Universitat Oberta de Catalunya

uoc.edu

Detección y prognosis de anomalías aplicada a máquinas industriales

Fernando de Castilla Parrilla
Máster Universitario en Ciencia de Datos
Minería de datos y Machine Learning

Raúl Parada Medina
Jordi Casas Roma

9 de junio de 2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Detección y prognosis de anomalías aplicada a máquinas industriales*
Nombre del autor:	*Fernando de Castilla Parrilla*
Nombre del consultor/a:	*Raúl Parada Medina*
Nombre del PRA:	*Jordi Casas Roma*
Fecha de entrega (mm/aaaa):	06/2019
Titulación:	*Máster Universitario en Ciencia de Datos*
Área del Trabajo Final:	*Minería de datos y Machine Learning*
Idioma del trabajo:	*Español*
Palabras clave	*industrial machines, anomaly detection, failure prediction*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo.*

El empleo de técnicas de minería de datos y aprendizaje automático en el marco industrial, aplicada sobre la maquinaria que conforma los procesos, supone un ahorro importante en los costes de mantenimiento, así como un alto impacto sobre la producción gracias a la detección temprana de problemas que provoquen indisponibilidades de estos equipos. Mediante la identificación de eventos anómalos acontecidos sobre estos equipos a lo largo de sus datos históricos de operación, se persigue el objetivo de predecirlos a futuro con la suficiente antelación y confianza, que permita planificar la reparación o sustitución del equipo previamente al fallo, con un coste económico más reducido. Además, la obtención de un índice de salud que mida el rendimiento de las máquinas resulta fundamental para planificar acciones de reparación sobre las mismas. El proyecto se ha planteado sobre un conjunto de datos de más de 2 millones de registros con información sobre el funcionamiento de 1900 máquinas durante varios años de operación.

Índice

1. Introducción

1.1 Contexto y justificación del Trabajo

Este trabajo se enmarca en el ámbito industrial. Dentro de la industria existen multitud de procesos en los que intervienen una serie de máquinas, cuyo objetivo común es la producción de un bien. La correcta operación y mantenimiento de estos equipos, que pertenecen a cada proceso, es un aspecto crítico que repercute en la propia disponibilidad del negocio. En esta nueva era de la industria 4.0, la revolución de los datos y el internet de las cosas, cada vez más, el entorno industrial demanda una reducción en los costes de operación y mantenimiento para seguir siendo competitivos. La manera más rentable de afrontar el reto se centra en la minimización de los periodos de indisponibilidad de las máquinas y equipos industriales, aplicando técnicas de aprendizaje automático y profundo, que permitan anticipar con una alta confianza y con la mayor antelación posible los modos de fallo que afectarán a cada máquina. A partir de los datos históricos que se disponen, se planteará la obtención de un aviso temprano de anomalía, alertando de un problema inminente que precisará de una intervención para repararlo o sustituirlo.

1.2 Objetivos del Trabajo

Los objetivos se han disgregado en objetivos principales y objetivos parciales (o hitos), que son necesarios para alcanzar un determinado objetivo principal:

- Detección de anomalías en máquinas industriales.
    - Análisis exploratorio de los datos.
    - Aplicación de técnicas de reducción de la dimensionalidad como PCA (Principal Component Analysis, para reducción de la dimensionalidad), o ‘AutoEncoders’, que constituyen una arquitectura particular de red neuronal.
    - Técnicas de clasificación como SVM (Support Vector Machines) lineal, regresión, o ‘Random Forest’, para aplicar separación lineal sobre las muestras.

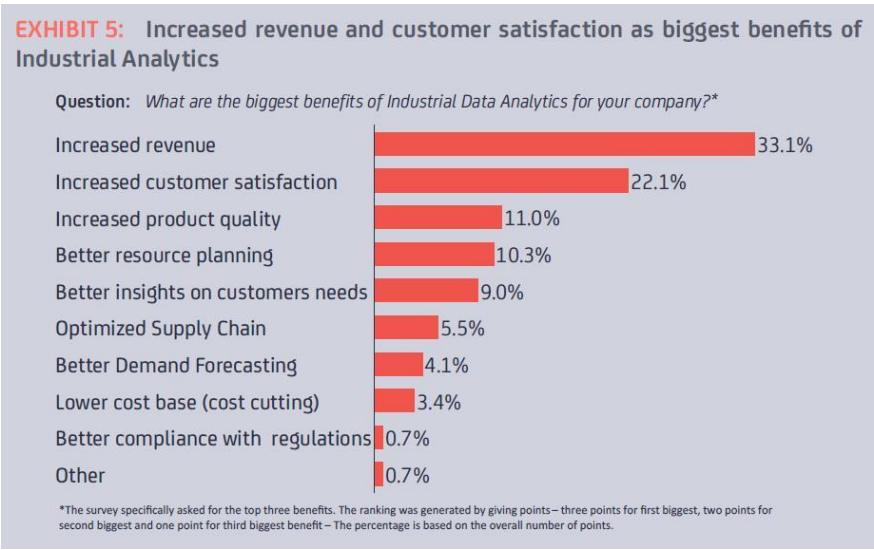
- Comparación de aciertos entre las anomalías detectadas y las anomalías ya catalogadas en el conjunto de datos.

- Predicción de las anomalías con varios días de antelación.
    - Aplicación de redes neuronales u otras técnicas sobre series temporales:
        - Multilayer perceptron
        - Convolutional Neural Networks
        - Long Short-Term Memory, dentro de las Recurrent Neural Networks
        - Resto de técnicas de aprendizaje automático.

- Obtener un índice de salud de las máquinas (RUL, Remaining Useful Life).

1.3 Motivación personal

La temática que aborda este trabajo es de total actualidad y relevancia en el marco industrial, tal y como se puede percibir en la ingente y creciente cantidad de servicios de analítica avanzada [1] que ofrecen terceras empresas como valor añadido dentro del contexto del mantenimiento predictivo [2]. Cada vez más, las empresas compiten por la supervivencia en un contexto altamente competitivo, en el que no aprovechar las bondades que puede proporcionar el paradigma de los datos, supone una desventaja insalvable frente al resto de competidores de mercado. El siguiente cuadro refleja cuáles son los mayores beneficios para las empresas industriales en el ámbito de la analítica de datos.



Resultados de encuesta realizada entre empresas del sector industrial que trabajan en analítica de datos [3]

Personalmente, acumulo una experiencia de 6 años en el sector de las energías renovables, concretamente termo-solar, que me permiten disponer de

un conocimiento de negocio válido y suficiente para poder aplicar mediante el paradigma de los datos [4].

Por tanto, si bien no dispongo de experiencia específica aplicando estas técnicas de aprendizaje automático en este entorno, este proyecto me aportará conocimientos muy útiles y específicos para poder aplicar en mi labor profesional.

Centrando el asunto en la industria de generación de energía, el Internet Industrial de las Cosas (IIoT) se ha convertido en todo un estándar de facto. En una primera fase, las empresas del sector desean tener todos sus equipos y máquinas de proceso totalmente sensorizadas, capturando datos con una frecuencia muy elevada (vibraciones, por ejemplo, para poder obtener espectros en frecuencia de máquinas rotativas), valorando tener un repositorio ideal en la nube para almacenar toda esa información proveniente de decenas de fuentes y sistemas auxiliares diferentes, y procesándola en tiempo real. A continuación, al margen de la propia operación de la planta de energía desde el centro de control, se plantea la construcción de un sistema de monitorización basado en condición que habilite una notificación y gestión de alarmas adecuados. No obstante, todo este esfuerzo no es suficiente, cuando pasados unos años de operación de la planta, los equipos y máquinas comienzan a degradarse, a requerir un mantenimiento correctivo (en el mejor de los casos, bajo costos no optimizados, preventivo) y, en definitiva, a proceder con paradas no programadas de planta, impactando de forma crítica en la cuenta de resultados. En este punto, el mayor recurso que aún queda por poner en valor no es otro que el histórico de datos almacenados durante los años previos. Aquí es donde entra en juego el paradigma de los datos: se buscará un periodo inicial de tiempo en el que la operación no fuera anómala, se detectarán anomalías en adelante en las señales de los equipos, el personal de negocio las analizará, se razonarán las causas de cada una, se etiquetarán manualmente, se confeccionará un modelo de Machine Learning que las clasifique, y se definirán indicadores que puedan estimar el tiempo restante hasta el próximo fallo. En ese punto, la compañía habrá dado un salto cualitativo hacia un mantenimiento predictivo, quedando tan sólo a un paso del mantenimiento prescriptivo, en el que el output de los modelos de datos desarrollados se transforme en una orden de trabajo con fecha de realización.

1.4 Enfoque y método seguido

Para lograr los diferentes objetivos planteados en el trabajo se utilizará un enfoque eminentemente práctico, tal y como aboga la propia ciencia de datos.

El ciclo de vida de la Ciencia de Datos es un proceso iterativo en el que se pretende alcanzar los objetivos mediante la búsqueda y obtención de datos que puedan aportar un valor diferencial en el conocimiento extraído por las diferentes técnicas de modelado existentes en el campo del Machine Learning y el Deep Learning.

Otro de los pilares en la Ciencia de Datos es la reutilización. Continuamente están desplegándose librerías de código que incluyen funcionalidades, cada vez, de más alto nivel. También aparecen herramientas y plataformas que están orientadas a acelerar ciertas fases del ciclo de vida. Y no menos importante, existe una amplia variedad de documentación al alcance de la mano con la que poder acometer un estudio del estado del arte sobre la parte teórica de una materia en concreto y sobre cómo aplicarla de una manera acorde a la naturaleza de los datos que se disponen.

Teniendo en cuenta estas premisas, se abordará este proyecto en el que, si bien se obtendrá un producto nuevo, por otro lado, cabe resaltar que dicho producto estará conformado por una conjunción de conocimientos teóricos y pragmáticos previamente utilizados en el mundo real, y convenientemente adaptados a nuestros propósitos.

1.5 Planificación del Trabajo

Los recursos mínimos necesarios son:

- Conjunto de datos históricos de operación de máquinas industriales.

- Instalación local de un cliente de Jupyter Notebook y librerías necesarias (matplotlib, widgets, seaborn, pandas, numpy, scikit-learn, keras, tensorflow, etc).

En cuanto a la planificación general del proyecto, a continuación, se detalla mediante un diagrama de Gantt el desglose de tareas, dependencias entre éstas y duración planificada. Las tareas asociadas a cada entrega parcial de la asignatura están clasificadas mediante una leyenda de colores:

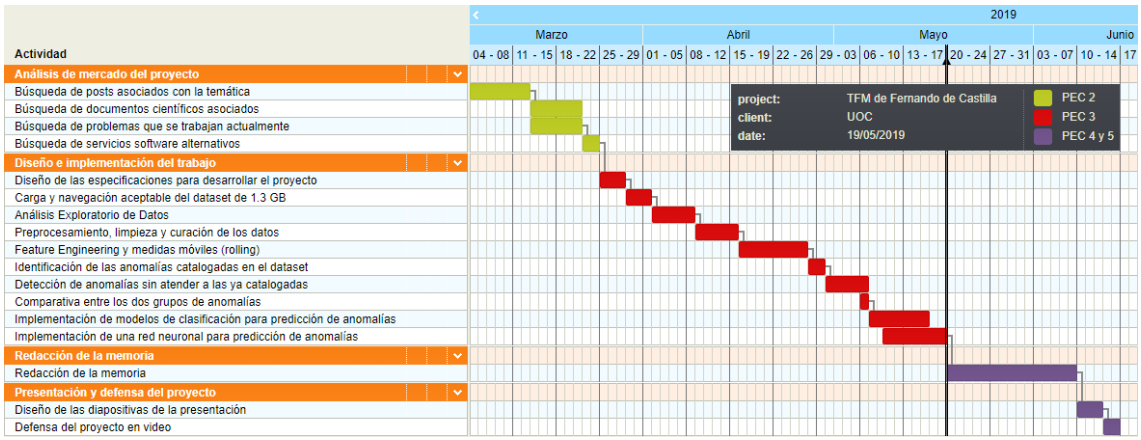


Diagrama de Gantt con la planificación global del trabajo

Se trata de una planificación total a 15 semanas (del 4 de marzo de 2019 al 14 de junio de 2019), con cinco días hábiles por semana. La planificación se divide en cuatro grandes bloques:

- Análisis de mercado del proyecto: 3 semanas de duración.
- Diseño e implementación del trabajo: 8 semanas de duración.
- Redacción de la memoria: 3 semanas de duración.
- Presentación y defensa del proyecto: 1 semana de duración.

Respecto de la primera entrega (tareas en color verde), se enumeran cuatro tareas enfocadas al análisis de mercado del proyecto. En una etapa inicial, se llevará a cabo una consulta de recursos web y posts (no científicos) que permitan vislumbrar las líneas de trabajo llevadas a cabo por otros, y así orientar la investigación exhaustiva de documentación científica específica relacionada. La tercera tarea del bloque se enfoca en la búsqueda de problemas similares en los que se ha acometido una aplicación práctica de los contenidos teóricos previamente recopilados. Como tarea adicional, se realizará una prospección de herramientas software en el mercado que aporten valor en la resolución del trabajo, ya sea facilitando el proceso o acelerando su coste temporal de implementación.

El segundo bloque de tareas (en color rojo) se engloban dentro del diseño e implementación del trabajo. Una vez analizado el estado del arte, como primera tarea se procede a establecer las especificaciones generales que persigue este trabajo. A continuación, se reserva una tarea para posibles dificultades que puedan surgir a la hora de cargar y tratar el amplio conjunto de datos de que se dispone (1.3 GB). La próxima tarea es la realización de un análisis exploratorio de datos que permita visualizar las características generales del conjunto de datos. Previamente a la aplicación de las diferentes técnicas de modelado de

datos, se reserva una tarea para toda la preparación de datos previa. Antes de pasar al desarrollo de modelos, se hace imprescindible la planificación de una tarea de Feature Engineering, que genere las variables apropiadas que dote de calidad al modelo. Parte de estas variables se obtendrán mediante medidas estadísticas móviles. Dentro de las tareas de modelado, se plantea una detección de anomalías mediante diferentes técnicas de agrupamiento y clasificación de datos. A continuación, se evalúa la bondad de la detección de anomalías llevada a cabo, comparando con las propias anomalías etiquetadas dentro del conjunto de datos. Finalmente, en las tareas de predicción de anomalías, se plantea el diseño de diferentes técnicas de modelado de datos (redes neuronales, Support Vector Machines y combinación de modelos) que permitan adelantar indisponibilidades y fallos en las máquinas industriales.

En el tercer (último) bloque de tareas, en color morado, se reservan tareas para la redacción de esta propia memoria, la elaboración de la presentación del trabajo y la creación de un esquema que sirva de guion durante la presentación del trabajo.

1.6 Breve sumario de productos obtenidos

Tras todo el trabajo llevado a cabo en la fase de diseño e implementación del proyecto, a continuación, se enumeran todos los productos obtenidos:

- Código en lenguaje Python implementado sobre siete ficheros de Jupyter Notebook.
- Modelo de detección de anomalías basado en un AutoEncoder.
- Código Python con la implementación de cinco modelos de clasificación para predicción de anomalías.
- Dos modelos de redes neuronales artificiales (Multilayer Perceptron) para predicción de anomalías.
- Todos los conjuntos de datos de entrenamiento y de prueba generados para entrenar cada uno de los modelos.

1.7 Breve descripción de los otros capítulos de la memoria

Esta memoria dedica un capítulo a exponer y presentar las vías de información aplicables al proyecto, de entre todas las que se han investigado, para obtener un análisis detallado del estado del arte, previamente al inicio de la fase de desarrollo e implementación del proyecto.

El siguiente capítulo plasma todas las especificaciones que rigen el desarrollo del trabajo. Por tanto, en él se enumeran una serie de requisitos cuyo cumplimiento es obligatorio, al igual que se enumeran otros requisitos cuyo cumplimiento sería deseable.

A continuación, se detallan todos los capítulos que están vinculados directamente con la fase de diseño e implementación del proyecto:

- Análisis exploratorio de datos. Se trata de un capítulo dedicado a mostrar de una manera gráfica y visual toda información que ayude a orientar los pasos posteriores del trabajo. Este tipo de análisis permite que el científico de datos orientado a negocio pueda comprender la naturaleza de los datos brutos, otorgándole una visión más clarificadora del negocio.
- Preprocesamiento de datos. Este capítulo aplica toda una serie de trasformaciones iniciales sobre los datos de partida, con el objetivo de hacerlos explorables, operables y modelables. Estas transformaciones suelen estar más enfocadas en aspectos muy particulares de los datos de que se dispone, por lo que pueden no ser aplicables en otros conjuntos de datos.
- Feature engineering: La ingeniería de características podría ser incluida en el apartado de preprocesamiento de los datos. No obstante, juega un papel fundamental en la ciencia de datos, ya que va más allá de hacer los datos operables; siendo una etapa crítica para obtener saltos cualitativos en la precisión de los modelos desarrollados. En este proyecto en cuestión, resulta de gran ayuda para generar ratios que relativicen los datos, así como para elaborar medidas estadísticas móviles (rolling).
- Detección de anomalías. Este capítulo pretende obtener un modelo no supervisado de detección de anomalías. La idea fundamental de este modelo es que, durante su entrenamiento, nunca tendrá conocimiento de los registros (o muestras) etiquetados realmente como anómalos, por lo que es preciso obtener un indicador que discierna aquello que se considera anómalo y lo que no. En ello juega un papel clave el umbral que se establezca, con el objetivo de minimizar los falsos positivos, o bien, los falsos negativos.
- Prognosis: Este capítulo se centra en todos los modelos supervisados de clasificación que pueden ayudar a predecir anomalías con una cierta antelación. Para ello, se trabajan dos bloques: modelos basados en redes neuronales artificiales (Multilayer perceptron), y modelos basados en otros clasificadores del estado del arte.

Tras estos bloques mencionados dentro del diseño e implementación del proyecto, se dedica un capítulo para realizar una valoración económica del trabajo.

Finalmente, las conclusiones del proyecto se agruparán en un capítulo propio. El último capítulo está dedicado a registrar todas las referencias bibliográficas aportadas durante la realización del proyecto.

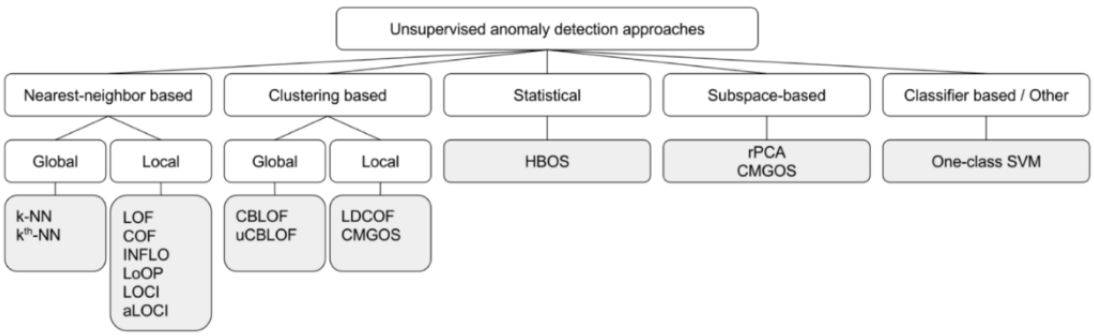
2. Estado del arte

El análisis del estado del arte se organiza en tres bloques. El primer bloque agrupa toda la documentación aplicable al proyecto en referencia a las técnicas existentes para detectar anomalías estadísticas en los equipos, con especial atención en el reconocimiento avanzado de patrones. El segundo bloque agrega toda la documentación relativa a los diferentes modelos de aprendizaje automático y profundo (redes neuronales) empleados para la tarea de clasificación de anomalías etiquetadas, lo que corresponde con una fase de diagnosis de los equipos en el ámbito industrial. Además, se alcanza el objetivo de este proyecto al plantear la implementación de un índice de salud, capaz de predecir dentro de un rango de confianza en qué momento se producirá un fallo del equipo, cubriendo así la fase de prognosis en el ámbito industrial. Finalmente, se incluye un tercer bloque en el que se presenta un listado de herramientas software disponibles y útiles para el proyecto.

Un documento que plantea las bases para organizar un proyecto de esta magnitud se presenta en [5], resultando ser de utilidad para confeccionar los subapartados siguientes a la vez que provee de cuatro estrategias actualmente utilizadas para guiar este tipo de proyecto. Dichas estrategias son: detección de anomalías, modelos de clasificación para predecir fallo en una ventana de tiempo, modelos de supervivencia para predecir la probabilidad de fallo en el tiempo y modelos de regresión para predecir el indicador RUL (Remaining Useful Life).

2.1 Detección de anomalías y reconocimiento avanzado de patrones

Del recurso [6], se extrae el siguiente cuadro resumen de técnicas algorítmicas no supervisadas relativas a la detección de anomalías:



Para construir una detección de anomalías adecuada, en primer lugar, es necesario explorar las diferentes técnicas de detección de outliers empleadas actualmente, tanto univariantes como multivariantes. En [7] se detallan algunas como Z-score, Dbscan e Isolation Forests.

Un primer enfoque que se plantea para acometer la detección de anomalías se presenta en [8], en el que se define un contexto colaborativo por el cual los equipos que son similares se agrupan. Dentro de cada grupo, se construye un perfil de degradación de cada máquina y se detectan anomalías comparando entre perfiles dentro del mismo grupo. El punto clave de esta técnica radica en definir de una manera formal el concepto de similitud entre pares de máquinas, ya que además de ser físicamente similares se requiere que el contexto en el que se encuentran dentro del proceso industrial también lo sea; esto es, las condiciones externas que afecten a estos equipos deben diferir mínimamente. Este condicionante dificulta la labor dentro de este proyecto, en el que los datos de los que se dispone no permiten definir condiciones de contorno en los equipos.
Un segundo enfoque se orienta a aglutinar todas las técnicas de detección de anomalías empleadas sobre los datos de sensores: sistemas basados en reglas, redes bayesianas, modelado estadístico paramétrico y técnicas basadas en vecinos cercanos [9]. Con la idea de poner a competir estas técnicas mencionadas, en el artículo mencionado en [10] se evalúa una comparativa entre algoritmos no supervisados de detección de anomalías, indicando que los algoritmos de técnicas basadas en vecinos cercanos se comportan de una manera más precisa que los de agrupación o clustering.

Ambos enfoques que se plantean requieren la construcción de un conjunto de datos de entrenamiento en el que se garantice que los datos contenidos reflejan un comportamiento normal de las máquinas, excluyendo periodos en los que hubiese algún comportamiento anómalo. No obstante, es probable que el conjunto de entrenamiento conformado contenga valores outliers. En el artículo citado en [11] se presenta el algoritmo One-SVM como un candidato robusto a estos outliers.

Un tercer enfoque [12] consiste en construir un modelo probabilístico de correlaciones, en el que se detecten anomalías por pares de variables, lo cual conforma la base del reconocimiento avanzado de patrones.

Un enfoque más ambicioso se presenta en [13]. En lo que respecta a la detección de anomalías, se presentan dos aproximaciones de modelado, las cuales se ha comprobado que tienen una precisión similar:

- Aplicación del algoritmo PCA (Principal Component Analysis) para reducir la dimensionalidad del conjunto de datos, unido a un algoritmo de clustering que utilice la distancia de Mahalanobis. Las muestras anómalas se definen en función de su distancia hacia todos los clústers.

- Construcción de un autoencoder (arquitectura particular de red neuronal artificial) que ayude a comprimir los datos (reducir la dimensionalidad) y construirlos de nuevo, con un determinado error cuadrático medio, MSE. Las muestras anómalas se definen en función de la diferencia de su propio error frente al MSE global del conjunto de datos entrenados.

2.2 Predicción de anomalías e índice de salud

En un primer trabajo en [14], se introducen unos conceptos básicos sobre la predicción de series temporales univariantes mediante un enfoque en el que se hace uso de redes neuronales artificiales. Se ha demostrado que las redes LSTM (Long Short-Term Memory) resultan ser muy útiles y las más populares en el aprendizaje de dependencias en el largo plazo en comparación con las redes neuronales artificiales estándar [15].

Una vez se tienen predichas las anomalías, el último paso dentro del alcance del proyecto es calcular el tiempo estimado hasta que cada anomalía se traduzca en un fallo de la máquina. Esto se conoce como índice de salud, y su indicador por excelencia no es otro que el de vida útil restante (RUL). Un primer enfoque con el que obtener este indicador se centra en el análisis estadístico de los datos, realizando una revisión de los modelos actuales a la hora de estimar el RUL [16]. Un enfoque más profundo implica la creación de una arquitectura de red neuronal artificial en la que se trabaje con la distribución de Weibull, conocida por tener un comportamiento similar al que reflejan, en general, los objetos a la hora de contabilizar su deterioro con el paso del tiempo. En [17] el análisis se centra en el cálculo del RUL sobre las vibraciones de cojinetes en bombas.

Existe un caso particular [18] de uso de las redes neuronales artificiales en el que su propia arquitectura se limita a devolver una única neurona de salida. La propia red neuronal, además, hace uso de la técnica de feed-forward, la distribución de Weibull y el coeficiente de curtosis. Esta única salida, entrenada

a partir de los datos no anómalos de las máquinas, modela el porcentaje de vida consumido de la máquina, indicando que un valor del 100% refleja el fin de la vida de la máquina.

Como línea de trabajo adicional, en [19] se presenta un entorno industrial para la generación de energía eléctrica, con una aplicación práctica de redes neuronales de Elman, un caso particular de redes neuronales recurrentes. Esta arquitectura de red resulta ser muy útil para el tratamiento de series temporales, debido a su facilidad, entre otras, de alimentar salidas de una capa cualquiera sobre las neuronas de una capa anterior.

2.3 Software

Para el desarrollo e implementación de este proyecto, a continuación, se presentan las siguientes herramientas y utilidades software:

- Scikit-learn [20], librería de código Python que permite la programación de alto nivel de una serie de algoritmos de aprendizaje automático y demás funcionalidades relacionadas. Contiene funciones de detección de outliers, reconocimiento de patrones, clustering, regresión y clasificación.

- Keras [21], API creada en Python para permitir el diseño de redes neuronales a alto nivel.

- Auto-Keras [22], alternativa de código abierto a AutoML de Google, cuyo objetivo es automatizar la tarea del aprendizaje automático y profundo. Los dos principales objetivos de dicha tarea son la optimización y el equilibrado a la hora de definir la arquitectura de la red neuronal adecuada para el conjunto de datos, así como a la hora de definir el conjunto de valores de todos los parámetros de manera empírica (fine-tuning).

- TPOT [23], Tree-Based Pipeline Optimization Tool, herramienta Python que, dentro de la Ciencia de Datos, automatiza la fase de modelado o Machine Learning, mediante algoritmos genéticos. Está diseñada para usarse de la misma manera que la librería scikit-learn.

- BigML [24], servicio web con posibilidad de crear una cuenta gratuita, cuyo principal objetivo no es sólo automatizar la fase de Machine Learning (MLaaS, Machine Learning as a Service), sino también hacerlo accesible y sencillo a un público no necesariamente especialista.

2.4 Valoración del estado del arte

Una vez expuesto todo el abanico de alternativas del estado del arte, convenientemente analizadas y estudiadas, el objetivo de este apartado consiste en argumentar la aplicabilidad de éstas, particularizado a la naturaleza del conjunto de datos que se dispone.

Estos datos padecen de serias limitaciones que impiden aplicar ciertas de las técnicas presentadas con un mínimo de garantías. Más adelante se detalla exhaustivamente las características específicas del conjunto de datos, así como de las variables que lo conforman. No obstante, si bien el disponer de datos simulados no debiera afectar de forma relevante a los resultados obtenidos, es cierto que la gran mayoría de variables disponibles son de la siguiente naturaleza:

- Variables contadoras: su rango de valores no sobrepasa el rango de los números enteros positivos y cero (números naturales). Estas variables son utilizadas para llevar un conteo de repeticiones de un cierto evento discreto.

- Variables bandera (flags): también conocidas como variables lógicas o booleanas. Su rango de valores es binario (0 ó 1). Estas variables sólo indican la ocurrencia o no de un evento determinado.

- Variables estáticas en el tiempo: se trata de variables con un valor constante por cada máquina del conjunto de datos, por lo que nunca pueden actuar como señales temporales.

Un aspecto que es también crucial a la hora de plantear la aplicabilidad del abanico de técnicas descritas en el estado del arte es la granularidad de las variables. Si bien disponemos de datos de varios años, el muestreo se limita a un valor diario, lo cual elimina cualquier posibilidad de plantear un modelo de precisión basado en el análisis de series temporales. El conjunto de datos, además, adolece de variables que representen datos flotantes con un rango de valores que pertenezca al subconjunto de los números reales.

Un último escollo en los datos surge al no disponer de la suficiente variabilidad en los valores de las variables. Una gran cantidad de los datos reproducen estados normales de funcionamiento de las máquinas, frente a un pequeño subconjunto desbalanceado de muestras en las que el funcionamiento no es normal.

Entrando ya en la fase de valoración del estado del arte, dadas las dificultades previamente expuestas, en [5] se plantean cuatro estrategias desde las que poder abordar el proyecto en cuestión de forma exitosa, en base a la tipología y calidad de los datos, así como en base a las necesidades de negocio. Para este trabajo en cuestión, se afrontarán dos de las estrategias:

- Comunicación de alertas sobre un comportamiento anómalo.

- Creación de modelos de clasificación para predecir fallos dentro de una ventana de tiempo.

En cuanto a [6] y [7], al no disponer de datos flotantes y existir muy baja dispersión en los valores de las variables, no es viable el planteamiento de una detección de valores fuera de rango (outliers).

En referencia a [8], no se dispone de variables con las condiciones de contorno de las máquinas (por ejemplo, medioambientales). Tampoco se conoce detalle alguno de cada una de las 1900 máquinas más allá de su identificador, por lo que ni siquiera se pueden establecer similitudes entre ellas. Es posible que se disponga de datos tanto de una turbina, como de un servidor.

Atendiendo a [9], resulta imposible plantear reglas bayesianas debido a la casi nula variabilidad de los datos y la ausencia de variables con valores flotantes.

En [10] se aventura una conclusión precipitada sobre la preferencia de un determinado algoritmo de aprendizaje automático frente a otro. Tal afirmación está sesgada por la propia naturaleza del conjunto de datos utilizado. En [11], en cambio, se justifica la bondad de un algoritmo One-SVM, ya que se ajusta muy bien con el hecho de separar linealmente (o no) un conjunto de muestras con dos posibles clases (anomalía o no).

Se descarta la técnica de [12], ya que no se dispone de variables flotantes entre las que establecer correlaciones.

Según se describe en [13], se han validado dos enfoques diferentes de detección de anomalías, ambos con una precisión muy similar. El enfoque orientado a AutoEncoders se atisba más sencillo de implementar, además de posibilitar el modelado no lineal de los datos.

De entre los modelos presentados en [14], las redes neuronales artificiales (ANN) y las memorias de corto y largo (LSTM) plazo se presentan como alternativas viables.

El cálculo del índice de degradación (RUL) expuesto en [16] y [17] se presenta inalcanzable, debido nuevamente a la tipología de los datos disponibles, que no contienen valores con números reales (flotantes).

El análisis de series temporales que se plantea en [18] y [19] resulta inviable al no disponer de la suficiente granularidad en los datos, ya que sólo existe un valor por día.

En cuanto a la parte de software, [20], [21] y [23] se presentan como herramientas apropiadas en el desarrollo e implementación de la solución. Al respecto de AutoKeras ([22]), al tratarse de una ejecución 'voraz' de arquitecturas de redes neuronales, sin atender a la propia naturaleza de los datos, queda fuera del alcance de este proyecto. Sobre la solución comercial presentada en [24], existe un limitante a los propósitos de este trabajo. La versión gratuita del servicio limita a un máximo de 32 MB el tamaño del conjunto de datos. No obstante, en [23] se presenta una utilidad similar en cuanto a la ejecución automática de una batería de modelos, pero usando además una heurística basada en algoritmos evolutivos.

2.5 Propuesta planteada

Siguiendo las dos estrategias seleccionadas como apropiadas para el conjunto de datos existente, se plantea lo siguiente:

- Para acometer la comunicación de alertas sobre un comportamiento anómalo, se presenta el desarrollo de un modelo de detección de anomalías basado en un AutoEncoder [13].

- Para afrontar la creación de modelos de clasificación para predecir fallos dentro de una ventana de tiempo, se presenta una serie de modelos de clasificación (como el del ejemplo expuesto en [11]), así como un modelo basado en redes neuronales artificiales (tal y como se menciona en [14]).

En lo referente al software empleado, se hace uso del lenguaje Python con el entorno Jupyter Notebook, y entre otras, las bibliotecas 'widgets', 'Science Kit Learn', 'Keras', y 'TPOT' (Tree-Based Pipeline Optimization Tool).

3. Especificaciones del trabajo

Para este proyecto se dispone de un conjunto de datos sobre la operación de 1900 máquinas y dispositivos industriales, que han sido registrados durante alrededor de tres años con una frecuencia diaria (granularidad de un registro por máquina y día).

El tamaño del conjunto de datos está en torno a 1,3 Gigabytes, con unas dimensiones de algo más de dos millones de filas (registros) y 172 columnas (variables). Tanto el conjunto de datos fuente como la implementación de código desarrollada en este proyecto se encuentra especificada en el apartado de anexos de esta memoria.

Al ser un conjunto de datos simulado, no se dispone de mayor detalle de cada variable que su propio nombre y una ligera descripción. Si bien es cierto que los modelos de aprendizaje automático no necesitan conocer ninguna información adicional a los valores de las variables, esta escasez de información sí redunda negativamente en el proceso de comprensión de los datos, para una posterior aplicación adecuada del conocimiento de negocio específico que aporte una característica diferencial en el ajuste de la bondad de los modelos. El detalle de dichas variables es el siguiente:

- DeviceID: identificador de la máquina a la que pertenece el registro en cuestión.
- Date: fecha asociada a todas las variables del registro.
- Categorical_[1 to 4]: con la notación acotada entre corchetes, se indica la existencia de un rango de variables con idéntica raíz y enumeradas, en este caso, de 1 a 4. Estas variables contienen valores fijos (constantes) en el tiempo para una misma máquina dada.
- Problem_Type_[1 to 4]: número total de problemas de un determinado tipo detectados cada día sobre cada máquina.
- Error_Count_[1 to 8]: número total de errores de un tipo reportados por la máquina durante un día.
- Warning_[1 to 147]: número total de avisos reportados por la máquina durante un día.
- Fault_Code_Type_[1 to 4]: código de fallo reportado por la máquina para un día dado. Es posible que no exista reporte de fallo durante dicho día.

- Usage_Count_[1 to 2]: número total de veces que la máquina ha sido usada en cada uno de los dos modos de funcionamiento.
- ProblemReported: variable objetivo (a predecir), que indica mediante valor booleano (binario, 0 ó 1) si la máquina tuvo un problema durante dicho día.

A partir de este conjunto de datos de entrada, se pretende obtener las siguientes salidas:

- Análisis exploratorio de datos.
- Aprendizaje automático no supervisado: detección de anomalías en base a todas las variables disponibles, a excepción de la variable objetivo.
- Aprendizaje automático supervisado: predicción de anomalías (valores de la variable objetivo) mediante varios algoritmos de clasificación.
- Valoración económica del proyecto.

4. Carga y preprocesamiento de datos

Esta etapa preliminar tiene como objetivo la carga exitosa del pesado conjunto de datos de que se dispone, para que pueda ser tratado a posteriori de manera correcta, dados los diferentes tipos de datos que contiene (fecha, numérico y texto); así como de manera eficiente en cuanto a coste temporal de las operaciones de cálculo implementadas.

Dado que una carga directa del conjunto de datos no es aceptada en tiempo de ejecución por la librería 'pandas', es preciso proveer a la rutina de carga de datos de un parámetro de tipo diccionario con el tipo de datos adecuado para varias variables de texto. También se necesita asignar el tipo adecuado a la variable de fecha, junto con su formato. El separador de valores dentro del fichero de datos es la coma (',').

Una vez cargado el conjunto de datos, se realiza un preprocesamiento básico de datos, en el cual se pretende eliminar cualquier tipo de inconsistencia en los datos. Se comienza por asegurar que los campos de fecha ('Date') e identificación de máquina ('DeviceID') hacen la función de clave primaria de la base de datos; es decir, que no exista registro cuyos valores en estas dos variables esté duplicado, o bien, alguno de ellos sea vacío. Existe la opción de configurar esta pareja de variables como índice del conjunto de datos; no obstante, no se procederá en tal sentido al no enfocarse el trabajo en el procesamiento de series temporales. A partir de la variable de fecha, se crean las variables de día, mes y año ('Day', 'Month' y 'Year').

A continuación, se definen listados para agrupar diferentes variables del conjunto de datos que serán utilizadas posteriormente en la implementación. Por ejemplo, variables categóricas, variables de fecha, variables de alerta (de warning) o variables numéricas.

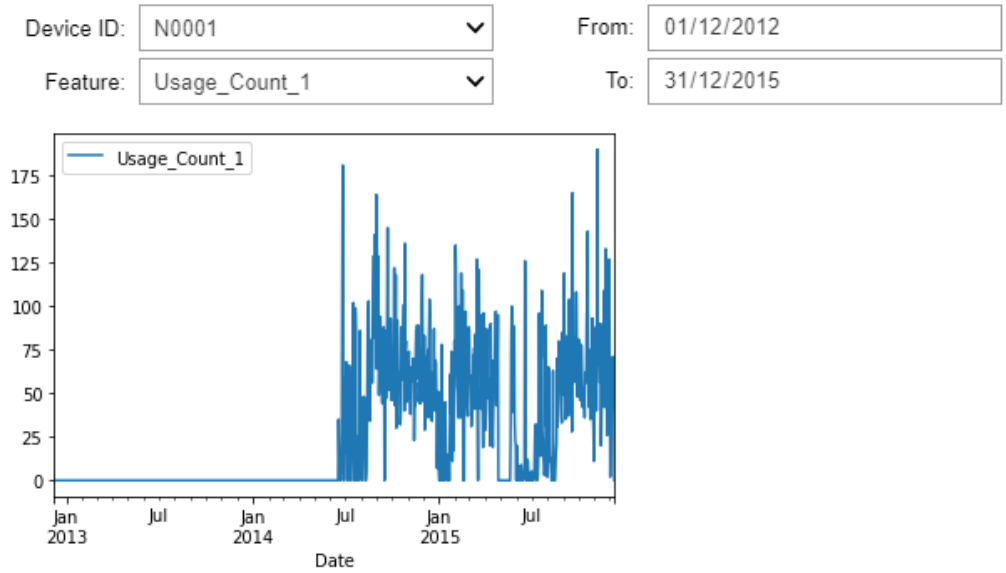
El siguiente preprocesamiento de datos trata de eliminar valores nulos en las variables numéricas (reemplazando por cero) y en las variables categóricas (texto 'Unknown').

Por último, se reemplazan todos los valores negativos de las variables numéricas por cero, ya que en toda variable que represente el número de ocurrencias de un evento, no tiene sentido físico un valor negativo.

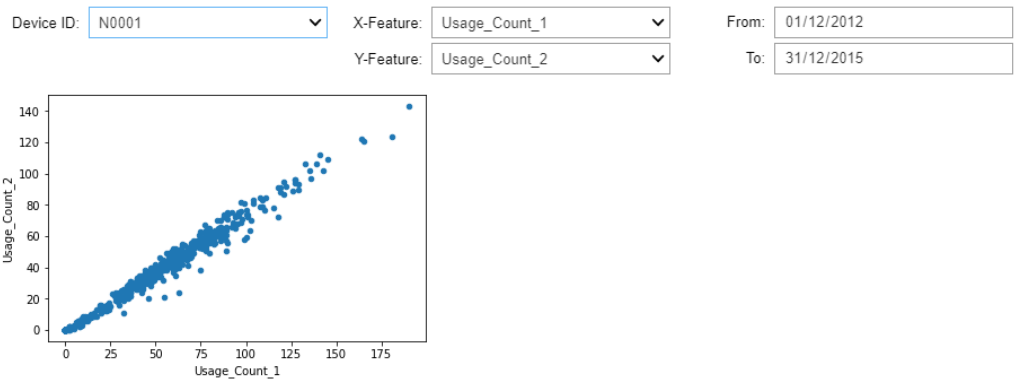
5. Análisis exploratorio de datos

Esta fase del ciclo de vida es muy importante para poder mostrar un primer análisis visual de los datos, poder obtener patrones de comportamiento y que permita orientar el rumbo hacia un análisis más profundo y pormenorizado.

En primer lugar, se propone un display de tendencia temporal en el que poder graficar cualquier variable numérica de una máquina en concreto que se desee visualizar, en un rango de fechas configurable.

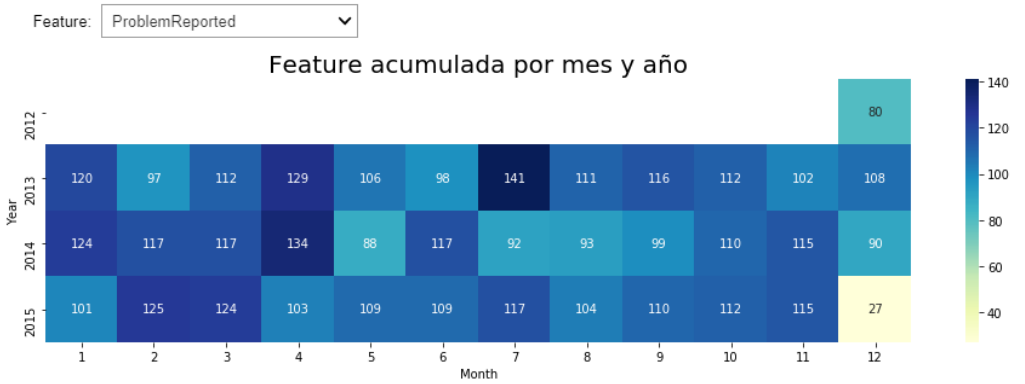


A continuación, se muestra un segundo display en el que poder enfrentar dos variables numéricas de un mismo dispositivo (una por cada eje), dentro de un rango cerrado de fechas, generando un gráfico de correlación, o scatter plot.

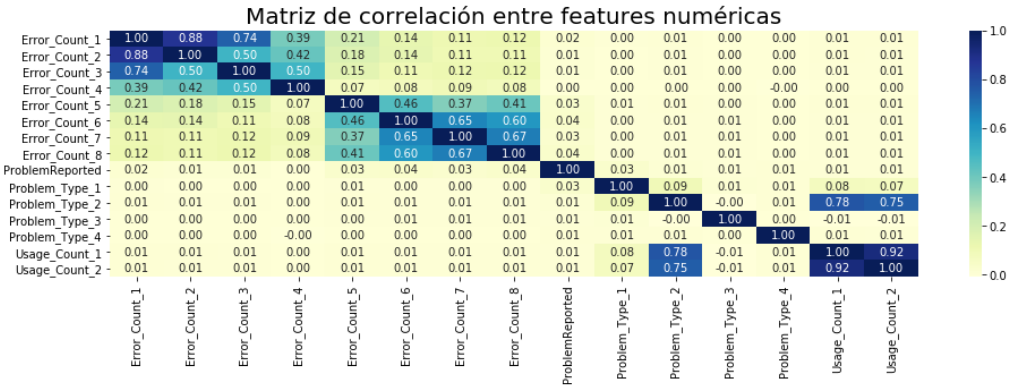


En el tercer display, se muestra un mapa de calor (heatmap) que agrupa el valor acumulado de la variable que se seleccione, por cada mes y año de su

histórico de valores. La tonalidad de cada celda representa una comparativa visual entre los valores acumulados (sumatorio) en cada periodo dentro del global.



Finalmente, se presenta un último análisis (estático, pero muy visual) en el que el objetivo que se persigue no es otro que poder identificar rápidamente la intensidad de la correlación entre cada par de variables numéricas del conjunto de datos.



Como se puede observar, existe una altísima correlación entre el modo de uso 1 y el 2, debido muy probablemente a que se trata de modos disjuntos. Además, los problemas de tipo dos tienen una correlación decente con ambos modos de uso. Los tres primeros contadores de errores también poseen una alta correlación entre ellos, lo que implica que se trata de errores con un alto índice de replicación durante un mismo día. Todas estas conclusiones repercuten a la hora de seleccionar las variables más adecuadas para modelar, no siendo muy útil incluir variables correlacionadas.

6. Feature Engineering

En esta etapa del proyecto siempre resulta favorable emplear esfuerzos para dotar de mayor precisión a los modelos. Sin atender, por ahora, al posible problema de la maldición de la dimensionalidad (se plantearán alternativas para paliarlo), en este apartado se propone la creación de nuevas variables que repercuten en el tamaño del conjunto de datos.

La filosofía que se persigue, por una parte, genera un grupo de nuevas variables 'ratio' (flotantes), y, por otra parte, transforma el grupo de variables numéricas en categóricas. De este modo, en la fase de modelado, las posibilidades de obtener un modelo ajustado a los datos, ya sea mediante regresión o clasificación, aumentan.

En primer lugar, se crea una variable que contenga la suma total de variables de alerta (warning) por cada uno de los dos subgrupos existentes. En caso de que la gestión del conjunto de datos sea pesada y hubiera que plantear la rescisión de las 172 variables originales de alerta, estas nuevas variables podrían aportar la suficiente información, sin renunciar a mantener un nivel aceptable de precisión en los algoritmos.

Respecto a las cuatro variables categóricas, el siguiente heatmap demuestra que sus valores son constantes dentro de una misma máquina.

Una vez verificado este comportamiento, se analizan los números de Tukey de estas variables y se propone una categorización de estas variables en no más de 4 valores diferentes, generando grupos de valores con la misma frecuencia de muestras, sin importar la amplitud de los grupos. La nomenclatura usada replica la terminología propia de la industria para establecer prioridades sobre alarmas de proceso: low low ('LL') para valores muy bajos, low ('L') para

valores bajos, high (‘H’) para valores altos y high high (‘HH’) para valores muy altos.

Al respecto de las ocho variables contadoras de errores, se observa que el dominio de sus valores, prácticamente, se reduce a los valores cero y uno. Muy eventualmente, aparece el valor dos solamente en la quinta variable del grupo. Por tanto, se plantea la idea de obviar esta cantidad mínima de registros, reemplazando el valor dos por el valor uno. Esta decisión favorecerá una hipotética categorización de nuevas variables generadas a partir de éstas.

Para las cuatro variables de tipos de código de fallo, cuyos valores son de tipo texto, se plantea una categorización binaria: cero para los valores ‘Unknown’, y 1 para cualquier código de error.

Para comenzar a ponderar correctamente los datos de los que se dispone, se plantea la creación de nuevas variables ‘ratios’ que relativicen la frecuencia de ocurrencia de ciertos eventos (problemas, códigos de fallo y agregado de alertas) en función del uso dado a la máquina en cuestión. Las variables que hacen de contadores de uso formarán parte del denominador de estas variables. Una vez creadas, además, se generan variables categóricas sobre ellas, nunca superando un máximo de tres categorías. Dadas las distribuciones de valores obtenida, las categorías separan valores nulos, valores mayores a cero y valores mayores a uno.

Con estas tres categorías, se generan tres nuevas variables por cada ‘ratio’; es decir, una variable por cada categoría particular indicando mediante cero o uno si el valor de la variable contiene la categoría en cuestión para un registro dado. Esta operación se conoce como ‘One-Hot Encoder’).

Como paso final sobre estas variables ‘ratio’, se realiza una normalización Min-Max que las escale al rango [0, 1], habilitando la posibilidad de introducir dichos datos en un modelo de aprendizaje automático en el que sea indispensable otorgarle la misma importancia a cada variable.

Volviendo sobre las variables de alerta (‘warning’), se plantea la necesidad de reducir su amplio número, con el objetivo de aligerar la carga de los modelos y evitar la ‘maldición de la dimensionalidad’. En primera instancia, se implementó el algoritmo de ‘Principal Component Analysis’, con la limitación de que sólo puede explicar la varianza del conjunto de datos linealmente. En cambio, un ‘AutoEncoder’ (arquitectura particular de red neuronal artificial) puede ser configurado para que modele datos de forma no lineal, otorgando mayor

flexibilidad a la hora de establecer diferenciación entre muestras no separables linealmente. Como anteriormente, se generan ‘ratios’ sobre estas variables para los dos tipos de funcionamiento que tienen las máquinas.

Para este proyecto, se implementan cuatro ‘AutoEncoders’. Cada uno de ellos modelará los ‘ratios’ de las variables de alerta de cada tipo, usando cada vez uno de los dos contadores de uso como denominador. Expresado de otro modo, para el primer ‘AutoEncoder’, se toman todos los ‘ratios’ de alertas del primer grupo, calculados sobre el primer contador de uso. Así sucesivamente, para el cuarto y último ‘AutoEncoder’ se reciben todos los ‘ratios’ de alertas del segundo grupo, calculados sobre el segundo contador de uso.

Se han probado diferentes configuraciones de ‘AutoEncoder’, sabiendo que entre los cuatro se repartirán 172 variables. La configuración mejor encontrada reduce la dimensión del conjunto de datos de entrada de cada ‘AutoEncoder’ a 5 variables (capa central del ‘AutoEncoder’, tras el codificador). La función de activación usada es ‘relu’, el optimizador usado es ‘nadam’, la función de pérdida es ‘mse’ (Mean Square Error) y el número de épocas, 3. Con estos parámetros, la precisión alcanza niveles entre el 80% y el 85%.

```
Epoch 1/3
2085102/2085102 [==============================] - 50s 24us/step - loss: 5.7278e-06 - acc: 0.8291
Epoch 2/3
2085102/2085102 [==============================] - 48s 23us/step - loss: 5.6689e-06 - acc: 0.8527
Epoch 3/3
2085102/2085102 [==============================] - 46s 22us/step - loss: 5.6540e-06 - acc: 0.8586
```

Traza de ejemplo con las estadísticas de los resultados obtenidos por uno de los AutoEncoders entrenados

Una vez introducidos todos los valores de las variables sobre los cuatro modelos para su predicción, las nuevas 20 variables a las que se ha reducido todo el conjunto de alertas (‘warning’) son incluidas al conjunto de datos global.

En este punto, es muy aconsejable guardar todo el progreso conseguido, con la intención de no volver emplear todo el tiempo que supone su ejecución y, además, evitar mensajes de error de memoria del compilador en tiempo de ejecución.

La segunda etapa de este apartado de ‘Feature Engineering’ consiste en generar una serie de variables con medidas estadísticas móviles (‘rolling’). La gran dificultad que se afronta en este punto es la compleja gestión de memoria del equipo en el que se trabaje, ya que ésta limita el tamaño máximo posible de un conjunto de datos, obligando a adaptar la implementación de código para trabajar con diferentes conjuntos de datos de manera secuencial, guardando y

cargando iterativamente el proceso alcanzado. Para este cometido, resulta de gran utilidad el uso de ficheros con extensión ‘pkl’ (pickle), que son ficheros binarios cuyos tiempos de guardado y carga en memoria están optimizados.

Una vez seleccionadas las variables, numéricas, a las que se puede aplicar este procesamiento, lo más importante es definir diferentes valores de ventana de tiempo sobre los que se realizarán los cálculos estadísticos que se fijen. La ventana de tiempo funciona agrupando un número fijo de registros contiguos (los primeros del conjunto de datos), y obteniendo medidas agregadas sobre los valores de sus variables. Se trata de un proceso iterativo en el que la ventana va avanzando por los registros, calculando las medidas estadísticas oportunas.

En este proyecto, se ha decidido obtener las siguientes medidas estadísticas: media, desviación estándar, máximo y mínimo. Las variables implicadas en este proceso son todas las relativas a errores, problemas, fallos, alertas y las propias generadas por los ‘AutoEncoders’. Las ventanas de tiempo definidas son de 3, 7, 15, 30 y 90 días. Dado que se dispone de un registro por máquina al día, el número de registros que tendrá cada ventana coincide con su número de días definido. Por otro lado, resulta crítico tener en cuenta que no podemos aplicar las ventanas de manera indiscriminada sobre todo el conjunto de datos, ya que cada una de las 1900 máquinas son independientes.

Tras estas operaciones realizadas, se dispone de 5 conjuntos de datos con 351 variables cada uno, cuyos tamaños están en torno a los 3 GB por fichero.

7. Detección de anomalías

Como ya se ha indicado anteriormente, las anomalías quedan definidas por la variable 'ProblemReported'. Sin embargo, en este apartado el alcance es el diseño de un modelo de aprendizaje automático, que sea entrenado con datos no anómalos y estableciendo el mejor umbral de separación entre falsos positivos y falsos negativos que optimice el valor económico desde el punto de vista de negocio, sin disponer de la variable objetivo. La variable objetivo sí será empleada para evaluar el ajuste del modelo propuesto.

Desde el punto de vista de negocio, se considera más crítico un falso positivo que un falso negativo. Los motivos son dos, principalmente:

- Siendo conscientes de las limitaciones del aprendizaje no supervisado, en el que gran parte del modelado se construye en base a límites estadísticos, un falso negativo es probable que se trate de un positivo (anomalía) de menor gravedad, ya que, según la métrica usada, no es muy diferente de los registros no anómalos.

- En cambio, un falso positivo, le resta credibilidad al modelo. Es preferible disponer de un modelo que alerte pocas veces, pero bien; en lugar de muchas veces, e incorrectamente. Las empresas, actualmente, están inmersas en la transformación digital y la implantación de sistemas inteligentes, en la que las expectativas están situadas a niveles altos. Por tanto, resulta fundamental no perder credibilidad frente a la Dirección, que financia este tipo de proyectos y espera un valor añadido.

Se afrontará la resolución de este problema haciendo uso del conjunto de datos obtenido tras la primera etapa de 'Feature Engineering' (antes de proceder con las operaciones de 'rolling'), junto con las variables generadas tras aplicar las operaciones de 'rolling' con la ventana de 3 días. Queda abierta la mejora del enfoque que se plantea, por ejemplo, añadiendo las variables obtenidas con los restantes tamaños de ventanas.

Como primera tarea, se plantea un algoritmo que etiquete cada registro del conjunto de datos como anómalo o no. Para ello, se crean dos nuevas variables calculadas a partir de la variable objetivo ('ProblemReported').

La primera de ellas será ‘Problem’, y tendrá un dominio binario de valores. En primera instancia, todos los registros con la variable objetivo igual a 1, pondrán su variable ‘Problem’ igual a 1. Además, los dos días anteriores a cada valor 1 en la variable objetivo, también pondrán su variable ‘Problem’ igual a 1. El resto, será 0. Con este procedimiento, se trata de identificar como anomalía cualquier registro que esté a menos de 3 días de reportar la anomalía. Este enfoque encaja con las estrategias seleccionadas entre las que se describen en [5].

La segunda variable por crear será ‘DaysTo’. Esta variable tendrá como dominio el conjunto de los números naturales. Su valor en cada registro indicará el número de días restantes hasta el próximo problema reportado (variable objetivo ‘ProblemReported’ igual a 1).

Con estas dos nuevas variables obtenidas, se acomete la creación de tres subconjuntos de datos disjuntos, con parte del total de registros contenidos en el conjunto de datos. Se pretende crear un subconjunto de datos sobre el que se pueda depositar la confianza en que no contiene registros anómalos, y con el que se pueda entrenar el modelo. Otro subconjunto se usará como test, y contendrá una pequeña muestra de registros no anómalos. Un tercer subconjunto, también de prueba, será exactamente el de aquellos registros cuya variable objetivo es igual a 1 (anómalos).

Una vez definidos los subconjuntos de datos que se usarán para modelar, se realiza un descarte de aquellas variables cuya desviación estándar de sus valores no supere el valor de 0,01. Con esto, se evita dotar al modelo de variables innecesarias cuya escasa o nula variabilidad no aportan información.

Siguiendo la metodología descrita en [13], donde se especifica el uso de, nuevamente, un ‘AutoEncoder’ para detectar anomalías; seleccionamos aquellas variables de tipo flotante del conjunto de datos que estén escaladas entre 0 y 1, ya que éstas son las que pueden ser empleadas.

Se plantea una arquitectura de ‘AutoEncoder’ con tres capas ocultas, de dimensiones 10, 2 y 10. La función de activación y pérdida, así como el optimizador, se mantienen con respecto a los cuatro ‘AutoEncoders’ anteriores. La novedad está en la utilización del inicializador de kernel ‘glorot_uniform’, y el regularizador L2. Con un número de 30 épocas, se obtiene una pérdida muy reducida en el conjunto de validación utilizado.

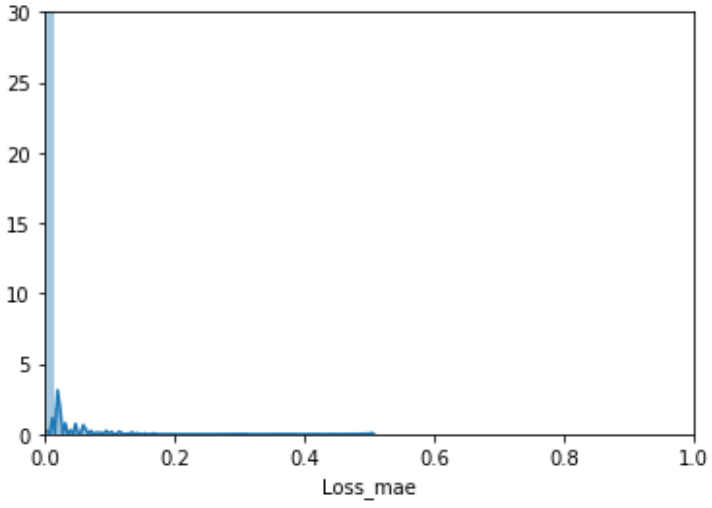
```
Train on 100097 samples, validate on 5269 samples
Epoch 1/30
100097/100097 [==============================] - 10s 100us/step - loss: 0.0017 - val_loss: 4.3012e-04
Epoch 2/30
100097/100097 [==============================] - 9s 90us/step - loss: 0.0011 - val_loss: 3.7041e-04
Epoch 3/30
100097/100097 [==============================] - 9s 90us/step - loss: 9.5808e-04 - val_loss: 2.4887e-04
Epoch 4/30
100097/100097 [==============================] - 9s 91us/step - loss: 8.3517e-04 - val_loss: 1.9046e-04
Epoch 5/30
100097/100097 [==============================] - 9s 91us/step - loss: 7.9249e-04 - val_loss: 1.7794e-04
…
Epoch 26/30
100097/100097 [==============================] - 10s 97us/step - loss: 7.7837e-04 - val_loss: 1.7475e-04
Epoch 27/30
100097/100097 [==============================] - 10s 96us/step - loss: 7.6221e-04 - val_loss: 1.7888e-04
Epoch 28/30
100097/100097 [==============================] - 10s 98us/step - loss: 7.3600e-04 - val_loss: 1.8400e-04
Epoch 29/30
100097/100097 [==============================] - 10s 99us/step - loss: 7.2468e-04 - val_loss: 2.3153e-04
Epoch 30/30
100097/100097 [==============================] - 10s 102us/step - loss: 7.2074e-04 - val_loss: 2.2726e-04
```

Evolución de la función de pérdida sobre el conjunto de entrenamiento y de evaluación

La clave de este problema está en calcular un valor de pérdida para cada registro que se pase al 'AutoEncoder' ya entrenado, y establecer un umbral para decidir si se trata de un registro anómalo o no. Este valor de pérdida se establece calculando el error absoluto medio (MAE, Mean Absolute Error) entre el resultado predicho por el modelo (salida del 'AutoEncoder') y el valor del registro (entrada al 'AutoEncoder').

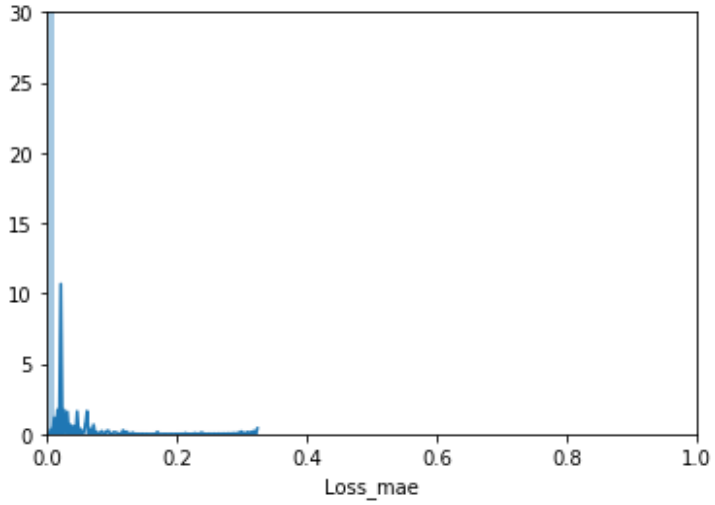
Como se puede observar, se pretende obtener una medida fidedigna de la diferencia entre el registro real y el registro replicado por el 'AutoEncoder'. A mayor diferencia en la reconstrucción de un registro, mayor probabilidad de que no sea similar a los que se usaron para entrenar el modelo. A raíz de estos resultados, se pueden extraer ciertas asunciones.

El gráfico siguiente muestra la distribución de valores de todos los errores calculados al nutrir el modelo con el propio conjunto de entrenamiento:

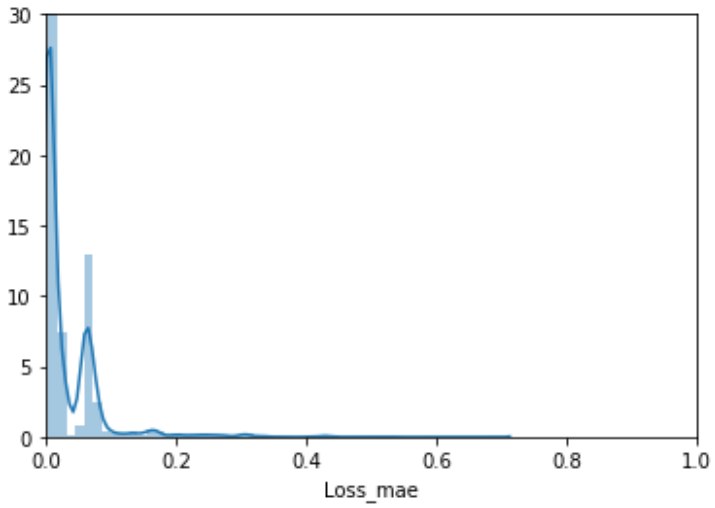


A continuación, se mostrarán un par de gráficos más, por lo que se utilizarán las mismas escalas en los ejes para poder comparar de manera visual.

El siguiente gráfico muestra la distribución obtenida mediante el conjunto de pruebas que contiene registros no anómalos (como los registros del conjunto de entrenamiento):



Si se comparan ambas distribuciones, se aprecia una similitud significativa. En cambio, si ejecutamos el segundo conjunto de prueba en el modelo (aquel cuyos registros sí son anómalos), se obtiene la siguiente distribución de errores:



En esta ocasión, la distribución sí presenta diferencias significativas.

En línea con el objetivo plasmado desde negocio, el umbral para considerar si un registro es anómalo o no, se fija en el percentil 99 de la distribución de errores obtenida sobre el propio conjunto de entrenamiento (gráfico 1).

Traduciendo a números estos resultados gráficos, y considerando el umbral definido:

- Se obtiene un acierto del 99% catalogando como no anómalos los registros del conjunto de prueba sin anomalías.

- En el conjunto de pruebas con anomalías, el modelo es capaz de detectar el 27% de ellas. Prácticamente, casi 3 de cada 4 anomalías no serán reportadas.

- El indicador de Precisión (verdaderos positivos sobre el total de positivos catalogados por el modelo) es del 95%; esto es, de todos los registros catalogados por el modelo como anómalos, sólo 1 de cada 20 no se corresponde con una anomalía. Por consiguiente, los 19 restantes sí lo son.

Estos resultados son muy esperanzadores, puesto que cumplen con el objetivo definido de disponer de un modelo de detección de anomalías que minimice el número de falsos positivos.

8. Prognosis

Cuando se afronta un problema de clasificación mediante algoritmos de aprendizaje automático, en función de los datos disponibles se pueden acometer diferentes estrategias. En este bloque se lanzarán varias propuestas de predicción de las anomalías ya catalogadas en el conjunto de datos.

Como tarea previa al planteamiento de las diferentes propuestas, se realiza un balanceo del conjunto de datos, que establezca una proporción entre registros anómalos y no anómalos (variable objetivo 1 y 0) que no se descompense más allá de un 90-10; es decir, al menos un 10% de los registros del conjunto de datos deben ser anómalos, aunque esta proporción no sea realista. Puesto que en el conjunto de datos existe un porcentaje muy reducido de registros anómalos, se procede a ignorar una parte de los registros no anómalos. Este procedimiento es importante que se aplique, si se desea que los modelos entrenados aprendan a diferenciar entre los dos valores de la variable objetivo.

Se plantea un primer enfoque en el que se haga uso de aquellas variables del conjunto de datos que sólo sean categóricas o binarias. De este conjunto de datos, se obtienen los conjuntos de entrenamiento y prueba de forma estratificada, en una proporción de 75-25 (conjunto de prueba constituido por el 25% del total de registros).

A continuación, se hace uso de la librería TPOT (Tree-Based Pipeline Optimization Tool) para ejecutar una batería de modelos con ajuste de parámetros inclusive, y cuya función objetivo busca maximizar la precisión de los modelos mediante una heurística basada en algoritmos evolutivos. Para el caso en cuestión, se configura un pipeline con un tamaño de población igual a 10, un total de 5 generaciones y validación cruzada igual a 3.

Los resultados obtenidos alcanzan una ‘Accuracy’ similar en varias ejecuciones lanzadas, alrededor del 91%. Cualquiera de dichos modelos, con el ajuste adecuado de sus parámetros, es válido para predecir. Los modelos en cuestión son:

- Un apilado de red bayesiana multinomial, seguido de una regresión logística.

- Un apilado de ‘Gradient Boosting’, seguido de un clasificador de soporte vectorial lineal.
- Un ‘Gradient Boosting’.
- Un ‘Extreme Gradient Boosting’.

Un detalle más preciso del proceso de evaluación del conjunto de prueba sobre cualquiera de los modelos es el siguiente:

	precision	recall	f1-score	support
0.0	0.91	0.99	0.95	26424
1.0	0.66	0.14	0.23	2936
avg / total	0.89	0.91	0.88	29360

Cambiando de planteamiento, el segundo enfoque suprime la limitación de restringir el uso de las variables numéricas flotantes para modelar. Por tanto, se procede a incluir todas las variables obtenidas de las operaciones de ‘rolling’ efectuadas sobre cada una de las ventanas de tiempo, resultando en un conjunto de datos con 1095 variables distintas.

Aplicando de nuevo el ‘clasificador’ TPOT con los mismos parámetros de configuración, se obtiene un modelo de regresión logística cuyo ajuste es similar al de los cuatro modelos anteriormente obtenidos. Como se aprecia en el detalle, el modelo que hace uso de todas las variables flotantes no mejora la calidad de los anteriores:

	precision	recall	f1-score	support
0.0	0.91	0.99	0.95	26424
1.0	0.67	0.12	0.21	2936
avg / total	0.89	0.91	0.88	29360

Como último enfoque de este apartado de predicción, se plantea el uso de una arquitectura de red neuronal artificial muy válida para este tipo de problemas: el perceptrón multicapa (MLP, Multilayer perceptron).

Tomando el último conjunto de datos generado con todas las variables, se toman dos subconjuntos de variables: por un lado, el total; por otro, tan sólo las variables cuya desviación estándar de sus valores es superior a 0,01. Cabe destacar que el primer subconjunto de datos posee el triple de variables que el segundo.

Obviando los detalles sobre la arquitectura de red definida, así como el optimizador usado y los valores de sus parámetros, la conclusión que se obtiene es la siguiente: ambos modelos obtienen resultados similares entre ellos, así como con respecto a los modelos de clasificación anteriormente propuestos en este apartado de la memoria.

```
Train on 83676 samples, validate on 4404 samples
Epoch 1/15
83676/83676 [==============================] - 6s 66us/step - loss: 0.3397 - acc: 0.8877 - val_loss: 4.2958 - val_acc: 0.1020
Epoch 2/15
83676/83676 [==============================] - 5s 61us/step - loss: 0.3078 - acc: 0.9029 - val_loss: 1.6293 - val_acc: 0.8980
Epoch 3/15
83676/83676 [==============================] - 5s 64us/step - loss: 0.3061 - acc: 0.9039 - val_loss: 0.3436 - val_acc: 0.8983
Epoch 4/15
83676/83676 [==============================] - 5s 55us/step - loss: 0.3050 - acc: 0.9038 - val_loss: 1.9042 - val_acc: 0.1020
Epoch 5/15
83676/83676 [==============================] - 5s 55us/step - loss: 0.3043 - acc: 0.9041 - val_loss: 0.8694 - val_acc: 0.8980
Epoch 6/15
83676/83676 [==============================] - 5s 59us/step - loss: 0.3041 - acc: 0.9042 - val_loss: 1.2762 - val_acc: 0.8980
Epoch 7/15
83676/83676 [==============================] - 5s 56us/step - loss: 0.3027 - acc: 0.9043 - val_loss: 0.5109 - val_acc: 0.7498
Epoch 8/15
83676/83676 [==============================] - 5s 59us/step - loss: 0.3030 - acc: 0.9048 - val_loss: 1.6433 - val_acc: 0.8980
Epoch 9/15
83676/83676 [==============================] - 6s 66us/step - loss: 0.3026 - acc: 0.9044 - val_loss: 0.3280 - val_acc: 0.9024
Epoch 10/15
83676/83676 [==============================] - 5s 61us/step - loss: 0.3028 - acc: 0.9040 - val_loss: 0.4556 - val_acc: 0.8980
Epoch 11/15
83676/83676 [==============================] - 5s 62us/step - loss: 0.3017 - acc: 0.9047 - val_loss: 2.2947 - val_acc: 0.1020
Epoch 12/15
83676/83676 [==============================] - 5s 64us/step - loss: 0.3023 - acc: 0.9038 - val_loss: 10.3071 - val_acc: 0.1020
Epoch 13/15
83676/83676 [==============================] - 5s 63us/step - loss: 0.3019 - acc: 0.9049 - val_loss: 0.7949 - val_acc: 0.8980
Epoch 14/15
83676/83676 [==============================] - 5s 64us/step - loss: 0.3018 - acc: 0.9046 - val_loss: 0.8425 - val_acc: 0.8980
Epoch 15/15
83676/83676 [==============================] - 6s 71us/step - loss: 0.3020 - acc: 0.9042 - val_loss: 0.8367 - val_acc: 0.8980
```

Evolución entre épocas durante el entrenamiento del segundo subconjunto de datos generado

9. Valoración económica del trabajo

Atendiendo a los diferentes productos obtenidos, se propone una estimación de ahorro en costes de mantenimiento.

El primer producto es el análisis exploratorio de datos presentado. Este tipo de visualizaciones y cálculos sencillos no sólo resulta útil dentro del ciclo de vida de la ciencia de datos, sino que también aportan valor al negocio. El escenario tipo en una empresa del sector industrial enfocada a producir un bien (por ejemplo, manufacturera o energética) es aquel en el que el departamento de operación/producción está totalmente volcado en obtener la máxima producción posible. Por otro lado, el departamento de mantenimiento dedica el grueso de su esfuerzo y recursos en las tareas correctivas y preventivas aplicadas sobre la maquinaria.

Sin embargo, un simple producto como el análisis exploratorio de datos permite graficar tendencias, valores límite, picos, valores fuera de rango, etc. cuyo comportamiento se aleje de lo que se considera normal, cuando aún no se han alcanzado los valores de alarma/disparo en el sistema de control distribuido (DCS, Distributed Control System). Existen dos maneras sencillas de definir qué es normal en una máquina: por un lado, se pueden comparar los valores actuales de la máquina frente a los valores que tuvo en el pasado; por otro lado, suele ser frecuente que se disponga de varias máquinas idénticas (o muy similares) ubicadas en paralelo dentro del proceso industrial, por lo que es posible establecer comparativas entre los valores actuales de dichas máquinas.

Hasta ahora, sin que aún haya intervenido ningún modelo de aprendizaje automático, se está aportando un valor añadido cuyo cálculo es complejo y queda fuera del alcance de este trabajo.

En cuanto al modelo no supervisado de detección de anomalías, se dispone de una proporción de verdaderos positivos sobre el total de positivos (‘recall’) de un 27%; y una proporción de verdaderos positivos sobre el total de positivos catalogados por el modelo (‘precision’) del 95%. En otras palabras, el modelo entregado es capaz de detectar el 27% de los problemas en las máquinas, por lo que a lo sumo podría materializarse en un ahorro del 27% en costes de mantenimiento. Como este caso ideal nunca es posible debido a causas como la poca anticipación en la detección del fallo, o la imposibilidad de disponer de

ventanas de tiempo para reparar/reemplazar el equipo fuera de periodos de producción; se establece que sólo un tercio de las anomalías detectadas podrá ser solventada antes de quedar la máquina indisponible. Por tanto, con este umbral conservador tomado, queda un margen del 9% de ahorro en costes de mantenimiento. En cuanto al 5% de falsos positivos, no tienen asociado coste económico.

Además, no se están teniendo en cuenta los tiempos de indisponibilidades asociadas a paradas no programadas de las máquinas, lo cual repercute directamente en una bajada de los ingresos por producción.

Finalmente, en el modelo supervisado de clasificación, si bien el valor de 'recall' es tan sólo del 14%, la ventaja radica en que se dispone de una anticipación de hasta 72 horas hasta la indisponibilidad. Reutilizando el paralelismo empleado en los párrafos anteriores entre el valor de 'recall' y el ahorro en costes, este valor del 14% presenta menor incertidumbre.

Como conclusión, los productos obtenidos en este trabajo reportan un ahorro en costes de mantenimiento entre el 9% y el 14%. En esta estimación no se ha considerado el aumento de ingresos por reducción de las indisponibilidades y las paradas de producción no programadas de las máquinas que, según el caso, pueden llegar a ser incluso muy superiores al ahorro en costes de mantenimiento.

La conclusión, por consiguiente, es que el resultado obtenido repercute en una aportación evidente de valor que, desde el punto de vista de negocio, interesa implantar a la brevedad.

10. Conclusiones

Este capítulo tiene que incluir:

Una descripción de las conclusiones del trabajo: ¿Qué lecciones se han aprendido del trabajo?

Una reflexión crítica sobre el logro de los objetivos planteados inicialmente: ¿Hemos logrado todos los objetivos? Si la respuesta es negativa, ¿por qué motivo?

Un análisis crítico del seguimiento de la planificación y metodología a lo largo del producto: ¿Se ha seguido la planificación? ¿La metodología prevista ha sido la adecuada? ¿Ha habido que introducir cambios para garantizar el éxito del trabajo? ¿Por qué?

Las líneas de trabajo futuro que no se han podido explorar en este trabajo y han quedado pendientes.

11. Anexos

Toda la documentación generada durante la fase de diseño e implementación del proyecto: ficheros Notebook de código (formatos ipynb y html), ficheros de código Python con los modelos diseñados (formato py) y modelos entrenados (formato sav); se encuentra alojada en el siguiente repositorio de GitHub:

https://github.com/fernandodecastilla/Machine-Learning-for-Predictive-Maintenance

El fichero de datos fuente a partir del cual se han generado todos los resultados obtenidos en el proyecto esta accesible desde el siguiente enlace:

https://pysparksampledata.blob.core.windows.net/sampledata/sampledata.csv

12. Bibliografía

[1] Christoph M. Flath, Nikolai Stein. Towards a data science toolbox for industrial analytics applications, 2017

[2] https://towardsdatascience.com/why-data-scientists-will-turn-to-industrial-and-manufacturing-industries-in-the-near-future-6f690e02dfd3%20 (11 de marzo de 2019)

[3] https://enosix.com/2016/12/14/2017-the-year-integration-enables-the-internet-of-things-and-industrial-analytics/ (11 de marzo de 2019)

[4] https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data (11 de marzo de 2019)

[5] https://medium.com/bigdatarepublic/machine-learning-for-predictive-maintenance-where-to-start-5f3b7586acfb (15 de marzo de 2019)

[6] https://towardsdatascience.com/a-note-about-finding-anomalies-f9cedee38f0b (14 de marzo de 2019)

[7] https://towardsdatascience.com/a-brief-overview-of-outlier-detection-techniques-1e0b2c19e561 (13 de marzo de 2019)

[8] K. Shi, H. Song, H. Yu, Z. Zhu, "Collaborative Contextual Anomaly Detection for Industrial Equipment Groups", 2016 9th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, 2016, pp. 381-385.

[9] Chandola, V., Banerjee, A. & Kumar, V., “Anomaly Detection: A Survey”, ACM Computing Surveys, July. 41(3), 2009

[10] Goldstein, M. & Uchida, S., “A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data”, PLoS ONE, 11(4), p. 31, 2016

[11] Shen Yin, Xiangping Zhu, Chen Jing, “Fault detection based on a robust one class support vector machine”, Neurocomputing, Volume 145, pp. 263-268, 2014

[12] Ding, Jianwei, Yingbo Liu, Xiang Lin, Jianmin Wang, Yong-hong Liu, “An anomaly detection approach for multiple monitoring data series based on latent correlation probabilistic model”, Applied Intelligence 44, pp. 340-361, 2015

[13] https://towardsdatascience.com/how-to-use-machine-learning-for-anomaly-detection-and-condition-monitoring-6742f82900d7 (13 de marzo de 2019)

[14] https://towardsdatascience.com/an-introduction-on-time-series-forecasting-with-simple-neura-networks-lstm-f788390915b (19 de marzo de 2019)

[15] Akash Singh, "Anomaly Detection for Temporal Data using Long Short-Term Memory (LSTM)", KTH Royal Institute of technology school of information and communication technology, Stockholm, 2017

[16] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, Dong-Hua Zhou, “Remaining useful life estimation – A review on the statistical data driven approaches”, European Journal of Operational Research, Volume 213, Issue 1, pp. 1-14, 2011

[17] Z. Tian, “An artificial neural network method for remaining useful life prediction of equipment subject to condition monitoring”, Journal of Intelligent Manufacturing, 2009

[18] Abd Kadir Mahamad, Sharifah Saon, Takashi Hiyama, “Predicting remaining useful life of rotating machinery based artificial neural network”, Computers & Mathematics with Applications, Volume 60, Issue 4, pp. 1078-1087, 2010

[19] A. Hajdarevic, L. Banjanovic-Mehmedovic, I. Dzananovic, F.Mehmedovic, M. Ayaz Ahmad, "Recurent Neural Network as a Tool for Parameter Anomaly Detection in Thermal Power Plant", International Journal of Scientific & Engineering Research, Volume 6, Issue 8, 2015

[20] https://scikit-learn.org/ (3 de marzo de 2019)

[21] https://keras.io/ (3 de marzo de 2019)

[22] https://www.pyimagesearch.com/2019/01/07/auto-keras-and-automl-a-getting-started-guide/ (3 de marzo de 2019)

[23] https://github.com/EpistasisLab/tpot/ (3 de marzo de 2019)

[24] https://bigml.com/ (3 de marzo de 2019)