

Trading retail

Es la operativa en mercados financieros realizada por personas individuales (no instituciones), usando su propio capital y plataformas públicas ofrecidas por brokers.

En otras palabras:

👉 trading hecho por traders “de a pie”, no por bancos, fondos o mesas profesionales.

EA

Un **Expert Advisor** es un **programa automático de trading** que corre dentro de **MetaTrader 4 / MetaTrader 5** y que puede:

- 📈 analizar el mercado
- 💡 aplicar una estrategia (reglas o señales)
- 🔍 **abrir, modificar y cerrar operaciones automáticamente**
- ⚠ gestionar riesgo (stop loss, take profit, trailing, etc.)

En resumen:

👉 **Un EA es un “trader automático” programado en MQL.**

Broker API

Una **API (Application Programming Interface)** es una interfaz que permite que **tu programa (Python)**:

- consulte precios en tiempo (casi) real,
- vea tu cuenta (balance, posiciones),
- envíe órdenes de compra/venta,
- gestione órdenes abiertas,

sin usar una plataforma gráfica como MetaTrader.

Binance

Qué es

- Un **exchange de criptomonedas** (Bitcoin, Ethereum, etc.).
- No es un broker tradicional de Forex/acciones.

Con Python puedes:

- Descargar datos históricos y en tiempo real
- Enviar órdenes (market, limit, stop)
- Gestionar posiciones y balances

Se usa para:

- **Trading algorítmico en crypto**
- Bots 24/7

Pros / Contras

- API muy completa
- Ideal para automatización
- Alta volatilidad
- Riesgo de cambios regulatorios

Interactive Brokers

Qué es

- Broker **institucional/semiprofesional**.
- Acceso a acciones, ETFs, opciones, futuros, Forex.

Con Python puedes:

- Operar directamente desde scripts
- Usar datos de mercado profesionales
- Integrar ML en producción

Se usa para:

- Trading cuantitativo serio

- Portafolios multi-activo

Pros / Contras

- Muy profesional
- Costos bajos
- API más compleja
- Curva de aprendizaje fuerte

Alpaca

Qué es

- Broker **orientado a desarrolladores.**
- Principalmente acciones y ETFs de EE. UU.

Con Python puedes:

- Hacer trading algorítmico fácilmente
- Paper trading sin dinero real

Se usa para:

- Aprender y prototipar
- Trading algorítmico en acciones

Pros / Contras

- API simple y bien documentada
- Paper trading gratuito
- Solo mercados específicos (EE. UU.)

OANDA

Qué es

- Broker especializado en **Forex**.
- API bien documentada.

Con Python puedes:

- Operar Forex sin MetaTrader
- Acceder a datos históricos y en vivo

Se usa para:

- Trading algorítmico en Forex vía API

Pros / Contras

-  Ideal para Forex
-  API clara
-  Menos popular que MT en retail

Resumen rápido

Plataforma	Mercado	Ideal para
Binance	Crypto	Bots 24/7
Interactive Brokers	Multi-activo	Trading cuantitativo serio
Alpaca	Acciones USA	Aprender + paper trading
OANDA	Forex	Forex sin MetaTrader

¿Cuál encaja contigo ahora?

Por todo lo que has dicho antes:

- Estás empezando **con Forex**
- Estás aprendiendo **pipeline ML**
- Luego quieres pasar a **MetaTrader**

 **OANDA o MT5** son los más coherentes.

Pero:

- para **aprender ejecución por API**, Alpaca es excelente,
- para **nivel pro**, Interactive Brokers.

Slippage

Slippage es la diferencia entre el precio al que esperas ejecutar una orden y el precio al que *realmente* se ejecuta.

Dicho simple:

👉 querías comprar/vender a un precio, pero te ejecutan a otro.

Spread

Diferencia entre el bid – ask (siempre existe)

Ordenes Limit

Las **órdenes limit** son un tipo de orden en la que **tú fijas el precio máximo (o mínimo) al que aceptas operar**.

Si el mercado **no llega a ese precio, la operación no se ejecuta**.

📌 Definición corta

Orden LIMIT = “operar solo a este precio o mejor”

A diferencia de una orden a mercado, **no aceptas cualquier precio**.

🧠 Cómo funcionan (BUY vs SELL)

🟢 BUY LIMIT (comprar más barato)

- Precio actual: 1.1000
- Colocas: **BUY LIMIT** en 1.0980

👉 Solo compras si el precio baja a 1.0980 o mejor

👉 Si no baja → no hay trade

🔴 SELL LIMIT (vender más caro)

- Precio actual: 1.1000
- Colocas: SELL LIMIT en 1.1020

👉 Solo vendes si el precio sube a 1.1020 o mejor

👉 Si no sube → no hay trade

Orden market

Una **orden market** (orden a mercado) es una orden en la que **le dices al broker que ejecute la operación inmediatamente al mejor precio disponible en ese instante.**

📌 Definición simple

Orden MARKET = “entra o sal ya, al precio que haya”

No fijas el precio exacto.

Priorizas **velocidad, no precio.**

⚖️ LIMIT vs MARKET (comparación clara)

Característica	MARKET	LIMIT
Precio	No garantizado	Garantizado (o mejor)
Ejecución	Inmediata	Solo si toca el precio
Slippage	Sí	✗ No (en la entrada)
Riesgo de no entrar	✗ No	✓ Sí

Latencia

Latencia es el tiempo de retraso entre una acción y su efecto. En trading, es el tiempo que tarda una orden en ir desde donde se genera hasta donde se ejecuta.



Definición simple

Latencia = retraso (en milisegundos)

En trading algorítmico:

desde que decides operar → hasta que el broker ejecuta la orden



Dónde aparece la latencia (paso a paso)

Cuando haces un BUY, ocurre esto:

```
Python / EA  
→ MT5  
→ Internet  
→ Servidor del broker  
→ Proveedor de liquidez  
→ Ejecución  
→ Respuesta de vuelta
```

Backtests

Un **backtest** (o **backtesting**) es probar una estrategia de trading usando datos históricos, como si la hubieras ejecutado en el pasado, para ver cómo habría funcionado.



Definición simple

Backtest = “simular el pasado para evaluar una estrategia”

No es predecir el futuro.

Es evaluar si tu idea tenía (o no) ventaja histórica.

BUY (Comprar)

Significa:

“Espero que el precio suba desde aquí.”

En la práctica:

- abres una **posición larga**
- ganas si el precio sube
- pierdes si baja

Ejemplo:

- compras EUR/USD
- compras barato para vender más caro después

SELL (Vender)

Significa:

“Espero que el precio baje desde aquí.”

En la práctica:

- abres una **posición corta**
- ganas si el precio baja
- pierdes si sube

Ejemplo:

- vendes EUR/USD
- vendes caro para recomprar más barato

HOLD (No hacer nada)

Significa:

“No tengo ventaja clara ahora.”

En la práctica:

- no abres posición
- conservas capital

- evitas operar en condiciones desfavorables

 **HOLD** también es una decisión.

Probabilidad

En trading con ML, la **probabilidad** es:

una estimación numérica de qué tan probable es un escenario futuro,

basada en datos históricos y patrones aprendidos.

Ejemplo típico:

$$P(\text{up} \mid X) = 0.72$$

Significa:

- dadas las condiciones actuales del mercado (X)
- hay un **72% de probabilidad estimada** de que el precio suba en el horizonte definido

 **Importante:**

- no es certeza
- puede fallar
- se valida **en promedio**, no en un solo trade

Stop / Take

Son **niveles de salida** que controlan el riesgo y el beneficio.

Stop Loss (Stop)

Es:

el precio donde aceptas que estabas equivocado.

En la práctica:

- limita la pérdida
- protege tu capital
- se ejecuta automáticamente

Ejemplo:

- BUY en 1.1000
 - Stop en 1.0975
- pierdes 25 pips y sales

Take Profit (Take)

Es:

el precio donde decides tomar ganancias.

En la práctica:

- asegura beneficios
- evita devolver ganancias al mercado
- se ejecuta automáticamente

Ejemplo:

- BUY en 1.1000
 - Take en 1.1050
- ganas 50 pips y sales

Cómo se relacionan entre sí (clave)



Cómo se relacionan entre sí (clave)

En una operación completa siempre tienes:

1. **Acción** → BUY / SELL / HOLD
2. **Confianza** → Probabilidad
3. **Riesgo** → Stop
4. **Objetivo** → Take

El ML suele ayudar con (2).

La estrategia define (1), (3) y (4).

Look-ahead

Look-ahead (o *look-ahead bias*) es un **error grave** en trading y backtesting que ocurre cuando **usas información del futuro para tomar una decisión en el pasado**, aunque sea sin darte cuenta.

Definición clara

Look-ahead = usar datos que todavía no existían en el momento de la decisión

Eso hace que:

- el backtest se vea **mucho mejor de lo real**
- el modelo parezca “genial”
- en real **falle**

Concept drift

Concept drift es cuando **la relación entre las variables del mercado y el resultado cambia con el tiempo**.

Dicho simple:

El mercado deja de comportarse como cuando entrenaste tu modelo.

Tu modelo no “se rompe” por estar mal hecho, se vuelve **obsoleto** porque **la realidad cambió**.

Gating

Gating es un **mecanismo de control** que decide **cuándo una señal o un modelo puede operar y cuándo NO**, aunque el modelo diga BUY o SELL.

Dicho simple:

Gating = una “puerta” que deja pasar trades solo si el contexto es adecuado.

Walk-forward

Walk-forward es una **técnica de validación** para trading/ML que **simula cómo usarías el modelo en el tiempo real**, reentrenándolo y probándolo **en ventanas temporales consecutivas**, sin mirar el futuro.

Dicho simple:

Entrenas en el pasado → pruebas en el futuro inmediato → avanzas la ventana → repites.

Timing

En trading, **timing** es **el momento exacto en el que decides actuar: cuándo entrar, cuándo salir o cuándo no hacer nada**.

Dicho simple:

Timing = “hacer lo correcto en el momento correcto”.

Bias

Bias es un **sesgo**: una **distorsión sistemática** que hace que tus resultados, decisiones o modelos **parezcan mejores (o peores) de lo que realmente son**.

En trading y ML, el bias **no es ruido**, es un **error estructural**.

Look-ahead bias

Look-ahead bias es un **sesgo crítico** que ocurre cuando **usas información del futuro para tomar decisiones en el pasado**, aunque sea sin querer. En trading y ML, **falsea los resultados** y hace que un sistema parezca rentable cuando no lo es.

➡ Definición clara

Look-ahead bias = el modelo “sabe” algo que en tiempo real aún no existía.

Eso **infla métricas, mejora backtests falsamente y rompe en real**.

Rolling Windows

Rolling windows (ventanas móviles) es una **técnica para entrenar, evaluar o calcular estadísticas usando solo los datos más recientes**, desplazando la ventana en el tiempo y **olvidando el pasado lejano**.

Dicho simple:

Siempre miras los últimos N datos, avanzas en el tiempo y vuelves a calcular.



Definición clara

Una **rolling window** toma un tramo fijo del tiempo (por ejemplo, últimos 6 meses) y:

- entrena el modelo **solo con ese tramo**
- prueba/decide en el siguiente tramo
- avanza la ventana
- repite

Es lo opuesto a usar “todo el histórico desde siempre”.

Tick/bar

En trading, **tick** y **bar** (o **vela / candle**) son **dos niveles distintos de información de precio**. Entender la diferencia es clave para timing, backtesting y ejecución.



Tick

Un **tick** es **cada cambio individual de precio**.

- Es el dato **más granular**
- Ocurre **cuando hay una actualización de precio**
- No tiene duración fija

- Puede haber **muchísimos ticks por segundo** en mercados líquidos

Ejemplo (EUR/USD):

```
1.10001 → 1.10002 → 1.10000 → 1.10003
```

Cada cambio = **un tick**.

● **Bar (vela / candle)**

Una **bar** agrupa muchos ticks **dentro de un intervalo de tiempo fijo**.

Cada bar tiene:

- **Open** (precio inicial)
- **High** (máximo)
- **Low** (mínimo)
- **Close** (precio final)

Ejemplo: bar H1

- Agrupa **todos los ticks de 1 hora**
- Resume el movimiento de esa hora

Se usa para:

- análisis técnico
- ML con timeframes
- estrategias 1H, 4H, Daily
- backtesting clásico

Tick vs Bar (comparación clara)

Aspecto	Tick	Bar
Qué es	Cambio individual de precio	Resumen OHLC
Frecuencia	Variable, muy alta	Fija (1m, 1H, etc.)
Ruido	Muy alto	Menor
Datos	Muchísimos	Mucho menos
Uso típico	Ejecución fina	Decisión estratégica

Shift

En trading y ML, **shift** significa **desplazar los datos en el tiempo** (hacia atrás o hacia adelante) para **alinear correctamente pasado, presente y futuro.**

Dicho simple:

Shift = “mover una serie temporal X pasos en el tiempo”.

Es una operación **fundamental** para:

- evitar **look-ahead bias**
- crear **features**
- definir **targets**
- backtestear correctamente

Ejemplo con precios de cierre:

Tiempo	Close
t0	100
t1	102
t2	101

```
Close.shift(1)
```

Tiempo	Valor
t0	NaN
t1	100
t2	102

👉 En `t1` ahora ves el cierre anterior.

🧠 Para qué se usa shift (casos clave)

1 Evitar look-ahead (el uso más importante)

Cuando decides en `t`, **solo puedes usar información hasta `t-1`**.

Ejemplo correcto:

```
df["close_lag1"] = df["close"].shift(1)
```

✗ Usar `close` sin `shift` es mirar el presente

✗ Usar `shift(-1)` es mirar el futuro

2 Crear features (lags)

Muy común en ML:

```
df["ret_1"] = df["close"].pct_change().shift(1)
df["ret_5"] = df["close"].pct_change(5).shift(1)
```

3 Definir el target (esto es CLAVE)

Ejemplo:

```
df["target"] = (df["close"].shift(-1) > df["close"]).astype(int)
```

Aquí:

- el **target** usa el futuro (correcto)
- las **features NO**

👉 El futuro **solo puede estar en Y**, nunca en X.

4 Backtesting realista

Cuando simulas:

- señal en t
- ejecución en t+1

Necesitas shift para alinear todo.

Intrabar

Intrabar se refiere a **todo lo que ocurre dentro de una vela (bar)**, antes de que esa vela cierre.

Dicho simple:

Intrabar = el comportamiento del precio dentro del intervalo de una vela.

📌 Definición clara

Si tienes una **vela H1** (1 hora):

- la vela se forma con **muchos ticks**
- esos ticks hacen subir y bajar el precio
- **todo ese movimiento interno es intrabar**

La vela solo te muestra el **resumen final** (OHLC), pero **no el camino que siguió el precio**.

Retraining programado

Retraining programado es **volver a entrenar tu modelo de forma automática y periódica**, usando datos nuevos, para que **no se quede obsoleto cuando el mercado cambia** (concept drift).

Dicho simple:

Es enseñarle al modelo “otra vez” cada cierto tiempo, con información reciente.

Definición clara

En trading algorítmico:

Retraining programado = reentrenar el modelo según un calendario o una regla,

no “cuando te acuerdas”.

Target

En trading y Machine Learning, el **target** es la variable objetivo: **lo que quieras que el modelo aprenda a predecir.**

Dicho simple:

Target = la respuesta correcta que el modelo intenta anticipar.

Definición clara

En ML:

- **X (features)** → información que le das al modelo
- **Y (target)** → lo que quieras que el modelo prediga

En trading, el target suele representar:

- si el precio **sube o baja**
- el **retorno futuro**
- si se alcanza **TP o SL**
- una **clase** (BUY / SELL / HOLD)

Baseline

En trading y ML, un **baseline** es un punto de referencia mínimo contra el cual comparas tu modelo o estrategia para saber **si realmente aporta valor.**

Dicho simple:

Baseline = “lo más simple que debería poder superar tu modelo”.

Si no lo supera, **tu modelo no sirve**, aunque sea sofisticado.

📌 Definición clara

Un baseline puede ser:

- una regla muy simple
- un modelo trivial
- una estrategia ingenua

Sirve para responder:

¿Mi modelo es mejor que no hacer nada (o que algo muy básico)?

Split temporal (80/20) + baseline y Gradient Boosting

Esa frase describe **un pipeline estándar y correcto de ML para series temporales (trading)**.

Vamos **parte por parte**, y luego lo juntito todo.

1 Split temporal (80/20)

Qué significa

Dividir los datos **por tiempo**, no al azar:

- **80% inicial** → entrenamiento (train)
- **20% final** → prueba (test)

Ejemplo con datos 2018–2024:



📌 Regla clave:

- el modelo **solo ve el pasado**
- se prueba en el **futuro**

👉 Esto **evita look-ahead bias**.

Lo que NO es

-  NO es `train_test_split(shuffle=True)`
-  NO mezcla fechas

En trading, **mezclar datos rompe todo.**

Baseline

Qué significa aquí

Antes de usar ML avanzado, defines **un modelo simple de referencia**.

Ejemplos de baseline:

- “siempre HOLD”
- “siempre BUY”
- “BUY si close > MA(20)”
- modelo ML trivial (logistic simple)

 El objetivo es responder:

¿Gradient Boosting realmente mejora algo?

Si no supera el baseline:

- no hay señal
- no sigues

Gradient Boosting

Qué es

Un **algoritmo de ML potente** que:

- combina muchos árboles pequeños
- corrige errores iterativamente
- capta relaciones no lineales

Ejemplos conocidos:

- `GradientBoostingClassifier`
- `XGBoost`
- `LightGBM`

- CatBoost

En trading se usa porque:

- maneja bien features técnicas
- no necesita escalado
- es robusto a ruido
- suele funcionar mejor que modelos lineales

4 Todo junto (la frase completa)

“Split temporal (80/20) + baseline y Gradient Boosting”

Significa:

1. Divido los datos por tiempo (80% pasado / 20% futuro)
2. Entreno y evalúo un **baseline**
3. Entreno y evalúo un **modelo Gradient Boosting**
4. Comparo ambos **en el mismo periodo de test**
5. Solo continúo si Gradient Boosting **superá claramente** al baseline

5 Ejemplo mental completo (trading 1H)

- Datos: EUR/USD 1H
- Target: dirección próxima vela
- Split:
 - 2019–2022 → train
 - 2023–2024 → test

Baseline

- Regla simple o modelo trivial
- PF = 1.05

Gradient Boosting

- PF = 1.25
- drawdown menor

- estabilidad mejor

👉 **Conclusión:** el ML aporta valor

Si fuera al revés:

👉 descartas el modelo

6 Por qué esta combinación es “buena práctica”

Porque:

- respeta el tiempo (split temporal)
- evita autoengaño (baseline)
- usa un modelo potente pero razonable (GB)

Es **la forma correcta de empezar**, sin sobreingeniería.

🧠 **En una frase**

“Entreno con pasado, comparo contra algo simple, y solo uso ML avanzado si realmente aporta ventaja.”

Métrica CAGR aprox

La **métrica CAGR (aprox)** es una forma **simple y estándar** de medir **cuánto crece tu capital por año, como si el crecimiento fuera constante**, aunque en la realidad no lo sea.

📌 **¿Qué es CAGR?**

CAGR = Compound Annual Growth Rate

(en español: **tasa de crecimiento anual compuesta**)

Dicho simple:

“Si mi capital hubiera crecido a una tasa fija cada año, ¿cuál sería esa tasa?”



Fórmula clásica

$$\text{CAGR} = \left(\frac{V_f}{V_i} \right)^{\frac{1}{n}} - 1$$

Donde:

- V_i = capital inicial
- V_f = capital final
- n = número de años

Métrica Max Drawdown

La **métrica Max Drawdown (MDD)** mide la peor caída máxima del capital desde un pico hasta el valle siguiente, antes de que se recupere.

Dicho simple:

Max Drawdown = la peor racha de pérdidas que habrías sufrido.



Definición clara

El **drawdown** es:

- la **caída desde un máximo histórico (peak)**
- hasta el **mínimo posterior (trough)**

El **Max Drawdown** es la mayor de esas caídas en todo el periodo.

Ejemplo sencillo

Supón este capital:

10000 → 12000 → 11000 → 13000 → 9000 → 14000

- Peak: 13 000
- Trough: 9 000

$$\text{Max Drawdown} = \frac{13000 - 9000}{13000} = 30.8\%$$

👉 MDD ≈ -30.8%

Cómo se expresa

- normalmente en **porcentaje**
- a veces en **dinero absoluto**

Ejemplos:

- **-12%** (moderado)
- **-35%** (agresivo)
- **-60%** (muy peligroso)

⚠ Por qué es tan importante

Porque responde a:

“¿Cuánto puedo perder antes de recuperarme?”

Muchas estrategias:

- tienen buen CAGR
- pero drawdowns **psicológicamente imposibles**

👉 El MDD define si una estrategia es **vivable**.

Métrica Sharpe aprox

La **métrica Sharpe (aprox)** mide **cuánta rentabilidad obtienes por cada unidad de riesgo que asumes**. Es una de las métricas más usadas para comparar estrategias **ajustadas por riesgo**.

Dicho simple:

Sharpe = “¿me pagan bien por el riesgo que estoy tomando?”



Definición clara

El **Sharpe Ratio** compara:

- el **retorno promedio** de la estrategia
- contra su **volatilidad** (riesgo)

Cuanto **más alto, mejor**.



Fórmula (versión práctica)

$$\text{Sharpe} \approx \frac{R - R_f}{\sigma}$$

Donde:

- R = retorno promedio (anualizado o del periodo)
- R_f = retorno libre de riesgo (a veces se asume 0 en trading)
- σ = desviación estándar de los retornos (volatilidad)

👉 En trading algorítmico suele usarse **Sharpe aprox** con $R_f = 0$.

💡 Ejemplo sencillo

Supón:

- retorno anual promedio = **12%**
- volatilidad anual = **8%**

$$\text{Sharpe} \approx 12\% / 8\% = 1.5$$

👉 Sharpe ≈ 1.5 (bueno)

📊 Cómo interpretar el Sharpe (guía práctica)

Sharpe	Interpretación
< 0.5	Malo
0.5 – 1.0	Débil
1.0 – 1.5	Aceptable
1.5 – 2.0	Bueno
> 2.0	Excelente (raro y sospechoso si es histórico largo)

📌 En trading realista:

- Sharpe > 1 ya es decente
- Sharpe > 1.5 es muy bueno

⚠ Por qué se dice “aprox”

Porque:

- asume retornos “normales”
- usa volatilidad como proxy de riesgo
- no distingue bien caídas grandes (colas)

En trading:

- los retornos **no son normales**
- hay **drawdowns**
- hay asimetrías

👉 Aun así, es **muy útil como comparación**.

✳ Relación con otras métricas

- **CAGR** → cuánto ganas
- **Max Drawdown** → cuánto puedes perder
- **Sharpe** → qué tan eficiente es ese riesgo

Una estrategia ideal:

- CAGR razonable
- drawdown controlado
- Sharpe > 1

✗ Errores comunes

- comparar Sharpe de periodos distintos
- ignorar costos
- usarlo como única métrica
- confiar en Sharpes muy altos en backtests cortos

🧠 En una frase

Sharpe no mide cuánto ganas.

Mide qué tan caro te cuesta ganar.

Threshold

Threshold (umbral) es **un valor límite que decides para tomar una acción.**

En trading y ML, sirve para **convertir números (probabilidades, scores, métricas)** en **decisiones concretas.**

Dicho simple:

Threshold = “a partir de aquí, actúo; antes de eso, no”.

📌 Definición clara

Un threshold responde a preguntas como:

- ¿Desde qué probabilidad compro?
- ¿Cuándo ignoro una señal?
- ¿Cuándo paro el sistema?

Es una **regla de decisión**, no un modelo.

Paper trading

Paper trading es **operar una estrategia en tiempo real pero con dinero ficticio**, usando precios reales del mercado, **sin riesgo financiero.**

Dicho simple:

Es “trading de mentira” con datos de verdad.

📌 Definición clara

En paper trading:

- el mercado es **real**
- las señales son **reales**
- la ejecución es **simulada**
- el dinero **no es real**

Sirve para **probar**, no para ganar.

Win rate

Win rate (tasa de acierto) es **el porcentaje de operaciones que terminan en ganancia** sobre el total de operaciones realizadas.

Definición clara

$$\text{Win Rate} = \frac{\text{nº de trades ganadores}}{\text{nº total de trades}} \times 100$$

Ejemplo:

- 100 trades
- 45 ganadores

 Win rate = 45%

Por qué el win rate NO lo es todo

Este es el punto más importante (y donde muchos se equivocan):

Un win rate bajo puede ser rentable.

Un win rate alto puede perder dinero.

Lo que importa es la **combinación** de:

- win rate
- tamaño de ganancias
- tamaño de pérdidas

Ejemplos claros

Win rate alto, estrategia mala

- Win rate: **80%**
- Ganas: +1
- Pierdes: -5

 Un par de pérdidas destruye todo.

● Win rate bajo, estrategia buena

- Win rate: **40%**
- Ganas: +3
- Pierdes: -1

👉 Ganas más de lo que pierdes.

✳ Relación con Expectancy (clave)

La esperanza matemática es:

$$E = (WR \times G) - ((1 - WR) \times P)$$

Donde:

- WR = win rate
- G = ganancia media
- P = pérdida media

Si $E > 0 \rightarrow$ la estrategia tiene ventaja.

🧠 En trading con ML

- ML puede mejorar **calidad** de señales
- No siempre mejora el win rate
- A veces:
 - reduce drawdown
 - mejora payoff
 - filtra trades malos

👉 Win rate no es el objetivo final.

⚠ Errores comunes

- obsesionarse con win rate > 50%

- descartar estrategias con 40–45%
 - comparar win rate sin mirar SL/TP
 - ignorar costos
-

En tu contexto

Con:

- ML
- thresholds
- SL / TP
- gating

 El win rate **es solo una métrica más**, no la meta.

Frase para recordar

No se gana por acertar mucho.

Se gana por acertar bien.