COIMBRA BUSINESS SCHOOL

iscac Politécnico de Coimbra

100 ANOS

Manutenção Preditiva em Máquinas Industriais:
Desenvolvimento Modelos de Aprendizagem Computacional em Python

Fernando Almeida

ISCAC | 2024

COIMBRA BUSINESS SCHOOL

iscac Politécnico de Coimbra

100 ANOS

Fernando Manuel Santos Almeida

Manutenção Preditiva em Máquinas Industriais: Desenvolvimento Modelos de Aprendizagem Computacional em Python

Coimbra, outubro de 2024

Fernando Manuel Santos Almeida

Manutenção Preditiva em Máquinas Industriais: Desenvolvimento Modelos de Aprendizagem Computacional em Python

Relatório de estágio submetido ao Instituto Superior de Contabilidade e Administração de Coimbra para cumprimento dos requisitos necessários à obtenção do grau de **Mestre em Análise de Dados e Sistemas de Apoio à Decisão** realizado sob a orientação da Professora Doutora Joana Jorge de Queiroz Leite e supervisão de Engenheiro Gonçalo Lourenço Martins.

Coimbra, outubro de 2024

TERMO DE RESPONSABILIDADE

Declaro ser o autor deste relatório de estágio, que constitui um trabalho original e inédito, que nunca foi submetido a outra Instituição de ensino superior para obtenção de um grau académico ou outra habilitação. Atesto ainda que todas as citações estão devidamente identificadas e que tenho consciência de que o plágio constitui uma grave falta de ética, que poderá resultar na anulação do presente relatório de estágio.

AGRADECIMENTOS

Quero agradecer a todas as pessoas que me apoiaram e ajudaram durante estes dois anos exigentes de mestrado, contribuindo assim para a conclusão de mais uma etapa.

Começo pela minha orientadora, a Professora Doutora Joana Jorge de Queiroz Leite, por todo o acompanhamento, dedicação, disponibilidade, ajuda, motivação e paciência ao longo deste último ano de desenvolvimento deste projeto, sem a qual não teria sido possível concluir.

Agradeço à Coimbra Business School | ISCAC que tem sido a minha segunda casa nestes últimos cincos anos, e aos excelentes docentes que, ao longo da licenciatura e mestrado, contribuíram para o meu sucesso académico.

À Professora Doutora Isabel Maria Mendes Pedrosa e ao Engenheiro Gonçalo Lourenço Martins, que tornaram possível a oportunidade de estágio.

À empresa SINMETRO, pela oportunidade e voto de confiança, e a todos os colegas pelo excelente acolhimento e integração na empresa, pelos momentos de convívio, companheirismo e amizade que fizeram com que seis meses passassem a voar. Uma palavra especial à colaboradora Viviana Domingues por toda a sua disponibilidade e ajuda no desenvolvimento do projeto, mesmo não estando alocada a ele.

A todos os meus amigos e amigas, agradeço por todos os momentos, bons ou maus, e desejo que as amizades continuem por muitos mais anos.

Por último, um agradecimento especial à minha família que me tem apoiado ao longo de toda a minha vida.

Muito obrigado a todos!

RESUMO

Na Indústria 4.0, a utilização de dados e tecnologia é essencial para otimizar a eficiência e a qualidade dos processos produtivos, fortalecendo a competitividade e a sustentabilidade das operações. A SINMETRO, através do ACCEPT, uma ferramenta poderosa na Indústria 4.0, integra módulos que oferecem análises estatísticas avançadas em tempo real, assegurando a eficácia da produção e a qualidade dos produtos. Este sistema, apesar de essencial no cálculo do indicador OEE (*Overall Equipment Effectiveness)*, não antecipa os fatores que afetam este indicador, o que dificulta a implementação de melhorias. A principal causa de perda no OEE é a disponibilidade, que é muitas vezes afetada por paragens não planeadas devido a avarias, evidenciando a necessidade de uma solução de manutenção eficiente.

A manutenção preditiva é uma estratégia que utiliza dados de sensores e técnicas de análise para prever falhas em equipamentos antes que ocorram, permitindo intervenções de manutenção assertivas, reduzindo os custos para aumentar a competitividade das empresas. A antecipação deste tipo de falhas pode ser feita através da previsão do RUL (*Remaining Useful Life*), um indicador fundamental na manutenção preditiva. A aplicação desta estratégia de manutenção permite aumentar a fiabilidade dos sistemas industriais ou mecânicos e segurança e, ao mesmo tempo, reduzir o custo de manutenção.

Assim, foi desenvolvida uma solução de manutenção preditiva baseada no RUL, capaz de antecipar falhas ou paragens dos equipamentos, aplicável a diferentes máquinas e empresas, tendo sido aplicados dois algoritmos, Random Forest e XGBoost, com o XGBoost a apresentar os melhores resultados. Reduzindo a variabilidade do RUL, foi possível obter melhorias nos resultados tendo em conta a sua aplicação em dados reais. Foi ainda possível recolher um conjunto de sugestões para aumentar a qualidade dos dados. Verificou-se que, sem dados de sensores num modelo de manutenção preditiva, esse modelo dificilmente conseguirá prever o RUL de forma eficaz.

Palavras-chave: Manutenção preditiva, RUL, Aprendizagem computacional, Random Forest, XGBoost

ABSTRACT

In Industry 4.0, the use of data and technology is essential to optimize the efficiency and quality of production processes, strengthening the competitiveness and sustainability of operations. SINMETRO, through ACCEPT, a powerful tool in Industry 4.0, integrates modules that offer advanced real-time statistical analysis, ensuring production effectiveness and product quality. This system, although essential for calculating the OEE (Overall Equipment Effectiveness) indicator, does not anticipate the factors that affect this indicator, making it challenging to implement improvements. The primary cause of loss in OEE is availability, which is often impacted by unplanned downtime due to breakdowns, highlighting the need for an efficient maintenance solution.

Predictive maintenance is a strategy that uses sensor data and analytical techniques to predict equipment failures before they occur, allowing assertive maintenance interventions and reducing costs to enhance companies' competitiveness. Anticipating such failures can be achieved by predicting the RUL (Remaining Useful Life), a fundamental indicator in predictive maintenance. The application of this maintenance strategy improves the reliability and safety of industrial or mechanical systems while simultaneously reducing maintenance costs.

Thus, a predictive maintenance solution based on RUL was developed, capable of anticipating equipment failures or downtimes and applicable to different machines and companies. Two algorithms were applied, Random Forest and XGBoost, with XGBoost delivering the best results. By reducing RUL variability, it was possible to achieve improved results considering its application to real data. Additionally, a set of suggestions was gathered to enhance data quality. It was found that without sensor data in a predictive maintenance model, it is challenging for a model to effectively predict RUL.

Keywords: Predictive Maintenance, RUL, Machine Learning, Random Forest, XGBoost

ÍNDICE GERAL

ÍNDICE DE TABELAS E FIGURAS

ÍNDICE DE TABELAS

Lista de abreviaturas, acrónimos e siglas

SINMETRO…….Sistemas de Inovação em Qualidade e Metrologia, Lda.

IoT……...*Internet of Things*

OEE .....….*Overall Equipment Effectiveness*

ML .....….*Machine Learning*

CRISP-DM ....….*CRoss-Industry Standard Process for Data Mining*

DM.....….*Data Mining*

RUL .....….*Remaining Useful Life*

PdM .....….*Predictive Maintenance*

PHM .....*Prognostics and Health Management*

RF……..*Random Forest*

RSME……*Root Mean Square Error*

MAE…….*Mean Absolute Error*

MSE…….*Mean Squared Error*

R2…….*Coefficient of Determination*

MAPE……..*Mean Absolute Percentage Error*

KDD…….*Knowledge Discovery in Databases*

I&D……..Inovação e Desenvolvimento

SQL……..*Structured Query Language*

1 INTRODUÇÃO

O presente relatório é referente ao projeto desenvolvido em ambiente empresarial, na empresa SINMETRO - Sistemas de Inovação em Qualidade e Metrologia, Lda., na modalidade de estágio curricular, no âmbito da componente não letiva do Mestrado em Análise de Dados e Sistemas de Apoio à Decisão. O estágio teve a duração de 6 meses, entre 02 de outubro de 2023 e 25 de março de 2024, num total de 960 horas. Decorreu em regime presencial nas instalações de Coimbra, no Iparque.

Neste capítulo, são apresentados o contexto, a motivação e objetivos do estágio, a metodologia adotada e, por último, a estrutura do relatório.

1.1 Contexto

Com o avanço da transformação digital na Indústria 4.0, torna-se essencial adotar uma abordagem orientada por dados e tecnologia para identificar ineficiências, reduzir desperdícios e promover continuamente melhorias na qualidade e eficiência dos processos de produção. Esta revolução industrial representa uma mudança profunda nos métodos tradicionais de produção, onde a integração de tecnologias como a Internet das Coisas (IoT, do inglês *Internet of Things*), a Inteligência Artificial e a automação desempenham papéis fundamentais.

A Indústria 4.0 não se limita apenas à adoção de máquinas e tecnologias avançadas, mas sim ao aproveitamento do valor dos dados gerados por estas tecnologias para impulsionar melhorias significativas na qualidade, eficiência e flexibilidade, resultando em operações industriais mais ágeis, competitivas, sustentáveis e resultados para os negócios e organizações.

De acordo com um estudo de mercado realizado pela Forrester[1] de 2016, entre 60% e 73% dos dados recolhidos em contexto empresarial não são utilizados, permanecendo inativos

[1] Hadoop Is Data's Darling For A Reason; https://www.forrester.com/blogs/hadoop-is-datas-darling-for-a-reason/

até serem eliminados para dar espaço a novos dados, perpetuando assim um ciclo de ineficiência na valorização dos dados.

A obtenção de informação em tempo real é vital para tomar decisões mais fundamentadas e acertadas, permitindo uma melhoria contínua tanto na qualidade dos produtos como nos processos de produção. Através de sensores e dispositivos conectados no chão de fábrica, as empresas industriais recolhem dados precisos e detalhados que permitem análises imediatas. Esta capacidade de monitorizar e analisar dados em tempo real não só aumenta a eficiência operacional, como permite a deteção antecipada de anomalias ou problemas de qualidade, possibilitando intervenções rápidas e precisas, antes que a produção seja afetada. Esta agilidade torna-as mais flexíveis e preparadas para responder às mudanças do mercado, resultando em operações mais precisas, ágeis e competitivas, caraterísticas essenciais na Indústria 4.0.

1.2 Motivação e objetivos

A SINMETRO, fundada em 2002, é uma empresa portuguesa de desenvolvimento de *software*, consultoria e serviços especializados nas áreas de qualidade, metrologia, e inovação tecnológica, dedicada ao desenvolvimento e implementação de sistemas de gestão e soluções tecnológicas para a melhoria contínua de processos e produtos nas organizações. O seu principal produto, o ACCEPT, é um *software* para Controlo de Qualidade Industrial, uma ferramenta poderosa na indústria 4.0, na análise estatística das variáveis críticas de produtos e processos.

O sistema ACCEPT integra diversos módulos que proporcionam uma visão abrangente do desempenho dos parâmetros críticos em produtos e processos, através de análises estatísticas avançadas em tempo real, assegurando a eficácia dos processos de produção, o cumprimento das especificações e a qualidade dos produtos. No contexto do chão de fábrica, o sistema ACCEPT destaca-se ao permitir que os profissionais responsáveis pela produção monitorizem em tempo real as variáveis críticas do processo. Esta monitorização contínua facilita a identificação imediata de quaisquer desvios, possibilitando intervenções rápidas e precisas para manter a qualidade e eficiência dos processos produtivos, o que é

fundamental para um ambiente de produção que exige respostas ágeis e tomadas de decisão eficazes.

Uma das principais características do ACCEPT é o cálculo preciso e em tempo real do indicador de Eficiência Global do Equipamento (OEE, do inglês *Overall Equipment Effectiveness*). Neste indicador, uma das principais causas de perda é a falta de disponibilidade do equipamento, que pode ocorrer por causas programadas, como troca do produto ou não programadas, como avaria do equipamento. No entanto, no contexto atual da Indústria 4.0, o ACCEPT está limitado no sentido em que não permite análises preditivas relativas à disponibilidade do equipamento, com base em fatores recolhidos pelo sistema. Concretamente, é pertinente que o ACCEPT integre uma solução de manutenção preditiva para prever falhas ou paragens não programadas dos equipamentos.

Neste âmbito, através de uma parceria entre a SINMETRO e a Coimbra Business School | ISCAC, foi proposto um projeto com o objetivo geral de desenvolver um modelo preditivo de aprendizagem computacional (ML, do inglês *machine learning*) capaz de prever paragens não programadas de máquinas industriais que possa, em fase posterior, ser integrado no ACCEPT. Uma vez que o sistema ACCEPT é transversal a máquinas e clientes e tem já definidos os parâmetros industriais monitorizados e a forma como são recolhidos, tal impôs que o modelo tivesse alguma generalidade para ser aplicável a vários tipos de máquinas de diferentes empresas. A linguagem de programação Python a usar foi também acordada para proporcionar uma boa integração do modelo no ACCEPT. É importante notar que a implementação no ACCEPT não foi definida como um objetivo do projeto, já que esta foi uma primeira abordagem a este problema por parte da SINMETRO.

1.3 Metodologia

A metodologia para o planeamento e desenvolvimento do projeto foi a metodologia CRISP-DM (*CRoss-Industry Standard Process for Data Mining*), que se serviu também de orientação à estruturação do presente relatório no que diz respeito à sua descrição.

A metodologia CRISP-DM é frequentemente utilizada para planear, documentar, comunicar e gerir projetos de Mineração de Dados (DM, do inglês *Data Mining*),

independentemente do setor industrial e da tecnologia utilizada. A versatilidade desta metodologia possibilita que seja aplicada tanto por indivíduos com experiência na implementação de projeto de DM, como por indivíduos sem experiência e com pouco tempo para experimentar outras abordagens. Esta metodologia apresenta várias vantagens, como a possibilidade de realizar mais projetos de DM com menor custo, maior fiabilidade, maior repetibilidade, gestão mais simples e execução mais rápida (Wirth & Hipp, 2000).

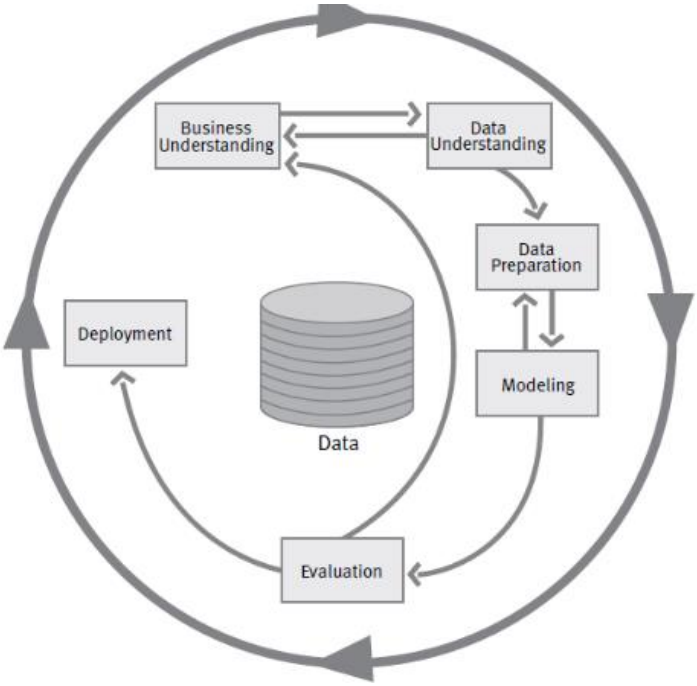


Figura 1.1 – Fases do modelo CRISP-DM

Fonte: Chapman et al. (2000)

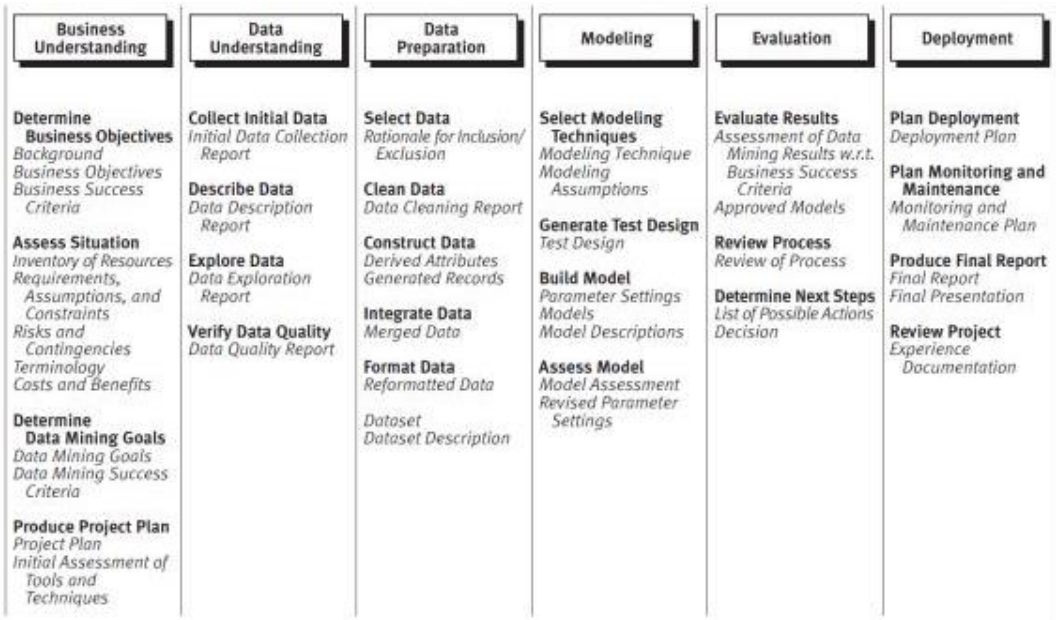


Figura 1.2 – Tarefas de cada fase do modelo CRISP-DM

Fonte: Chapman et al. (2000)

A metodologia CRISP-DM fornece uma visão geral do ciclo de vida de um projeto organizando-o em seis fases, como esquematizado na Figura 1.2, descrevendo, detalhadamente, cada uma delas no que diz respeito às tarefas e resultados, conforme sistematizado na Figura 1.2. Segue-se uma descrição sumária das seis fases desta metodologia.

1. Compreensão do negócio: Nesta primeira fase, define-se o problema, as metas e os objetivos no contexto do negócio, convertendo depois esta informação num problema de DM e num plano para atingir os objetivos definidos;
2. Entendimento dos dados: Nesta fase, recolhe-se e explora-se os dados inicialmente disponíveis, identificando problemas de qualidade, como dados em falta ou inconsistências e ganhando um entendimento inicial sobre a sua estrutura e relevância;
3. Preparação de dados: Esta fase cobre todas as tarefas para a construção do conjunto de dados que irá ser utilizado na modelação. Vai desde a limpeza, construção de

novas variáveis, transformação e integração dos dados para os colocar num formato adequado para a modelação, garantindo a qualidade e a consistência necessárias;

4. Modelação: Nesta fase, selecionam-se algoritmos e técnicas apropriadas, construindo e testando modelos, ajustando os parâmetros para valores ótimos. Determinadas técnicas requerem formatos de dados específicos, sendo muitas vezes necessário voltar à fase anterior;
5. Avaliação: Nesta fase do projeto, avalia-se a qualidade e o desempenho dos modelos otimizados, selecionando-se o melhor e validando se atende aos objetivos do negócio, antes de se proceder à sua implementação;
6. Implementação: Procede-se à implementação do modelo num ambiente de produção. Geralmente, esta não é a última fase do projeto.

Conclui-se que a estrutura CRISP-DM é uma abordagem flexível e adaptável, capaz de atender às necessidades específicas de diferentes projetos de DM. Esse processo colaborativo reúne indivíduos de diversas áreas e níveis de experiência, proporcionando uma metodologia estruturada e abrangente. Assim, o CRISP-DM auxilia as organizações a transformar dados em informações valiosas e a aplicar ações de forma eficaz e eficiente (Wirth & Hipp, 2000).

1.4 Estrutura

O presente relatório está estruturado em cinco capítulos.

O primeiro capítulo que é o atual, é onde se apresenta o contexto, a motivação e objetivos, a metodologia a ser aplicada e por último a estrutura seguida pelo relatório.

No segundo capítulo, do enquadramento teórico, é feita uma revisão de literatura que permite compreender o que é a manutenção industrial, os tipos de manutenção existente, a aplicação de aprendizagem computacional para auxiliar nas ações de manutenção.

No terceiro capítulo, é feita uma apresentação da entidade acolhedora, a sua missão, serviços disponíveis e o seu principal produto, o ACCEPT.

Estes primeiros capítulos consideram-se como fazendo parte da fase do entendimento do negócio, segundo a metodologia CRISP-DM.

No quarto capítulo, do desenvolvimento do projeto, encontram-se as restantes fases desta metodologia concretizadas neste projeto, o entendimento dos dados, preparação dos dados, modelação e avaliação. Todo o código desenvolvido para cada uma das fases anteriores encontra-se disponível na pasta partilhada:

https://drive.google.com/drive/folders/1_CySqulWRom622tJYMyv1FDrcuPLEHkA?usp=sharing

Por fim, o quinto capítulo, da conclusão, expõe um sumário do trabalho desenvolvido e principais resultados, as limitações e dificuldades que foram encontradas e ultrapassadas, o cumprimento dos objetivos, sugestões de trabalhos futuros, terminando com algumas considerações finais.

2 Enquadramento Teórico

A indústria de manufatura tem evoluído consideravelmente no século XXI com o aparecimento de tecnologias avançadas nas áreas da robótica, produção aditiva e máquinas. Aliadas à conetividade digital, cibersegurança e análise de dados, as empresas têm aplicado processos mais complexos e técnicas no chão de fábrica, aumentando a sua competitividade no mercado (Thomas & Weiss, 2021).

Segundo um estudo da Technavio[2], prevê-se que o mercado da Indústria 4.0 tenha um crescimento de 357,6 mil milhões de dólares, entre 2023 e 2028, com uma taxa de crescimento anual de cerca de 28,53%.

Assim, Dogan e Birant (2021) recomendam que as empresas utilizem diferentes técnicas de ML e DM e ferramentas que permitam alcançar os objetivos definidos e ultrapassar os problemas de fabrico que são diversos e abrangem vários aspetos do processo produtivo. Destacam que a análise de dados tem um papel crucial na deteção de falhas, onde a monitorização constante de parâmetros operacionais permite identificar anomalias e antecipar falhas, evitando paragens inesperadas. A recolha e tratamento de dados são também importantes no controlo da qualidade, assegurando que os produtos cumprem os padrões estabelecidos. A previsão de ciclos é outro exemplo, onde os dados podem ser vantajosos, ajudando a estimar com maior precisão o tempo necessário para completar processos produtivos. Além disso, a análise de dados permite a otimização do planeamento da produção e a previsão do consumo energético, permitindo às empresas reduzir custos operacionais. Desta forma, a utilização eficaz dos dados melhora significativamente a eficiência, reduzindo desperdícios e aumentando a produtividade.

Neste capítulo, são apresentados os três tipos de manutenção existente, destacando a importância da manutenção preditiva, os seus desafios, a relevância do RUL (do inglês, *Remaining Useful Life)* neste tipo de manutenção e as técnicas para a sua previsão, bem

[2] Industry 4.0 Market Analysis North America, Europe, APAC, Middle East and Africa, South America - US, China, Germany, Japan, UK - Size and Forecast 2024-2028; https://www.technavio.com/report/industry-4-0-market-analysis

como os benefícios dessa abordagem. É feita uma comparação entre os diferentes tipos de manutenção e os custos associados à manutenção industrial. Aborda-se a importância da aplicação de aprendizagem computacional para auxiliar nas ações de manutenção, descrevendo os algoritmos aplicados na parte prática do projeto, os conjuntos de dados normalmente utilizados, a importância do processamento de dados, as métricas para avaliar os modelos e as vantagens da aplicação de aprendizagem computacional. Por fim, são apresentadas as metodologias utilizadas em projetos de aprendizagem computacional.

2.1 Manutenção Industrial

A manutenção tem um papel crucial na indústria, dado que os seus custos podem representar uma percentagem significativa dos custos de produção, o que afeta a capacidade de uma empresa de ser competitiva com preços baixos, elevada qualidade e desempenho. Paragens não programadas podem afetar a produção de uma empresa, levar à perda de reputação e a perdas económicas. Como Zhu et al. (2024) referem, em 2013, a Amazon esteve inativa durante 49 minutos, que resultou na perda de 4 milhões de dólares e salientam que estratégias de manutenção eficazes são capazes de prevenir paragens inesperadas de produção, reduzir custos e prolongar o tempo de vida útil das máquinas industriais.

A manutenção industrial consiste num conjunto de serviços prestados por técnicos ou mecânicos que visam gerir máquinas e equipamentos com o objetivo de maximizar o tempo de funcionamento. Como qualquer outro aspeto da indústria, a manutenção industrial encontra-se em contínuo desenvolvimento e aperfeiçoamento. Com o aumento do conhecimento acerca das melhores práticas neste domínio, a tecnologia também evoluiu, contribuindo para a criação de métodos mais eficazes de manutenção e gestão de equipamentos. O estudo Nunes et al. (2023) mostra que, por estas razões, as abordagens de manutenção têm sofrido transformações significativas, fruto da preocupação e dos esforços contínuos de investigadores, engenheiros, técnicos e especialistas.

As empresas vivem num ambiente extremamente competitivo, e a utilização de métodos de manutenção eficientes pode ser a diferença entre o lucro e o prejuízo. A manutenção

afeta a qualidade dos produtos, bem como os custos de capital, mão de obra e inventário. Compreender e investir em métodos de manutenção avançada pode trazer uma vantagem competitiva às empresas de manufatura (Thomas, 2018).

A Figura 2.1, mostra os três principais tipos de manutenção industrial existentes. Seguindo por ordem cronológica temos a manutenção reativa, a manutenção preventiva e, por último, a manutenção preditiva (PdM, do inglês *Predictive Maintenance*).

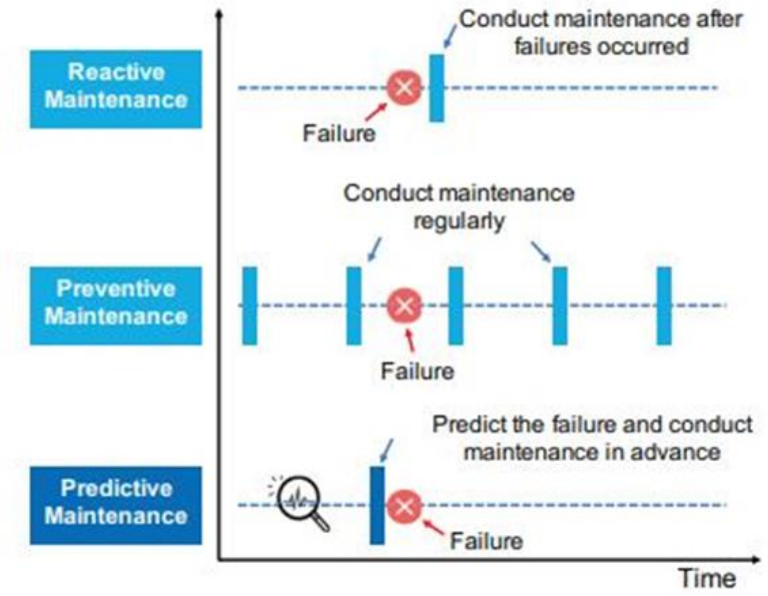


Figura 2.1 – Três principais tipos de manutenção industrial

Fonte: Zhu et al. (2024)

2.1.1 Manutenção Reativa

A manutenção reativa, que é a mais antiga e simples de aplicar, ocorre após uma falha ou paragem, ilustra Figura 2.1. Nunes et al (2023) referem que é a manutenção conhecida por "produzir até falhar", ou seja, só se procede à substituição ou reparação de uma máquina ou equipamento quando este está danificado e não consegue operar sem uma intervenção direta. Esta abordagem, também conhecida como manutenção corretiva, apesar de ser uma componente essencial de qualquer plano de manutenção, é considerada o método mais

ineficaz devido aos elevados custos associados a paragens de produção não programadas, resultantes do excessivo tempo de inatividade, o que levou ao aparecimento de métodos mais proativos, tendo a manutenção preventiva a primeira.

2.1.2 Manutenção Preventiva

Na manutenção preventiva, também conhecida como manutenção planeada, a realização da manutenção é feita de forma regular e programada, tendo por base ciclos de operação ou intervalos de tempo fixos, como se pode verificar na Figura *2.1*. Envolve inspeções periódicas realizadas por técnicos especializados e a substituição de peças é feita antes da ocorrência de uma falha crítica, ocorrendo em períodos iguais ou após um determinado número de ciclos de operação. Embora a manutenção preventiva seja útil para prevenir falhas e prolongar a vida útil dos equipamentos, também pode conduzir a desperdícios se não for bem gerida. Como alertam Zhu et al. (2024), realizar intervenções antecipadas pode, por um lado, levar a gastos excessivos em manutenção e, por outro lado, o adiamento da substituição pode ter consequências mais graves, exigindo uma intervenção de manutenção corretiva.

2.1.3 Manutenção Preditiva

Com os avanços da tecnologia na Indústria 4.0 e o desenvolvimento da IoT, surgiram métodos de manutenção baseados em condições atuais dos sistemas industriais. As inspeções, anteriormente realizadas por técnicos, passaram a ser automatizadas através de sensores e dispositivos capazes de medir, monitorizar e processar sinais que representam parâmetros físicos dos equipamentos industriais que possam indicar a diminuição da vida ou desempenho do equipamento (Nunes et al., 2023).

Muitas variáveis de fabrico são continuamente medidas em diversas fases, e os seus valores são armazenados nas bases de dados das organizações. Estes dados relacionam-se com as características dos produtos, as máquinas, a linha de produção (ou seja, que máquina foi utilizada com que parâmetros de configuração); os recursos humanos que operam a linha de produção (como o nível de experiência do trabalhador, tipo de turno); as matérias-primas utilizadas no processo; o ambiente (humidade, temperatura, entre outros); sensores ligados

às máquinas (vibração, força, pressão, tensão, entre outros); falhas/manutenções das máquinas; qualidade do produto; e outros fatores de produção significativos (Dogan & Birant, 2021).

Como resultado da evolução tecnológica, são geradas enormes quantidades de dados diariamente na indústria de produção. Segundo dados da Statista[3], prevê-se que até 2025 o volume de dados gerados e utilizados atinja os 181 zettabytes.

A manutenção preditiva ou manutenção baseada em condição, é considerada uma técnica de manutenção avançada, realiza-se antes de ocorrer uma falha, com base na sua previsão, como demonstrado na Figura 2.1. De acordo com Zhu et al. (2024), a manutenção preditiva representa o próximo passo na evolução da manutenção, destacando-se pela sua capacidade de prever falhas e estimar o RUL contribuindo assim para o planeamento e a eficiência operacional.

2.1.3.1 Desafios

A manutenção preditiva apresenta vários desafios, que Nunes et al. (2023) sistematizam. Um desses desafios é que os dados industriais podem estar sujeitos a erros nas medições, devido a condições ambientais adversas ou falhas nos sensores. Por outro lado, o aumento da quantidade de dados que têm de ser processados, armazenados e analisados representam um desafio específico, especialmente quando são necessárias ações em tempo real ou quase real. Além disso, o ambiente industrial atual é composto por equipamentos muito diversos e sistemas de gestão e produção flexíveis, que exigem que os sistemas de PdM sejam eficazes em cenários muito diferentes. Uma solução robusta deve incluir sistemas de monitorização eficazes, bem como técnicas de análise apropriadas para prever falhas e o RUL, e também fornecer uma arquitetura que possa processar grandes quantidades de dados em tempo real.

[3] Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2020, with forecasts from 2021 to 2025; https://www.statista.com/statistics/871513/worldwide-data-created/

2.1.3.2 Vida Útil Remanescente

A previsão do RUL de um equipamento é uma das principais características do PdM e faz parte do prognóstico e gestão da saúde (PHM, do inglês *Prognostics and Health Management*). Frequentemente, prognóstico e deteção de anomalias são abordados individualmente; no entanto, ambos são fundamentais para PdM, uma vez que os resultados da deteção de anomalias podem fornecer informações valiosas para os modelos preditivos e melhorar o seu desempenho, enquanto contribui para uma abordagem preditiva mais abrangente (Nunes et al., 2023).

Chen et al. (2020) definem o RUL como o tempo ou o número de ciclos entre o momento atual e o final da vida útil de uma componente ou sistema. Os processos de previsão do RUL dependem de indicadores do estado de saúde dos equipamentos, da calibração da curva de degradação e da estimativa do limite de falha. O RUL fornece informações relevantes para o PHM em vários aspetos:

1. A fiabilidade e segurança são dos fatores mais críticos em todo o ciclo de vida de equipamentos industriais ou mecânicos; falhas excecionais podem causar graves acidentes, levando à perda de propriedade e, nos casos mais graves, à perda de vidas. No entanto, as manutenções tradicionais estão programadas para corrigir problemas, podendo não conseguir detetar potenciais riscos. Através da monitorização da condição e da previsão do RUL, as falhas anormais podem ser detetadas antecipadamente para que a decisão de manutenção seja tomada no momento certo.
2. Os custos de manutenção representam uma grande parte do custo total de produção, e alguns deles são desnecessários. Com a previsão RUL, a manutenção de equipamentos em boas condições pode ser evitada, reduzindo custo.
3. Erros inesperados são, às vezes, causados por más configurações especificas de operação. Com a previsão do RUL, essas configurações podem ser ajustadas para prolongar vida útil dos equipamentos.

Com todos os benefícios descritos acima, a estimativa do RUL é aplicada em várias áreas, como motores de aviões, produção industrial, baterias de veículos, entre outros. No entanto, como alertam Chen et al. (2020), as previsões imprecisas do RUL, especialmente em estimativas de longo prazo, podem causar manutenções tardias e perdas irreparáveis.

2.1.3.3 Técnicas de previsão do RUL

Atualmente, as técnicas usadas para a previsão de RUL podem ser agrupadas em três categorias: baseadas em modelos físicos, modelos baseados em dados e abordagens híbridas. Habibullah et al. (2021) definem cada uma da seguinte forma:

- As técnicas baseadas em modelos físicos incorporam conhecimento prévio de modelos físicos, analíticos e/ou leis empíricas, juntamente com dados medidos, para representar o comportamento do equipamento. Embora o foco esteja no desenvolvimento de uma descrição precisa da condição do sistema, muitas vezes é matematicamente difícil caraterizar esses sistemas, tornando-os demasiado exigentes computacionalmente para serem aplicados;
- Os modelos baseados em dados dependem de dados históricos e da tentativa de criar modelos de previsão a partir desses dados. Assim, o foco está na utilização de algoritmos de aprendizagem computacional para fazer o mapeamento entre dados históricos e as condições de saúde dos equipamentos, tornando esses modelos viáveis desde que existam dados suficientes;
- Os modelos híbridos tentam combinar o conhecimento sobre o processo físico com as informações retiradas dos dados observados, de modo a melhorar o desempenho das previsões.

Habibullah et al. (2021) referem que todas estas categorias podem ser aplicadas em situações reais; no entanto, o uso de abordagens baseadas em modelos físicos e híbridas podem estar limitadas na prática, devido à imprecisão ou indisponibilidade do conhecimento físico em muitos equipamentos e sistemas. Em contrapartida, o uso de modelos baseados em dados tem vindo a ganhar destaque, devido à não exigência de

conhecimento físico, bem como ao avanço de sistemas de sensores modernos e técnicas de armazenamento e de análise de dados.

2.1.4 Comparação entre os diferentes tipos de manutenção

Nesta subsecção, são apresentadas as diferenças entre os três tipos de manutenção em termos de custo, benefícios, desafios, aplicações adequadas e inadequadas.

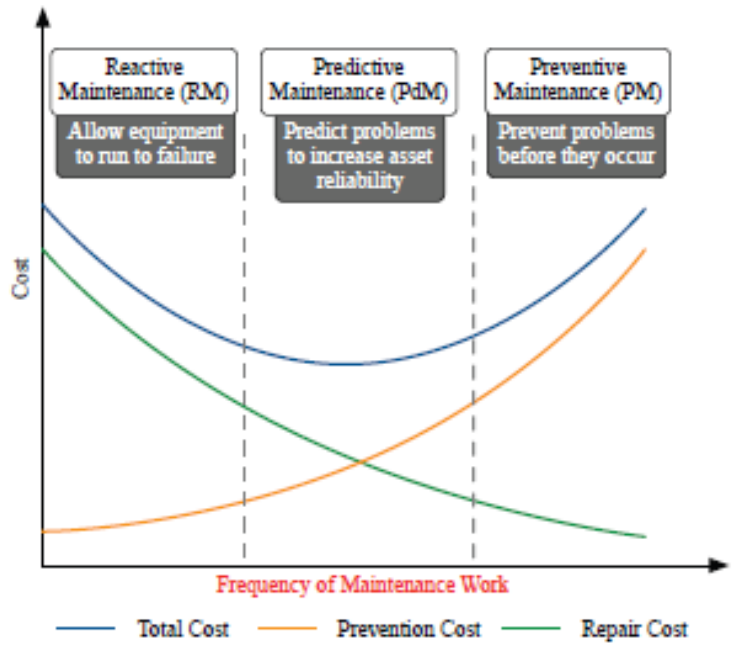


Figura 2.2 – Custos de cada manutenção

Fonte: Adaptado de Zhu et al. (2024)

Segundo a interpretação de Zhu et al. (2024) da Figura 2.2, a manutenção reativa tem o menor custo de prevenção por trabalhar até à falha. A manutenção preventiva tem o menor custo de reparação devido ao tempo de inatividade bem programado, enquanto a manutenção preditiva apresenta melhor balanceamento entre o custo de reparação e o custo de prevenção. Idealmente, a manutenção preditiva permite que a frequência de manutenção seja a mais baixa possível para evitar manutenções reativas não planeadas, sem incorrer a custos associados a fazer muita manutenção preventiva. O custo de prevenção contém custos de inspeção, custo de substituição preventiva, etc., enquanto o custo de reparo se

refere ao custo de reparação após a falha. Por fim, podemos ver de uma forma sucinta a lista de benefícios, desafios e aplicações para cada tipo de manutenção na Tabela 2.1.

Tabela 2.1 – Benefícios, desafios e aplicações de cada tipo de manutenção

	Benefícios	**Desafios**	**Aplicações Adequadas**	**Aplicações inadequadas**
Manutenção Reativa	Máximo aproveitamento e valor de produção Menor custo de prevenção	Tempo de inatividade não planeado Elevado custo de peças suplentes Maior potencial de danos ao equipamento Maior custo de reparação	Equipamento redundante ou não crítico Reparar equipamentos com baixo custo após a falha	A falha do equipamento cria um risco de segurança Disponibilidade de equipamento 24/7 é necessária
Manutenção Preventiva	Menor custo de reparação Menos falhas de equipamento e tempo de inatividade não planeado	Necessidade de inventário Aumento do tempo de inatividade planeado Manutenção em equipamentos em boas condições	Equipamentos em que aumenta probabilidade de falha com o tempo ou uso	Falhas aleatórias que não estão relacionadas com a manutenção
Manutenção Preditiva	Visão constante da saúde do equipamento; Melhorias de análise de dados; Evitar trabalhar até falha e substituição de um componente com vida útil;	Maior custo inicial de infraestrutura (por exemplo, sensores); Sistema mais complexo;	Modos de falha que podem ser previstos de forma económica e com monitorização regular;	Não possuem falha que pode ser previsto de forma económica;

Fonte: Adaptado de Zhu et al. (2024)

Complementando a Tabela 2.1, Thomas (2018) refere que a manutenção reativa apresenta um custo elevado de mão de obra e peças, sendo considerada não rentável. A manutenção preditiva, por outro lado, tem um custo de mão de obra relativamente baixo e custo médio em peças, além de proporcionar uma poupança significativa de custos.

Por exemplo, Thomas e Weiss (2021) analisaram um conjunto de 71 empresas e concluíram que aquelas que utilizam mais manutenção preditiva e preventiva, invés da manutenção reativa, registaram 52,7% menos paragens não planeadas e 78,5% menos defeitos nos produtos. Entre as que utilizam mais manutenção preditiva e preventiva, apresentaram 18.5% menos paragens não programadas e 87.3% menos defeitos utilizando mais manutenção preditiva.

Segundo a literatura, a implementação de técnicas avançadas de manutenção, como a manutenção preditiva tem apresentado resultados em diferentes áreas, como refere Thomas (2018), sendo eles:

- Redução dos custos de manutenção até 83%;
- Redução em defeitos e retrabalho dos produtos até 72%;
- Redução de paragens forçadas até 25%;
- Aumento de produtividade laboral até 10%;
- Redução de inventário até 31%:
- Aumento da produção até 38%;
- Redução de acidentes até 25%;
- Redução na rejeição dos clientes até 19%;
- Redução do tempo de paragens não programadas até 10%.

2.1.5 Custos de Manutenção Industrial

Thomas (2018) destaca as seguintes estimativas da manutenção industrial:

- Os custos de manutenção industrial podem variar entre 15% e 70% dos custos de produção dos produtos;
- Aproximadamente um terço dos custos de manutenção são desnecessários ou indevidamente realizados;
- A manutenção preventiva é aplicada desnecessariamente até 50% do tempo;
- Na Suécia, os custos das paragens não programadas representam 23,9% do custo total de produção.

A literatura não é consensual neste tema, uma vez que os custos de manutenção variam de país para país, da indústria e das métricas utilizadas para calcular os custos. Por exemplo, Thomas e Weiss (2021) estimaram que, em 2016, o total de custos/perdas associados à manutenção foi, em média, de 222 mil milhões de dólares. Desse montante, 15,7 mil milhões referiam-se a custos provocados por falhas e erros; e 18,4 mil milhões a custos de paragens não programadas.

2.2 Aprendizagem Computacional na Manutenção Preditiva

A aprendizagem computacional é uma parte da inteligência artificial e pode ser definida segundo os autores Mathew et al. (2017) como a utilização e desenvolvimento de sistemas computacionais, capazes de aprender a partir dos dados disponíveis, identificar padrões e tomar decisões com uma reduzida intervenção humana utilizando algoritmos e modelos estatísticos. Uma das suas possíveis aplicações é a previsão de resultados com base num modelo treinado com dados históricos. Existem duas categorias de ML, são elas a aprendizagem supervisionada (ou preditiva) e aprendizagem não-supervisionada (ou descritiva).

Na aprendizagem supervisionada, o conjunto de dados inclui as suas classes de saída, que são utilizadas para treinar o modelo, enquanto na aprendizagem não supervisionada a informação sobre a classe de saída não está disponível; em vez disso, os dados são agrupados em classes desconhecidas.

Existem dois tipos de tarefas na manutenção preditiva, as tarefas de classificação e de regressão. Nas tarefas de classificação é prevista a possibilidade de haver ou não falha n ciclos à frente, enquanto nas tarefas de regressão é feita a previsão do tempo restante até à próxima falha, conhecida como RUL.

Neste projeto é desenvolvido um modelo de previsão baseado em aprendizagem computacional supervisionada, aplicando uma tarefa de regressão calculando o RUL.

2.2.1 Algoritmos

Na revisão de literatura de Dogan e Birant (2021) e nos estudos de Chen et al. (2020) e Mathew et al. (2017), são referidos os métodos de aprendizagem computacional tradicionalmente aplicados, as suas vantagens e desvantagens. Essa informação encontra-se de seguida sintetizada.

Dogan e Birant (2021) destacam os seguintes pontos. Consideram benéfico o uso de uma abordagem diversificada de algoritmos ML em PdM, com diferentes algoritmos aplicáveis para tarefas específicas. Redes neuronais, máquinas de vetores de suporte (SVM), árvores de decisão e métodos *ensemble* são destacados pela sua eficácia em monitorização, previsão de falhas e análise de qualidade.

As redes neuronais são consideradas eficazes em grandes volumes de dados e padrões complexos, ideais para se identificarem padrões de falha em contextos de alta dimensionalidade.

A aplicação de SVM para classificação e regressão, especialmente em dados de alta dimensionalidade.

Árvores de Decisão e KNN são valorizados pela simplicidade e interpretabilidade, sendo considerados úteis em tarefas de classificação em bases de dados de menor dimensão.

Recomendam a utilização de múltiplos métodos para aumentar a precisão e resiliência dos modelos PdM em cenários industriais, combinando, por exemplo, *deep learning* para precisão com métodos como RF para explicabilidade.

Como resultado do seu estudo, Chen et al. (2020) destacam os seguintes pontos. As árvores de decisão aleatórias (RF, do inglês *Random Forest*) são uma abordagem eficaz para a previsão da vida útil restante (RUL), equilibrando precisão com eficiência computacional. Comparando com outros seis modelos, incluindo modelos de aprendizagem profunda, o RF é o mais eficiente em computação e apresenta um melhor desempenho na menor raiz do erro quadrático médio (RSME, do inglês *Root Mean Square Error*), facilitando, assim, a sua implementação prática em contextos industriais. Consideram o método robusto para evitar sobreajuste e adequado para identificar padrões de degradação. O sobreajuste é um

problema na estimação do RUL, assim, as árvores de decisão ganham destaque na sua utilização.

Referem ainda que, apesar dos modelos de aprendizagem profunda apresentarem geralmente elevado desempenho, são menos aplicados na prática devido ao elevado custo computacional e à dificuldade de interpretação dos resultados. A complexidade destes modelos também afeta o tempo de treino do modelo, que é tipicamente muito maior que nos modelos tradicionais de aprendizagem computacional.

Os resultados obtidos por Mathew et al. (2017) vão ao encontro dos resultados obtidos no estudo anterior. Entre os 10 algoritmos testados para a previsão do RUL de motores de avião, o RF obteve a maior precisão, apresentando o menor RMSE, destacando-se como o algoritmo mais eficaz para a tarefa. Esse desempenho sugere que o RF é especialmente adequado para PdM em ambientes com dados históricos de sensores.

As RF são uma técnica de agrupamento (em inglês, *ensemble*) capaz de realizar tarefas de regressão e classificação utilizando múltiplas árvores de decisão e técnicas de *bagging*. Cria tantas árvores quantas as que forem possíveis no subconjunto de dados e combina os resultados de todas as árvores através da média, o que leva à redução do sobreajuste (do inglês, *overfitting*), bem como a variância, melhorando assim a precisão (do inglês, *accuracy*). Árvores de decisão não necessitam de muito pré-processamento, aceitam tanto variáveis numéricas como categóricas e não são sensíveis a valores inconsistentes (Chen et al., 2020).

XGBoost é um algoritmo de aprendizagem computacional dos modelos de agrupamento, baseado em árvores de reforço de gradiente (em inglês, *gradiant boosting trees*), que constrói modelos preditivos poderosos ao integrar múltiplos modelos de árvore de decisão. Apresenta vantagens, tais como, alto desempenho, escalabilidade e flexibilidade, assim como a avaliação da importância de variáveis e computação paralela, que permite lidar com grandes conjuntos de dados e evitar o sobreajuste. Normalmente, é utilizado para resolver problemas como classificação e regressão. Consegue isso treinando iterativamente vários aprendizes fracos, ou seja, árvores de decisão. Cada iteração corrige o erro do modelo com base nos resultados da previsão da iteração anterior (Tian, 2024).

Em síntese, no RF, as árvores são independentes e são construídas em paralelos, já no XGBoost, as árvores estão interligadas e são construídas sequencialmente, basicamente pega no que estava mal numa árvore e melhora para a árvore seguinte e assim sucessivamente.

2.2.2 Conjuntos de dados benchmark

Segundo Dogan e Birant (2021), na literatura alguns conjuntos de dados são amplamente utilizados para aplicar algoritmos de aprendizagem computacional. Estes conjuntos de dados *benchmark* são importantes porque permitem a comparação de algoritmos, garantem a reprodução de experiências com as mesmas condições e fornecem em alguns casos desafios realistas nos que utilizam dados reais. Além disso, levam ao desenvolvimento de soluções robustas, capazes de lidar com dados complexos e, por terem um histórico de uso na comunidade, permitem acompanhar a evolução dos algoritmos ao longo do tempo.

No seu estudo, Chen et al. (2020) utilizaram o conjunto de dados dos motores de avião disponibilizado pela NASA, mais conhecido como NASA C-MAPSS (em inglês, *Turbofan Engine Degradation Simulation Data Set).* Este conjunto de dados é muito utilizado em competições de aprendizagem computacional, patrocinadas pelo repositório de dados abertos Kaggle.

O conjunto de dados SECOM, derivado de um processo de produção de semicondutores, é um dos mais populares e é utilizado para avaliar modelos que lidam com a previsão de falhas e controlo de qualidade. Outro conjunto de dados, é o *Steel Plates Faults*, que contém informações sobre falhas em placas de aço e é aplicado para treinar algoritmos de reconhecimento automático de padrões de defeitos. O conjunto de dados *Bosch Production Line Performance* também é bastante utilizado para testar a classificação de variáveis relacionadas com o chão de fábrica. Estes conjuntos de dados oferecem bases robustas para comparar a eficácia de diferentes algoritmos em cenários de manufatura Dogan e Birant (2021).

2.2.3 Processamento de dados

Nunes et al. (2023) destacam que o desenvolvimento de um sistema de manutenção preditiva eficaz exige um tratamento rigoroso dos dados. A qualidade dos dados é fundamental, sendo essencial a limpeza de dados para resolver problemas como dados omissos, redundância, ruído e *outliers*, questões comuns em dados industriais que podem comprometer a precisão dos algoritmos. Dados redundantes, que não acrescentam informação nova, são considerados irrelevantes e podem ser identificados através de gráficos de correlação. Dados omissos referem-se a valores não disponíveis no conjunto de dados e podem ser tratados para evitar que influenciem os resultados. A suavização dos dados é outra etapa importante, pois ajuda a reduzir o ruído e a corrigir variações temporais anómalas, o que melhora a precisão da análise.

A seleção e transformação de dados desempenham um papel fundamental. A escolha adequada dos sensores e a transformação dos dados recolhidos permitem o processamento apenas das informações relevantes, aumentando assim a eficácia do sistema de manutenção preditiva. A transformação pode envolver a adaptação de variáveis e a criação de novas variáveis derivadas, assim como a criação de novos registos. Os dados provenientes de diferentes sensores geralmente estão em escalas distintas; por isso, a normalização é fundamental para manter a integridade e a comparabilidade dos dados. Entre as técnicas práticas de normalização encontram-se a normalização min-max, a normalização de pontuação Z e a normalização de escala decimal.

A redução de dados pode incluir a seleção de variáveis e a combinação destas com técnicas de redução de dimensionalidade, assim como a seleção de amostras representativas. Adicionalmente, a calibração dos modelos aplicando validação cruzada nos dados de treino e durante a fase de hiperparametrização garante a robustez do sistema preditivo.

2.2.4 Métricas de desempenho para avaliação de modelos

Para avaliar os modelos de aprendizagem computacional construídos em cenários de manufatura, as métricas de desempenho mais utilizadas variam de acordo com o tipo de problema abordado. Segundo Dogan e Birant (2021) em tarefas de classificação, métricas

como precisão (*accuracy*), *f-measure* e a matriz de confusão são normalmente utilizadas para avaliar a eficácia dos modelos na previsão correta das categorias dos dados. Para problemas de regressão, o erro médio absoluto (MAE, do inglês *Mean Absolute Error*) e o RMSE são utilizados para medir a precisão das previsões de variáveis de saída contínuas. Estas métricas permitem uma avaliação quantitativa da capacidade do modelo para generalizar e realizar previsões precisas na indústria da manufatura.

Para avaliar o desempenho do modelo, foram escolhidas as seguintes métricas:

- RMSE: O RMSE é o valor do erro obtido pela raiz quadrada do erro quadrático médio (MSE, do inglês *Mean Squared Error*) e é a métrica mais utilizado na análise de regressão. A razão para utilizar a raiz do erro é para que o erro esteja na mesma unidade da variável de resultado, facilitando a interpretação. O RMSE varia sempre entre 0 e infinito positivo. Valores mais baixos de RMSE indicam um melhor desempenho predito.
- Coeficiente de Determinação (R2, do inglês *Coefficient of Determination*): O R-quadrado descreve a quantidade de variação capturada pelo modelo desenvolvido. O seu valor varia sempre entre 0 e 1. Quanto maior o valor de R-quadrado, melhor o modelo se ajusta aos dados.
- MAE: Mede a magnitude média absoluta entre os valores reais e os valores previstos por um modelo de regressão. Valores mais baixos indicam um melhor desempenho preditivo.
- Erro Percentual Absoluto Médio (MAPE, do inglês *Mean Absolute Percentage Error*): O MAPE é fácil de interpretar, pois mostra os valores de erro em percentagens. Por exemplo, um MAPE de dez por cento significa que os valores reais e previstos apresentam uma diferença de dez por cento.

2.2.5 Vantagens

A utilização de técnicas de aprendizagem computacional na manufatura oferece diversas vantagens, tanto em termos de eficiência como de poupança de recursos. Uma das principais vantagens é a possibilidade de realizar manutenção preditiva, prevendo falhas

nas máquinas antes que ocorram, o que reduz os custos associados à manutenção corretiva e às paragens inesperadas. Além disso, as técnicas de ML contribuem para a otimização da utilização de recursos, melhorando o planeamento e a alocação de equipamentos e mão de obra. Estas técnicas também desempenham um papel fundamental na melhoria do design de produtos e na previsão de erros de qualidade, permitindo um controlo de qualidade mais preciso e uma redução de desperdícios. Com a capacidade de analisar grandes volumes de dados em tempo real, o ML permite tomadas de decisão mais informadas e rápidas (Dogan & Birant, 2021).

2.3 Metodologias aplicáveis

Existem duas metodologias aplicadas, a já descrita CRISP-DM e a Descoberta de Conhecimento em Bases de Dados (KDD, do inglês, *Knowledge Discovery in Databases*).

A metodologia KDD consiste em cinco etapas e é semelhante à metodologia CRISP-DM. A primeira etapa é a compreensão do domínio do negócio, onde se define o objetivo da aplicação, os recursos disponíveis, as restrições e os critérios de sucesso. Segue-se a preparação dos dados, que envolve a recolha, integração, limpeza, redução e transformação dos dados, garantindo a sua qualidade antes de serem aplicados os algoritmos. A terceira etapa consiste na aplicação dos algoritmos de aprendizagem computacional ou mineração de dados, a fim de extrair padrões dos dados. Após isso, os modelos são avaliados através de métricas apropriadas, como a precisão ou o erro quadrático médio. A última etapa é a apresentação dos resultados, geralmente através de visualizações ou indicadores de desempenho que apoiam a tomada de decisão. Este processo pode ser repetido até que os resultados desejados sejam alcançados (Dogan & Birant, 2021).

A Tabela 2.2 permite verificar as semelhanças e diferenças entre a metodologia KDD e a CRISP-DM.

Tabela 2.2 – Comparação entre as duas metodologias

Metodologia	**KDD**	**CRISP-DM**
Foco	Técnico, focado na descoberta de conhecimento em bases de dados	Orientado para negócios, aplicável em diversas indústrias
Etapas	As fases incluem seleção, transformação de dados, mineração, avaliação e interpretação	Compreende seis fases: compreensão do negócio, entendimento dos dados, preparação dos dados, modelação, avaliação e implementação
Flexibilidade	Menos flexível, com foco nos detalhes técnicos específicos das operações de mineração de dados	Altamente flexível, adaptável a diversos contextos de negócios
Iteratividade	Processo iterativo que permite aperfeiçoar as técnicas de seleção e transformação de dados conforme novas informações são obtidas	Estrutura iterativa e cíclica que suporta revisões frequentes em todas as fases para um ajuste continuo e otimizado
Governança	Menor ênfase na governança estruturada, focando mais na eficácia técnica das operações de mineração de dados	Forte ênfase na governança do projeto, com diretrizes claras para documentação e comunicação ao longo de todas as fases
Aplicabilidade	Eficaz para projetos focados em TI e análise de dados, onde a manipulação técnica dos dados é fundamental	Aplica-se a uma variedade de indústrias, incluindo marketing, risco, saúde e finanças, devido à sua adaptabilidade e abordagem focada no negócio

Em suma, foi selecionada a metodologia para o desenvolvimento deste projeto, por ser uma metodologia criada pela indústria, sendo por isso muito focada no negócio, é flexível e adaptável ao contexto do negócio.

3 SINMETRO

Este capítulo começa com a apresentação da entidade acolhedora do estágio, seguido do seu organograma e, por fim, uma descrição do seu *software* e de uma das soluções disponível aos clientes.

3.1 Apresentação

A SINMETRO foi fundada em 2002 para lançar no mercado o *software* ACCEPT, uma solução para controlo da Qualidade Industrial. Iniciou atividade no setor da indústria alimentar, concretamente nos pré-embalados. A versatilidade do sistema ACCEPT permitiu expandir para outros setores, como a indústria química e metalomecânica, tabaco, automóvel, construção, cerâmica, vidros, plásticos e moldes. Com 22 anos de experiência, conta até ao momento com 26 colaboradores e dispõe de dois escritórios, em Coimbra e Leiria, sendo este último a sede da empresa. Está presente em oito países, num total de mais de 220 clientes em diversos setores da indústria (SINMETRO, 2023b).

A SINMETRO tem como missão a criação e execução de projetos que promovam a inovação organizacional, de ideias, processos, produtos, serviços, tecnologias e de mercados, com recurso ao sistema ACCEPT e outras soluções comerciais. Exerce atividades de consultoria tecnológica, desenvolvimento de *software* e formação (SINMETRO, 2023c).

O serviço de consultoria, disponível a diferentes setores industriais, tem como foco a Investigação e Desenvolvimento (I&D), Inovação Organizacional e Produtiva, Estratégias Globais de Indústria 4.0 e Melhoria Contínua e/ou Reengenharia de Processos através das metodologias Lean & 6 Sigma (SINMETRO, 2023c).

No âmbito do desenvolvimento de *software* à medida, são criadas soluções adaptadas às necessidades, preferências e restrições dos clientes e utilizadores, procurando sempre compreender todas as expetativas antes de recomendar a digitalização das organizações, que tem o foco centrado nas pessoas e nos processos (SINMETRO, 2023c).

Na área de formação/academia, elaboram planos de formação personalizados para empresas em diversas áreas, como Lean & 6 Sigma, ferramentas de qualidade, metrologia, MINITAB, entre outras. Estes programas têm como objetivo principal contribuir para o desenvolvimento de organizações sustentáveis, onde o valor está intrinsecamente ligado à geração de conhecimento. Promovendo a integração desse novo conhecimento em novos produtos, serviços e processos, que resultam em vantagem competitiva, abrindo novas oportunidades e mercados (SINMETRO, 2023c).

Desde a sua criação, tem vindo a receber reconhecimentos significativos. Desde 2004, são uma entidade credenciada pela Direção-Geral do Emprego e das Relações de Trabalho para ministrar formação profissional nas áreas de ciências empresariais, enquadramento na organização/empresas, estatística, engenharia e disciplinas relacionadas. Além disso, desde 2019, são reconhecidos pela Agência Nacional de Inovação para a realização de atividades de I&D. Uma demonstração do excelente trabalho realizado nos últimos 3 anos, foi a atribuição do Estatuto de Inovadora pela COTEC Portugal - Associação Empresarial para a Inovação, devido aos elevados níveis de solidez financeira, competências em inovação e desempenho económico alcançados (SINMETRO, 2023c).

3.2 Organograma

Como já foi referido, a SINMETRO é constituída por 26 colaboradores, estando cada um alocado a um departamento. Na figura Figura 3.1, estão representadas todos os departamentos e as diferentes equipas.

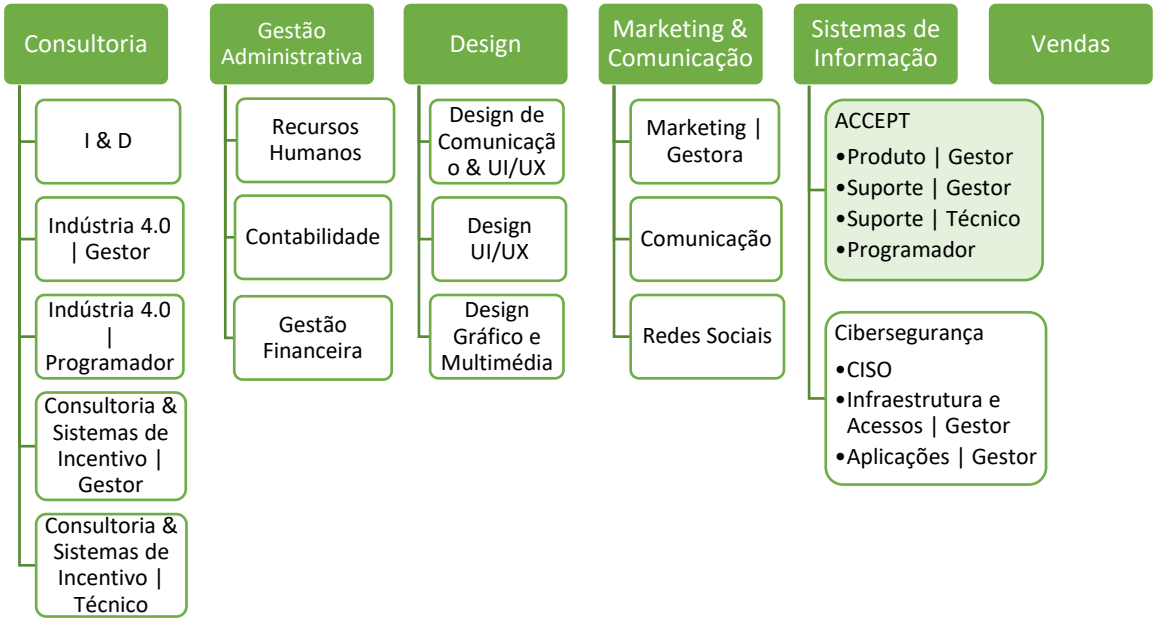


Figura 3.1 – Modelo Orgânico da SINMETRO

Fonte: Adaptado de (SINMETRO, 2023b)

Durante o decorrer do estágio, a integração foi no departamento de Sistemas de Informação, na equipa do ACCEPT, conforme ilustrado na Figura 3.1.

3.3 ACCEPT

A sua principal solução tecnológica, o sistema ACCEPT, é uma plataforma para controlo da qualidade industrial que reúne um conjunto de módulos que permitem obter uma visão global do desempenho dos parâmetros críticos dos produtos e processos, através de uma análise estatística avançada em tempo real, de forma a assegurar a eficiência dos processos produtivos, o cumprimento de especificações e a qualidade do seu produto, tornando numa

poderosa ferramenta da Indústria 4.0. O sistema ACCEPT, é constituído por três áreas funcionais, assegurando um fluxo contínuo de informação: QUALITY, para o controlo estatístico da qualidade; OEE, para a monitorização do desempenho das linhas de produção; e MSA, para a análise e gestão dos sistemas de medição (SINMETRO, 2023a).

Este sistema, apresenta várias vantagens (SINMETRO, 2023a):

- *dashboards* e relatórios parametrizáveis;
- visão centralizada de toda a informação necessária, para a gestão dos seus processos da qualidade;
- integração com equipamentos e sistemas para total automação da produção;
- exportação de dados e integração nos fluxos de informação das empresas;
- sistema de alertas em tempo real, através de notificações, e-mails ou relatórios.

3.4 ACCEPT OEE

O sistema ACCEPT tem disponível várias soluções como: QualityHub, 360º, Quality, Energy, MSA e OEE. Cada solução tem a sua relevância, dependendo das necessidades dos clientes.

O conceito de OEE foi desenvolvido em 1960 por Seiichi Nakajima, sendo um indicador-chave de desempenho que permite avaliar e identificar melhorias na eficiência e eficácia do chão de fábrica (SINMETRO, n.d.).

O objetivo do OEE é monitorizar o desempenho de linhas de produção, utilizando métricas bem definidas que permitam comparar facilmente diferentes linhas e acompanhar a sua evolução ao longo do tempo. A comparação entre o desempenho real e o desempenho ideal das linhas de produção resulta num valor percentual (rácio), identificado como OEE (Li et al., 2022).

Li et al. (2022) explicam que este indicador é constituído por três rácios.

1. Disponibilidade: indica o tempo de produção, tendo em conta as paragens planeadas e as não planeadas, sendo a sua diferenciação muito importante para a correta definição do tempo planeado de produção.

2. Desempenho: indica a velocidade de produção real em relação à velocidade previsional, tem em conta ciclos lentos e paragens pequenas.
3. Qualidade: mostra o controlo de qualidade dos produtos, tendo em consideração defeitos.

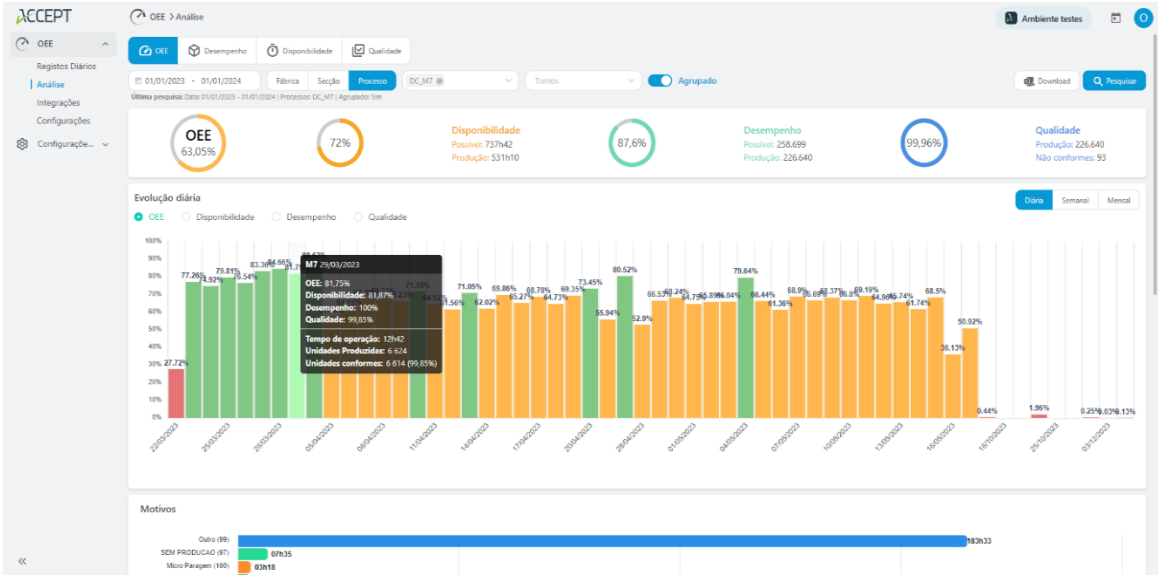


Figura 3.2 – Captura de ecrã de análise do OEE no sistema ACCEPT

Fonte: ACCEPT OEE

Utilizando como referência a Figura 3.2, é possível perceber que o ACCEPT permite realizar análises em tempo real, apresentando os resultados sob a forma de *dashboards*. Estas análises fornecem informações relevantes sobre o desempenho da linha de produção, incluindo motivos de paragens, trocas de produção e quantidades produzidas. É possível analisar rapidamente a evolução do indicador do OEE ao longo de semanas, meses ou anos, por linha, secção, turno ou até mesmo para a fábrica, e por cada parâmetro: disponibilidade, desempenho e qualidade. Desta forma, é possível monitorizar o desempenho das linhas de produção, permitindo identificar tendências, as linhas de produção que estão a operar abaixo dos parâmetros normais, bem como definir e priorizar iniciativas de melhoria do processo para evitar paragens. Todas as paragens não planeadas, como avarias, preparação do arranque, mudanças de produto ou limpezas, devem ser corretamente categorizadas.

4 Desenvolvimento do Projeto

Neste capítulo, são descritas as restantes fases da metodologia CRISP-DM, concretamente o entendimento dos dados, a preparação dos dados, a modelação e por último a avaliação. A fase da implementação não ocorreu neste projeto em concreto.

4.1 Entendimento dos dados

Nesta secção, é desenvolvida a fase do entendimento dos dados, descrevendo a extração dos dados, seguida da análise exploratória de dados, terminando com a verificação da qualidade dos dados.

4.1.1 Extração dos dados

Neste projeto, foram utilizados dados reais de um cliente da SINMETRO, pertencente à indústria de manufatura. Os dados, já existentes nas bases de dados da SINMETRO, foram extraídos utilizando SQL (do inglês, *Structured Query Language)*. Embora SQL não seja uma linguagem de programação, funciona como tal na criação de tabelas, consulta e manipulação de informações em bases de dados relacionais. A manipulação é feita através de *queries*, comandos que, ao serem executados, retornam as informações armazenadas.

Antes de se proceder à extração dos dados para serem analisados em Python, foi necessário, numa primeira fase, compreender estes dados e identificar as informações/variáveis eram relevantes para o projeto através de *queries* em SQL. Após a seleção das variáveis relevantes de cada tabela, foram criados dicionários que apresentam as descrições relevantes, conforme ilustram as Tabela 4.1 e Tabela 4.2. Toda a informação presente nesses dicionários foi validada pela pessoa responsável pelas bases de dados e pela posterior análise de dados em Python.

Na Tabela 4.1 são apresentadas as variáveis que foram inicialmente consideradas relevantes para a criação do conjunto de dados a ser utilizado. Na primeira coluna encontra-se o código da variável em SQL. Na segunda coluna, está a designação da variável em Python, criada para uma melhor interpretação. Na última coluna, consta uma explicação de cada variável. É importante referir que as variáveis “Programada”,

"Nao_Programada_Avaria" e "Nao_Programada_Externa" foram criadas a partir da variável "Paragem_Ativa".

Tabela 4.1 – Dicionário de Variáveis

Código da variável (SQL)	**Designação da variável (Python)**	**Descrição da variável**
TS_Inicio_RD	Inicio	Timestamp inicial do registo validado do estado de produção
TS_Fim_RD	Fim	Timestamp final do registo validado do estado de produção
Tempo_Decorrido_RD	Duracao	Duração do estado de produção do registo, em segundos(s) TS_Fim_RD - TS_Inicio_RD
ID_Processo	ID_Maquina	Identificação da máquina
Estado_Processo	Estado_Maquina	Indicador do estado do processo 0 – parado; 1 – produzir Os dados deste indicador são de sensores
Paragem_Ativa	Paragem	Código, numérico ou string, do motivo da paragem
QTD_OK_RD	Qtd_OK	quantidade peças produzidas conformes (nº)
QTD_NOK_RD	Qtd_NOK	quantidade peças produzidas não conformes (nº)
QTD_Total_RD	Qtd_Total	quantidade total de peças produzidas (nº); QTD_OK_RD + QTD_NOK_RD;
Ignora_contadores	Invalida_Qtd_OEE	Variável de validação das quantidades (QTD_Total_RD, QTD_OK_RD, QTD_NOK_RD) para o cálculo do OEE em função do estado de processo
ID_Produto	ID_Produto	ID produto, é um identificador numérico
Paragem_Ativa	Programada	Indicador binário das paragens programadas 1 = Verdadeiro ; 0 = Falso
Paragem_Ativa	Nao_Programada_Avaria	Indicador binário das paragens não programadas por avarias relacionadas com a máquina 1 = Verdadeiro ; 0 = Falso
Paragem_Ativa	Nao_Programada_Externa	Indicador binário das paragens não programadas por fatores externos à máquina 1 = Verdadeiro ; 0 = Falso
Tipo_estado	Dia_Valido	Indicador binário se o processo do dia é válido 1 = Verdadeiro ; 0 = Falso

Nas duas primeiras colunas temos o código e a descrição de cada paragem retirado do SQL. Na última coluna, temos a categorização de cada paragem em programada, não programada por avaria ou não programada por causas externas, como se pode verificar na Tabela 4.2.

Tabela 4.2 – Dicionário de paragens

Código	Descrição	Categorização
01	Afinação de Ferramenta	Paragem programada
02	Avaria (Man. Curativa)	Paragem não programada - avaria
03	Controlo da 1ª Peça	Paragem programada
04	Dedicação a outra Máquina	Paragem não programada – causas externas
05	Deslocações ao Armazém (Turno)	Paragem programada
06	Falta de instrução de Fabrico	Paragem não programada – causas externas
07	Falta de Matéria-Prima	Paragem não programada – causas externas
08	Falta de Operador	Paragem não programada – causas externas
09	Ligar Máquina + Referência (Turno)	Paragem programada
10	Manutenção preventiva	Paragem programada
11	Protótipo	Paragem programada
12	Reparação de Ferramenta de Posição	Paragem não programada - avaria
13	Setup (Preparação + 1ª Peça)	Paragem programada
14	Troca de peça	Paragem programada
15	Almoço	Paragem programada
16	Controlo Qualidade (Ajustes)	Paragem programada
17	Jantar	Paragem programada
18	Fim de Turno	Paragem programada
20	Falta ferramenta de corte	Paragem não programada – causas externas
21	Reparação de ferramenta de CORTE	Paragem não programada - avaria
22	Peça retrabalhada	Paragem programada
23	Controlador de qualidade indisponível	Paragem não programada – causas externas
24	Procura de ferramenta de posicionamento/corte	Paragem não programada – causas externas
25	Peça Produzida 1ª Vez no posto	Paragem programada
26	Melhoria Processo de Maquinagem	Paragem programada
27	BACKUP PROGRAMA	Paragem programada
28	Falha de comunicação do Sistema (ACCEPT)	Paragem não programada – causas externas
29	Acerto do Posicionamento da Peça	Paragem programada
95	Paragem planeada (M01/ M0)	Paragem programada
100	Micro Paragem	Micro-paragem

A categorização de cada tipo de paragem foi feita apenas com base nas informações fornecidas pela SINMETRO, não tendo sido possível confirmar esses dados junto do cliente. As paragens programadas são aquelas que empresa tem supostamente conhecimento de quando vão acontecer, estão agendadas, por exemplo, pausas para almoço. As paragens não programadas por causas externas, são aquelas que a empresa não sabe quando vão acontecer, nem é possível prevê-las com base nos dados disponíveis, por exemplo, falhas de energia. As paragens não programadas por avaria são aquelas que a empresa não sabe que vão acontecer, mas que são relevantes para a manutenção preditiva e que se espera poder prever com os dados disponíveis. As micro-paragens fazem parte do processo de produção e, portanto, foram consideradas como tempo de operação numa fase posterior.

Foi também criada um dicionário de produtos com a identificação e descrição de cada produto fabricado, que, contudo, acabou por não ser utilizado no desenvolvimento do projeto, não sendo, portanto, relevante para este relatório.

Após este passo, por meio de uma *super query*, as variáveis foram selecionadas das respetivas tabelas originais e inseridas numa nova tabela temporária, designada "Temp_Table". Ao executar a *super query* (que pode ser consultada no APÊNDICE 1), foi gerada a "Temp_Table". A Figura 4.1 apresenta os cinco primeiros registos dessa tabela.

	Inicio	Fim	Duracao	ID_Maquina	Estado_Maquina	Paragem	Qtd_OK	Qtd_NOK	Qtd_Total	Invalida_Qtd_OEE	ID_Produto	Programada	Nao_Programada_Avaria	Nao_Programada_Externa	Valida_Dia
1	2022-01-03 06:00:00.000	2022-01-03 06:01:46.000	106	11	0	09	NULL	NULL	NULL	0	2	1	0	0	1
2	2022-01-03 06:00:00.000	2022-01-03 06:06:53.000	413	6	0	09	NULL	NULL	NULL	0	2	1	0	0	1
3	2022-01-03 06:00:00.000	2022-01-03 06:07:46.000	466	8	0	09	NULL	NULL	NULL	0	545	1	0	0	1
4	2022-01-03 06:00:00.000	2022-01-03 06:09:16.000	556	7	0	01	NULL	NULL	NULL	0	492	0	1	0	1
5	2022-01-03 06:01:46.000	2022-01-03 06:01:56.000	10	11	1	NULL	NULL	NULL	NULL	NULL	2	0	0	0	1

Figura 4.1 – Cinco primeiros resultados da "Temp_Table"

A empresa possui um conjunto de máquinas e, em termos de OEE, é relevante registar os períodos temporais em que cada uma destas máquinas está a produzir ou está parada. O estado da máquina reflete-se no estado da produção e, consequentemente, no estado do processo. A tabela original contém os registos detalhados dos períodos de atividade e inatividade das máquinas a analisar. Assim, cada linha representa um intervalo de tempo em que uma máquina esteve num de dois estados: a produzir ou parada.

Inicialmente, a extração dos dados do SQL seria feita para um ficheiro em formato csv. No entanto, após alguma investigação, foi possível aceder diretamente ao SQL via VSCode, através da instalação da extensão mssql e da biblioteca pyodb. O código completo encontra-se no APÊNDICE 2. Com esta abordagem, reduziu-se a utilização de ficheiros csv, permitindo adicionalmente ter os dados atualizados no momento, caso houvesse alguma alteração no SQL.

4.1.2 Análise exploratória de dados

Durante a inspeção dos dados, verificou-se que algumas colunas apresentavam formatos de dados incorretos, conforme ilustrado na Figura 4.2 e descrito a seguir.

```
#Data Type das colunas
df.dtypes
```

```
Inicio                  object
Fim                     object
Duracao                  int64
ID_Maquina               int64
Estado_Maquina            bool
Paragem                 object
Qtd_OK                 float64
Qtd_NOK                float64
Qtd_Total              float64
Invalida_Qtd_OEE        object
ID_Produto             float64
Programada                bool
Nao_Programada_Avaria     bool
Nao_Programada_Externa    bool
dtype: object
```

Figura 4.2 – Formato das colunas do conjunto de dados inicial

As colunas “Inicio” e “Fim” deveriam estar no formato de data; a coluna “Estado_Maquina” deveria ter o formato inteiro; as colunas das quantidades estão no

formato decimal, mas deveriam ser inteiras; a coluna "Paragem" deveria ser inteira; a coluna 'Invalida_Qtd_OEE' está no formato *object* e deveria ser booleana; o "ID_Produto" está no formato decimal e deveria ser inteiro. As restantes colunas encontram-se no formato desejado.

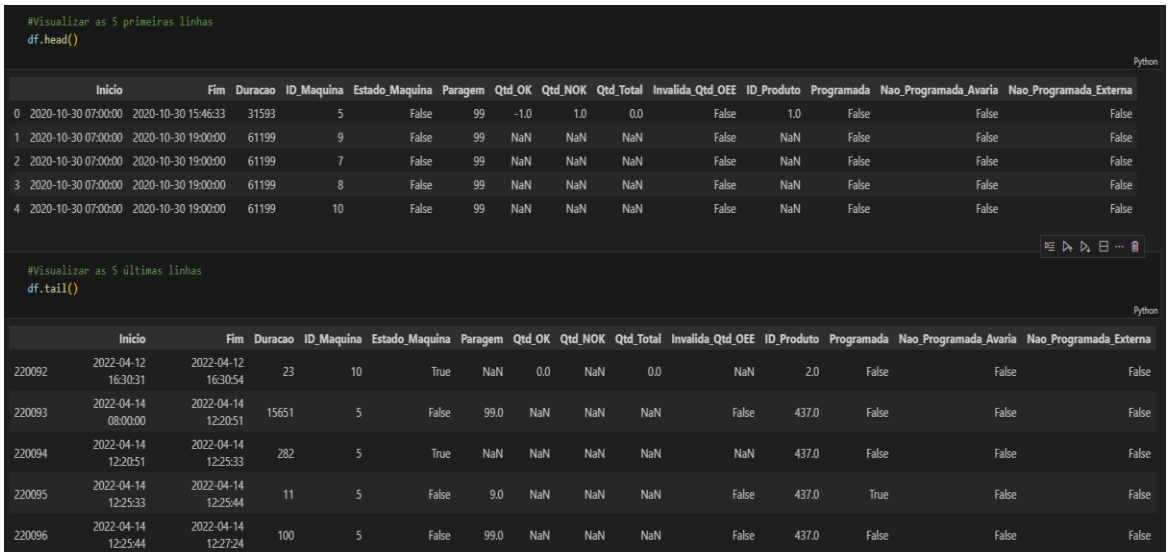
Procedeu-se, então, ao tratamento de dados, convertendo as colunas "Inicio" e "Fim" para o formato 'Datetime', como se verifica na Figura 4.3. Os formatos das restantes colunas foram mantidos até ação em contrário.

```
#Converter as colunas Inicio e Fim para Datetime
df = df.assign(Inicio = pd.to_datetime(df.Inicio), Fim = pd.to_datetime(df.Fim))
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 220097 entries, 0 to 220096
Data columns (total 14 columns):
 #   Column                 Non-Null Count   Dtype
---  ------                 --------------   -----
 0   Inicio                 220097 non-null  datetime64[ns]
 1   Fim                    220097 non-null  datetime64[ns]
 2   Duracao                220097 non-null  int64
 3   ID_Maquina             220097 non-null  int64
 4   Estado_Maquina         220097 non-null  bool
 5   Paragem                112790 non-null  object
 6   Qtd_OK                 18001 non-null   float64
 7   Qtd_NOK                34 non-null      float64
 8   Qtd_Total              18001 non-null   float64
 9   Invalida_Qtd_OEE       112789 non-null  object
 10  ID_Produto             217102 non-null  float64
 11  Programada             220097 non-null  bool
 12  Nao_Programada_Avaria  220097 non-null  bool
 13  Nao_Programada_Externa 220097 non-null  bool
dtypes: bool(4), datetime64[ns](2), float64(4), int64(2), object(2)
memory usage: 17.6+ MB
```

Figura 4.3 – Tratamento de dados

Durante a exploração dos dados em SQL, verificou-se que a base de dados era constituída por um total de 220 097 registos, entre o período de 30-10-2020 e 14-04-2022. Esta informação foi confirmada na análise realizada em Python, como se verifica na Figura 4.4.



```
#Visualizar as 5 primeiras linhas
df.head()
```

Python

	Inicio	Fim	Duracao	ID_Maquina	Estado_Maquina	Paragem	Qtd_OK	Qtd_NOK	Qtd_Total	Invalida_Qtd_OEE	ID_Produto	Programada	Nao_Programada_Avaria	Nao_Programada_Externa
0	2020-10-30 07:00:00	2020-10-30 15:46:33	31593	5	False	99	-1.0	1.0	0.0	False	1.0	False	False	False
1	2020-10-30 07:00:00	2020-10-30 19:00:00	61199	9	False	99	NaN	NaN	NaN	False	NaN	False	False	False
2	2020-10-30 07:00:00	2020-10-30 19:00:00	61199	7	False	99	NaN	NaN	NaN	False	NaN	False	False	False
3	2020-10-30 07:00:00	2020-10-30 19:00:00	61199	8	False	99	NaN	NaN	NaN	False	NaN	False	False	False
4	2020-10-30 07:00:00	2020-10-30 19:00:00	61199	10	False	99	NaN	NaN	NaN	False	NaN	False	False	False

```
#Visualizar as 5 últimas linhas
df.tail()
```

Python

	Inicio	Fim	Duracao	ID_Maquina	Estado_Maquina	Paragem	Qtd_OK	Qtd_NOK	Qtd_Total	Invalida_Qtd_OEE	ID_Produto	Programada	Nao_Programada_Avaria	Nao_Programada_Externa
220092	2022-04-12 16:30:31	2022-04-12 16:30:54	23	10	True	NaN	0.0	NaN	0.0	NaN	2.0	False	False	False
220093	2022-04-14 08:00:00	2022-04-14 12:20:51	15651	5	False	99.0	NaN	NaN	NaN	False	437.0	False	False	False
220094	2022-04-14 12:20:51	2022-04-14 12:25:33	282	5	True	NaN	NaN	NaN	NaN	NaN	437.0	False	False	False
220095	2022-04-14 12:25:33	2022-04-14 12:25:44	11	5	False	9.0	NaN	NaN	NaN	False	437.0	True	False	False
220096	2022-04-14 12:25:44	2022-04-14 12:27:24	100	5	False	99.0	NaN	NaN	NaN	False	437.0	False	False	False

Figura 4.4 – Cinco primeiros e cinco últimos registos

De seguida, verificou-se a existência de dados em duplicado e em falta, como mostra a Figura 4.5.

```
#Verificar duplicados
df.duplicated().sum()
```

```
0
```

```
#Verificar dados em falta
df.count()
```

```
Inicio                 220097
Fim                    220097
Duracao                220097
ID_Maquina             220097
Estado_Maquina         220097
Paragem                112790
Qtd_OK                  18001
Qtd_NOK                    34
Qtd_Total               18001
Invalida_Qtd_OEE       112789
ID_Produto             217102
Programada             220097
Nao_Programada_Avaria  220097
Nao_Programada_Externa 220097
dtype: int64
```

Figura 4.5 – Dados duplicados e em falta

A existência de dados em falta na coluna "Paragem", significa que a máquina não esteve parada. Nas colunas das quantidades, a falta de dados significa que não houve produção de peças. Não foram encontrados dados duplicados.

A análise das paragens foi feita de acordo com a categorização, programadas, não programadas por avaria e não programadas por causas externas. Foi realizada uma análise semelhante para cada categorização; contudo, apenas é apresentada aqui a análise das paragens programadas por avaria, por serem relevantes para a manutenção preditiva.

No estudo das paragens não programadas por avaria, foram inicialmente consideradas as paragens com os códigos ‘01’, ‘02’, ‘100’, ‘12’, ‘14’ e ‘21’. Posteriormente, a categorização das paragens ‘01’, ‘100’, ‘14’ foi atualizada conforme consta na Tabela 4.2. Na Figura 4.6 apresenta-se o gráfico de Pareto da duração das paragens com a categorização inicial.

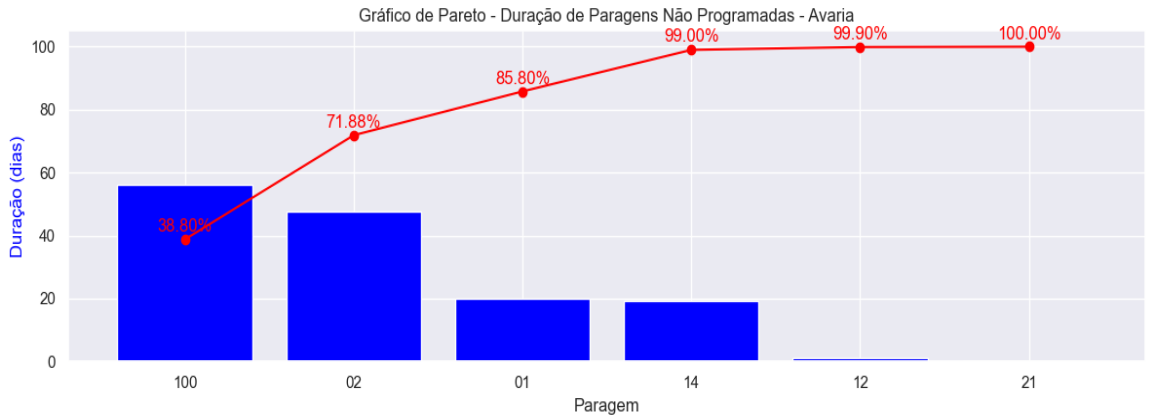


Figura 4.6 – Gráfico de Pareto da duração das paragens não programadas por avaria (categorização inicial)

Segundo o Princípio de Pareto, 80% dos resultados provêm de 20% das suas causas, permitindo destacar com objetividade as principais causas de um resultado ou evento específico, neste caso, as paragens não programadas por avaria. Tendo em conta este princípio e a Figura 4.6, podemos verificar que as paragens '100' (micro-paragem) e '02' (manutenção curativa) são responsáveis por 71.88% dos resultados. Para atingir os 80%, é

necessário considerar maior parte dos registos da paragem '01' (afinação ferramenta). Constata-se que as paragens com maior duração em dias são: '100' (micro-paragem), '02' (manutenção curativa) e '01' (afinação ferramenta), sendo que a paragem '14' (Troca de peça) também apresenta um número elevado.

Na Figura 4.7 encontra-se a análise gráfica da duração das paragens por máquina.

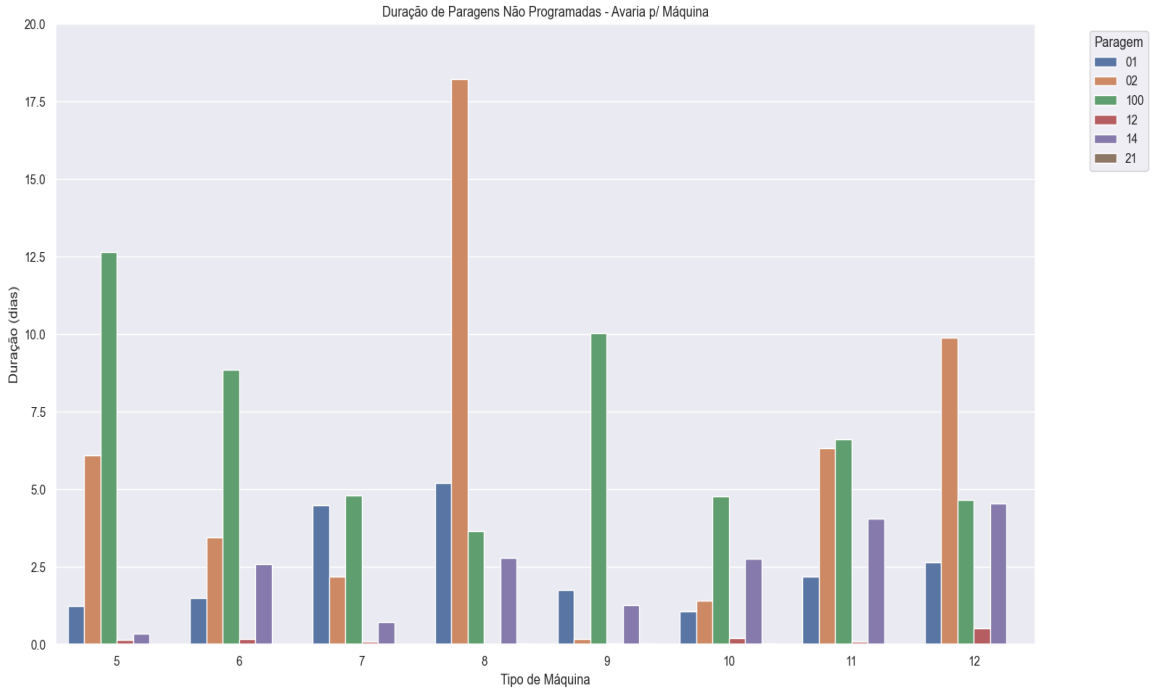


Figura 4.7 – Duração de paragens não programadas – avaria p/máquina

Das oito máquinas presentes no conjunto de dados, em seis delas a paragem '100' (micro-paragem) é a que apresenta maior duração, sendo nas restantes a paragem '02' (manutenção curativa), como demonstra a Figura 4.7. Ainda com base nesta figura, observa-se que, na maioria das máquinas analisadas, as micro-paragens são a paragem com a maior duração em dias, o que levou à necessidade de compreender melhor essa informação. Foram complementarmente obtidas as estatísticas descritivas apresentadas na Figura 4.8.

```
df.query('Paragem=="100"').describe().round(2)
```

	Inicio	Fim	Duracao
count	86717	86717	86717.00
mean	2021-09-04 09:06:56.664287232	2021-09-04 09:07:52.549384448	55.89
min	2020-11-02 08:20:06	2020-11-02 08:20:56	0.00
25%	2021-05-20 13:38:48	2021-05-20 13:40:11	10.00
50%	2021-09-01 15:55:14	2021-09-01 15:55:34	30.00
75%	2022-01-03 10:30:57	2022-01-03 10:31:47	71.00
max	2022-04-12 15:12:00	2022-04-12 15:12:22	22463.00
std	NaN	NaN	108.45

Figura 4.8 – Estatísticas das micro-paragens

Outro valor discrepante em relação às micro-paragens é o valor máximo de 22 463 segundos de duração, como demonstrado na Figura 4.8. Após validação pela pessoa responsável pelos dados, concluiu-se que as micro-paragens são catalogadas automaticamente no sistema quando a paragem dura até 120 segundos. A presença de registos de micro-paragens com valores superiores a 120 segundos resulta de erros de parametrização, tendo-se verificado a sua existência neste conjunto de dados.

Toda esta análise para as paragens não programas por avaria foi também realizada para as paragens programadas e paragens não programadas por causas externas.

4.1.3 Verificação da qualidade dos dados

Após as análises por cada tipo de categorização de paragens, analisou-se o tempo de operação por máquina através de um gráfico e de uma tabela de estatísticas, como se pode visualizar na Figura 4.9 e na Figura 4.10.

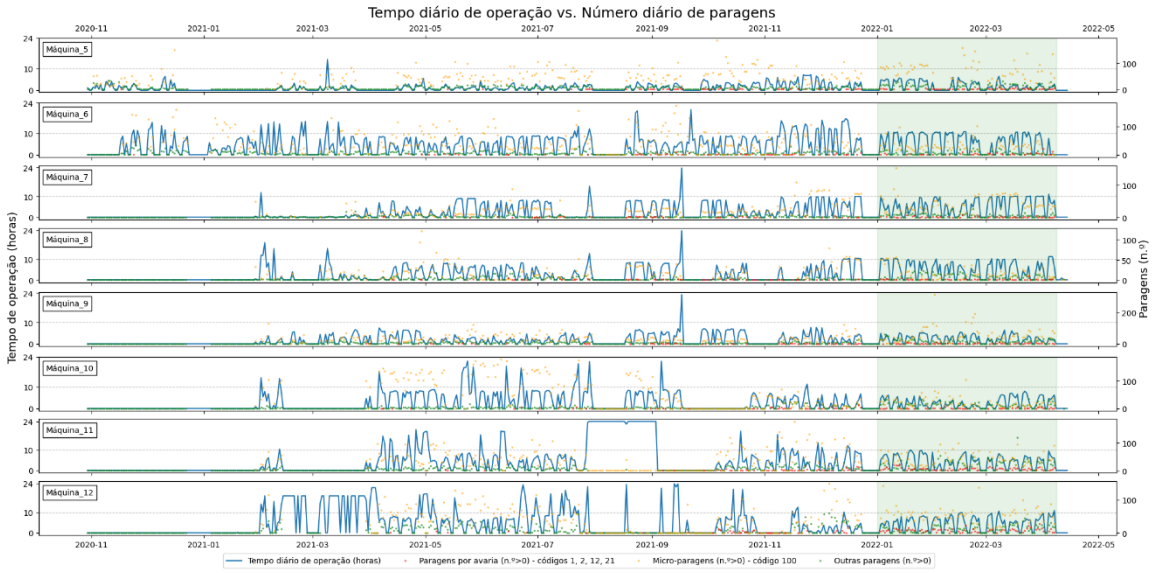


Figura 4.9 – Tempo diário de operação e número diário de paragens por máquina

Na criação do gráfico, foi necessário remover os fins de semana, uma vez que as máquinas não operam nesses dias para este cliente. Analisando o gráfico da Figura 4.9, especificamente a linha a azul que mostra a evolução das horas diárias em que as máquinas estão em funcionamento, é percetível que esses tempos são consistentemente mais regulares, em todas as máquinas, só a partir de 2022. Os dados estatísticos comprovam essa regularidade, conforme demonstrado na Figura 4.10.

	período	máquina	to_média	to_coefvar	to_máx	av_média	av_coefvar	av_máx	mp_média	mp_coefvar	mp_máx	op_média	op_coefvar	op_máx
0	2020-10-30 - 2021-12-31	5	1.58	1.12	14	0.31	2.73	7	33.67	0.95	186	7.79	0.83	34
8	2022-01-01 - 2022-04-07	5	2.80	0.68	6	1.93	1.15	10	60.22	0.68	159	15.45	0.55	34
1	2020-10-30 - 2021-12-31	6	6.03	0.73	20	0.41	2.82	9	42.76	0.88	175	7.03	0.84	33
9	2022-01-01 - 2022-04-07	6	7.82	0.42	10	1.97	1.70	21	29.04	0.67	103	7.55	0.90	33
2	2020-10-30 - 2021-12-31	7	2.91	1.30	23	0.33	2.74	6	13.68	1.45	110	3.09	0.95	19
10	2022-01-01 - 2022-04-07	7	6.94	0.47	11	1.57	1.12	8	41.09	0.64	153	8.38	0.67	27
3	2020-10-30 - 2021-12-31	8	2.94	1.30	23	0.36	2.30	6	11.70	1.49	122	3.27	1.09	17
11	2022-01-01 - 2022-04-07	8	7.04	0.45	11	1.35	1.09	9	16.68	0.96	57	10.03	0.60	23
4	2020-10-30 - 2021-12-31	9	2.24	1.17	23	0.37	2.36	5	21.87	1.12	131	6.01	0.91	28
12	2022-01-01 - 2022-04-07	9	3.18	0.47	6	2.61	0.94	11	55.09	0.93	313	20.62	0.58	63
5	2020-10-30 - 2021-12-31	10	3.51	1.31	22	0.30	3.64	12	37.68	1.38	178	3.84	1.24	27
13	2022-01-01 - 2022-04-07	10	3.47	0.63	7	1.30	1.15	6	20.84	0.88	104	16.68	0.47	34
6	2020-10-30 - 2021-12-31	11	5.24	1.39	23	0.42	3.85	15	25.53	1.35	177	5.71	1.15	31
14	2022-01-01 - 2022-04-07	11	5.78	0.44	10	7.07	0.72	23	38.93	0.60	98	25.55	0.71	119
7	2020-10-30 - 2021-12-31	12	5.22	1.27	23	0.46	3.87	16	24.66	1.37	148	8.39	1.40	60
15	2022-01-01 - 2022-04-07	12	6.42	0.35	10	8.33	0.66	28	50.25	0.52	142	15.23	0.64	45

Figura 4.10 – Tabela de dados estatísticos

Analisando a coluna 'to_coefvar', que representa o coeficiente de variação do tempo de operação, calculado pelo quociente entre o desvio-padrão e a média, percebe-se que este valor é mais baixo em todas as máquinas a partir de 2022. Isto indica que a dispersão do tempo diário de operação reduz consideravelmente neste período face ao período temporal anterior a 2022.

No gráfico, é visível que a qualidade da catalogação das paragens melhora a partir de 2022, concretamente, ao observar as contagens diárias das paragens por avaria (pontos a vermelho), micro-paragens (pontos a amarelo) e outras paragens (pontos a verde). Verifica-se que, no início dos registos, as paragens por avaria nem sequer eram assinaladas. Além disso, as micro-paragens aparentam não ter sido registadas de forma consistente antes de 2022. No entanto, é a tabela das estatísticas que evidencia a falta de consistência inicial, pois a média diária das paragens dos três tipos (colunas 'av_média', 'mp_média' e 'op_média') é, de forma geral, significativamente mais alta após o início 2022, com o respetivo coeficiente de variação tendencialmente mais baixo (colunas 'av_coefvar', 'mp_coefvar' e 'op_coefvar'). Destaca-se particularmente o registo das paragens por avaria, cuja média é claramente superior, em todas as máquinas, a partir de 2022.

A análise conjunta do gráfico e da tabela de estatísticas permite concluir que a qualidade dos dados é superior a partir do início de 2022.

Ainda em relação às paragens por avaria, é importante assinalar que todas as máquinas apresentam, em média, mais de três paragens diárias por motivo de avaria. Este facto tem fortes implicações no horizonte das previsões que é razoável definir.

Dado que o conjunto de dados, após a verificação da qualidade continha apenas 62 594 registos referentes a três meses de 2022, foi necessário obter um conjunto de dados maior e com dados mais recentes. O novo conjunto de dados disponibilizado continha 288 674 registos, entre o período de 03-01-2022 e 12-05-2023.

4.2 Preparação dos dados

Nesta secção, é abordado o pré-processamento do novo conjunto de dados que foi necessário para a criação de um modelo de previsão através do RUL, ou seja, como preparar os dados para fazer previsão, que variáveis podem ser criadas para ajudar na previsão, e termina com explicação de como foi feita a divisão dos dados em conjuntos de treino, validação e teste.

4.2.1 Pré-processamento de dados

Neste ponto do projeto, utilizando apenas os dados de três meses, foi escolhida uma única máquina para prosseguir com a análise; a máquina 8 foi selecionada por ser considerada a mais regular em operação.

Tendo em conta o objetivo de generalidade do modelo, foi necessário criar um novo *dataframe* com ciclos de operação de igual duração, ou seja, dentro do mesmo *dataframe*, teríamos vários conjuntos de ciclos de operação com início após uma paragem por avaria e fim antes da paragem por avaria seguinte. Desta forma, a previsão produzida pelo modelo corresponde ao número de horas ou ciclos até que a máquina pare por avaria. A previsão não indica, portanto, o dia e a hora exatos da paragem, mas isto não constitui uma limitação, dado que se pode assumir que a empresa conhece o momento e a duração das paragens programadas. Para a construção do *dataframe*, foram considerados ciclos de operação com uma duração de 5 minutos (300 segundos), para permitir granularidade suficiente na previsão.

Para tornar o modelo mais transversal, optou-se por trabalhar exclusivamente com variáveis numéricas relacionadas com os ciclos de operação, evitando a utilização de variáveis categóricas, uma vez que estas variáveis podem diferir de cliente para cliente. Foi necessário selecionar as variáveis a serem utlizadas no modelo e verificar a necessidade de criar novas variáveis ou transformar as existentes de formar a melhorar o desempenho do modelo. Esta criação de novas variáveis é conhecida como *feature engineering.*

Neste caso, tratando-se de um problema supervisionado de regressão para prever o RUL, era necessário que o modelo tivesse *features* e um *target/label*. As *features* são as variáveis independentes utilizadas pelo modelo, enquanto o *target/label* é a variável dependente que queremos prever. O *label* teria de ser uma variável numérica e decrescente, que permitisse saber quantos ciclos faltariam até a máquina parar. Vamos aplicar a *linear degradation* porque a relação entre a coluna *label* e as *features* é linear.

Na Figura 4.11 temos a estrutura do *dataframe* inicial constituído pelas variáveis “Inicio”, “Fim”, “Duracao”, “ID_Maquina”, “Estado_Maquina” e a “Paragem”.

	start	end	duration	machine_id	machine_state	stop_id
0	2022-01-03 06:00:00	2022-01-03 06:01:46	106	11	False	09
1	2022-01-03 06:00:00	2022-01-03 06:06:53	413	6	False	09
2	2022-01-03 06:00:00	2022-01-03 06:07:46	466	8	False	09
3	2022-01-03 06:00:00	2022-01-03 06:09:16	556	7	False	01
4	2022-01-03 06:01:46	2022-01-03 06:01:56	10	11	True	None

Figura 4.11 – *Dataframe* inicial

Para a criação do *dataframe* constituído por ciclos, foram consideradas apenas as seguintes variáveis iniciais, a “Duracao” e a “Paragem”. O código para a criação dos ciclos foi desenvolvido utilizando dados provisórios, de forma a facilitar o tratamento de erros que fossem surgindo; só depois de o código estar solidado é que foi aplicado ao conjunto de dados reais. O código completo encontra-se no APÊNDICE 3 e a Figura 4.12 apresenta a última função desse código, que contém os parâmetros para a criação dos ciclos.

```python
#This function returns a tuple of values
def parameters():
    data_machine = df.loc[df["machine_id"] == 8].reset_index(drop=True)
    cycle_duration = 300
    stop_ids = ['02','12','21']
    prog_stop_ids = ['01','03','05','11','14','15','16','17','22','25','26','95','99']
    specific_stop_ids = ['09','10','13','18']
    ext_stop_ids = ['04','06','07','08','20','23','24']

    return data_machine, cycle_duration, stop_ids, prog_stop_ids, specific_stop_ids, ext_stop_ids

result = create_cycle_dataframe(*parameters()) # '*parameters()' is a syntax that unpacks the values
from the tuple
```

Figura 4.12 – Função com os parâmetros dos ciclos

Explicando o código da Figura 4.12, como se pretendia que o modelo fosse geral, considerou-se importante que os parâmetros necessários para a criação dos ciclos fossem de fácil acesso, por isso, colocaram-se dentro de uma função. No primeiro parâmetro "data_machine", seleciona-se a máquina que se pretende utilizar; o segundo parâmetro "cycle_duration" define a duração dos ciclos, podendo ser ajustado conforme necessário; o terceiro parâmetro "stop_ids" contém as paragens programadas por avaria, para que o código crie ciclos após essas paragens e pare de criar ciclos antes delas; no quarto parâmetro "prog_stop_ids" encontram-se as paragens programadas; no quinto parâmetro "specific_stop_ids" inclui-se paragens específicas que possam ser relevantes para análise e no último parâmetro "ext_stop_ids" encontram-se as paragens não programadas por causas externas.

Entre as paragens por avaria, ocorrem paragens programadas e externas, sendo essencial ter essa informação acessível, por isso, criaram-se colunas com as contagens de cada tipo de paragem, conforme se verifica no dicionário das variáveis na Tabela 4.3.

Tabela 4.3 - Dicionários das variáveis do dataframe com ciclos de operação

Designação variável	**Descrição da variável**
"stop_number"	Ordem de determinado conjunto de ciclos
"cycle"	Ciclo de operação pertencente ao respetivo "stop_number"
"cum_duration"	Duração acumulada de operação
"cum_stop_100_count"	Contagem acumulada das micro-paragens
"cum_stop_100_duration"	Duração acumulado das micro-paragens
"prog_stop_count"	Contagem das paragens programadas
"spec_stop_count"	Contagem de paragens especificas selecionadas
"ext_stop_count"	Contagem das paragens por avaria causas externas
"RUL"	Indica os ciclos de operação até próxima falha por avaria

A execução do código completo, que se encontra no APÊNDICE 3, resulta na construção de um *dataframe* com ciclos de operação de igual duração, apresentando informações relevantes previamente selecionadas, como se pode verificar na Figura 4.13.

	stop_number	cycle	cum_duration	cum_stop_100_count	cum_stop_100_duration	prog_stop_count	spec_stop_count	ext_stop_count	RUL
0	1	1	300	0.0	0	0	0	0	115
1	1	2	600	0.0	0	0	0	0	114
2	1	3	900	1.0	10	0	0	0	113
3	1	4	1200	2.0	30	0	0	0	112
4	1	5	1500	2.0	30	0	0	0	111

Figura 4.13 – Primeiros registos do *dataframe* com ciclos

Como já referido, as micro-paragens fazem parte do processo de produção e, portanto, foram consideradas como tempo de produção. Assim, os ciclos de operação são constituídos por tempo de operação e micro-paragens. Por exemplo, se uma micro-paragem dura 700 segundos, esta pode ser dividida entre vários ciclos: 179 segundos num ciclo, 300 seguntos no seguinte, e assim por diante. Na contagem, considera-se o resultado da divisão, como (179/700), (300/700), seguindo este mesmo raciocínio. Para que esta informação não fosse perdida, foi necessário criar uma coluna que indicasse as contagens e duração acumulada das micro paragens. Existe tempo de operação que não é suficiente para completar um ciclo de operação e, portanto, não é contabilizado. Para resolver esse problema, o RUL termina sempre em 1 e não em 0, para ter um ciclo de operação como margem de erro.

O *dataframe* inicial, resultante da execução do código com os ciclos da máquina 8, era constituído por 142 "stop_number" num total de 32 800 ciclos de operação. Cada "stop_number" corresponde a um conjunto de ciclos de operação identificado pela coluna "cycle". Foi necessária uma limpeza de dados: cada "stop_number" deveria ter um mínimo de 12 ciclos (1 hora de operação). O primeiro e o último "stop_number" foram removidos porque o conjunto de dados não começava nem terminava com uma paragem por avaria. A numeração do "stop_number" foi atualizada para não haver números em falta. O *dataframe* final era constituído por 97 "stop_number" num total de 32 582 ciclos de operação. Graficamente, a variabilidade do RUL por "stop_number" pode ser observada na Figura 4.14.

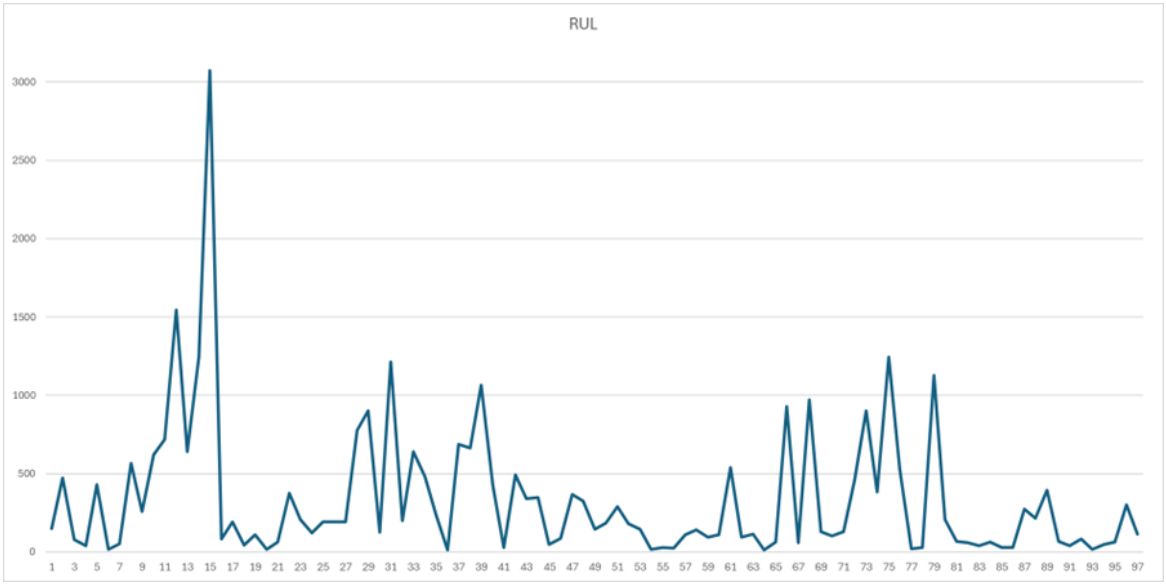


Figura 4.14 – RUL por stop-number

4.2.2 Divisão dos dados em conjuntos de treino, validação e teste

Estes é um modelo global de previsão, denominado *Global Forecasting Model* (GFM), que é treinado com todos os dados disponíveis no conjunto de dados, ou seja, dados de várias séries temporais da mesma máquina.

Nesta etapa, procedeu-se à divisão do conjunto de dados em conjuntos de treino, validação e teste. Para as experiências seguintes, não era necessário utilizar o conjunto de validação, mas a divisão foi feita de maneira que o modelo já estivesse preparado para a fase da validação cruzada.

O conjunto de treino é utilizado para treinar o modelo, o conjunto de validação ajuda a ajustar os hiperparâmetros, e o conjunto de teste permite avalia o desempenho do modelo em dados desconhecidos. Não foi possível utilizar a função *train_test_split* do *scikit-learn*, pois é essencial manter a sequência temporal dos dados, sem aleatoriedade. Assim, para dividir os dados, foi necessária uma abordagem diferente. A divisão foi feita utilizando a coluna 'stop_number', aplicando um rácio de 70% para o treino, 15% para a validação e os restantes 15% para teste, como se pode verificar na primeira linha de código da Figura 4.15.

```python
def split_data(df, train_ratio=0.7, valid_ratio=0.15):
    # Get unique stop_numbers
    stop_numbers = df['stop_number'].unique()

    # Calculate the indices for splitting
    total_stop_numbers = len(stop_numbers)
    train_stop_numbers = int(train_ratio * total_stop_numbers)
    valid_stop_numbers = int(valid_ratio * total_stop_numbers)

    # Get the indices for train, valid, and test sets
    train_indices = stop_numbers[:train_stop_numbers]
    valid_indices = stop_numbers[train_stop_numbers:train_stop_numbers + valid_stop_numbers]
    test_indices = stop_numbers[train_stop_numbers + valid_stop_numbers:]

    # Create the training, validation, and test sets based on 'stop_number'
    train = df[df['stop_number'].isin(train_indices)]
    valid = df[df['stop_number'].isin(valid_indices)]
    test = df[df['stop_number'].isin(test_indices)]

    # Display the lengths of the splits
    print("Training set length:", len(train))
    print("Validation set length:", len(valid))
    print("Test set length:", len(test))

    return train, valid, test

train, valid, test = split_data(new_df_cycles)
```

✓ 0.0s

```
Training set length: 24448
Validation set length: 6297
Test set length: 1837
```

Figura 4.15 – Função Python para a divisão dos dados em treino, validação e teste

De seguida, foi necessário separar as variáveis da coluna *label*, neste caso o RUL, que foi feito da seguinte forma como mostra a Figura 4.16.

```python
def prepare_data(train, valid, test):
    # X contains only features
    X_train = train.drop(['RUL'], axis=1)
    X_valid = valid.drop(['RUL'], axis=1)
    X_test = test.drop(['RUL'], axis=1)

    # y contains only the label
    y_train = train[['RUL']]
    y_valid = valid[['RUL']]
    y_test = test[['RUL']]

    # Display the lengths of the splits
    print("Training set length:", len(y_train))
    print("Validation set length:", len(y_valid))
    print("Test set length:", len(y_test))

    return X_train, X_valid, X_test, y_train, y_valid, y_test

X_train, X_valid, X_test, y_train, y_valid, y_test = prepare_data(train, valid, test)
```

✓ 0.0s

```
Training set length: 24448
Validation set length: 6297
Test set length: 1837
```

Figura 4.16 – Função em Python da separação das *features* da *label*

4.3 Modelação

Nesta secção, são apresentados os resultados de experiências com os algoritmos escolhidos para a criação dos modelos e análise dos resultados de ambos, terminando com a aplicação da validação cruzada.

4.3.1 Experiências preparatórios com os algoritmos selecionados

Atendendo à revisão de literatura, os algoritmos considerados foram o RF Regressor e XGBoost Regressor, por apresentarem bons resultados e não obrigam a grande processamento. Estes algoritmos baseados em árvores de decisão de regressão, permitem então prever valores numéricos contínuos.

Antes de se proceder à análise do desempenho dos algoritmos, é importante referir que foram realizadas várias experiências com ambos os algoritmos para ganhar sensibilidade do seu funcionamento. Dado o tempo disponível ser limitado, a validação cruzada e o ajuste de hiperparâmetros foram aplicados apenas ao algoritmo que apresentou os melhores resultados. Tendo em conta esta restrição, focou-se também em otimizar apenas os hiperparâmetros que controlam o sobreajuste. Esses hiperparâmetros foram os seguintes:

- "N_estimators": Refere-se ao número de árvores a serem treinadas. Tipicamente, um número muito pequeno de árvores leva a *underfitting*, enquanto um número muito grande leva a *overfitting*.
- "learning_rate": Este parâmetro controla a velocidade de ajuste e ajuda a combater o *overfitting.* Funciona como um fator de ponderação para as correções realizadas por novas árvores. Uma taxa de aprendizagem mais baixa significa que é dada menor importância às correções das novas árvores, evitando, assim, o *overfitting*. Os valores típicos para este parâmetro variam entre 0,1 e 0,3.
- "max_depth": Representa a profundidade máxima entre o nó raiz da árvore e o nó folha. É um dos parâmetros mais importantes para controlar o *overfitting.* O intervalo típico de valores é de 3 a 10, mas raramente é necessário ultrapassar 5 a 7. Além disso, utilizar árvores mais profundas torna o XGBoost extremamente consumidor de memória.
- "min_child_weight": Controla a soma dos pesos de todas as amostras nos dados ao criar um novo nó. Quando este valor é baixo, cada nó agrupa um número cada vez menor de amostras, aumentando a probabilidade de *overfitting* devido às especificidades dos dados de treino. Por isso, deve ser escolhido um valor alto para evitar *overfitting.* O valor padrão é 1, sendo o valor máximo limitado apenas pelo número de linhas nos dados de treino. No entanto, um intervalo adequado para ajuste está entre 2 e 10 ou até 20.

Na Figura 4.17, encontram-se os resultados obtidos pelo modelo aplicando o RF. Importante referir que os hiperparâmetros selecionados e respetivos valores resultaram das experiências realizadas com este algoritmo e da investigação de outros projetos. Foram testados diferentes valores para cada hiperparâmetro, sendo que os valores a seguir apresentados proporcionaram os melhores resultados.

```
Random Forest Regressor Hyperparameters:
max_depth: 6
max_features: None
min_samples_leaf: 25
n_estimators: 50
n_jobs: -1
random_state: 42
Training Execution Time: 0.9797 seconds

train set RMSE:118.0572, MAE:84.9196, MAPE:1.8483, R2:0.9592
Train Prediction Execution Time: 0.0282 seconds

valid set RMSE:486.1052, MAE:355.6159, MAPE:1.116, R2:-1.2581
Valid Prediction Execution Time: 0.0147 seconds

test set RMSE:128.7227, MAE:90.4806, MAPE:1.3636, R2:-0.6934
Test Prediction Execution Time: 0.0158 seconds
```

Figura 4.17 – Desempenho do RF

Analisando o desempenho do RF, apresentado na Figura 4.17, observa-se que os resultados do conjunto de treino indicam que o modelo teve um bom desempenho, com valores baixos de RMSE e MAE, além de um valor de R-quadrado elevado (próximo de 1), o que sugere um bom ajuste aos dados de treino. No entanto, os resultados dos conjuntos de validação e teste levantam preocupações. Os valores elevados de RMSE e MAE, juntamente com valores de R-quadrado negativos, indicam que o modelo não está a generalizar bem para dados não vistos, podendo estar a sofrer de *overfitting* ou enfrentar problemas com a distribuição dos dados nos conjuntos de validação e teste. Conclui-se, portanto, que o bom ajuste no treino representa, na verdade, um sobreajuste.

Analisando ainda Figura 4.17, os valores superiores a 1 de MAPE nos três conjuntos são bastante preocupantes, pois indicam uma grande discrepância entre as previsões e os valores observados, sugerindo que o modelo pode não estar a prever com precisão e que há um erro elevado nas previsões em relação aos valores reais. O valor MAE, convertido em tempo, corresponde a 7 horas e 30 minutos. Observando graficamente os resultados para o conjunto de teste, através da Figura 4.18, percebe-se que o modelo não está a captar corretamente a tendência de diminuição do RUL.

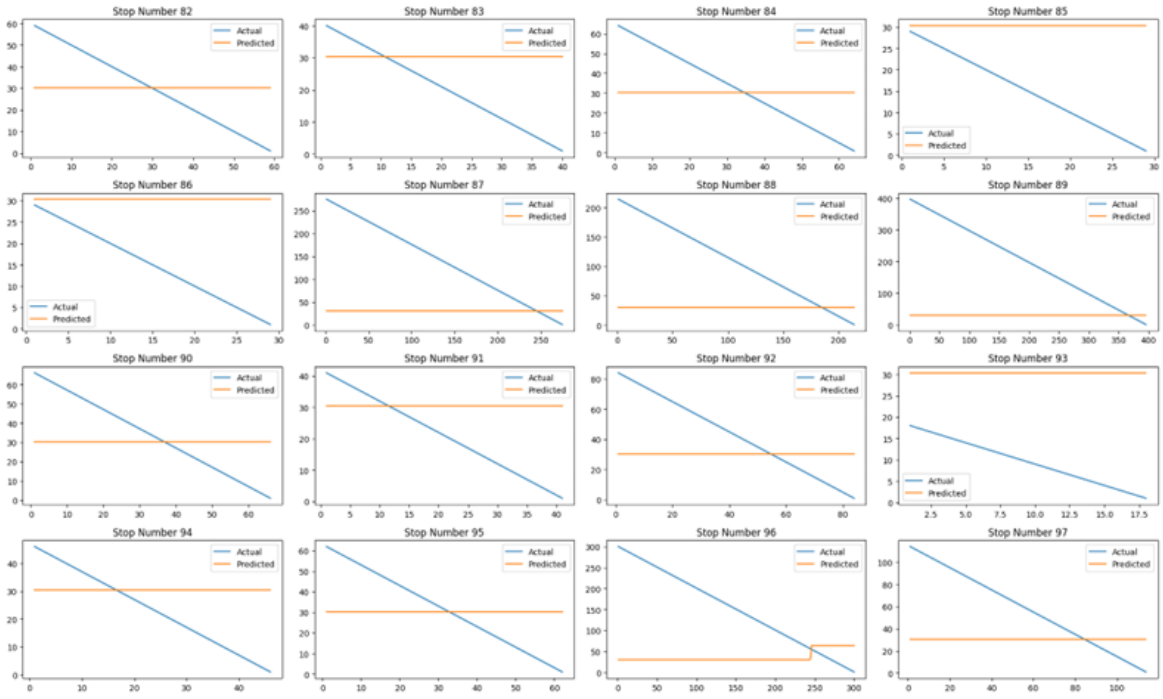


Figura 4.18 - Valor atual vs valor previsto do RUL no teste - RF

À semelhança das experiências realizadas com o RF, procedeu-se da mesma forma com algoritmo XGBoost. Na Figura 4.19 encontram-se os resultados do modelo aplicando o XGBoost.

```
booster: gbtree
gamma: 80
learning_rate: 0.03
max_depth: 6
min_child_weight: 16
missing: nan
n_estimators: 100
n_jobs: -1
random_state: 42
tree_method: hist
Training Execution Time: 0.6323 seconds

train set RMSE:65.9888, MAE:50.9281, MAPE:1.4213, R2:0.9873
Train Prediction Execution Time: 0.0269 seconds

valid set RMSE:403.8589, MAE:292.1742, MAPE:2.1034, R2:-0.5586
Valid Prediction Execution Time: 0.0129 seconds

test set RMSE:108.1309, MAE:93.3327, MAPE:5.4566, R2:-0.1949
Test Prediction Execution Time: 0.005 seconds
```

Figura 4.19 – Desempenho do XGBoost

Analisando os resultados do desempenho do RF, apresentados na Figura 4.19, os dados de treino indicam que o modelo alcançou uma boa performance, refletida por valores baixos de RMSE e MAE e um valor de R-quadrado elevado (próximo de 1), sugerindo um ajuste adequado aos dados de treino. No entanto, os resultados nos conjuntos de validação mostram uma diminuição no desempenho, enquanto no conjunto de teste a situação se inverte, com melhorias em várias métricas. Ainda assim, essas métricas continuam a apresentar valores superiores aos do conjunto de treino, o que pode apontar para um possível sobreajuste.

Além disso, os valores de MAPE superiores a 1 nos três conjuntos permanecem preocupantes, indicando uma significativa discrepância entre as previsões e os valores reais, o que sugere um nível elevado de erro nas previsões. A análise gráfica dos resultados para o conjunto de teste que se encontra na Figura 4.20 revela que o modelo ainda não consegue capturar a tendência decrescente esperada no RUL, evidenciando a dificuldade em prever corretamente a redução progressiva deste indicador.

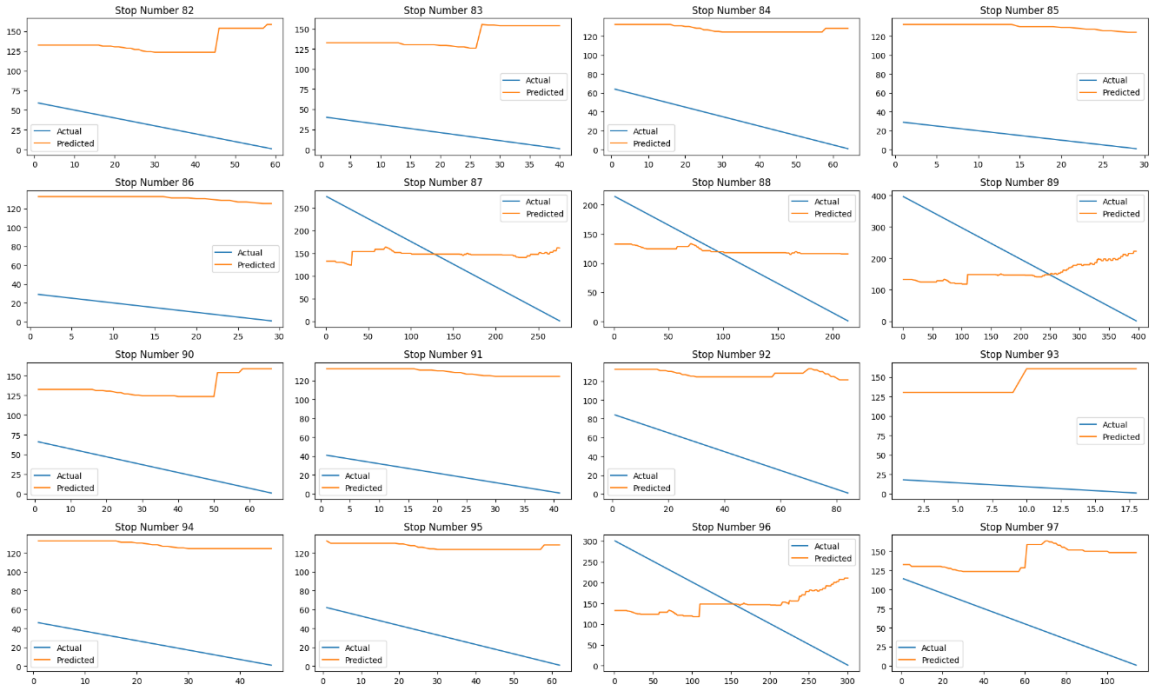


Figura 4.20 – Valor atual vs valor previsto do RUL no teste - XGBoost

Para melhorar o desempenho dos algoritmos, foram realizadas várias experiências. No caso do RF, testou-se o modelo utilizando apenas variáveis de contagem (“cum_stop_100”, “prog_stop_count”, “spec_stop_count”, “ext_stop_count”) e, posteriormente, apenas variáveis de duração (“cum_duration”, e “cum_stop_100_duration”), conforme listadas na Tabela 4.3.

Para ambos os algoritmos, foram também testadas diferentes abordagens, incluindo a alteração da percentagem de divisão dos conjuntos, treino, validação e teste; a divisão dos conjuntos de treino, validação e teste por ordem e de forma aleatória; a aplicação do *piecewise linear degradation* definindo um valor inicial para o RUL; e a alteração da variabilidade do RUL utilizando valores máximos e mínimos. Nessa última experiência, onde o *dataframe* dos ciclos continha apenas “stop_number” com um mínimo e máximo de ciclos de operação, observaram-se melhorias significativas no XGBoost. Reduzindo a variabilidade do RUL para um mínimo de 250 e um máximo de 800 ciclos, o modelo apresentou resultados superiores, como ilustrado na Figura 4.21.

```
booster: gbtree
gamma: 80
learning_rate: 0.03
max_depth: 6
min_child_weight: 16
missing: nan
n_estimators: 100
n_jobs: -1
random_state: 42
tree_method: hist
Training Execution Time: 0.4071 seconds

train set RMSE:20.0912, MAE:15.5045, MAPE:0.3449, R2:0.9877
Train Prediction Execution Time: 0.0127 seconds

valid set RMSE:102.7759, MAE:89.167, MAPE:0.8883, R2:0.4301
Valid Prediction Execution Time: 0.0051 seconds

test set RMSE:84.9993, MAE:67.092, MAPE:0.8781, R2:0.5461
Test Prediction Execution Time: 0.0049 seconds
```

Figura 4.21 – Desempenho do XGBoost com redução de variabilidade RUL

Analisando os resultados apresentados na Figura 4.21, o conjunto de treino indica que o modelo alcançou um desempenho muito bom, com valores baixos de RMSE, MAE e de MAPE, além de um valor de R-quadrado elevado (próximo de 1), sugerindo um ajuste aos dados de treino.

Mais uma vez, os resultados dos conjuntos de validação e teste levantam preocupações. A aumento dos valores de RMSE, MAPE e MAE, juntamente com a diminuição do R-quadrado, indicam que o modelo não está a generalizar bem para dados não vistos. Pode estar a sofrer de *overfitting* ou a enfrentar problemas com a distribuição dos dados nos conjuntos de validação e teste.

Neste caso, o MAE é 5 horas e 35 minutos, uma redução significativa comparativamente com o RF. A análise gráfica dos resultados do conjunto de teste, através da Figura 4.22, revela que modelo já foi capaz de entender que o RUL deve ser decrescente.

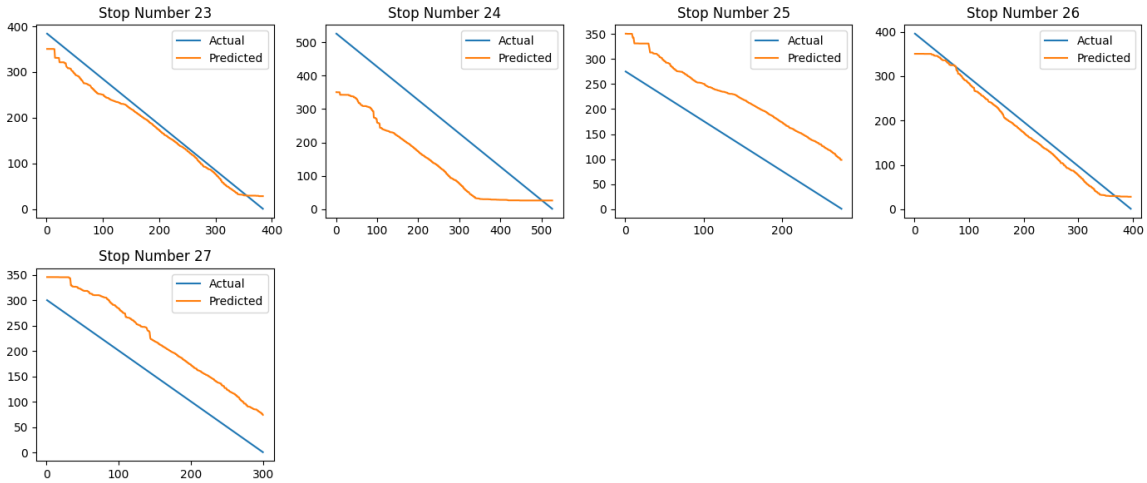


Figura 4.22 – Valor atual vs valor previsto do RUL no teste – XGBoost com variabilidade reduzida

Podemos confirmar que um dos problemas com estes dados é a elevada variabilidade do RUL. Quando os valores de RUL são mais consistentes, o modelo base apresenta um desempenho muito melhor, uma vez que os valores elevados de RUL pertencem ao "stop_number" no conjunto de treino.

Em síntese, reduzir a variabilidade do RUL permite uma melhor distribuição nos conjuntos de validação e teste, resultando em métricas significativamente melhores. Ao observar os resultados graficamente na Figura 4.22, percebe-se que o modelo compreende que o RUL deve ser decrescente. Esta análise revela também problemas sérios com os dados: a variabilidade excessiva do RUL não é normal, e por outro lado, nota-se a inexistência de dados de sensores.

4.3.2 Validação cruzada

Após a obtenção dos resultados do melhor algoritmo o XGBoost para uma variabilidade do RUL mínimo de 250 e um máximo de 800 ciclos, o *dataframe* ciclos a ser utlizado daqui em frente, passou a ser constituído por 27 “stop_number” num total de 12 800 ciclos de operação.

A validação cruzada, conhecida como *cross-validation*, é uma técnica utilizada para avaliar a performance e a robustez de um modelo de ML, particularmente sua capacidade de generalizar para dados novos. Como estamos a trabalhar com dados temporais, a validação cruzada não pode ser feita de forma aleatório, foi preciso criar os *folds* a serem passados ao modelo. Neste caso em concreto, os *folds* foram criados filtrando os ciclos de operação por "stop_number" até não haver dados sobrantes. Na Figura 4.23, está representado graficamente a distribuição do "stop_number" por cada *fold.*

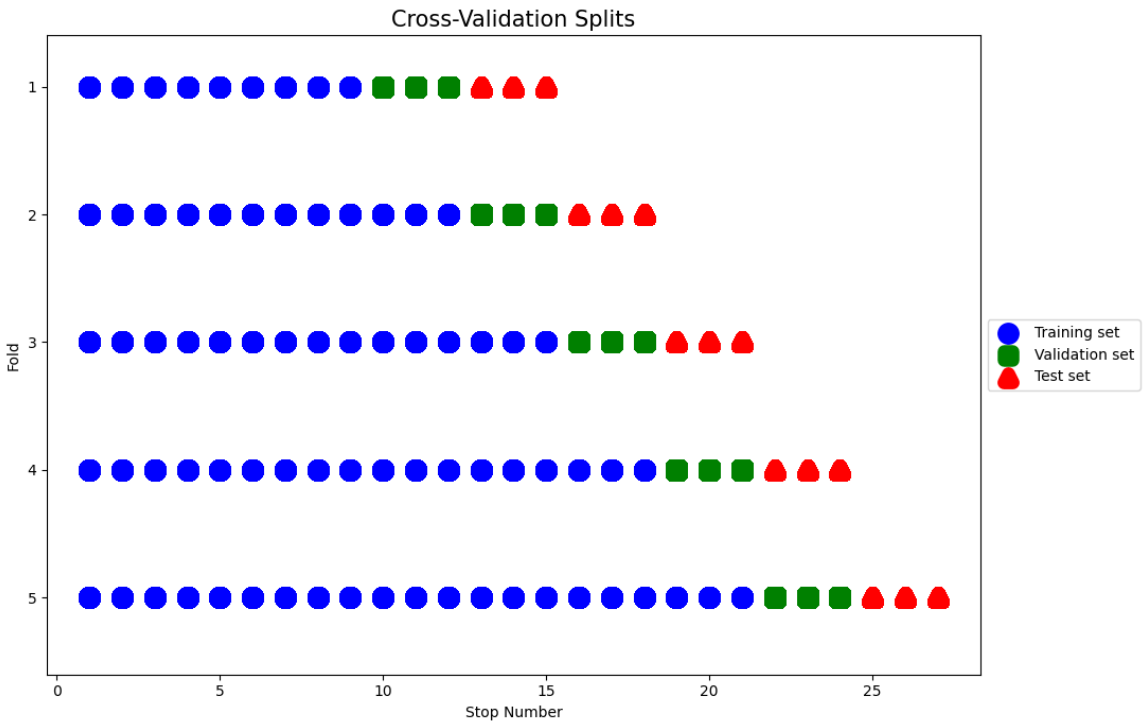


Figura 4.23 – K-*fold* para a validação cruzada

O *fold 1* é constituído por quinze "stop_number", divididos da seguinte forma: os primeiros nove "stop_number" para treino, os três seguintes para validação e os três seguintes para teste. O *fold 2* é constituído por dezoito "stop_number", com os os doze primeiros para treino, os três seguintes para validação e os três seguintes para teste. Nos *folds* seguintes, são sempre introduzidos três novos "stop_number" que serão destinados ao conjunto de teste, num total de cinco folds. O *fold 5* inclui a totalidade dos vinte e sete "stop_number",

sendo os primeiros vinte e um para treino, os três seguintes para validação e os três últimos para teste.

Na Figura 4.24 temos os resultados do modelo aplicando a validação cruzada.

```
Fold 1:
Training Execution Time: 0.2119 seconds

train set RMSE:8.6723, MAE:6.8608, MAPE:0.1818, R2:0.9979
valid set RMSE:124.3629, MAE:105.8996, MAPE:1.5449, R2:0.546
test set RMSE:209.7862, MAE:180.6485, MAPE:2.282, R2:-0.5525

Fold 2:
Training Execution Time: 0.202 seconds

train set RMSE:12.7054, MAE:9.9446, MAPE:0.2332, R2:0.9954
valid set RMSE:127.1533, MAE:104.8098, MAPE:1.5105, R2:0.4297
test set RMSE:343.7234, MAE:333.7269, MAPE:6.8776, R2:-10.3404

Fold 3:
Training Execution Time: 0.1921 seconds

train set RMSE:15.9264, MAE:12.586, MAPE:0.2715, R2:0.9926
valid set RMSE:218.6013, MAE:206.5358, MAPE:4.8331, R2:-3.5868
test set RMSE:203.3613, MAE:180.0021, MAPE:4.4785, R2:-1.1968

Fold 4:
Training Execution Time: 0.2172 seconds

train set RMSE:14.9077, MAE:11.4147, MAPE:0.2483, R2:0.9932
valid set RMSE:111.0707, MAE:95.1617, MAPE:0.995, R2:0.3447
test set RMSE:97.2808, MAE:80.0005, MAPE:0.4892, R2:0.5016

Fold 5:
Training Execution Time: 0.2316 seconds

train set RMSE:15.909, MAE:12.5099, MAPE:0.2997, R2:0.992
valid set RMSE:90.3063, MAE:71.3527, MAPE:1.2429, R2:0.5705
test set RMSE:207.3205, MAE:200.7922, MAPE:4.027, R2:-3.2313
```

Figura 4.24 – Resultados da validação cruzada

Analisando os resultados da Figura 4.24, observa-se que, em todos os *folds*, o modelo apresenta um desempenho muito bom no conjunto de treino, com valores baixos de RMSE, MAPE e MAE e valores elevados de R-quadrado (próximos de 1). Isso sugere que o modelo está a ajustar-se bem aos dados de treino. Nos conjuntos de validação, já levanta

algumas preocupações, verifica-se uma variação considerável nos resultados entre os *folds*. Embora alguns *folds* apresentem um R-quadrado positivo no conjunto de validação, como o *fold* 1 (R2 = 0.546) e *fold* 5 (R2 = 0.5705), outros têm valores de R-quadrado negativos, como o fold 3 (R2 = -3.5868). Esse comportamento sugere que o modelo não está a generalizar bem para dados não vistos e pode estar a sofrer de sobreajuste.

O desempenho no conjunto de teste é ainda mais preocupante, com valores de R-quadrado frequentemente negativos e altos valores de RMSE e MAE em todos os *folds* (por exemplo, *fold 2* com R2 = -10.3404 e *fold* 3 com R2 = -1.1968). Isso indica que o modelo não consegue generalizar para dados realmente não vistos e sugere problemas de adaptação entre os conjuntos de treino e teste, possivelmente devido a diferenças na distribuição dos dados.

A variação nos resultados entre os diferentes *folds,* especialmente no conjunto de teste, indicam que o modelo pode ser sensível à divisão dos dados. A consistência é baixa, o que implica que o modelo pode não ser robusto o suficiente para fazer previsões fora dos dados de treino, evidenciando ainda mais que a variabilidade do RUL é um grave problema.

4.4 Avaliação

Na última secção deste capítulo, são apresentados os ajustes de hiperparâmetros que foram aplicados, juntamento com os resultados e conclusões obtidas a partir do modelo ajustado.

4.4.1 Ajuste de hiperparâmetros

O ajuste de hiperparâmetros, conhecido como *hyperparameter tuning*, foi aplicado utilizando o GridSearchCV, que procura pelos melhores valores para cada parâmetro do modelo, aplicando validação cruzada de forma aleatório. Por essa razão, foi necessário passar os *k-folds* pretendidos, para preservar ordem temporal (criados na subsecção 4.3.2).

Conforme mencionado no início da subsecção 4.3.1, devido à limitação de tempo, o GridSearchCV foi aplicado apenas ao algoritmo XGBoost para uma variabilidade do RUL entre um mínimo de 250 e um máximo de 800 ciclos.

Na Figura 4.25 podem-se verificar os hiperparâmetros e os respetivos intervalos de valores.

```
# Define the hyperparameter grid
param_grid = {
    'n_estimators': [50, 100, 150],
    'learning_rate': [0.003, 0.03, 0.3],
    'max_depth': [4, 6, 8],
    'min_child_weight': [6, 12, 16],
}

# Create an instance of XGBRegressor
xgb_model = xgb.XGBRegressor(
    gamma=80,
    tree_method='hist',
    booster='gbtree',
    objective ='reg:squarederror',
    n_jobs = -1,
    random_state=42)
```

Figura 4.25 – Intervalos valores para cada parâmetro

Os hiperparâmetros selecionados para serem otimizados são aqueles focados no controlo do sobreajuste, e os intervalos definidos foram escolhidos com base nas experiências mencionadas na subsecção 4.3.1.

4.4.2 Resultados

Para os possíveis valores de cada parâmetro do modelo presente no Figura 4.25, o GridSearchCV escolheu os valores que se encontram na Figura 4.26.

```
Best hyperparameters: {'learning_rate': 0.003, 'max_depth': 8, 'min_child_weight': 6, 'n_estimators': 100}
Train set RMSE: 136.5234, MAE: 113.1537, MAPE: 226.3894, R2: 0.4327
Test set RMSE: 119.3388, MAE: 101.4588, MAPE: 305.2229, R2: 0.1052
```

Figura 4.26 – Resultados do GridSearchCV

Segundo os resultados do GridSearchCV, a melhor combinação de hiperparâmetros encontrado para o modelo XGBoost é:

- 'learning_rate' = 0.003: uma taxa de aprendizagem baixa que permite ajustes graduais, reduzindo o risco de sobreajuste.
- 'max_depth' = 8: profundidade máxima das árvores, permitindo ao modelo captar padrões complexos sem se tornar excessivamente complexo.
- 'min_child_weight' = 6: exige um peso mínimo para criar novos nós, ajudando o modelo a evitar ajustes excessivos.
- 'n_estimators' = 100: o modelo tem capacidade suficiente para capturar padrões nos dados, mantendo um equilíbrio entre complexidade e capacidade de generalização.

Analisando os resultados do desempenho do modelo após o ajuste de hiperparâmetros, presentes na Figura 4.26, e comparando com o modelo base, é evidente que o ajuste não trouxe melhorias. Na verdade, observam-se resultados significativamente inferiores, especialmente no que respeita à capacidade de generalização e precisão.

Para o modelo base, os dados de treino apresentavam valores baixos de RMSE e MAE, e um valor elevado de R-quadrado (0.9873), o que indicava um bom ajuste e uma boa capacidade de captar padrões nos dados de treino. Contudo, mesmo no modelo base, os conjuntos de validação e teste já evidenciavam dificuldades de generalização, com valores elevados de RMSE, MAE e um R-quadrado negativo. Apesar destas limitações, o modelo base ainda demonstrava uma capacidade moderada de generalização, apresentando um desempenho aceitável nos dados de teste.

Após o ajuste de hiperparâmetros com GridSearchCV, o desempenho do modelo piorou substancialmente. No conjunto de treino, o R-quadrado caiu de 0.9873 para 0.4327, e tanto o RMSE como o MAE aumentaram drasticamente. Este resultado indica que o modelo ajustado perdeu a capacidade de se ajustar adequadamente aos dados de treino. No conjunto de teste, o valor de R-quadrado caiu ainda mais, para 0.1052, e os valores de RMSE e MAE permaneceram elevados. Este valor de R-quadrado próximo de zero sugere que o modelo ajustado tem uma capacidade muito limitada de explicar a variabilidade dos dados de teste, mostrando dificuldades graves em prever adequadamente novos dados.

Além disso, os valores de MAPE superiores a 1 nos três conjuntos são especialmente preocupantes. Estes valores de MAPE indicam uma discrepância significativa entre as previsões e os valores reais, evidenciando um erro percentual elevado tanto no conjunto de treino como no conjunto de teste. Enquanto o modelo base tinha um MAPE mais baixo, o modelo ajustado apresentou valores de MAPE extremamente altos, sugerindo uma perda significativa de precisão.

Comparando o modelo base com o modelo ajustado, é claro que o ajuste de hiperparâmetros com GridSearchCV não resultou numa melhoria de desempenho. Pelo contrário, o ajuste tornou o modelo mais complexo e instável, levando a um pior desempenho. Isto pode ser causado pela alta variabilidade dos dados ou por uma incompatibilidade dos hiperparâmetros otimizados com as características do conjunto de dados, possivelmente indicando sobreajuste nos dados de treino ou uma insuficiência de dados de qualidade para suportar o ajuste do modelo.

5 CONCLUSÃO

Neste último capítulo, é apresentado um sumário do trabalho desenvolvido, as limitações encontradas e superadas ao longo do projeto, uma síntese dos objetivos alcançados, sugestões para trabalhos futuros e algumas considerações finais.

5.1 Sumário do trabalho desenvolvido

Foi desenvolvido um modelo preditivo capaz de prever paragens por avaria, prevendo o RUL. Para isso, foi necessário passar por várias etapas, tendo começado pelo entendimento dos dados. Nesta etapa, foi criado um dicionário de dados com toda a informação relevante para o desenvolvimento do projeto, realizou-se uma análise exploratória ao primeiro conjunto de dados e, na análise gráfica da qualidade, verificou-se, na análise da qualidade dos dados, que estes só apresentavam qualidade suficiente em todas as máquinas apenas a partir de 2022. Assim, foi necessário mudar para um novo conjunto de dados, com informação a partir dessa data.

Tendo em conta o objetivo de um modelo geral que fosse aplicável a diferentes máquinas e empresas, foi necessário fazer o processamento dos dados, transformando registos relativos a intervalos de tempo em que uma máquina esteve num de dois estados, a produzir ou parada, em ciclos de operação de igual duração, apresentando a variável RUL, calculada automaticamente.

Foram escolhidos modelos *ensemble*, especificamente o RF e XGBoost, que criam várias árvores de regressão e combinam os diversos resultados. Após a seleção e construção destes modelos, realizou-se a sua avaliação, aplicando validação cruzada, verificando-se que o XGBoost apresentou melhores resultados em comparação com o RF, quando reduzindo a variabilidade do RUL. Os resultados obtidos foram validados graficamente. Por fim, foi aplicado o GridSearchCV ao algoritmo com melhores resultados para identificar os melhores hiperparâmetros.

5.2 Limitações

A principal limitação encontrada foi a impossibilidade de reunir com o cliente para obter uma melhor compreensão das máquinas, do seu funcionamento, das paragens e dos produtos. A falta dessa interação não permitiu perceber as máquinas a serem analisadas, perceber a qualidade da catalogação das paragens em paragens programadas, não programadas por avaria e não programadas por causas externas. Não permitiu definir o intervalo de RUL que o modelo deveria prever com eficácia, para que pudessem ser aplicadas as correções necessárias.

Outra limitação foi a ausência de variáveis relacionadas diretamente com a máquina, como dados de sensores que indicassem a sua saúde. Na manutenção preditiva, normalmente são utilizados dados de sensores recolhidos de forma regular, o que não aconteceu neste projeto. Não existiam dados de sensores, e os registos não eram de igual duração. Foram utilizados dados superficiais sobre o estado da máquina (saúde da máquina), ou seja, dados de eventos que ocorreram com a máquina, que apenas permitem saber se a máquina está a trabalhar ou parada, recolhidos e validados manualmente pelo operador, permitindo apenas identificar o motivo da paragem. Deste modo, foi necessário transformar os dados, criando variáveis e definindo cada registo como um ciclo de operação. Além disso, não tínhamos explicitamente o target, que precisou de ser criado.

Estas limitações fizeram com que os objetivos inicialmente previstos fossem ajustados ao longo do projeto, exigindo uma adaptação contínua.

5.3 Cumprimento dos objetivos

Apesar das limitações encontradas e das alterações aos objetivos iniciais, considera-se que os objetivos foram alcançados com sucesso. Foi possível desenvolver uma solução de manutenção preditiva através do cálculo do RUL, capaz de antecipar falhas ou paragens dos equipamentos.

Verificou-se a importância de ter disponível dados sobre o estado de saúde da máquina num modelo de manutenção preditiva; sem esses dados, o modelo não consegue obter informação útil suficiente para prever o RUL de forma eficaz.

Ao longo do projeto, surgiu a necessidade de elaborar um conjunto de sugestões para aumentar a qualidade dos dados. Essas sugestões incluem a definição de uma estratégia, objetivos e metas antes de se iniciar o desenvolvimento de um modelo de manutenção preditiva. Foi também identificada a necessidade de reduzir a quantidade de códigos de paragem, considerando que 30 é um número excessivo, com algumas designações redundantes. Sugere-se sensibilizar os operadores para uma caraterização correta das paragens, uma vez que é improvável que uma máquina tenha uma manutenção curativa de 2 a 3 minutos. Da mesma forma, não é consistente que a mesma máquina apresente ciclos de operação de 1 hora e outros de 250 horas. É importante ainda compreender se é o operador quem faz a classificação da paragem e se este está alocado de forma estacionária a uma máquina ou trabalha de forma rotativa entre várias.

5.4 Sugestões de trabalhos futuros

Como sugestão para trabalho futuro, seria interessante aplicar outros algoritmos ao modelo desenvolvido para verificar se há alterações significativas nos resultados. Além disso, recomenda-se aplicar o modelo a um conjunto de dados de sensores, com as adaptações necessárias, de forma a explorar a possibilidade de melhorias na precisão das previsões. O código da criação dos ciclos, é sobre todos os dados disponíveis, ou seja, se forem adicionados novos dados, a criação de ciclos será sobre todos os dados e não só dos novos dados e, portanto, é algo a melhorar no futuro. Quando não havia tempo suficiente para criar um ciclo de operação, esse tempo era excluído, seria interessante arranjar forma de adicionar esse tempo após o último ciclo de operação. Fazer otimizações ao código, na criação dos ciclos, o código completo demora 2/3 minutos a executar.

5.5 Considerações finais

Para aplicar e desenvolver uma solução de manutenção preditiva, é essencial, numa fase inicial, definir e implementar uma estratégia de acordo com os objetivos e metas que se pretendem alcançar, em colaboração com o cliente. É necessário avaliar a infraestrutura da empresa, ou seja, verificar se existem os equipamentos necessários para a recolha de dados, neste caso, sensores. Por fim, é fundamental a formação da equipa, garantindo que saibam interpretar os resultados das análises e que estejam familiarizados com as ferramentas e os procedimentos adotados, de modo a garantir o sucesso da solução a implementar.

A nível pessoal, o desenvolvimento deste projeto permitiu aprofundar os conhecimentos em linguagem de programação Python, em bibliotecas de aprendizagem computacional e em sistemas de gestão de bases de dados, como o SQL Server. Permitiu ainda adquirir conhecimentos no setor industrial, concretamente na área da manutenção. Além destas competências técnicas, outras competências foram desenvolvidas, como a resiliência e capacidade de adaptação, que foram decisivas para o desenvolvimento e conclusão do projeto.

REFERÊNCIAS BIBLIOGRÁFICAS

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0: Step-by-step Data Mining Guide* (SPSS Incorporated, Ed.; pp. 1–76). CRISP-DM Consortium.

Chen, X., Jin, G., Qiu, S., Lu, M., & Yu, D. (2020). Direct Remaining Useful Life Estimation Based on Random Forest Regression. *2020 Global Reliability and Prognostics and Health Management, PHM-Shanghai 2020*. https://doi.org/10.1109/PHM-Shanghai49105.2020.9281004

Dogan, A., & Birant, D. (2021). Machine learning and data mining in manufacturing. *Expert Systems with Applications*, *166*(114060). https://doi.org/10.1016/j.eswa.2020.114060

Habibullah, M., Shen, Y., Yang, F., Aggarwal, S., Zhou, Y., Hussain, S., Kumar Das, A., & Jhinaoui, A. (2021). *Chapter 4: A Perspective into Analysing Tool Wear Condition in Hard-Turning Process—The Key Lessons Learnt* (C. Toro, W. Wang, & H. Akhtar, Eds.; Vol. 202, pp. 79–111). https://doi.org/10.1007/978-3-030-67270-6

Li, Y. H., Inoue, L. C. G. V., & Sinha, R. (2022). Real-time OEE visualisation for downtime detection. *IEEE International Conference on Industrial Informatics (INDIN)*, *2022-July*, 729–734. https://doi.org/10.1109/INDIN51773.2022.9976067

Mathew, V., Toby, T., Singh, V., Rao, B., & Kumar, M. (2017). *Prediction of Remaining Useful Lifetime (RUL) of turbofan engine using machine learning*. 306–311. https://doi.org/10.1109/ICCS1.2017.8326010

Nunes, P., Santos, J., & Rocha, E. (2023). Challenges in predictive maintenance – A review. *CIRP Journal of Manufacturing Science and Technology*, *40*, 53–67. https://doi.org/10.1016/j.cirpj.2022.11.004

SINMETRO. (n.d.). *OEE Da teoria à prática* (pp. 1–18). www.accept.pt

SINMETRO. (2023a). *ACCEPT*. https://www.accept.pt/

SINMETRO. (2023b). *Manual de Acolhimento e integração do colaborador* (pp. 1–30).

SINMETRO. (2023c). *SINMETRO - Consultoria Tecnológica, Software e Formação*. https://www.sinmetro.pt/

Thomas, D. (2018). Advanced Maintenance in Manufacturing: Costs and Benefits. *Proceedings of the Annual Conference of the PHM Society*, *10*(1), 1–12. https://doi.org/https://doi.org/10.36001/phmconf.2018.v10i1.536

Thomas, D., & Weiss, B. (2021). Maintenance costs and advanced maintenance techniques in manufacturing machinery: Survey and analysis. *International Journal of Prognostics and Health Management*, *12*(1), 1–13. https://doi.org/10.36001/IJPHM.2021.V12I1.2883

Tian, P. (2024). Research On Laptop Price Predictive Model Based on Linear Regression, Random Forest and Xgboost. *Highlights in Science, Engineering and Technology CSIC*, *2023*, 265–271. https://doi.org/10.54097/9nx5ad16

Wirth, R., & Hipp, J. (2000). *CRISP-DM: Towards a Standard Process Model for Data Mining*. 29–40. https://www.tib.eu/en/search/id/BLCP:CN039162600/CRISP-DM-Towards-a-Standard-Process-Model-for-Data?cHash=16bb58abe6455a858f809fd55fb0ca8f

Zhu, T., Ran, Y., Zhou, X., & Wen, Y. (2024). A Survey of Predictive Maintenance: Systems, Purposes and Approaches. *IEEE Communications Surveys & Tutorials*, 1–38. http://arxiv.org/abs/1912.07383

APÊNDICES

APÊNDICE 1. Super query para criação da tabela temporária das variáveis relevantes

```
Use database

SELECT
    TS_Inicio_RD as Inicio,
    TS_Fim_RD as Fim,
    Tempo_Decorrido_RD as Duracao,
    OEE_v2.tl_oee.ID_Processo as ID_Maquina,
    Estado_Processo as Estado_Maquina,
    Paragem_Ativa as Paragem,
    QTD_OK_RD as Qtd_OK,
    QTD_NOK_RD as Qtd_NOK,
    QTD_Total_RD as Qtd_Total,
    Ignora_contadores as Invalida_Qtd_OEE,
    OEE_v2.Producoes.ID_Produto AS ID_Produto,
    CAST((Case When Paragem_Ativa in
('MAN','03','05','09','10','11','13','15','16','17','18','22','95','25','26') then 1 else 0 end) as bit)
as Programada,
    CAST((Case When Paragem_Ativa in ('01','02','12','100','14','21') then 1 else 0 end) as bit) as
Nao_Programada_Avaria,
    CAST((Case When Paragem_Ativa in ('04','06','07','08','20','23','24') then 1 else 0 end) as bit) as
Nao_Programada_Externa,
    OEE_v2.RegistosDiarios.tipo_estado as Valida_Dia
INTO ##Temp_Table
FROM OEE_v2.tl_oee with(nolock)
LEFT JOIN OEE_v2.Producoes ON OEE_v2.tl_oee.ID_Producao = OEE_v2.Producoes.ID
JOIN OEE_v2.RegistosDiarios ON OEE_v2.tl_oee.ID_Processo = OEE_v2.RegistosDiarios.ID_Processo AND
OEE_v2.tl_oee.Data_Producao = OEE_v2.RegistosDiarios.Data_Producao
WHERE
    OEE_v2.tl_oee.Deleted_at IS NULL
    AND OEE_v2.tl_oee.Pertence_RD = 1
    AND OEE_v2.RegistosDiarios.tipo_estado = 1
    AND OEE_v2.tl_oee.TS_Inicio_RD >= '2022-01-01'
ORDER BY 1, 2;

Drop table ##Temp_Table;
```

APÊNDICE 2. Código de conexão do SQL ao VSCode

```
import json
import pyodbc

# Ler as informações do ficheiro de configuração
with open('login.json', 'r') as login_file:
    login = json.load(login_file)

db_user = login['username']
db_password = login['password']

# Conexão ao SQL Server
connection = pyodbc.connect(f"DRIVER={{ODBC Driver 17 for SQL Server}};\
                            SERVER=[redacted];DATABASE=[redacted]ID={db_user};PWD=
{db_password}")

# Definir consulta SQL
sql_query = "SELECT * FROM ##Temp_Table ORDER BY 1,2;"

# Usar a função read_sql para criar um DataFrame a partir da consulta
df = pd.read_sql(sql_query, connection)

# Terminar a ligação
connection.close()

Exibir o DataFrame
#print(df)
```

APÊNDICE 3. Código completo para criação de ciclos

```python
def calculate_cycle_durations(df_copy, cycle_duration, prog_stop_ids, specific_stop_ids, ext_stop_ids):
    df_without_stops = df_copy[~df_copy['stop_id'].isin(prog_stop_ids + specific_stop_ids +
ext_stop_ids)]
    num_cycles = df_without_stops['duration'].sum() // cycle_duration
    #range(start, stop, step)
    if not df_without_stops.empty:  #needed to add this because we have empty subsets
        cycle_starts = pd.Series(range(df_without_stops['start'].min(),
                                       df_without_stops['start'].min() + (num_cycles + 1) *
cycle_duration, cycle_duration))
        cycle_ends = cycle_starts.shift(-1).dropna().astype(int)

        return [
            (df_without_stops['end'].clip(upper=end).max() -
df_without_stops['start'].clip(lower=start).min())
            for start, end in zip(cycle_starts, cycle_ends)
        ]
```

```python
def calculate_stop_info(df_copy, cum_cycle_durations, prog_stop_ids, specific_stop_ids, ext_stop_ids):
    df_copy1 = df_copy[~df_copy['stop_id'].isin(prog_stop_ids + specific_stop_ids +
ext_stop_ids)].reset_index(drop=True)
    # Check if the first row 'start' value is not 0, then adjust it along with 'end'
    if df_copy1.at[0, 'start'] != 0:
        start_diff = df_copy1.at[0, 'start']
        df_copy1['start'] -= start_diff
        df_copy1['end'] -= start_diff

    for i in range(1, len(df_copy1)):
        if df_copy1.at[i, 'start'] != df_copy1.at[i - 1, 'end']:
            df_copy1.at[i, 'start'] = df_copy1.at[i - 1, 'end']
            df_copy1.at[i, 'end'] = df_copy1.at[i, 'start'] + df_copy1.at[i, 'duration']

    # Add a new column 'cum_duration'
    df_copy1['cum_duration'] = df_copy1['duration'].cumsum()
    cycle_starts = cum_cycle_durations.shift(fill_value=0)
    cycle_ends = cum_cycle_durations

    stop_100_counts = []
    stop_100_durations = []

    stop_100_groups = df_copy1[df_copy1['stop_id'] == '100'].groupby('start')
['duration'].sum().to_dict()

    for start, end in zip(cycle_starts, cycle_ends):
        stops_within_cycle = df_copy1[(df_copy1['start'] <= end) \
                                      & (df_copy1['end'] >= start)]

        stop_100 = stops_within_cycle[stops_within_cycle['stop_id'] == '100']

        total_duration_within_cycle = 0
        total_stop_100_counts = 0

        for _, stop in stop_100.iterrows():
            stop_start = max(start, stop['start'])
            stop_end = min(end, stop['end'])
            stop_duration = stop_end - stop_start if stop_start < stop_end else 0
            total_duration_within_cycle += stop_duration

            stop_group_duration = stop_100_groups.get(stop['start'], 0)
            if stop_group_duration > 0:
                fraction_duration = stop_duration / stop_group_duration
                total_stop_100_counts += fraction_duration

        stop_100_counts.append(total_stop_100_counts)
        stop_100_durations.append(total_duration_within_cycle)

    return stop_100_counts, stop_100_durations
```

```python
def calculate_prog_stop_info(df_copy, cum_cycle_durations, prog_stop_ids, specific_stop_ids,
ext_stop_ids):
    cycle_starts = cum_cycle_durations.shift(fill_value=0)
    cycle_ends = cum_cycle_durations

    df_copy1 = df_copy.copy()

    df_copy1['cum_duration'] = df_copy1['duration'] \
                               .where(~df_copy1['stop_id']
                                      .isin(prog_stop_ids + specific_stop_ids + ext_stop_ids), 0) \
                                       .cumsum()

    prog_stop_counts = []
    specific_stops_counts = []
    ext_stops_counts = []

    for start, end in zip(cycle_starts, cycle_ends):
        cycles_mask = (df_copy1['cum_duration'] >= start) & (df_copy1['cum_duration'] <= end)

        prog_stop_counts.append(df_copy1[cycles_mask &
df_copy1['stop_id'].isin(prog_stop_ids)].shape[0])
        specific_stops_counts.append(df_copy1[cycles_mask &
df_copy1['stop_id'].isin(specific_stop_ids)].shape[0])
        ext_stops_counts.append(df_copy1[cycles_mask &
df_copy1['stop_id'].isin(ext_stop_ids)].shape[0])

    return prog_stop_counts, specific_stops_counts, ext_stops_counts
```

```python
def create_cycle_dataframe(data_machine, cycle_duration, stop_ids, prog_stop_ids, specific_stop_ids,
ext_stop_ids):
    df = pd.DataFrame(data_machine).drop(columns=['machine_id','machine_state'])
    #df.reset_index(drop=True, inplace=True)
    df['start'] = df['duration'].shift(1).fillna(0).cumsum().astype(int)
    df['end'] = df['duration'].cumsum().astype(int)
    df = df[['start', 'end', 'duration', 'stop_id']]

    stop_indices = df[df['stop_id'].isin(stop_ids)].index

    cycle_dfs = []
    prev_idx = 0

    stop_indices = list(stop_indices) + [len(df)]
    stop_number = 1

    for idx in stop_indices:
        if prev_idx == 0 and idx == 0 and df.iloc[0]['stop_id'] in stop_ids:
            prev_idx = 1
            continue

        df_copy = df.iloc[prev_idx:idx]
        if len(df_copy) > 0:
            cycle_durations = calculate_cycle_durations(df_copy, cycle_duration, prog_stop_ids,
specific_stop_ids, ext_stop_ids)
            if not cycle_durations:  # Check if the list is empty
                continue  # Skip this iteration if 'cycle_durations' is empty

            cum_cycle_durations = pd.Series(cycle_durations).cumsum()

            stop_100_counts, stop_100_durations = \
                calculate_stop_info(df_copy, cum_cycle_durations, prog_stop_ids, specific_stop_ids,
ext_stop_ids)
            #cum_stop_100_counts = pd.Series(stop_100_counts).cumsum()
            #cum_stop_100_durations = pd.Series(stop_100_durations).cumsum()

            prog_stop_counts, specific_stops_counts, ext_stops_counts = \
                calculate_prog_stop_info(df_copy, cum_cycle_durations, prog_stop_ids,
specific_stop_ids, ext_stop_ids)

            cycle_df = pd.DataFrame({
                'stop_number': [stop_number] * len(cycle_durations),
                'cycle': range(1, len(cycle_durations) + 1),
                'duration': cycle_durations,
                'cum_duration': cum_cycle_durations,
                'stop_100_count': stop_100_counts,
                #'cum_stop_100_count': cum_stop_100_counts,
                'stop_100_duration': stop_100_durations,
                #'cum_stop_100_duration': cum_stop_100_durations,
                'prog_stop_count': prog_stop_counts,
                'spec_stop_count': specific_stops_counts,
                'ext_stop_count': ext_stops_counts,
                'RUL': range(len(cycle_durations), 0, -1)
            })

            cycle_dfs.append(cycle_df)
            prev_idx = idx + 1
            stop_number += 1

    result = pd.concat(cycle_dfs)
    return result.reset_index(drop=True)
```