Universidade do Minho
Escola de Engenharia

Afonso José Torres da Silva e Sousa

Aplicação de Técnicas de *Machine Learning* para a Previsão de Processos Industriais

October 2022

Aplicação de Técnicas de *Machine Learning* para a previsão de Processos Industriais

Afonso Sousa

UMinho |

Universidade do Minho
Escola de Engenharia

Afonso José Torres da Silva e Sousa

Aplicação de Técnicas de *Machine Learning* para a Previsão de Processos Industriais

Dissertação de Mestrado
Mestrado Integrado em Engenharia e Gestão de Sistemas de Informação

Trabalho efetuado sob a orientação de:
Professor Doutor Paulo Alexandre Ribeiro Cortez

Outubro 2022

AGRADECIMENTOS

A felicidade está caminho que percorremos e não no destino ao qual chegamos, por isso, quero agradecer a todos que caminharam ao meu lado. Tornaram esta viagem única.

Ao professor Doutor Paulo Cortez, agradeço a partilha do conhecimento, o qual me permitiu desenvolver competências na área de *Machine Learning*, e da sua disponibilidade demonstrada, para a elaboração deste projeto.

Aos meus colegas do CCG, obrigado por todo o apoio e companheirismo. Vocês tiveram um papel fundamental no meu percurso pessoal e profissional, durante o último ano.

À minha Mãe e ao meu Pai, agradeço por tudo o que fizeram por mim nos últimos 24 anos da minha existência.

À Joanna, com quem tenho o prazer de partilhar todos os meus momentos, o meu mais profundo obrigado. És a minha inspiração, sem ti ao meu lado, não seria a pessoa que sou hoje.

A todos que marcaram o meu trajeto, o meu obrigado.

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.
Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Resumo

No âmbito do desenvolvimento de novas soluções para a otimização dos processos industriais de produção, novos desafios têm surgidos devido à complexidade de criação de valor, e às implicações associadas aos processos produtivos ao longo das cadeias de abastecimento. Embora o desenvolvimento tecnológico seja um objetivo inerente a todas as empresas, é necessário assegurar que o crescimento económico seja proporcional aos benefícios que este irá providenciar à sociedade, assim como, realizado segundo políticas de desenvolvimento sustentável e circular.

Esta dissertação tem como objetivo conceber uma investigação relativa à implementação de técnicas de *Machine Learning* (ML), que poderão ser associadas aos sistemas de planeamento de produção, permitindo automatizar, prever e atuar sobre a gestão dos processos de produção. Os objetivos passam por investigar quais os benefícios da implementação de determinado método e técnica de *Machine Learning*, de modo a criar previsões essenciais para o desenvolvimento de um plano de produção, com base nos resultados obtidos através de dados históricos.

Palavras-Chave: Análises Preditivas, AutoML, Indústria 4.0, Machine Learning, Séries Temporais.

Abstract

In the development of new solutions for the optimization of industrial production processes, new challenges have risen due to the complexity of value creation and the implications associated with production processes, along the supply chains. While technological development is an inherent goal for every company, it is necessary to ensure that economic growth is proportional to the benefits it provides to society, as well as carried out under policies of sustainable and circular development.
This dissertation aims to research the implementation of Machine Learning (ML) techniques that can be associated with production planning systems, allowing to automate, predict and act on the management of production processes. The objectives are to investigate the benefits of implementing certain methods and techniques of Machine Learning, in order to create essential predictions for the development of a production plan, based on the results obtained through historical data.

Keywords: AutoML, Industry 4.0, Machine Learning, Predictive Analysis, Time-Series.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas, Siglas e Acrónimos

AR	*Autoregressive*.
ARIMA	*Autoregressive Integrated Moving Average*.
ARMA	*Autoregressive Moving Average*.
AutoML	*Automated Machine Learning*.
CCG	Centro de Computação Gráfica.
CRISP-DM	*CRoss Industry Standard Process for Data Mining*.
GLM	*Generalized Linear Model*.
IA	Inteligência Artificial.
IDF	*Inverse Document Frequency*.
KNN	*K Nearest Neighbors*.
LSTM	*Long Short-Term Memory Recurrent Neural Network*.
MA	*Moving Average*.
MAE	*Mean Absolute Error*.
ML	*Machine Learning*.
NMAE	*Normalized Mean Absolute Error*.
RNN	*Recurrent Neural Network*.
SVM	*Support Vector Machine*.
TPOT	*Tree-based Pipeline Optimization Tool*.

1

Introdução

1.1 Enquadramento e Motivação

Esta dissertação surge no âmbito do projeto *"PRODUTECH4S&C: PRODUTECH SUSTENTÁVEL & CIRCULAR"*, o qual foi desenvolvido no CCG.

Este projeto de mestrado consiste em desenvolver estratégias e serviços que permitam empresas tomar decisões devidamente suportadas, de forma a controlarem as suas cadeias de abastecimento. Ao obter o controlo completo face às entradas e saídas de materiais necessários na produção de um produto, assim como, do tempo necessário para a produção, as empresas tem a possibilidade de prever e concluir as encomendas dos clientes de forma eficiente. A implementação dos métodos preditivos irá desempenhar um papel fundamental no processo de desenvolvimento de um plano de produção, e consequentemente, melhorias nos processo de produção da empresa.

Particularmente, um dos componentes do projeto *PRODUTECH4S&C*, o *PPS4 - Cadeia digital de fornecimento em contexto circular* consiste no estudo e desenvolvimento de novos processos de otimização e controlo das cadeias de abastecimento. Pretende alcançar este objetivo recorrendo à investigação de tecnologias emergentes no mercado, através da implementação de soluções de *software* e de técnicas ligadas à Inteligência Artificial, mais concretamente a algoritmos de *Machine Learning*. O objetivo destes modelos e algoritmos, face ao planeamento dos processos produtivos, terá como principal função, permitir às empresas sustentarem as suas decisões relativas aos processos de produção, com base nas previsões resultantes dos modelos de *Machine Learning*.

1.2 Objetivos e Resultados Esperados

O principal objetivo deste projeto consiste no estudo de ferramentas de análise preditivas, as quais permitam prever o número de encomendas que uma empresa irá receber ao longo de um período, assim como, prever o tempo de produção de uma encomenda. Estas previsões irão permitir elaborar um plano produtivo com base no histórico da organização.

Para responder a este problema, a solução consiste em realizar um estudo relativo às técnicas de *Machine Learning* existentes, que face ao histórico de encomendas da empresa, permita prever o número de encomendas que irá receber ao longo de um intervalo de tempo, assim como, prever o tempo de produção de uma encomenda, de acordo com os processos de manufatura da empresa. Esta proposta visa desenvolver uma solução que permita obter previsões com suporte em dados, de modo a facilitar o processo de tomada de decisão, relativamente à elaboração de um plano de produção.

No âmbito desta dissertação serão realizadas tarefas de investigação com o objetivo de explorar os processos organizacionais da empresa, que poderão ser otimizados através de modelos de *Machine Learning*. Esta solução irá consistir na utilização do histórico de dados de uma empresa da indústria de manufatura. Os dados serão devidamente tratados, para servirem como *input* para o desenvolvimento dos modelos preditivos de ML. Por fim, o objetivo passará por realizar uma síntese do trabalho realizado, composto pelos resultados obtidos em cada um dos casos de estudo.

O sucesso no âmbito desta dissertação irá resultar dos modelos preditivos a desenvolver, e consequentemente, da eficácia dos resultados obtidos. A introdução das técnicas de *Machine Learning* deverão ter um impacto positivo no desenvolvimento do planeamento dos processos de produção, contribuindo com melhorias significativas nas previsões de encomendas e dos tempos de produção. Os resultados preditivos irão servir como base de suporte, para processo de tomada de decisão, ao longo do desenvolvimento do plano produtivo da indústria.

1.3 Organização do Documento

O presente documento encontra-se dividido em cinco capítulos, os quais abordam diferentes tópicos, apresentando todos os conteúdos relevantes para a dissertação.

Primeiramente, o capítulo 1 consiste na introdução ao tema da dissertação, iniciando-se com um enquadramento e motivações para o projeto, de seguida, uma apresentação dos objetivos e resultados esperados, e por fim, a descrição das metodologias utilizadas para o desenvolvimento da investigação e das tarefas de *Machine Learning*.

No capítulo 2, Estado da Arte, são introduzidos os conceitos relevantes para a dissertação e há um esclarecimento dos mesmos, de modo a alcançar uma visão abrangente à cerca do tema. O capítulo inicia-se com uma introdução à Indústria 4.0, de seguida, aos conceitos de Inteligência Artificial e *Machine Learning*, sendo detalhadas técnicas fundamentais para o desenvolvimento das soluções presentes nesta dissertação.

No capítulo 3, o caso de estudo *Previsão de Encomendas* é inicialmente apresentado e a formulação do problema é elaborada. A proposta para solucionar o problema é descrita ao longo do capítulo, sendo que, por fim, os resultados obtidos por parte dos modelos de *Machine Learning* são apresentados.

O capítulo 4, apresenta o caso de estudo *Previsão do Tempo de Produção* segue uma estrutura semelhante ao capítulo anterior. O caso de estudo está divido em três iterações da metodologia CRISP-DM. Desta forma, o caso de estudo é explorado, a formulação do problema é concebida, e a demonstração da solução e os resultados obtidos, são expostos.

Por fim, surge o capítulo 5, referente à Conclusão, onde são reunidas um conjunto de considerações finais relativas a este documento.

1.4 Abordagem Metodológica

Com o objetivo de desenvolver e alcançar um projeto final bem sucedido, de acordo com a complexidade relacionada com a elaboração de uma dissertação, de modo a compreender os aspetos fundamentais, é necessário definir um conjunto de metodologias de trabalho. Para auxiliar o desenvolvimento deste projeto, técnicas de boas práticas de investigação deverão ser implementadas.

Neste contexto foi utilizada a metodologia CRISP-DM *(CRoss-Industry Standard Process for Data Mining)*, que serve de suporte a projetos de *Data Mining*, que permitirá definir o plano de ação. Esta metodologia foi desenvolvida com o objetivo de servir como um guião para o desenvolvimento de projetos de *Data Mining*.

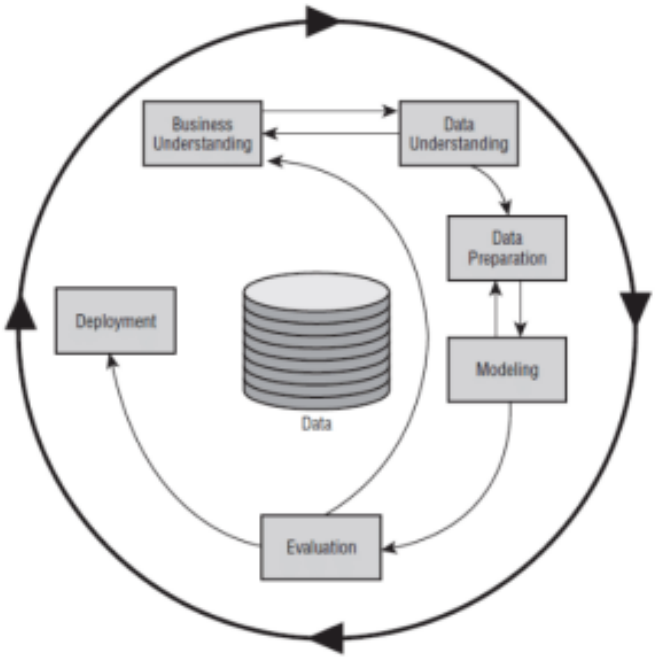

Figura 1: Modelo CRISP-DM (adaptado de Wirth, 2000)

Assim sendo, a metodologia CRISP-DM está dividida em seis etapas: Business Understanding, Data Understanding, Data Preparation, Evaluation e Deployment. Cada fase possui um conjunto de tarefas específicas:

1. *Business Understanding* (Compreensão do Negócio): Sendo esta a etapa inicial, é expectável identificar os objetivos do projeto e os seus requisitos, de forma a compreender a perspetiva desta dissertação. Após estarem os requisitos definidos, deverá ser estabelecido um plano de ação para alcançar os objetivos propostos.

2. *Data Understanding* (Compreensão dos Dados): Nesta etapa, os dados serão recolhidos e posteriormente será realizada uma análise dos mesmos, para compreender as diversas relações entre as variáveis, e identificar os problemas relacionados com a qualidade dos dados, como incoerências que estes possam apresentar.

3. *Data Preparating* (Preparação dos Dados): Esta etapa é caracterizada pelo tratamento e processamento de dados. Dependendo das características dos mesmos serão selecionadas as técnicas necessárias para o seu tratamento, de forma a elaborar um *dataset* final para a modelação. Sendo o CRISP-DM uma metodologia cíclica, caso sejam detetadas falhas que necessitem de serem tratadas novamente, esta tarefa deverá ser novamente executada.

4. *Modeling* (Modelação): Esta etapa consiste na seleção das técnicas de modelação e definir os parâmetros, que poderão ser ajustados para obter os melhores valores finais. Com um conjunto de requisitos específicos de acordo com as limitações de cada projeto e das características dos dados poderá ser necessário voltar à etapa anterior, para um ajuste no tratamento de dados, para que a seleção dos modelos seja a mais eficaz, para garantir os melhores resultados possíveis.

5. *Evaluation* (Avaliação): Esta etapa consiste na avaliação dos modelos aplicados na etapa anterior, de forma a selecionar um, ou vários modelos, cuja qualidade cumpra com os requisitos estipulados.

6. *Deployment* (Implementação): A fase da implementação pode ter diversos objetivos conforme os requisitos propostos para cada projeto, e não significa necessariamente o fim do projeto.

2

Estado de Arte

2.1 Estratégia de Pesquisa Bibliográfica

Para o desenvolvimento desta dissertação é fundamental seguir um método de investigação adequado, que permita alcançar os objetivos propostos, e uma solução devidamente estruturada. Desta forma, a escolha de uma metodologia adequada, que seja devidamente relacionada com o tema, é fundamental para potenciar a elaboração da investigação.

Com o objetivo de elaborar o presente documento, a revisão da literatura foi realizada seguindo uma abordagem assistemática. A pesquisa dos conceitos fundamentais no contexto do problema, foi realizada sem descrever as diversas etapas, assim como, não houve a definição de critérios de pesquisa, pois novos conceitos surgiam à medida que a investigação avançava. Sendo esta uma das maiores vantagens relativamente à abordagem sistemática, na qual a pesquisa é definida no ponto de partida e rigorosamente seguida durante a revisão da literatura.

Para cumprir com a necessidade de escolher fontes de informações credíveis foi utilizado o motor de busca *Google Scholar*, mas também o *Scopus* e o *Science Direct*. Estas fontes foram escolhidas para obter artigos científicos, livros e dissertações com potencial de serem determinantes na compreensão e desenvolvimento do projeto. Para esta pesquisa também foi tido em consideração o ano das publicações, de forma a obter informação atualizada e representativa da realidade.

Assim sendo, para a realização desta pesquisa foram utilizados conceitos fundamentais para a elaboração do documento como: *"Industry 4.0"*, *"Artificial Intelligence"*, *"Machine Learning"*, *"AutoML"* e *"Time Series"*. À medida que os artigos foram sendo analisados, novos conceitos surgiram, e por vezes, esses termos foram utilizados para melhorar a pesquisa.

Com um conjunto de artigos selecionados, foi realizado um estudo da literatura selecionada, de modo a utilizar os artigos mais relevantes para apoiar cientificamente, com base nas referências bibliográficas, a elaboração do presente documento.

2.2 Indústria 4.0

O fenómeno da Indústria 4.0 foi mencionado pela primeira vez em 2011 na Alemanha como uma proposta para o desenvolvimento de um novo conceito de política económica alemã, baseado em estratégias de alta tecnologia. Consiste numa era que veio revolucionar a indústria pela interação entre o mundo digital e o real, com a adoção de tecnologias responsáveis por conectar tudo e todos. Esta indústria é a componente económica encarregue da produção de bens materiais muito mecanizados e automatizados, também encarregue da forma como os *shop floors* operam (Vaidya et al., 2018).

Definido por muitos como uma transformação global da indústria de manufatura, com a introdução da digitalização e da Internet, estas transformações envolvem melhorias revolucionárias nos processos de manufatura, operações e serviços de produtos e sistemas. Partilham muitas semelhanças com os desenvolvimentos no âmbito das *Smart Factories*, *Smart Industry*, *Advanced Manufacturing* e *Industrial Internet of Things* (Tjahjono et al., 2017). No contexto da Indústria 4.0, há ainda a possibilidade de conectar máquinas com humanos através da utilização de Sistemas Ciber-Físicos. Este tipo de sistemas monitoriza os processos físicos, criando uma cópia virtual do mundo real, que toma decisões descentralizadas (Lu, 2017).

Os principais atributos da Indústria 4.0 são a digitalização, otimização e a customização da produção; automatização e adaptação; interação homem-máquina; serviços e negócios de valor acrescentado; comunicação e troca de dados automática. A Indústria 4.0 fornece mudanças disruptivas nas cadeias de abastecimento, modelos de negócio e processos de negócio. A Indústria 4.0 pode ser resumida como um processo de manufatura integrado, adaptado, otimizado, orientado a serviços e interoperável, que está correlacionado com algoritmos, *Big Data* e tecnologias de ponta. Sendo assim, os princípios da Indústria 4.0 são a interoperabilidade, virtualização, descentralização, capacidade em tempo real, orientação para serviço e modularidade (Shafiq et al., 2015).

Apesar do conceito de Indústria 4.0 se estar a desenvolver, torna-se necessário destacar os fatores que a impulsionam. Atualmente, já se verifica um progresso notável na capacidade dos computadores, assim como, na quantidade de informação digitalizada, (*Big Data*), isto devido aos atuais procedimentos de inovação.

São diversos benefícios associados como, o aumento da produtividade, sendo que, esta indústria oferece modelos de automação e otimização dos processos eficientes, os quais contribuem para um aumento da produtividade. Consequentemente, há uma redução dos custos associados, uma quebra nas falhas do sistema e um aumento da eficácia da produção. A melhoria das condições de trabalho, ainda que esta indústria seja fundamentada em tecnologias, as questões sociais e ambientais não são ignoradas, pelo contrário, são aspetos importantes uma vez que se pretende trabalhar de forma sustentável.

Na Indústria 4.0, não só importam os sistemas e tecnologias, mas os colaboradores que também têm um papel relevante. Para os operários esta mudança vai simplificar o cotidiano das funções, de forma a não serem stressantes nem de risco. Outros dos benefícios da Indústria 4.0 consiste na produção de produtos e serviços personalizados, pois com o passar dos anos, a sociedade tornou-se mais consciencializada e exigente. Desta forma, não existe apenas o desejo por um produto, mas sim pelo bem maior e a sua responsabilidade social para o planeta.

Posto isto, há a necessidade cada vez mais de despertar o interesse das pessoas, e para isso é necessário responder aos desafios da sustentabilidade, assim como a personalização para o cliente. A personalização vai ajudar a ir ao encontro das expectativas dos clientes e assim ganhar vantagem competitiva no mercado atual. Por fim, a Quarta Revolução Industrial trouxe também como benefício a capacidade de ser ágil, flexível e escalável. Os avanços na inovação a nível da Inteligência Artificial, Robótica, *Big Data* e Sistemas Ciber-Físicos são alguns exemplos que facilitam o cálculo e as previsões de produção e compra (Lasi et al., 2014).

2.3 Inteligência Artificial

A Inteligência Artificial tem como principal objetivo procurar métodos e formas dos computadores fazerem o mesmo tipo de análises que a mente humana faz sistematicamente. Esta definição de IA apesar de ser bastante simples de compreender, a verdade é que por detrás desta afirmação tão breve, não revela a tamanha complexidade e potencial deste tema. Ao afirmar que este conceito relaciona a mente humana, referimo-nos às capacidades psicológicas como a previsão, associação, perceção, planeamento, entre outras.

O primeiro sistema de Inteligência Artificial apareceu primeiramente em 1955, concebido pelo professor *Allen Newell*, que atribui o nome de *Logic Theorist* ao sistema que tinha criado, o qual provava cerca de 40 teoremas. Um ano depois, em 1956, é o ano em que o conceito de Inteligência Artificial surge, devido à conferência em *Dartmouth College*.

Em investigações futuras, *Allen Newell*, *Simon* e *Shaw*, desenvolveram um sistema de processamento de habilidades mentais humanas que permitia resolver um conjunto de problemas como a Integração Simbólica, fazer o puzzle da *Torre de Hanoi*, etc. Desta forma, o paradigma da simulação cognitiva ficou definido, o qual afirma que, um sistema de Inteligência Artificial é estruturado pela forma como os humanos resolvem os problemas (Chang, 2020). Mais tarde, já em 1972, *Allen Newell* e *Simon*, reconheceram a demonstração de *Oliver Selfridge*, sobre a manipulação para o reconhecimento de padrões. Este trabalho consistia numa aprendizagem e numa abordagem com múltiplos agentes para a resolução de problemas. Os outros trabalhos relativos aos anos de 1950 também foram considerados, concluindo que as demonstrações realçavam o impressionante poder das heurísticas para o

desenvolvimento da IA. Assim sendo, a tese de *Thomas Evans* em 1963, sobre a resolução de problemas de analogia para testes de QI, foi o primeiro a explorar o raciocínio analógico através da utilização de um programa em execução.

Vários avanços científicos foram alcançados ao perceber o raciocínio por detrás dos sistemas baseados em conhecimento e analogias, tornando-se o gatilho para diversas investigações e desenvolvimentos nas áreas como a medicina, indústria e em áreas como o desenvolvimento de mecanismos de condução autónoma de veículos (Buchanan, 2005). Atualmente os desenvolvimentos em torno da Inteligência Artificial tem como foco as áreas de engenharias, estando associado ao baixo nível de reconhecimento de padrões, assim como, no âmbito estatístico com foco no reconhecimento de padrões, através de dados armazenados, possibilitando o teste de hipóteses assim como apoio à tomada de decisão.

A Inteligência Artificial tem dois objetivos principais, um relacionado com o aspeto tecnológico, em que os computadores chegam a resultados úteis através da integração de métodos de aprendizagem. Por outro lado, através da utilização dos conceitos e dos modelos de Inteligência Artificial e com uma visão em torno de um aspeto científico, o objetivo passa por tentar responder a questões relacionadas com humanos e outros seres vivos, no aspeto em que, devido aos sistemas de IA, a intervenção humana deixa de ser necessária, e o processo de tomada de decisão é com base no resultado dos métodos de IA, como por exemplo na medicina.

Com a evolução das tecnologias de informação, e com o desenvolvimento tecnológico baseado em dados, os modelos de *Machine Learning* são cada vez mais importantes neste contexto, devido à capacidade de modelação de processos computacionais para responder aos desafios da IA. Assim sendo, surge o conceito de *Machine Learning* que irá responder ao subconjunto de objetivos da IA com foco no desenvolvimento de máquinas inteligentes (Arrieta et al., 2020).

2.4 Machine Learning

Machine Learning tem vindo a ser desenvolvido desde meados do século XX, no qual se acreditava na possibilidade de um computador aprender novos padrões ou regras, estando ele programado para essa função. Tendo este conceito de *Machine Learning* ter sido descrito pela primeira vez por *Arthur Samuel*, em 1959, da seguinte forma: *"Campo de estudo que dá aos computadores a capacidade de aprender sem serem especificamente programados"* (Samuel, 1967). Mais tarde, por *Tom Mitchell*, em 1997, que atribuiu uma definição mais formal a este conceito: *"É dito a um programa de computador para aprender através de uma experiência E relativamente a uma tarefa T e da medição da performance P, se a performance em T, medida através de P, melhora através da experiência E"* (Mitchell and Mitchell, 1997).

O conceito de *Machine Learning*, sendo este uma vertente da Inteligência Artificial, tem como principal objetivo a implementação de capacidades de aprendizagem nas máquinas, através do estudo de técnicas de modelação de processos computacionais, que permitam alcançar soluções para diversos problemas num vasto conjunto de áreas, que métodos convencionais não permitem resolver.

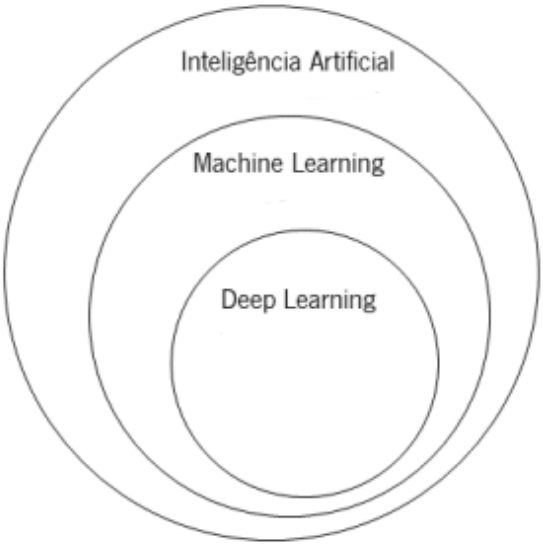

Figura 2: Relacionamento da Inteligênica Artificial com *Machine Learning* e *Deep Learning* (adaptado de Goodfellow, 2016)

Num contexto de *Machine Learning*, os métodos de análise de dados que automatizam o processo de construção de modelos analíticos, têm por base um algoritmo que recebe os dados, na qual as abordagens à análise variam entre si, dependendo do tipo de dados recebidos. Através do processamento dos mesmos, o sistema é capaz de identificar padrões nos dados adquiridos, dando suporte ao processo de tomada de decisão através da previsão de situações futuras, com o mínimo de intervenção humana (Mechelli and Viera, 2019).

Assim sendo, os algoritmos usados em *Machine Learning* utilizam conjuntos de dados históricos, com o intuito de adaptar o modelo, ou por outras palavras, treinar o modelo, com o objetivo de obter previsões no âmbito do problema, que se pretende solucionar. Este processo de concessão de um modelo de *Machine Learning*, independentemente do modelo a ser treinado, de forma geral, segue uma metodologia dividida em seis diferentes etapas, possibilitando a criação de um modelo robusto, capaz de criar valor, no âmbito do problema (Alzubi et al., 2018).

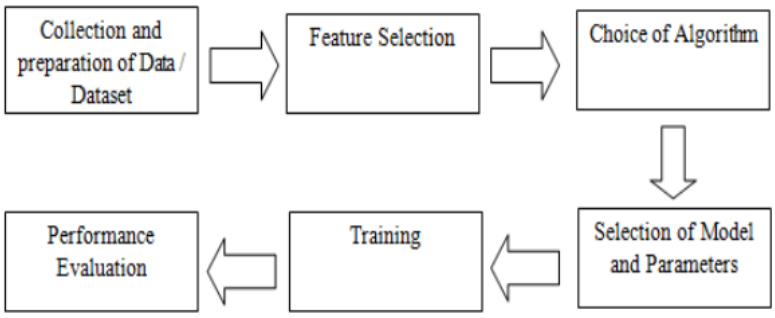

Figura 3: Etapas do desenvolvimento de um modelo de *Machine Learning* (adaptado de Alzubi et al., 2018)

- *Collection and Data Preparation* (Recolha e Preparação dos Dados): Esta fase consiste na recolha dos dados, que irão servir como input do modelo, os quais devem ser devidamente tratados, de modo a construir uma base sólida para as fases seguintes;
- *Feature Selection* (Seleção de Atributos): Após os dados serem recolhidos e preparados, existe a possibilidade de existirem atributos desnecessários, os quais devem ser removidos, ficando apenas com os atributos relevantes para a conceção do modelo;
- *Selection of Models and Paramaters* (Seleção dos Modelos e Parâmetros): Seleção dos algoritmos que permitam responder ao problema da melhor forma, e customização dos parâmetros dos mesmos;
- *Training* (Treino): Etapa na qual os dados recolhidos são utilizados para treino no modelo;
- *Performance Evaluation* (Avaliação da Performance): Após o treino do modelo, é necessário avaliar o desempenho do modelo, através de métricas, antes do modelo ser implementado.

2.4.1 Aprendizagem Supervisionada

O contexto de *Machine Learning* tem diversos tipos de aprendizagem subjacentes, podendo ser feito tanto de uma forma supervisionada como não supervisionada. A Aprendizagem Supervisionada consiste num sistema, que adquire um conjunto de dados com diversos exemplos de valores e decisões, na qual a partir de uma fórmula matemática, mapeia variáveis de *input* para variáveis de *output*. O principal objetivo passa por adequar e aperfeiçoar a função, de modo a fazer previsões para pontos desconhecidos, isto é, quando forem introduzidos novos *inputs*, o sistema será capaz de prever os *outputs* (Mohri et al.,

2018). Conforme os *inputs* são introduzidos no modelo, este ajusta os valores até que o modelo tenha ajustado adequadamente.

A Aprendizagem Supervisionada auxilia as organizações na resolução de diversos problemas do mundo real a grande escala. Sendo que existem diferentes tipos de problemas dependendo do *output*, como problemas de classificação, quando são variáveis discretas não ordenadas, ou então problemas de regressão, quando o resultado é contínuo.

Uma tarefa da Aprendizagem Supervisionada é designada de regressão quando os *outputs* são valores numéricos, e chamados de classificação quando os *outputs* tomam valores categóricos ou discretos. Por exemplo, quando se tenta prever um preço de uma televisão a partir de um *dataset*, este é um caso de regressão, pois o preço vai ser um resultado contínuo. Analisando o mesmo *dataset* sobre televisões, seria um problema de classificação, no caso do algoritmo prever se o preço vai ser superior ou inferior ao preço de custo, ou seja, o *output* só pode ser entre duas categorias (Matloff, 2017).

Exemplos de alguns algoritmos mais comuns de regressão são, *Linear Regression*, *Support Vector Machine (SVM)* e *Regression Tree*. Quanto a algoritmos de classificação, alguns exemplos comuns são, *Naive Bayes*, *Decision Trees* e *K Nearest Neighbors (KNN)*, (Hastie et al., 2009):

- Linear Regression: A regressão linear já existe há muito tempo, embora possa parecer uma abordagem monótona comparada às abordagens mais recentes, é uma ferramenta simples de implementar, e continua a desempenhar um papel bastante útil na análise estatística (Su et al., 2012);

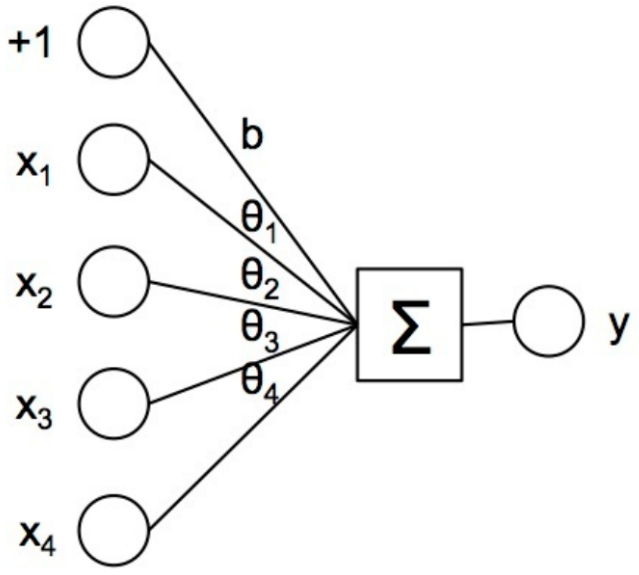

Figura 4: Arquitetura de um modelo de *Linear Regression* (adaptado de Briot et al., 2017)

- Support Vector Machine (SVM): Sendo este um modelo mais complexo em comparação com outros modelos, por outro lado, permite obter resultados com elevada

precisão. Pode ser usado tanto em problemas de regressão ou classificação. Este é caracterizado por um hiperplano num espaço com *N*-dimensões (no qual *N* representa o número de atributos), de modo a classificar distintamente os pontos de dados, existindo um vasto número de possibilidades no qual o hiperplano pode ser escolhido (Chauhan et al., 2019);

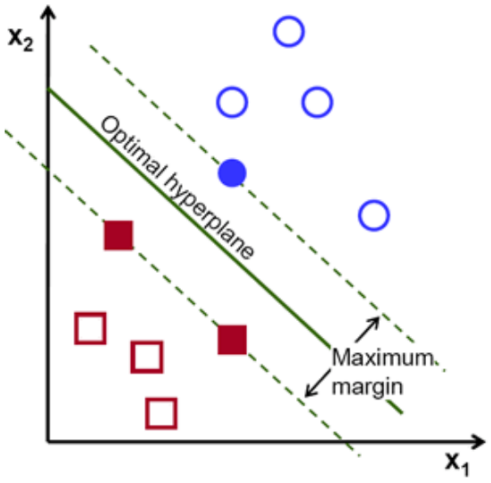

Figura 5: Conceito de hiperplano no modelo *SVM* (adaptado de Agarap, 2018)

- Decision Trees: Sendo esta técnica uma das mais comuns dentro da Aprendizagem Supervisionada, este modelo é caracterizado pela construção de uma árvore, na qual a sua raiz consiste na representação do conjunto de dados, e os diversos ramos representam uma decisão. Este algoritmo é bastante utilizado em contextos de previsão, tanto em problemas de classificação ou regressão, diminuindo a complexidade, simplificando o processo de decisão (Myles et al., 2004).

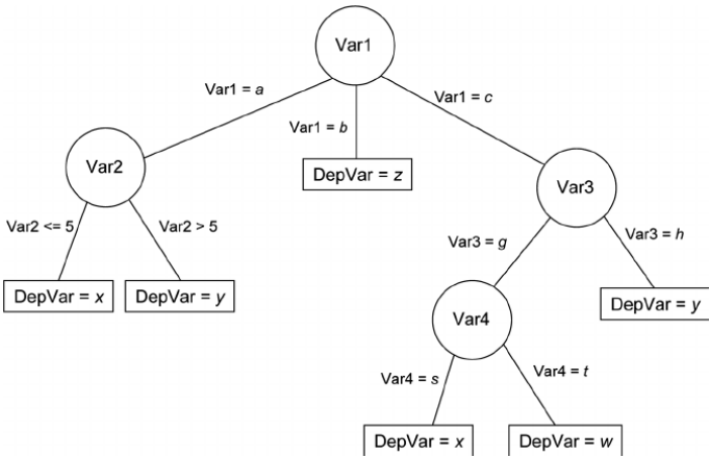

Figura 6: Arquitetura geral de um modelo de *Decision Tree* (adaptado de Iorkyase et al., 2019)

- Regression Tree: É um modelo de *Machine Learning*, que possibilita a criação de modelos de previsão com base nos dados. O modelo é obtido através do particionamento recursivo dos dados, estabelecendo uma simples previsão a cada partição. Desta forma, as *Regression Tree* são utilizadas no âmbito da existência de variáveis dependentes, constituídas por valores contínuos, no qual o erro da previsão é calculado através da diferença entre o quadrado do valor observado e do quadrado do valor previsto (Loh, 2011);

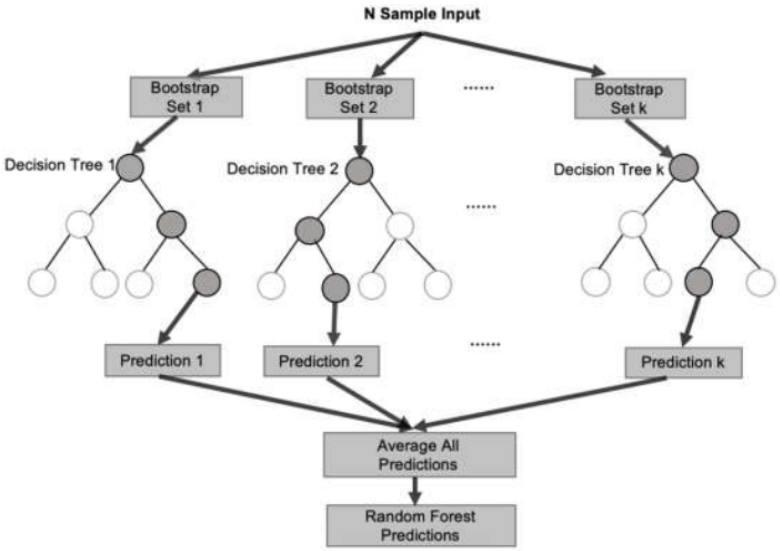

Figura 7: Arquitetura de um modelo *Regression Tree* (adaptado de Murray and Scime, 2010)

- Naive Bayes: O conceito deste modelo consiste em utilizar a regra de *Bayes* juntamente com atributos que são condicionalmente independentes entre si. Assim sendo, apesar desta independência por vezes não ser praticada, o modelo é capaz de obter bons resultados preditivos comparado a outros modelos (Webb et al., 2010).

2.4.2 Aprendizagem Não-Supervisionada

Numa outra abordagem, a Aprendizagem Não-Supervisionada o principal objetivo do sistema consiste em identificar padrões ou propriedades de um conjunto de dados não rotulados, extraindo o maior conhecimento possível dos mesmos, que ao contrário da Aprendizagem Supervisionada, o sistema não é responsável por identificar um *output*, (Mohri et al., 2018).

Esta aprendizagem trata apenas dos dados de *input* e nenhuma variável de *output*. O principal objetivo passa por adequar a estrutura do modelo ou a distribuição dos dados com o intuito de os perceber melhor. Numa fase inicial, esta abordagem apresenta um

desempenho limitado, mas à medida que aprende mais sobre os dados, o desempenho aumenta consequentemente.

Dois dos principais métodos usados na aprendizagem não supervisionada são a componente principal e a análise de agrupamento. Um exemplo é a análise de *clusters* que é utilizada na aprendizagem não supervisionada, para criar conjuntos de dados com atributos comuns, e fomentar os relacionamentos algorítmicos. A análise de *clusters* é um domínio que agrupa os dados que não foram rotulados, classificados ou categorizados (Jain, 2010).

2.4.3 Reinforcement Learning

O *Reinforcement Learning* consiste numa das áreas do *Machine Learning* que lida com os processos de tomada de decisão ao longo de uma determinada sequência de acontecimentos.

Este conceito pode ser formalizado como um agente que toma determinadas ações com base numa determinada recompensa, dentro de um ambiente. Este ambiente é formulado como um processo de decisão de *Markov*, no qual o agente interage com um ambiente dinâmico através de tentativa e erro. O algoritmo inicialmente desconhece a solução ótima, mas o agente através de um sistema de recompensas, com base em experiências, é capaz de descobrir quais as ações mais benéficas dentro do ambiente ao longo do tempo. Um aspeto fundamental é que o agente não necessita de conhecer o ambiente, apenas é necessário que este consiga interagir com o ambiente e recolher informações ao longo das diversas interações. Assim sendo, é possível identificar 4 elementos principais num sistema de *Reinforcement Learning* (Sutton and Barto, 2018):

- *Policy*: Consiste no processo de decisão do algoritmo, traduzido numa função que recebe o estado atual e determina uma ação a executar.
- *Reward*: O agente em cada iteração recebe uma recompensa, e este, ao longo das várias experiências de tentativa e erro, procura maximizar a *reward* de acordo com o objetivo do problema de *Reinforcement Learning*.
- *Mode*: Descreve as físicas do ambiente e como este se comporta, dando informações necessárias ao algoritmo para ajudar a definir as ações futuras.
- *Value*: À medida que o agente vai recolhendo informações relativamente ao ambiente, esta função indica quais as ações que devem ser tomadas de acordo com o estado em que o agente se encontra.

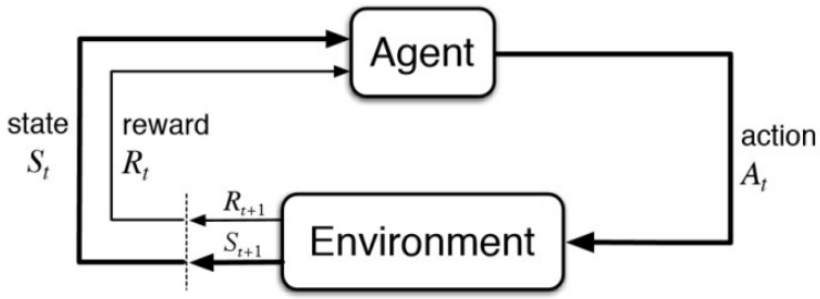

Figura 8: Arquitetura de um modelo *Reinforcement Learning* (adaptado de Nasir et al., 2021)

2.4.4 Automated Machine Learning (AutoML)

Na área de *Machine Learning*, as tarefas mais desafiantes consistem na seleção do modelo a utilizar, assim como, da otimização de hiperparâmetros, juntamente com os atributos do conjunto de dados a ser utilizado.

Analisando a metodologia CRISP-DM, o desenvolvimento de uma solução baseada em *Machine Learning* está dividido em várias etapas, com especial foco na preparação dos dados e na modelação. Caracterizados por serem os processos mais demorosos, no qual é essencial tomar boas decisões relativamente à melhor configuração do modelo, que por norma, nem sempre é esse o desfecho, consumindo bastante tempo útil ao investigador, o ciclo do CRISP-DM poderá iterar diversas vezes, em busca dos melhores resultados, (Feurer et al., 2015).

Desta forma, o conceito de AutoML surge com o objetivo de facilitar o processo de tomada de decisão do utilizador, no que diz respeito à seleção do algoritmo a utilizar, assim como, da configuração dos seus parâmetros (He et al., 2021). Assim, do vasto conjunto de algoritmos disponíveis para dar resposta ao problema e das diversas possibilidades de configuração dos mesmos, o AutoML, tem a função de gerar automaticamente os modelos, assim como as suas configurações. Deste modo, independente do conhecimento do investigador relativamente às características dos modelos, será possível obter bons resultados preditivos, com relativa facilidade através da utilização de uma ferramenta como o AutoML (Ferreira et al., 2021).

Podemos afirmar, que o AutoML permite facilitar o processo associado ao desenvolvimento de soluções, com base no conceito de *Machine Learning*, através da automatização da seleção do modelo adequado, sem a necessidade de intervenção humana. Posto isto, a definição do conceito de AutoML consiste no processo de automatização da aplicação de *Machine Learning*.

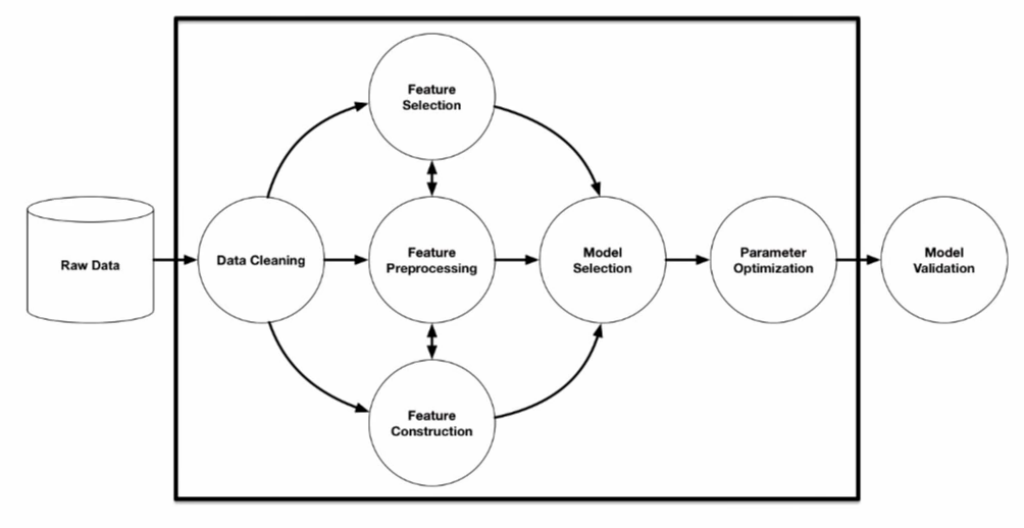

Figura 9: Pipeline *AutoML* (adaptado de Olson et al., 2016)

Com o intuito da aplicação deste conceito, diversas ferramentas de AutoML, foram desenvolvidas:

- TPOT: O *TPOT (Tree-based Pipeline Optimization Tool)* é uma ferramenta *open source*, que permite otimizar uma série de *features* associadas ao pré-processamento de dados, e selecionar um conjunto de modelos com o objetivo de maximizar a eficácia dos mesmos. Assim sendo, o TPOT tem como objetivo a automatização da pipeline de *Machine Learning* face a um problema, sem a necessidade de intervenção humana (Olson and Moore, 2016). utiliza uma versão de *Genetic Programing*, que consiste numa técnica automática de programação, que permite a evolução de programas de computadores, que resolvem problemas.

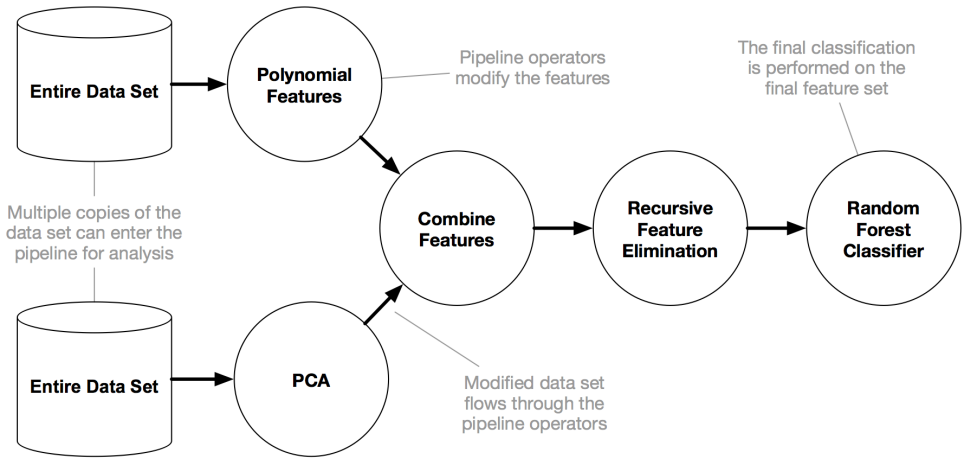

Figura 10: Pipeline *TPOT* (adaptado de Olson et al., 2016)

- H20 AutoML: Esta ferramenta consiste na automatização do processo de *Machine Learning*. O *H20 AutoML*, é composto por diversos algoritmos como *XGBoost*, *Generalized Linear Model (GLM)* , *SVM*, *Deep Neural Net*, *etc.*, o qual permite automatizar o *workflow*, no qual o investigador poderá configurar parâmetros associados ao *H20 AutoML*, dado que, o número de *fold* para a *cross-validation*, o tempo de execução, as métricas de avaliação, entre outros. Esta ferramenta é *open source*, estando operacional em ambientes *Python* e *R*, permitindo o desenvolvimento de uma solução automatizada e escalável;

2.4.5 Séries Temporais

O crescente aumento de grandes quantidades de dados históricos ao longo dos últimos anos, aliado à necessidade de realizar previsões de comportamentos no futuro, surge o conceito de séries temporais, ou em inglês, *Time-Series*.

Tem como base a ideia de que é possível obter padrões nos conjuntos de dados históricos, permitindo criar previsões futuras, de acordo com o âmbito dos dados. É através das circunstâncias em que os padrões foram descobertos que é possível criar essas previsões e perceber qual o tipo de mudanças que esses podem sofrer ao longo do tempo (Parmezan et al., 2019). Inicialmente este conceito começou por ser desenvolvido, a partir da década de 1960 através de métodos estatísticos lineares (Bontempi et al., 2012).

Nos últimos anos, diversos estudos têm sido desenvolvidos no âmbito da modelação de algoritmos de *Machine Learning*, para a previsão de séries temporais. A principal vantagem associada a este conceito consiste na natureza da distribuição dos dados. Deste modo, os algoritmos de *Machine Learning* para a previsão de séries temporais, na maior parte das aplicações, acaba por superar os modelos estatísticos (Parmezan et al., 2019).

No âmbito da previsão de séries temporais, correspondentes à previsão de vendas, existe um problema comum associado à falta de dados, dados nulos, ou mesmo *outliers*. Dessa forma diferentes modelos foram desenvolvidos para colmatar esta dificuldade, como por exemplo *ARIMA*, *LSTM*, *Holt-Winters*, entre outros:

- *ARIMA* (*Auto-Regressive Integrated Moving Average)*: Esta ferramenta consiste num modelo que permite criar previsões com base em dados históricos. Este modelo é uma generalização do modelo *ARMA (Autoregressive Moving Average)*, que é resultado da combinação de dois processos, *Autoregressive (AR)* e *Moving Average (MA)*, gerando o acrónimo *ARIMA (p, d, q)*. O *p* (*AR* de *Autoregressive*), consiste no modelo de regressão que usa as dependências entre uma observação e o número de latência utilizada no treino. O *d* (*I* de *Integrated*) refere-se à medição da diferença do número de observações em diferentes intervalos de tempo. Por fim, o *q* (*MA* de *Moving Average*) tem em conta

a dependência entre as observações e os erros residuais, defenindo o tamanho da janela da *moving average* (Siami-Namini et al., 2018);

- *LSTM*: *Long Short-Term Memory* é um tipo especial de uma *Recurrent Neural Network (RNN)* com características adicionais de modo a memorizar a sequência do tempo. Cada *LSTM* consiste num conjunto de células, onde os fluxos de dados são guardados. Uma célula conecta dois módulos, transportando os dados entre os dois. Esta também é composta por *gates*, baseados na camada de uma rede neuronal, que permite ao módulo armazenar ou descartar os dados. Assim sendo, uma *LSTM* pode armazenar e aprender através de longas sequências de dados (LeCun et al., 2015);

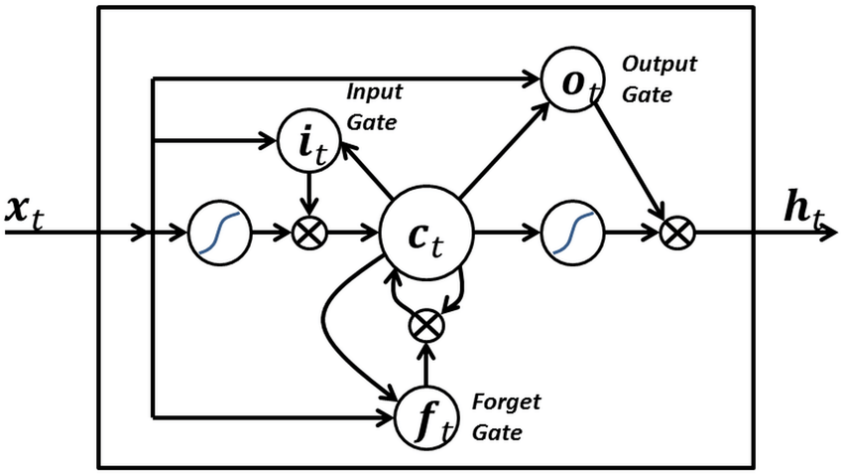

Figura 11: Arquitetura de um *LSTM gate* com um *input, output* e um *forget gate* adaptado de Zaytar and El Amrani, 2016)

- Holt-Winters: Esta ferramenta consiste num método de séries temporais, onde os resultados preditivos são melhores quando os conjuntos de dados a serem usados como *input* para o treino do modelo, são compostos por algum padrão de sazonalidade, devido à utilização de uma suavização exponencial, à medida que os dados são consumidos pelo modelo (Gelper et al., 2010).

2.5 Machine Learning no Planeamento de Cadeias de Abastecimento

As ferramentas de gestão das cadeias de abastecimento, tem permitido às empresas criarem os seus planos de produção e fornecimento de acordo com os recursos disponíveis no momento, com a capacidade de reagir rapidamente às oscilações no mercado. Um dos principais inibidores de uma cadeia de abastecimento responsiva, consiste em planos de

produção parados, que se estendem por um determinado intervalo de tempo, devido a atrasos no fornecimento de um material específico.

As tecnologias de *Supply Chain Planning* permitem organizar os planos de produção de curto prazo, de forma, que estes estejam sincronizados. No planeamento simultâneo, o plano de produção engloba o tempo de configuração para cada máquina, assim como a capacidade de produção.

A introdução do *Machine Learning* nas *Supply Chain* tem como principal objetivo, descobrir padrões nos dados obtidos ao longo de toda a cadeia, permitindo elaborar algoritmos capazes de identificar quais os fatores determinantes para o sucesso da *Supply Chain*. Este processo tem permitido revolucionar toda a indústria de produção, alcançando resultados significativos de melhoria. Neste caso, relativamente à cadeia de abastecimento, os principais fatores que geralmente tem maior impacto consistem nos níveis de stock, qualidade do fornecedor, previsão da procura, planeamento da produção e a gestão do transporte. Assim sendo, os algoritmos de *Machine Learning*, irão contribuir para descobrir *insights* que proporcionem a otimização da *Supply Chain* (Makkar et al., 2019) :

- Algoritmos de *Machine Learning*, serão capazes de analisar um grande volume de dados melhorando a precisão da procura;
- Otimização do transporte das mercadorias de ponta a ponta, reduzindo custos associados e minimizando risco nos atrasos;
- Capacidade de reconhecimento de padrões visuais que possam indicar defeitos ou danos nos produtos, abrindo um conjunto de possibilidades como a manutenção preventiva;
- Através da junção do *Machine Learning*, com tecnologias relacionadas com as operações na *Supply Chain*, é possível reduzir custos de stock e das diversas operações, assim como, reduzir os tempos de resposta ao cliente;
- É possível prever a procura no mercado por novos produtos, assim como, identificar os fatores que contribuíram para o mesmo;
- A vida útil dos principais ativos da empresa, tem sido ampliada, através da análise de padrões nos dados coletados, que tornam possível detetar quais os fatores que determinam, por exemplo, o desempenho das máquinas através de métricas mais precisas, como Eficácia Geral do Equipamento;
- Melhorar a gestão da cadeia de abastecimento, através da conformidade entre fornecedores e da busca de padrões de qualidade graças ao *Machine Learning*.

A implementação da Inteligência Artificial, mais concretamente de *Machine Learning* nas cadeias de abastecimento, permite melhorar o processo de planeamento de produção, com

um elevado nível de precisão, relativamente ao agendamento da fábrica, elaborando um conjunto de restrições e também, determinando um conjunto de fatores, que deverão ser otimizados. A combinação do *Machine Learning*, com tecnologias avançadas de análise de dados, tem fornecido uma visibilidade de ponta a ponta aos gestores, sobre toda a Supply Chain.

Em suma, a introdução de algoritmos de *Machine Learning* (descritivos, preventivos ou prescritivos), juntamente com o planeamento concorrente, poderá melhorar a eficácia do planeamento, devido à possibilidade de definir as ações a desenvolver com base em dados históricos da organização e desta forma, potenciar os lucros, diminuir o *lead-time*, desperdício e custos associados ao processos de produção, prever a oferta e procura no mercado, com um objetivo global de satisfazer o cliente e alcançar uma produção sustentável.

Tabela 1: Síntese da revisão da literatura.

Autor	***Título***	***Objetivo***	***Métodos Utilizados***
(Feurer et al., 2015)	*Efficient and Robust Automated Machine Learning.*	*Desenvolver um sistema baseado em, AutoML, biblioteca Scikit-Learn, dando origem ao nome Auto-Sklearn. Melhora os métodos existentes de AutoML ao ter em conta, automaticamente, o desempenho dos modelos anterior.*	*Hyperopt-Sklearn Auto-Sklearn Auto-Weka*
(Ferreira et al., 2021)	*A Comparison of AutoML Tools for Machine Learning, Deep Learning and XGBoost.*	*Desenvolvimento de um benchmark de ferramentas de AutoML para o uso em tarefas de Machine Learning, como classificação e regressão.*	*AutoGluon H2O AutoML TPOT*
(He et al., 2021)	*AutoML: A Survey of the State-of-the-Art.*	*Revisão da literatura relativamente à pipeline de AutoML, desde a fase do pré-processamento dos dados até à fase de avaliação do modelo*	*AutoML*

Autor	*Título*	*Objetivo*	*Métodos Utilizados*
(Parmezan et al., 2019)	*Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model*	*Descrição dos principais algoritmos preditivos, no âmbito do desenvolvimento de modelos de previsão de séries temporais.*	*ARIMA SARIMA SVM*
(Olson and Moore, 2016)	*Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science.*	*Desenvolvimento de modelos preditivos, com base na ferramenta TPOT, no âmbito de AutoML .*	*TPOT*
(Siami-Namini et al., 2018)	*A comparison of ARIMA and LSTM in forecasting time series.*	*Comparação entre algoritmos preditivos, em relação à eficácia do modelos desenvolvidos para tarefas de séries temporais.*	*ARIMA LSTM*

2.6 Ferramentas Tecnológicas

Com o objetivo de contextualizar o trabalho realizado ao longo desta dissertação, é importante identificar as ferramentas utilizadas, que permitiram alcançar os objetivos propostos.

O código desenvolvido foi totalmente em `Python`, devido à elevada relevância desta ferramenta para o desenvolvimento de projeto de *Machine Learning*. Associada a esta linguagem de programação, destacam-se quantidade de bibliotecas disponíveis para as etapas do CRISP-DM. Na tabela 2 podemos observar os *packages* mais relevantes para esta dissertação.

Tabela 2: Módulos `Python` utilizados neste trabalho.

Nome	Descrição	Licença	Área	Versão Python
Anaconda	A plataforma Anaconda, consiste numa ferramenta desenvolvida para o âmbito da *data science*, oferecendo um conjunto de aplicações *open source*, as quais permitem a implementação de tecnologias relacionas com *Machine Learning*. No desenvolver desta dissertação foi utilizado o *IDE Jupyter Notebook*, com a versão 6.3.0 para o desenvolvimento do código em *Python*.	BSD 3-Clause	Suporte Desenvolvimento	>=2.7
AutoGluon	A biblioteca *AutoGluon* permite o desenvolvimento de modelos de AutoML, fáceis de usar com especial foco na automatização de modelos no âmbito de dados tabulares, texto ou imagens. Desenhado para a utilização de iniciantes, na área de *Machine Learning* através da automatização dos hiperparâmetros, seleção de modelos e processamento de dados.	MIT License	Modelação	>=0.5
Cane	Cane ou *Categorical Attribute Transformation Enviorenment* consiste num *package*, criado para o âmbito do pré-processamento dos dados. Este é composto por diversas técnicas como o IDF *(Inverse Document Frequency)*, *One-hot Encondig* e PCP *(Percentage Categorical Pruned*, onde é possível transformar variaveis categóricas em valores numéricos	MIT License	Pré-Processamento	>=3.6

Nome	Descrição	Licença	Área	Versão Python
H2O AutoML	H20 AutoML, consiste numa ferramenta, que permite automatizar o fluxo de trabalho de *Machine Learning*, ao incluir a seleção dos modelos e os respetivos hiperparâmetros, de modo a facilitar o processo de treino dos modelos. O utilizador desta ferramenta pode ainda definir o período de tempo de treino nos modelos. Esta ferramenta, oferece também uma interface gráfica, que foi projetada para ter o mínimo de parâmetros possíveis, de modo a que o utilizador, apenas necessita de definir o conjunto de dados, identificar o *target*, e o tempo de treino.	MIT License	Modelação	>=3.6
Matplotlib	Sendo esta biblioteca uma extensão da biblioteca *NumPy*, que suporta *Python* da versão 2.7 à 3.6, tem como objetivo a criação de gráficos e visualização dos dados.	Python Software Foundation License	*Análise*	>=2.7
Numpy	A biblioteca Numpy *(Numerical Python)*, consiste, numa ferramenta desenvolvida para *Python*, composta por um conjunto de funções matemáticas para operar sobre matrizes. Esta biblioteca, suporta um elevado processamento de *arrays* multidimensionais.	BSD	Análise Pré- -Processamento	>=3.6
Pandas	A biblioteca *Pandas* foi criada para o processamento e análise de dados, em *Python*. A estrutura da ferramenta e das operações disponíveis, tem como objetivo a manipulação de tabelas numéricas e de séries temporais.	BSD 3-Clause	Análise Pré- -Processamento	>=2.7

Nome	Descrição	Licença	Área	Versão Python
Plotly	O plotly consiste numa ferramenta *open source*, que permite o desenvolvimento de gráficos iterativos.	MIT License	Analytics	>=2.7
Scikit-learn	O *Skikit-learn*, consiste numa biblioteca, para o uso no desenvolvimento de projetos, no âmbito de *Machine Learning*. É uma ferramenta *open source* para uso em *Python*. Incluí um conjunto de algoritmos de classificação como de regressão. Outra característica desta biblioteca é que esta foi desenvolvida, com o intuito de interagir com as bibliotecas *NumPy* e *SciPy*.	BSD 3-Clause	Pré- -Processamento Modelação	>=3.6

3

Caso de Estudo - Previsão de Encomendas

Com o crescente desenvolvimento das tecnologias no meio industrial, e a ambição das empresas criarem valor, inerente ao desenvolvimento do produto, estas tem procurado soluções através de ferramentas no âmbito da Inteligência Artificial.

Este tipo de abordagem permite às organizações descobrir novos *insights*, que permitam desenvolver novas estratégias, sustentadas em dados históricos, de modo a tomarem decisões, baseadas em previsões do futuro da organização.

Relativamente a este caso de estudo, desenvolvido com base no projeto *Produtech4S&C: Produtech Sustentável e Circular*, mais concretamente no *PSS4 – Cadeia digital de fornecimento em contexto circular*, uma das empresas parceiras do projeto, que atua no meio da Indústria de Manufatura, tem como objetivo, prever o número de encomendas que irá receber num determinado período de tempo, de forma a facilitar o desenvolvimento de um plano produtivo por parte dos gestores. Ao antecipar o número de encomendas que irão receber, através de ferramentas de *Machine Learning*, questões como a compra de matéria-prima, gestão de stocks e previsão da entrega das encomendas, poderão ser resolvidas antecipadamente, permitindo assim, à empresa otimizar os seus processos produtivos.

Assim sendo, este caso de estudo consiste no desenvolvimento de modelos preditivos de *Time Series*, seguindo a metodologia CRISP-DM (*CRoss Industry Standard Process for Data Mining*), para a previsão do número de encomendas. Com este objetivo definido, foram utilizadas ferramentas *open-source* no âmbito de *Machine Learning* como, *Auto-ARIMA*, *Prophet* e *Neural-Prophet* para o desenvolvimento das previsões.

3.1 Iteração 1

Este caso de estudo consiste, num problema que utilizará técnicas de *Machine Learning* para a sua resolução, portanto, ao longo desta primeira iteração da metodologia CRIPS-DM, o objetivo consiste na previsão do número de encomendas.

3.1.1 Compreensão do Negócio

A empresa parceira ao projeto *Produtech4S&C*, inserida na indústria de produção, é composta por quatro fábricas, localizadas em diversos países, sendo o seu processo produtivo principal, a litografia. Este processo, consiste na transformação da matéria-prima utilizada no processo de litografia, no produto final, latas ou recipientes construídos à base de folhas de metal.

Inserida num mercado bastante competitivo, onde diversos fatores refletem o preço final do produto, esta empresa tem como objetivo, no âmbito do desenvolvimento do projeto, obter previsões do número de encomendas que irão receber num determinado período de tempo. De modo a desenvolver um plano de produção interno, que englobe diversos fatores, como a gestão de stocks, ou por exemplo, a compra de matérias-primas, com perspetivas no futuro face ao número de encomendas que irão receber, e consequentemente, o número de unidades de latas que irão ter de produzir.

A interesse da empresa associada ao projeto, e como requisito do mesmo, o número de dias para qual estas previsões deverão ser realizadas foram delineadas pela mesma. Por motivos relacionados com o planeamento interno da empresa, os requisitos passam por prever o número de encomendas de recipiente de metal, que irão receber nos próximos 7/14/21 e 28 dias (dias não-úteis incluídos), possibilitando assim, sustentar as tomadas de decisão no âmbito do planeamento produtivo, com base nos resultados obtidos, através de métodos de *Machine Learning*, desenvolvidos através do histórico de encomendas da mesma.

3.1.2 Compreensão dos Dados

Nesta fase, foi analisado um *dataset* disponibilizado pela empresa, relativo às encomendas recebidas num período de um ano. O *dataset Orders Recived*, consiste num conjunto de dados composto por 81648 registos, com datas entre 01-01-2020 e 31-12-2020. Estes registos referem-se às encomendas de recipiente de metal, por diversos clientes.

Desta forma, este *dataset* é composto por 8 colunas: *Sold-to Party, Name 1, Sales Document, Purchase Order No, Document Date, Delivery Data, Order Quantity* e *Factory*. Por motivos de interesse da empresa, os dados dos clientes serão mantidos anonimizados. Na tabela abaixo podemos ver o resumos dos atributos do *dataset*:

Tabela 3: Descrição dos atributos do *dataset Orders Recived*

Atributo	Descrição	Tipo	Distintos	Exemplo
Sold-to Party	Código do cliente	Numérico	218	Anónimo
Name 1	Nome do cliente	Categórico	218	Anónimo
Sales Document	Ndo documento da encomenda	Numérico	15278	Anónimo
Purchase Order	Código do documento de faturação	Categórico	10902	Anónimo
Document Date	Data da encomenda	Data	281	02/01/2020
Delivery Date	Data de entrega	Data	451	07/02/2020
Order Quantity	Quantidade encomendada	Numérico	7925	6137
Factory	Fábrica onde a encomenda foi realizada	Categórico	4	Anónimo

3.1.3 Preparação dos Dados

A fase da preparação dos dados, tem como objetivo a estruturação do *dataset*, de modo a que este possa servir como *input* para os modelos preditivos de *Time Series*.

Inicialmente foram removidas todas as colunas desnecessárias para a elaboração dos modelos, ficando apenas com os atributos, *Document Date* e *Order Quantity*. Concluído este passo, o *dataset* ficou reduzido apenas a duas colunas: *Document Date*, atributo relativo à data da encomenda, e à variável *target, Order Quantity*, relativa à quantidade de produto encomendada.

Com o objetivo de criar uma série temporal, os valores da coluna *Document Date*, foram agregados pela data, e os valores da coluna *Order Quantity* somados conforme o dia correspondente. Os dados recebidos consistem nas encomendas ao longo do ano de 2020. Como é possível analisar, num período de um ano, 366 dias, apenas existem 281 registo temporais diferentes, isto significa, que em 85 dias ao longo do ano, não existem quaisquer registos de encomendas. De modo a criar uma série temporal contínua no tempo, foi necessário criar novos registos, preenchendo a coluna *Document Date* com os dias em falta e a respetiva data. Como a falta de registos corresponde à ausência de encomendas, o valor da coluna *Order Quantity* foi preenchido com o valor 0.

Por fim, após este processo de transformação dos dados, resultou um *dataset* composto por 366 linhas e 2 colunas, correspondentes à quantidade de produto encomendado ao longo dos dias do ano de 2020.

3.1.4 Modelação

A fase de modelação, consiste na etapa onde os modelos de análise preditiva são construídos. Para esse efeito, foram selecionadas três ferramentas que irão permitir a elaboração dos modelos de *Time Series*: *Auto-Arima, Prophet e Neural-Prophet*.

Como é comum na elaboração de um modelo de *Machine Learning*, foi necessário definir o método de aplicação. Comum em processos idênticos, um dos métodos de desenvolvimento

consiste na definição de um *training split* e *test split*, dividindo a totalidade do conjunto de dados em dois subconjuntos, o *training split* para o treino do modelo, com cerca de 70% do volume de dados, e o *test split* com os restantes 30%, de modo a prever o *target* e a avaliar os resultados obtidos pelo modelo. Apesar deste ser um método de prática comum, optou-se pela utilização do método de *Rolling Window*, caracterizado por gerar diversas iterações de treino e teste sobre o conjunto de dados, permitindo assim, treinar e testar vários modelos, aumentando a robustez da tarefa realizada.

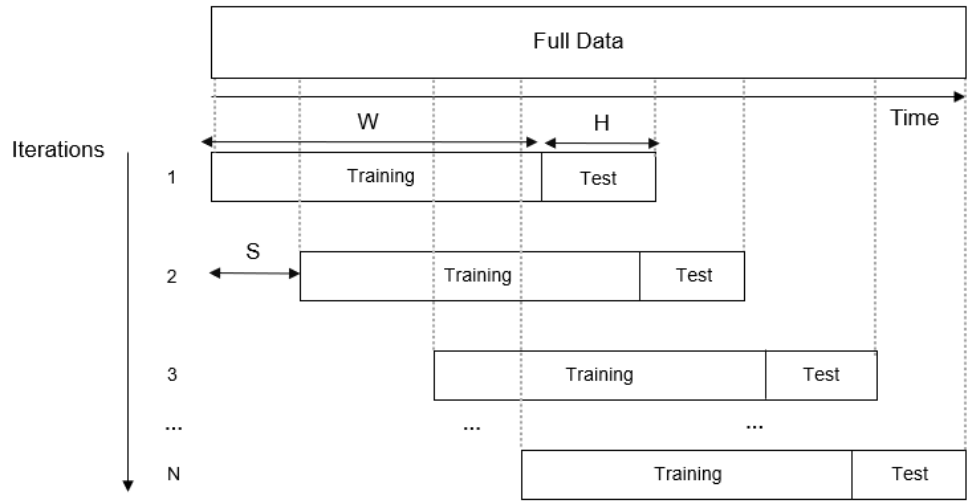

Figura 12: Estrutura do conceito *Rolling Window* (adaptado de Matos et al., 2022)

Na figura acima, podemos observar o comportamento deste método, constituído por três parâmetros:

- W: Tamanho da janela de dados de treino;
- H: Número de previsões (no âmbito do projeto, consiste na quantidade de dias que se pretende prever a quantidade de encomendas);
- S: Número de elementos que serão atualizados a cada iteração da *Rolling Window*

Com o objetivo de desenvolver uma solução robusta, optou-se por realizar 20 iterações da *Rolling Window* (Cortez et al., 2020), sobre o *dataset* resultante da etapa anterior, o qual está ordenado no tempo. O número de iterações é possível obter através da seguinte fórmula:

$$U = \frac{D_{\mathrm{L}} - (W + H)}{H} + 1 \tag{1}$$

Como o número de dias para o qual se pretende obter as previsões não é constante, variando entre 7/14/21 e 28 dias, os valores atribuídos a *W, H* e *S*, foram ligeiramente

alterados à medida que o número de dias de previsão também se alterava, realizando sempre um total de 20 iterações da *Rolling Window*. Por exemplo, para a previsão a 7 dias, num conjunto de dados com 366 registos, para obter um total de 20 iterações, foi atribuído aos parâmetros *W, H* e *S* os valores 192, 8, 8, respetivamente. Por outras palavras, é utilizada uma janela de treino de 192 registos, equivalente a 192 dias, e com uma janela de teste e saltos de 8 valores.

As previsões obtidas a cada iteração da *Rolling Window* foram avaliadas através de duas métricas, o MAE (*Mean Absolute Error* e o NMAE *Normalized Mean Absolute Error*. O MAE, consiste na média da soma das diferenças absolutas, entre o valor real e a previsão associada:

$$MAE = (\frac{1}{n}) \sum_{i=1}^{n} |y_i - x_i| \tag{2}$$

O NMAE, representa o valor em percentagem, do valor do MAE, dividido pela diferença entre o valor máximo e o valor mínimo do *target* no respetivo conjunto de dados. Quanto menor o resultado obtido, melhor é a *performance do modelo*. Esta métrica serve como complemento ao cálculo do MAE, pois dependendo da escala de valor que está a ser utilizada, o resultado obtido pode ser interpretado incorretamente (Oliveira et al., 2017),

$$NMAE = \frac{MAE}{y_{\max} - y_{\min}} * 100 \tag{3}$$

3.1.5 Avaliação

De forma a avaliar os resultados obtidos, através das 20 iterações da *Rolling Window*, da primeira iteração da metodologia CRISP-DM, utilizaram-se as métricas descritas na secção anterior. Os resultados apresentados na tabela abaixo, referem-se à média dos valores das métricas, obtidos ao longo das 20 iterações, para as previsões a 7/14/21 e 28 dias, através das ferramentas selecionadas:

Tabela 4: Média dos resultados obtidos na primeira iteração do CRISP-DM.

Algoritmo Utilizado	Previsão de Dias	MAE	NMAE
Auto-Arima	7	7754.95	19.80%
	14	7975.50	20.12%
	21	8218.59	21.04%
	28	9903.46	25.36%
Prophet	7	5415.86	13.86%
	14	6110.92	15.64%
	21	3482.13	8.92%
	28	3790.82	9.70%
Neural-Prophet	7	1610.25	4.12%
	24	2326.91	5.95%
	21	2472.72	6.32%
	28	3219.08	8.25%

Os resultados obtidos mostram que, nesta iteração do CRISP-DM, o modelo com os melhores resultados foi o *Neural-Prophet*, com um MAE médio 1610.25, por outras palavras, existe um desvio nas previsões de cerca de 1611 unidades por dia, valor arredondado às unidades, o que equivale a um NMAE de 4.12%. Podemos também observar que à medida, que o período temporal da previsão aumenta, a eficácia dos resultados também vai diminuindo, exceto nos resultados obtidos pelo *Prophet*, no qual o inverso acontece.

3.2 Iteração 2

Uma nova iteração do CRISP-DM surge com a perspetiva de melhorar os resultados obtidos anteriormente. Durante reuniões do projeto *Produtech4S&C*, novos *insights* foram proporcionados pela empresa, o que fez despoletar esta iteração, com o intuito de obter melhores resultados.

3.2.1 Compreensão do Negócio

Um dos principais motivos que levou a realizar uma nova iteração consistiu no facto de novos aspetos relevantes sobre o comportamento do negócio terem surgido.

A empresa, ao possuir quatro fábricas distintas, distribuídas em quatro localizações diferentes, cada uma delas, responsável por receber e produzir as suas encomendas, feitas pelos clientes nessas mesmas fábricas. Exceto esta nova condição introduzida pela empresa, os restantes fatores relacionados com o negócio mantiveram-se constantes.

3.2.2 Compreensão dos Dados

Nesta fase o conjunto de dados a analisar é o mesmo que na iteração anterior, o *dataset Orders Recived*. Por conseguinte, não existem novos aspetos para serem detalhados nesta etapa.

3.2.3 Preparação dos Dados

Nesta fase de preparação dos dados, a abordagem foi semelhante à iteração anterior. Face ao novos *insights* recolhidos, o *dataset* foi novamente preparado para potenciar a previsão da quantidade de produtos encomendados por fábrica, através de métodos de *Machine Learning*.

Deste modo, o *dataset Orders Recived* foi dividido em quatro subconjuntos de dados, através da coluna *Factory*. Este atributo possuí apenas quatro valores distintos, cada um deles referentes às fábrica sobre a qual a encomenda é realizada. Cada um dos novos *datasets* sofreu o processamento efetuado na iteração anterior, remoção das colunas desnecessárias para a elaboração do modelo preditivo, ficando apenas com duas colunas, *Document Date* e *Order Quantity*, a qual continua a ser o *target* neste caso de estudo. De forma semelhante, e com o objetivo de criar uma série temporal, o valor das datas foram agregados, somando os valores da quantidade encomenda. Foi necessário criar novos registos com a data em falta na coluna *Document Date*, preenchendo com o valor 0 a coluna *Order Quantity*, pois a ausência de registos no *dataset*, significa ausência de encomendas nesse dia. Por último, os dados foram ordenados no tempo.

Por fim, após este processamento dos dados, resultaram 4 *datasets*, cada um deles constituído por 366 linhas e 2 colunas. Os conjuntos de dados originados foram renomeados, de forma a manter o anonimato dos dados, a pedido da empresa, dando origem aos *datasets Factory A, Factory B, Factory C* e *Factory D*.

3.2.4 Modelação

Para esta fase de modelação, as ferramentas utilizadas para a elaboração dos modelos de *Time Series* foram as mesmas da iteração anterior, *Auto-Arima, Prophet* e *Neural-Prophet*.

Com o objetivo de manter a consistência do trabalho realizado, o método de desenvolvimento passou pela utilização da *Rolling Window*, com os mesmos parâmetros da fase de modelação da iteração 1, mantendo assim, as 20 iterações do método *Rolling Window* ao longo das previsões para 7/14/21 e 28 dias, ao longo dos quatro novos conjuntos de dados. As métricas selecionadas para a fase de avaliação dos modelos também se mantiveram, possibilitando assim a comparação dos resultados entre as duas iterações.

3.2.5 Avaliação

Para avaliar os resultados obtidos em cada um dos *datasets* utilizaram-se as métricas MAE e NMAE, ao longo dos resultados obtidos pelos modelos. Nas tabelas abaixo podemos observar os resultados obtidos, referentes à média dos valores das métricas ao longo do processo de *Rolling Window*, para as previsões de 7/14/21 e 28 dias, por cada um dos algoritmos selecionados, para cada um dos novos *datasets*.

Tabela 5: Média dos resultados da Fábrica A

Dataset	Algoritmo Utilizado	Previsão de Dias	MAE	NMAE
Factory A	Auto-Arima	7	890,45	3.70%
		14	1733.56	7.20%
		21	923.79	3.84%
		28	1097.48	4.56%
	Prophet	7	975.87	4.05%
		14	1137.59	4.72%
		21	915.17	3.80%
		28	922.89	3.83%
	Neural-Prophet	7	332.52	2.05%
		14	447.69	2.26%
		21	573.82	2.34%
		28	701.81	2.62%

Tabela 6: Média dos resultados da Fábrica B

Dataset	Algoritmo Utilizado	Previsão de Dias	MAE	NMAE
Factory B	Auto-Arima	7	2060,00	18.90%
		14	1903.48	17.93%
		21	1805.60	17.01%
		28	2135.20	20.11%
	Prophet	7	1373.91	12.94%
		14	1552.47	14.62%
		21	972.52	9.16%
		28	1060.20	9.99%
	Neural-Prophet	7	625.85	5.89%
		14	625.34	5.88%
		21	668.71	6.29%
		28	916.47	8.63%

Tabela 7: Média dos resultados da Fábrica C

Dataset	Algoritmo Utilizado	Previsão de Dias	MAE	NMAE
Factory C	Auto-Arima	7	7053.32	19%
		14	5434.14	15.05%
		21	5456.28	15.80%
		28	7473.54	20.69%
	Prophet	7	3459.11	9.57%
		14	4224.42	11.70%
		21	2648.51	7.33%
		28	2844.15	7.87%
	Neural-Prophet	7	969.90	2.68%
		14	1547.47	4.28%
		21	1856.09	5.13%
		28	2343.86	6.49%

Tabela 8: Média dos resultados da Fábrica D

Dataset	Algoritmo Utilizado	Previsão de Dias	MAE	NMAE
Factory D	Auto-Arima	7	4189.79	5.54%
		14	5199.90	6,60%
		21	4926.89	6.28%
		28	7012.25	8.94%
	Prophet	7	4100.85	5.22%
		14	4888.95	6.23%
		21	2874.20	7.33%
		28	4079.18	5.20%
	Neural-Prophet	7	1462.07	2.36%
		14	2702.12	3.99%
		21	2875.29	4.16%
		28	3624.56	5.12%

Através da análise da tabela, podemos observar que mais uma vez, o *Neural Prophet*, obteve os melhores resultados para cada uma das quatro fábricas distintas. Para o *dataset Factory A*, o melhor resultado obtido foi a previsão para 7 dias, com um MAE de 333.52 e um NMAE de 2.05%. Para o *dataset Factory B*, o melhor resultado obtido foi a previsão para 14 dias, com um MAE de 625.34 e um NMAE de 5.88%, apesar de muito semelhante à previsão apenas para 7 dias. Relativamente ao *dataset Factory C*, o melhor resultado foi a previsão para 7 dias, com um MAE de 969.90 e um NMAE de 2.68%. Por fim, o *dataset Factory D*, o melhor resultado obtido foi para previsão a 7 dias, com um MAE de 1462.07 e um NMAE de 2.36%.

3.3 Considerações finais

Ao longo das duas iterações podemos observar que o algoritmo *Neural Prophet* proporcionou os melhores resultados na globalidade das previsões, tanto para o *dataset Order Quantity*, como para os *dataset* divididos por fábrica.

Outro aspeto fundamental nesta análise, consistiu na melhoria do valor do NMAE entre as duas iterações. A divisão do conjunto de dados principal por fábrica é um dos principais motivos para esse acontecimento. Sendo que, cada uma das fábrica possuí uma periodicidade entre encomendas diferente, assim como, diferenças a nível das quantidades encomendadas, a divisão do *dataset*, foi benéfica para os resultados obtidos pelo modelo. Este processamento de dados permitiu aos modelos a detenção de padrões temporais específicos, para cada uma das fábricas, traduzindo-se na melhoria dos resultados.

Neste caso de estudo é importante realçar a diferença dos resultados entre cada tipo de previsão. Como seria expectável, quanto menor o espaço temporal da previsão, menor seria o erro. Outro aspeto relevante neste caso de estudo, é o facto de o conjunto de dados possuir registos de apenas um ano, traduzindo-se em apenas 366 linhas, no qual, padrões relevantes, como a sazonalidade anual, não foram possíveis de obter.

Podemos concluir que a segunda iteração da metodologia permitiu obter melhores resultados preditivos, especificamente para cada uma das fábricas.

No âmbito deste projeto não foi possível passar à fase de implementação dos modelos na infraestrutura tecnológica da empresa. Assim, o trabalho desenvolvido teve como objetivo a apresentação dos contributos alcançados, através das experiências realizadas, provando a eficácia dos modelos utilizados, para a previsão do número de encomendas a 7/14/21 e 28 dias. No decorrer do projeto *Produtech4S&C*, existirá a probabilidade de implementar os modelos preditivos, nas plataforma de gestão da empresa, provando a sua utilidade no mundo industrial.

4

Caso de Estudo - Previsão do Tempo de Produção

O atual mercado da indústria de produção, é extremamente competitivo, sendo que, as empresas de manufatura são constantemente desafiadas pela crescente procura de produtos individualizados, que apresentem cada vez maior qualidade, e simultaneamente, menor custo de produção. Estas características aliadas a métodos de logística modernos (Okoshi et al., 2019), que permitam diminuir os tempos de produção, e consequentemente, o prazo de entrega da encomenda (Lingitz et al., 2018), de modo a diferenciar as empresas dos seus principais concorrentes (Reuter and Brambring, 2016).

O âmbito deste caso de estudo, consiste na implementação da metodologia CRISP-DM e *Automated Machine Learning (AutoML)* para prever o tempo de produção de encomendas relativamente aos processos da indústria de manufatura. Com base nas expectativas no decorrer do desenvolvimento do *PSS4* do *Produtech4S&C: Produtech Sustentável e Circular*, foram utilizados dados reais, extraídos da empresa envolvida no projeto, uma empresa portuguesa responsável pela produção de embalagens metálicas, mais concretamente, latas e aerossóis, no qual o objetivo é prever o tempo de produção de uma encomenda composta por um destes artigos, de modo a otimizar o plano de produção da empresa.

A abordagem a este problema, consistiu em diversas iterações da metodologia CRISP-DM, para obter a previsão do tempo necessário, para concluir uma ordem de produção. Para tal, foram usadas quatro ferramentas *open-source* de AutoML para a fase de modelação, automatizando a seleção de algoritmos de *Machine Learning* e da afinação dos hiperparâmetros associados. As ferramentas utilizadas foram: *AutoGluon, H20 AutoML,rminer* e *TPOT*. Outra das principais vantagens da utilização do AutoML é a possibilidade de alterar o foco do desenvolvimento para a fase de *Data Preparation* (Preparação dos Dados) do CRISP-DM.

Para cumprir com os requisitos deste caso de estudo, foram executas um total de três iterações do CRISP-DM, nas quais testamos diferentes técnicas de pré-processamento de dados e de *feature engineering*.

4.1 Iteração 1

Nesta primeira iteração, foram realizadas a cinco primeiras fases da metodologia CRISP-DM, com o objetivo de prever o tempo de produção de uma determinada encomenda.

4.1.1 Compreensão do negócio

Na indústria de produção, de modo geral, o processo de manufatura consiste na transformação das matérias-primas em diversos componentes, ou em certos casos, num produto final, por meio de um conjunto de transformações sobre a matéria-prima, que ocorrer numa sequência de operações.

Como um plano de produção delineado pela empresa, pode conter várias ordens de produção com diversas operações, a previsão exata do tempo de produção de uma encomenda, consiste num elemento fundamental na elaboração do plano, tendo um papel determinante na eficiência do mesmo, de modo a cumprir com os prazos acordados com o cliente.

Sendo uma empresa presente na indústria da manufatura, a primeira etapa do seu processo produtivo consiste na litografia, a qual inicia com o corte de uma bobina de metal. Devido à diversidade de produtos que são produzidos, as especificações do corte são inúmeras, pois a dimensão do produto a produzir, pode variar consoante os requisitos de cada cliente.

A etapa seguinte do processo consiste no envernizamento. Neste processo é aplicado um verniz ou esmalte, à folha de metal. De seguida, o produto resultante, é transferido para a fase de impressão, onde é aplicada uma tinta, numa determinada área da folha, traduzindo a ilustração definida pelo cliente para a folha de metal. Por fim, a chapa passa por um processo de corte secundário, transformando-a em diversas tiras metálicas.

Cada uma destas etapas do processo de litografia, tem um conjunto de operações associadas, que influenciam o tempo de produção e consequentemente tornam o processo de elaboração de um plano produtivo desafiante. Deste modo, foi identificada uma tarefa de regressão, a qual pretende prever o tempo de produção do processo de litografia, necessário para concluir uma ordem de produção, de forma a possibilitar a elaboração de um plano de produção eficaz.

4.1.2 Compreensão dos dados

Para esta fase, foi analisada a primeira fonte de dados fornecida pela empresa: *Production Orders*. Este conjunto de dados é composto por 14614 registos, os quais foram recolhidos

ao longo de 24 meses, desde dezembro de 2018 até dezembro de 2020. Cada um destes registos, está relacionado com uma ordem de produção finalizada.

Este *dataset* é composto por cinco colunas diferentes: *OrderID, Material, Order_quantity, Start_order* e *End_order*. Na tabela 9 é possível ver o resumo dos atributos do conjunto de dados.

Tabela 9: Descrição dos atributos do *dataset Production Orders*

Atributo	Descrição	Tipo	Distintos	Exemplo
Order_ID	ID da ordem de produção	Categórico	14438	1000234997
Material	Material do produto final	Categórico	4490	51-49012
Order_quantity	Quantidade a produzir	Numérico	3616	423
Start_order	Data de início da ordem de produção	Data	394	28/03/2019
End_order	Data de fim da ordem de produção	Data	341	06/01/2020

4.1.3 Preparação dos dados

A primeira etapa, deste processo de preparação dos dados, consistiu na criação da variável *target*: o número de dias necessário para finalizar uma ordem de produção. A esta variável, foi atribuída o nome de Delta. Esta nova coluna, é o resultado da subtração das colunas *End_order* e *Start_order*, em dias, já que a empresa também produz aos fins de semana, de modo a não parar o processo de litografia.

Para o pré-processamento de dados, foram excluídos os registos com valores nulos, como por exemplo, o material ou datas em falta. Em casos onde a *Order_ID* era duplicada (175 registos), esses registos foram excluídos, mantendo apenas o primeiro registo com um ID distinto. De seguida, foi removida a esta mesma coluna, *Order_ID*, pois não tem qualquer valor preditivo. Além disso, foram removidos os valores considerados *outliers* da coluna *target (Delta)*, quando esse valor era superior a 150 dias, com base nas sugestões da empresa, pois esses valores não fariam sentido. Em certos casos, registamos que existiam valores do Delta que eram negativos, e esses registos também foram consequentemente removidos.

Por fim, foi aplicada uma técnica de standardização à coluna *Order_quantity*, que transformou os valores numéricos numa nova escala com média igual a 0 e um desvio padrão igual a 1. Para a coluna Material, foi utilizada um técnica denominada por *Inverse Document Frequency (IDF)*, que converte uma coluna categórica numa coluna numérica de valores positivos, com base na frequência de cada nível do atributo (Matos et al., 2022). Os dados pré-processados resultaram num valor total de registos de 14101 ordens de produção.

4.1.4 Modelação

Para a fase de modelação, foram utilizadas quatro ferramentas de AutoML *open-source* descritas acima: *AutoGluon, H20 AutoML, rminer* e *TPOT*.

O método de validação utilizado, foi baseado num *benchmark* elaborado recentemente (Ferreira et al., 2021), utilizando uma validação cruzada externa de 10 *folds*. Para cada treino do modelo de AutoML, foi utilizada uma validação cruzada interna de 5 *folds*, de modo a selecionar os melhore hiperparâmetros para o modelo. Esta tarefa é feita automaticamente pelas ferramentas de AutoML.

As previsões dos conjuntos de teste para cada um dos 10 *folds*, foram avaliadas através do MAE (*Mean Absolute Error*) ($\in [0.0,\infty[$, onde quanto menor o valor, melhores são as previsões). Foi calculado o NMAE (*Normalizaed Mean Absolute Error*), que é resultado da divisão do MAE, pelo intervalo de valores da coluna *target* (Delta), em percentagem.

Para a validação interna, a qual é realizada automaticamente pelas ferramentas de AutoML, foi utilizado o MAE. De modo a manter a comparação justa entre as diversas ferramentas de AutoML, foi definido um máximo de uma hora de treino para os modelos, exceto no *rminer*, pois essa função não está disponível.

4.1.5 Avaliação

Para avaliar os resultados preditivos, foram calculadas as médias das métricas dos 10 *folds* externos. Também foi adicionado um intervalo de confiança com base na distribuição de t, com 95% de confiança, de modo a garantir a significância estatística de cada uma das experiências. A tabela 10, mostra os resultados médios para a primeira iteração do CRISP-DM.

Para cada uma das ferramentas selecionadas, a tabela mostra o algoritmo que obteve melhores resultados durante o treino dos 10 *folds* (Melhor Algoritmo). Os resultados obtidos também mostram os valores médios dos 10 *folds* externos e os seus intervalos de confiança (MAE e NMAE)

Tabela 10: Média dos resultados obtidos na primeira iteração do CRISP-DM

Ferramenta de AutoML	Melhor Algoritmo	MAE	NMAE
AutoGluon	Ensemble	4.00 ± 0.15	3.46 ± 0.26
H2O AutoML	Stacked Ensemble	3.70 ± 0.10	3.21 ± 0.24
rminer	Stacked Ensemble	3.98 ± 0.12	3.45 ± 0.27
TPOT	Gradient Boosting	3.77 ± 0.12	3.26 ± 0.25

Através dos resultados obtidos, os quais podemos observar na tabela acima, o melhor resultado foi obtido pelo *H20 AutoML*, com uma média de erro de 3,70 dias, o que corresponde

a um NMAE de 3,21%. No entanto, as restantes ferramentas também obtiveram resultados muito aproximados (MAE médio entre 3,70 e 4,00 e um NMAE médio entre o intervalo de 3,21% e 3,46%).

4.2 Iteração 2

No decorrer do desenvolvimento deste caso de estudo, especificamente ao longo desta primeira iteração do CRISP-DM, a empresa disponibilizou mais um *dataset*, relacionado com as operações individuais, pelas quais cada uma das ordens de produção passa, ao longo do processo produtivo. Esta ação, resultou na necessidade de realizar uma nova iteração da metodologia CRISP-DM, com o objetivo de obter melhores resultados.

4.2.1 Compreensão do negócio

Durante esta nova fase da compreensão do negócio, a empresa forneceu novos *insights* sobre o processo de litografia, relevantes no âmbito deste caso de estudo. O processo produtivo composto por diversas operações como por exemplo o corte ou a pintura das folhas metálicas.

Cada uma destas operações, são realizadas num centro específico de operações, com a possibilidade de envolver diferentes tipos de matérias-primas. Cada uma das operações de produção está associada a um *Order_ID*, sendo também composto por um registo temporal de início e fim da operação. O objetivo desta iteração do CRISP-DM, consiste em melhorar os resultados da iteração anterior, através da previsão do tempo necessário a realizar cada uma das operações de produção, em vez da previsão do tempo de produção das encomendas.

Ao tentar prever este atributo, podemos estimar o tempo total necessário para concluir a ordem de produção associada às operações definidas.

4.2.2 Compreensão dos dados

Para esta fase do CRISP-DM, foi analisado um novo *dataset* (*Operations*), diretamente relacionado com as operações de produção associadas a uma ordem de produção específica. Este conjunto de dados é composto por 40610 registos e 8 colunas. Cada uma das ordens de produção tem entre uma e nove operações. O número mais comum de operações pela qual uma ordem de produção passa, é de três. Na tabela 11 podemos ver a descrição dos atributos do novo *dataset*.

Tabela 11: Descrição dos atributos do *dataset* Operações.

Atributo	Descrição	Tipo	Distintos	Exemplo
Order_ID	ID de ordem de produção	Categórico	14438	1000234997
Operation_ID	ID do tipo de produção	Categórico	36	25
Work_Center	Centro de trabalho onde a operação ocorre	Categórico	21	1LE02
Operation_quantity	Quantidade associada à operação	Numérico	4369	44587
Start_date_op	Data de início de operação	Data	385	16/09/2020
End_date_op	Data de fim da operação	Data	390	04/11/2020
Start_time_op	Hora de inicio da operação	Hora	29362	10:54:27
End_time_op	Hora de fim da operação	Hora	29369	14:26:02

4.2.3 Preparação dos dados

A criação do novo *target*, seguiu a mesma abordagem que na iteração anterior. Foi criada uma coluna *target*, que consiste no número de dias que uma operação demora para ser concluída. Neste caso em específico, como o *dataset Operations* é composto por dados relativos ao horário de início e de fim de uma operação, calculamos a diferença entre estes dois valores, de modo a obter a duração da operação. O resultado obtido, consiste nos valores da coluna *target*, os quais, foram convertidos para um valor decimal, por exemplo 1,50 dias, correspondendo a um dia e meio que leva uma operação a ser concluída.

Relativamente à preparação dos dados, foram usadas transformações semelhantes à iteração anterior. Primeiro foram removidos os registos com valores vazios, e de seguida, foi utilizado o mesmo *threshold* para a remoção de *outliers* da variável *target* (operações que registassem mais de 150 dias para serem concluídas, foram removidas). Este processo apenas resultou na remoção de 12 registos.

Sendo que, várias operações de produção estão associadas ao mesmo pedido de produção, não houve a necessidade de remover os registos com o *Order_ID* duplicado. Por outro lado, houve a necessidade de remover os registos com Order_ID que não estavam presentes no *dataset Production Orders*, utilizado na iteração anterior.

Por fim, a coluna *Operation_quantity*, passou por processo de standardização, e às colunas categóricas (*Order_ID, Operation_ID e Work_Center*). Nesta fase o número de registos do *dataset* passou de 40610 registo para 39225.

4.2.4 Modelação

A fase de modelação desta iteração do CRISP-DM, foi semelhante à primeira iteração, a qual se encontra detalhada na secção 4.1.4. Foram usadas as mesmas ferramentas de AutoML (*AutoGluon, H2O AutoML,rminer* e *TPOT*), a mesma validação (*cross-validation* externa de 10 *folds* e *cross-validation* interna de 5 *folds*), bem como as mesmas métricas preditivas (MAE e NMAE).

4.2.5 Avaliação

Nesta iteração do CRISP-DM, o objetivo consistiu, na previsão do tempo de operação, em vez do tempo total de produção de uma ordem de produção. No fim da obtenção dos resultados das previsão, o conjunto de dados *Operations* foi agregado. Agrupando o *dataset* por *Order_ID* e somando o tempo que cada operação demorou para ser concluída, foi possível, obter o tempo total de produção previsto.

De seguida, o tempo de produção, foi comparado com os valores reais do *dataset Production Order*. Desta forma, em vez de utilizar as métricas MAE e NAME, para as operações individuais, os resultados foram agregados para manter uma comparação consistente entre as iterações do CRISP-DM, assim como, o objetivo deste caso de estudo, vai de encontro com os objetivos do *PPS4 do projeto Produtech4S&C*, e da empresa associada, que é prever o tempo total de produção de uma ordem de produção. A tabela 12 mostra os resultados obtidos para a segunda iteração do CRISP-DM.

Tabela 12: Média dos resultados obtidos na segunda iteração do CRISP-DM

Ferramenta de AutoML	Melhores Algoritmos	MAE	NMAE
AutoGluon	Ensemble	6.00$\pm$0.10	4.46$\pm$0.35
H2O AutoML	Deep Learning	5.83$\pm$0.68	4.35$\pm$0.76
rminer	Suppor Vector Machine (SVM)	6.51$\pm$0.11	4.82$\pm$0.38
TPOT	Gradient Boosting	6.64$\pm$0.11	4.92$\pm$0.40

Os resultados, nesta segunda iteração revelam que, de forma geral, esta abordagem obteve piores resultados, quando comparados com os resultados obtidos na primeira iteração. Mesmo comparando a melhor ferramenta AutoML, para esta iteração (H20 AutoML), que obteve MAE de 5,83 dias, este resultado é pior que todos os resultados obtidos por todas as ferramentas da primeira iteração, na qual foi obtido um MAE entre 3,70 a 4,00 dias.

Estes resultados, sugerem que neste caso, prever o tempo de produção por meio da previsão do tempo de operação, é menos eficaz que usar os modelos de *Machine Learning* para prever o tempo total de produção.

Por fim, depois de analisar os resultados desta iteração com os especialistas da empresa, foi possível concluir, que a degradação dos resultados foi causada por dois fatores: primeiramente, algumas operações podem ser executas simultaneamente em duas ou mais linhas de produção, e a segunda razão, deve-se ao facto de que diferentes operações também têm tempos diferentes de configuração da máquina, onde a operação irá ser realizada. Estes registos relativos ao tempo de configuração da máquina, não são armazenados pela empresa.

4.3 Iteração 3

Para verificar se é possível melhorar os resultados obtidos, foi decidido realizar uma nova etapa do CRISP-DM, utilizando em simultâneo ambos os conjuntos de dados, *Production Orders* e *Operations*. Os *dataset*, foram intercalados ficando com apenas um registo por cada *Order_ID*, mas com colunas adicionais, relativas às operações executadas por cada uma das ordens de produção. Como os conjuntos de dados a serem usados são os mesmos que nas iterações anteriores, a fase de compreensão do negócio e de compreensão dos dados foram ignoradas para esta iteração.

4.3.1 Preparação dos dados

De modo a intercalar os dois *datasets* (*Production Orders* e *Operations*), foram removidos os registos com valores nulos em ambos os conjuntos de dados. De seguida, foi feito um *pivot* sobre o *dataset* relativo às operações, de forma, que cada operação diferente se transformasse numa nova coluna. Como este conjunto de dados era composto por 36 operações diferentes, o resultado foi a criação de 36 novas colunas para o *dataset Operations*. Depois de realizada esta transformação, os valores foram alterados para 1 e 0 (1, se a ordem de produção incluir a operação; 0, caso contrário).

Adiante, foi realizado um *left join* entre o *dataset Production Order* e *Operations*. Esta transformação resultou na adição de 36 novas colunas no *dataset Operations* (as novas colunas criadas através do *pivot*). Por fim, a coluna *target* foi criada através da subtração da *End_Order* com a *Start_Order*. A figura 13 representa o processo de obtenção do *dataset* final.

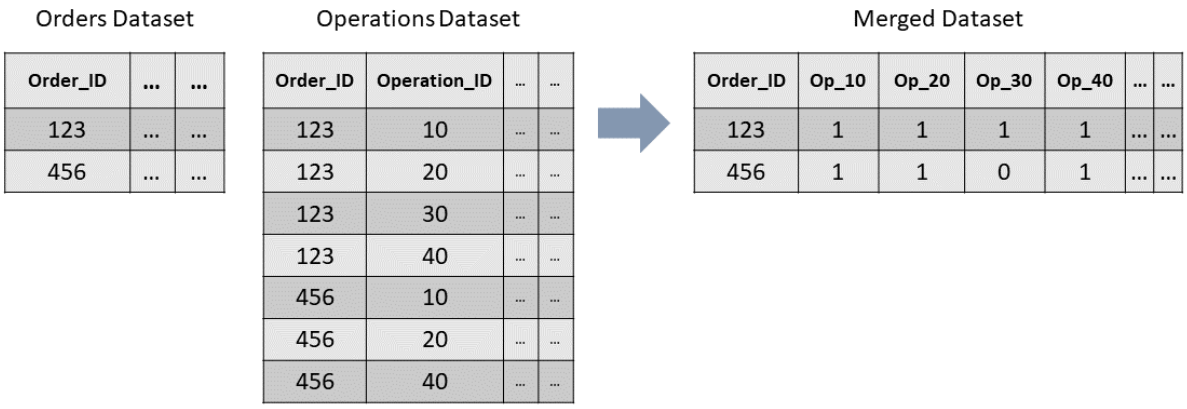

Figura 13: Representação do processo de transformação do dataset

Semelhante à primeira iteração, a coluna *Order_quantity* passou por um processo de standardização e para a coluna Material, foi aplicada a técnica de IDF. O *dataset* resultante é composto por 14106 linhas e 39 colunas, excluindo a coluna *target*.

4.3.2 Modelação

O processo de modelação desta iteração, foi similar às outras duas iterações do CRISP-DM, na qual foram usadas as mesmas ferramentas de AutoML (*AutoGluon, H2O AutoML, rminer* e *TPOT*).

4.3.3 Avaliação

A tabela 13 mostra os resultados para esta terceira iteração do CRISP-DM, incluindo, para cada ferramenta de AutoML, o algoritmo que obteve melhores resultados, através das métricas MAE e NMAE e os respetivos intervalos de confiança (distribuição de t com 95% de confiança).

Tabela 13: Média dos resultados obtidos na terceira iteração do CRISP-DM

Ferramenta de AutoML	Melhor Algoritmo	MAE	NMAE
AutoGluon	Ensemble	3.38$\pm$0.07	2.81$\pm$0.31
H2O AutoML	Stacked Ensemble	3.03$\pm$0.04	2.53$\pm$0.29
rminer	Stacked Ensemble	3.42$\pm$0.08	2.85$\pm$0.31
TPOT	Gradient Boosting	3.17$\pm$0.08	2.64$\pm$0.29

Os resultados finais, mostram que, na terceira iteração do CRISP-DM, o *H20 AutoML*, obteve os melhores resultados, com um MAE de 3,03 dias, correspondendo a um NMAE de 2,53%. Semelhante à primeira iteração, todas as ferramentas obtiveram resultados aproximados, com uma diferença máxima de 0,39 relativamente ao MAE, e 0,32 pontos percentuais para o NMAE. Esta iteração, foi a que obteve melhores resultados a nível da previsão do tempo de produção.

4.4 Considerações finais

A tabela 14, apresenta um resumo dos resultados preditivos obtidos, ao longo das três etapas do CRISP-DM. Para cada uma das iterações, podemos observar os *datasets* utilizados, assim como, as ferramentas de AutoML aplicadas e os respetivos algoritmos que obtiveram os melhores resultados ao longo do treino com 10 *folds*. Os resultados são apresentados através da média das métricas (MAE e NMAE), utilizadas para avaliar os resultados de teste.

Após realizadas três diversas iterações do CRISP-DM, podemos observar, que os piores resultados foram obtidos durante a segunda iteração, como podemos observar na tabela Y, quando o objetivo passou por prever o tempo de produção final, utilizando apenas o *dataset Operations*. Enquanto na primeira iteração a melhor ferramenta de AutoML, o H20

AutoML, obteve um MAE médio de 3,70 dias, o melhor resultado na segunda iteração obteve quase o dobro, mais concretamente, um MAE médio de 5.83 dias.

Como explicado anteriormente, a degradação dos resultados obtidos durante a segunda iteração, deve-se ao facto, da falta de informação, relativa ao tempo que demora a preparar uma máquina para outra operação, assim como, com a possibilidade de ocorrerem operações em paralelo.

Tabela 14: Média dos resultados obtidos nas várias iterações do CRISP-DM.

Iteration	Datasets	AutoML Tool	Best Algorithm	MAE	NMAE
1	Orders	AutoGluon	Ensemble	4.00±0.15	3.46±0.26
		H2O AutoML	Stacked Ensemble	3.70±0.10	3.21±0.24
		rminer	Stacked Ensemble	3.98±0.12	3.45±0.27
		TPOT	Gradient Boosting	3.77±0.12	3.26±0.25
2	Operations	AutoGluon	Ensemble	6.00±0.10	4.46±0.35
		H2O AutoML	Deep Learning	5.83±0.68	4.35±0.76
		rminer	SVM	6.51±0.11	4.82±0.38
		TPOT	Gradient Boosting	6.64±0.11	4.92±0.40
3	Orders and Operations	AutoGluon	Ensemble	3.38±0.07	2.81±0.31
		H2O AutoML	Stacked Ensemble	3.03±0.04	2.53±0.29
		rminer	Stacked Ensemble	3.42±0.08	2.85±0.31
		TPOT	Gradient Boosting	3.17±0.08	2.64±0.29

A terceira iteração, onde os *datasets Orders* e *Operantios* foram intercalados, foi onde se obteve os melhores resultados preditivos, com um MAE médio de 3.03 dias e um NMAE de 2.53%, através do *H20 AutoML*. No entanto, nesta iteração, é possível observar que todas as ferramentas alcançaram resultados médios melhores do que, nas outras duas iterações. Isto deve-se ao facto, das transformações realizadas na etapa Preparação dos Dados da metodologia CRISP-DM, através da utilização dos dados referentes às operações individuais.

Por fim, em relação às ferramentas de AutoML utilizadas, podemos observar que o H20 AutoML, obteve os melhores resultados ao longo das três iterações.

5

Conclusão

Este último capítulo, consiste em apresentar uma visão global do trabalho realizado, sintetizando os principais aspetos desenvolvidos ao longo desta dissertação. Dividido em três subcapítulos, na primeira parte, Síntese do Trabalho Realizado, é descrito brevemente o trabalho realizado ao longo deste documento. Seguidamente os contributos alcançados e por último, são identificados os próximos passos no âmbito deste projeto.

5.1 Síntese do Trabalho Realizado

Esta dissertação, foi realizada no âmbito do projeto *PRODUTECH4S&C: PRODUTECH SUSTENTÁVEL & CIRCULAR*, especificamente no *PPS4 - Cadeia digital de fornecimento em contexto circular*, e dessa forma, os objetivos eram comuns: desenvolvimento de previsões por métodos preditivos de *Machine Learning*, relativamente à previsão do número de encomendas e à previsão do tempo de produção. Com os objetivos traçados, e o planeamento das atividades delineado, este estudo inicia-se, com a explicação da metodologia a utilizar, para o desenvolvimento dos casos de estudo, mais concretamente o CRISP-DM.

Segue-se a revisão do estado da arte, onde os conceitos fundamentais para a realização desta dissertação foram descritos. Importante destacar as áreas de conhecimento estudadas, relativamente aos conceitos de *Machine Learning*, como *Automated Machine Learning* e *Time Series*, pois estes foram os conceitos explorados durante o desenvolvimentos dos casos de uso. Finalmente, as ferramentas que possibilitaram o desenvolvimento dos casos de estudo propostos são apresentados.

Em síntese, os casos de estudo Previsão de Encomendas e Previsão do Tempo de Produção são detalhadamente analisados, e o trabalho realizado, na sequência da metodologia CRISP-DM é exposto, descrevendo as atividades desenvolvidas em cada uma das fases, desde a Compreensão do Negócio, até à quinta fase da metodologia, a Avaliação.

5.2 Contributos

O trabalho realizado neste ensaio, encontra-se concluído, e deste modo, é possível destacar o conhecimento alcançado no âmbito de *Machine Learning*. Tratando-se de um projeto mobilizador, é importante destacar a utilização de dados do mundo real para o desenvolvimento das experiências.

Foi possível, desenvolver uma solução para cada um dos casos de uso, através de métodos de *Machine Learning*. A solução encontrada para cada um dos problemas, foi possível obter, através da aplicação da metodologia CRISP-DM. À medida que novos *insights* relativos ao negócio surgiam, novos conjuntos de dados eram fornecidos pela empresa, foi possível, a realização de diversas iterações da metodologia. Os resultados obtidos, permitiram a comparação das experiências, de modo a concluir, qual a melhor abordagem.

Para além do estudo teórico, relativo às tecnologias da área da Inteligência Artificial, nesta tese, podemos encontrar uma solução para os problemas industriais inicialmente identificados, através da comparação, de diversos modelos preditivos de *Machine Learning* e de ferramentas de AutoML.

Para terminar, é possível destacar a publicação de um artigo de investigação na conferência *18th International Conference on Artificial Intelligence Applications and Innovations*.

5.3 Trabalho Futuro

O desenvolvimento desta tese, especificou-se em dois aspetos, de uma empresa do meio industrial. Isto não significa que o trabalho neste âmbito este, já concluído. É fundamental para o desenvolvimento de modelos preditivos, um volume de dados considerável, o qual não foi possível obter no decorrer do projeto. A quantidade e a qualidade dos dados de *input*, para um modelo, são determinantes para o sucesso do mesmo, portanto, com o decorrer do tempo, novos dados serão armazenados, e os modelos serão novamente treinados, possibilitando o aumento da robustez e eficácia dos mesmos.

Com o intuito de validar os resultados obtidos, um dos próximos passos, consistiria em colocar os modelos desenvolvidos num estado de produção na empresa e comparar os seus resultado preditivos com os valores reais. Assim, seria possível comprovar as vantagens que estes modelos preditivos oferecem às organizações.

Tratando-se de uma dissertação na área de *Machine Learning*, onde novas tecnologias surgem com elevada frequência, os melhores métodos de desenvolvimento que temos acesso hoje, podem não ser os mais indicado no dia de amanhã. À medida que novas tecnologias são geradas, um novo espetro de alternativas é criado, e novas possibilidades terão de ser tomadas em conta.

Referências Bibliográficas

Agarap, A. F. M. (2018). A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data. In Proceedings of the 2018 10th International Conference on Machine Learning and Computing. ACM.

Alzubi, J., Nayyar, A., and Kumar, A. (2018). Machine learning from theory to algorithms: An overview. Journal of Physics: Conference Series, 1142:012012.

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al. (2020). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information fusion, 58:82–115.

Bontempi, G., Ben Taieb, S., and Borgne, Y.-A. L. (2012). Machine learning strategies for time series forecasting. In European business intelligence summer school, pages 62–77. Springer.

Briot, J.-P., Hadjeres, G., and Pachet, F. (2017). Deep learning techniques for music generation - a survey.

Buchanan, B. G. (2005). A (very) brief history of artificial intelligence. Ai Magazine, 26(4):53–53.

Chang, A. C. (2020). Intelligence-based medicine: artificial intelligence and human cognition in clinical medicine and healthcare. Academic Press.

Chauhan, V. K., Dahiya, K., and Sharma, A. (2019). Problem formulations and solvers in linear svm: a review. Artificial Intelligence Review, 52(2):803–855.

Cortez, P., Pereira, P. J., and Mendes, R. (2020). Multi-step time series prediction intervals using neuroevolution. Neural Comput. Appl., 32(13):8939–8953.

Ferreira, L., Pilastri, A., Martins, C. M., Pires, P. M., and Cortez, P. (2021). A comparison of automl tools for machine learning, deep learning and xgboost. In 2021 International Joint Conference on Neural Networks (IJCNN), pages 1–8.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. (2015). Efficient and robust automated machine learning. Advances in neural information processing systems, 28.

Gelper, S., Fried, R., and Croux, C. (2010). Robust forecasting with exponential and holt–winters smoothing. Journal of forecasting, 29(3):285–300.

Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction, volume 2. Springer.

He, X., Zhao, K., and Chu, X. (2021). Automl: A survey of the state-of-the-art. Knowledge-Based Systems, 212:106622.

Iorkyase, E., Tachtatzis, C., Glover, I., Lazaridis, P., Upton, D., Saeed, B., and Atkinson, R. (2019). Improving rf-based partial discharge localization via machine learning ensemble method. IEEE Transactions on Power Delivery.

Jain, A. K. (2010). Data clustering: 50 years beyond k-means. Pattern recognition letters, 31(8):651–666.

Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., and Hoffmann, M. (2014). Industry 4.0. Business & information systems engineering, 6(4):239–242.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. nature, 521(7553):436–444.

Lingitz, L., Gallina, V., Ansari, F., Gyulai, D., Pfeiffer, A., Sihn, W., and Monostori, L. (2018). Lead time prediction using machine learning algorithms: A case study by a semiconductor manufacturer. Procedia CIRP, 72:1051–1056.

Loh, W.-Y. (2011). Classification and regression trees. Wiley interdisciplinary reviews: data mining and knowledge discovery, 1(1):14–23.

Lu, Y. (2017). Industry 4.0: A survey on technologies, applications and open research issues. Journal of industrial information integration, 6:1–10.

Makkar, S., Devi, G., and Solanki, V. K. (2019). Applications of machine learning techniques in supply chain optimization. In International Conference on Intelligent Computing and Communication Technologies, pages 861–869. Springer.

Matloff, N. (2017). Statistical regression and classification: from linear models to machine learning. Chapman and Hall/CRC.

Matos, L. M., Azevedo, J., Matta, A., Pilastri, A., Cortez, P., and Mendes, R. (2022). Categorical attribute transformation environment (cane): A python module for categorical to numeric data preprocessing. Software Impacts, 13:100359.

Mechelli, A. and Viera, S. (2019). Machine learning: methods and applications to brain disorders. Academic Press.

Mitchell, T. M. and Mitchell, T. M. (1997). Machine learning, volume 1. McGraw-hill New York.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2018). Foundations of machine learning. MIT press.

Murray, G. and Scime, A. (2010). Microtargeting and electorate segmentation: Data mining the american national election studies. Journal of Political Marketing, 9:143–166.

Myles, A. J., Feudale, R. N., Liu, Y., Woody, N. A., and Brown, S. D. (2004). An introduction to decision tree modeling. Journal of Chemometrics: A Journal of the Chemometrics Society, 18(6):275–285.

Nasir, Y., He, J., Hu, C., Tanaka, S., Wang, K., and Wen, X. (2021). Deep reinforcement learning for constrained field development optimization in subsurface two-phase flow. Frontiers in Applied Mathematics and Statistics, 7:689934.

Okoshi, C. Y., Pinheiro de Lima, E., and Gouvea Da Costa, S. E. (2019). Performance cause and effect studies: Analyzing high performance manufacturing companies. International Journal of Production Economics, 210:27–41.

Oliveira, N., Cortez, P., and Areal, N. (2017). The impact of microblogging data for stock market prediction : Using Twitter to predict returns , volatility , trading volume and survey sentiment indices. Expert Systems With Applications, 73:125–144.

Olson, R. S., Bartley, N., Urbanowicz, R. J., and Moore, J. H. (2016). Evaluation of a tree-based pipeline optimization tool for automating data science.

Olson, R. S. and Moore, J. H. (2016). Tpot: A tree-based pipeline optimization tool for automating machine learning. In Workshop on automatic machine learning, pages 66–74. PMLR.

Parmezan, A. R. S., Souza, V. M., and Batista, G. E. (2019). Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. Information sciences, 484:302–337.

Reuter, C. and Brambring, F. (2016). Improving data consistency in production control. Procedia CIRP, 41:51–56.

Samuel, A. L. (1967). Some studies in machine learning using the game of checkers. iirecent progress. IBM Journal of research and development, 11(6):601–617.

Shafiq, S. I., Sanin, C., Toro, C., and Szczerbicki, E. (2015). Virtual engineering object (veo): Toward experience-based design and manufacturing for industry 4.0. Cybernetics and Systems, 46(1-2):35–50.

Siami-Namini, S., Tavakoli, N., and Namin, A. S. (2018). A comparison of arima and lstm in forecasting time series. In 2018 17th IEEE international conference on machine learning and applications (ICMLA), pages 1394–1401. IEEE.

Su, X., Yan, X., and Tsai, C.-L. (2012). Linear regression. Wiley Interdisciplinary Reviews: Computational Statistics, 4(3):275–294.

Sutton, R. S. and Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

Tjahjono, B., Esplugues, C., Ares, E., and Pelaez, G. (2017). What does industry 4.0 mean to supply chain? Procedia manufacturing, 13:1175–1182.

Vaidya, S., Ambad, P., and Bhosle, S. (2018). Industry 4.0–a glimpse. Procedia manufacturing, 20:233–238.

Webb, G. I., Keogh, E., and Miikkulainen, R. (2010). Naïve bayes. Encyclopedia of machine learning, 15:713–714.

Wirth, R. (2000). Crisp-dm: Towards a standard process model for data mining. In Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining, pages 29–39.

Zaytar, M. A. and El Amrani, C. (2016). Sequence to sequence weather forecasting with long short-term memory recurrent neural networks. International Journal of Computer Applications, 143:7–11.

A

Anexos

a.1 Código Python

```
df = df.drop(columns=['Sold-To Party', 'Name 1', 'Sales Document', '
    Purchase order no.', 'Material', 'Description', 'Delivery Date', '
    Created By'])

df['Order Quantity'] = df['Order Quantity'].str.replace(",", ".")
df['Order Quantity'] = pd.to_numeric(df['Order Quantity'])

df2020['Dias'] = pd.to_datetime(df2020['Dias'], dayfirst=True)
df2020 = df2020.sort_values(by='Dias').reset_index(drop=True)

df['Document Date'] = pd.to_datetime(df['Document Date'], dayfirst=True)

#Groupby Dia
df_gb = df.groupby(df['Document Date'].dt.date).sum().reset_index(drop=
    False)

df = pd.DataFrame(df_gb)
df2020 = pd.DataFrame(df2020)

#Formataçao
df2020.rename(columns={"Dias" : "Document Date"}, inplace=True)
df2020["Document Date"] = df2020["Document Date"].astype("str")

df_test = pd.merge(df, df2020, on="Document Date", how="outer")
df_test["Document Date"]  = pd.to_datetime(df_test["Document Date"])
df_test.sort_values(by='Document Date').reset_index(drop=True)

df_test = df_test.groupby(['Document Date']).apply(lambda x: x[~x['Document
     Date'].duplicated() | ~x['Order Quantity'].isna()])
df = df_test.reset_index(drop=True)
df['Order Quantity'] = df['Order Quantity'].fillna(0)
```

Listing A.1: Processamento Dataset Order Quantity

```
def hold(y, ratio=2 / 3, mode="rolling", it=1, window=10, increment=1):
    ALLITR = None
    VAL = None
    NSIZE = len(y)
    if mode == "incremental" or mode == "rolling":
        aux = window + increment * (it - 1)
        aux = min(aux, NSIZE)
        if mode == "rolling":
            iaux = max((aux - window + 1), 1)
        else:
            iaux = 1
        ALLTR = range(iaux, aux)
        end = aux + ratio
        end = min(end, NSIZE)
        iend = aux + 1
        if iend < end:
            TS = range(iend, end)
        else:
            TS = None
    return {"tr": ALLTR, "itr": ALLITR, "val": VAL, "ts": TS}

def calcIterationsRW(datasetSize, trainingSize, testSize, stepSize):

    trainingSize = 192
    testSize = 8
    ratio = testSize
    stepSize = 8
    #totalLenght = df.shape[0]
    mode = "rolling"

    return int((datasetSize - (trainingSize + testSize))/ stepSize)

iterations = calcIterationsRW(df.shape[0], trainingSize, testSize, stepSize
    )
```

Listing A.2: Rolling Window

```
def runneuralprophet(data, iterations):
    max_value = data['y'].max()
    min_value = data['y'].min()
    denominator = max_value - min_value
```

```
dfFinal = pd.DataFrame(columns = ['MAE', 'NMAE', 'MSE'])

MAEVals = []
NMAEVals = []
MSEVals = []

for i in trange(iterations):

        holds = hold(data[['y']], ratio, mode, i+1, trainingSize,
            stepSize)
        print('holds')
        print("Iteration ", i+1 , "of ", iterations, " TR: ", holds['tr
            '] , "TS: ", holds['ts'])
        tr = data.iloc[holds['tr']].reset_index(drop=True)
        ts = data.iloc[holds['ts']].reset_index(drop=True)

        m = NeuralProphet()

        m.fit(ts, freq="D")

        print(tr)

        p = m.predict(ts)

        MAE = mean_absolute_error(ts['y'].values.tolist(), p['yhat1'])
        NMAE = (MAE/denominator)*100 #o denominador é calculado com o
            maximo e minimo global
        MSE = mean_squared_error(ts['y'], p['yhat1'])

        print("Iteration ", i , "MAE ", MAE, "NMAE of ", NMAE, "MSE of"
            , MSE)

        y_true = ts['y'].values
        y_pred = p['yhat1'].values
        pyplot.plot(y_true, label='Actual')
        pyplot.plot(y_pred, label='Predicted')
        pyplot.legend()
        pyplot.show()


        fig_forecast = m.plot(p)
        fig_components = m.plot_components(p)
        fig_model = m.plot_parameters(p)


        MAEVals.append(MAE)
        NMAEVals.append(NMAE)
```

```
            MSEVals.append(MSE)

            new = {'MAE': MAE, 'NMAE': NMAE, 'MSE': MSE}
            dfFinal = dfFinal.append(new, ignore_index=True)
            dfFinal = pd.DataFrame(dfFinal)

    return dfFinal
```

Listing A.3: Aplicação da Rolling Window com Neural-Prophet

```
# production
df = pd.read_csv('Orders.csv', sep=";")
df = df[['Ordem', 'Material', 'Qtd.confirmada', 'Iníc.real', 'DtaRealFim']]
df = df.rename(columns={"Iníc.real":"Inic_Ordem", "DtaRealFim":"Fim_Ordem",
    "Qtd.confirmada":"Qtd_Ordem"})
df = df.dropna().reset_index(drop=True)
df["Qtd_Ordem"]= df["Qtd_Ordem"].str.replace(' ','')
df['Qtd_Ordem'] = df['Qtd_Ordem'].astype(float)
df.head(3)

# operations
df1 = pd.read_csv('Operations.csv', sep=";")
df1 = df1[['Ordem', 'Operação', 'Centro de trabalho', 'Quantidade operação
   (MEINH)', 'Data início real de execução', 'Data fim real da execução', '
   Hora início real de execução', 'Hora fim real da execução']]
df1 = df1.rename(columns={"Quantidade operação (MEINH)":"Qnt_Operacao", "
   Data início real de execução":"D_Inic_Operacao",
                         "Data fim real da execução":"D_Fim_Operacao", "
                             Hora início real de execução":"H_Inic_Operacao"
                             ,
                         "Hora fim real da execução":"H_Fim_Operacao"})

df1['Qnt_Operacao'] = df1['Qnt_Operacao'].str.replace(",", ".")
df1['Qnt_Operacao'] = df1['Qnt_Operacao'].astype(float)
df1 = df1.dropna().reset_index(drop=True)
df1.head(3)

# merge operations and production
df_merge = pd.merge(df, df1, on='Ordem')
df_merge.head(3)

# combine date and time
df_merge['DT_Inic_Operacao'] = pd.to_datetime(df_merge['D_Inic_Operacao'] +
    ' ' + df_merge['H_Inic_Operacao'])
df_merge['DT_Fim_Operacao'] = pd.to_datetime(df_merge['D_Fim_Operacao'] + '
    ' + df_merge['H_Fim_Operacao'])
```

```
df_merge.drop(columns=['D_Inic_Operacao', 'H_Inic_Operacao', '
    D_Fim_Operacao', 'H_Fim_Operacao'], inplace=True)
df_merge.head(3)

# change types
df_merge['Operação'] = df_merge['Operação'].astype(str)

# label encoding
label_encoder = preprocessing.LabelEncoder()
df_merge['Operação'] = label_encoder.fit_transform(df_merge['Operação'])
df_merge['Centro de trabalho'] = label_encoder.fit_transform(df_merge['
    Centro de trabalho'])
df_merge.head(3)

# create target column
df_merge['Delta'] = df_merge['DT_Fim_Operacao'] - df_merge['
    DT_Inic_Operacao']
df_merge['Delta'] = df_merge['Delta']/np.timedelta64(1,'D')
df_merge['Delta'] = df_merge['Delta'] + 1
df_merge.head(3)


df_merge['Inic_Ordem'] = pd.to_datetime(df_merge['Inic_Ordem'], errors='
    coerce', dayfirst=True)
df_merge['Fim_Ordem'] = pd.to_datetime(df_merge['Fim_Ordem'], errors='
    coerce', dayfirst=True)
df_merge = df_merge.dropna().reset_index(drop=True)
df_merge['DeltaT'] = df_merge['Fim_Ordem'] - df_merge['Inic_Ordem']
df_merge['DeltaT'] = df_merge['DeltaT']/np.timedelta64(1,'D')
df_merge.head(3)

# pivot table
pivot = df1[['Ordem', 'Operação']].pivot(index='Ordem', columns='Operação',
     values='Operação').fillna(0).reset_index()
pivot.drop(columns=['Ordem'], inplace=True)
pivot[pivot != 0] = 1
pivot = pivot.astype(int)

# group operations by order
df1_group = df1.groupby(['Ordem']).sum()
df1_group = df1_group.drop(columns=['Operação']).reset_index(drop=False)

df2 = pd.merge(pivot, df1_group, left_index=True, right_index=True)
#df2 = df2.drop(columns=['index'])
cols = df2.columns.tolist()
cols = cols[-1:] + cols[:-1]
df2 = df2[cols]
```

```
df_final = df_merge.merge(df2, how="left", on="Ordem")
df_final.drop(columns=['Operação', 'Centro de trabalho',
    'Qnt_Operacao_x', 'Qnt_Operacao_y', 'DT_Inic_Operacao', '
       DT_Fim_Operacao', 'Delta'], inplace=True)
df_final.drop(df_final[df_final['DeltaT'] < 0].index, inplace=True)
df_final.drop(df_final[df_final['DeltaT'] > 150].index, inplace=True)
df_final.drop_duplicates(inplace=True)
df_final.reset_index(drop = True, inplace=True)
df_final.head(3)

# normalize qtd_ordem
scaler = StandardScaler()
scaler.fit(pd.DataFrame(df_final["Qtd_Ordem"]))
df_final["Qtd_Ordem"] = pd.DataFrame(scaler.transform(pd.DataFrame(df_final
   ["Qtd_Ordem"])))

# apply idf
cols = ["Ordem", "Material"]
df_final = cane.idf(df_final, columns_use = cols)

df_final.to_csv("production_orders.csv", sep=",", index=False)
```

Listing A.4: Pré-Processamento Orders e Operations

```
fold1 = pd.read_csv(data_path + "-fold1.csv")
fold2 = pd.read_csv(data_path + "-fold2.csv")
fold3 = pd.read_csv(data_path + "-fold3.csv")
fold4 = pd.read_csv(data_path + "-fold4.csv")
fold5 = pd.read_csv(data_path + "-fold5.csv")
fold6 = pd.read_csv(data_path + "-fold6.csv")
fold7 = pd.read_csv(data_path + "-fold7.csv")
fold8 = pd.read_csv(data_path + "-fold8.csv")
fold9 = pd.read_csv(data_path + "-fold9.csv")
fold10 = pd.read_csv(data_path + "-fold10.csv")

cols_to_drop = ["Fim_Ordem"]

fold1 = fold1.drop(columns=cols_to_drop)
fold2 = fold2.drop(columns=cols_to_drop)
fold3 = fold3.drop(columns=cols_to_drop)
fold4 = fold4.drop(columns=cols_to_drop)
fold5 = fold5.drop(columns=cols_to_drop)
fold6 = fold6.drop(columns=cols_to_drop)
fold7 = fold7.drop(columns=cols_to_drop)
fold8 = fold8.drop(columns=cols_to_drop)
```

```
fold9 = fold9.drop(columns=cols_to_drop)
fold10 = fold10.drop(columns=cols_to_drop)

folds = [fold1, fold2, fold3, fold4, fold5, fold6, fold7, fold8, fold9,
    fold10]

import autogluon as ag
from autogluon.tabular import TabularDataset, TabularPredictor

results = pd.DataFrame({'fold': [], 'model' : [], 'mae' : [], 'nmae' : [],
    'time': []})

for i in range(0, 10):
    fold_folder = "./fold" + str(i+1)
    folds = [fold1, fold2, fold3, fold4, fold5, fold6, fold7, fold8, fold9,
        fold10]
    test_df = folds[i]
    del folds[i]
    train_df = pd.concat(folds)
    train_data = TabularDataset(train_df)
    start = datetime.now().strftime("%H:%M:%S")
    predictor = TabularPredictor(label=label_column, eval_metric="
        mean_absolute_error", path = fold_folder).fit(train_data,
        num_bag_folds=5, time_limit = 3600)
    end = datetime.now().strftime("%H:%M:%S")
    predictor.leaderboard().to_csv(fold_folder + "/leaderboard.csv")
    leader = predictor.leaderboard()["model"][0].split("_")[0]
    test_data = TabularDataset(test_df)
    y_test = test_data[label_column]
    y_pred = predictor.predict(test_data)
    perf = predictor.evaluate_predictions(y_true=y_test, y_pred=y_pred,
        auxiliary_metrics=True)
    perf = dict(perf)
    mae = 0 - perf["mean_absolute_error"]
    nmae = 100 * (mae / abs(max(y_test) - min(y_test)))
    t1 = datetime.strptime(start, "%H:%M:%S")
    t2 = datetime.strptime(end, "%H:%M:%S")
    duration = (t2 - t1).seconds
    perf["start"] = start
    perf["end"] = end
    perf = json.dumps(perf)
    f = open(fold_folder + "/perf.json","w")
    f.write(perf)
    f.close()
    results = results.append({'fold': (i+1),'model' : leader, 'mae' : mae,
        'nmae' : nmae, 'time': duration}, ignore_index=True)
```

```
    results.to_csv("./metrics.csv", index=False)
```

Listing A.5: Aplicação Auto-Gluon

```
target = "DeltaT"

fold1 = pd.read_csv(data_path + "-fold1.csv")
fold2 = pd.read_csv(data_path + "-fold2.csv")
fold3 = pd.read_csv(data_path + "-fold3.csv")
fold4 = pd.read_csv(data_path + "-fold4.csv")
fold5 = pd.read_csv(data_path + "-fold5.csv")
fold6 = pd.read_csv(data_path + "-fold6.csv")
fold7 = pd.read_csv(data_path + "-fold7.csv")
fold8 = pd.read_csv(data_path + "-fold8.csv")
fold9 = pd.read_csv(data_path + "-fold9.csv")
fold10 = pd.read_csv(data_path + "-fold10.csv")

folds = [fold1, fold2, fold3, fold4, fold5, fold6, fold7, fold8, fold9,
    fold10]

metrics = pd.DataFrame(columns=['fold', 'model_id', 'mae', 'nmae'])

for i in range(0, 10):
    h2o.init(port = 54325)

    fold_folder = "./fold" + str(i + 1)
    folds = [fold1, fold2, fold3, fold4, fold5,
                fold6, fold7, fold8, fold9, fold10]
    test_df = folds[i]
    test = h2o.H2OFrame(test_df)

    del folds[i]
    train_df = pd.concat(folds)
    train = h2o.H2OFrame(train_df)

    x = train.columns
    y = target
    x.remove(y)
    x.remove("Fim_Ordem")

    aml = H2OAutoML(
        seed=42,
        sort_metric="MAE",
        nfolds=5,
        #exclude_algos=["DeepLearning"],
```

```
        max_runtime_secs=3600,
    )

    aml.train(x=x, y=y, training_frame=train)

    lb = aml.leaderboard.as_data_frame()
    lb.to_csv(fold_folder + "/leaderboard.csv", index=False)

    perf = aml.training_info
    perf = json.dumps(perf)
    f = open(fold_folder + "/perf.json", "w")
    f.write(perf)
    f.close()

    h2o.download_model(aml.leader, path=fold_folder)

    preds = aml.leader.predict(test)
    mae = mean_absolute_error(test[y].as_data_frame(), preds.as_data_frame
        ()['predict'])
    nmae = mae / (max(test.as_data_frame()[y]) - min(test.as_data_frame()[y
        ]))
    metrics = metrics.append({'fold': (i+1), 'model_id': lb["model_id"][0],
         'mae': mae, 'nmae': nmae}, ignore_index=True)

    metrics.to_csv("./metrics.csv", index=False)

    h2o.cluster().shutdown()

    import time
    time.sleep(5)
```

Listing A.6: Aplicação AutoMLH20

```
label_column = 'DeltaT'
metric="MAE"
task = "reg"

fold1 = read.csv(paste(data_path, "-fold1.csv", sep = ""), sep=",", header
    = T)
fold2 = read.csv(paste(data_path, "-fold2.csv", sep = ""), sep=",", header
    = T)
fold3 = read.csv(paste(data_path, "-fold3.csv", sep = ""), sep=",", header
    = T)
fold4 = read.csv(paste(data_path, "-fold4.csv", sep = ""), sep=",", header
    = T)
```

```
fold5 = read.csv(paste(data_path, "-fold5.csv", sep = ""), sep=",", header
    = T)
fold6 = read.csv(paste(data_path, "-fold6.csv", sep = ""), sep=",", header
    = T)
fold7 = read.csv(paste(data_path, "-fold7.csv", sep = ""), sep=",", header
    = T)
fold8 = read.csv(paste(data_path, "-fold8.csv", sep = ""), sep=",", header
    = T)
fold9 = read.csv(paste(data_path, "-fold9.csv", sep = ""), sep=",", header
    = T)
fold10 = read.csv(paste(data_path, "-fold10.csv", sep = ""), sep=",",
    header = T)

results <- data.frame(matrix(ncol = 6, nrow = 0))
col_names <- c("target", "tool", "model", "mae", "nmae", "time")
colnames(results) <- col_names

for (x in 0:9) {
    fold_folder = paste("./fold", as.character(x+1), sep="")
    folds = list(fold1, fold2, fold3, fold4, fold5, fold6, fold7, fold8,
        fold9, fold10)
    test_df = folds[[(x+1)]]
    folds = folds[-(x+1)]
    train_df <- do.call(rbind, folds)

    train_df = train_df[,-c(4,5)]
    test_df = test_df[,-c(4,5)]

    inputs=ncol(train_df)-1

    train_df[is.na(train_df)] = 0
    test_df[is.na(test_df)] = 0

    sm=mparheuristic(model="automl3",n=NA,task=task, inputs=inputs)
    method=c("kfold",5,123)
    search=list(search=sm,smethod="auto",method=method,metric=metric,convex
        =0)

    M=fit(DeltaT~.,data=train_df,model="auto",search=search,fdebug=TRUE)

    P=predict(M,test_df)

    mae = round(mmetric(test_df$DeltaT,P,metric=metric),2)
    nmae = round(mmetric(test_df$DeltaT,P,metric="NMAE"),2)
    best_model = M@model
```

```
    results[nrow(results) + 1,] = c(label_column,"rminer",best_model,mae,
        nmae,M@time)

    perf = paste('{"time":',
                 M@time,
                 ',"best_model":"',
                 best_model,
                 '","test_metric":',
                 mae,
                 '}',
                 sep = "")

    write(perf, paste(fold_folder, "/perf.json", sep= ""))
    save(M, file = paste(fold_folder, "/model.RData", sep= ""))

    write.csv(results,"./metrics.csv", row.names = FALSE)
```

Listing A.7: Aplicação rminer

```
target = 'DeltaT'

fold1 = pd.read_csv(data_path + "-fold1.csv")
fold2 = pd.read_csv(data_path + "-fold2.csv")
fold3 = pd.read_csv(data_path + "-fold3.csv")
fold4 = pd.read_csv(data_path + "-fold4.csv")
fold5 = pd.read_csv(data_path + "-fold5.csv")
fold6 = pd.read_csv(data_path + "-fold6.csv")
fold7 = pd.read_csv(data_path + "-fold7.csv")
fold8 = pd.read_csv(data_path + "-fold8.csv")
fold9 = pd.read_csv(data_path + "-fold9.csv")
fold10 = pd.read_csv(data_path + "-fold10.csv")

cols_to_drop = ["Inic_Ordem", "Fim_Ordem"]

fold1 = fold1.drop(columns=cols_to_drop)
fold2 = fold2.drop(columns=cols_to_drop)
fold3 = fold3.drop(columns=cols_to_drop)
fold4 = fold4.drop(columns=cols_to_drop)
fold5 = fold5.drop(columns=cols_to_drop)
fold6 = fold6.drop(columns=cols_to_drop)
fold7 = fold7.drop(columns=cols_to_drop)
fold8 = fold8.drop(columns=cols_to_drop)
fold9 = fold9.drop(columns=cols_to_drop)
fold10 = fold10.drop(columns=cols_to_drop)
```

```
folds = [fold1, fold2, fold3, fold4, fold5, fold6, fold7, fold8, fold9,
    fold10]

from tpot import TPOTRegressor

results = pd.DataFrame({'fold': [], 'model' : [], 'mae' : [], 'nmae' : [],
    'time': []})

for i in range(0, 10):
    fold_folder = "./fold" + str(i+1)
    folds = [fold1, fold2, fold3, fold4, fold5, fold6, fold7, fold8, fold9,
        fold10]
    test_df = folds[i]
    X_test = test_df.drop(columns=[target]).to_numpy()
    y_test = test_df[target].to_numpy()

    del folds[i]
    train_df = pd.concat(folds)
    X_train = train_df.drop(columns=[target]).to_numpy()
    y_train = train_df[target].to_numpy()

    tpot = TPOTRegressor(max_time_mins=60, verbosity=3, random_state=42,
       scoring='neg_mean_absolute_error', cv=5, n_jobs=-3, early_stop=3)

    start = datetime.now().strftime("%H:%M:%S")
    tpot.fit(X_train, y_train)
    end = datetime.now().strftime("%H:%M:%S")

    t1 = datetime.strptime(start, "%H:%M:%S")
    t2 = datetime.strptime(end, "%H:%M:%S")
    duration = (t2 - t1).seconds

    tpot.export(fold_folder + "/pipeline.py")

    with open(fold_folder + '/pipeline.py') as f:
        x = True
        while x:
            line = f.readline()
            if "Average CV score on the training set was" in line:
                mae = line
                x = False

    train_mae = float(mae.split(":")[1][2:].replace("\n", ""))

    mae = tpot.score(X_test, y_test)
    nmae = 100 * (mae / abs(max(y_test) - min(y_test)))
```

```
perf = {
"training_score": train_mae,
"test_score": mae,
"start": start,
"end": end
}

perf = json.dumps(perf)
f = open(fold_folder + "/perf.json","w")
f.write(perf)
f.close()

results = results.append({'fold': (i+1), 'model' : "", 'mae' : mae, '
    nmae' : nmae, 'time': duration}, ignore_index=True)
results.to_csv("./metrics.csv", index=False)
```

Listing A.8: Aplicação TPOT