

Detección de Cambios de Régimen en EUR/USD con Modelos HMM

1. Introducción

El objetivo de este estudio es aplicar **Modelos Ocultos de Markov (HMM)** para identificar regímenes de mercado en el par **EUR/USD intradía**.

Los mercados financieros presentan dinámicas no estacionarias: periodos de baja volatilidad intercalados con episodios de alta volatilidad o tendencias abruptas.

Un HMM permite modelar estos **estados ocultos** y estimar probabilísticamente en qué régimen se encuentra el mercado en cada momento.

1.2. Datos y preprocesado

- Fuente: **Dukascopy EUR/USD**, velas de 15 minutos, periodo **2000–2025**.
 - Variables originales: `open`, `high`, `low`, `close`, `volume`.
 - Features generados (`preprocessing.py`):
 - **Retornos**: `ret_log`.
 - **Rangos**: `range_hl`, `range_co`.
 - **Volumen**: `log_volume`.
 - **Indicadores técnicos**: `rsi_14`, `macd`, `macd_signal`, `adx`, `ma20_rel`, `mfi_14`.
 - **Volatilidad**: `atr_14`, `rolling_std_1h` ($\approx 1h$), `rolling_std_1d` (≈ 1 día).
 - **Tiempo**: `hour_sin`, `hour_cos`, `dow_sin`, `dow_cos`.
 - **Extremos recientes**: `dist_max20`, `dist_min20`.
 - Preprocesamiento:
 - (Opcional) **winsorización** de colas extremas.
 - **Estandarización z-score con media y desviación del TRAIN** de cada split → evita *data leakage*.
 - Splits:
 - **Por tamaño de ventana**: típicamente `train=200k`, `val=50k`, `test=50k` observaciones.
 - Opción de ventana deslizante con `step_size`.
-

1.3. Metodología – Fase 1 (HMM Gauss)

- Modelo: **Gaussian HMM** multivariante (`hmmlearn`).
 - Selección de *features*: **manual**, probando unos pocos conjuntos:
 - `[ret_log, rolling_std_1h, atr_14]`
 - `[rsi_14, macd, macd_signal]`
 - Configuraciones de entrenamiento:
 - Estados ocultos: `2`, `3`, `4`.
 - Covarianza: `full` y `diag`.
 - `n_iter=200`, `seed=42`.
 - Evaluación:
-

- `ll_val_per_obs` como métrica principal.
- AIC y BIC en TRAIN para penalizar complejidad.
- Robustez temporal entre splits.
- Interpretabilidad: duración media razonable de los estados y varianza de retornos diferenciada.
- Identificación del estado más volátil: el de **máxima `var(ret_log)`**.

1.4. Resultados de experimentación

1.4.1 Ranking de configuraciones (Top-K)

Los mejores resultados por `ll_val_per_obs_mean` (promedio sobre splits) fueron:

Rank	Configuración	Estados	Cov	Features	LL Val/Obs
1	st4_full_rsi_14-macd-macd_signal	4	full	rsi_14, macd, macd_signal	...
2	st3_full_rsi_14-macd-macd_signal	3	full	rsi_14, macd, macd_signal	...
3	st2_full_rsi_14-macd-macd_signal	2	full	rsi_14, macd, macd_signal	...

Fuente: `grid_summary.csv` con el **short-list manual** de features.

1.4.2 Visualizaciones clave

- Precio coloreado por estados.
- Probabilidad del estado más volátil.
- Histogramas de duración y matriz de transición.

1.4.3 Estadísticas por estado (ejemplo)

Estado	Count	Duración media (barras)	Var(<code>ret_log</code>)	Interpretación
0	...	~26	0.75	Tranquilo (baja vol.)
1	...	~27	0.70	Tranquilo (otra fase)
2	...	~30	1.55	Volátil alcista/bajista
3	...	~27	1.60	Volátil alcista/bajista

1.5. Interpretación de resultados

- **2 estados:** distingue calma vs turbulencia, útil como filtro de riesgo.
- **3 estados:** degenerado (dos estados con duración mediana ≈ 1 barra).
- **4 estados:** la estructura más rica y estable:

- Dos regímenes de baja volatilidad (posibles sesgos alcista/bajista).
 - Dos regímenes de alta volatilidad (alcista/bajista).
 - Duraciones medias $\approx 6h$ (en M15).
 - Buena separación de varianzas.
- 4 estados → sobreajuste (AIC/BIC altos, estados redundantes).

1.6. Conclusiones de Fase 1

- Mejor configuración: **Gaussian HMM, 4 estados, cov=full, features [rsi_14, macd, macd_signal]**.
- Descubrimientos:
 - Los regímenes son persistentes, no cambian vela a vela.
 - La volatilidad es el eje dominante de separación.
- Limitaciones:
 - Distribución gaussiana sub-modela colas pesadas → outliers mal explicados.
 - Posible "label switching" entre splits.
 - Estados >4 producen sobreajuste.
- Puente a Fase 2: en Fase 1 la selección de features fue **manual**; en la siguiente se hará de forma **automática** priorizando conjuntos poco correlacionados.

1.7. Próximos pasos

- **Fase 2:** ampliar experimentación con subconjuntos de features poco correlacionados (selección automática).
- **Fase 3:** repetir configuraciones con **HMM Student-t** para robustez frente a colas pesadas.
- **Fase 4:** migrar a **HSMM (Hidden Semi-Markov Models)** para modelar duraciones explícitas de regímenes.

2.1. Metodología – Fase 2 (HMM Gauss con diferentes features (poco correlacionados + selección automática))

Objetivo. Construir y evaluar subconjuntos de features (subsets) que sean informativos para cambios de régimen y poco redundantes entre sí. Esta fase sustituye la elección manual de Fase 1 por una búsqueda sistemática de combinaciones plausibles y un filtro de correlación intra-subset.

Evaluación de cada subconjunto/subset.

Antes de entrenar: filtro intra-subset por correlación $|p| \geq 0.85$, eliminando la feature con menor puntuación de relevancia (no se toca el DataFrame global, solo el subset).

Después de entrenar (con `experiment_hmm.py`):

Métrica principal: `ll_val_per_obs_mean` (validación).

Penalización de complejidad: `AIC_mean`, `BIC_mean`.

Estabilidad/interpretabilidad (con `evaluate_hmm.py`):

Duraciones medianas por estado (evitar degeneración $L \approx 1$).

Matriz de transición con diagonales altas.

Separación por `var(ret_log)` entre estados (estado “volátil” bien diferenciado).

Criterio final. Elegimos los Top-K subsets por `ll_val_per_obs_mean`. Empates se resuelven con `BIC_mean` (menor es mejor) y sanidad de estados (duraciones razonables y separación clara).

2.2 Selección automática de subsets y filtrado de features por correlación

Métricas para decidir si un subset es “bueno”

- Previo a entrenar:

Relevancia media de sus features (ej. suma de MI).

Diversidad (baja correlación intra-subset).

- Post-entrenar:

`ll_val_per_obs_mean` alto.

AIC/BIC más bajos que otros.

Estados con duraciones >2–3 barras y transiciones persistentes.

Separación clara de volatilidad/retornos entre estados.

Flujo automático de selección de subsets

- Calcular relevancia de cada feature
- Usa un proxy de régimen (volatilidad futura, retornos siguientes) y mide:
- Mutual Information (MI) o correlación absoluta con el proxy.
- Esto te da un ranking global de features (las más “informativas” sobre dinámicas futuras arriba).
- Generar candidatos de subsets
- Construyes automáticamente subsets de 3–5 features tomando las Top-N del ranking.

Si un feature actual (ej. `RSI`, `log_volume`) tiene alta correlación o alta información mutua con un proxy de régimen futuro, significa que ese feature contiene información que ayuda a anticipar o explicar cambios de régimen.


Ejemplo:

`rolling_std_1h` tendrá alta correlación con la volatilidad futura `rv_1h`.

rsi_14 puede correlacionarse con el retorno futuro.

log_volume con ambos, ya que picos de volumen anticipan volatilidad.

Esto nos da una forma no supervisada pero guiada de rankear features:

- Ejemplo: todas las combinaciones posibles de 3 features entre las 10 mejores (eso son 120 subsets).
- Filtro de correlación intra-subset
- Para cada subset candidato:
- Calcula matriz de correlación entre sus features.
- Si algún par tiene $|p| > 0.85$, elimina la menos relevante (según ranking inicial).
- Si el subset se queda con  features → se descarta.
- Resultado: un conjunto de subsets filtrados y diversos.
- Deduplicación y poda
- Quita duplicados.
- Si hay demasiados subsets, conserva solo los Top-M según un criterio rápido (ej. suma de MI de las features del subset).

Evaluación con el modelo

- Pasas todos esos subsets a experiment_hmm.py.
- Se entrenan los HMM y se comparan por ll_val_per_obs_mean, AIC/BIC, estabilidad de estados.

2.3 Resultados de experimentación

Una vez generados los subsets, ejecutamos experiment_hmm.py:

```
"python.exe "e:/Tradingg Bot/MM-Algo-Trading_BOT/v6/Market-Master-Algorithmic-Trading-System/MODELOS/AI/HMM/experiment_hmm.py" --feature-sets fs_phase2.json --states-list "2,3,4" --keep-top-k 10"
```

3.1. Metodología – Fase 3 (HMM t-student)

3.2 Resultados usando los mismos subconjuntos de features que en fase 2

4. Metodología - Fase 4 (HSMM)