es terminar el trabajo.
- Construir análisis situados: entender a quién le hablamos y qué preguntas está tratando de responder.
- Acompañar la entrega con explicaciones: sin sobreinformar, pero sin soltar el dato sin contexto.
- Validar el impacto: ¿Tomaron decisiones con esto? ¿Sirvió para algo? ¿Entendieron lo importante?

El dato es conversación, no dictado

Trabajar con datos es también trabajar con personas.
Si no afinamos esa escucha, esa mirada estratégica, nos convertimos en emisores sordos: soltamos *outputs* sin verificar si alguien los puede usar.

Pensar con datos no es solo limpiar, agrupar o visualizar. Es imaginar qué necesitan los *stakeholders* para decidir mejor, para impulsar una acción, y asegurarnos de que la información llegue viva, no muerta en un *dashboard* que a nadie le interesa ver.

12. MAS ALLÁ DEL CÓDIGO: FORMAR CRITERIO

Y vamos cerrando…

“Hice lo que decía el enunciado.”
“Yo no decido, yo solo ejecuto.”
“Le pasé los datos y listo.”
“Ese dato estaba, si no lo usaron no es mi problema.”

Ese es el momento en el que el pensamiento crítico se fue del grupo.

Vimos que saber usar datos no es solo saber limpiarlos, consultarlos o visualizarlos. Es saber interpretarlos. Y eso no se aprende en la sintaxis de un lenguaje: se construye con criterio.

El criterio no viene en el paquete de instalación

Podemos tener las herramientas más sofisticadas, el código más limpio, el *pipeline* más elegante… y aun así no

entender qué estamos viendo.
¿Por qué? Porque el criterio no se automatiza. Se entrena.

Y no se entrena repitiendo fórmulas, sino haciéndonos preguntas incómodas:

- ¿Por qué esta métrica y no otra?
- ¿Qué está dejando afuera este modelo?
- ¿Qué implica esta decisión para los actores involucrados?
- ¿Qué consecuencias tiene lo que vamos a mostrar (o lo que decidimos no mostrar)?

Pensar con datos es pensar con contexto

No analizamos en el vacío.
El análisis ocurre en instituciones, en equipos, en conflictos, en decisiones que impactan a personas reales.
Si no entendemos ese contexto, lo que hagamos con datos va a ser técnicamente correcto pero éticamente miope.

Hay decisiones que parecen neutras: un filtro, un umbral, un promedio.
Pero cada una de ellas puede cambiar la historia que contamos, el problema que visibilizamos, o el actor que queda afuera.

Del ejecutor al analista

Tomar decisiones con datos es una responsabilidad.
Implica elegir qué mostrar y cómo mostrarlo.
Implica anticipar cómo puede ser leído, usado o incluso malinterpretado.

Por eso, lo que separa al técnico del analista no es el conocimiento de herramientas, sino la capacidad de pensar con criterio.

El técnico ejecuta una instrucción.
El analista la pone en duda, la adapta, la mejora.
No se conforma con saber *"cómo se hace"*. Quiere entender *"por qué se hace"*, *"para qué sirve"* y *"a quién afecta"*.

Mapa de transiciones

Hay un punto en el que pasamos de "saber hacer" a *"saber decidir".*
Este mapa no es una tabla de conversión. Es una brújula para saber si nos estamos moviendo en la dirección adecuada:

De...	A...
Limpiar datos sin criterio	Entender qué se pierde al limpiar
Consultar sin contexto	Analizar lo que se pidió y lo que se omitió
Automatizar sin pensar	Interrogar el proceso automatizado
Medir por medir	Evaluar qué métricas importan realmente
Mostrar resultados planos	Comunicar lo que otros necesitan ver
Pedir datos genéricamente	Formular preguntas situadas
Ejecutar sin cuestionar	Analizar con criterio

Cada paso no es un requisito técnico. Es un cambio de mentalidad.

Formar criterio es aprender a sostener preguntas

Ya vimos qué pasa cuando limpiamos sin pensar, consultamos sin criterio o visualizamos sin contexto. Ahora es momento de ir más allá: sostener el análisis cuando no hay una única respuesta.

Podemos escribir funciones complejas, armar *dashboards* impecables, correr modelos avanzados…

Pero si no entendemos el fenómeno, si no sabemos explicar por qué tomamos cada decisión y no entendemos las

consecuencias de nuestras omisiones, seguimos jugando en la superficie mientras el problema real sigue intacto.

Formar criterio es aceptar que no siempre hay una única respuesta.
Es sostener el análisis incluso cuando no sabemos si estamos cerca de una conclusión.
Es priorizar la comprensión por sobre la comodidad.

Checklist de criterio

Algunas preguntas mínimas para no perder el eje cuando trabajamos con datos:

- ☐ ¿Entiendo de dónde vienen estos datos?
- ☐ ¿Tengo claridad sobre para qué estoy haciendo este análisis?
- ☐ ¿Sé a quién afecta lo que voy a mostrar?
- ☐ ¿Qué se queda afuera con las decisiones que tomé?
- ☐ ¿Puedo explicar por qué elegí este corte, este filtro o esta visualización?
- ☐ ¿Podría haber interpretaciones equivocadas de lo que hice?
- ☐ ¿Estoy priorizando la prolijidad técnica o la comprensión del fenómeno?
- ☐ ¿Qué tipo de acción permite o limita este análisis?

No hace falta tener todas las respuestas. Pero hace falta al menos hacerse las preguntas.

Casos mínimos para pensar

No necesitamos más datasets. Necesitamos ojos entrenados para leerlos con criterio y coraje para cuestionar lo que devuelven.

Estos mini casos no tienen solución única. Sirven para entrenar el músculo más importante: el que duda.

Caso A
Te piden medir “la eficacia de un programa de formación técnica”.
Tenés datos de asistencia, finalización y encuestas de satisfacción.
La tasa de finalización es del 82%.
La satisfacción es de 3,9/5.
Pero en los comentarios, 1 de cada 4 menciona problemas de accesibilidad.

¿Qué hacés con eso? ¿Qué mostrás? ¿Qué dejás en el informe?

Caso B

Tu modelo predice la deserción en estudiantes.

Funciona bien, pero marca como “alto riesgo” a todos los mayores de 35 años.

¿Mejorás el modelo? ¿Lo ajustás? ¿Advertís que eso es un sesgo?

¿Qué harías si tu informe impacta en el ingreso de esas personas?

Caso C

Un *dashboard* muestra que la satisfacción bajó 12 puntos en un mes.

Tu equipo quiere reportar “crisis de experiencia de usuario”.

Pero hubo una campaña masiva que incorporó nuevos usuarios con menos conocimiento.

¿El dato refleja un problema real o una transición esperable?

¿Qué tipo de análisis necesitás para responder con honestidad?

Y entonces… ¿qué significa tomar buenas decisiones con datos?

Significa mirar más allá del código y del dato.

Significa que lo que hacemos tiene consecuencias.

Significa asumir que interpretar datos es un acto técnico,

sí… pero también ético, estratégico y profundamente humano.

Y eso —justamente eso— no lo resuelve ningún algoritmo.

Mini manifiesto final

Un buen análisis no se mide por la prolijidad técnica. Se nota cuando sale a la luz un problema que antes era invisible, cuando una decisión se vuelve más justa, cuando algo que estaba oculto se ilumina.

Cuando una tabla deja de ser una fila y se convierte en una historia y no dice cómo podemos mejorarla.

Eso no lo hace ningún modelo, ninguna query, ningún dashboard.

Eso lo hacemos nosotros, con criterio y la valentía de pensar más allá del código.

Este no es el final del análisis. Es el punto en que dejamos de ejecutar y empezamos a pensar.

EPÍLOGO
Lo que no podemos automatizar, tal vez valga más

“Eso no se puede automatizar, así que no lo hago.”
“Si no hay un método claro, es pérdida de tiempo.”
“Lo ambiguo me rompe todo.”
“Quiero reglas. Si no hay reglas, no se puede analizar.”

Lo escuchamos todo el tiempo. Como si el único dato valioso fuera el que se puede medir, codificar, automatizar y ejecutar en bucle.

Pero hay algo que este libro repitió una y otra vez (sí, con obstinación):
Lo importante rara vez está en lo predecible. Muchas veces vive en lo que no encaja, en lo que incomoda, en lo que ningún modelo sabe procesar.

No todo puede convertirse en fórmula. Y eso no es un problema. Es una invitación.

Una invitación a explorar lo irrepetible. A leer lo ambiguo. A mirar lo que no entra en ninguna tabla.
Una invitación a pensar, aunque no haya una regla. A sostener la pregunta, aunque no venga con una métrica.

La sensibilidad analítica no está en el código. Está en la cabeza.
Automatizar es útil. Nos da velocidad, consistencia, escalabilidad.
Pero pensar es otra cosa.
Pensar es saber cuándo no automatizar.
Saber cuándo un loop no alcanza.
Saber que a veces, lo más valioso no es lo que se resuelve… sino lo que se comprende.

Receta para arruinar un análisis

1. Tomá un dataset sin contexto.
2. Aplicale un modelo que no entendés.
3. Hacé un gráfico llamativo sin revisar las categorías.
4. Presentalo en una reunión sin saber quién lo va a mirar.
5. Terminá diciendo: *“no lo digo yo, lo dicen los datos.”*

Y listo:

Acabamos de hacer todo lo que este libro intenta evitar.

Con eficiencia impecable, con prolijidad absurda… y con cero criterio.

Una última cosa para cerrar

Todo lo que viste en este libro —las dudas, las ambigüedades, los errores, los *outliers*, los datos mal cargados, las categorías mal pensadas, las visualizaciones innecesarias, las consultas mal planteadas— no son obstáculos.

Son la materia prima del pensamiento analítico.
Lo que parecía ruido… era señal.
Lo que parecía desorden… era una historia.

Si llegaste hasta acá, no necesitás un *framework* nuevo.
Solo seguir entrenando la mirada.
Porque trabajar con datos no es repetir recetas.
Es tener criterio.
Y eso, por suerte, no lo puede automatizar ningún algoritmo. Solo nosotros podemos hacerlo, y ahí está el verdadero valor del análisis.

NOTA FINAL — VERSIÓN 1.0

Este libro no pretende enseñarte a programar ni a usar una herramienta. Si te resultó obvio en algunas partes, incómodo en otras, o reiterativo en más de una, entonces funcionó.

Porque no vinimos a hacer un manual: vinimos a sacudir hábitos, romper automatismos y desafiar esas ideas que seguimos usando sin pensar demasiado por qué.

En esta primera edición decidimos dejar abierta la puerta a la duda. Algunas ideas se repiten porque todavía cuesta procesarlas. Algunos ejemplos se estiran porque nos educaron para priorizar la técnica, no el criterio. Algunas partes son más básicas, porque no todos venimos del mismo lado. Y otras son más filosas, porque hay que pellizcar para que algo despierte.

Esto es una versión 1.0. Imperfecta a propósito. Deliberada. Necesaria para empezar a mover el tablero.

Más adelante vendrá una revisión. Tal vez una nueva edición. Una que te anime a ir más hondo en la fusión entre lo técnico y lo conceptual. Que explore mejor cómo se entrelazan los modelos, los contextos, y las decisiones. Que incomode distinto. Tal vez una versión más formal.

Pero esta es la que había que escribir primero.

Nos vemos en la próxima versión.

PARA SEGUIR EN CONTACTO

Este libro fue escrito y distribuido de forma libre, con el deseo de compartir una mirada distinta sobre los datos y el análisis.

Si te resultó útil, te inspiró o te acompañó en alguna transformación profesional, podés ayudarme a seguir creando contenidos abiertos con una colaboración voluntaria.

Alias MercadoPago: webchick73.mp

📍 LinkedIn: linkedin.com/in/regina-molares

📊 Kaggle: kaggle.com/dataregina

📬 Contacto directo: data.regina.cursos@gmail.com

Gracias por leer. Y sobre todo, por pensar.

Este libro no enseña herramientas. Enseña a mirar distinto.

Como desarrolladores, solemos pensar en los datos como insumos para que el código funcione. Pero los datos no solo se ejecutan. Se interpretan.

A lo largo de este recorrido, revisamos los sesgos técnicos que dificultan el análisis: la obsesión por lo automatizable, la subestimación del contexto, la ilusión de limpieza estructural.

El análisis de datos no es una extensión de la programación. Es otro paradigma.

Proponemos pensar con criterio antes de operar con librerías. Leer patrones antes de graficarlos. Entender que los condicionales no solo bifurcan programas, también crean significado.

Este libro es una intervención quirúrgica al mindset técnico. Como toda cirugía, puede incomodar, pero si logramos mirar más allá del código, tal vez encontremos una nueva forma de pensar con datos.

DATA REGINA
FORMACIÓN PARA MIPYMES