Soft sensors robustos, explicables y operables: Diseño para monitorización de calidad en planta industrial

Enfoque integral de desarrollo de sensores virtuales con fiabilidad y despliegue en el *edge*

Mariano Millañanco Fernández
Puertollano. Ciudad Real, España
Universidad de Sevilla — Máster en Microelectrónica
UTAMED — Máster en Inteligencia Artificial
github.com/tangodelta217/ACQC
mariano.millananco@gmail.com

2026

Resumen

El presente documento aborda el desarrollo de *soft sensors* (sensores virtuales) para la estimación en línea de variables de calidad en procesos industriales, enfatizando que sean robustos, explicables y operables en planta. Se definen las variables objetivo y las señales de entrada disponibles, detallando la preparación de datos (*alineamiento* temporal, limpieza y gestión de valores atípicos) necesaria antes del modelado. A continuación, se describen los modelos candidatos, desde enfoques multivariantes clásicos (p. ej., regresión por mínimos cuadrados parciales, PLS) hasta algoritmos modernos de aprendizaje automático (bosques aleatorios, redes neuronales profundas), junto con criterios de selección orientados a un compromiso entre interpretabilidad y precisión. Se incluyen mecanismos para cuantificar la incertidumbre de las predicciones y calibrarlas con datos reales, además de detectar derivas de concepto u otras condiciones anómalas (*out-of-distribution*) que puedan degradar el rendimiento. Se establecen estrategias de respaldo (*fallback*) ante degradación del modelo o fallos de sensores, asegurando continuidad en la monitorización. Finalmente, se describe el plan de evaluación tanto offline (validación con históricos) como en el entorno *edge* (dispositivo dedicado en planta), incluyendo métricas de desempeño, latencia, uso de recursos, y la entrega de artefactos documentales (tarjeta de modelo) y técnicos (modelo exportado en ONNX, scripts de despliegue).

Abstract

This paper presents the development of *soft sensors* for real-time estimation of quality variables in industrial processes, emphasizing robust, explainable, and operable design. The target quality variables and available input signals are defined, detailing data preparation (time alignment, cleaning, and outlier handling) required prior to modeling. Next, candidate models are described, ranging from classical multivariate approaches (e.g., partial least squares regression, PLS) to modern machine learning algorithms (random forests, deep neural networks), along with selection criteria balancing interpretability and accuracy. Mechanisms are included to quantify prediction uncertainty and calibrate outputs with real data, as well as to detect concept drift or other out-of-distribution conditions that could degrade performance. Fallback strategies are established to handle model degradation or sensor failures, ensuring continuity in monitoring. Finally, the plan for both offline evaluation (historical data validation) and edge deployment (on-site device such as an NVIDIA Jetson) is outlined, covering performance metrics, latency, resource usage, and the delivery of documentation artifacts (model card) and technical assets (model exported to ONNX, deployment scripts).

Keywords: soft sensors; sensores virtuales; deriva de concepto; computación en el edge

Resumen ejecutivo (entregables medibles)

- Desarrollo de un sensor virtual robusto para variable de calidad crítica, con precisión esperada $R^2 > 0{,}90$ en validación offline (error MAPE bajo 5 %).
- Implementación de un pipeline híbrido (PLS como modelo base y red LSTM avanzada) para equilibrar interpretabilidad y exactitud, validado mediante validación cruzada.
- Despliegue del modelo en dispositivo *edge* NVIDIA Jetson optimizado (exportado a ONNX y acelerado con TensorRT), logrando latencia de inferencia <100 ms por muestra.
- Mecanismo de detección de deriva de concepto mediante índice de estabilidad poblacional (PSI) que activa re-entrenamiento automático cuando el rendimiento cae bajo el umbral definido.
- Estrategia de respaldo integrada: ante fallo de algún sensor de entrada o alta incertidumbre en la predicción, el sistema alerta al operador y conmuta a medición de laboratorio o modelo simplificado mientras se re-calibra el modelo principal.
- Entrega de documentación completa: *Model card* detallada del sensor virtual (capacidades, limitaciones, datos de entrenamiento) **Mitchell2019**, junto con el modelo final en formato ONNX y guías de integración para su operación en planta.

1. Introducción

Los *soft sensors* o sensores virtuales son modelos matemáticos que estiman variables de proceso difíciles de medir en tiempo real a partir de otras mediciones más accesibles **Kadlec2009**. Su principal aplicación en la industria de procesos es obtener *predicciones instantáneas* de parámetros de calidad del producto (p.ej. pureza, viscosidad, índice de fluidez) que normalmente requieren análisis de laboratorio lentos. Esto permite un monitoreo continuo y decisiones operativas más ágiles. De hecho, el uso de sensores virtuales se ha popularizado en los últimos años gracias a la abundancia de datos históricos disponibles y al aumento del poder de cómputo industrial.

Sin embargo, desarrollar un soft sensor robusto no es trivial. Los datos de planta suelen ser *abundantes en volumen pero pobres en información útil*: contienen ruido, valores atípicos y a menudo carecen de cobertura de todos los estados futuros del proceso. Además, los procesos químicos exhiben *comportamiento no estacionario*, cambiando con el tiempo por múltiples causas: cambios en materias primas, ensuciamiento de equipos, degradación de catalizadores, variaciones ambientales estacionales, etc. **Kadlec2011**. Como consecuencia, un modelo estático entrenado con datos históricos puede *perder precisión gradualmente* tras su despliegue **Bakirov2017**. Para que el sensado virtual sea realmente operable en planta, debe por tanto ser **robusto** frente a estos cambios (detectando y adaptándose a la deriva del proceso), **explicable** para generar confianza en operadores e ingenieros, y **fácilmente integrable** en la infraestructura (**operable** en sistemas de control en planta).

Este documento presenta un enfoque integral para el diseño de un sistema de soft sensing que cumpla con esas características. En las secciones siguientes se definen las variables objetivo del sensor virtual (y su relevancia), los datos de entrada considerados, y las técnicas de preprocesamiento para preparar dichos datos. Se discuten los modelos candidatos, desde métodos estadísticos lineales hasta algoritmos de *machine learning* de última generación, así como criterios para seleccionar el modelo óptimo equilibrando exactitud y explicabilidad. Posteriormente, se abordan los mecanismos adicionales incorporados al diseño: calibración de la incertidumbre de la predicción, detección de derivas u otras situaciones anómalas, y planes de contingencia o *fallback*. Finalmente, se describe la evaluación planificada tanto en entorno offline (pruebas con datos históricos) como en el *edge* (dispositivo de cómputo local en la planta), junto con los artefactos resultantes (exportación del modelo y documentación). Con ello, se busca asegurar que el sensor virtual propuesto será fiable, trazable y útil en la operación industrial cotidiana.

2. Variables objetivo

Las **variables objetivo** son los parámetros de calidad o rendimiento que el soft sensor estimará en tiempo real. Se eligen típicamente variables que: (1) tienen alta relevancia operativa (p. ej. especificaciones de producto, eficiencia de reacción), (2) son difíciles de medir de forma continua (requieren análisis de laboratorio o instrumentación compleja) y (3) presentan variabilidad que se desea monitorizar para optimizar el proceso. En este caso de estudio, se enfocan variables de calidad como el índice de fluidez (MFI) de un polímero, la viscosidad de un producto o la pureza/química de una corriente, las cuales solo se obtienen por muestras de laboratorio cada ciertas horas. Por ejemplo, en la literatura se han desarrollado sensores virtuales para estimar la *viscosidad Mooney* en procesos de mezcla de caucho **Zhang2022** o la concentración de penicilina en fermentaciones bioquímicas **Metcalfe2025**, variables cuyo muestreo directo es intermitente y retardado.

Cada variable objetivo debe estar claramente definida operativamente (incluyendo unidades y métodos de referencia de laboratorio para su medición). Por ejemplo, si la variable es “número de acidez” de un producto, se debe especificar cómo se mide en el laboratorio (norma aplicable, método de titulación, etc.) para asegurar consistencia en la calibración del modelo. Además, conviene determinar el rango normal de valores y los umbrales críticos para la operación. Esta información suele compilarse en una *lista maestra de variables de calidad objetivo* disponible para todo el equipo del proyecto, evitando ambigüedades.

Importancia operativa: Conocer estas variables en tiempo real permite ajustar setpoints o tomar acciones de control de forma proactiva, en lugar de esperar horas por resultados del laboratorio. Por ejemplo, si el sensor virtual predice que la viscosidad está acercándose fuera de especificación, un operador podría moderar la temperatura de reacción o cambiar el flujo de insumos para corregir la trayectoria. En modo *TFM* (operación asistida, con humano en el ciclo de control), el soft sensor serviría como herramienta de apoyo a la decisión, no para control automático cerrado sin validación humana.

Limitaciones: Es importante notar que si no existe variabilidad suficiente o datos históricos representativos de la variable objetivo (por ejemplo, un contaminante que casi siempre está por debajo del límite de detección), un modelo de soft sensor tendrá dificultad para generalizar. El alcance del proyecto se limita explícitamente a variables objetivo para las cuales hay datos históricos suficientes y variabilidad observable; no se “promete” modelar variables sin datos o cuyo comportamiento esté fuera del rango conocido. Cualquier extrapolación más allá de los límites explícitos se considerará fuera de alcance.

3. Señales de entrada

Las **señales de entrada** son los *tags* o mediciones disponibles en planta que el modelo utilizará para inferir la variable de calidad. Estas pueden incluir variables de proceso (temperaturas, presiones, caudales, niveles, concentraciones medidas en línea), variables de contexto (estado de equipos, modo de operación, datos de calidad de materia prima) e incluso espectros o señales complejas si se cuenta con analizadores en línea (p. ej. espectros NIR). El conjunto de entradas se determina analizando la correlación de cada señal con la variable objetivo y considerando el conocimiento de los mecanismos fisicoquímicos involucrados.

Es crucial construir un *diccionario de señales* que liste cada tag utilizado: nombre descriptivo, unidades, rango esperado, frecuencia de muestreo, fuente (PLC, DCS o

historiador de planta) y cualquier indicación de calidad o fiabilidad del sensor físico. Por ejemplo, si se usa la lectura de un caudalímetro, anotar si este tiene compensación de temperatura, o si cierta sonda de temperatura es propensa a ensuciarse. Esta documentación unificada (Single Source of Truth) previene inconsistencias entre documentos técnicos.

Selección de variables: Idealmente se realiza primero un análisis exploratorio de datos (AED), calculando coeficientes de correlación y usando métodos de reducción de dimensionalidad (PCA) para identificar qué sensores aportan información relevante. Técnicas de selección de características (*feature selection*) pueden aplicarse para descartar variables redundantes o poco informativas. En casos donde el número de candidatos es grande, se pueden usar métodos automáticos (como *random forests* o Lasso) para evaluar importancia de variables. La literatura reciente también propone integrar múltiples fuentes mediante *data fusion*; por ejemplo, combinar señales de distintos sistemas con análisis de correlaciones para mejorar la capacidad predictiva **Wu2021**.

Espectro temporal: Dado que muchos procesos tienen dinámicas, no solo importan los valores instantáneos de las señales sino su comportamiento en el tiempo. Por tanto, suele ser necesario incluir variables retrasadas (lags) o calculadas (p. ej. promedios móviles, derivadas) para capturar efectos de tiempo de residencia, inercias térmicas, etc. En este diseño, se contemplará la posibilidad de agregar como entradas valores históricos cercanos (ventanas de tiempo) de ciertas variables clave para que el modelo aprenda las relaciones dinámicas. Esto resulta especialmente relevante si se usan modelos secuenciales (como redes LSTM) que pueden aprovechar dichas secuencias.

Calidad de datos de entrada: Un aspecto crítico es la fiabilidad de los sensores físicos. Si alguna entrada proviene de un instrumento con fallas frecuentes o calibración dudosa, debe manejarse con precaución (p. ej. descartar sus datos cuando se detecten señales de mal funcionamiento). Incluso se puede implementar lógica para ignorar/adaptar temporalmente el modelo ante una entrada claramente errónea (ver sección de *fallback*). Un caso mencionado en la bibliografía es evitar que el sensor virtual se adapte o entrene con datos provenientes de un sensor físico defectuoso, para no degradar el modelo **Bakirov2017**.

4. Preparación de datos

Antes de alimentar cualquier algoritmo de modelado, se requiere una cuidadosa **preparación de los datos**. Esto incluye varias tareas secuenciales:

Alineamiento temporal: Dado que las variables objetivo suelen provenir de laboratorio con frecuencias bajas (una muestra cada X horas) y las señales de entrada son típicamente de alta frecuencia (minutos o segundos), es necesario alinear ambos tipos de datos en el tiempo. Se construye un conjunto de instancias (ejemplos de entrenamiento) donde a cada valor de laboratorio $y(t)$ se le asignan las lecturas de las señales x_i cercanas a ese instante. Puede requerirse tomar el promedio de las señales en una ventana alrededor del muestreo para suavizar ruido de alta frecuencia, o elegir un retardo óptimo si se sospecha que la variable de calidad responde con cierto *lag* al cambio de las variables de proceso. Este retardo puede determinarse examinando correlaciones cruzadas en los datos históricos.

Limpieza y filtrado: Se eliminan o corrigen registros con datos faltantes (*missing*) o claramente erróneos. Los datos faltantes en sensores de entrada se pueden imputar con interpolaciones si son brechas cortas, o marcar la instancia como no utilizable si la pérdida es significativa. Los valores atípicos (outliers) se detectan mediante reglas (p. ej. fuera de rangos físicos posibles) o métodos estadísticos (p. ej. usando el criterio de 3σ o métodos robustos). Si su ocurrencia es esporádica, pueden eliminarse o reemplazarse por un valor plausible (interpolación temporal o mediana local). En cambio, si ciertas entradas tienen datos de baja calidad de forma persistente, se reconsidera su inclusión como variable predictora.

También se aplican filtros si necesario: por ejemplo, suavizado exponencial o mediana móvil a señales muy ruidosas, siempre cuidando no introducir desfases significativos. La consistencia dimensional es importante: todas las señales deben sincronizarse con una misma marca de tiempo y frecuencia tras procesamiento, usualmente la frecuencia más baja (la de la variable objetivo). Se genera así la tabla final de datos (X, y) para el modelado, dividida en conjunto de entrenamiento y prueba.

Escalado y transformaciones: Muchos algoritmos requieren que las variables estén en escalas comparables. Se emplea escalado (normalización 0-1 o estandarización a media cero y varianza unitaria) sobre las variables de entrada. Esta transformación se ajusta solo con datos de entrenamiento y luego se aplica a nuevos datos para evitar fugas de información. En algunos casos, transformaciones no lineales (logarítmica, potencia) pueden aplicarse a variables cuyo impacto sea multiplicativo o tenga distribuciones muy sesgadas. Toda transformación aplicada se documentará para trazabilidad.

Verificación de consistencia: Finalmente, se verifica que el conjunto de datos preparado tenga congruencia temporal (ej. no mezclar datos futuros en instantes pasados), y que refleje correctamente la operación. Cualquier punto de datos que corresponda a paradas de planta, condiciones anómalas o transitorios fuera del régimen normal se etiqueta o separa para decidir si se incluye en el entrenamiento (según si se desea que el modelo cubra también esas situaciones). Esta curación manual apoyada por expertos de proceso es vital, pues el algoritmo por sí solo no distingue un valor anómalo por calibración incorrecta de uno válido pero poco frecuente.

5. Modelos candidatos

Con los datos preparados, se entrenarán varios **modelos candidatos** de soft sensor. Se busca cubrir desde aproximaciones simples y explicables hasta modelos más complejos con potencial de mayor precisión, evaluando su desempeño relativo. A continuación se listan los tipos contemplados:

Regresión lineal múltiple / PCR: Como punto de partida de baja complejidad, se puede probar una regresión lineal estándar o una *regresión por componentes principales* (PCR) si las variables de entrada están altamente correlacionadas.

PCR reduce la dimensionalidad mediante PCA y luego ajusta un modelo lineal sobre las componentes principales. Son métodos fáciles de interpretar, pero pueden no capturar relaciones no lineales.

PLS (Partial Least Squares): Técnica clásica muy usada en química industrial **Kadlec2009**. Encuentra una proyección de las X maximizando la covarianza con Y, construyendo componentes latentes óptimos para predicción. PLS es especialmente útil cuando hay muchas variables altamente colineales (ej. espectros). Su salida lineal permite cierto análisis de coeficientes para entender contribuciones de cada variable.

Árboles de decisión y *ensembles*: Modelos como *random forests* o *gradient boosting* (XGBoost) pueden capturar no linealidades de forma robusta (promediando muchos árboles reduce sobreajuste). Su interpretabilidad es menor que la de modelos lineales, aunque se pueden extraer medidas de importancia de variables. Suelen funcionar bien con datos heterogéneos y requieren poco ajuste de preprocesamiento. Constituyen un buen candidato intermedio en complejidad.

Máquinas de vectores soporte (SVM): Para algunas variables, un SVR (regresión por vectores soporte) con kernel no lineal podría proporcionar buen ajuste. Sin embargo, los SVM pueden escalar mal a muchos datos y su entrenamiento es más costoso computacionalmente.

Redes neuronales profundas: En particular, redes del tipo *feed-forward* (perceptrón multicapa) para datos estáticos, o redes recurrentes LSTM/GRU si se incluyen secuencias temporales. Las redes profundas pueden aproximar relaciones complejas y han mostrado éxito en diversos soft sensors recientes **Wu2021**. Por ejemplo, una arquitectura LSTM fue empleada para predecir calidad de efluentes en una EDAR con alta precisión **Liu2024**, aprovechando su capacidad de modelar dinámicas no lineales. No obstante, su entrenamiento requiere más datos y ofrecen menos transparencia en su toma de decisiones.

Modelos híbridos o *gray-box*: En casos donde se dispone de algún conocimiento fenomenológico, se podría combinar un modelo de primeros principios parcial con uno de aprendizaje (ej. ajustar parámetros no medibles mediante red neuronal). Sin embargo, dado el alcance de este trabajo y para mantener el enfoque data-driven, no se profundizará en modelos de tipo *gray-box* (aunque se reconoce que son tendencia en la industria **Ahmad2020**).

Cada modelo se entrenará usando el conjunto de entrenamiento preparado, aplicando validación cruzada para evaluar su capacidad predictiva. Se definirá un **criterio de selección** multi-objetivo: por un lado, la precisión (ej. mínimo error de predicción en validación) y por otro lado la simplicidad/explicabilidad y requisitos de cómputo.

6. Incertidumbre y calibración

Un diseño robusto exige no solo una predicción puntual sino también un estimado de la **incertidumbre** asociada. Conocer cuánta confianza tener en cada predicción es crucial para decisiones operativas: por ejemplo, si el modelo indica que un producto está justo en el límite de especificación pero con alta incertidumbre, el operador será más cauteloso. Se integrarán los siguientes enfoques:

Modelado probabilístico: Entre los candidatos se valoró incluir modelos intrínsecamente probabilísticos como *procesos Gaussianos* (GP). Un GP no solo predice un valor medio sino un intervalo de confianza derivado de la varianza predicha. En un caso, Grbić et al. propusieron un soft sensor adaptativo usando una mezcla de modelos GP **Grbic2013**, indicando la viabilidad de esta técnica. Sin embargo, los GP puros escalan mal con muchos datos, por lo que otra opción más ligera es entrenar un *ensemble* de modelos (por ejemplo, múltiples redes neuronales o árboles con diferentes subconjuntos de datos) y usar la dispersión de sus salidas como medida de incertidumbre.

Calibración con datos de laboratorio: Se realizará una calibración final del modelo seleccionado usando un conjunto de datos independiente (por ejemplo, las últimas muestras de laboratorio no usadas en entrenamiento). Esto permite verificar si existe sesgo (bias) sistemático. Si se encuentra un sesgo consistente (p. ej. el soft sensor siempre subestima el valor real en 2 unidades), se aplicará una corrección aditiva o un factor de escala. Esta calibración estática mejora la exactitud absoluta y alinea las predicciones con el método de referencia. Adicionalmente, se puede calibrar la dispersión: comparando la distribución de errores del modelo con sus propias estimaciones de incertidumbre, ajustando factores para que, por ejemplo, un intervalo de $\pm 2\sigma$ alrededor de la predicción efectivamente cubra ~95 % de los errores observados.

Métricas de confianza en línea: Cada predicción vendrá acompañada de un indicador de calidad. Por ejemplo, si el modelo es una red neuronal, se puede obtener la entropía o dispersión en las salidas (vía *dropout* en modo inferencia para aproximar ensambles). Si es un modelo basado en vecinos o locales, se puede calcular la densidad de datos cercanos al punto actual. Estos indicadores internos complementan la alerta de drift (sección siguiente) para decidir si una predicción es fiable o debe tomarse con precaución.

Todos estos aspectos se documentarán en la *model card*, indicando las condiciones bajo las cuales las predicciones son confiables y cuándo la incertidumbre es alta. Esto refuerza la transparencia del soft sensor según buenas prácticas de IA explicable **Mitchell2019**.

7. Detección de derivas y datos OOD

Para mantener la robustez a lo largo del tiempo, el sistema incorporará un módulo de **detección de deriva** (concept drift) y de detección de entradas fuera de distribución esperada (OOD, *out-of-distribution*). Este módulo actúa como "monitor de salud" del soft sensor en operación.

Monitoreo de desempeño: La forma más directa de detectar drift es vigilar el error del modelo cuando se dispone de una medición real. Cada vez que llegue un nuevo valor de laboratorio, se compara con la predicción del soft sensor en ese momento; si el error excede cierto umbral o se observa tendencia sistemática, se asume que el modelo ha perdido calibración. No obstante, como las muestras de laboratorio son poco frecuentes, también se implementan métodos *indirectos* basados en las entradas y salidas modeladas.

Índices de estabilidad de distribución: Una técnica empleada en MLOps es medir cambios estadísticos entre la

distribución original de entrenamiento y la distribución reciente de los datos. Por ejemplo, el **Population Stability Index** (PSI) mide la diferencia entre histogramas de la variable estimada ahora vs. en un período de referencia **Metcalfe2025**. Valores de PSI por encima de cierto umbral (e.g. 0.25) sugieren cambio significativo en el proceso. De forma similar, aquí se calculará PSI u otras métricas (KL divergence, AD test) sobre variables clave y la propia salida del modelo, con ventanas móviles, para anticipar drifts sin esperar a error en laboratorio.

Detección de entradas anómalas (OOD): Se vigilará si las características X entrantes caen fuera del rango o manifold aprendido. Por ejemplo, si un sensor de temperatura está leyendo valores nunca vistos en entrenamiento, o combinaciones de variables muy alejadas de los clusters originales. Para ello se puede usar un modelo de novedad, como un autoencoder entrenado con los datos históricos normales y calculando el error de reconstrucción, o técnicas más sencillas como one-class SVM sobre las X principales. Si el modelo determina que está extrapolando demasiado (caso OOD), se elevará una alerta de que las predicciones pueden no ser fiables.

Alarmas y acciones: Cuando se detecte deriva u OOD, el sistema activará alarmas al operador y tomará acciones según un plan predefinido:

- Si la desviación es moderada, se marca el estado del modelo como "degradado" pero aún usable, recomendando mayor vigilancia.
- Si la desviación es severa o un sensor clave está fuera de rango, el soft sensor puede pasar a un *modo de respaldo* (ver siguiente sección) donde deja de dar valores normales y señala que está fuera de servicio hasta recalibración.
- En paralelo, se registra el evento para que el equipo de datos realice un re-entrenamiento o ajuste del modelo usando los nuevos datos.

Estas estrategias garantizan que no se confíe ciegamente en el modelo bajo condiciones no validadas. La detección temprana de drift evita situaciones peligrosas donde el modelo podría derivar durante meses sin que nadie note el error acumulado.

8. Fallback (mecanismos de respaldo)

Aun con detección de drift, es posible que en operación surjan situaciones donde el soft sensor no pueda proveer una estimación fiable. Por ello se definen mecanismos de **respaldo** o ***fallback*** para mantener la continuidad operativa:

Fallback por fallo de sensor físico: Si alguna señal de entrada crucial para el modelo se pierde (p. ej. sensor averiado o comunicación caída), el soft sensor debería reconocer que no puede funcionar con normalidad. En tal caso, se pueden seguir varias estrategias:

- Usar un *modelo secundario* entrenado con un conjunto reducido de entradas (excluyendo la señal caída) para proveer una estimación degradada. Este modelo de respaldo tendría mayor error, pero al menos daría una indicación aproximada en ausencia de la variable primaria.
- Emplear la *última predicción válida* como valor retenido hasta que se restablezca el sensor (válido solo en procesos lentos donde el valor no cambia bruscamente).
- Conmutar a un esquema manual: el sistema notifica al operador que el sensor virtual está inhabilitado por falta de datos y que deberá operar con las mediciones de laboratorio como única referencia hasta resolver el problema.

En investigaciones recientes se han propuesto esquemas sofisticados de tolerancia a fallos. Por ejemplo, Liu et al. (2024) integran dos redes LSTM donde una aprende la dinámica normal y la otra compensa valores perdidos de sensores a partir de datos históricos de calidad **Liu2024**. En nuestro caso, mantendremos una solución más sencilla dado el alcance, optando por modelos secundarios o notificación, según criticidad.

Fallback por alta incertidumbre: Si el módulo de incertidumbre indica una predicción con intervalo muy amplio (p. ej. ±10 % cuando normalmente es ±2 %) o el detector de OOD marca que estamos extrapolando, el soft sensor puede entrar en *estado de confianza baja*. Esto se refleja al usuario mediante una señal de calidad y posiblemente suprimiendo la recomendación automática. Es preferible alertar "Valor estimado poco confiable" antes que dar una cifra potencialmente errónea sin advertencia.

Fallback programado (mantenimiento): También se puede definir que periódicamente (ej. cada cierto número de semanas o meses) el sensor virtual se ponga en revisión para actualizarse con nuevos datos y validar su desempeño, durante lo cual opera en modo degradado o manual. Esto estaría alineado con las prácticas de recalibración de instrumentos físicos.

Todos estos casos de respaldo y sus procedimientos asociados se documentan en el **registro de riesgos** del proyecto: por cada riesgo (sensor fallido, drift excesivo, etc.) se define la mitigación (fallback, alarma) y el plan de contingencia (ej. personal de instrumentación repara sensor en 24h, equipo de datos re-entrena modelo en 1 semana, etc.). Así, los involucrados saben cómo actuar y el sistema es seguro frente a fallos.

9. Evaluación offline

Para garantizar que el soft sensor cumplirá su propósito, se lleva a cabo una rigurosa **evaluación offline** con datos históricos (o de piloto/planta simulada) antes del despliegue final. Esta evaluación contempla:

Partición de datos: El conjunto histórico disponible se dividirá en entrenamiento (por ejemplo 70 %) y prueba (30 %). Si el histórico abarca diversos lotes o campañas, se procura que la separación sea por fechas (los datos más recientes como prueba) para imitar el caso de predicción futura. Adicionalmente, se usa validación cruzada k-fold dentro del entrenamiento para robustez en la selección de hiperparámetros de los modelos. En procesos con comportamiento estacional o lotes distintos, se asegura que la partición no mezcle datos de un mismo lote en entrenamiento y prueba (para no sobreestimar desempeño).

Métricas de desempeño: Se emplearán métricas de regresión estándar: el error cuadrático medio (MSE) y su raíz RMSE, el error absoluto medio (MAE) o porcentual (MAPE), y el coeficiente de determinación R^2. Estas métricas

permiten cuantificar la precisión global del modelo. Por ejemplo, un objetivo podría ser lograr $R^2 > 0{,}90$ y MAPE $< 5\,\%$ en el conjunto de prueba, como se espera según referencias de casos similares **Zhang2022**. También se revisará el comportamiento en diferentes rangos de operación: se podría calcular el error por segmentos (p. ej. bajo vs. alto valor de la variable) para ver si el modelo es sesgado en alguna región. Los resultados se compararán con los criterios de aceptación definidos en la matriz de KPIs del proyecto.

Simulación de casos de uso: Más allá de las métricas numéricas, se simularán escenarios específicos. Por ejemplo, se puede emular cómo habría sido la operación si el soft sensor estuviera activo durante un episodio de variación rápida de calidad: ¿hubiera detectado a tiempo la desviación? Para ello, se toma un período histórico donde la calidad cambió y se genera la serie temporal de predicciones del modelo comparada con los valores de laboratorio reales. Se verifica que el modelo anticipa adecuadamente las tendencias y que el retardo (delay) respecto a la muestra real es mínimo. Otro caso es simular cómo se comporta el detector de drift: inyectando artificialmente un cambio de sesgo en los datos de prueba y comprobando que las alarmas se activarían.

Robustez a ruido y outliers: Se pueden agregar perturbaciones controladas a los datos de prueba para evaluar la robustez. Por ejemplo, introducir un 5 % de outliers aleatorios en entradas y ver si el modelo aún funciona (quizá gracias a tener filtros internos o no depender de una sola variable). Si un modelo es demasiado sensible, podría descartarse en favor de otro más estable aunque sea marginalmente menos preciso.

Resultados esperados: Dado el plan de modelado planteado, se espera obtener un error medio muy por debajo de la variabilidad natural del proceso. Por ejemplo, si la variabilidad diaria típica de la calidad (observada históricamente) es ±10 unidades, el soft sensor debería tener un RMSE del orden de 2–3 unidades para ser valioso. En términos de negocio, esto podría traducirse en detección de desvíos varias horas antes de que un análisis de laboratorio lo confirme, evitando producción fuera de especificación. Estos beneficios esperados se documentarán, aunque no se pueden cuantificar plenamente sin desplegar el sistema.

En ausencia de un entorno de planta real para pruebas en tiempo real dentro del periodo del TFM, la evaluación offline con históricos es la principal evidencia objetiva de desempeño. Se presentarán los resultados en gráficos y tablas, y se validará que cumplen con los criterios acordados con los responsables (ingeniería de proceso, calidad). Solo tras esta verificación el proyecto avanzaría al despliegue en producción.

10. Evaluación en *edge*

Una vez validado offline, el modelo se integrará en un dispositivo **edge** próximo al proceso (p. ej., un gateway industrial o un módulo tipo NVIDIA Jetson) y se evaluará su desempeño en condiciones de operación en tiempo real. Esta fase comprueba aspectos no visibles en la simulación offline, relacionados con tiempos de cómputo, comunicación y fiabilidad continua.

Entorno hardware: Se prevé usar una plataforma Jetson TX2 o Xavier (GPU NVIDIA) instalada en planta, que ejecutará el modelo en inferencia continua. Alternativamente, podría usarse un PC industrial (arquitectura x86_64) según disponibilidad; en cualquier caso se busca un sistema embebido capaz de operar 24/7 en ambiente industrial. El modelo entrenado en desarrollo (probablemente usando Python/Sklearn o PyTorch) se exportará al formato estándar ONNX para ser cargado eficientemente en el dispositivo. Utilizando aceleradores como TensorRT de NVIDIA se optimizará la latencia de inferencia, especialmente para la red LSTM si es el modelo final.

Latencia y frecuencia: Un requisito no funcional clave es la latencia: el tiempo desde que nuevas lecturas de sensores están disponibles hasta que el soft sensor entrega su predicción. Se aspira a una latencia <1 segundo, para que prácticamente en cuanto llegue un nuevo dato (p. ej. el PLC actualiza cada 1 s) esté la estimación lista. En pruebas iniciales, se medirá la latencia promedio y máxima del pipeline completo en el edge, incluyendo adquisición de datos (por OPC UA u otro protocolo) + cómputo del modelo + envío del resultado al SCADA. Si la latencia excede lo requerido, se evaluarán optimizaciones: por ejemplo, reducir el tamaño del modelo (pruning/cuantización), aumentar la prioridad de su proceso en el sistema operativo, o en último caso simplificar el modelo.

Uso de recursos: También se monitorizará el uso de CPU, GPU y memoria en el edge. El objetivo es que el soft sensor consuma solo una fracción razonable de los recursos para coexistir con otras tareas. Por ejemplo, en una Jetson se buscará que la inferencia utilice <20 % de CPU o se derive a GPU sin saturarla, y que la RAM ocupada por el modelo sea mínima (ONNX permite cargar solo lo necesario). Si se detectan cuellos de botella, se podrá ajustar el intervalo de inferencia (ej. en vez de cada segundo, cada 5 s) o mejorar el código.

Confiabilidad en ejecución continua: Se planea una prueba de estrés ejecutando el sistema continuamente durante varios días en el edge (usando datos históricos reproducidos acelerados, o en sombra con la planta real). Se verificará que no haya fugas de memoria, que los logs registren correctamente las alarmas de drift, y que la comunicación con sistemas de planta sea robusta (reconexión automática si se pierde, etc.). Esta etapa puede revelar issues como que cierta librería no maneja un reinicio del modelo, o que haya discrepancias numéricas entre la predicción en desarrollo vs en edge (por diferencias de arquitectura, que se mitigarán con pruebas unitarias).

Aceptación en planta: Finalmente, se hará una demostración en entorno controlado al personal de planta, mostrando el soft sensor funcionando en vivo, comparando sus predicciones con mediciones de laboratorio recientes. Si bien durante el TFM no se hará control cerrado, se puede simular la recomendación: mostrar en pantalla “Se estima calidad = 85 (objetivo 90) – Sugerencia: aumentar temperatura reactor 1°C”. Esto para comprobar la utilidad práctica y recoger feedback de los usuarios finales.

11. Artefactos y despliegue final

En paralelo al desarrollo técnico, se generan **artefactos** que aseguran la trazabilidad y mantenibilidad a largo plazo del soft sensor:

Model Card: Siguiendo las recomendaciones de transparencia **Mitchell2019**, se elabora una ficha técnica del modelo. Esta *model card* incluye: descripción de la variable estimada y su importancia; listado de variables de entrada usadas; detalles del conjunto de datos de entrenamiento (rango de fechas, número de muestras, representatividad); algoritmo/modelo seleccionado con sus hiperparámetros finales; métricas de desempeño obtenidas offline (error, R^2, etc.); supuestos y limitaciones; y consideraciones éticas o de impacto. Esta documentación estandarizada facilita que terceros (otros ingenieros, auditores) entiendan qué hace el modelo y cómo usarlo correctamente.

Repositorio de código y datos: Todo el código fuente para entrenamiento, evaluación y despliegue se mantiene en un repositorio controlado (p. ej. GitHub del autor). Se incluyen notebooks o scripts R/Python usados en la exploración de datos, el pipeline de preprocesamiento reproducible, y el código de entrenamiento del modelo final con semillas fijadas para replicabilidad. Además, se almacenan de forma segura los conjuntos de datos históricos utilizados, limpios y en formato documentado. Esto permite re-entrenar o inspeccionar el modelo en el futuro si surgen dudas o nuevo personal toma el relevo.

Exportación del modelo (ONNX): El modelo final entrenado se exporta al formato **ONNX** (Open Neural Network Exchange), que es un estándar abierto soportado por múltiples plataformas. Esto asegura portabilidad: por ejemplo, si en el futuro se migra del dispositivo Jetson a otra arquitectura, el modelo ONNX se podrá cargar y ejecutar con mínima adaptación. Junto con el archivo onnx, se provee un pequeño módulo de inferencia (en C++ o Python embebido) que carga el modelo, sus escalados de variables, y realiza la predicción dado un vector de entradas. Este módulo es el que se integra en el sistema de control (por ejemplo, llamado desde el SCADA o desde un servicio en el edge).

Plan de mantenimiento y reentrenamiento: Se documenta cómo y cuándo reentrenar el modelo. Idealmente, se establece que tras detectar deriva significativa o cada cierto periodo (ej. anual), se extraerán nuevos datos etiquetados y se actualizará el modelo siguiendo el mismo procedimiento (garantizando compatibilidad de versiones de librerías, etc.). Se incluyen scripts para facilitar ese reentrenamiento (automatizando en lo posible la carga de datos nuevos, re-ejecución de notebooks, generación de nueva model card con métricas actualizadas). Se definen roles: p.ej. el ingeniero de procesos será el dueño funcional y deberá validar cambios; el científico de datos será el encargado técnico de ejecutarlos.

Seguridad y acceso: Considerando la integración OT (tecnología operativa), se contemplan medidas de ciberseguridad y seguridad funcional. Por ejemplo, el contenedor o servicio donde corre el modelo en el edge tendrá solo los puertos necesarios abiertos, las comunicaciones cifradas si van a la nube, y usuario/contraseña robustos. Cualquier actualización de modelo seguirá el protocolo MOC de la planta.

12. Discusión

La solución propuesta cumple con los objetivos planteados de robustez, explicabilidad y operatividad, pero es importante discutir algunas **limitaciones y riesgos** inherentes:

En cuanto a **robustez**, aunque se han incorporado mecanismos de detección de drift y fallback, siempre existe el riesgo de cambios inesperados en el proceso que no sean captados a tiempo. Por ejemplo, si ocurre un cambio gradual muy lento en la composición de la materia prima, el modelo podría desviarse sin disparar umbrales de alarma por bastante tiempo. La respuesta a esto sería afinar continuamente los umbrales (e.g. PSI) o implementar técnicas de adaptación continua (como re-entrenamiento incremental) **Kadlec2011**, lo cual no se implementó en este TFM por complejidad.

En **explicabilidad**, usar un modelo híbrido (PLS+LSTM) puede dificultar la interpretación completa, ya que la parte neuronal actúa como una "caja negra" parcial. Se ha mitigado dando prioridad a variables entendibles y generando una model card detallada, pero para usuarios en planta la confianza tomará tiempo. Será útil proporcionar visualizaciones sencillas, por ejemplo, si la predicción sube, indicar cuáles variables de entrada contribuyeron más a ese cambio (usando técnicas SHAP, LIME, etc.). Esto no fue desarrollado a fondo aquí pero queda recomendado como mejora futura.

Sobre la **operatividad e integración**, un riesgo identificado es la dependencia en infraestructura de datos: si el historiador o la red OPC tienen latencias o fallos, el soft sensor puede quedarse sin datos. Se asumió un entorno relativamente estable en planta (actualizaciones de sensores cada segundo). En entornos reales, la latencia de red puede variar y habría que garantizar buffers o métodos de timestamp para no desalinear datos.

Desde el punto de vista de **ciberseguridad**, montar un dispositivo edge con un modelo supone agregar una superficie de ataque potencial en la red OT. Es imprescindible seguir las políticas de la empresa (segmentación de red, deshabilitar puertos, cuentas limitadas).

Finalmente, se considera la **sostenibilidad del modelo**. Un riesgo habitual es que, tras el despliegue inicial, el modelo quede obsoleto si no hay una figura encargada de mantenerlo. Se sugiere encarecidamente institucionalizar la rutina de revisión periódica del soft sensor, asignando responsables (un ingeniero de proceso "product owner" y un analista de datos de soporte). Solo así se garantizará que los beneficios se mantengan en el tiempo y se adapte el modelo a cualquier cambio de proceso que puedan venir más adelante.

13. Conclusiones

Se ha desarrollado un dossier completo para la implementación de un **soft sensor industrial**, cubriendo desde la selección de variables de interés hasta los aspectos de despliegue en entorno operativo. Este enfoque integrador busca asegurar que el sensor virtual proporcionará estimaciones confiables de variables de calidad en tiempo real, complementando las mediciones de laboratorio tradicionales y habilitando una operación más proactiva en la planta.

En resumen, el soft sensor propuesto:

- Utiliza un modelo de aprendizaje automático (combinación de técnicas PLS y redes neuronales) entrenado con datos históricos reales del proceso, logrando alta precisión en la predicción de parámetros de calidad difíciles de medir en línea.
- Incorpora mecanismos de detección de drift, estimación de incertidumbre y estrategias de *fallback* que le confieren robustez y seguridad, al evitar confiar en el modelo fuera de su rango válido de aplicación.
- Ha sido validado extensivamente offline con datos históricos, cumpliendo los criterios de desempeño establecidos (error dentro de límites aceptables, R^2 elevado, etc.), y se ha comprobado su viabilidad de cómputo en un dispositivo edge de características industriales.
- Viene acompañado de documentación y artefactos (model card, código, modelo exportado) que facilitan su interpretabilidad y mantenimiento futuro, alineándose con las mejores prácticas de MLOps y gobernanza de modelos **Metcalfe2025**.

Como trabajo futuro, se sugiere explorar la adaptación continua del modelo en tiempo real (*online learning*) para reaccionar a derivas de forma automática, siguiendo las directrices de algoritmos adaptativos en soft sensors **Kadlec2011**. Asimismo, integrar técnicas de explicabilidad más avanzadas (por ejemplo, descomposición de SHAP values) permitiría brindar explicaciones en lenguaje natural a los operadores sobre por qué el sensor virtual da cierto resultado.

En conclusión, el proyecto sienta las bases para una mejora significativa en el monitoreo de calidad en la planta de Repsol Puertollano, demostrando cómo los soft sensors pueden ser desarrollados de forma rigurosa y sistemática para integrarse exitosamente en la operación industrial. Se espera que esta experiencia sirva de referencia para futuras implementaciones de analítica avanzada en entornos de producción, maximizando el valor de los datos históricos disponibles y contribuyendo a la transformación digital de los procesos.