Predicción de fallas usando datos sintéticos de una máquina fresadora

Autor: Mtr. César Augusto Arroyo Cárdenas

Resumen

La predicción de fallas de maquinarias industriales es una de las aplicaciones del aprendizaje automático que más expectativa causa en el marco de la revolución industrial 4.0. En este trabajo, mediante datos sintéticos de una máquina fresadora, exploramos técnicas para resolver desafíos que se presentan en la aplicación real. En particular, hablamos del desbalance de datos. Mediante este estudio, encontramos que los métodos de sobremuestreo y SMOTE presentan técnicas prometedoras para implementar en sistemas productivos. Mientras que el método de submuestreo no es recomendado para este tipo de aplicaciones. Sobretodo para el problema multiclase dónde algunas clases pueden ser muy poco representadas.

Introducción

Una de las aplicaciones más solicitadas de los métodos de aprendizaje automático es el mantenimiento predictivo. Esta aplicación, la cual es solicitada principalmente en un marco industrial manufacturero y de operaciones, va acompañado del gran crecimiento que ha habido a nivel mundial en la implementación de las tecnologías IoT.

El IoT o internet de las cosas, involucra dispositivos con sensores y capacidad de procesamiento conectados con otros dispositivos en red. Dispositivos, los cuales intercambian información mediante la red. Así, el internet de las cosas está asociado de forma directa con la electrónica, las telecomunicaciones, la ingeniería de software y la ciencia de la computación. También asociado al internet de las cosas, tenemos la revolución industrial 4.0, la cual implica una compleja integración de muchos sistemas en aras de la eficiencia, auto-configuración, predicción de diferentes tipos de eventos, menor impacto ambiental. Dentro de estos sistemas, una de las principales componentes viene siendo justo la implementación del IoT.

En la actualidad, la adopción de IoT viene liderada por Estados Unidos, China, Japón, y países europeos como Gran Bretaña, Alemania, Francia e Italia. En México, aunque aún rezagado con respecto a estos países, ya se encuentran planes de acción para incluir estas nuevas tecnologías. Uno de los ejes por los que esta tecnología está siendo implementada es mediante los planes de modernización de la red eléctrica. Estos planes se encuentran detallados a través de la PRODESEN o Programa para el Desarrollo del Sistema Eléctrico Nacional. Este, a su vez, implementa los programas para la ampliación y modernización para la Red Nacional de Transmisión y la Red Nacional de Distribución.

Por otro lado, la industria manufacturera en México, representa uno de los rubros que más contribuye al Producto Interno de México. Este representa unos $6.26B de pesos mexicanos al primer trimestre de 2024. Esta industria es una de las principalmente beneficiadas por la adopción del IoT (Data México, 2024). La instalación del internet de las cosas en los dispositivos industriales conlleva a un monitoreo más sistematizado. La recolección de esta información, y su uso, puede inducir una optimización de la operación en un contexto industrial. Es aquí que las tecnologías de datos se vuelven muy importantes. Tecnologías como: Mantenimiento basado en condiciones, solución de problemas avanzados y mantenimiento predictivo pueden ayudar a reducir costos en la operación reduciendo el tiempo de suspensión de la operación debido a fallas de maquinaria (McKinsey, 2021).

Estas tecnologías funcionan en base a la recolección de gran cantidad de datos. Constan de sistemas expertos y de sistemas de aprendizaje automático. Se estima que el mantenimiento predictivo puede tener una reducción de costos del 8% al 12% respecto al mantenimiento preventivo (mantenimientos cíclicos con el objetivo de que la maquinaria se encuentre en buen estado) y de un 40% respecto al mantenimiento reactivo (reparar la maquinaria cuando esta ya ha fallado). Sin embargo, el mantenimiento predictivo solo puede proveer un beneficio en tanto la tasa de falsos negativos y de falsos positivos sean suficientemente bajas.

Problema

La maquinaria industrial es altamente confiable y no falla tan seguido. Esto implica que, incluso en empresas con políticas de recolección de datos y tecnologías como IoT, los registros de fallos tienden a ser muy pocos. Es aquí dónde surge el primer problema en cuanto a la implementación de mantenimiento predictivo. Para poder implementar modelos de aprendizaje automático, tener suficiente cantidad de datos de calidad es indispensable. De esta forma, al presentar pocos fallos naturalmente en la operación, se tienen conjuntos de datos que están altamente desbalanceados. Es decir, se tienen muchos registros de datos de operación dónde la maquinaria no falla, pero se tienen muy pocos registros de datos de operación dónde la maquinaria sí falla. El **primer problema** consiste en: **los conjuntos de datos de mantenimiento predictivo contienen una proporción mucho más alta de datos de no fallo que de datos de fallo**.

El segundo punto que representa un problema, consiste en el costo de los falsos negativos y los falsos positivos. En este contexto, nos referimos como positivo a la presencia de una falla. De este modo, un falso negativo representa una instancia en la que el sistema predice que el sistema no va a fallar cuando el sistema en realidad va a presentar una falla. Los falsos positivos constan de instancias en las que el sistema predice que el sistema va a fallar pero en realidad el sistema no representa una falla.

El costo de los falsos negativos tiene que ver con que el sistema brinda una certeza de operación normal al operador, pero en realidad se va a presentar una falla. Esto genera tiempo de inactividad de la máquina y suspensión de la operación relacionada lo que conlleva a gastos operativos ya que se deja de producir por cierta cantidad de tiempo. Al no estar preparado el equipo de operación, este tiempo de inactividad puede impactar muy

fuertemente. Por otro lado, el costo de los falsos positivos tiene que ver con que el sistema brinda una certeza de operación fallida de la maquinaria, cuando en realidad la máquina se encuentra operando normalmente. Esto genera una atención predictiva a la maquinaria para determinar si se debe reparar o no. La gran mayoría de las veces, la maquinaria industrial debe ser completamente detenida antes de hacer una operación exhaustiva de su estado. De esta forma, si bien un falso positivo es menos costoso que un falso negativo de por sí, una alta presencia de falsos positivos puede ser más costoso que los pocos falsos negativos presentados ya que es necesario detener la operación para revisión de las máquinas. Esta situación elimina rápidamente cualquier beneficio que un sistema predictivo puede proveer.

Los problemas que este proyecto plantea resolver son los siguientes:

1. Desarrollar modelos predictivos en el marco de conjuntos de datos altamente desbalanceados. Dónde aproximadamente el 95% de los datos recogidos representan instancias de operación normal de la maquinaria y el 5% de los datos recogidos representan instancias de fallas de la máquina.
2. Un sistema predictivo para maquinaria industrial debe tener una tasa baja de falsos negativos. Esto debido a que si no cubre efectivamente los casos en que se presenta la falla, inevitablemente los costos de operación seguirán siendo muy altos debido a las fallas de la maquinaria.
3. Dependiendo del caso de uso, los falsos positivos en gran frecuencia pueden generar costos igual de altos que los falsos negativos. Por esta razón, un modelo predictivo para mantenimiento de máquinas debe presentar una tasa baja de falsos positivos preferiblemente. Si el costo de un falso positivo es muy bajo, se puede dar más libertad respecto a esta condición.

Este problema será atacado analizando un conjunto de datos sintéticos de una máquina fresadora. Este conjunto de datos tiene información sobre diferentes tipos de fallas que pueden ocurrir al mismo tiempo. Este proyecto se planteó como un problema de clasificación donde cada registro puede pertenecer sólo a una clase. Esto se explica con más detalle en las secciones Análisis exploratorio de los datos (EDA) y Preprocesamiento de datos.

Propuesta

Para atacar este problema se plantean usar métodos de muestreo estadístico que contrarresten el efecto del desbalance de datos. La expectativa es que mediante estos métodos se pueda obtener un buen rendimiento del modelo predictivo y reducir principalmente los falsos negativos.

Los métodos a usar son los siguientes:

1. Sobremuestreo.
2. Submuestreo.
3. SMOTE que significa “técnica de sobremuestreo sintético de minorías” y proveniente del inglés Synthetic Minority Oversampling Technique.

Sistema planteado

El sistema planteado consta de dos ejes principales:

1. Notebooks: Las notebooks de Python permiten explorar y desarrollar de forma interactiva diferentes ideas y opciones.
2. Librería “mltools”: Esta corresponde a una pequeña librería desarrollada como parte del proyecto la cual permite abstraer y automatizar algunos de los procesos que se presentan en los notebooks.

Mapeo del sistema y flujo de trabajo

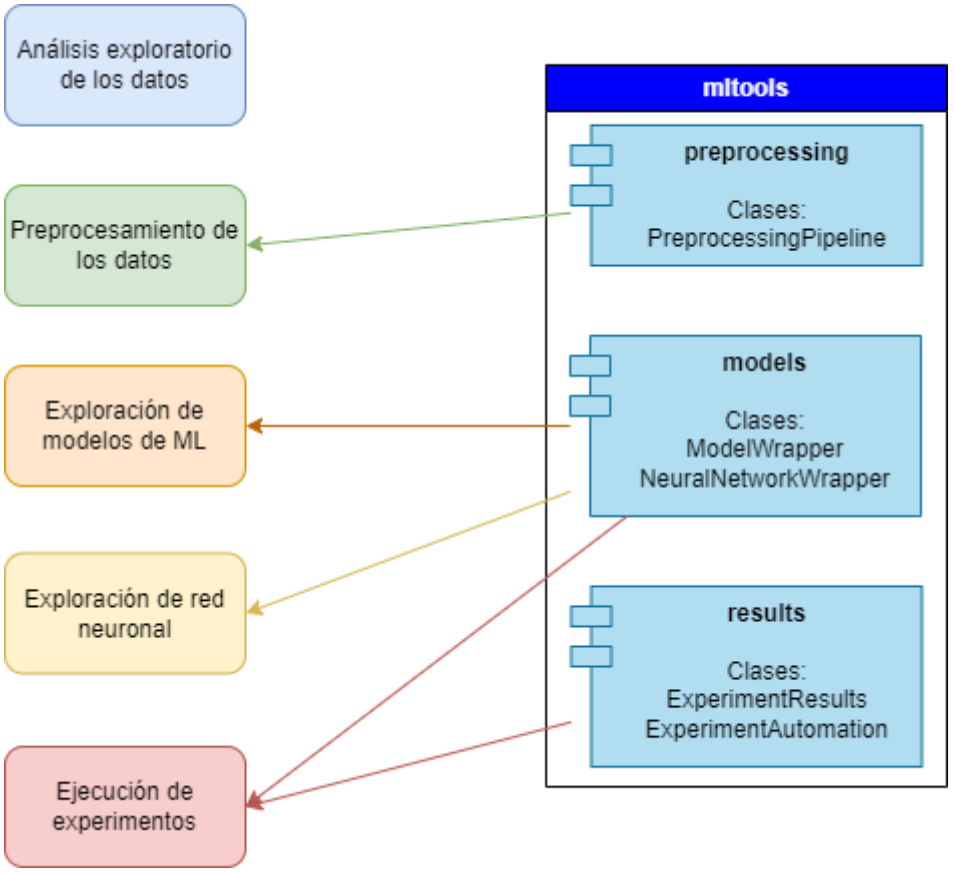

Figura 1. Mapeo del sistema.

El proceso comienza con los datos descargados y se realizan los siguientes procesos:

1. Análisis exploratorio de los datos: Este se ve implementado en la notebook EDA.ipynb.
2. Preprocesamiento de los datos: Este proceso continúa con los datos de origen y se implementa en la notebook preprocessing.ipynb. La notebook feature_importance.ipynb realiza un análisis de importancia de características usando diferentes combinaciones de características y un clasificador de bosque aleatorio.
3. Exploración de modelos de ML: Se ejecuta en models_testing.ipynb, donde se experimenta con los parámetros de los modelos de ML usados en el proyecto para encontrar unos parámetros con un rendimiento aceptable.

4. Exploración de red neuronal: Se ejecuta en neural_network_testing.ipynb. Es análogo al punto anterior pero usando una red neuronal.
5. Ejecución de experimentos: Se ejecutan las diferentes combinaciones de los modelos con las técnicas de muestreo, se almacenan y se imprimen los resultados. Esto se realiza en la notebook experiments.ipynb.

A partir de la ejecución de los procesos se van creando los módulos que la librería “mltools” en la cual pretendemos abstraer y automatizar los procesos requeridos para la ejecución del proyecto. Desarrollamos entonces los siguientes módulos cuyas interacciones con el flujo de trabajo se pueden ver en la gráfica inicial de esta sección.

1. preprocessing: Módulo con clase PreprocessingPipeline la cual ejecuta todo el proceso de preprocesamiento de forma automática con los métodos fit y transform implementados adecuadamente.
2. models: Módulo con clases ModelWrapper y NeuralNetworkWrapper los cuales se pueden usar para entrenar los modelos y botar los resultados correspondientes bajo una API unificada.
3. results: Módulo con clases ExperimentResults y ExperimentAutomation. La primera clase almacena los resultados de un experimento con cierta combinación de modelo y muestreo. El segundo automatiza la ejecución de los experimentos tomando una lista que contenga ModelWrappers y NeuralNetworkWrappers.

Conjunto de datos

El conjunto de datos a usar fue publicado como parte del paper "Explainable Artificial Intelligence for Predictive Maintenance Applications" de Stephan Matzka. El conjunto de datos puede ser encontrado en las siguientes localizaciones:

- UCI Machine Learning Repository: https://archive.ics.uci.edu/dataset/601/ai4i+2020+predictive+maintenance+dataset
- Kaggle: https://www.kaggle.com/datasets/stephanmatzka/predictive-maintenance-dataset-ai4i-2020

Columnas características del conjunto de datos

- UID: Identificador único que varía de 1 a 10000.
- Product ID: consiste en una letra L, M o H para bajo (50% de todos los productos), medio (30%) y alto (20%) como variantes de calidad del producto y un número de serie específico de la variante.
- Type: El tipo de producto L, M o H de la columna 2.
- Air temperature [K]: Representa la temperatura del aire. Es generada utilizando un proceso de paseo aleatorio normalizado a una desviación estándar de 2 K alrededor de 300 K.

- Process temperature [K]: Representa a la temperatura del proceso. Es generada mediante un proceso de paseo aleatorio normalizado a una desviación estándar de 1 K. A este proceso se le suma la Air temperature más 10 K.
- Rotational speed [rpm]: Velocidad rotacional en rpm (revoluciones por minuto). Calculada a partir de una potencia de 2860 W, superpuesta con un ruido distribuido normalmente.
- Torque [Nm]: Se obtienen a partir de una distribución normal alrededor de 40 Nm con desviación estándar de 10 Nm y sin valores negativos.
- Tool wear [min]: Las variantes de calidad H/M/L añaden 5/3/2 minutos de desgaste de herramienta a la herramienta utilizada en el proceso.

Columnas para predecir

- Machine failure: Etiqueta que es 1 si la máquina falló y 0 si la máquina no falla para ese punto. Si presenta un 1, significa que alguna de las siguientes 4 etiquetas es 1.
- Tool wear failure (TWF): Falla que representa si la herramienta que se está usando en la fresadora se desgasta. Etiqueta con 1 o 0.
- Heat dissipation failure (HDF): La disipación del calor puede causar una falla en la máquina. Etiqueta con 1 o 0.
- Power failure (PWF): Falla asociada a que el torque y la velocidad rotacional en producto son iguales a la potencia requerida para el proceso. Etiqueta con 1 o 0.
- Overstrain failure (OSF): Falla asociada al producto del desgaste de la herramienta y el torque. Etiqueta con 1 o 0.
- Random failures (RNF): Cada proceso de la máquina tiene una probabilidad de fallar 0,1% de las veces aleatoriamente sin importar los valores de operación. Etiqueta con 1 o 0.

Análisis exploratorio de los datos (EDA)

Se encontraron las siguientes características para este conjunto de datos:

1. El conjunto de datos consta de 14 columnas y 10000 filas.
2. No se encuentran valores nulos para ninguno de los valores en todo el conjunto de datos.
3. La columna “Type” está relacionada con “Product ID” ya que el primer carácter de “Product ID” debe ser igual a “Type”. Esto se verificó y es correcto.
4. Se calculó la matriz de correlación solamente usando las variables numéricas. Estas variables son: “Air temperature [K]”, “Process temperature [K]”, “Rotational speed [rpm]”, “Torque [Nm]” y “Tool wear [min]”. Se identificaron los siguientes hallazgos:
    a. “Air temperature [K]” y “Process temperature [K]” tienen un coeficiente de correlación de 0.88.
    b. “Rotational speed [rpm]” y “Torque [Nm]” tienen un coeficiente de correlación de -0.88.
    c. El resto de variables presentaron una correlación despreciable en comparación a las dos anteriormente mencionadas.
    d. La matriz de correlación **completar!!!!!!!!!!!!!!!!!!!!**

5. Las interacciones de los pares de variables (“Air temperature [K]”, “Process temperature [K]”) y (“Rotational speed [rpm]”, “Torque [Nm]”) no parecen ser lineales. Esto se puede observar en la Figura 1 y la Figura 2.
6. Según la documentación, la columna “Machine failure” debía presentar un valor de 1 si alguna de las otras 5 columnas representando los diferentes tipos de fallas eran igual a 1. Procedimos a verificar esto en el conjunto de datos, sin embargo, encontramos 9 filas marcadas con “Machine failure” = 1 que no presentaban ninguna de las banderas de los tipos de fallas activadas.
7. De la misma forma que el punto anterior, encontramos 18 filas que tenían “RNF” = 1 pero tenían “Machine failure” = 0. Por los puntos 5 y 6, se decidió preprocesar estas columnas usando criterios un poco diferentes.

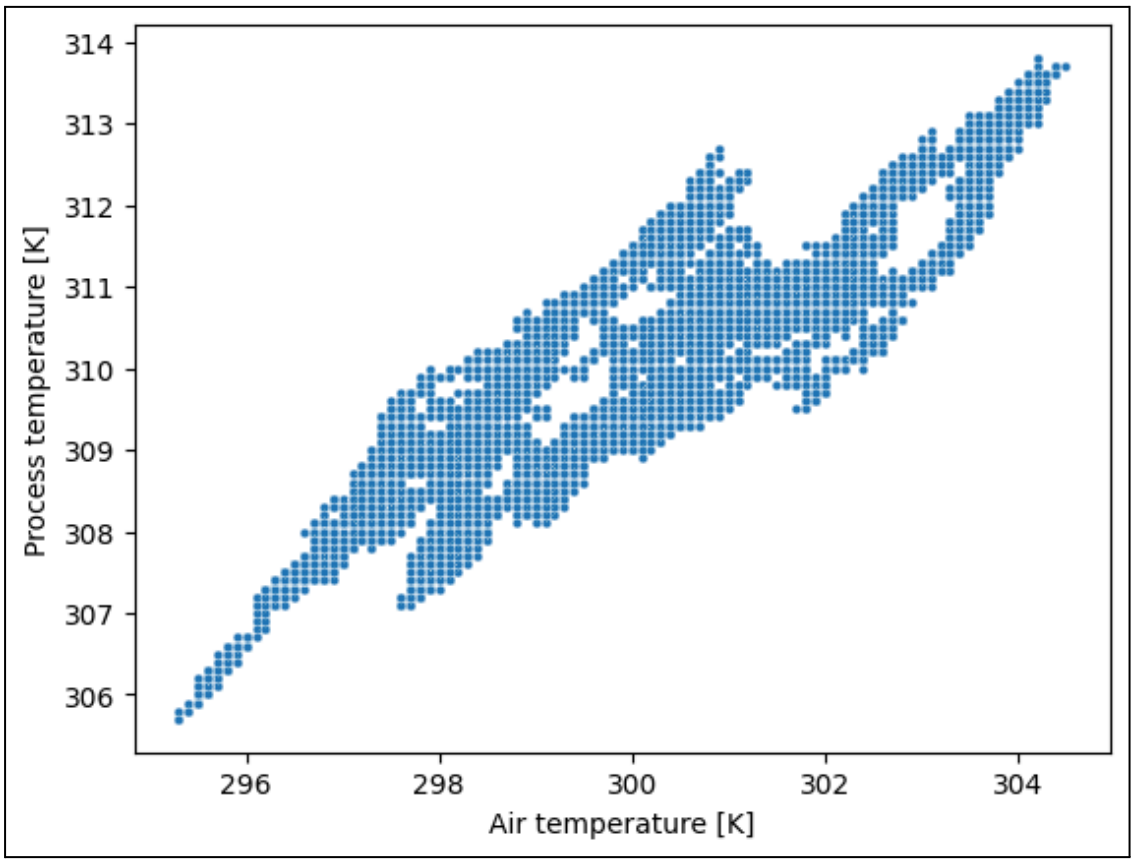

Figura 2. Gráfico de dispersión de las variables “Process temperature [K]” y “Air temperature [K]”.

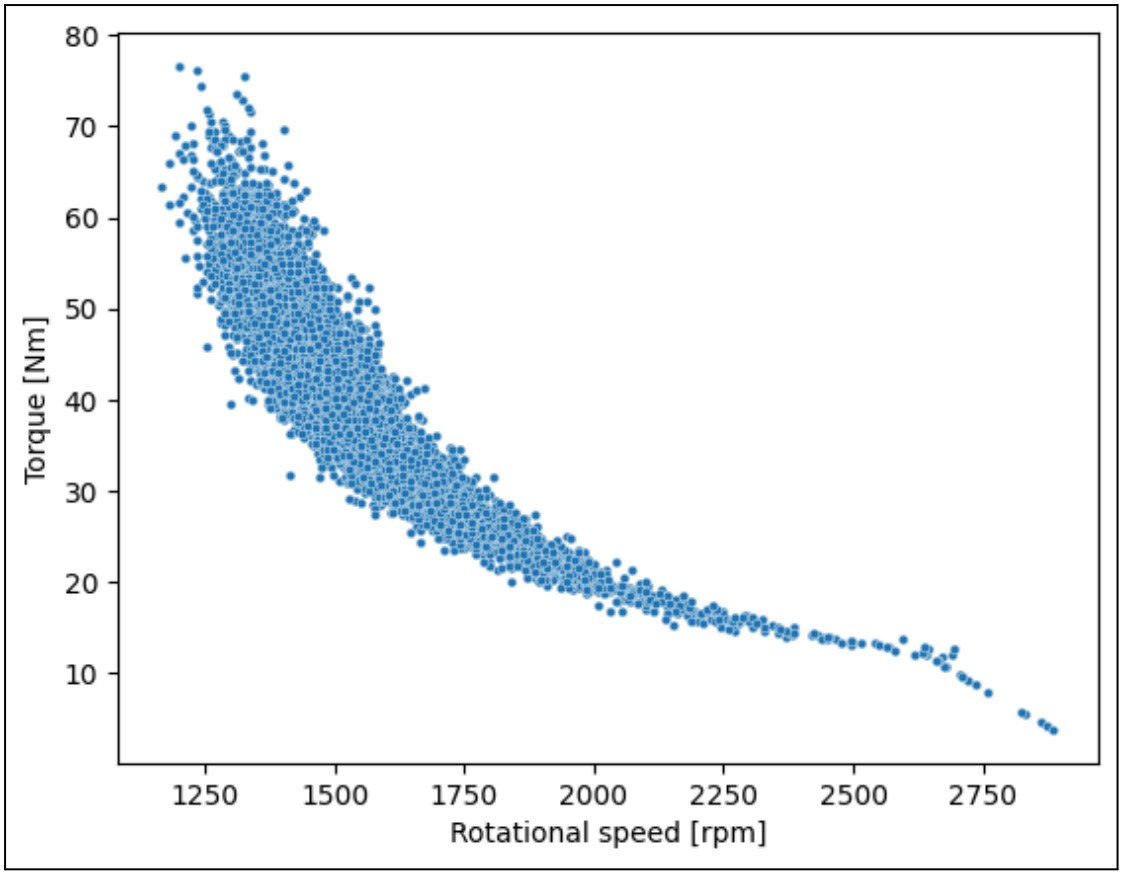

Figura 3. Gráfico de dispersión de las variables “Torque [Nm]” y “Rotational speed [rpm]”.

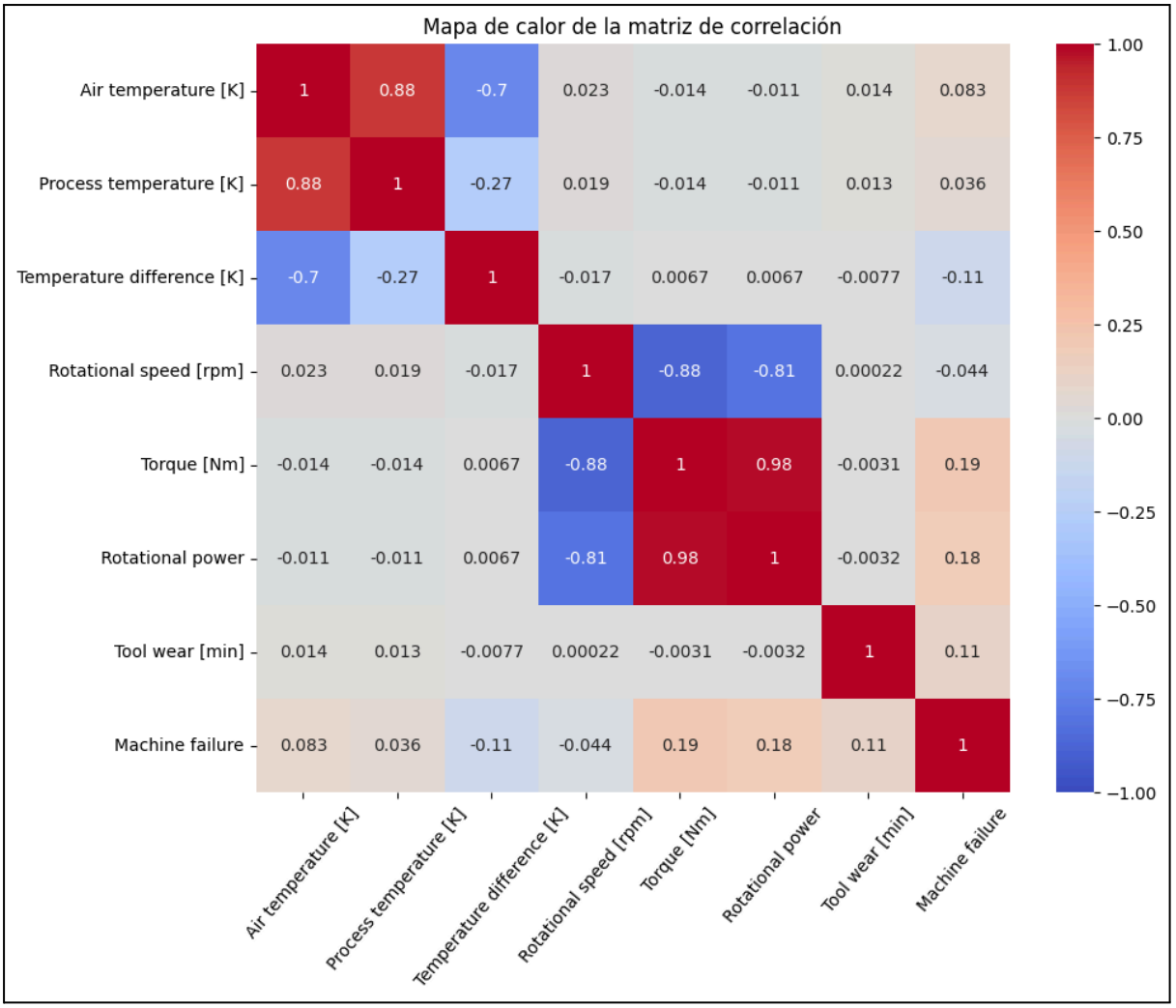

Figura 4.

Preprocesamiento de datos

Teniendo en cuenta los resultados de la sección Análisis exploratorio de los datos (EDA) se planteó el siguiente preprocesamiento de los datos:

1. Se descartan las variables “UID” y “Product ID” ya que estas columnas solo consisten de identificadores.
2. Solo existe una variable categórica “Type” a la cual se le aplicó one-hot encoding obteniendo dos nuevas columnas “Type_L” y “Type_M”. Se descarta la primera columna “Type_H” ya que esta información es redundante al estar presente las otras dos columnas.
3. La variable objetivo “Failure type” se origina a partir de las variables “TWF”, “HDF”, “PWF”, “OSF” y “RNF” usando el siguiente criterio:
    a. Se ignora la columna “Machine failure”.
    b. Si solo una de las banderas está activada para el registro, se le asigna la falla correspondiente a la columna que está activada.
    c. Si se tienen dos fallas y una de ellas es “RNF”, se asigna la que no es “RNF”. Esto debido a que las fallas “RNF” son fallas aleatorias, por lo que en este caso sería información irrelevante.

    d. Si se tienen dos fallas y ninguna de ellas es “RNF” se le asigna una nueva etiqueta “Mútiples fallas” representada por el número 6.
4. La variable objetivo “Failure type” presenta un esquema similar al de “LabelEncoder”. En este caso no usamos scikit-learn sino que creamos la función nosotros mismos. Se genera la variable objetivo “Failure type” con las siguientes etiquetas:
    a. 0 - No hay falla.
    b. 1 - TWF (Tool Wear Failure).
    c. 2 - HDF (Heat Dissipation Failure).
    d. 3 - PWF (Power Failure).
    e. 4 - OSF (Overstrain Failure).
    f. 5 - RNF (Random Normal Failure).
    g. 6 - Múltiples fallas.
5. Se generaron dos variables adicionales teniendo en cuenta los puntos 4 y 5 de la sección Análisis exploratorio de los datos (EDA). Estos términos de interacción buscan dar cuenta de la no-linealidad que se observa en las figuras 1 y 2. Las dos variables adicionales son:
    a. “Temperature difference [K]” = “Process temperature [K]” - “Air temperature [K]”.
    b. “Rotational power” = “Torque [Nm]” x “Rotational speed [rpm]”.
6. Las variables numéricas se procesaron de la siguiente forma:
    a. Se añaden las variables descritas en el punto 5 al conjunto de datos.
    b. Se hace la división del conjunto de datos en datos de entrenamiento y datos de prueba. Los datos de entrenamiento corresponden a 8000 registros y los datos de prueba a 2000 registros.
    c. Al hacer la división, los datos fueron estratificados basados en la variable “Failure type”.
    d. A todas las 7 variables numéricas “Air temperature [K]”, “Process temperature [K]”, “Rotational speed [rpm]”, “Torque [Nm]”, “Tool wear [min]”, “Temperature difference [K]” y “Rotational power” se les aplica un StandardScaler. Este se fittea con los datos de entrenamiento y luego la transformación es aplicada a ambos conjuntos: entrenamiento y prueba.

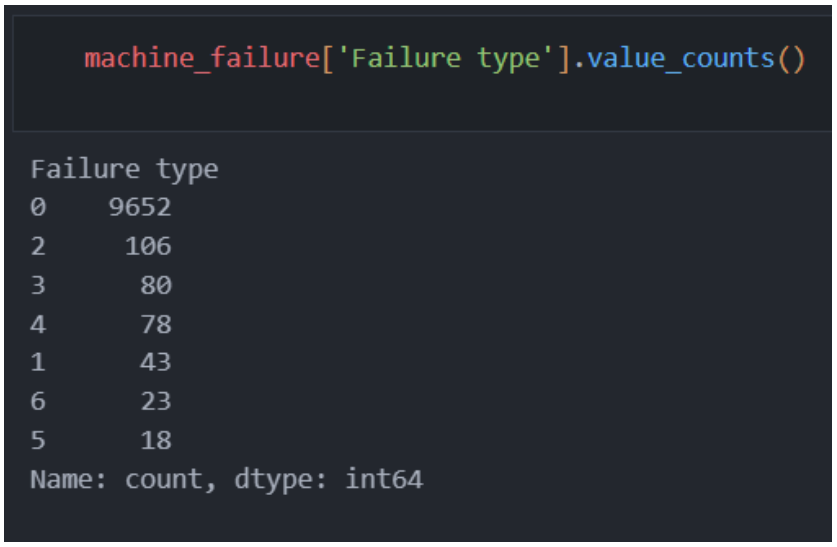

Figura 5.

Modelos de aprendizaje automático

Se usaron 3 métodos de Machine Learning y una red neuronal para este proyecto.
Los métodos de Machine Learning fueron los siguientes:

1. Clasificador de bosque aleatorio. Se uso la implementación de RandomForestClassifier de scikit-learn.
    a. “n_estimators”: 200.
    b. “max_depth”: 10.
    c. “min_samples_leaf”: 10.
    d. “min_samples_split”: 10.
    e. “class_weight”: “balanced”.
    f. “max_features”: 5.
    g. “criterion”: “gini”.
2. Clasificador de soporte vectorial. Se usó la implementación SVC de scikit-learn.
    a. “C”: 8.0. Corresponde al parámetro de regularización.
    b. “kernel”: “rbf”. Radial basis function.
    c. “gamma”: “scale”. Coeficiente del kernel.
    d. “decision_function_shape”: “ovr”. Estrategia one-vs-rest (uno contra el resto).
    e. “class_weight”: “balanced”.
3. Regresión logística. Se usó la implementación LogisticRegression de scikit-learn.
    a. “C”: 0.5. Parámetro de regularización.
    b. “solver”: “lbfgs”.
    c. “max_iter”: 1500.
    d. “multi_class”: “multinomial”.
    e. “class_weight”: “balanced”.

La red neuronal está definida de la siguiente forma:

```
# Define the neural_network
neural_network = tf_models.Sequential([
    layers.Dense(64, activation='relu', input_shape=(7,)),  # 7 input features
    layers.Dense(32, activation='relu'),
    layers.Dense(7, activation='softmax')  # 7 classes for the output
])

# Compile the neural_network
neural_network.compile(optimizer='adam',
                       loss='categorical_crossentropy',
                       metrics=['recall', 'f1_score'])
```

✓ 0.6s

Figura 6.

Técnicas de muestreo usadas

Todo el uso de las técnicas de muestreo está basado en la librería imbalanced-learn.

Sobremuestreo y submuestreo

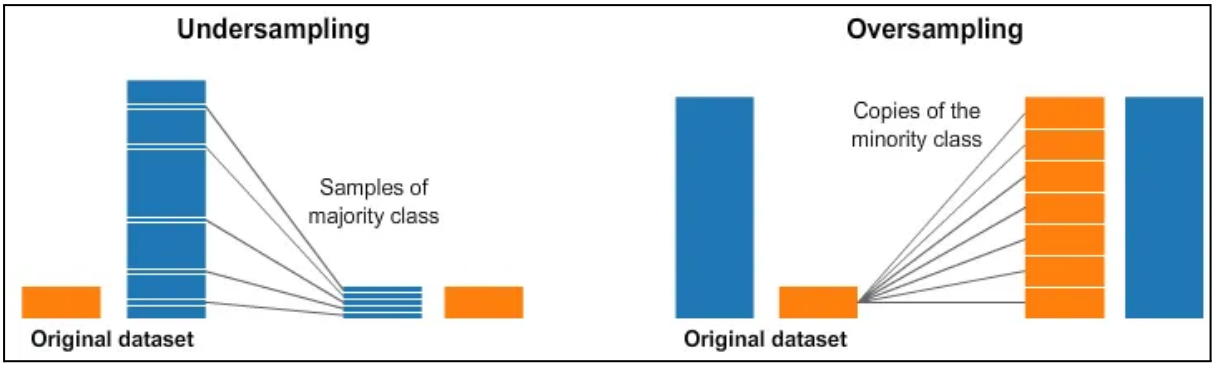

Figura 7.

Para el sobremuestreo y submuestreo se usaron las clases RandomOverSampler y RandomUnderSampler respectivamente. Estas clases provienen de la librería imbalanced-learn y ejecutan el muestreo de tal forma que las clases se vean igualmente representadas de diferentes formas.

El sobremuestreo corresponde a tomar las clases minoritarias y generar muestras tomando registros aleatorios de esta clase con reemplazo, es decir, con la posibilidad de tomar el mismo registro varias veces. Los registros son tomados hasta que las clases minoritarias se encuentren en la misma cantidad que la clase mayoritaria.

El submuestreo por otro parte, corresponde a tomar las clases mayoritarias y escoger muestras aleatorias sin reemplazo hasta que todas las clases tengan la misma frecuencia que la clase minoritaria.

SMOTE

La técnica de Synthetic Minority Oversampling Technique corresponde a una técnica con el mismo principio que el sobremuestreo aleatorio. En esta, se busca generar más muestras de las clases minoritarias. La diferencia con respecto al sobremuestreo aleatorio es que las muestras generadas no son exactamente iguales a las muestras existentes. Se generan nuevos registros interpolando los valores numéricos entre los puntos de las clases minoritarias hasta que todas las clases se encuentren representadas con la misma frecuencia que la clase mayoritaria.

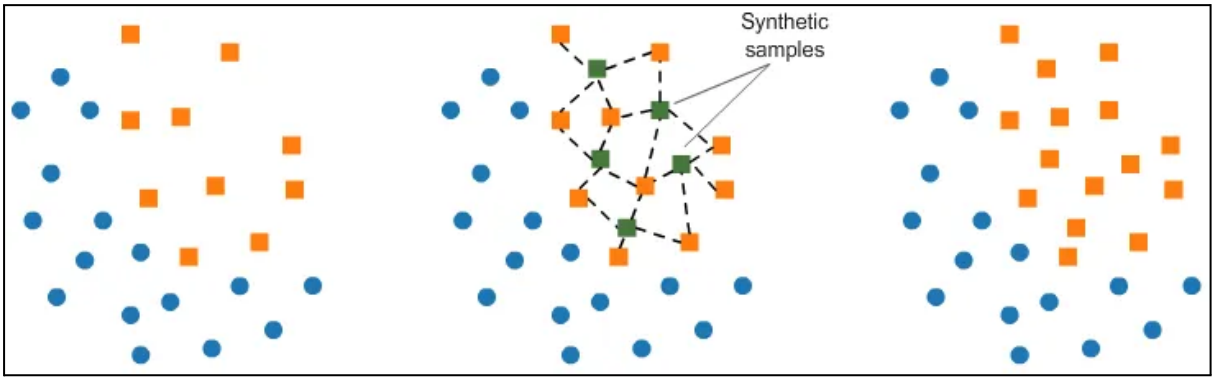

Figura 8.

Métricas analizadas

Para este proyecto analizamos las siguientes métricas con el siguiente orden de prioridad:

1. Exhaustividad o recall.
2. F1-score.
3. Precisión.

Aunque tratamos un problema de clasificación, la exactitud o accuracy no será tenido en cuenta. Esto se debe a que debido al desbalance de clases, un modelo que siempre prediga que no va a haber fallos va a acertar el 96% de las veces. El desbalance de clases hace que la exactitud pierda su utilidad para describir la bondad de cada modelo en nuestro caso de estudio.

Exhaustividad o recall

La exhaustividad en el caso de clasificación multi-clase se define para cada una de las clases. Para la clase k, la exhaustividad se define como la razón entre la cantidad de muestras que fueron correctamente clasificadas (verdaderos positivos) y el total de muestras que realmente corresponden a esa clase (suma de los verdaderos positivos y los falsos negativos).

La razón por la que esta es la métrica más importante para nuestro caso de estudio es porque el principal objetivo corresponde a cubrir correctamente todas las instancias en las que efectivamente se presenta una falla. Así, el tener una exhaustividad más alta implica que la clase de falla se haya cubierto correctamente en la predicción y los falsos negativos son menores.

F1-score

El F1-score corresponde a la media armónica entre la exhaustividad y la precisión. La razón por la que esta métrica es importante para este estudio es porque nos brinda el comportamiento de ambas métricas de las que está compuesta en una sola métrica. Así, tener una exhaustividad y un F1-score alto, implica que tanto la tasa de falsos negativos como la tasa de falsos positivos son bajas. Finalmente, tener un modelo predictivo con una alta cantidad de falsos positivos tampoco funciona para una implementación en un escenario real.

Precisión

La precisión, para una clase k, se define como la cantidad de registros correctamente clasificados de esa clase (verdaderos positivos) sobre la cantidad de registros que fueron predichos para esa clase por el modelo (suma de los verdaderos positivos más los falsos positivos). Esta métrica es considerada sólo porque brinda información complementaria a las otras dos ya mencionadas.

Resultados

Teniendo en cuenta los modelos presentados y las técnicas de muestreo expuestas se generaron 16 variantes entre combinaciones de modelo y técnica de muestreo. Así, cada uno de los modelos fue entrenado sin muestreo, con sobremuestreo, con submuestreo y con SMOTE. Los parámetros de los modelos no variaron en las diferentes combinaciones con las técnicas de muestreo. De la misma forma, los parámetros de las técnicas de muestreo no variaron con los diferentes modelos. De esta forma nos aseguramos que podemos comparar los modelos en igualdad de condiciones.

recall	model
0.79	random_forest_oversampling
0.78	neural_network
0.77	random_forest
0.76	random_forest_smote
0.76	logistic_regression_smote
0.76	logistic_regression_oversampling
0.76	logistic_regression
0.75	svc
0.72	svc_oversampling
0.72	neural_network_undersampling
0.72	svc_smote
0.69	neural_network_smote
0.68	random_forest_undersampling
0.68	svc_undersampling
0.67	neural_network_oversampling
0.67	logistic_regression_undersampling

f1-score	model
0.66	random_forest_oversampling
0.62	random_forest_smote
0.61	random_forest
0.59	neural_network_smote
0.57	svc_smote
0.56	svc
0.56	svc_oversampling
0.55	neural_network_undersampling
0.54	neural_network_oversampling
0.38	logistic_regression_smote
0.35	logistic_regression_oversampling
0.35	logistic_regression
0.33	random_forest_undersampling
0.31	svc_undersampling
0.30	neural_network
0.25	logistic_regression_undersampling

Figure 9.

precision	model
0.62	random_forest_oversampling
0.58	random_forest_smote
0.57	neural_network_smote
0.55	random_forest
0.54	neural_network_oversampling
0.52	svc_smote
0.50	svc_oversampling
0.49	svc
0.49	neural_network_undersampling
0.37	random_forest_undersampling
0.33	logistic_regression_smote
0.31	logistic_regression_oversampling
0.30	logistic_regression
0.28	svc_undersampling
0.25	neural_network
0.23	logistic_regression_undersampling

Figure 10.

Análisis de resultados

1. Podemos observar que el clasificador de bosque aleatorio tiene muy buen comportamiento con estos datos. En general, las variantes de este modelo son las que presentan los mejores resultados.
2. Los métodos de sobremuestreo y SMOTE aumentan la exhaustividad de los modelos de bosque aleatorio y regresión logística.
3. Por debajo del clasificador de bosque aleatorio, la máquina de soporte vectorial presenta el segundo mejor rendimiento.
4. La red neuronal con SMOTE también se encuentra dentro de las combinaciones destacadas en el balance de exhaustividad y precisión. Sin embargo, presenta mucho menor rendimiento en la exhaustividad con otros modelos.
5. En general, se observa que la aplicación del submuestreo causa que los modelos tengan un rendimiento subóptimo.

Conclusiones

1. Las técnicas de sobremuestreo y SMOTE corresponden a un acercamiento adecuado para problemas con conjuntos de datos altamente desbalanceados.
2. La técnica de submuestreo presenta un perjuicio para el entrenamiento de modelos muy desbalanceados. En particular cuando la clase minoritaria es extremadamente poco representada.
3. Se sugiere como forma de resolver este problema en un ambiente industrial, implementación de IoT para tener acceso a la información y plantear sistemas que funcionen mediante aprendizaje semi-supervisado. Que se tenga una base y a medida que se acumulan más datos, los modelos se re-entrenen.

Bibliografía

Data México. (2024, Junio 1). *Data México* | *Industrias manufactureras*. Industrias Manufactureras: Salarios, producción, inversión, oportunidades y complejidad | Data México. Retrieved Julio 28, 2024, from https://www.economia.gob.mx/datamexico/es/profile/industry/manufacturing

McKinsey. (2021, July 19). *New potential from analytics-driven maintenance technologies*. McKinsey. Retrieved July 28, 2024, from https://www.mckinsey.com/capabilities/operations/our-insights/establishing-the-right-analytics-based-maintenance-strategy

S. Matzka, "Explainable Artificial Intelligence for Predictive Maintenance Applications," 2020 Third International Conference on Artificial Intelligence for Industries (AI4I), 2020, pp. 69-74, doi: 10.1109/AI4I49448.2020.00023.

Propuesta inicial del proyecto

En el mundo actual, los métodos para asistir a la industria en sus diferentes tipos de operaciones son cada vez más solicitados. Un ejemplo de esto se puede ver en la industria eléctrica. Allí, el poder predecir la falla de las maquinarias en la transmisión y distribución de energía, es indispensable para mantener a la red eléctrica estable. Esto debido a que una falla en un solo componente puede propagarse y afectar gravemente a todo el sistema.

Esquema del proyecto

Objetivo general

Predecir correctamente los eventos de falla para un activo industrial basado en sus variables teniendo en cuenta que los eventos de falla son muy poco frecuentes (clases desbalanceadas).

Metodología

Preparación de los datos

- La primera parte de esto consiste en un análisis exploratorio de datos (EDA). En esta sección se estudiarán las distribuciones de las variables, las posibles correlaciones que puedan haber entre ellas y se hace el estudio de valores faltantes y valores inválidos.
- La segunda parte corresponde en aplicar transformaciones necesarias como puede ser escalamiento de datos, descartar variables que tengan una alta correlación con otras y cualquier otra etapa de preprocesamiento necesaria.

Técnicas para clases desbalanceadas

- Inicialmente, se planteará un preprocesamiento base que no tome en cuenta el desbalance de clases que existe.
- Posteriormente, se experimentará con posibles métodos que puedan contribuir a un mejor rendimiento en la predicción de las clases desbalanceadas. Entre estos métodos se buscará probar con:
    - Sobremuestreo (Over-sampling).
    - Submuestreo (Under-sampling).
    - Técnica SMOTE.

Desarrollo del modelo predictivo

Se plantea explorar los siguientes modelos para la clasificación:

- Regresión logística.
- SVM.
- Bosques aleatorios.

- Redes neuronales densas.

Métricas para estimar el rendimiento del modelo

En este caso debemos considerar el punto de que las clases que representan las fallas tienen muy pocos ejemplos y que falso negativo en la predicción de la falla es más costoso que un falso positivo.

Para esta situación son entonces más adecuadas las siguientes métricas:

- F1-Score. Ya que pondera precisión y recall en una sola métrica.
- Matriz de confusión. Esta permite ver el efecto absoluto de las fallas en la clasificación.
- Sensibilidad o recall.
- Precisión.
- Curva ROC y área bajo la curva ROC (AUC-ROC).

Análisis de resultados y conclusiones

Esta parte se enfocará en el problema de la clasificación correcta de las fallas y los efectos obtenidos mediante las diferentes técnicas usadas. Además, habrá una indagación sobre la posibilidad de ejecutar un análisis parecido en diferentes tipos de maquinarias. Sobre todo con miras a qué tipo de estrategia deben adoptar las plantas para tener este tipo de datos, cómo recolectar los datos, qué tipos de datos deben recolectar y cómo aplicar de forma adecuada un sistema de este tipo.

Justificación del proyecto

La Inteligencia Artificial ha tenido éxito en varios ámbitos: coches autónomos, traducción entre lenguajes, generación de textos e imágenes. En un contexto industrial, una de las aplicaciones más pedidas es la de mantenimiento predictivo en cualquier tipo de activo industrial. Industrias como la industria automotriz; la industria electrónica; la industria eléctrica; entre otras, cuentan con máquinas interconectadas entre sí que forman sistemas muy complejos. Usualmente, para estas industrias el costo asociado a detener la operación debido a la falla de alguna maquinaria es muy alto. Por otro lado, los conjuntos de datos que se usan para predecir fallas (si es que alguno) están sellados bajo el secreto industrial.

Aumentar la eficiencia industrial mediante el uso de métodos predictivos tiene dos usos muy importantes:

1. Reduce los costos de operación al reducir el tiempo de inactividad no planificado.
2. Algunas industrias, como la industria eléctrica, enfrentan como desafío operar de la forma más eficiente y sostenible debido a la presión por reducir emisiones de carbono y controlar el gasto energético. Es decir, la eficiencia en la operación tiene un beneficio medioambiental.

El tipo de estudios a realizar en este proyecto son muy importantes debido a que casi no se cuentan con conjuntos de datos abiertos que sean realistas respecto al comportamiento de la maquinaria industrial. Es conocido que en la operación de una máquina, regularmente el tiempo de operación sin falla es muy alto. Por esta razón, predecir fallas se vuelve un

problema de clasificación altamente desbalanceado. Se justifica, entonces, la realización de este proyecto como un primer avance para la aplicación de las técnicas usadas en muchos más ámbitos de activos industriales.

Conjunto de datos a usar

El conjunto de datos a usar fue publicado como parte del paper "Explainable Artificial Intelligence for Predictive Maintenance Applications" de Stephan Matzka. El conjunto de datos puede ser encontrado en las siguientes localizaciones:

- UCI Machine Learning Repository: https://archive.ics.uci.edu/dataset/601/ai4i+2020+predictive+maintenance+dataset
- Kaggle: https://www.kaggle.com/datasets/stephanmatzka/predictive-maintenance-dataset-ai4i-2020

Columnas características del conjunto de datos

- UID: Identificador único que varía de 1 a 10000.
- product ID: consiste en una letra L, M o H para bajo (50% de todos los productos), medio (30%) y alto (20%) como variantes de calidad del producto y un número de serie específico de la variante.
- type: El tipo de producto L, M o H de la columna 2.
- air temperature [K]: Representa la temperatura del aire. Es generada utilizando un proceso de paseo aleatorio normalizado a una desviación estándar de 2 K alrededor de 300 K.
- process temperature [K]: Representa a la temperatura del proceso. Es generada mediante un proceso de paseo aleatorio normalizado a una desviación estándar de 1 K. A este proceso se le suma la air temperature más 10 K.
- rotational speed [rpm]: Velocidad rotacional en rpm (revoluciones por minuto). Calculada a partir de una potencia de 2860 W, superpuesta con un ruido distribuido normalmente.
- torque [Nm]: Se obtienen a partir de una distribución normal alrededor de 40 Nm con desviación estándar de 10 Nm y sin valores negativos.
- tool wear [min]: Las variantes de calidad H/M/L añaden 5/3/2 minutos de desgaste de herramienta a la herramienta utilizada en el proceso.

Columnas para predecir

- machine failure: Etiqueta que es 1 si la máquina falló y 0 si la máquina no falla para ese punto. Si presenta un 1, significa que alguna de las siguientes 4 etiquetas es 1.
- tool wear failure (TWF): Falla que representa si la herramienta que se está usando en la fresadora se desgasta. Etiqueta con 1 o 0.

- heat dissipation failure (HDF): La disipación del calor puede causar una falla en la máquina. Etiqueta con 1 o 0.
- power failure (PWF): Falla asociada a que el torque y la velocidad rotacional en producto son iguales a la potencia requerida para el proceso. Etiqueta con 1 o 0.
- overstrain failure (OSF): Falla asociada al producto del desgaste de la herramienta y el torque. Etiqueta con 1 o 0.
- random failures (RNF): Cada proceso de la máquina tiene una probabilidad de fallar 0,1% de las veces aleatoriamente sin importar los valores de operación. Etiqueta con 1 o 0.

Resultados esperados del proyecto

Mediante la ejecución del proyecto se espera:

- Plantear técnicas eficaces en la etapa de preprocesamiento para tratar con el desbalance de clases.
- Reducir a lo más mínimo los errores en los que se predice que no va a fallar, pero en realidad corresponde a una falla (falsos negativos o errores tipo II).
- Poder representar adecuadamente el rendimiento de los modelos de cara al desbalance de clases.
- Extraer información útil sobre cómo implementar este tipo de procesos de predicción de fallas para clases más generales de maquinarias.

Referencias

S. Matzka, "Explainable Artificial Intelligence for Predictive Maintenance Applications," 2020 Third International Conference on Artificial Intelligence for Industries (AI4I), 2020, pp. 69-74, doi: 10.1109/AI4I49448.2020.00023.