Predicción de Fallos en Máquinas mediante Clasificación, Reducción de Dimensionalidad y Agrupamiento con Modelos de Ensemble

Sergio Soler Rocha

5 de noviembre de 2024

Resumen

Este informe presenta un enfoque integral para la predicción de fallos en máquinas utilizando un conjunto de datos sintético generado a partir de un modelo de deep learning. Para mejorar la comprensión y el rendimiento del modelo, se implementan técnicas de **reducción de dimensionalidad**, como **PCA** y **t-SNE**, lo que permite visualizar patrones subyacentes en un espacio bidimensional. Además, se emplean métodos de **detección de anomalías** (DBSCAN, Isolation Forest, One-Class SVM) para identificar posibles observaciones atípicas y compararlas con los fallos conocidos.

Como parte del proceso de agrupamiento, se utiliza **K-Means** para segmentar los datos en zonas de alto y bajo riesgo, aplicando el método del codo para determinar el número óptimo de clusters. Posteriormente, se construyen varios clasificadores (Logistic Regression, Random Forest, XGBoost y LightGBM) que se combinan en un modelo de **Ensemble** para maximizar la precisión y robustez de las predicciones. Este enfoque permite una identificación más efectiva de los fallos, lo cual es fundamental para estrategias de mantenimiento predictivo y mejora de la confiabilidad en sistemas industriales.

1. Introducción

La predicción de fallos en sistemas industriales es un aspecto clave para optimizar la eficiencia y reducir los costos operativos derivados de fallos imprevistos. Con los avances en técnicas de *machine learning* y el crecimiento en la disponibilidad de datos, se han desarrollado modelos predictivos que permiten implementar estrategias de mantenimiento preventivo más efectivas. Este informe explora un enfoque integral para la predicción de fallos en máquinas, empleando un conjunto de datos sintético diseñado para este propósito, que permite estudiar y probar diferentes técnicas de análisis.

El conjunto de datos utilizado en esta competencia fue generado a partir de un modelo de *deep learning* entrenado en predicciones de fallos en máquinas. Aunque las distribuciones de las características en los datos de entrenamiento y prueba son similares a las del conjunto de datos original, presentan ligeras variaciones. Este conjunto contiene características relevantes del entorno y condiciones operativas de las máquinas, tales como temperatura del aire y del proceso, velocidad de rotación, par de torsión y desgaste de herramientas, además de variables específicas de fallos: **TWF** (Fallo de Herramienta), **HDF** (Fallo Hidráulico), **PWF** (Fallo de Energía), **OSF** (Fallo de Sobrecalentamiento) y **RNF** (Fallo de Enfriamiento).

Para abordar este problema, se emplean técnicas avanzadas de **reducción de dimensionalidad** como **PCA** y **t-SNE** para visualizar patrones en los datos en un espacio de menor dimensión. Asimismo, se aplican métodos de **detección de anomalías** —como **DBSCAN**, **Isolation Forest** y **One-Class SVM**— para identificar observaciones atípicas que podrían corresponder a fallos.

Además, el algoritmo de **K-Means** se utiliza para agrupar los datos en zonas de alto y bajo riesgo de fallo, aplicando el método del codo para determinar el número óptimo de clusters. Posteriormente, se construyen varios modelos de clasificación, incluidos *Logistic Regression*, *Random Forest*, *XGBoost* y *LightGBM*, que se integran en un modelo de **Ensemble** para maximizar la precisión y robustez de las predicciones.

El objetivo de este estudio es demostrar cómo la combinación de técnicas avanzadas de análisis de datos y modelos de *machine learning* puede mejorar la identificación de fallos en sistemas industriales, proporcionando una base sólida para una gestión predictiva más efectiva.

2. Análisis Estadístico

En esta sección se presentan los análisis estadísticos realizados sobre los datos de entrenamiento y de prueba, evaluando la similitud entre ambos conjuntos y explorando sus distribuciones y relaciones a través de representaciones gráficas.

2.1. Estadísticas Descriptivas

Las estadísticas descriptivas de los conjuntos de datos de entrenamiento y de prueba muestran valores y distribuciones similares para cada una de las características principales. Esta similitud entre ambos conjuntos es un indicador positivo para la generalización del modelo, ya que minimiza el riesgo de discrepancias que podrían afectar su rendimiento en el conjunto de prueba. Por lo tanto, esta consistencia en las distribuciones nos permite construir y evaluar el modelo con mayor confianza en su capacidad de generalización.

2.2. Análisis de Pairplot en el Conjunto de Entrenamiento

Para comprender mejor la relación entre las características y la clase objetivo, se generó un *pairplot* del conjunto de entrenamiento, diferenciando entre observaciones con fallos y sin fallos, como se muestra en la Figura 1. A través de esta representación, se observan ciertas zonas donde los fallos tienden a acumularse, lo que resulta relevante para el análisis. Esta tendencia en la acumulación de fallos en áreas específicas de las características podría facilitar la tarea del clasificador al intentar distinguir entre observaciones con fallo y sin fallo.

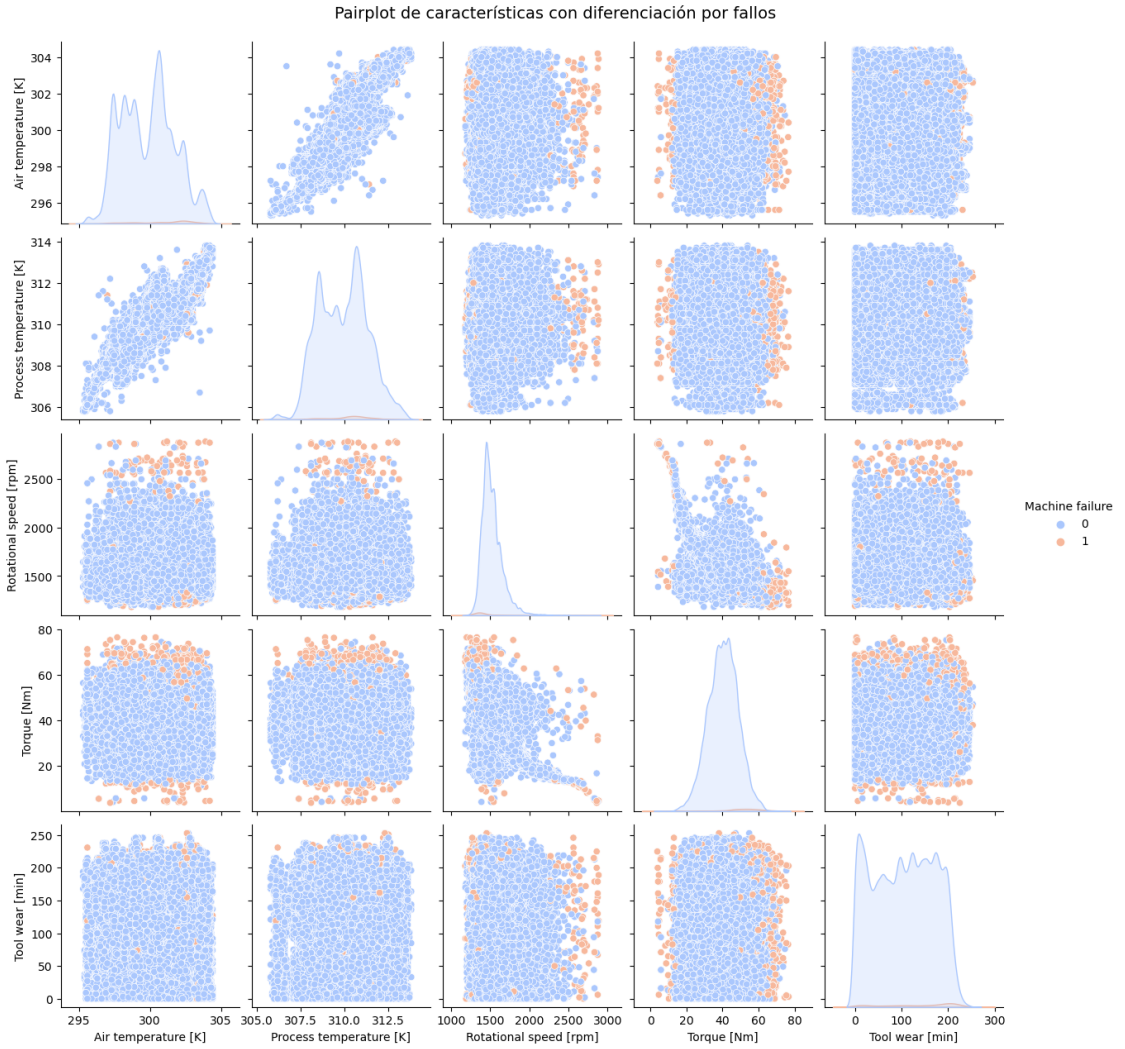

Figura 1: Representación *pairplot* del conjunto de entrenamiento con diferenciación de fallos y no fallos.

2.3. Análisis de Pairplot en el Conjunto de Prueba

Al observar el *pairplot* generado para el conjunto de prueba, como se muestra en la Figura 2, se constata que tanto las distribuciones univariantes de las características como las representaciones de pares muestran una estructura muy similar a la del conjunto de entrenamiento. Este comportamiento confirma la similitud ya observada en las estadísticas descriptivas, reforzando la hipótesis de que el conjunto de prueba tiene una estructura compatible con la del entrenamiento, lo cual favorece la generalización del modelo.

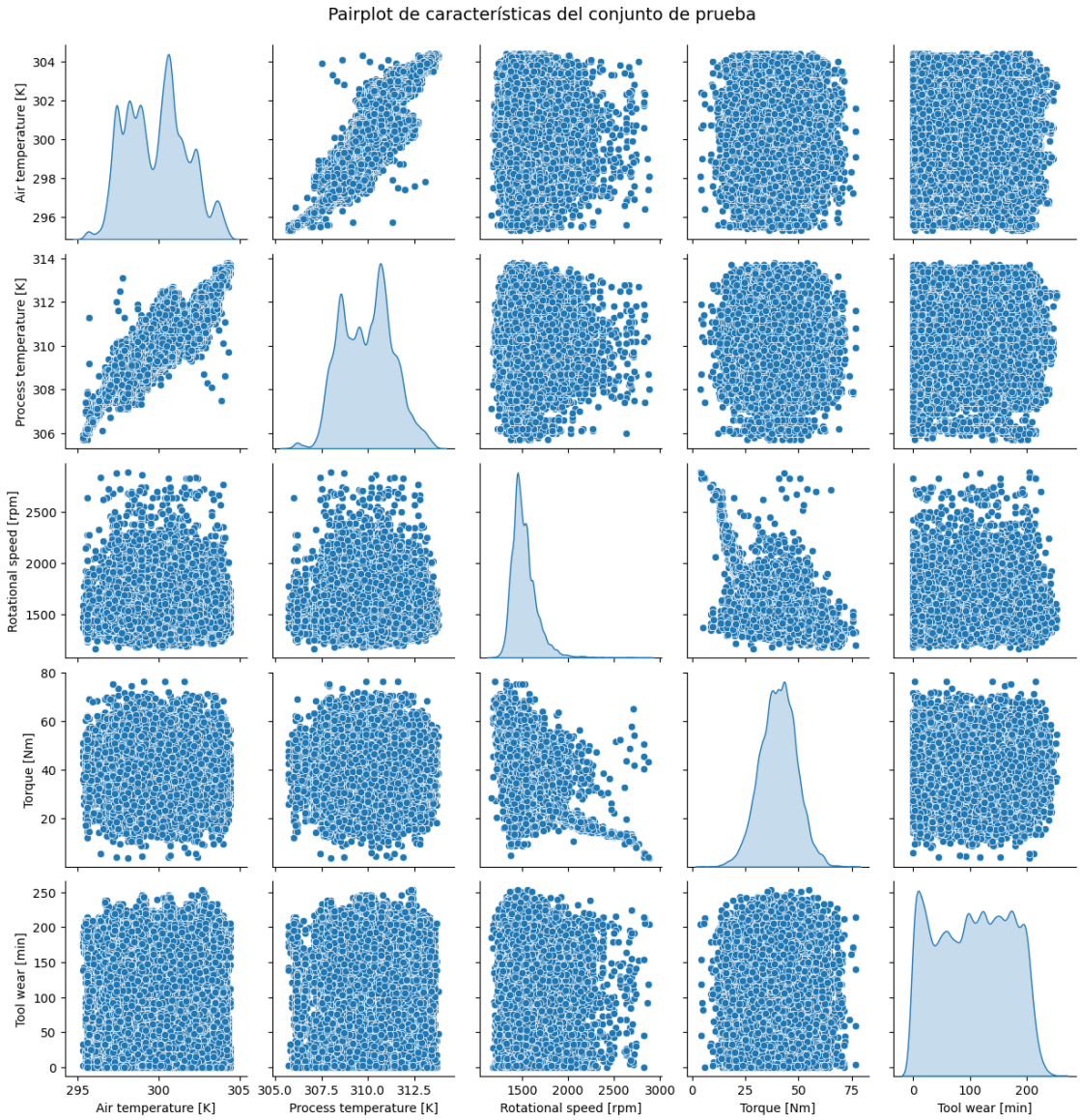

Figura 2: Representación *pairplot* del conjunto de prueba, mostrando similitudes con la estructura del conjunto de entrenamiento.

2.4. Desbalanceo de Clases

Es importante señalar que la clase objetivo está altamente desbalanceada, con solo un 1.57 % de las observaciones correspondientes a fallos. Este desbalanceo puede representar un desafío para el modelo, ya que podría llevarlo a priorizar la clase mayoritaria. Por ello, será crucial considerar técnicas de balanceo y evaluar el impacto del desbalance en el rendimiento del clasificador para asegurar una detección eficaz de fallos.

3. Visualización de Datos

En la sección de Visualización de Datos, se emplearán diversas técnicas avanzadas de reducción de dimensionalidad y detección de anomalías para analizar el conjunto de datos y explorar patrones subyacentes. La reducción de dimensionalidad mediante **PCA** (Análisis de Componentes Principales) y **t-SNE** (T-distributed Stochastic Neighbor Embedding) permitirá proyectar los datos en espacios bidimensionales, facilitando así su visualización y la identificación de posibles agrupamientos o estructuras inherentes a las características.

Para detectar posibles anomalías y compararlas con los fallos conocidos, se utilizarán métodos

como **DBSCAN** (Clustering Basado en Densidad para Aplicaciones con Ruido), **Isolation Forest** y **One-Class SVM**. Estas técnicas permitirán identificar observaciones atípicas en el conjunto de datos que podrían corresponder a fallos en el sistema. Al comparar las anomalías detectadas con los registros de fallos, se buscará comprender mejor las características de los fallos y evaluar la efectividad de cada método de detección en este contexto.

3.1. PCA (Análisis de Componentes Principales)

En esta subsección, se ha aplicado **PCA** (Análisis de Componentes Principales) para reducir la dimensionalidad del conjunto de datos a dos dimensiones, lo que facilita su visualización. La Figura 3 muestra la representación del conjunto de entrenamiento con diferenciación de fallos y no fallos. En esta visualización, se puede observar que muchos de los fallos se localizan de forma separada en la parte inferior de la representación y en el lado derecho, lo cual podría indicar zonas específicas de riesgo de fallo en el espacio de características.

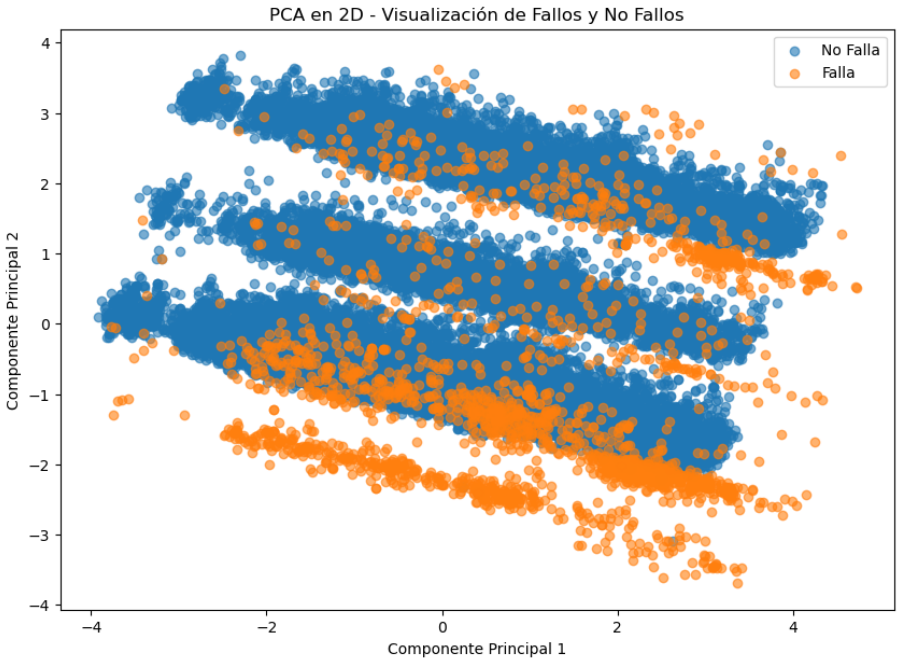

Figura 3: Representación de PCA en 2D del conjunto de entrenamiento, diferenciando entre fallos y no fallos.

La Figura 4 muestra la representación del conjunto de prueba utilizando PCA. En esta visualización, se observa una parte inferior claramente separada del resto de los datos, lo que sugiere que podría tratarse de una zona con tendencia a errores, aunque sin confirmación de fallos específicos en este conjunto.

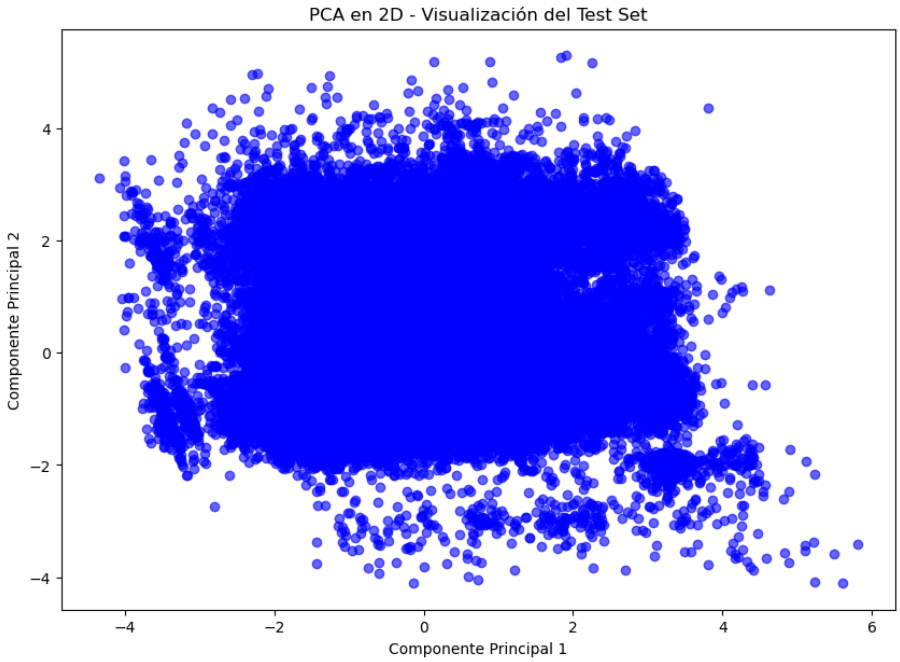

Figura 4: Representación de PCA en 2D del conjunto de prueba.

Sin embargo, debido a las limitaciones inherentes al algoritmo PCA, que solo captura relaciones lineales en los datos, se utilizará el algoritmo **t-SNE** en la siguiente sección para realizar un análisis más detallado y potencialmente descubrir patrones no lineales en los datos.

3.2. t-SNE

La técnica de **t-SNE** (T-distributed Stochastic Neighbor Embedding) se utiliza para proyectar los datos en dos dimensiones, permitiendo revelar agrupaciones y relaciones entre observaciones que pueden no ser evidentes en el espacio original. Esta técnica es especialmente valiosa para explorar diferencias y similitudes entre puntos, proporcionando una visualización más intuitiva y detallada de la estructura interna del conjunto de datos.

En la Figura 5, correspondiente al conjunto de entrenamiento, se puede observar una estructura más compleja que la obtenida con PCA, con tres zonas destacadas donde se concentran los fallos. Estas zonas podrían considerarse como potenciales áreas de alto riesgo en futuras etapas del modelo. Sin embargo, no se observa una separación total de los fallos, ya que también existen zonas en las que aparecen mezclados con los datos sin fallos.

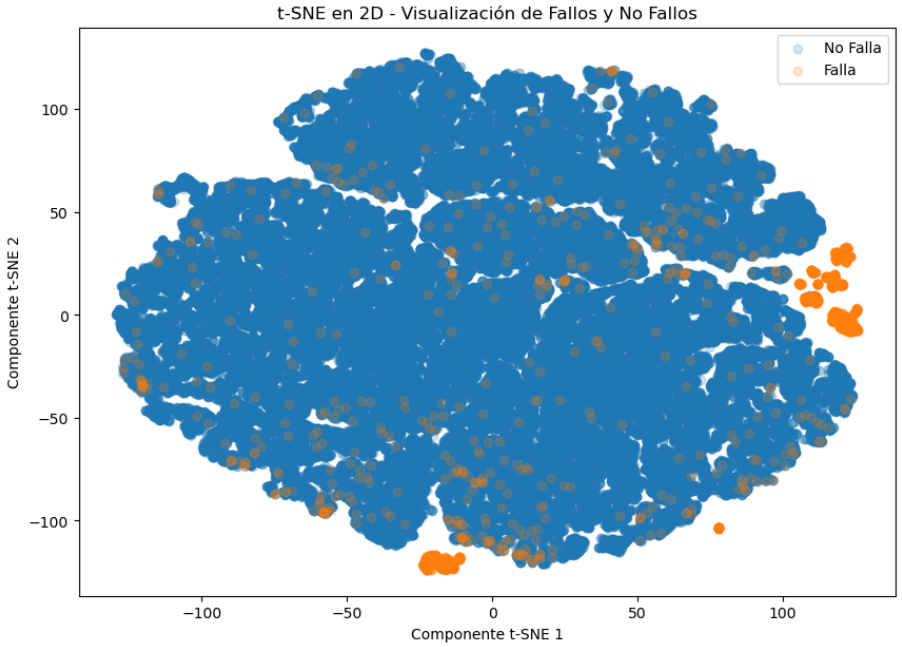

Figura 5: Representación de t-SNE en 2D del conjunto de entrenamiento, diferenciando entre fallos y no fallos.

La Figura 6 muestra la proyección de t-SNE para el conjunto de prueba. A grandes rasgos, la estructura visual es similar a la observada en el conjunto de entrenamiento, con islotes que parecen indicar áreas de riesgo potenciales. Aunque los fallos no están completamente aislados, es posible intuir la ubicación de estas zonas de riesgo con mayor claridad que en la representación con PCA.

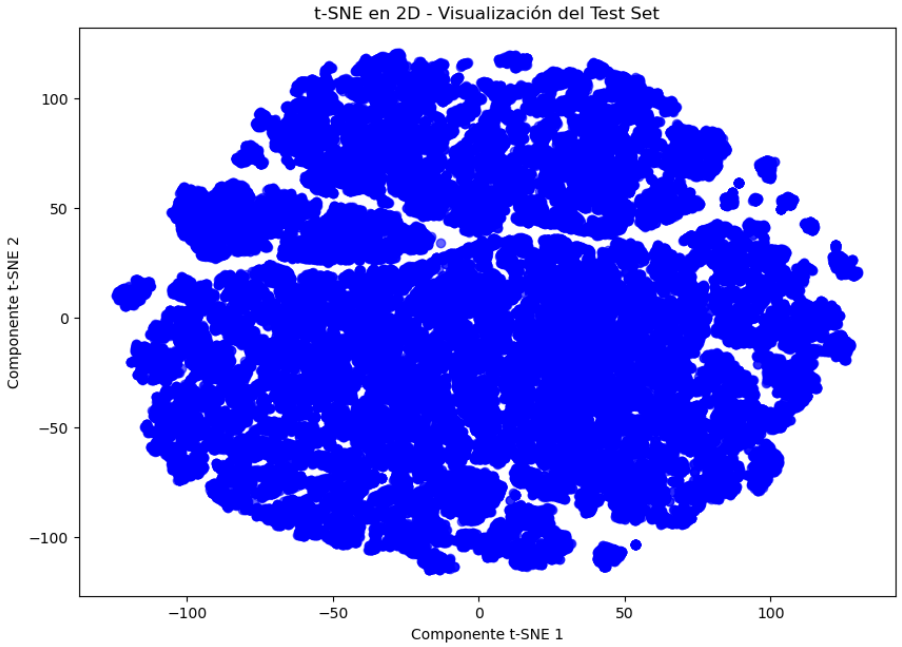

Figura 6: Representación de t-SNE en 2D del conjunto de prueba.

3.3. DBSCAN y One-Class SVM para Detección de Anomalías

En esta sección se emplean los métodos de **DBSCAN** y **One-Class SVM** para la detección de anomalías en el conjunto de datos. Ambos métodos permiten identificar observaciones atípicas que podrían estar asociadas con fallos en el sistema, y se comparan sus resultados en términos de

precisión y relevancia en la detección de fallos.

3.3.1. DBSCAN

Para ajustar el parámetro *eps* en DBSCAN, se utilizó el *método del codo* en la Figura 7, estimando el valor óptimo en aproximadamente 0.95 con 5 vecinos. Con estos hiperparámetros, los resultados obtenidos fueron los siguientes:

- **Total de anomalías detectadas**: 936
- **Total de anomalías que coinciden con fallos**: 498
- **Porcentaje de anomalías que coinciden con fallos**: 53.21 %

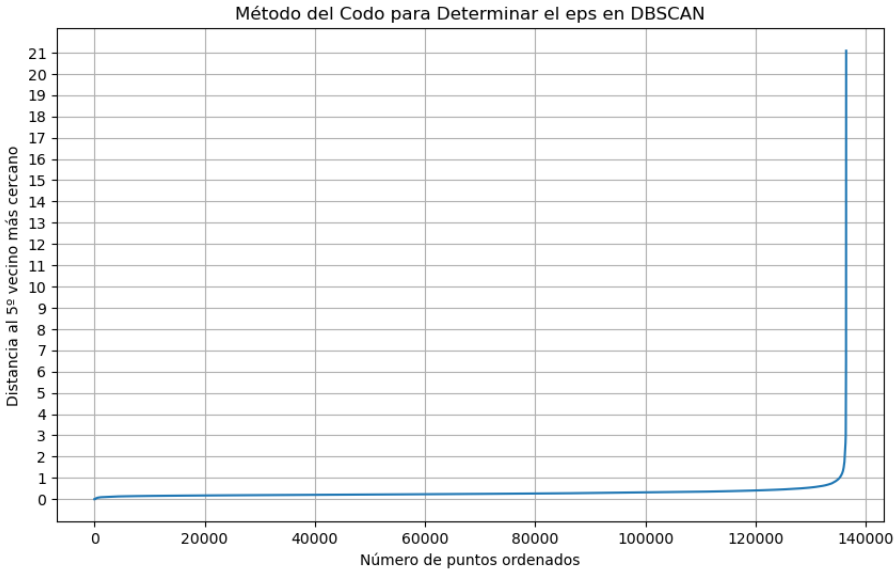

Figura 7: Método del codo para determinar el valor óptimo de *eps* en DBSCAN.

3.3.2. One-Class SVM

Para One-Class SVM, se ajustó el hiperparámetro *nu* en un valor de 0.005, logrando los siguientes resultados:

- **Total de anomalías detectadas**: 688
- **Total de anomalías que coinciden con fallos**: 425
- **Porcentaje de anomalías que coinciden con fallos**: 61.77 %

Comparando los resultados, se observa que One-Class SVM detecta menos anomalías que DBSCAN, pero con una precisión ligeramente mayor en la coincidencia de fallos. Esto es notable considerando el bajo porcentaje de fallos en el conjunto de datos, lo que subraya la efectividad de ambos métodos para identificar posibles áreas de riesgo.

3.3.3. Visualización de Anomalías en t-SNE

A continuación, se presentan visualizaciones en el espacio de t-SNE para el conjunto de entrenamiento y de prueba, mostrando las anomalías detectadas con One-Class SVM. Estas anomalías se concentran en gran medida en los islotes identificados previamente en las visualizaciones, sugiriendo que estas áreas podrían ser consideradas como zonas de alto riesgo.

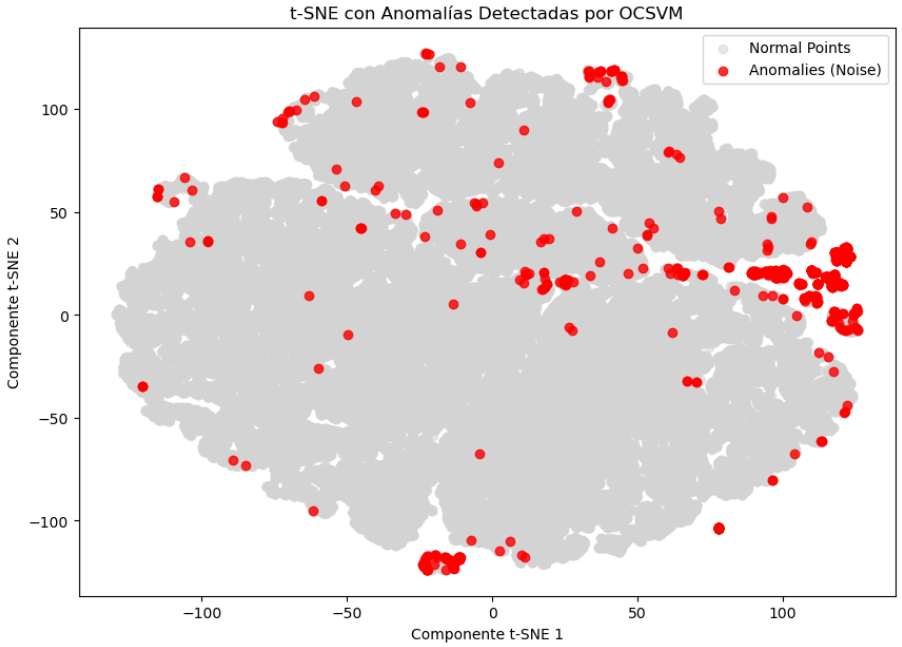

Figura 8: Visualización de t-SNE del conjunto de entrenamiento mostrando las anomalías detectadas con One-Class SVM.

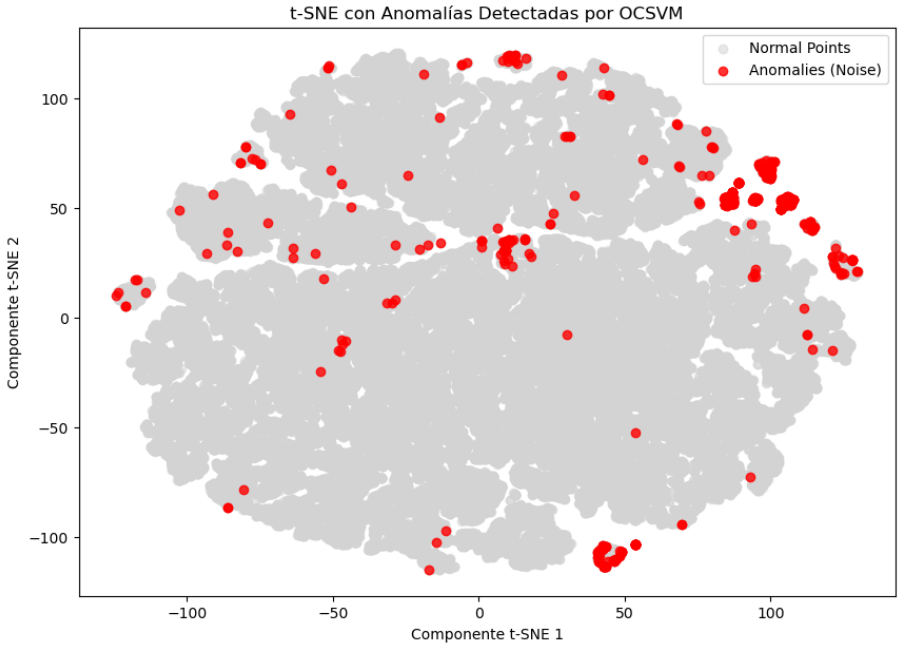

Figura 9: Visualización de t-SNE del conjunto de prueba mostrando las anomalías detectadas con One-Class SVM.

4. Uso de K-Means para Clustering

En el contexto de la predicción de fallas, el algoritmo **K-Means** permite agrupar los datos en áreas de alto y bajo riesgo en función de los patrones históricos y las características compartidas por los puntos de datos. Al analizar los clusters generados, es posible identificar grupos que presentan un mayor porcentaje de fallas y clasificarlos como áreas de alto riesgo. Esta información puede ser aprovechada por el modelo de clasificación para mejorar la precisión de las predicciones, permitiendo anticipar fallas de manera más efectiva.

4.1. Determinación del Número Óptimo de Clusters

Para identificar el número óptimo de clusters, se utilizó el *método del codo*, representado en la Figura 10. Según este método, el número ideal de clusters se encuentra entre 8 y 10, donde la curva comienza a suavizarse. Para nuestros propósitos, seleccionamos **8 clusters** como valor óptimo.

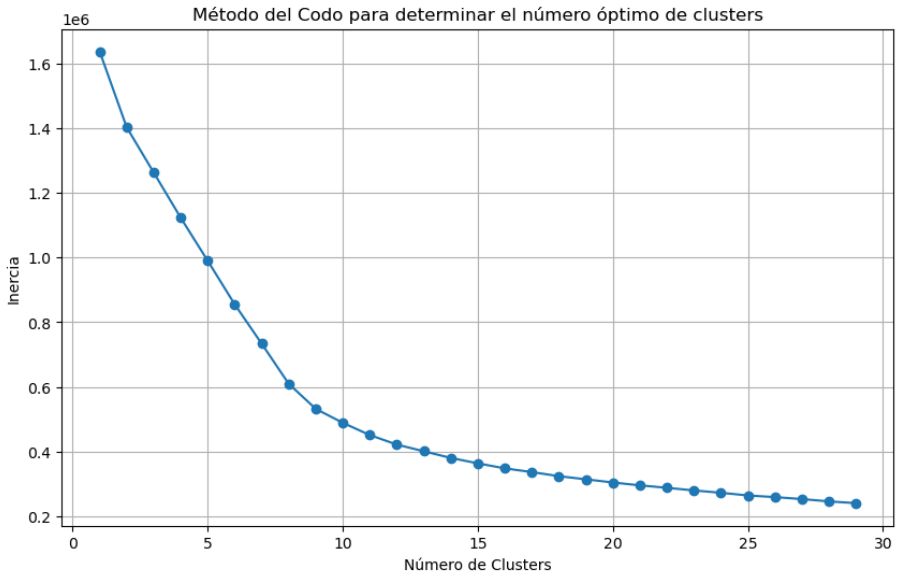

Figura 10: Método del codo para determinar el número óptimo de clusters en K-Means.

4.2. Análisis de Fallas en Clusters

Con la aplicación de K-Means en el conjunto de datos, obtenemos el siguiente porcentaje de fallas en cada cluster:

Cluster	Porcentaje de Fallas (%)
0	0.31
1	0.35
2	0.44
3	99.05
4	1.63
5	99.05
6	99.18
7	100.00

Cuadro 1: Porcentaje de fallas en cada cluster.

Como se observa en la Tabla 1, cuatro de los clusters presentan una concentración significativa de fallas, con porcentajes cercanos al 100 %. Estos clusters se identifican como áreas de alto riesgo.

4.3. Creación de la Columna `Zona_Riesgo`

Para aprovechar esta información en el modelo de clasificación, creamos una columna binaria llamada `Zona_Riesgo`. Esta columna toma el valor de 1 cuando un punto pertenece a un cluster de alto riesgo (clusters 3, 5, 6 y 7) y 0 en caso contrario. La inclusión de esta variable ayudará al clasificador a identificar áreas de posible falla con mayor precisión.

4.4. Visualización de Zonas de Riesgo mediante t-SNE

Utilizando la proyección en dos dimensiones generada con t-SNE, visualizamos las zonas de riesgo en el conjunto de entrenamiento y en el conjunto de prueba. En ambas representaciones

(Figuras 11 y 12), se evidencian los ïslotesïdentificados anteriormente, que concentran una cantidad considerable de fallas, confirmando la utilidad de esta agrupación.

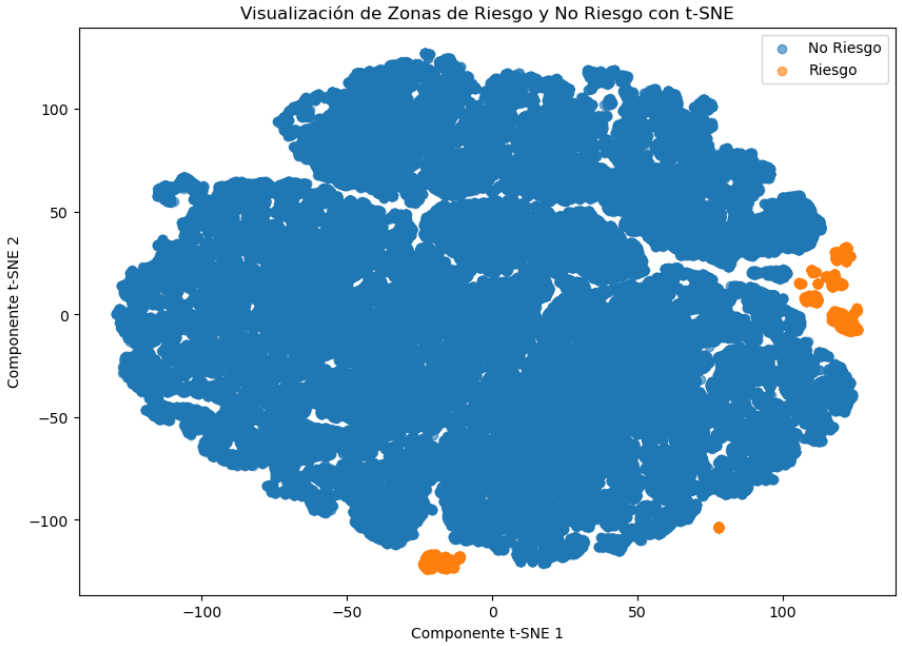

Figura 11: Visualización de t-SNE del conjunto de entrenamiento, mostrando zonas de riesgo y no riesgo.

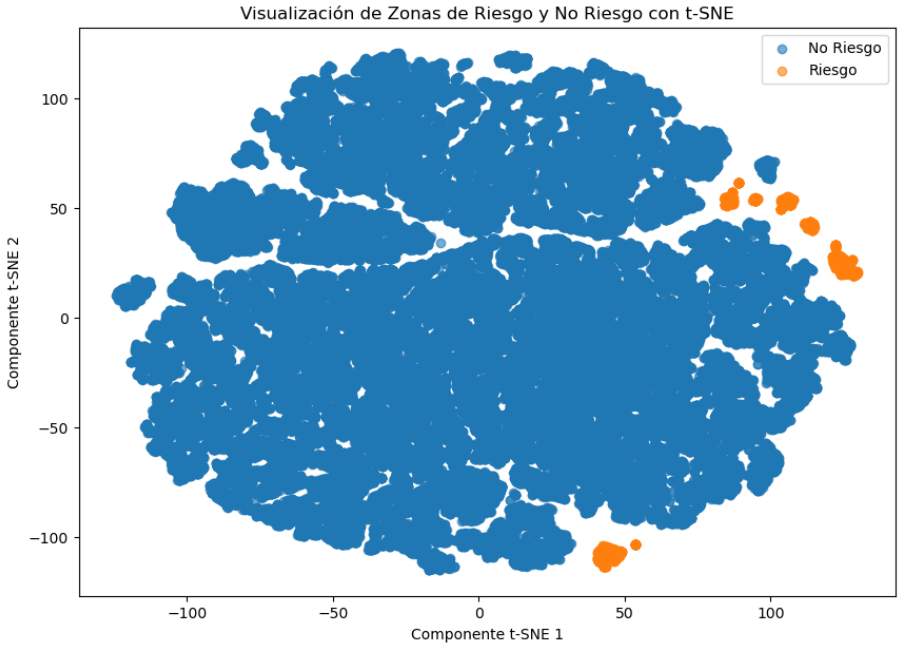

Figura 12: Visualización de t-SNE del conjunto de prueba, mostrando zonas de riesgo y no riesgo.

5. Modelado

Para el modelado de predicción de fallos, se han seleccionado las siguientes variables: `Air temperature [K]`, `Process temperature [K]`, `Rotational speed [rpm]`, `Torque [Nm]`, `Tool wear [min]`, `TWF`, `HDF`, `PWF`, `OSF`, `RNF`, `Type_L`, `Type_M` y `Zona_Riesgo`. Estas variables abarcan caracte-

rísticas operativas, de tipo de fallo y zonas de riesgo, proporcionando al modelo información clave para diferenciar entre condiciones de fallo y no fallo.

La métrica seleccionada para evaluar los modelos es el **AUC-ROC** (Área bajo la Curva Receiver Operating Characteristic). El AUC-ROC es especialmente ventajoso en problemas de clasificación desbalanceada, como el presente, ya que mide la capacidad del modelo para distinguir entre las clases, siendo robusto frente a la distribución desigual de fallos. Un AUC-ROC alto indica que el modelo puede asignar correctamente una mayor probabilidad de fallo a observaciones con errores en comparación con las observaciones sin fallo, lo que es crucial para este problema de predicción de fallos.

5.1. Regresión Logística

La **Regresión Logística** es un método de clasificación lineal que modela la probabilidad de que una observación pertenezca a una clase en particular en función de las características seleccionadas. Dado su enfoque interpretativo y su simplicidad, se utiliza como modelo *baseline* en este estudio, sirviendo de referencia para evaluar la mejora de modelos más complejos.

Para esta tarea, la Regresión Logística se aplica sobre los datos escalados y los resultados muestran un AUC-ROC de 0.9300, indicando un desempeño sólido inicial para este tipo de modelo en el problema de predicción de fallos.

$$\text{AUC-ROC de la Regresión Logística Múltiple } = 0{,}9300 \tag{1}$$

5.2. Random Forest

El modelo **Random Forest** es un método de clasificación basado en el ensamble de múltiples árboles de decisión. Cada árbol en el bosque se construye utilizando un subconjunto aleatorio de las observaciones y de las características, lo que reduce la varianza del modelo y mejora su capacidad de generalización. La clasificación final de Random Forest se obtiene combinando las predicciones de todos los árboles, ya sea por votación mayoritaria o mediante el promedio de las probabilidades predichas.

Para optimizar el rendimiento del modelo, se utilizó la técnica de **RandomizedSearchCV**, la cual permite realizar una búsqueda aleatoria en el espacio de hiperparámetros definidos. A diferencia de una búsqueda exhaustiva, *RandomizedSearchCV* explora una muestra aleatoria de combinaciones de hiperparámetros, logrando una optimización eficiente en menor tiempo. Mediante esta técnica, se identificaron los siguientes hiperparámetros óptimos:

- `n_estimators`: 1000
- `min_samples_split`: 5
- `min_samples_leaf`: 2
- `max_features`: log2
- `max_depth`: None

Con estos hiperparámetros ajustados, el modelo de Random Forest obtuvo un **AUC-ROC** de 0.9532, demostrando una mejora significativa respecto al modelo baseline y un excelente rendimiento en la detección de fallos.

$$\text{AUC-ROC del modelo Random Forest ajustado} = 0{,}9532 \tag{2}$$

5.3. XGBoost

El modelo **XGBoost** (eXtreme Gradient Boosting) es un método de aprendizaje basado en ensambles que emplea el enfoque de *boosting* de gradiente. A diferencia de los modelos de *bagging* como Random Forest, XGBoost ajusta secuencialmente árboles de decisión, de manera que cada nuevo árbol intenta corregir los errores de los anteriores. Este enfoque permite que el modelo se enfoque en las observaciones mal clasificadas, logrando una mayor precisión en comparación con otros modelos de árboles de decisión. XGBoost es además altamente eficiente y escalable, ya que

aprovecha técnicas de regularización y paralelización para optimizar el rendimiento y reducir el sobreajuste.

Para encontrar los hiperparámetros más efectivos en el modelo, se utilizó la técnica de **RandomizedSearchCV**, que permite realizar una búsqueda aleatoria en el espacio de hiperparámetros definidos, logrando una optimización eficiente sin la necesidad de explorar todas las combinaciones posibles. Los hiperparámetros óptimos encontrados para XGBoost fueron los siguientes:

- `subsample`: 0.8
- `n_estimators`: 100
- `min_child_weight`: 3
- `max_depth`: 15
- `learning_rate`: 0.05
- `gamma`: 0.3
- `colsample_bytree`: 0.6

Con estos hiperparámetros ajustados, el modelo XGBoost alcanzó un **AUC-ROC** de 0.9547, lo cual evidencia una excelente capacidad predictiva para la detección de fallos en comparación con el baseline y otros modelos previos.

$$\text{AUC-ROC del modelo XGBoost ajustado} = 0{,}9547 \tag{3}$$

5.4. LightGBM

El modelo **LightGBM** (Light Gradient Boosting Machine) es una implementación optimizada del algoritmo de *boosting* de gradiente, diseñada específicamente para mejorar la velocidad y eficiencia en el entrenamiento de modelos con grandes volúmenes de datos. LightGBM utiliza una estructura de histogramas y una técnica de crecimiento de árboles llamada *leaf-wise* (en lugar de *level-wise* como en otros modelos de boosting), lo cual permite reducir el tiempo de entrenamiento y mejorar el rendimiento del modelo en términos de precisión. Gracias a su enfoque eficiente y a sus técnicas avanzadas de optimización, LightGBM es especialmente adecuado para problemas de clasificación de alta complejidad.

Para ajustar los hiperparámetros de LightGBM, se utilizó **RandomizedSearchCV**, una técnica que permite realizar una búsqueda aleatoria en el espacio de hiperparámetros, explorando una muestra aleatoria de combinaciones y optimizando el modelo de manera eficiente. Los mejores hiperparámetros encontrados para LightGBM fueron los siguientes:

- `subsample`: 0.8
- `num_leaves`: 70
- `n_estimators`: 300
- `min_child_samples`: 10
- `max_depth`: 10
- `learning_rate`: 0.01
- `colsample_bytree`: 0.6

Con estos hiperparámetros ajustados, el modelo LightGBM obtuvo un **AUC-ROC** de 0.9555, logrando un rendimiento superior en la tarea de predicción de fallos en comparación con los modelos previos y resaltando su eficacia en problemas de clasificación complejos.

$$\text{AUC-ROC del modelo LightGBM ajustado} = 0{,}9555 \tag{4}$$

5.5. Modelo Ensemble mediante Promedio Simple

Para mejorar la capacidad predictiva del modelo, se construyó un **modelo ensemble** mediante el promedio simple de las probabilidades de los tres modelos más efectivos: *Random Forest*, *XGBoost* y *LightGBM*. La combinación de estos modelos permite aprovechar la fortaleza de cada uno en la tarea de predicción de fallos, y el promedio de sus probabilidades contribuye a obtener una clasificación final más robusta.

El procedimiento seguido es el siguiente:

1. Se obtienen las probabilidades de fallo predichas por cada modelo individual para el conjunto de prueba, es decir, las probabilidades de que cada observación pertenezca a la clase de fallo.
2. Se calcula el promedio simple de estas probabilidades para obtener la probabilidad final de fallo del modelo ensemble.
3. Finalmente, se calcula el **AUC-ROC** del modelo ensemble, que en este caso es de 0.9586, superando a los modelos individuales.

$$\text{AUC-ROC del modelo ensemble (promedio simple)} = 0{,}9586 \tag{5}$$

La Figura 13 muestra la curva ROC del modelo ensemble, representando la Tasa de Verdaderos Positivos (TPR) en función de la Tasa de Falsos Positivos (FPR) para diferentes umbrales de clasificación. La curva ROC confirma el alto desempeño del modelo ensemble en la tarea de clasificación, logrando un AUC-ROC significativamente elevado.

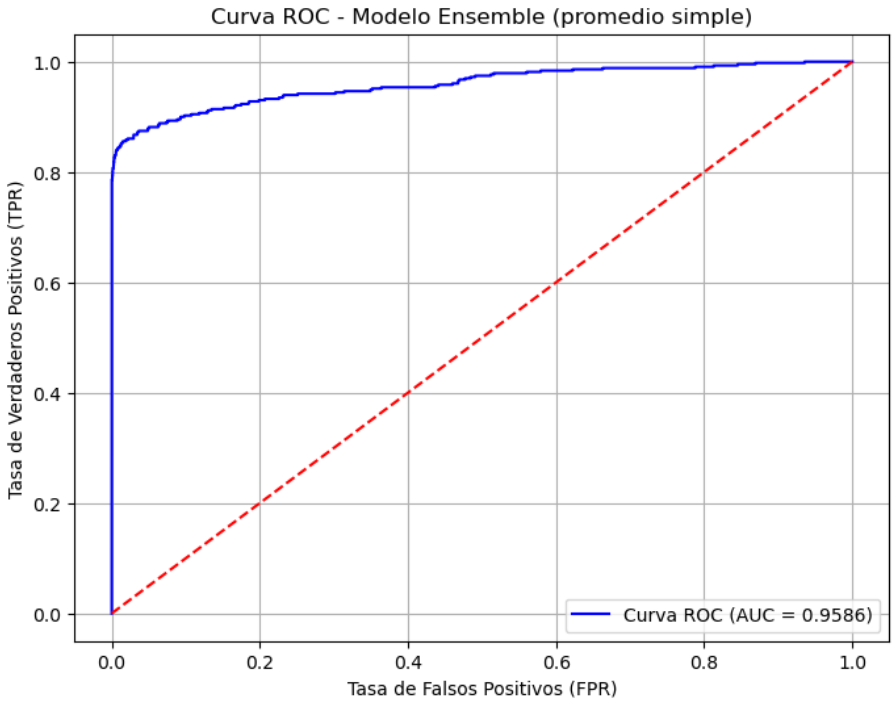

Figura 13: Curva ROC del modelo ensemble (promedio simple) con un AUC de 0.9586.

6. Conclusiones

En este estudio, se desarrolló un modelo de predicción de fallos en sistemas industriales mediante el uso de diversas técnicas de reducción de dimensionalidad, detección de anomalías, agrupamiento y clasificación, aplicadas a un conjunto de datos sintético. Los principales hallazgos y conclusiones obtenidos son los siguientes:

- Las técnicas de reducción de dimensionalidad, como **PCA** y **t-SNE**, permitieron visualizar patrones y agrupaciones en los datos, facilitando la identificación de zonas de alto riesgo de

fallo. La proyección en dos dimensiones reveló la presencia de islotes en los que se concentran los fallos, lo cual fue de gran ayuda para la posterior segmentación en zonas de riesgo.

- La detección de anomalías mediante **DBSCAN** y **One-Class SVM** proporcionó una estrategia adicional para identificar observaciones que podrían representar fallos. Estas técnicas mostraron un alto porcentaje de coincidencia con los fallos, lo que sugiere su utilidad para anticipar posibles eventos de riesgo.
- El uso de **K-Means** para agrupar los datos en clusters de alto y bajo riesgo fue eficaz para segmentar los datos en función de la probabilidad de fallo. La creación de la variable `Zona_Riesgo`, derivada de estos clusters, permitió mejorar la precisión del modelo de clasificación al añadir una capa adicional de información sobre las zonas propensas a fallos.
- En cuanto a los modelos de clasificación, **Random Forest**, **XGBoost** y **LightGBM** demostraron ser efectivos, con valores elevados de AUC-ROC en la predicción de fallos. La optimización de hiperparámetros mediante **RandomizedSearchCV** mejoró aún más el desempeño de cada modelo, destacándose **LightGBM** como el más preciso entre los modelos individuales.
- El desarrollo de un **modelo ensemble** mediante el promedio simple de las probabilidades de los tres mejores modelos individuales (Random Forest, XGBoost y LightGBM) resultó en el valor de AUC-ROC más alto (0.9586). Este modelo ensemble demostró ser la estrategia más efectiva para la predicción de fallos, aprovechando las fortalezas de cada modelo individual y proporcionando una clasificación robusta.

En conclusión, el enfoque propuesto combina múltiples técnicas avanzadas de análisis de datos y modelado, ofreciendo una solución integral para la predicción de fallos. Este enfoque permite identificar zonas de riesgo y mejorar la precisión en la detección de fallos, lo que puede contribuir significativamente a la implementación de estrategias de mantenimiento predictivo y a la optimización de la confiabilidad en sistemas industriales. Los resultados obtenidos demuestran que una combinación adecuada de reducción de dimensionalidad, detección de anomalías, agrupamiento y técnicas de clasificación puede mejorar significativamente el rendimiento predictivo en problemas de alto impacto como la detección de fallos.