Facsímil 07

Evaluar, calibrar e interpretar

Capítulo 05

Calibración e incertidumbre: de scores a decisiones

686f6c61 · 2026

Capítulo 05: Calibración e incertidumbre: de scores a decisiones

Entrando en el tema

En el capítulo 02 vimos que un score se convierte en acción cuando le ponemos umbrales. En el capítulo 04 vimos que un evaluador LLM también debe medirse antes de confiar en sus veredictos. Ahora falta una pregunta incómoda: **¿ese 0,82 que aparece en una métrica, un clasificador, un recuperador o un evaluador significa algo parecido a “82 %”?**

Muchas veces no. Un sistema puede ordenar bien los casos y estar mal calibrado. También puede decir “alta confianza” sin que esa confianza corresponda a una frecuencia real de acierto. Para ingeniería, esa diferencia importa muchísimo: no es lo mismo una puntuación útil para ordenar que una probabilidad útil para automatizar.

Qué deberías poder hacer al terminar

Al terminar deberías poder hacer esto:

RESULTADO DE APRENDIZAJE	EVIDENCIA DE QUE LO SABES HACER
Separar score, probabilidad y decisión.	Puedes explicar por qué un 0,9 puede ordenar bien y aun así no ser una probabilidad fiable.
Medir calibración.	Calculas Brier score, log loss, ECE y lees un reliability diagram.
Elegir una técnica de calibración.	Distingues Platt scaling, isotonic regression, histogram/binning y temperature scaling.
Separar datos sin contaminar evaluación.	Distingues train, calibration, test final y producción revisada.
Leer calibración multiclase.	Sabes cuándo mirar top-label, por clase o distribución completa.
Usar incertidumbre sin teatralizarla.	Diseñas zona de revisión, abstención o salida con intervalo cuando el sistema no tiene suficiente evidencia.
Entender conformal prediction.	Construyes conjuntos o intervalos con cobertura objetivo y sabes qué supuesto los sostiene.
Convertir calibración en política operativa.	Escribes umbrales, costes, revisión y criterios de publicación.
Conectar umbrales con capacidad humana.	Usas tasa de revisión, cola y SLO para no diseñar una política imposible.
Medir incertidumbre estadística.	Añades intervalos, bootstrap y lectura por slice antes de sacar conclusiones.
Vigilar deriva.	Identificas dataset shift, slices nuevos y triggers de recalibración.
Entregar una práctica reproducible.	Produces datos, script, reporte JSON, manifest y decisión Markdown que un equipo puede revisar.

La idea central del capítulo es sencilla: **un score no merece mandar hasta que sabes qué significa cuando se equivoca**.

El 0,92 que no significa 92 %

Imagina un sistema que prioriza tickets de soporte. Un caso llega con score 0,92 de “urgente”. Producto quiere automatizar: si supera 0,90, se marca como urgente y se salta la cola.

Antes de hacerlo, necesitamos saber qué significa ese 0,92.

LECTURA INGENUA	LECTURA DE INGENIERÍA
“El sistema está seguro al 92 %”.	“En casos parecidos con score cercano a 0,92, ¿cuántos eran realmente urgentes?”
“El score es alto, automatizamos”.	“¿Qué coste tiene equivocarnos aquí y cuántos casos de esta banda hemos medido?”
“Si ordena bien, ya sirve”.	“Ordenar bien no basta si el score alimenta umbrales, revisión o SLA.”

Esto aparece por todas partes en IA aplicada:

LUGAR DONDE APARECE UN SCORE	ERROR TÍPICO
Clasificador de tickets	Leer `0.87` como probabilidad sin medir calibración.
RAG	Confundir similitud de embedding con probabilidad de que la respuesta esté soportada.
Evaluador automático	Tratar una nota `4/5` como verdad operacional sin calibrarla contra casos revisados.
Agente con tool calls	Usar “confidence” textual del modelo como si fuera una métrica medida.
Modelo local o API	Comparar scores de proveedores distintos como si vivieran en la misma escala.

Un score puede servir para ordenar. Una probabilidad calibrada sirve para decidir bajo coste. No son la misma promesa.

Qué no es calibrar

Calibrar no es subir la accuracy. Un sistema puede tener la misma accuracy antes y después de calibrar, pero producir probabilidades más honestas.

Tampoco es “hacer que el modelo dude”. A veces calibrar baja scores exagerados; otras veces sube scores demasiado conservadores. La dirección no importa. Importa que el número signifique algo empírico.

Y calibrar tampoco sustituye a una buena evaluación. Si tu dataset no representa el uso real, si mezclas datos de ajuste con datos de evaluación o si cambias el umbral después de mirar el resultado, tendrás una apariencia de rigor, no una política fiable.

Podemos resumirlo así:

CONCEPTO	PREGUNTA QUE RESPONDE	QUÉ NO RESPONDE
Discriminación	¿Ordena positivos por encima de negativos?	Si el 0,8 significa 80 %.
Accuracy	¿Cuántos acierta con un umbral dado?	Si sus probabilidades son honestas.
Calibración	¿El score coincide con frecuencia real?	Si el modelo entiende el dominio.
Incertidumbre	¿Cuánto margen de duda queda?	Qué decisión de producto conviene tomar.
Política	¿Qué hacemos con esa duda?	Si los datos de partida eran buenos.

Qué sí es una probabilidad calibrada

Una predicción probabilística está calibrada cuando, entre los casos a los que asigna probabilidad p, la frecuencia real del evento también es p.

La expresión siguiente no es una fórmula inventada para el capítulo: es la definición estándar de calibración probabilística o fiabilidad de probabilidades. La verás escrita con esta forma, o con bandas aproximadas, en literatura de predicción probabilística y calibración de clasificadores. En el capítulo la usamos como definición ideal; en datos finitos la aproximamos con reliability diagrams y métricas por bandas.

$$\mathbb{P}(Y = 1 \mid \hat{p}(X) = p) = p$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
X	Entrada del sistema.	Texto de un ticket.
Y	Etiqueta real.	1 si el ticket era urgente.
$\hat{p}(X)$	Probabilidad predicha por el sistema.	0,80.
p	Banda o valor de confianza.	Casos alrededor de 0,80.
$\mathbb{P}(Y = 1 \mid \hat{p}(X) = p)$	Frecuencia real de positivos dentro de esa banda.	0,78 en la muestra.

En palabras: si el sistema asigna 0,80 a muchos casos parecidos, alrededor del 80 % de esos casos deberían ser positivos. Si no ocurre, el número puede ordenar, pero no se puede leer como probabilidad honesta.

En la práctica no tenemos infinitos casos con exactamente $p = 0,80$. Agrupamos predicciones en bandas: 0,0-0,1; 0,1-0,2; 0,2-0,3; y así sucesivamente. Si en la banda 0,8-0,9 el score medio es 0,84 y el 62 % de los casos son realmente positivos, el modelo está sobreconfiado en esa banda.

La calibración es local. Un promedio bonito puede esconder bandas malas. Por eso miramos la curva completa.

Fecha de corte del estado del arte

Fecha de corte: 1 de junio de 2026.
Fuentes consultadas: trabajos clásicos sobre probabilidades calibradas, Brier score, reliability diagrams, calibración supervisada, calibración de redes neuronales modernas y conformal prediction; además de documentación de scikit-learn, trabajos sobre incertidumbre en LLMs, documentación técnica de modelos/datos y guías de producción de sistemas ML.

Brier propuso en 1950 una puntuación para predicciones probabilísticas que mide el error cuadrático entre probabilidad y resultado observado.[1] Murphy descompuso después esa puntuación en componentes relacionados con fiabilidad, resolución e incertidumbre.[2]

Niculescu-Mizil y Caruana mostraron que clasificadores distintos producen probabilidades con comportamientos de calibración muy distintos, aunque su capacidad de ranking sea buena.[3] Platt scaling popularizó una calibración sigmoidal para salidas de SVM.[4] Guo y otros mostraron que redes neuronales modernas pueden estar muy bien en accuracy y mal calibradas, y que temperature scaling puede corregir parte de esa sobreconfianza con un ajuste simple.[5]

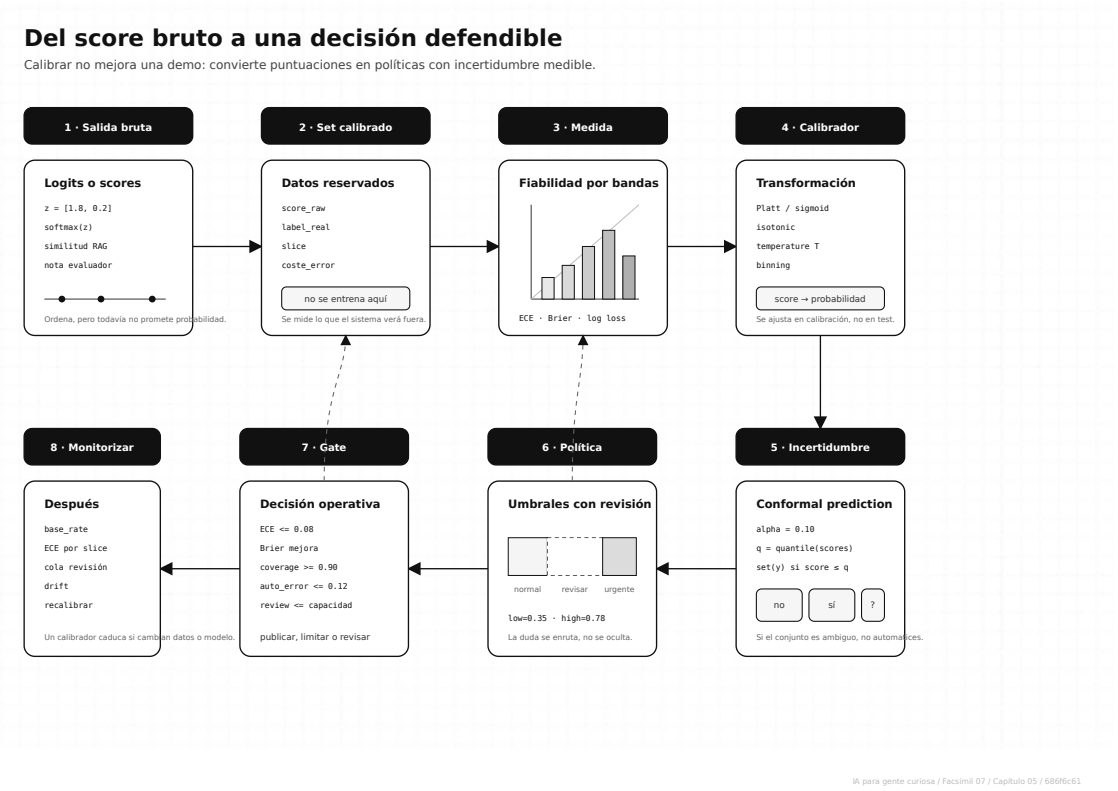
Para cuantificar incertidumbre con garantías finitas, la familia de conformal prediction viene de los trabajos de Vovk, Gammerman y Shafer.[6] Shafer y Vovk ofrecen una introducción tutorial al enfoque.[7] Angelopoulos y Bates escribieron una introducción moderna a conformal prediction y cuantificación de incertidumbre libre de distribución.[8]

La documentación de scikit-learn resume la diferencia práctica entre calibración, curvas de fiabilidad y métodos como sigmoid e isotonic calibration.[9]

Para llevar esto a ingeniería de IA moderna, también nos apoyamos en trabajos sobre incertidumbre en modelos de lenguaje, documentación de modelos y datos, y preparación para producción.

PIEZA DE INGENIERÍA	FUENTE USADA
Incertidumbre en modelos de lenguaje	Kadavath y otros estudian cuándo los modelos de lenguaje pueden reconocer límites de conocimiento bajo determinados protocolos.[10]
Incertidumbre semántica	Kuhn, Gal y Farquhar proponen agrupar respuestas que dicen lo mismo aunque usen palabras distintas.[11]
Documentación de modelos	Model Cards propone reportar usos previstos, límites y resultados por segmentos.[12]
Documentación de datos	Data Cards estructura información sobre origen, composición, límites y uso previsto de datasets.[13]
Preparación para producción	Sculley y otros describen deuda técnica propia de sistemas ML.[14]
Readiness de ML	El ML Test Score propone pruebas y necesidades de monitorización para sistemas ML en producción.[15]
SLI/SLO	La guía SRE de Google separa indicador, objetivo y presupuesto de error para decidir con datos.[16]

Anatomía de una política calibrada



Una política calibrada conecta score bruto, datos reservados, medición, calibrador, incertidumbre, umbrales, gate y monitorización.

Medir calibración: Brier, log loss y ECE

Hay tres medidas que conviene tener cerca. No dicen exactamente lo mismo, y esa diferencia es útil.

Brier score

Brier score mide el error cuadrático entre la probabilidad predicha y la etiqueta real:

La fórmula viene del Brier score de Brier (1950), una regla de puntuación probabilística para predicciones expresadas como probabilidades. En binario se escribe como error cuadrático entre probabilidad y resultado observado.

$$BS = \frac{1}{N}\sum_{i=1}^{N}(\hat{p}_i - y_i)^2$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
BS	Brier score; cuanto menor, mejor.	0,142.
N	Número de casos.	200 tickets.
$\hat{p}_i$	Probabilidad predicha para el caso i.	0,80.
y_i	Etiqueta real del caso i: 0 o 1.	1 si era urgente.

En palabras: Brier mide cuánto se separa cada probabilidad de lo que ocurrió. Si das mucha probabilidad a algo que no pasa, pagas mucho; si das una probabilidad cercana al resultado real, pagas poco.

Si predices 0,80 y el caso era positivo, aportas $(0,80-1)^2 = 0,04$. Si predices 0,80 y el caso era negativo, aportas $(0,80-0)^2 = 0,64$. El Brier castiga tanto mala calibración como mala discriminación, pero es fácil de explicar.

Log loss

Log loss penaliza de forma dura estar muy seguro y equivocarte:

Esta fórmula es la pérdida logarítmica binaria, equivalente a la negative log-likelihood de una variable Bernoulli. En la literatura de scoring rules aparece dentro de las reglas de puntuación propias: si quieres incentivar probabilidades honestas, no basta con contar aciertos; debes puntuar la distribución probabilística. Gneiting y Raftery (2007) revisan esta familia de reglas de puntuación propias.

$$LL = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{p}_i) + (1-y_i)\log(1-\hat{p}_i)\right]$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
LL	Log loss; cuanto menor, mejor.	0,41.
$\log$	Logaritmo natural.	$\log(0,9)$.
$\hat{p}_i$	Probabilidad predicha, recortada para evitar 0 o 1 exactos.	0,97.
y_i	Etiqueta real.	0.

En palabras: log loss castiga la mala sorpresa. Equivocarse con duda moderada duele; equivocarse diciendo casi 1 o casi 0 duele muchísimo más.

Si un sistema dice 0,99 y falla, log loss lo deja en evidencia. Eso es sano cuando una decisión automática usa el score como confianza.

ECE

Expected Calibration Error agrupa predicciones por bandas y compara confianza media con accuracy real:

La ECE no tiene la misma solidez histórica que Brier o log loss: es una métrica práctica por bandas usada mucho en calibración moderna. La asociamos a trabajos como Naeini, Cooper y Hauskrecht (2015), y aparece después en Guo y otros (2017) para analizar calibración de redes neuronales modernas. Por eso la tratamos como señal útil, no como veredicto suficiente.

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{N} \left|\mathrm{acc}(B_m) - \mathrm{conf}(B_m)\right|$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
M	Número de bandas.	10 bandas.
B_m	Casos que caen en la banda m.	Scores entre 0,8 y 0,9.
$	B_m	$
$\mathrm{acc}(B_m)$	Frecuencia real de acierto en esa banda.	0,65.
$\mathrm{conf}(B_m)$	Confianza media predicha en esa banda.	0,84.

En palabras: ECE mira banda a banda cuánto se separa lo que el sistema decía de lo que ocurrió. Una banda con muchos casos pesa más que una banda pequeña.

ECE es intuitivo, pero depende de cómo elijas bandas. Por eso no lo usaría solo. Lo pondría junto a Brier, log loss, reliability diagram y análisis por slice.

Reliability diagram: mirar la curva, no solo el número

Un reliability diagram pone en el eje X la confianza media de cada banda y en el eje Y la frecuencia real de acierto. La diagonal perfecta significa calibración ideal.

PATRÓN VISUAL	LECTURA
Curva por debajo de la diagonal	El sistema está sobreconfiado: promete más de lo que cumple.
Curva por encima de la diagonal	El sistema es conservador: acierta más de lo que su score dice.
Bien en scores bajos, mal en scores altos	Peligroso si automatizas por umbral alto.
Bien en promedio, mal en un slice	No publiques sin política específica para ese slice.

Para proyectos de IA, el reliability diagram debe mirarse por familias de caso: idioma, dominio, fuente documental, tipo de usuario, longitud de entrada, modelo usado, recuperador usado, herramienta invocada o evaluador aplicado.

Una calibración global puede esconder que el sistema es honesto en tickets simples y sobreconfiado en documentos largos.

Métodos de calibración que sí se usan

Calibrar significa aprender una transformación que convierte scores brutos en probabilidades más fiables. Esa transformación debe ajustarse en un conjunto de calibración reservado, no en el test final.

MÉTODO	IDEA	CUÁNDO ENCAJA	CUIDADO
Platt scaling	Ajusta una sigmoide sobre el score bruto.	Clasificadores binarios con forma de error suave.	Puede quedarse corto si la curva real no es sigmoidal.
Isotonic regression	Aprende una función monótona por tramos.	Tienes suficientes datos de calibración.	Puede sobreajustar con pocos casos.
Histogram/ binning	Agrupa scores y sustituye por frecuencia observada.	Necesitas algo explicable y auditable.	Bandas con pocos casos son ruidosas.
Temperatur e scaling	Divide logits por una temperatura T antes de softmax.	Redes neuronales y clasificación multicategoría.	No cambia el ranking; solo suaviza o endurece confianza.
Vector/matr ix scaling	Ajustes más flexibles sobre logits multicategoría.	Multiclase con datos suficientes.	Más parámetros, más riesgo de ajuste a calibración.
Calibración por slice	Calibradores separados o correcciones por segmento.	Los segmentos se comportan de forma distinta.	Necesita volumen y control de deriva.

Temperature scaling se escribe así:

Esta fórmula es la softmax aplicada a logits después de dividirlos por una temperatura T. En este capítulo aparece por Guo y otros (2017), que muestran temperature scaling como calibración posentrenamiento sencilla para redes neuronales: ajusta confianza sin cambiar el ranking de clases.

$$\hat{p}_k = \frac{\exp(z_k / T)}{\sum_{j=1}^{K} \exp(z_j / T)}$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
z_k	Logit bruto de la clase k.	3,2 para `urgente`.
T	Temperatura aprendida en calibración.	1,7.
K	Número de clases.	3: normal, revisar, urgente.
$\hat{p}_k$	Probabilidad calibrada de la clase k.	0,74.

En palabras: la temperatura no cambia cuál era la clase más alta; cambia cuánto se concentra o se reparte la confianza entre clases.

Si $T > 1$, la distribución se suaviza: menos seguridad extrema. Si $T < 1$, se endurece: más masa en la clase dominante. Guo y otros mostraron que, para muchas redes modernas, una temperatura aprendida podía mejorar mucho la calibración sin cambiar la clase predicha.[17]

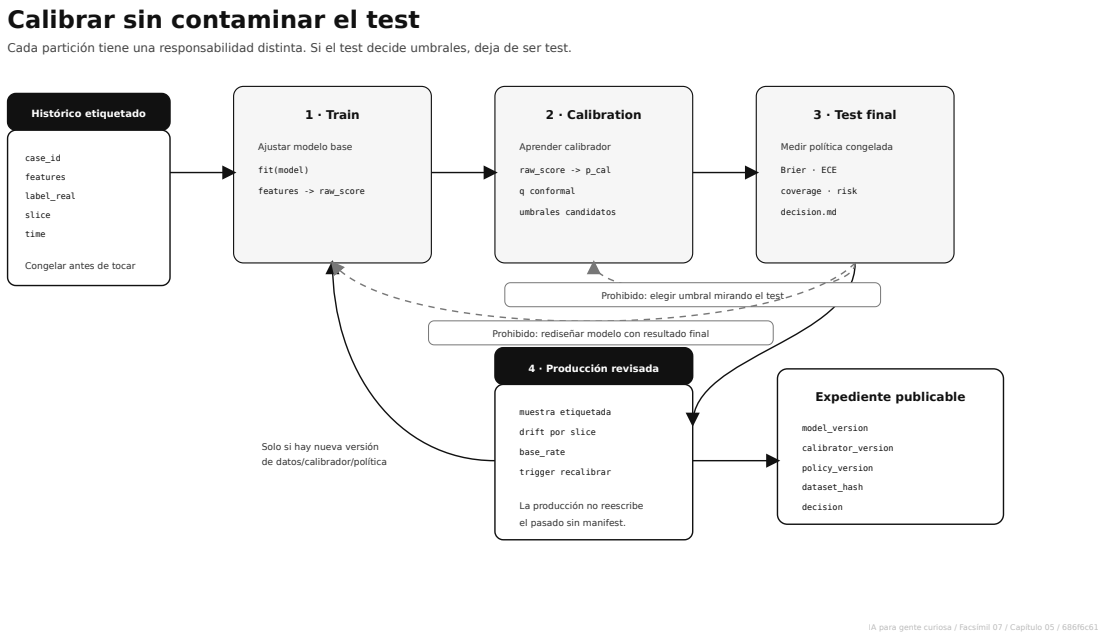
Partición de datos: calibrar no es mirar el examen final

El error más peligroso en calibración no es elegir Platt, isotonic o temperature scaling. Es usar mal los datos. Si entrenas el modelo, ajustas el calibrador, eliges umbrales y después dices que has validado todo con el mismo conjunto, no has medido una política: la has ido moldeando hasta que encaja con el examen.

La partición mínima debería separar cuatro papeles:

PARTE	QUÉ SE HACE AHÍ	QUÉ NO DEBERÍA HACERSE
Entrenamiento	Ajustar pesos, árbol, prompt entrenado, clasificador o modelo base.	Medir el resultado final de publicación.
Calibración	Aprender el calibrador y, si procede, estimar cuantiles conformales.	Cambiar la arquitectura principal mirando el test.
Evaluación final	Medir la política completa ya congelada.	Elegir umbrales después de ver el resultado.
Producción revisada	Vigilar deriva, base rate y errores reales.	Recalibrar en caliente sin control de versiones.

En scikit-learn, `CalibratedClassifierCV` existe precisamente porque calibrar con predicciones sesgadas por el entrenamiento puede engañarte: la documentación separa el ajuste del estimador base y el ajuste del calibrador mediante particiones de validación cruzada.[18] MAPIE usa el término `conformalization` para el conjunto donde se estiman scores de conformidad antes de producir intervalos o conjuntos predictivos.[19]



El split separa entrenamiento, calibración, evaluación final y producción revisada. El test no se usa para elegir umbrales.

En proyectos pequeños no siempre hay datos para particiones perfectas. Aun así, la intención debe quedar escrita. Si hay pocos casos, prefiero una política conservadora con revisión humana y una muestra nueva cada semana antes que una calibración que presume de precisión con doce ejemplos.

Calibración multiclase: cuando no basta mirar la clase ganadora

Hasta aquí el ejemplo ha sido binario porque ayuda a pensar. En producción aparecen tareas multicategoría: tipo de ticket, intención del usuario, ruta del agente, clase documental, nivel de severidad o decisión de negocio. Ahí la calibración se vuelve más delicada.

Una trampa común es mirar solo la confianza de la clase ganadora. Si el modelo dice `beca` con 0,82, parece razonable preguntar si los casos con top-label 0,82 aciertan alrededor del 82 %. Eso es útil, pero no agota el problema: una clase minoritaria puede estar muy mal calibrada aunque el promedio global parezca decente. Gupta y Ramdas estudian precisamente calibración top-label y reducciones multiclase-a-binario para razonar sobre esta diferencia.[20]

Para un sistema de IA, yo miraría tres niveles:

NIVEL	QUÉ PREGUNTA RESPONDE	EJEMPLO DE FALLO
Confianza top-label	“Cuando el modelo dice 0,8 en la clase elegida, ¿acierta cerca del 80 %?”	Parece bien en global, pero falla una intención rara.
Calibración por clase	“Para `becas`, `matrícula` o `producción`, ¿el score significa lo mismo?”	`becas` está sobreconfiado porque hay pocos ejemplos recientes.
Distribución completa	“¿La probabilidad repartida entre clases alternativas tiene sentido?”	El modelo pone 0,78 en `normal` y 0,20 en `urgente`, pero ambas entran en un conjunto conformal.

Esto cambia cómo decides. Si solo necesitas enrutar tickets y siempre habrá revisión, top-label puede bastar al principio. Si una clase dispara una acción cara o irreversible, necesitas medir esa clase como producto propio. Un “buen ECE global” no compensa automatizar mal el 4 % de casos que más daño hacen.

En LLMs y agentes, esta misma idea aparece con otros nombres. Un router de modelos puede elegir `modelo barato`, `modelo grande` o `revisión humana`. Un agente puede elegir `buscar`, `calcular`, `escribir` o `pedir permiso`. La calibración que importa no es una media estética: es si cada ruta tiene una probabilidad defendible de salir bien.

Incertidumbre: no todo tiene que salir como sí o no

En ingeniería, la incertidumbre no es una disculpa. Es una señal de control.

SEÑAL DE INCERTIDUMBRE	ACCIÓN RAZONABLE
Score cerca del umbral.	Mandar a revisión o pedir más evidencia.
Reliability diagram malo en esa banda.	No automatizar esa banda.
Conformal set con dos clases posibles.	No elegir una sola clase en automático.
RAG con citas débiles.	Abstenerse o responder con alcance limitado.
Evaluador automático con desacuerdo alto.	Revisar muestra humana y recalibrar.
Drift de base rate.	Recalcular calibración antes de mantener umbrales.

La salida profesional no siempre es “sí” o “no”. A veces es:

```
{
  "decision": "revisar",
  "reason": "score calibrado dentro de zona gris",
  "calibrated_probability": 0.61,
  "conformal_set": ["normal", "urgente"],
  "next_step": "enviar a cola de soporte nivel 2"
}
```

Esto no es menos inteligente. Es más honesto.

Conformal prediction: garantías con supuestos claros

Conformal prediction no intenta adivinar si el modelo “está seguro” en abstracto. Construye conjuntos o intervalos que contienen la respuesta correcta con una cobertura objetivo, bajo un supuesto clave: los datos de calibración y los datos futuros son intercambiables, es decir, vienen del mismo mecanismo de generación en el sentido estadístico que necesitamos para el problema.

En clasificación, una forma sencilla es usar como score de no conformidad:

Las fórmulas de esta sección vienen de conformal prediction. La familia clásica está desarrollada por Vovk, Gammerman y Shafer, y la exposición moderna de Angelopoulos y Bates ayuda a leerla como cuantificación de incertidumbre con cobertura finita bajo intercambiabilidad. Aquí usamos una versión sencilla para clasificación: score de no conformidad, cuantil y conjunto predictivo.

$$a_i = 1 - \hat{p}_{y_i}(x_i)$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
a_i	Score de no conformidad del caso i.	0,18.
$\hat{p}_{y_i}(x_i)$	Probabilidad asignada a la clase correcta del caso i.	0,82.
x_i	Entrada del caso i.	Ticket de soporte.
y_i	Clase real del caso i.	`urgente` .

En palabras: un caso es poco conforme si el modelo dio poca probabilidad a la clase que luego resultó ser la correcta.

Después elegimos un cuantil de esos scores en el conjunto de calibración:

Este cuantil es la pieza central del split conformal: se calcula sobre los scores de no conformidad de calibración y se elige de forma conservadora para sostener la cobertura finita descrita por Vovk, Gammerman y Shafer, y por la formulación moderna de Angelopoulos y Bates.

$$q = \text{Quantile}_{\lceil (n+1)(1-\alpha) \rceil / n} \left(a_1, \ldots, a_n \right)$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
q	Umbral conformal de no conformidad.	0,42.
n	Número de casos de calibración.	100.
α	Tasa de error permitida.	0,10 para cobertura 90 %.
$\lceil \cdot \rceil$	Redondeo hacia arriba.	Garantiza una elección conservadora.

En palabras: elegimos un umbral suficientemente alto para cubrir la fracción prometida de casos bajo el supuesto de intercambiabilidad.

Para un caso nuevo, incluimos cada clase cuyo score de no conformidad no supera q:

El conjunto predictivo $\Gamma_\alpha(x)$ es la salida propia de la predicción conformal en clasificación: no fuerza una única clase si varias siguen siendo plausibles bajo el umbral de no conformidad aprendido.

$$\Gamma_\alpha(x) = \{ y : 1 - \hat{p}_y(x) \leq q \}$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
$\Gamma_\alpha(x)$	Conjunto de clases plausibles para x.	`{normal, urgente}`.
y	Clase candidata.	`urgente`.
$\hat{p}_y(x)$	Probabilidad de esa clase para el caso nuevo.	0,63.
q	Umbral aprendido en calibración.	0,42.

En palabras: una clase entra en el conjunto si no parece demasiado rara comparada con los errores vistos en calibración.

Si el conjunto tiene una sola clase, quizá podemos automatizar. Si tiene dos o más, el sistema está diciendo: “con la cobertura que me pediste, no puedo elegir solo una sin perder garantía”. Esa frase vale oro en un producto real.

En regresión, la versión más sencilla usa residuos absolutos:

La forma de regresión conformal más básica usa residuos absolutos en calibración. No inventa una incertidumbre nueva: convierte los errores observados del modelo en un margen empírico.

$$a_i = |y_i - \hat{f}(x_i)|$$

Y construye un intervalo:

El intervalo conformal simétrico alrededor de $\hat{f}(x)$ es la traducción directa de ese margen q a una predicción numérica. Es sencillo, auditable y útil como punto de partida antes de variantes adaptativas.

$$C_\alpha(x) = \left[\hat{f}(x) - q, \hat{f}(x) + q\right]$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
$\hat{f}(x)$	Predicción numérica del modelo.	18 minutos de espera.
q	Cuantil de residuos en calibración.	6 minutos.
$C_\alpha(x)$	Intervalo conformal.	[12, 24] minutos.

En palabras: en regresión, miramos cuánto se equivocó el modelo en calibración y usamos ese margen para construir un intervalo alrededor de la nueva predicción.

La parte honesta: conformal prediction no arregla un dataset que ya no representa producción. Si cambia el canal de entrada, el idioma, el modelo base, la política de producto o el tipo de caso, recalibramos.

Conformal en proyectos reales: cobertura no es utilidad

La promesa de conformal prediction es potente: bajo supuestos claros, puedes construir conjuntos o intervalos con cobertura. Pero un sistema de ingeniería no vive solo de cobertura. Vive de utilidad. Un conjunto que contiene siempre cinco clases puede cubrir mucho y decidir poco.

Por eso hay que mirar dos cosas a la vez:

MEDIDA	QUÉ TE DICE	CUÁNDO PREOCUPA
Cobertura empírica	Si el conjunto contiene la respuesta real con la frecuencia prometida.	Si cae por debajo de la cobertura objetivo en evaluación o producción revisada.
Tamaño del conjunto	Cuántas respuestas plausibles deja abiertas.	Si casi todo acaba en conjuntos grandes y el sistema no decide nada.
Cobertura por slice	Si la garantía se sostiene por segmento.	Si `general` cubre bien y `becas` no.
Tasa de abstención	Cuánto trabajo manda a revisión.	Si supera la capacidad humana o hace inútil el producto.

En la práctica, hay variantes que conviene conocer:

VARIANTE	QUÉ INTENTA RESOLVER	CUÁNDO LA MIRARÍA
Split conformal	Separar entrenamiento y calibración para obtener un umbral sencillo.	Primer sistema defendible, fácil de auditar.
Cross-conformal	Usar particiones cruzadas para aprovechar mejor pocos datos.	Dataset pequeño donde separar demasiado duele.
Mondrian o por grupos	Buscar cobertura separada por categorías o slices.	Segmentos con comportamiento distinto: idioma, canal, producto, severidad.
Conformal para regresión	Crear intervalos alrededor de predicciones numéricas.	Tiempos, costes, demanda, riesgo continuo.
Conformal para clasificación	Crear conjuntos de clases plausibles.	Intenciones, rutas, etiquetas, severidades.

MAPIE es una librería Python orientada a cuantificación de incertidumbre y control de riesgo con métodos conformal para regresión, clasificación y series temporales.[21] No hace magia. Te ayuda a implementar patrones conocidos, pero sigues teniendo que decidir qué datos representan producción, qué cobertura necesitas y qué haces cuando el conjunto sale grande.

Un criterio útil para un equipo:

SI OCURRE ESTO	DECISIÓN SENSATA
Cobertura buena y conjuntos pequeños.	Automatizar con monitorización.
Cobertura buena y conjuntos grandes.	El modelo sabe cubrirse, pero no separa suficiente; revisar features, RAG, modelo o tarea.
Cobertura mala en un slice.	No automatizar ese slice y reunir datos.
Cobertura buena en test y mala en producción.	Sospechar dataset shift o cambio de proceso.

La frase que quiero que te lleves: **conformal prediction no sustituye al producto; le da una forma honesta de decir “aquí no sé decidir solo”**.

De probabilidad a decisión: coste, revisión y cobertura

Una probabilidad calibrada no decide sola. Necesita una política.

Para un caso binario, podemos comparar coste esperado de automatizar frente a revisar.

No lo escribo como fórmula porque en este contexto es una política operativa. La idea sí es de decisión bajo coste: si automatizar mal cuesta mucho, el umbral de automatización debe ser más exigente.

PREGUNTA	QUÉ ESTIMAS	EJEMPLO
¿Qué probabilidad calibrada tiene el caso?	Confianza de que la acción automática sea correcta.	0,93.
¿Qué cuesta automatizar mal?	Daño económico, legal, operativo o reputacional.	6 unidades.
¿Qué cuesta revisar?	Tiempo humano, cola, espera y coste de oportunidad.	1,20 unidades.
¿Qué restricciones no son negociables?	Cumplimiento, capacidad, severidad y calidad por slice.	Casos de privacidad no se automatizan.

Automatizar tiene sentido cuando el coste esperado de automatizar mal queda por debajo del coste de revisar, siempre que no viole restricciones de producto, cumplimiento, capacidad o calidad por slice.

En sistemas con zona gris, usamos dos umbrales.

Esta regla convierte una probabilidad calibrada en tres acciones: automatizar negativo, revisar o automatizar positivo. En producción habría que añadir restricciones por slice, capacidad de cola y severidad del caso.

ZONA	CONDICIÓN PRÁCTICA	ACCIÓN
Baja probabilidad	La probabilidad calibrada queda por debajo del umbral bajo.	Automatizar como caso normal si la política lo permite.
Zona gris	La probabilidad queda entre umbral bajo y umbral alto.	Revisar; el sistema no decide solo.
Alta probabilidad	La probabilidad queda por encima del umbral alto.	Automatizar como urgente si no hay restricción de severidad o cumplimiento.

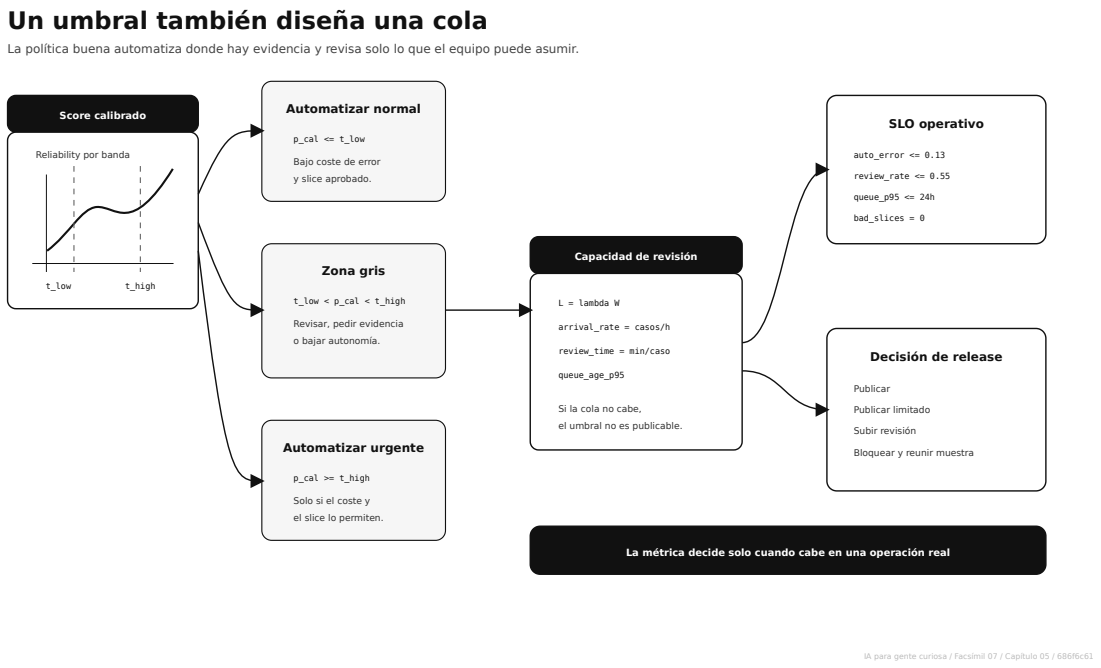
Pero falta una pregunta muy de producción: **¿quién revisa la zona gris?** Si un umbral técnicamente bonito manda 3.000 casos diarios a revisión y el equipo puede revisar 400, ese umbral no es una política. Es una deuda operativa.

Para razonar sobre colas, una ley clásica es la ley de Little. Little demostró en 1961 la relación $L = \lambda W$ para sistemas de colas en régimen estable.[22] No es una fórmula de IA, pero sí es muy útil para no diseñar una zona de revisión imposible.

$$L = \lambda W$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
L	Número medio de casos en el sistema o cola.	180 tickets esperando revisión.
λ	Tasa media de llegada efectiva.	60 tickets por hora entran a revisión.
W	Tiempo medio que un caso pasa en el sistema.	3 horas hasta resolverse.

En palabras: si entran muchos casos a revisión y cada uno tarda mucho, la cola crece. Por eso calibrar umbrales también exige mirar capacidad humana, no solo métricas del modelo.



> La zona gris conecta score calibrado, umbrales, revisión humana, SLO y decisión de release. Si la cola no cabe, la política no está lista.

La política correcta no es la que más automatiza. Es la que automatiza donde tiene evidencia suficiente y deja trazas para mejorar.

En el día a día

Si trabajas con sistemas de IA, calibración aparece de maneras menos limpias que en el ejemplo de manual.

SISTEMA	QUÉ CALIBRARÍA	QUÉ MEDIRÍA
Clasificador de soporte	Score de prioridad.	Brier, ECE, error automático, cola de revisión.
RAG documental	Probabilidad de que la respuesta esté soportada.	Groundedness por banda, abstención correcta, evidencia faltante.
Evaluador automático	Veredicto `pass/fail` o nota por rúbrica.	Acuerdo con referencia humana, ECE por criterio, pases indebidos.
Router de modelos	Probabilidad de que un modelo barato baste.	Calidad por ruta, coste por aceptada, fallback rate.
Agente con tools	Probabilidad de éxito sin revisión.	Error por trayectoria, permisos, latencia, coste y cobertura.

Un patrón útil es guardar siempre estos campos por caso:

```
{
  "case_id": "ticket_1842",
  "raw_score": 0.91,
  "calibrated_probability": 0.76,
  "decision": "revisar",
  "conformal_set": ["normal", "urgente"],
  "slice": "becas",
  "model_version": "support-prioritizer-2026-06-01",
  "calibrator_version": "histogram-v1",
  "policy_version": "support-thresholds-v3"
}
```

Sin esos campos, luego nadie sabe si falló el modelo, el calibrador, el umbral, el slice, el evaluador o la política.

Por qué debería importarte

Hasta aquí parece que calibrar consiste en tomar un score y ajustarlo. En sistemas de IA reales, el problema suele ser más amplio: primero hay que decidir **qué score** merece ser calibrado.

CASO	SCORE TENTADOR	POR QUÉ NO BASTA	QUÉ CALIBRARÍA DE VERDAD
LLM con logprobs	Probabilidad media de tokens.	Una respuesta larga acumula logprob distinto a una corta; token probable no implica respuesta correcta.	Resultado de tarea: respuesta aceptada, cita correcta, formato válido, acción correcta.
RAG	Similitud de embedding o score del reranker.	Similaridad no es soporte factual.	Probabilidad de groundedness o de respuesta soportada por evidencia.
Router de modelos	Score de “modelo barato basta”.	El coste bajo puede esconder más fallback o más revisión.	Probabilidad de salida aceptada sin fallback y coste por aceptada.
Evaluador automático	Nota de rúbrica.	La nota puede estar desplazada por estilo, longitud o versión del modelo.	Acuerdo con referencia humana por criterio y tasa de pases indebidos.
Agente con herramientas	“Éxito” declarado por el agente.	La salida final puede sonar bien aunque la trayectoria falle.	Éxito de run: herramienta correcta, permisos, evidencia, coste y finalización limpia.

Para LLMs, conviene escribir una regla brutalmente clara:

> No calibres la sensación verbal de confianza. Calibra un evento observable.

Ejemplos de eventos observables:

EVENTO CALIBRABLE	ETIQUETA REAL
“La respuesta contiene JSON válido y completo”.	1 si el parser y el contrato pasan.
“La respuesta está soportada por las citas”.	1 si una revisión o validador de evidencia lo confirma.
“El modelo barato basta para este caso”.	1 si pasa la misma eval que el modelo de referencia.
“El agente puede actuar sin revisión”.	1 si la run cumple trayectoria, permisos y resultado.

Esto evita una trampa muy común: convertir una frase como “estoy bastante seguro” en una métrica. Esa frase puede servir para UX, pero no para gates.

LLMs reales: calibrar respuestas, no frases bonitas

En clasificación clásica, el modelo suele devolver un score por clase. En LLMs generativos, la cosa se complica: el modelo produce texto token a token, puede dar varias respuestas distintas que significan lo mismo y puede sonar seguro aunque la evidencia sea débil.

Por eso, para ingeniería, hay que separar tres niveles:

NIVEL	QUÉ MIDE	POR QUÉ IMPORTA
Token	Probabilidad del siguiente token.	Sirve para entender generación, pero no prueba que la respuesta completa sea correcta.
Secuencia	Probabilidad agregada de una salida concreta.	Penaliza longitud y redacción; dos respuestas equivalentes pueden tener probabilidades diferentes.
Evento de tarea	Si la respuesta cumple el contrato.	Es lo que realmente decide producto, operación o evaluación.

Un ejemplo: ante una pregunta documental, el modelo puede responder:

RESPUESTA	TOKENS DISTINTOS	MISMO SIGNIFICADO
“La matrícula cierra el 15 de julio.”	Sí	Sí
“El plazo termina el 15/07.”	Sí	Sí
“La fecha límite es el quince de julio.”	Sí	Sí

Si miramos solo tokens, tratamos esas salidas como objetos diferentes. Si miramos la tarea, las tres dicen lo mismo. Ahí entra la incertidumbre semántica: no preguntamos solo “¿qué tan probable era esta frase exacta?”, sino “¿cuánta dispersión hay entre significados posibles?”.

Una práctica seria con LLMs suele medir al menos esto:

SEÑAL	CÓMO SE MIDE	QUÉ DECISIÓN PERMITE
Validez de formato	Parser, JSON Schema, Pydantic o contrato equivalente.	Reintentar, reparar o rechazar salida.
Soporte documental	Comparación con citas, spans o evidencia recuperada.	Responder, pedir más contexto o revisar.
Consistencia semántica	Varias muestras agrupadas por significado.	Detectar preguntas ambiguas o información insuficiente.
Acuerdo con referencia	Evaluación humana o dataset revisado.	Calibrar score de aceptabilidad.
Coste de fallback	Tokens, latencia y llamadas adicionales.	Decidir cuándo usar modelo grande o revisión.

El punto de ingeniería es incómodo pero liberador: **la incertidumbre útil no vive en una frase de confianza; vive en una variable que puedes medir contra realidad**.

Rigor estadístico mínimo: no publiques un ECE desnudo

ECE es útil, pero no es garantía suficiente. Depende del número de bandas, del tamaño de muestra y de cómo se distribuyen los casos. Con veinte ejemplos puedes fabricar una tabla que parece precisa y, en realidad, solo tiene ruido.

Para no engañarnos, cada política de calibración debería traer tres capas de incertidumbre:

CAPA	QUÉ AÑADE	QUÉ EVITA
Conteo por banda	Cuántos casos sostienen cada punto del reliability diagram.	Concluir demasiado con una banda de 2 casos.
Intervalo de proporción	Rango plausible de accuracy real por banda o slice.	Leer 0,75 como si fuera exacto.
Bootstrap de métricas	Variabilidad aproximada de Brier o ECE al remuestrear.	Comparar mejoras microscópicas como si fueran sólidas.

Para una proporción, un intervalo Wilson es más informativo que enseñar solo el punto medio. Viene del trabajo de Wilson sobre inferencia de proporciones y evita parte de los problemas de intervalos ingenuos con muestras pequeñas.[23] Si en una banda hay 6 aciertos de 8 casos, la accuracy observada es 0,75, pero el intervalo es amplio. No es lo mismo decir “esta banda acierta el 75 %” que decir “he observado 6 de 8; todavía necesito más muestra”.

El bootstrap usa una idea práctica: tomar muchas muestras con reemplazo del conjunto de evaluación, recalcular la métrica y mirar su distribución.[24] No convierte un dataset malo en bueno, pero obliga a ver si una mejora tiene cuerpo o es una fluctuación.

Una revisión profesional debería bloquear o limitar despliegue cuando vea cualquiera de estas señales:

SEÑAL	LECTURA
Bandas con muy pocos casos	El reliability diagram no sostiene decisiones finas.
ECE global baja y slice malo	La media esconde un segmento problemático.
Mejora de Brier menor que su variabilidad bootstrap	No hay evidencia fuerte de mejora.
Base rate distinto entre calibración y evaluación	El calibrador ya nace con deriva posible.
Umbral elegido después de mirar demasiadas variantes	Estás ajustando a la evaluación, no validando.

Riesgo-cobertura: automatizar menos, fallar mejor

Cuando un sistema puede abstenerse, revisar o escalar, no miramos solo accuracy. Miramos la curva riesgo-cobertura: qué error queda cuando automatizamos cierto porcentaje de casos.

Las expresiones siguientes son una forma operativa de la idea de clasificación selectiva: aceptar solo una parte de los casos y medir el error en ese subconjunto. Geifman y El-Yaniv (2017) estudian esta lógica para redes profundas; aquí la traducimos a una política de automatización, revisión y coste.

Definimos una función de aceptación $A_i(c)$, que vale 1 si el caso i se automatiza bajo una política con cobertura objetivo c, y 0 si se revisa:

$$Coverage(c) = \frac{1}{N}\sum_{i=1}^{N} A_i(c)$$

$$Risk(c) = \frac{\sum_{i=1}^{N} \ell_i A_i(c)}{\sum_{i=1}^{N} A_i(c)}$$

SÍMBOLO	SIGNIFICADO	EJEMPLO
$Coverage(c)$	Proporción de casos automatizados.	0,62.
$Risk(c)$	Error medio dentro de los casos automatizados.	0,08.
$A_i(c)$	Indicador de automatización del caso i.	1 si sale de la zona gris.
ℓ_i	Pérdida o coste del caso i.	0 si acierta, 8 si pierde un urgente.
N	Número total de casos evaluados.	500.

En palabras: cobertura dice cuánto automatizas; riesgo dice cuánto error o coste queda dentro de lo que has decidido automatizar.

La lectura profesional es esta:

RESULTADO	DECISIÓN
Alta cobertura y bajo riesgo	Buen candidato para automatización.
Alta cobertura y alto riesgo	El sistema automatiza demasiado.
Baja cobertura y bajo riesgo	Puede servir como primera fase conservadora.
Baja cobertura y alto riesgo	El score no separa bien; no basta con calibrar.

La literatura de clasificación selectiva trabaja precisamente con esta idea: permitir que el modelo responda solo cuando su confianza supera una condición y medir el error en el subconjunto aceptado.[25]

En un producto de IA, esta curva te ayuda a defender frases como:

> “Automatizamos el 40 % de los casos con error automático menor del 13 %, y el resto pasa a revisión porque el conjunto conformal sigue ambiguo”.

Eso es mucho más útil que “el modelo tiene 86 % de accuracy”.

Contrato de calibración en producción

Un calibrador no debería vivir como una función suelta escondida en código. Debería tener un contrato operativo.

CAMPO	PREGUNTA QUE RESPONDE
`model_version`	¿Qué modelo produjo el score bruto?
`score_name`	¿Qué número estamos calibrando exactamente?
`score_semantics`	¿Qué evento observable intenta predecir?
`calibration_dataset_hash`	¿Con qué datos se ajustó?
`evaluation_dataset_hash`	¿Con qué datos se validó?
`calibrator_type`	¿Qué transformación se usó?
`calibrator_version`	¿Qué versión del calibrador está desplegada?
`policy_version`	¿Qué umbrales y costes deciden?
`valid_slices`	¿En qué segmentos se ha medido?
`known_bad_slices`	¿Dónde no debe automatizar?
`recalibration_triggers`	¿Qué cambios obligan a recalibrar?
`owner`	¿Quién responde por esta política?

Un manifest mínimo podría verse así:

```
{
  "model_version": "support-prioritizer-2026-06-01",
  "prompt_version": "ticket-routing-prompt-v4",
  "retrieval_version": "support-docs-index-2026-05-30",
  "score_name": "raw_score",
  "score_semantics": "probabilidad de que el ticket requiera prioridad urgente",
  "calibrator_type": "histogram_laplace",
  "calibrator_version": "cal-ticket-v1",
  "policy_version": "support-thresholds-v3",
  "valid_slices": ["general", "becas", "matricula"],
  "known_bad_slices": [],
  "recalibration_triggers": {
    "model_changed": true,
    "prompt_changed": true,
    "base_rate_relative_change": 0.20,
    "ece_regression": 0.03,
    "new_slice_without_50_cases": true
  },
  "owner": "equipo-ia"
}
```

Este manifest no es burocracia. Es lo que permite revisar una incidencia tres semanas después y saber qué número mandaba, con qué datos se calibró y cuándo dejó de ser fiable.

Documentación profesional: model card, data card y SLO

Una política calibrada no debería quedarse encerrada en un notebook. Si va a afectar a un sistema real, necesita tres documentos vivos:

DOCUMENTO	QUÉ CONTIENE	PREGUNTA QUE RESUELVE
Model card	Modelo, uso previsto, límites, métricas, resultados por slice y cambios relevantes.	“¿Para qué sirve este modelo y dónde no deberíamos usarlo?”
Data card	Origen de datos, composición, etiquetas, cobertura, huecos y transformaciones.	“¿Qué mundo representa este dataset y cuál deja fuera?”
SLO de IA	Objetivos medibles de calidad, revisión, latencia, coste y disponibilidad.	“¿Cuándo decimos que el sistema está suficientemente sano?”

En calibración, esos documentos deberían conectarse así:

PIEZA	CAMPO MÍNIMO
Model card	`model_version`, `score_name`, `score_semantics`, métricas por slice, límites conocidos.
Data card	`dataset_hash`, split, fecha, criterio de etiquetado, distribución por slice.
SLO	`max_auto_error_rate`, `max_review_rate`, `min_auto_coverage`, latencia y coste máximo.
Manifest	Qué combinación exacta de modelo, datos, calibrador y política está aprobada.

Esto no es papeleo académico. Sculley y otros muestran que los sistemas ML acumulan deuda técnica por dependencias ocultas, cambios silenciosos y límites difusos entre componentes. En calibración, una dependencia oculta puede ser un prompt, un índice RAG, un proveedor, una plantilla de salida, un criterio de etiquetado o una cola de revisión.

Una frase útil para equipos:

> Si no puedes decir qué cambió entre dos runs, no puedes decir si el calibrador sigue siendo válido.

El SLO de IA tampoco debería sonar genérico. Debe ser medible:

SLI	SLO POSIBLE
Error automático en casos aceptados	Menor o igual que 18 % en evaluación revisada.
Tasa de revisión	Menor o igual que 60 % con capacidad operativa disponible.
Cobertura automática	Mayor o igual que 40 % sin romper el error automático.
ECE calibrado	Menor o igual que 0,16 en evaluación y revisado por slice.
Latencia de decisión	p95 menor de 1,5 s si no hay revisión.

Si el SLO se rompe, la acción debe estar escrita: limitar automatización, volver a una política anterior, aumentar revisión, recalibrar o bloquear despliegue hasta reunir muestra suficiente.

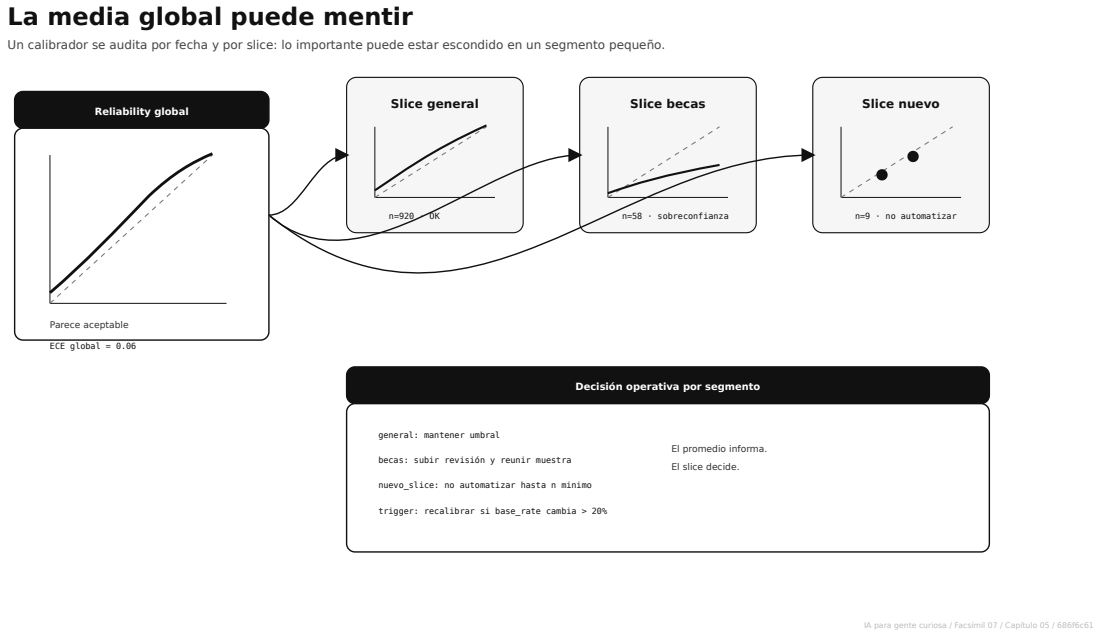
Para ordenar responsabilidades, el NIST AI RMF también es útil porque separa gobernar, mapear, medir y gestionar riesgos de sistemas de IA.[26] En este capítulo no lo usamos como marco legal, sino como recordatorio técnico: medir sin gestionar no cambia el sistema.

Deriva: cuando el calibrador envejece

Un calibrador aprende una relación entre score y frecuencia real en un contexto. Si ese contexto cambia, el calibrador puede seguir ejecutándose y estar equivocado. Eso es especialmente peligroso porque no falla con una excepción: falla con una falsa sensación de control.

La literatura sobre dataset shift describe precisamente el problema de entrenar o evaluar bajo una distribución y usar el sistema bajo otra.[27] En calibración, esa diferencia suele aparecer en tres formas:

CAMBIO	QUÉ CAMBIA	EFECTO SOBRE CALIBRACIÓN
Covariate shift	Cambian las entradas: canal, idioma, longitud, tipo documental.	Las bandas de score pueden mezclar casos que antes no existían.
Label shift o base rate shift	Cambia la proporción real de positivos.	Un umbral que antes revisaba poco puede empezar a aceptar demasiado.
Concept drift	Cambia la relación entre entrada y etiqueta.	El score deja de significar lo mismo aunque su distribución parezca estable.



La calibración se revisa por segmentos. Un promedio global aceptable puede esconder un slice sobreconfiado o sin muestra suficiente.

La decisión técnica no debería ser “recalibrar todo” cada vez que algo se mueve. A veces basta con bloquear un slice nuevo, subir revisión en un canal, recoger más etiquetas o volver temporalmente a un umbral conservador. Lo importante es que la política diga qué señal dispara qué acción.

Monitorización: cuándo recalibrar

La calibración no es una ceremonia de una vez. Se vigila.

SEÑAL ONLINE	QUÉ INDICA	ACCIÓN
Cambia el base rate	La proporción real de positivos ya no se parece a calibración.	Recalcular reliability diagram por fecha y slice.
Sube ECE en muestra revisada	El score ya no corresponde a frecuencia real.	Recalibrar o limitar automatización.
Crece la cola de revisión	La zona gris está absorbiendo demasiados casos.	Revisar umbrales, capacidad o calidad del modelo.
Aumentan errores automáticos	La política acepta casos que antes separaba bien.	Bajar cobertura automática y abrir análisis de regresión.
Aparece un slice nuevo	No hay evidencia para automatizar ese segmento.	Revisar hasta reunir muestra mínima.
Cambia modelo, prompt, RAG o herramienta	Cambió el sistema que produce el score.	Invalidar calibrador anterior salvo prueba contraria.

Para ingenieros de IA, el patrón operativo sano es:

1. Versionar modelo, prompt, datos, calibrador y política.
2. Mantener una muestra revisada de producción.
3. Medir Brier, ECE, error automático y revisión por slice.
4. Tener una acción automática si el calibrador caduca: limitar automatización, subir revisión o volver a política anterior.

Si no hay acción asociada, la métrica es decoración.

Herramientas que sí usaría con criterio

No hace falta empezar con una plataforma enorme. Hace falta saber qué problema tienes. Las herramientas son útiles cuando acompañan una decisión concreta: calibrar probabilidades, construir intervalos, vigilar drift o dejar evidencias.

HERRAMIENTA	PARA QUÉ SIRVE AQUÍ	CUÁNDO LA USARÍA	LÍMITE
scikit-learn	`CalibratedClassifierCV`, curvas de calibración y calibradores clásicos.	Clasificación tabular o pipeline Python clásico.	No sustituye una política de producto ni monitoriza producción sola.
netcal	Medir y mitigar miscalibration en estimaciones de confianza.	Redes neuronales o experimentos donde quieres más métodos de calibración.	Exige entender qué score estás calibrando.
MAPIE	Prediction intervals, prediction sets y conformal prediction.	Cuando necesitas cuantificar incertidumbre con conjuntos o intervalos.	La garantía depende de datos representativos y supuestos de intercambio.
Evidently	Evaluación, tests y monitorización de drift/datos/modelos.	Para vigilar producción, comparar datasets y generar reportes operativos.	Detectar drift no decide automáticamente qué acción tomar.

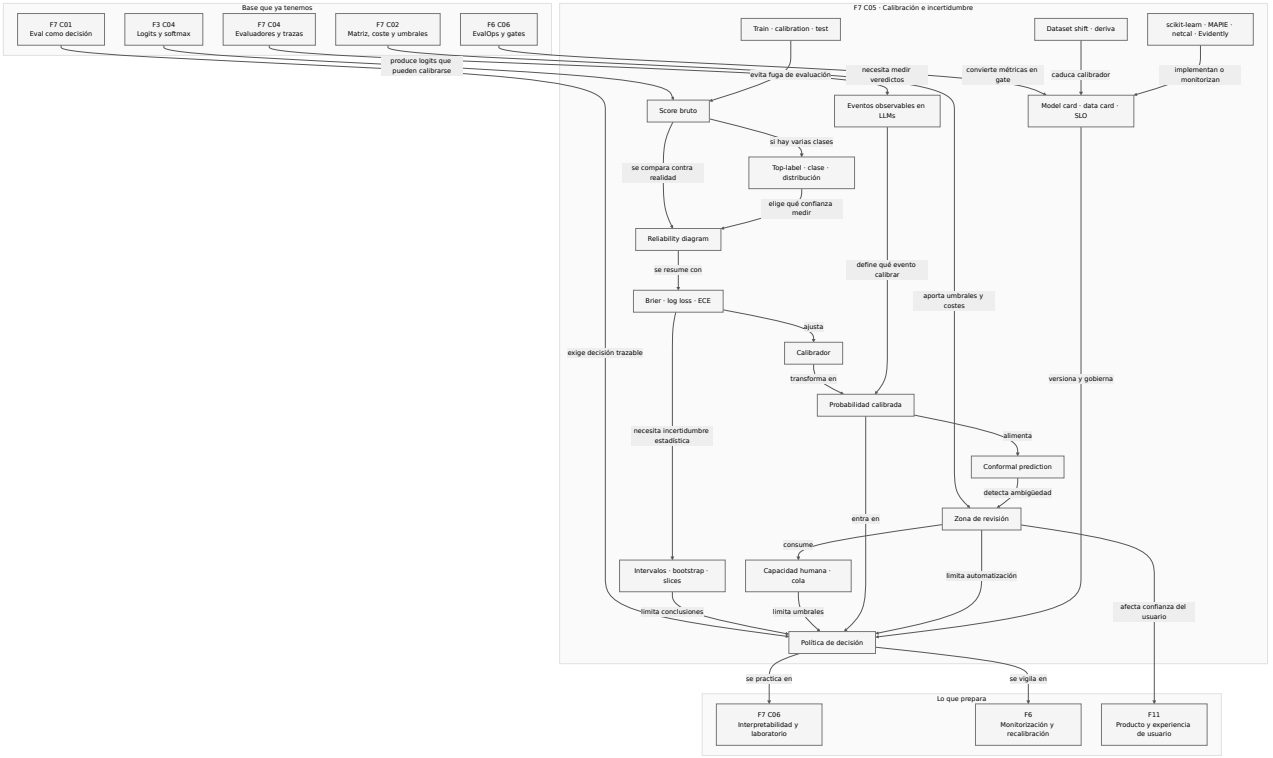
netcal se presenta como framework para medir y mitigar miscalibration de estimaciones de confianza.[28] Evidently documenta métricas y presets para detectar cambios en distribución de datos, además de monitorización de sistemas de ML y LLM.[29]

El orden sensato para un alumno o equipo pequeño sería:

1. Empezar con CSV, script y manifest como en el cuaderno del facsímil.
2. Si el modelo es clásico, probar calibración con scikit-learn y comparar con la implementación propia.
3. Si necesitas intervalos o conjuntos, estudiar MAPIE y contrastarlo con el split conformal explicado aquí.
4. Si ya tienes producción, añadir monitorización de drift y reportes con una herramienta como Evidently.
5. Si la calibración es central en el producto, documentar versión de calibrador, dataset y política como artefactos de release.

La herramienta buena no es la más completa. Es la que te permite contestar: qué score calibré, con qué datos, qué cambió, qué decisión toma y cuándo deja de ser fiable.

Cómo encaja todo



Vocabulario aprendido

TÉRMINO	DEFINICIÓN BREVE
Score bruto	Puntuación salida del modelo antes de calibrar.
Probabilidad calibrada	Score interpretable como frecuencia esperada de acierto.
Discriminación	Capacidad de ordenar casos positivos por encima de negativos.
Calibración	Correspondencia entre confianza predicha y frecuencia real.
Brier score	Error cuadrático medio de probabilidades.
Log loss	Pérdida que castiga mucho equivocarse con confianza alta.
ECE	Error de calibración esperado por bandas.
Reliability diagram	Gráfico de confianza media frente a accuracy por banda.
Platt scaling	Calibración sigmoidal de un score.
Isotonic regression	Calibración monótona por tramos.
Conjunto de calibración	Partición reservada para ajustar calibradores o cuantiles sin tocar el test final.
Top-label calibration	Calibración de la confianza de la clase predicha.
Temperature scaling	Ajuste de logits con una temperatura aprendida.
Conformal prediction	Construcción de conjuntos o intervalos con cobertura objetivo.
Cobertura	Proporción de casos donde el conjunto contiene la respuesta correcta.
Score de no conformidad	Medida de rareza que decide si una clase o predicción entra en el conjunto conformal.
Calibración por slice	Medición o ajuste de calibración por segmento operativo.
Zona de revisión	Banda donde el sistema no automatiza por incertidumbre.
Riesgo-cobertura	Curva que compara porcentaje automatizado y error en lo automatizado.
Deriva de calibración	Cambio de la relación entre score y frecuencia real.
Dataset shift	Cambio entre distribución de entrenamiento, evaluación o producción.

TÉRMINO	DEFINICIÓN BREVE
Capacidad de revisión	Volumen de casos que el equipo humano puede revisar sin romper SLO.
Manifest de calibración	Contrato versionado de score, calibrador, política, datos y triggers.
Incertidumbre semántica	Duda sobre el significado de una respuesta, no solo sobre su texto exacto.
Intervalo Wilson	Intervalo para una proporción observada, útil con muestras pequeñas.
Bootstrap	Remuestreo con reemplazo para estimar variabilidad de métricas.
Model card	Documento de modelo con uso previsto, límites y métricas por segmento.
Data card	Documento de datos con origen, composición, huecos y uso previsto.
SLO de IA	Objetivo medible de calidad, coste, revisión, latencia o disponibilidad.

Dónde solía tropezar yo

TROPIEZO	POR QUÉ OCURRE	ANTÍDOTO
Leer cualquier score como probabilidad	El número parece probabilístico aunque solo ordene.	Medir calibración antes de automatizar con umbrales.
Mirar solo accuracy	Puedes acertar mucho y estar sobreconfiado.	Añadir Brier, log loss, ECE y reliability diagram.
Calibrar con el test final	El resultado queda contaminado por decisiones de ajuste.	Separar train, calibration y evaluation.
Usar ECE como número absoluto	Depende de bandas y puede esconder slices malos.	Mirar curva, slices y casos frontera.
Automatizar la zona gris	La presión por reducir revisión empuja a decidir donde falta señal.	Diseñar revisión como parte del sistema, no como fracaso.
Olvidar que la calibración caduca	Cambian datos, modelo, prompt, retrieval o usuarios.	Versionar calibrador y recalibrar con monitorización.
Confundir logprob con verdad	Un token probable no garantiza una respuesta correcta.	Calibrar eventos observables de tarea.
Publicar sin intervalos	Una métrica puntual puede parecer más estable de lo que es.	Añadir Wilson, bootstrap y mínimos por slice.

Antes de pasar página

Antes de avanzar, deberías poder responder:

1. ¿Por qué un score alto no implica probabilidad calibrada?
2. ¿Qué diferencia hay entre discriminación y calibración?
3. ¿Cómo se calcula Brier score y qué penaliza?
4. ¿Por qué log loss castiga tanto una predicción confiada que falla?
5. ¿Qué mide ECE y por qué depende de las bandas?
6. ¿Cómo leerías un reliability diagram por debajo de la diagonal?
7. ¿Cuándo usarías temperature scaling frente a isotonic regression?
8. ¿Por qué no se ajusta el calibrador con el test final?
9. ¿Qué diferencia hay entre calibrar top-label y calibrar por clase?

10. ¿Qué supuesto sostiene conformal prediction?
11. ¿Qué significa que un conjunto conformal tenga dos clases?
12. ¿Por qué cobertura alta no implica utilidad alta?
13. ¿Por qué la curva riesgo-cobertura es más útil que una accuracy global para decidir automatización?
14. ¿Cómo afecta la capacidad de revisión a los umbrales?
15. ¿Qué debería contener un manifest de calibración?
16. ¿Por qué no basta con mirar logprobs para calibrar una respuesta de LLM?
17. ¿Qué aporta un intervalo Wilson en una banda pequeña?
18. ¿Qué diferencia hay entre model card, data card y manifest de calibración?
19. ¿Qué señales de dataset shift obligarían a limitar o recalibrar?
20. ¿Qué herramienta usarías para calibrar, conformalizar o monitorizar y por qué?
21. ¿Qué SLO de IA escribirías para decidir si esta política puede desplegarse?
22. ¿Qué archivos entrega la práctica del capítulo?

En resumen

IDEA	QUÉ TE LLEVAS
Un score no es automáticamente una probabilidad.	Primero mide si su confianza coincide con frecuencias reales.
Calibrar no es subir accuracy.	Es hacer que el número sea útil para decidir bajo coste.
El split importa tanto como la métrica.	Train, calibration, test y producción revisada no cumplen el mismo papel.
Multiclase exige mirar más que la clase ganadora.	Top-label puede orientar, pero los slices y clases críticas deciden.
Brier, log loss, ECE y reliability diagram se complementan.	Ninguna métrica aislada basta para publicar una política.
Conformal prediction convierte incertidumbre en conjuntos o intervalos.	Si el conjunto es ambiguo, el sistema debe revisar o abstenerse.
Cobertura no es utilidad.	Un conjunto enorme puede cubrir mucho y decidir poco.
La zona gris consume operación.	Si revisión humana no cabe, el umbral no es viable.
En LLMs se calibran eventos, no frases de confianza.	El evento debe ser observable: formato válido, respuesta soportada, acción correcta o salida aceptada.
La calibración es operativa.	Versiona calibrador, umbrales, política, datos, SLOs, herramientas y monitorización porque todo eso caduca.

Para saber más

Angelopoulos, A. N. y Bates, S. (2021). A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification. *arXiv*. https://arxiv.org/abs/2107.07511

Brier, G. W. (1950). Verification of Forecasts Expressed in Terms of Probability. *Monthly Weather Review*, 78(1), 1-3. https://doi.org/10.1175/1520-0493(1950)078<0001:VOEPIO>2.0.CO;2

Breck, E., Cai, S., Nielsen, E., Salib, M. y Sculley, D. (2017). The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction. *IEEE Big Data*, 1123-1132. https://research.google/pubs/pub46555/

Efron, B. (1979). Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1), 1-26. https://doi.org/10.1214/aos/1176344552

Evidently AI. (2026). *Data Drift Documentation*. https://docs.evidentlyai.com/metrics/explainer_drift

Geifman, Y. y El-Yaniv, R. (2017). Selective Classification for Deep Neural Networks. *Advances in Neural Information Processing Systems*. https://proceedings.neurips.cc/paper/2017/hash/4a5cfa9281924139db466a8a19291aff-Abstract.html

Gneiting, T. y Raftery, A. E. (2007). Strictly Proper Scoring Rules, Prediction, and Estimation. *Journal of the American Statistical Association*, 102(477), 359-378. https://doi.org/10.1198/016214506000001437

Guo, C., Pleiss, G., Sun, Y. y Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. *Proceedings of the 34th International Conference on Machine Learning*, 70, 1321-1330. https://proceedings.mlr.press/v70/guo17a.html

Gupta, C. y Ramdas, A. (2021). *Top-label Calibration and Multiclass-to-Binary Reductions*. Workshop on Uncertainty and Robustness in Deep Learning. https://www.gatsby.ucl.ac.uk/~balaji/udl2021/accepted-papers/UDL2021-paper-060.pdf

Jones, C., Wilkes, J., Murphy, N. y Smith, C. (2016). Service Level Objectives. En *Site Reliability Engineering*. https://sre.google/sre-book/service-level-objectives/

Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Hatfield-Dodds, Z., DasSarma, N., Tran-Johnson, E., Johnston, S. y otros. (2022). *Language Models (Mostly) Know What They Know*. arXiv. https://arxiv.org/abs/2207.05221

Kuhn, L., Gal, Y. y Farquhar, S. (2023). Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. *International Conference on Learning Representations*. https://arxiv.org/abs/2302.09664

Little, J. D. C. (1961). A Proof for the Queuing Formula: L = lambda W. *Operations Research*, 9(3), 383-387. https://doi.org/10.1287/opre.9.3.383

MAPIE. (2026). *MAPIE: Model Agnostic Prediction Interval Estimator*. https://mapie.readthedocs.io/

Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D. y Gebru, T. (2019). Model Cards for Model Reporting. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 220-229. https://doi.org/10.1145/3287560.3287596

Murphy, A. H. (1973). A New Vector Partition of the Probability Score. *Journal of Applied Meteorology*, 12(4), 595-600. https://doi.org/10.1175/1520-0450(1973)012<0595:ANVPOT>2.0.CO;2

Naeini, M. P., Cooper, G. F. y Hauskrecht, M. (2015). Obtaining Well Calibrated Probabilities Using Bayesian Binning. *AAAI*. https://ojs.aaai.org/index.php/AAAI/article/view/9602

netcal. (2026). *API Reference of netcal*. https://efs-opensource.github.io/calibration-framework/build/html/index.html

Niculescu-Mizil, A. y Caruana, R. (2005). Predicting Good Probabilities with Supervised Learning. *Proceedings of the 22nd International Conference on Machine Learning*, 625-632. https://doi.org/10.1145/1102351.1102430

Platt, J. C. (1999). Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. En *Advances in Large Margin Classifiers*. MIT Press.

Pushkarna, M., Zaldivar, A. y Kjartansson, O. (2022). *Data Cards: Purposeful and Transparent Dataset Documentation for Responsible AI*. arXiv. https://arxiv.org/abs/2204.01075

Quiñonero-Candela, J., Sugiyama, M., Schwaighofer, A. y Lawrence, N. D. (Eds.). (2009). *Dataset Shift in Machine Learning*. MIT Press. https://mitpress.mit.edu/9780262170055/dataset-shift-in-machine-learning/

scikit-learn. (2026). *Probability Calibration*. https://scikit-learn.org/stable/modules/calibration.html

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J. F. y Dennison, D. (2015). Hidden Technical Debt in Machine Learning Systems. *Advances in Neural Information Processing Systems*. https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems

Shafer, G. y Vovk, V. (2008). A Tutorial on Conformal Prediction. *Journal of Machine Learning Research*, 9, 371-421. https://www.jmlr.org/papers/v9/shafer08a.html

Tabassi, E. (2023). *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.100-1

Vovk, V., Gammerman, A. y Shafer, G. (2005). *Algorithmic Learning in a Random World*. Springer. https://doi.org/10.1007/b106715

Wilson, E. B. (1927). Probable Inference, the Law of Succession, and Statistical Inference. *Journal of the American Statistical Association*, 22(158), 209-212. https://doi.org/10.1080/01621459.1927.10502953

Notas

1. Brier, G. W. (1950). Verification of Forecasts Expressed in Terms of Probability. *Monthly Weather Review*, 78(1), 1-3. https://doi.org/10.1175/1520-0493(1950)078<0001:VOEPIO>2.0.CO;2 ↩

2. Murphy, A. H. (1973). A New Vector Partition of the Probability Score. *Journal of Applied Meteorology*, 12(4), 595-600. https://doi.org/10.1175/1520-0450(1973)012<0595:ANVPOT>2.0.CO;2 ↩

3. Niculescu-Mizil, A. y Caruana, R. (2005). Predicting Good Probabilities with Supervised Learning. *ICML*. https://doi.org/10.1145/1102351.1102430 ↩

4. Platt, J. C. (1999). Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. En *Advances in Large Margin Classifiers*. MIT Press. ↩

5. Guo, C., Pleiss, G., Sun, Y. y Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. *ICML*. https://proceedings.mlr.press/v70/guo17a.html ↩

6. Vovk, V., Gammerman, A. y Shafer, G. (2005). *Algorithmic Learning in a Random World*. Springer. https://doi.org/10.1007/b106715 ↩

7. Shafer, G. y Vovk, V. (2008). A Tutorial on Conformal Prediction. *Journal of Machine Learning Research*, 9, 371-421. https://www.jmlr.org/papers/v9/shafer08a.html ↩

8. Angelopoulos, A. N. y Bates, S. (2021). A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification. arXiv. https://arxiv.org/abs/2107.07511 ↩

9. scikit-learn. (2026). *Probability Calibration*. https://scikit-learn.org/stable/modules/calibration.html. Consultado el 1 de junio de 2026. ↩

10. Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Hatfield-Dodds, Z., DasSarma, N., Tran-Johnson, E., Johnston, S. y otros. (2022). *Language Models (Mostly) Know What They Know*. arXiv. https://arxiv.org/abs/2207.05221 ↩

11. Kuhn, L., Gal, Y. y Farquhar, S. (2023). Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. *ICLR*. https://arxiv.org/abs/2302.09664 ↩

12. Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D. y Gebru, T. (2019). Model Cards for Model Reporting. *FAT*, 220-229. https://doi.org/10.1145/3287560.3287596 ↩

13. Pushkarna, M., Zaldivar, A. y Kjartansson, O. (2022). *Data Cards: Purposeful and Transparent Dataset Documentation for Responsible AI*. arXiv. https://arxiv.org/abs/2204.01075 ↩

14. Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J. F. y Dennison, D. (2015). Hidden Technical Debt in Machine Learning Systems. *NeurIPS*. https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems ↩

15. Breck, E., Cai, S., Nielsen, E., Salib, M. y Sculley, D. (2017). The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction. *IEEE Big Data*. https://research.google/pubs/pub46555/ ↩

16. Jones, C., Wilkes, J., Murphy, N. y Smith, C. (2016). Service Level Objectives. En *Site Reliability Engineering*. https://sre.google/sre-book/service-level-objectives/ ↩

17. Guo y otros, 2017. ↩

18. scikit-learn. (2026). Probability calibration. https://scikit-learn.org/stable/modules/calibration.html ↩

19. MAPIE. (2026). The conformalization calibration set. https://mapie.readthedocs.io/en/stable/split_cross_conformal.html ↩

20. Gupta, C. y Ramdas, A. (2021). Top-label Calibration and Multiclass-to-Binary Reductions. https://www.gatsby.ucl.ac.uk/~balaji/udl2021/accepted-papers/UDL2021-paper-060.pdf ↩

21. MAPIE. (2026). MAPIE: Model Agnostic Prediction Interval Estimator. https://mapie.readthedocs.io/ ↩

22. Little, J. D. C. (1961). A Proof for the Queuing Formula: L = lambda W. *Operations Research*, 9(3), 383-387. https://doi.org/10.1287/opre.9.3.383 ↩

23. Wilson, E. B. (1927). Probable Inference, the Law of Succession, and Statistical Inference. *Journal of the American Statistical Association*, 22(158), 209-212. https://doi.org/10.1080/01621459.1927.10502953 ↩

24. Efron, B. (1979). Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1), 1-26. https://doi.org/10.1214/aos/1176344552 ↩

25. Geifman, Y. y El-Yaniv, R. (2017). Selective Classification for Deep Neural Networks. *NeurIPS*. https://proceedings.neurips.cc/paper/2017/hash/4a5cfa9281924139db466a8a19291aff-Abstract.html ↩

26. Tabassi, E. (2023). *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.100-1 ↩

27. Quiñonero-Candela, J., Sugiyama, M., Schwaighofer, A. y Lawrence, N. D. (eds.). (2009). *Dataset Shift in Machine Learning*. MIT Press. https://mitpress.mit.edu/9780262170055/dataset-shift-in-machine-learning/ ↩

28. netcal. (2026). API Reference of netcal. https://efs-opensource.github.io/calibration-framework/build/html/index.html ↩

29. Evidently AI. (2026). Data Drift Documentation. https://docs.evidentlyai.com/metrics/explainer_drift ↩

Cuadernos para practicar

Escanea el código con el móvil para abrir el cuaderno en Google Colab y ejecutarlo sin instalar nada.

Calibración: cuando el modelo dice «90%», ¿es un 90%?

https://colab.research.google.com/github/686f6c61/iaparagentecuriosa-notebooks/blob/main/Facs%C3%ADmil%2007/05-calibracion-probabilidades.ipynb