

TÉRMINOS Y DEFINICIONES

CONCEPTOS DE RESULTADOS / MODELOS

Ranking

👉 Ordenar activos, setups o señales **del mejor al peor** según un score del modelo.

Ejemplo:

“De 50 acciones, estas 5 tienen mejor expectativa”

Se usa para **elegir dónde operar**, no para decidir entradas exactas.

Clustering / HMM

👉 Técnicas para **detectar estados del mercado**, no direcciones.

- **Clustering**: agrupa datos en “regímenes” similares (tendencia, rango, alta vol...)
- **HMM (Hidden Markov Model)**: modelo que asume que el mercado cambia entre estados ocultos

Sirven para **filtros**, no para entrar directamente.

MÉTRICAS DE MODELO (NO dinero)

Accuracy

👉 Porcentaje de predicciones correctas.

Ejemplo:

“El modelo acierta 54 de cada 100”

⚠ En trading **no basta sola** (puedes acertar mucho y perder dinero).

AUC

👉 Mide qué tan bien el modelo **distingue buenos vs malos casos**, sin fijar un umbral.

- 0.5 → azar
- 0.55 → posible edge
- 0.6 → bastante bueno en trading

Muy usada en **clasificación con probabilidades**.

RMSE

👉 Error medio de predicción en modelos de **regresión** (forecast).

Cuanto más bajo, mejor.

Mide **qué tan lejos estás del valor real**.

BACKTEST Y RESULTADOS

Curvas de backtest

👉 Simulación histórica de cómo habría funcionado la estrategia.

Incluye:

- entradas
 - salidas
 - costos
 - capital
-

Equity curve

👉 Gráfica del capital a lo largo del tiempo.

Es la forma visual de ver:

- crecimiento
 - drawdowns
 - estabilidad
-

Estimación de costos / riesgo

👉 Evaluar si la estrategia **sobrevive a la realidad**.

Incluye:

- spread
 - slippage
 - drawdown
 - volatilidad
 - riesgo de ruina
-

Predicción + confianza (probabilidad) + condiciones

👉 La **tríada correcta para operar**.

- **Predicción**: lo que dice el modelo
- **Confianza**: probabilidad (p)
- **Condiciones**: filtros, riesgo, contexto

Si falta una → no operas.

CONCEPTOS DE TRADING CUANTITATIVO

Quant trading

👉 Trading basado en **datos, estadística y reglas**, no intuición.

Decisiones repetibles, medibles y automatizables.

Position sizing

👉 Decidir **cuánto capital** poner en cada trade.

No es lo mismo:

- entrar
 - que **cuánto arriesgar**
-

Cartera

👉 Conjunto de activos operados al mismo tiempo.

Ejemplo:

BTC + ETH + EURUSD + Oro

$P(\text{up})=0.63$ / retorno esperado = +0.2%

👉 Salida típica de un modelo:

- $P(\text{up})=0.63 \rightarrow$ probabilidad de subida
- +0.2% → magnitud esperada

Es **información**, no orden de trading.

GESTIÓN DE RIESGO

Definir gestión de riesgo

👉 Reglas para **no perder demasiado**, incluso con malas rachas.

Incluye todo lo que sigue ↓

Stop / Take / Trailing / Máx. Drawdown

- **Stop Loss:** pérdida máxima por trade
 - **Take Profit:** ganancia objetivo
 - **Trailing:** stop que se mueve a favor
 - **Max Drawdown:** caída máxima tolerada
-

Time stop

👉 Salir si pasa demasiado tiempo sin resultado.

Ejemplo:

“Cerrar después de 6 velas”

Riesgo por trade: 0.5%

👉 En cada trade solo puedes perder el 0.5% del capital.

Base de supervivencia.

SL/TP: ATR

👉 Stops y takes basados en **volatilidad** (ATR).

Más robusto que pips fijos.

Monitoreo: apagar si DD > 8%

👉 Regla de seguridad:

“Si el sistema cae más de 8%, se apaga”

VALIDACIÓN CORRECTA

Walk-forward

👉 Entrenas, pruebas, avanzas la ventana y repites.

Simula **uso real en el tiempo**.

Out-of-sample

👉 Datos que el modelo **nunca vio**.

Sirven para validar si hay edge real.

Paper trading

👉 Operar en simulado en tiempo real.

Prueba ejecución, no predicción.

Forward test

👉 Operar con reglas fijas, sin reentrenar.

Detecta problemas de implementación.

Rolling window

👉 Ventana de datos que se va desplazando con el tiempo.

Evita usar información futura.

Stress tests

👉 Pruebas extremas:

- más costos
 - menos liquidez
 - mercados malos
-

Control de drift

👉 Detectar cuando el mercado **cambia** y el modelo deja de funcionar.

DATOS Y FEATURES

BTCUSDT

👉 Par de trading: Bitcoin vs USDT.

Spread

👉 Diferencia entre precio de compra y venta.

Costo oculto.

Slippage

👉 Ejecución a peor precio del esperado.

Muy común en volatilidad.

Retorno futuro

👉 Cambio porcentual del precio en el futuro.

Es lo que suele predecir el modelo.

OHLCV + opcional

👉 Datos base:

- Open
 - High
 - Low
 - Close
 - Volume
-

Order book

👉 Profundidad del mercado (ofertas y demandas).

Más avanzado.

Macro

👉 Datos económicos: tasas, inflación, empleo, etc.

Indicadores

👉 Transformaciones técnicas (RSI, MACD, ATR, etc.).

Retornos

👉 Cambios porcentuales del precio.

Momentum

👉 Fuerza de la tendencia.

Data leakage

👉 Error grave: usar información del futuro en el entrenamiento.
Mata cualquier estrategia.

Split temporal / train–val–test (no shuffle)

👉 Separar datos **en orden de tiempo**, nunca mezclados.
Regla sagrada en trading.

Umbrales

👉 Valores que convierten probabilidades en acciones.
Ejemplo:
 $p > 0.58 \rightarrow$ entrar

Sizing

👉 Sinónimo práctico de position sizing.

MÉTRICAS DE TRADING

Profit factor

👉 Ganancias totales / pérdidas totales.

1.3 ya es interesante

Sharpe / Sortino

👉 Retorno ajustado por riesgo.

- Sharpe: volatilidad total
 - Sortino: solo riesgo negativo
-

Max drawdown

👉 Mayor caída histórica del capital.

% tiempo en mercado

👉 Qué tanto estás expuesto.

Menos tiempo ≠ peor sistema.

MODELOS / IMPLEMENTACIÓN

XGBoost

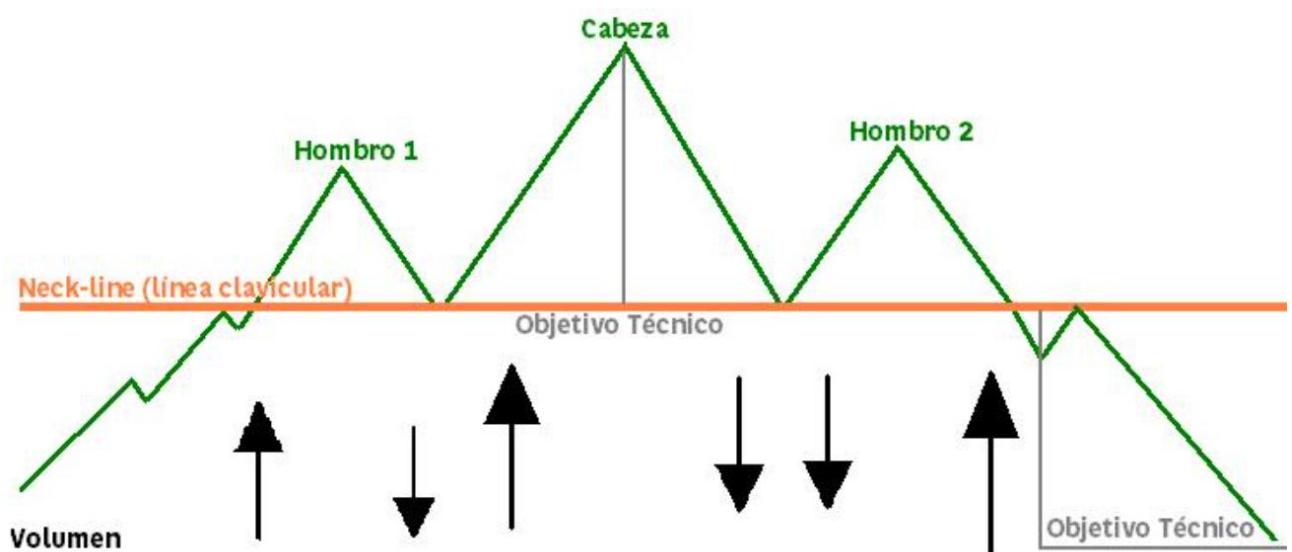
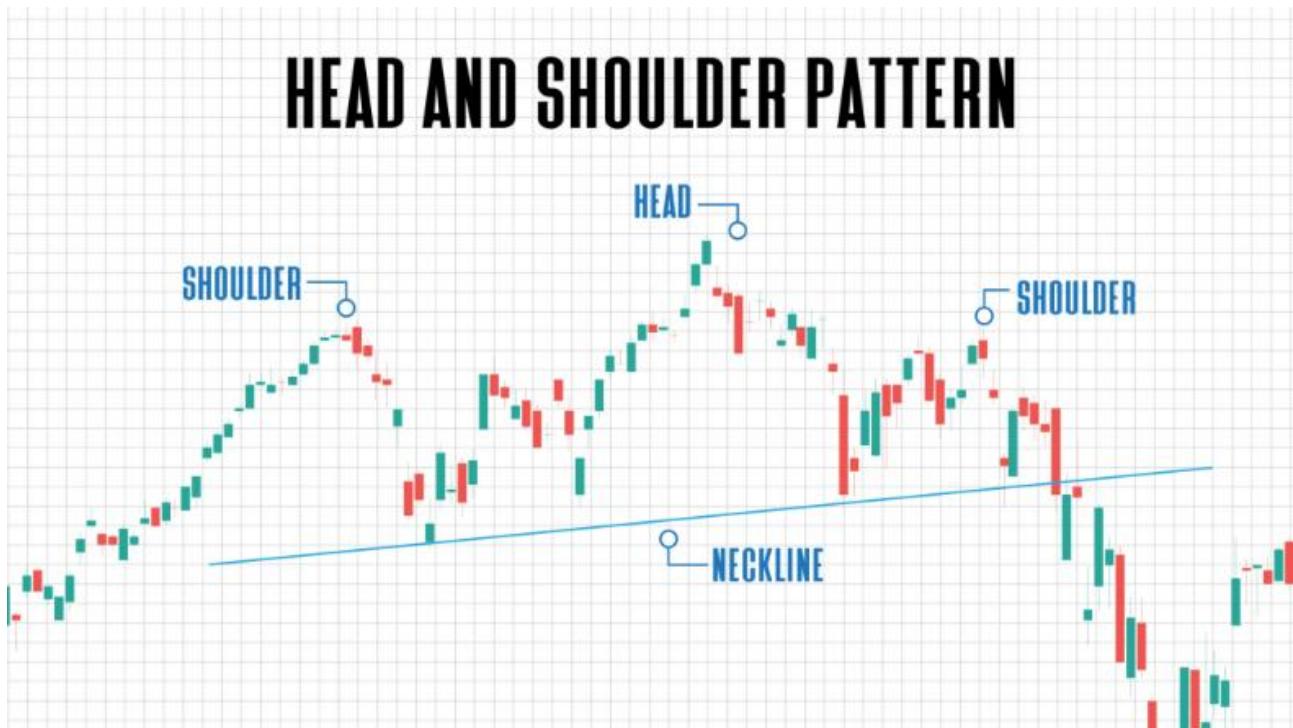
👉 Modelo de árboles muy potente para ML tabular.

Muy usado en trading cuantitativo.

ONNX

👉 Formato para exportar modelos y usarlos fuera de Python (por ejemplo, integrarlos con otros sistemas).

HCH (Hombro-Cabeza-Hombro)



¿Qué es?

El **HCH** (**Hombro-Cabeza-Hombro**) es un **patrón gráfico de cambio de tendencia**.

Cómo se ve (idea mental)

Tiene **tres picos**:

1. **Hombro izquierdo** → sube y baja
2. **Cabeza** → sube más alto y baja
3. **Hombro derecho** → sube menos y vuelve a bajar

Debajo suele trazarse una línea llamada **línea de cuello**.

Qué “significa” para un trader clásico

👉 “La tendencia alcista se está debilitando y puede venir una caída”

Cuando el precio **rompe la línea de cuello**, muchos traders:

- venden (SHORT)
- o cierran posiciones largas

Versión inversa

Existe el **HCH invertido**, que se usa como señal alcista.

📈 Doble Techo





¿Qué es?

El **doble techo** es otro **patrón de cambio de tendencia**, más simple que el HCH.

Cómo se ve

- El precio sube
- llega a un máximo
- baja
- vuelve a subir **hasta un nivel similar**
- no logra superarlo
- y cae

Tiene forma de “M”.

Qué “significa” para un trader clásico

👉 “El mercado intentó subir dos veces y falló → posible caída”

Cuando rompe el mínimo entre los dos picos:

- se considera confirmación bajista

¿Por qué existen estos conceptos?

Estos patrones vienen del **análisis técnico discrecional**, donde:

- el trader **mira el gráfico**
- interpreta formas
- toma decisiones “visuales”

Son muy antiguos y populares.

Punto MUY importante (conexión con lo que tú estás aprendiendo)

HCH y doble techo son:

- **X subjetivos**
- **X dependen del ojo del trader**
- **X difíciles de definir exactamente en código**
- **X inconsistentes entre personas**

Por eso:

👉 **Los traders cuantitativos y los sistemas con ML NO suelen usarlos directamente**

Relación con ML y trading cuantitativo

En lugar de decir:

“Veo un HCH”

Un trader cuantitativo diría:

“La probabilidad de continuación alcista, dada esta estructura de precios, es baja”

Es decir:

- el ML **no busca dibujos**
- busca **patrones estadísticos**, aunque a veces *indirectamente* capture cosas parecidas

Comparación rápida

Enfoque	Cómo decide
HCH / Doble techo	Formas en el gráfico
ML / cuantitativo	Probabilidades y datos

Resumen claro (para que te quede grabado)

- **HCH y doble techo son patrones gráficos clásicos**
- Se usan para **anticipar cambios de tendencia**
- Funcionan de forma **visual y subjetiva**
- **No son la base del trading cuantitativo moderno**
- El ML reemplaza “ver figuras” por **medir probabilidades**

CIERRE CLAVE

Todo esto **no se usa a la vez**.

Un buen sistema usa **lo justo y necesario**.

Para tu caso:

- ML → **clasificación**
- estrategia → riesgo
- análisis cuantitativo → validación

Siquieres, el próximo paso ideal es:

👉 **ordenar todo esto en un flujo mínimo y limpio**, solo con lo que realmente necesitas.

Dime y lo armamos paso a paso.

INTERPRETAR RESULTADOS

Todos los resultados válidos y reales que puede darte un modelo de ML en trading.

No son teorías distintas ni cosas incompatibles: son **formas diferentes de “empaquetar” la información** que el ML aprende del mercado.

Ahora te lo explico **de forma ordenada y coherente**, para que veas **qué es cada uno, qué objetivo tiene y cómo se relacionan entre sí**.

Cuando entrenas un modelo de ML **siempre obtienes información**, pero esa información puede responder **preguntas distintas**:

- ¿Cuánto?
- ¿En qué dirección?
- ¿Cuál es mejor?
- ¿En qué contexto estoy?

Eso es exactamente **A, B, C y D**.

Los resultados útiles de los ML son los siguientes:

A) Un pronóstico (forecast)

- Ej: predice el **retorno** del próximo día, o el **precio** en 1 hora.
- Salida típica: un número continuo (regresión).
 - Ej: $\hat{y} = +0.0025$ (retorno esperado +0.25%)

En realidad, este resultado me indica:

“¿Cuánto va a subir o bajar el precio (o el retorno)?”

Objetivo:

Estimar el valor futuro.

Qué resultado obtienes:

Un **número continuo** (regresión), por ejemplo:

- $+0.0025 \rightarrow$ retorno esperado $+0.25\%$
- $-0.0018 \rightarrow$ retorno esperado -0.18%

Cómo se usa

TÚ decides:

- Si el retorno esperado es suficientemente grande \rightarrow entras
- Si es pequeño \rightarrow no operas

Ejemplo real:

- “En la próxima vela de 1H, el retorno esperado es **+0.2%**”

Ejemplo mental

Modelo: retorno esperado = $+0.05\% \rightarrow$ NO TRADE (costos)

Modelo: retorno esperado = $+0.35\% \rightarrow$ TRADE

Ventajas

- ✓ Te dice **cuánto** podrías ganar
- ✓ Útil para position sizing

Desventajas

- ✗ Muy sensible al ruido
- ✗ Difícil de acertar magnitudes en mercados financieros
- ✗ **Conclusión:** se usa, pero **no es lo más común para ganar.**
- ✗ **Qué tipo de información es**
- 👉 Información **cuantitativa de magnitud**

Ejemplo:

- $+0.15\%$
- -0.40
- $+120$ puntos
- $+0.15\%$

+0.15%

El modelo estima que el precio **subirá un 0.15%** en el horizonte que definiste.

Ejemplo:

- Precio actual: 10 000
- $+0.15\% \approx +15$ puntos
- Precio esperado $\approx 10\,015$

Cuando se usa

- Cuando el modelo predice **retornos porcentuales**
- Muy común en:
 - acciones
 - crypto
 - trading cuantitativo

Interpretación correcta

“No es seguro que suba,
pero el movimiento esperado medio es +0.15%”

En trading real

- +0.15% puede ser:
 - ✗ poco (si tus costos son altos)
 - ✓ suficiente (en timeframes cortos y costos bajos)

-0.40

Aquí hay **dos interpretaciones posibles**, y esto es MUY importante.

◆ Caso A: -0.40 como precio

👉 El modelo predice que el precio **bajará 0.40 unidades de precio**.

Ejemplo:

- Precio actual: 100.0
- Precio esperado: 99.6

Esto es típico en:

- acciones
- forex
- futuros

◆ **Caso B: -0.40 como retorno logarítmico o normalizado**

👉 Es un valor **adimensional**, usado internamente en ML.

Ejemplo:

- retorno log
- retorno estandarizado

⚠ **Esto NO se usa directamente para trading**

Primero se transforma a:

- porcentaje
- puntos
- decisión (long/short)

Interpretación correcta

Siempre debes preguntarte:

“¿En qué unidades está entrenado el modelo?”

Sin eso, el número **no significa nada operativo**.

+120 puntos

Qué significa

👉 El modelo estima que el precio **subirá 120 puntos**.

Ejemplo:

- Índice: SP500
- Forex: pips

- Crypto: dólares

Ejemplo real:

- BTCUSDT en 40 000
- Forecast: +120
- Precio esperado \approx 40 120

Cuando se usa

- Índices
- Futuros
- Trading institucional
- Sistemas donde el punto tiene significado económico

👉 IDEA CLAVE (muy importante)

Estos **tres números NO son órdenes de trading.**

Son **estimaciones estadísticas.**

```
Forecast ≠ ENTRAR
Forecast ≠ GANAR
Forecast = información
```

¿Cómo se usa un forecast en una estrategia?

Ejemplo lógico:

```
Si retorno esperado > costos + margen
    entonces considerar trade
Si no
    no operar
```

Ejemplo concreto:

- Forecast: +0.15%
- Costos: 0.08%
- Margen mínimo: 0.05%

👉 +0.15% > 0.13% → possible trade

👉 +0.15% ≤ 0.13% → no trade

Por qué el forecast es peligroso para principiantes

- El mercado es ruidoso
- Predecir **magnitudes exactas** es muy difícil
- Fácil sobreajustar
- Muchos modelos “aciertan” pero **pierden dinero**

Por eso te insistí antes:

👉 **forecast NO es obligatorio**

👉 Clasificación suele ser más robusta

Resumen claro (quédate con esto)

- +0.15% → retorno porcentual esperado
- -0.40 → depende de la unidad (precio o retorno transformado)
- +120 puntos → movimiento absoluto esperado
- Ninguno es una orden de trading
- Siempre debes saber:
 - **qué predices**
 - **en qué unidades**
 - **para qué horizonte**

B) Una señal de clase (clasificación)

- Ej: predice si el precio **sube/baja** en la próxima vela.
- Salida típica:
 - Clase: UP o DOWN
 - o probabilidad: $P(UP)=0.63$

“**¿Sube o baja?**”

Qué devuelve el modelo

- LONG / SHORT
 -

- $P(\text{up}) = 0.63$

¿Quién me da P?

Primero lo primero.

“P” sí te lo da alguien:

👉 te lo da tu modelo de Machine Learning

```
p = model.predict_proba(X_t)[0, 1]
```

Eso significa:

- $p = 0.61 \rightarrow$ el modelo estima **61% de probabilidad de subida**
- $p = 0.48 \rightarrow$ incertidumbre
- $p = 0.30 \rightarrow$ mayor probabilidad de bajada

Hasta aquí **solo tienes probabilidades**, no decisiones.

¿Quién decide entonces 0.58 y 0.42?

👉 TÚ, usando análisis cuantitativo.

Esos números **no son mágicos**, son **umbrales operativos** que se construyen **después** de entrenar el modelo.

¿Por qué NO usar simplemente 0.50?

Porque en mercados financieros:

- 0.50 no tiene ventaja estadística
- los costos (spread, slippage) matan señales débiles
- muchas predicciones “apenas mejores que azar” pierden dinero

👉 Por eso se crea una **zona gris**.

Cómo se construyen esos umbrales (paso a paso)

Paso 1 — Guardas las probabilidades del modelo

Después de entrenar, haces predicciones **out-of-sample**:

Fecha	p_up	Retorno real
t1	0.61	+0.3%
t2	0.57	-0.1%
t3	0.43	-0.2%
t4	0.39	+0.1%

Paso 2 — Analizas qué pasa según el valor de p

Agrupas por rangos:

Rango de p	Nº trades	Retorno medio
$p > 0.65$	120	+0.24%
0.60–0.65	210	+0.11%
0.55–0.60	300	+0.02%
0.45–0.55	500	-0.01%
0.40–0.45	260	-0.08%
$p < 0.40$	180	-0.19%



Aquí ves algo clave:

- **el edge solo existe en los extremos**
- el centro es ruido

Paso 3 — Definir la zona gris

Ves que:

- arriba de **0.58** empieza a haber ventaja clara
- abajo de **0.42** empieza a haber ventaja short

- en medio → ruido + costos = pérdida

Entonces defines:

```
LONG si p > 0.58
SHORT si p < 0.42
NO TRADE si 0.42 ≤ p ≤ 0.58
```

¿Esto es parte del ML?

No.

⚠ Esto ya NO es ML

Esto es **diseño de estrategia cuantitativa**.

El error típico es pensar:

“el modelo debería decirme cuándo entrar”

No.

El modelo **solo estima probabilidades**.

La **estrategia** decide **qué hacer con ellas**.

¿Dónde lo veo en la práctica?

Normalmente lo ves en:

1 Curvas por cuantiles

Divides las predicciones en deciles (10%, 20%, etc.) y miras retornos.

2 Backtest por umbral

Pruebas:

- $p > 0.52$
- $p > 0.55$
- $p > 0.58$
- $p > 0.60$

Y comparas:

- profit factor
- drawdown

- estabilidad

3 Heatmaps (muy común)

Ejes:

- X = umbral long
- Y = umbral short
- Color = Sharpe o retorno

El “mejor” punto suele estar **lejos de 0.50.**

Entonces, respondiendo directo a tu pregunta

¿Quién me da esto?

- ✗ No te lo da Python
- ✗ No te lo da scikit-learn
- ✗ No te lo da el modelo
- ✗ No te lo da MT5

✓ Te lo da tu análisis cuantitativo sobre las salidas del modelo

Es una **decisión estadística + económica**, no de ML puro.

Regla de oro (muy importante)

Si tu estrategia:

- solo funciona con un umbral ultra específico (ej. 0.5837)
- cambia mucho con pequeñas variaciones

👉 probablemente está **sobreajustada**.

Los buenos umbrales:

- funcionan en rangos (0.56–0.60)
- son estables en distintos períodos

Ejemplo real:

“Hay un **63% de probabilidad** de que el precio suba en la próxima vela”

Cómo se usa

- Defines umbrales (0.58 / 0.42)
- Entras solo cuando hay ventaja clara

Ejemplo mental

```
p = 0.61 → LONG  
p = 0.52 → NO TRADE  
p = 0.38 → SHORT
```

Ventajas

- ✓ Más estable que predecir precios
- ✓ Fácil de convertir en reglas
- ✓ Compatible con MT5 / MQL

Desventajas

- ✗ Accuracy puede ser baja (52–56%)
- ✗ Necesita buena gestión de riesgo

✍️ Conclusión:

- 👉 Este es el más usado por traders cuantitativos independientes
- 👉 Es el que mejor encaja con lo que tú estás planteando

C) Ranking (Selección) 🏆

Ej: de 50 acciones, el modelo ordena cuáles tienen mejor expectativa.

Salida: lista ordenada + score.

“**¿Cuál es mejor que cuál?**”

Qué devuelve el modelo

Una **lista ordenada** de activos o setups.

Ejemplo:

- 1** EURUSD (score 0.72)
- 2** GBPUSD (0.61)
- 3** USDJPY (0.44)

Cómo se usa

- Operas solo el TOP 1 o TOP 3
- Ignoras el resto

Ejemplo mental

Opero solo los 2 activos con mayor score

Ventajas

- ✓ Excelente para **carteras**
- ✓ Reduce overtrading
- ✓ Muy usado en fondos

Desventajas

- ✗ Menos útil si solo operas un activo
- ✗ Requiere varios instrumentos

❖ Conclusión:

👉 Muy usado por **fondos y carteras**, menos por traders retail de un solo activo.

D) Régimen / Estado de mercado ⚡

Ej: “mercado en tendencia vs rango” y ajustas reglas según el estado.

Pregunta que le haces al ML

“¿El mercado está en tendencia, rango o alta volatilidad?”

❖ Importante:

El modelo **no te da una estrategia lista**. Te da una **predicción o probabilidad** sobre un evento futuro **bajo tu definición de objetivo (label)**.

Además, a nivel técnico, también “obtienes”:

- métricas offline (accuracy, AUC, RMSE, etc.)
- curvas de backtest (equity curve)
- estimación de costos/riesgo (si lo evaluaste bien)

Pero lo que sirve para operar es: **predicción + confianza (probabilidad) + condiciones**.

LÓGICA ML VS MQL

Debe trazarse una estrategia completa en **MQL** con todas las reglas clásicas:

- filtros
- gestión de riesgo
- SL / TP
- time stop
- horarios
- spreads, etc.

ML ya entrenado solo te dice algo como:

- **ENTRAR LONG**
- **ENTRAR SHORT**
- **NO HACER NADA**
- (y opcionalmente) **SALIR**

El **ML no gobierna la estrategia, solo la asiste.**

Forma mental correcta (muy importante)

Piensa esto así:

MT5 = piloto

ML = copiloto

El copiloto dice:

“Las condiciones estadísticas son favorables ahora”

El piloto decide:

“Vale, entro, pero con este riesgo y estas reglas”

Arquitectura típica (la que estás describiendo)

En MQL5 (estrategia principal)

Aquí vive TODO lo operativo:

- Mercado y timeframe
- Filtros:
 - spread máximo
 - horario
 - volatilidad mínima/máxima
 - tendencia mayor
- Gestión:
 - tamaño de posición
 - stop loss
 - take profit
 - trailing
 - max DD diario
- Control:
 - nº máximo de trades
 - no revenge trading
 - cierre forzado

El ML (ya entrenado)

Solo responde preguntas del tipo:

¿Hay ventaja estadística ahora?

Y devuelve algo como:

LONG / SHORT / NONE

(o una probabilidad **p** que tú conviertes en eso)

Ejemplo EXACTO de lo que estás diciendo

Supón que tu ML devuelve:

p = 0.64

En MQL5 tu lógica sería algo así (conceptual):

```
if (modelo_dice_LONG)
    if (filtros_ok && riesgo_ok && no_trade_activo)
        abrir LONG
```

Si el ML dice:

- **p = 0.51**
→ MT5 ignora completamente esa señal

Si dice:

- **p = 0.38**
→ señal SHORT (si tu estrategia lo permite)

¿Y la SALIDA? (esto es clave)

A Opción A (la más común y robusta)

👉 El ML SOLO decide entradas

La salida la maneja **la estrategia clásica**:

- Stop Loss
- Take Profit
- Trailing Stop
- Time Stop
- Cierre por riesgo

📌 Ventaja:

- Mucho más estable

- Menos sobreajuste
- Más fácil de controlar en MT5

👉 Desventaja:

- No “exprime” tanto el modelo

👉 **Esta es la opción que recomiendo al empezar.**

➡ Opción B (más avanzada)

👉 El ML decide **entrar y salir**

Ejemplo:

- ML dice LONG → entras
- Luego ML cambia y dice NONE o SHORT → sales

👉 Problemas:

- Mucho ruido
- Muchas entradas/salidas
- Costos altos
- Más difícil de validar

👉 Solo vale la pena si:

- el modelo es muy estable
- timeframe alto
- costos bajos

Regla de oro (apúntala)

El ML es mejor para decidir CUÁNDO ENTRAR

La gestión clásica es mejor para decidir CÓMO SALIR

Flujo real tal como tú lo estás entendiendo (correcto)

1. Diseñas una estrategia completa en MT5
2. Añades un “input externo”: el ML
3. El ML solo responde:

- ¿sí o no?
- ¿long o short?

4. MT5 ejecuta **solo si todo cuadra**

5. El riesgo manda, no el modelo

Eso **no solo tiene sentido**, es exactamente como trabajan muchos desks cuantitativos y traders algorítmicos serios.

Última aclaración importante

El ML **no reemplaza** una mala estrategia.

Pero **puede potenciar** una estrategia decente:

- menos trades
- mejor timing
- menor drawdown
- más consistencia

“ya terminé mi ML” ¿qué hago después?

Supón que ya entrenaste un clasificador que devuelve:

- $P(\text{up_next_bar})$ para la próxima vela de 1H en BTCUSDT.

Resultado del ML (lo que tienes)

Cada hora, tu modelo produce:

- $p = 0.61$ (61% probabilidad de subida próxima hora)

Paso 1: Convertir a señal

Ejemplo de reglas:

- **Entrada LONG** si $p > 0.58$
- **Entrada SHORT** si $p < 0.42$
- Si está entre 0.42 y 0.58 → no hacer nada (zona gris)

Esto reduce “ruido”.

Paso 2: Reglas de salida y riesgo (obligatorio)

- Stop Loss = $1.2 * \text{ATR}(14)$
- Take Profit = $1.8 * \text{ATR}(14)$
- Time stop: salir si pasan 6 velas sin TP/SL
- Máximo 1 operación activa a la vez
- Riesgo por trade: 0.5% de la cuenta

Paso 3: Backtest realista

Simulas:

- spread + comisiones
- slippage (ej. 0.5–1 tick promedio)
- ejecuciones parciales si aplica

Evalúas:

- Profit factor
- Sharpe / Sortino
- Max drawdown
- % tiempo en mercado
- estabilidad por meses (no solo total)

Paso 4: Si funciona → “trazar estrategia”

Aquí “trazar” significa: escribir tu plan exacto:

- Instrumento: BTCUSDT
- Timeframe: 1H
- Features: X, Y, Z
- Señal: umbrales $p>0.58$ / $p<0.42$
- Entradas: al cierre de vela
- SL/TP: ATR
- Tamaño: 0.5% riesgo

- Filtros: no operar en noticias / o no operar cuando spread > X
- Monitoreo: apagar si DD > 8%

Eso ya es una estrategia.

Paso 5: Implementación (MQL5 o puente)

Opción A: Codificar la estrategia en MQL5

- Si el “modelo” se puede portar (ej. reglas, pesos lineales, árbol pequeño), lo implementas directo.
- Si es complejo (XGBoost grande, redes), lo normal es:
 - exportar a ONNX (cuando posible) o
 - usar un servicio externo (Python) que calcule p y MT5 solo ejecute.

Opción B: Mantener todo en Python y operar por API

- Tu notebook se vuelve script/servicio.
 - Más flexible para ML, pero necesitas infraestructura.
-

Punto clave para que “tenga sentido”

En trading cuantitativo el ML **no es el final**, es el “predictor”.
El valor real aparece cuando:

- defines reglas robustas (umbral + filtro + riesgo)
- haces backtest serio (costos)
- validas out-of-sample y walk-forward
- operas con disciplina y control de drift