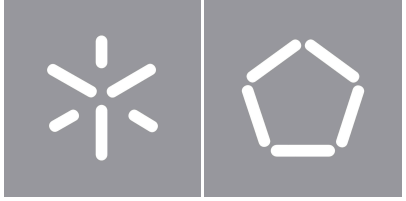
Universidade do Minho
Escola de Engenharia
Departamento de Informática

Rodolfo António Vieira da Silva

Inteligência Artificial aplicada à Infraestrutura de Carregamentos para Veículos Elétricos

janeiro 2023



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Rodolfo António Vieira da Silva

Inteligência Artificial aplicada à Infraestrutura de Carregamentos para Veículos Elétricos

Dissertação de Mestrado
Mestrado Integrado em Engenharia Informática

Trabalho realizado sob a orientação do
Professor Doutor César Analide
Professor Doutor Bruno Fernandes

janeiro 2023

Direitos de autor e condições de utilização do trabalho por terceiros

Agradecimentos

Durante o decorrer da presente dissertação de mestrado, tive a oportunidade de contar com o apoio de diversas pessoas. Gostava de utilizar esta oportunidade para lhes agradecer e demonstrar quão grato estou pelo seu suporte.

Em primeiro lugar, quero agradecer ao Professor Doutor César Analide, por me ter disponibilizado esta dissertação e também pela sua orientação e disponibilidade demonstradas.

Agradeço, de igual forma, ao Professor Doutor Bruno Fernandes, por me ter ajudado ao longo deste projeto, fornecendo a sua opinião quando necessário e instruindo-me para o caminho correto.

À minha família, por todo o suporte que me forneceram ao longo do meu percurso académico, e por me aceitarem mesmo nos tempos difíceis.

Por fim, quero agradecer a todos os meus amigos, que me disponibilizaram o tempo deles para me ajudar quando precisava e por todo o tempo que passei com eles no decorrer do meu percurso académico. Significou bastante para mim, e estou eternamente grato.

Rodolfo Silva.

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

Inteligência artificial aplicada à infraestrutura de carregamentos para veículos elétricos

A capacidade de resolução de problemas pela Inteligência Artificial encontra-se em constante expansão, pelo que a otimização do seu funcionamento em sociedade está dependente da capacidade de extração das aplicabilidades da tecnologia. Uma das áreas da Inteligência Artificial em que tem sido visível uma evolução significativa é a de *Machine Learning*, cujos algoritmos têm revelado um crescente nível de especialização na resolução de diversos problemas.

Na presente dissertação, pretendeu-se construir um modelo capaz de auxiliar na recomendação de configurações ideais para novas estações de carga para veículos elétricos, com a assistência de modelos de *Machine Learning*. A revisão da literatura revelou uma extensiva análise sobre problemas de previsão na área de *Machine Learning*, pelo que algoritmos tradicionais de *Machine Learning* e algoritmos da subárea de *Deep Learning* se demonstram adequados para a resolução do problema proposto nesta dissertação. O *dataset* empregue neste projeto, com dados referentes a Portugal, foi construído com a assistência de diversas *API* e, posteriormente ao seu tratamento, foram aplicados seis algoritmos de *Machine Learning*, com o intuito de treinar um modelo que conseguisse prever a utilização futura de postos de carga.

De entre os algoritmos avaliados, o *Random Forest Regressor* e *eXtreme Gradient Boosting* são aqueles que apresentam maior capacidade na resolução do problema em questão, com um MAE 6.6220 e 6.6310, respetivamente. O modelo de *Random Forest Regressor* é aquele que melhor se adequou para a previsão de utilização futura dos postos de carga, tendo sido utilizado para a construção do modelo de recomendação.

Palavras-Chave: Machine Learning, Mobilidade Elétrica, Problemas de Previsão, Séries Temporais

Abstract

Artificial Intelligence applied to the charging infrastructure for Electric Vehicles

The ability to solve problems through the use of Artificial Intelligence finds itself in constant expansion, its proper use in society at large is dependent on our ability to properly put this technology to use. One of the fields of Artificial Intelligence where significant progress has been noted, is the field of Machine Learning, where it's algorithms have found themselves showing an increasing level of specialization in solving diverse tasks.

In this dissertation, the main goal was to build a model capable of advising the user of ideal arrangements for electric vehicle charging stations, with the assistance of Machine Learning models. The literature review revealed that there have been extensive studies about forecasting problems within the field of Machine Learning, where both Machine Learning algorithms and Deep Learning algorithms are up to the task presented in dissertation. The employed dataset used within this dissertation focused on Portugal. Its construction was assisted through the use of several different API that allowed garnering the necessary data. Six Machine Learning algorithms were applied, the intent was to build a model that could predict occupancy rates for electric vehicle charging stations.

Of the six, Random Forest Regressor and eXtreme Gradient Boosting were the ones that stood out the most in solving the task at hand, they obtained an MAE of 6.6220 and 6.6310, respectively. Having shown the best results, the Random Forest Regressor model was the one employed in the creation of the final model that advises the user.

Keywords: Electric Mobility, Forecasting Problems, Machine Learning, Time Series

Índice

Acrónimos

ANN Artificial Neural Networks. 17

Bi-LSTM Bi-directional Long Short Term Memory Networks. 17

CCS Combined Charging System. 4

CDR Charge Detail Record. 21

CNN Convolutional Neural Network. 8, 9

DFN Deep Feedforward Network. 8, 10

DL Deep Learning. 7, 8, 16, 17, 19

EVSE Electric Vehicle Supply Equipment. 20

GB Gradient Boosting. 18

GBRT Gradient Boosting Regression Trees. 17

GRU Gated Recurrent Unit. 16, 17

IA Inteligência Artificial. 2, 5

LSTM Long Short Term Memory Networks. v, viii, 13–17, 19, 22

MAE Mean Absolute Error. 7, 17, 18

MAPE Mean Absolute Percentage Error. 7, 17

ML Machine Learning. 1–3, 5–8, 16–19

PM Persistence Method. 17

RF Random Forests. 17, 18

RMSE Root Mean Squared Error. 6, 17, 18

RNN Recursive Neural Network. viii, 8–14, 16, 17, 22

SAE Sparse AutoEnconder. 17

SARIMAX Seasonal AutoRegressive Integrated Moving Average model with eXternal predictors. 17

VE Veículo Elétrico. 1–4, 16, 18–21

XGBoost eXtreme Gradient Boosting. 18

Índice de Figuras

Índice de Tabelas

Índice de Considerações

Capítulo 1

Introdução

1.1 Contexto e motivação

A prosperidade da humanidade ao longo da sua existência tem vindo a trazer notáveis melhorias na qualidade e esperança de vida das pessoas. No entanto, estas não ocorrem sem as suas consequências negativas, uma delas sendo as alterações ambientais, mais concretamente o aquecimento global.

Atualmente, o setor automóvel é dos setores que mais contribuem para a poluição do ar (Deb, 2021). Tendo este fator em consideração, foi sugerida a utilização de Veículos Elétricos (VE) como forma de abrandar o progresso do aquecimento global. Os VE apresentam uma pegada ecológica de até 70% mais pequena do que os carros de combustão (Harrabin, 2020), representando um dos principais motivos para o notório crescimento do número de VE.

Atualmente, Portugal é um dos países que apresenta maior expansão nas vendas de VE (Razmjoo et al., 2022), conduzindo à necessidade de aumentar a rede de carregamento para VE de forma robusta.

Assim sendo, surgiu a necessidade de criar uma ferramenta capaz de auxiliar empresas na toma de decisões mais acertadas sobre onde investir em novos postos de carga para VE, de um modo lucrativo para a empresa e eficiente para a rede.

1.2 Objetivos

A presente dissertação tem como objetivo a realização de uma aglomeração de diferentes dados relativos à utilização dos postos de carga da rede nacional, dados geográficos e dados meteorológicos, de modo a criar um modelo de *Machine Learning* (ML) (uma das áreas constituintes do campo de Inteligência Artificial (IA)), que auxilie na instalação de postos de carga, fornecendo informação sobre as localizações mais rentáveis destes postos, bem como o tipo de equipamento mais adequado a utilizar.

1.3 Questões de investigação

Na presente dissertação, pretende-se responder a três questões:

- Qual é o estado atual da evolução da infraestrutura de carregamento?
- Como se deve utilizar a IA para prever a utilização futura dos postos de carga?
- Como se pode evoluir a infraestrutura de carregamento através da utilização de IA?

Na primeira questão, pretende-se conhecer como se pode encontrar a infraestrutura de carregamento em Portugal, isto é, se o dilema da galinha e do ovo já foi resolvido e se existe uma proporção apropriada entre número de VE e postos de carga e, ainda, se esses mesmos postos de carga estão apropriadamente distribuídos pelo país. No que respeita à segunda questão, tenciona-se avaliar os modelos e métodos de ML a utilizar para melhorar a previsão da utilização futura de postos de carga. Por fim, a terceira e principal questão pretende estudar a melhor forma de auxiliar na evolução da infraestrutura, neste caso através da melhor recomendação de lugares para colocação dos postos de carga, através do recurso a dados de diferentes fontes, inclusive os dados de previsão obtidos no modelo da segunda questão.

1.4 Estrutura da dissertação

Esta dissertação encontra-se dividida entre 6 capítulos principais. O presente capítulo introduz os conteúdos que serão abordados e estudados nesta dissertação e respetiva motivação. No capítulo 2, será realizada uma revisão bibliográfica que aborda os tópicos de maior relevância para a concretização do atual projeto. Já no capítulo 3, será descrita a estrutura do principal *dataset* utilizado, e os meios pelos quais a informação contida no *dataset* foi adquirida. Será também apresentado um breve estudo do *dataset* para compreensão das nuances presentes nos dados. Além disso, será descrito o pré-processamento aplicado aos dados. Quanto ao capítulo 4, são apresentados os detalhes de implementação do modelo e interpretados alguns dos resultados. No que respeita o capítulo 5, são discutidos os resultados obtidos no desenvolver deste projeto e, uma pequena interpretação dos resultados. Por fim, no último capítulo, são expostas as principais conclusões retiradas na elaboração do projeto e possíveis melhorias ao mesmo.

Capítulo 2

Estado de arte

Nesta secção, será fornecida uma visão geral do estado atual da mobilidade elétrica em Portugal. Além disso, será apresentado o estado da arte sobre a área de ML, sendo explorados modelos cruciais para o desenvolvimento do projeto em capítulos subsequentes e, por fim, serão apresentados alguns estudos previamente realizados que englobam a área do planeamento das infraestruturas de postos de carga e ML.

2.1 Mobilidade elétrica

A mobilidade elétrica pode ser descrita como o uso de carros, bicicletas, motas, autocarros e camiões elétricos. A característica comum entre estes é o facto de que operam inteira ou parcialmente de energia elétrica, sendo esta obtida a partir da rede de energia da cidade.

Para a obtenção dessa energia, é necessário um intermediário (neste caso os postos de carga). Existem três categorias de postos: *slow charging*, *fast charging* e *rapid charging*, pelo que existe ainda uma grande variedade de conectores que podem possuir.

2.1.1 Slow/Fast charging

Os modos *slow* e *fast charging* utilizam corrente alternada e estão tipicamente presentes no carregamento dos VE nas casas, no trabalho ou até no destino para onde uma pessoa se desloca, de modo a poder carregar o veículo enquanto lá se encontra.

Estes modos possuem até dois tipos de conectores: o conector tipo 1, padrão dos Estados Unidos; e o conector tipo 2, predominante em novos carros e também mais recente.

2.1.2 Rapid charging

Os postos de carga com *Rapid Charging* utilizam uma corrente direta, contrariamente aos previamente mencionados, permitindo que os VE sejam carregados mais rapidamente. Estes postos são habitualmente

colocados em lugares como as estações de serviço, que permitem fazer uma pequena paragem para carregar antes de se continuar viagem.

Todos os postos de carga de corrente direta possuem cabos para conectores do estilo CHAdeMO e Combined Charging System (CCS). Os CHAdeMO são os conectores originais destes postos de carga, mas os CCS estão a tornar-se frequentes e apresentam também melhor poder de carga.

2.1.3 Dilema da galinha e do ovo

Um dos problemas mais discutidos referente à adoção e desenvolvimento da área de mobilidade elétrica é o da galinha e do ovo (Mathieu, 2018). O dilema consiste na pergunta "Qual precisa de aparecer primeiro? Os VE ou a infraestrutura de carregamento?". Por um lado, se não existir um número considerável de VE, é improvável que nasçam potenciais investidores em novas estruturas de carregamentos. Por outro lado, se não existir uma sólida infraestrutura de carregamento, a população não se vai comprometer na compra de um VE.

Para a resolução do problema, a União Europeia forneceu uma estrutura de políticas na sua legislação sobre mobilidade elétrica, para ser utilizada como base pelos Estados Membros nos seus planos, com o objetivo de aumentar a aceitação de VE por parte da população (of Auditors, 2021).

No caso de Portugal, este dilema está a ser resolvido de duas formas. A primeira consiste na criação de incentivos monetários, constituídos pelo reembolso de um montante dependente do VE adquirido, pela dedução total do IVA para VE abaixo de 62.500€ e através da isenção das taxas de estrada e das taxas de veículos (PROTESTE, 2022). A segunda forma consiste no comprometimento, em conjunto com entidades privadas, em desenvolver uma rede de postos de carga. Esta iniciativa tinha como objetivo um total de 2000 postos de carga ativos durante 2021 (PROTESTE, 2022). Dado que, atualmente existem 3268 postos de carga em Portugal (MOBI.E, 2021), pode considerar-se esta iniciativa um sucesso.

Apesar da resolução deste dilema estar a progredir de forma positiva em Portugal, ainda se encontra longe do ideal. Segundo a Diretiva 2014/94/EU, é recomendado um total de 10 VE por posto de carga (Commision, 2021; Parliament and the council of the European Union, 2014). No entanto, existem cerca de 73 mil VE em Portugal (PROTESTE, 2021) para um total de 3268 postos de carga, resultando em cerca de 22 VE por posto de carga, valor consideravelmente superior à recomendação Europeia.

2.2 Dados temporais

Uma **série temporal** é um conjunto de observações onde a informação está dispersa por intervalos de tempo. Estes intervalos de tempo podem ser contínuos (métricos) ou podem ser irregulares (eventos). No contexto de previsão futura, os dados constituídos por eventos não são utilizados, dado serem imprevisíveis. Os dados presentes numa série temporal referem-se a um sujeito apenas. A titulo de exemplo, o conjunto dos dados referentes ao preço diário de um produto p no supermercado S durante seis meses seria uma série temporal.

Os **dados de corte transversal** são referentes a um único instante temporal, mas sobre múltiplos sujeitos. Neste caso, a evolução dos dados ao longo do tempo não é relevante, pelo que o interesse está na comparação do estado atual de todos os sujeitos abrangidos. Deste modo, os dados de corte transversal são o contrário de séries temporais. Como exemplo para dados de corte transversal tem-se o salário médio de cada família num país num certo mês.

Os **dados em painel** consistem na junção de séries temporais com dados de corte transversal, isto é, um conjunto de observações dispersas por intervalos de tempo, métricos ou eventos, sobre múltiplos sujeitos. Como resultado, os dados em painel apresentam vantagens comparativamente aos dois formatos discutidos anteriormente. A vantagem mais notável é a habilidade dos dados em painel conterem mais informação e variabilidade, permitindo o mapeamento dos comportamentos comuns e individuais dos grupos. Um possível exemplo representativo de dados em painel seria o preço das ações de várias empresas ao longo do tempo.

2.3 Machine Learning

A *Machine Learning* (ML) é uma área da ciência de computadores e tem origem na IA. A ideia fundamental desta área consiste em treinar algoritmos de aprendizagem para a realização automática de análises de dados, com o objetivo de obter um modelo capaz de atuar perante novas entradas, fundamentando-se na informação aprendida (P.Murphy, 2012).

A primeira fase do treino de um algoritmo é a aprendizagem, onde é recomendado serem empregues *datasets* de treino bem formatados e estruturados. Estes *datasets* são frequentemente compostos por exemplos, cada um constituído por um conjunto de valores (atributos), representados como um vetor $x_i \in \mathbb{R}^n$, onde n é o número de atributos dentro de um exemplo. A segunda fase do treino corresponde à avaliação do modelo obtido posteriormente à aprendizagem do algoritmo, onde é tipicamente fornecido um *dataset* de teste com exemplos novos para o modelo, com a intenção de serem avaliados.

2.3.1 Paradigmas de aprendizagem

A ML é habitualmente dividida em três módulos principais: a aprendizagem supervisionada, aprendizagem não supervisionada e aprendizagem por reforço. No que concerne a aprendizagem supervisionada, o objetivo consiste em, para um dado *dataset* de treino $D = \{(x_i, y_i)\}_{i=1}^{N}$, realizar um mapeamento do vetor de *inputs* x para os *outputs* y, onde N é o número total de exemplos e $y \in \{0, ..., K\}$, em que K corresponde ao número de classes (*targets*). O seu propósito está em conseguir prever um *output* y a partir de um dado *input* x. Alguns exemplos de algoritmos de aprendizagem supervisionada são: *Support Vector Machines*, *Naive Bayes*, *Linear Regression*, entre outros (P.Murphy; Goodfellow et al., 2012; 2016). Comparativamente à aprendizagem supervisionada, a aprendizagem não supervisionada recebe apenas um *dataset* $D = \{(x_i)\}_{i=1}^{N}$, isto é, o algoritmo utiliza os atributos sem conhecer a sua

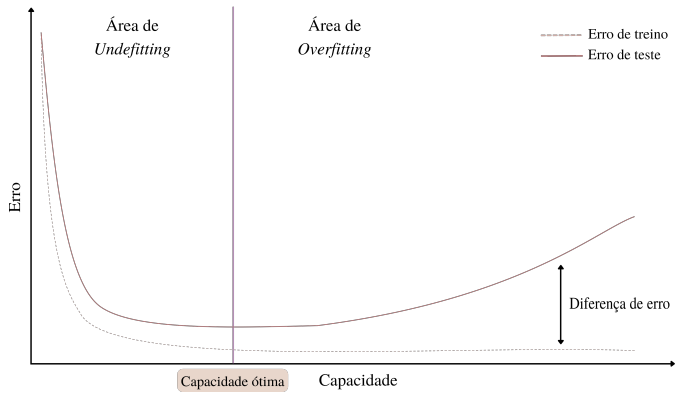
classificação, permitindo encontrar padrões e estruturas pelas quais agrupar os *inputs*. Alguns exemplos de algoritmos de aprendizagem não supervisionada são: *K-means Clustering*, *Isolation Forest*, *Principal Component Analysis*, entre outros (P.Murphy; Goodfellow et al., 2012; 2016). Por fim, a aprendizagem por reforço recorre a recompensas após uma ação positiva e penalizações no caso em que uma ação negativa ocorre. O objetivo destes modelos consiste em atingir a maior recompensa possível (Otterlo and Wiering, 2012). No problema de estudo desta dissertação, foi desconsiderada a utilização deste último paradigma.

2.3.2 Capacidade de generalização

Um dos maiores desafios em ML esta em garantir o correto funcionamento dos modelos desenvolvidos quando expostos a informação previamente estudada e informação nova. Nesta área, está presente o conceito da generalização em modelos de ML, ou seja, para um modelo demonstrar um bom desempenho, é necessário que a taxa de erro na classificação de *inputs* já estudados não tenha uma grande disparidade da taxa de erro na classificação de *inputs* não vistos, e que ambas as taxas tenham um valor reduzido (Goodfellow et al., 2016).

Quando um modelo desenvolvido apresenta um mau desempenho, pode dever-se a um dos seguintes problemas que ocorrem mais frequentemente na área de generalização, *underfitting* ou *overfitting*. Por um lado, o *underfitting* ocorre quando um algoritmo não se adapta o suficiente aos dados de treino e, consequentemente, obtém um erro bastante elevado na classificação dos *inputs* de treino. Por outro lado, o *overfitting* ocorre quando um algoritmo se adapta em demasia aos dados de treino, tornado-se incapaz de realizar previsões em dados não analisados previamente, originando uma elevada discrepância entre a taxa de erro na classificação de *inputs* de treino e a taxa de erro na classificação de novos *inputs* (P.Murphy, 2012). O modo de evitar a ocorrência destes dois fenómenos é através da otimização da capacidade do modelo, que determina a aptidão do modelo em aprender uma vasta variedade de funções. Se a capacidade for demasiado elevada, o algoritmo entra em *overfitting*; se a capacidade for muito reduzida o modelo entra em *underfitting* (Goodfellow et al., 2016). Na Figura 1, observa-se a evolução do erro de treino e do erro de teste com o aumento da capacidade do modelo.

Figura 1: Relação entre capacidade e erro. Adaptado de Goodfellow et al. (2016)



2.3.3 Avaliação de desempenho

Para a avaliação do desempenho de modelos de ML, é necessário recorrer a métricas capazes de fornecer uma noção deste desempenho. Habitualmente, são utilizadas três métricas para avaliar o desempenho, *Root Mean Squared Error* (RMSE) (Equação 2), *Mean Absolute Percentage Error* (MAPE) (Equação 3) e *Mean Absolute Error* (MAE) (Equação 4).

$$MSE = \frac{1}{N}\sum_{i_1}^{n}\left((\widehat{y}_i - y_i)^2\right) \tag{1}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i_1}^{n}\left((\widehat{y}_i - y_i)^2\right)} \tag{2}$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{n}\left(\left|\frac{\widehat{y}_i - y_i}{y_i}\right| \times 100\right) \tag{3}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{n}|\widehat{y}_i - y_i| \tag{4}$$

Nas equações 1, 2 e 3, N corresponde ao número total de casos avaliados, $\widehat{y}_i$ ao valor previsto pelo modelo para o caso i e y_i ao valor real de i presente nos dados.

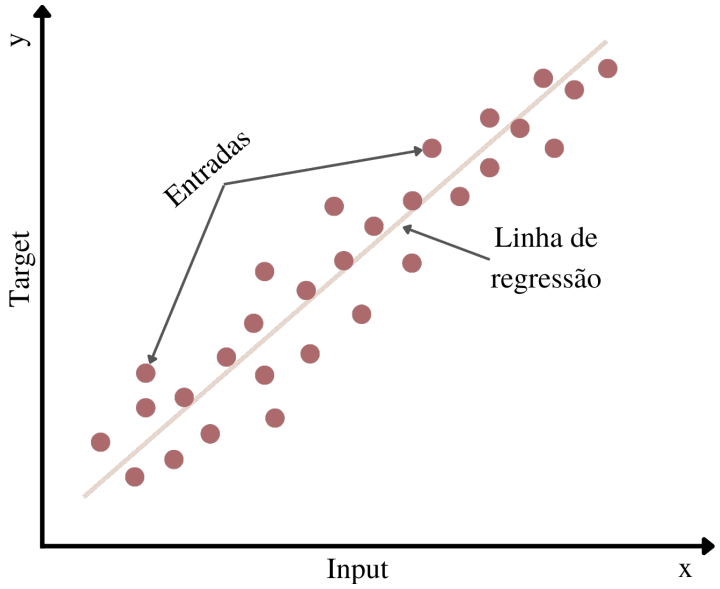
2.3.4 Algoritmos de Machine Learning

Dentro da área de ML, existe uma imensidão de diferentes algoritmos, cada um apresentando um conjunto de aspetos positivos e negativos. Assim sendo, estudou-se um grupo de algoritmos de ML mais frequentemente utilizados.

2.3.4.1 Linear Regression

A *Linear Regression* (LR) é um algoritmo popular e vastamente procurado devido à sua simples representação do problema. Este algoritmo mapeia os valores x_i de um *input* numa relação linear com o *target* y (Figura 2). Um algoritmo de LR denomina-se de simples ou multivariado dependendo do número de atributos por *input*. A cada atributo é atribuído um coeficiente B_x no cálculo da função (Equação 5) (Nasteski, 2017; Maulud and Abdulazeez, 2020).

Figura 2: Exemplo gráfico da função gerada por um modelo de LR.



$$y = B_0 + B_1 \times x_1 + B_2 \times x_2 + B_3 \times x_3 + ... + \varepsilon \tag{5}$$

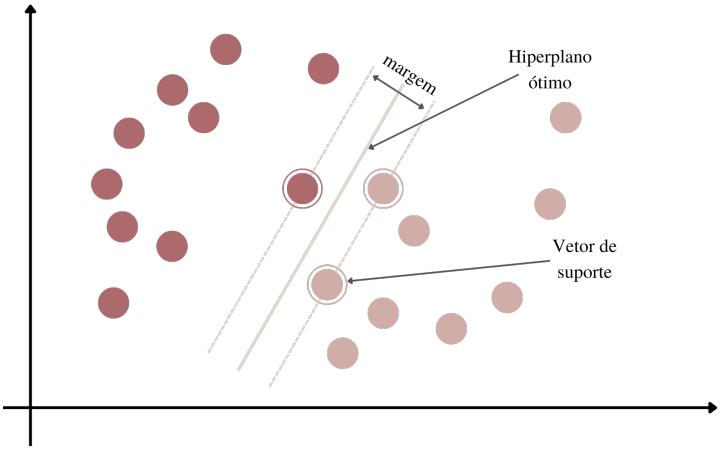
Para treinar um modelo de LR, é necessário estimar os valores apropriados para os coeficientes B_x, recorrendo-se ao método do quadrado mínimo. Este método procura obter o menor valor possível para a soma dos erros ao quadrado. Assim sendo, para todas as entradas do *input*, é calculada a distância da entrada à linha de regressão e realizada a soma do quadrado de todas as distâncias obtidas. O algoritmo visa minimizar este valor através da alteração dos valores dos coeficientes. O método do quadrado mínimo é o mais frequente e simples de utilizar, existindo, no entanto, outros métodos que podem ser utilizados, não tendo sido abordados (Nasteski, 2017; Maulud and Abdulazeez, 2020).

2.3.4.2 Support Vector Machines

As *Support Vector Machines* (SVM) são um algoritmo de aprendizagem supervisionada com elevada popularidade. De um modo geral, este algoritmo consiste em produzir diversos planos de corte entre os dados de treino, de modo a atribuir um *target* às entradas, dependendo do lado do plano de corte em que estes se encontram.

Na Figura 3, observa-se um plano de corte simples aplicado a um problema de classificação binário. De igual modo, as SVM procuram aprender uma função para calcular o hiperplano ótimo. Esta função procura maximizar a distância entre as classes. Os pontos do *dataset* para os quais o valor dessa função é 1 serão aqueles pertencentes aos vetores de suporte, sendo que o hiperplano ótimo encontra-se entre os vetores. (Cristianini et al., 2000)

Figura 3: Exemplo de um plano de corte num problema de classificação binário.



No entanto, em casos reais, é raramente possível atingir uma separação linear dos dados, sendo necessário recorrer ao *kernel*. O *kernel* tem como objetivo a projeção dos dados num espaço superior. O *kernel* é uma função que, quando aplicada aos dados de um *dataset*, retorna um valor que pode ser representado num outro espaço (Figura 4) (Cristianini et al., 2000). Dado existirem diversos *kernels*, um passo fulcral no treino de uma SVM está em encontrar o *kernel* adequado.

As SVM são um algoritmo que pode ser utilizado para classificação e regressão. Apesar do funcionamento interno da SVM alterar entre os dois, o funcionamento geral é idêntico ao previamente descrito.

2.3.4.3 Random Forest

O algoritmo de *Random Forests* (RF) é um algoritmo de combinação que pode ser utilizado em problemas de classificação e de regressão. Os algoritmos de combinação são algoritmos cuja resposta é um amalgamo de múltiplos algoritmos que originam um só. No caso das RF, o algoritmo é composto por N *decision trees*, em que N é um valor definido durante o treino (Horning et al., 2010; Breiman, 2001).

Para problemas de classificação, o *output* corresponde à moda dos valores previstos pelas N *decision trees*. Por exemplo, para um modelo de previsão de cores com $N = 200$ *decision trees*, se 150 delas

Figura 4: Exemplo de uma transformação dos dados para um hiperplano onde uma separação pode ser realizada.

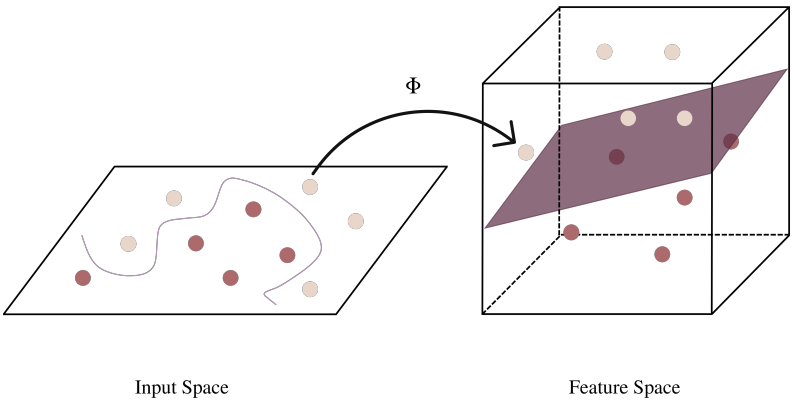
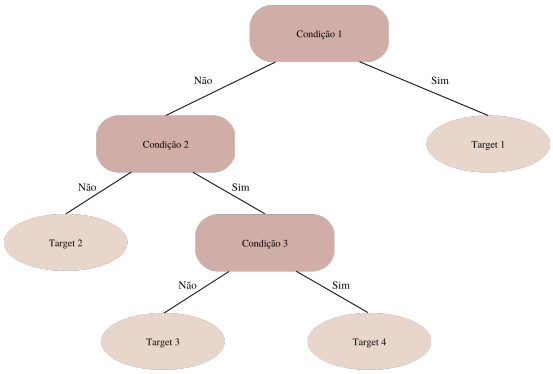


Figura 5: Exemplo de uma *decision tree* simples.



previrem a cor azul e as restantes 50 reportarem a cor rosa, o *output* do modelo é azul. No que respeita a problemas de regressão, o *output* corresponde à média dos valores previstos. Por exemplo, num modelo com $N = 3$ e valores $o_1 = 20 \,|\, o_2 = 13 \,|\, o_3 = 18$, o *output* é 17 (Horning et al., 2010; Breiman, 2001).

Para compreender o treino do algorimo de RF, é necessário conhecimento sobre o funcionamento das *decision trees*. De uma forma geral, as *decision trees* (Figura 5) são treinadas através de duas etapas fundamentais. A primeira etapa corresponde à decisão das melhores divisões. Para a árvore realizar uma divisão, é necessário utilizar uma condição baseada num dos atributos do *dataset* (por exemplo, na condição $regiao = porto$). Será escolhida a combinação de condições que apresenta o menor erro. A segunda etapa corresponde à poda da *decision tree*. Depois de obtido o menor erro possível na primeira etapa, o algoritmo irá podar as pontas da árvore com atributos de menor relevância (permitindo diminuir o *overfitting*) (Horning et al., 2010; Breiman, 2001).

O algoritmo de RF emprega um valor arbitrário de *decision trees* que são desenvolvidas através de duas etapas que envolvem seleção aleatória: na primeira etapa, e para cada *decision tree* (dado um *dataset* de tamanho N), serão amostradas N entradas com reposição, garantindo que cada árvore tenha representações diferentes do mesmo *dataset* durante o treino; na segunda etapa, serão selecionados os atributos utilizados para a construção das condições de divisão. Para cada nodo da árvore, serão selecionados a atributos de entre A, com $a \ << \ A$, determinando a condição de divisão. A grande popularidade das RF na área de ML deve-se à concretização destas duas etapas (Horning et al., 2010; Breiman, 2001).

2.3.4.4 eXtreme Gradient Boosting

Similarmente às RF, o algoritmo *Extreme Gradient Boosting* (XGBoost) pertence à família de algoritmos de combinação, onde o *output* é uma junção de múltiplas previsões, sendo igualmente composto por *decision trees*. Do mesmo modo, o algoritmo de XGBoost emprega um valor arbitrário de *decision trees* que são desenvolvidas através de duas etapas: na primeira etapa, para cada iteração do treino do algoritmo (dado um *dataset* de tamanho N), será amostrada uma percentagem desse *dataset* para treinar o algoritmo durante aquela iteração; na segunda etapa, e para cada *decision tree*, serão selecionados uma percentagem dos atributos do *dataset* para treinar cada uma das árvores. Este algoritmo difere do RFR

principalmente no modo como as *decision trees* são aplicadas e expandidas. Apesar do XGBoost poder ser utilizado para problemas de regressão e de classificação, as *decision trees* presentes no modelo são árvores de regressão que recorrem a resíduos (Equação 6) para determinar as melhores divisões através da sua propriedade contínua (Chen and Guestrin, 2016; Dmlc, 2023).

$$residuos = valor\_real - valor\_previsto \tag{6}$$

As divisões das *decision trees* são decididas através do cálculo do *similarity score* (Equação 7) e *gain* (Equação 8) pelo que será selecionada a divisão que apresenta o melhor *gain*. Para o caso do *similarity score*, "Probabilidade Anterior" corresponde à probabilidade de um evento calculado no passo anterior. λ é o parâmetro de regularização, cujo objetivo é combater o *overfitting*.

$$SimilarityScore = \frac{(\sum_{i=1}^{n} residuo_i)^2}{\sum_{i=1}^{n}[ProbabilidadeAnterior_i \times (1 - ProbabilidadeAnterior_i)] + \lambda} \tag{7}$$

$$Gain = FolhaEsquerda_{similarity} + FolhaDireita_{similarity} - Raiz_{similarity} \tag{8}$$

2.4 Deep Learning

Dada a crescente obtenção de resultados insatisfatórios, por parte dos tradicionais algoritmos de ML, na resolução de problemas, como reconhecimento de discurso ou reconhecimento de objetos, foi necessário o desenvolvimento de ML para uma metodologia mais complexa capaz de abordar estes problemas.

A área de *Deep Learning* (DL) surgiu então como forma de colmatar os problemas apresentados pelo ML. Os algoritmos de DL permitem ao computador responder a problemas complexos, separando-os em problemas mais simples e compactos. Tipicamente, um algoritmo de DL é composto por múltiplas camadas escondidas que se encontram entre a camada de entrada e a camada de saída. Estas camadas analisam diferentes vertentes diferentes dos mesmo dados, isto é, uma camada pode fazer decisões sobre as cores dos píxeis de uma imagem e a camada seguinte pode analisar os contornos do objeto presente na imagem (Goodfellow et al., 2016).

2.4.1 Deep Feedforward Network

As *Deep Feedforward Networks* (DFN), são os algoritmos característicos de DL, que servem de base para outros algoritmos mais complexos. A título de exemplo, têm-se as *Convolutional Neural Networks* (CNN) e as *Recurrent Neural Networks* (RNN).

O alvo destes modelos é desenvolver uma aproximação de uma função f^*. Assim sendo, dado o *dataset* de treino que foi fornecido, existe uma função f^* que, para o *input* x e o respetivo *output* y, é

capaz de mapear $y = f^*(x)$, pelo que, durante o treino, o modelo aproxima uma função $f(x)$ para que condiga com $f^*(x)$ o melhor possível.

As características e funcionamento destes modelos estão de certa forma associados com a sua designação. O nome *network* tem origem no facto de que estes modelos são representados através de uma composição de várias funções. O modo como estas funções se interligam entre si é denotado através de um grafo acíclico. Na prática, para um exemplo onde se possui três funções, $f^{(1)}$, $f^{(2)}$ e $f^{(3)}$, a sua composição será $f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$. Neste exemplo, o modelo possui uma profundidade de três camadas e, quanto maiores forem as estruturas em cadeia de funções, maior é a profundidade de um modelo. A última camada de um modelo é denominada de camada de saída. É nesta camada que os *datasets* de treino guiam a construção do modelo. Para todo o x presente no *dataset*, a saída na última camada é o y correspondente, ou seja, o algoritmo de aprendizagem tem de decidir por si mesmo o que analisar em cada camada escondida, de forma a que, quando chegue à camada de saída, seja possível classificar as entradas tal como indicado pelo *dataset*. A neurociência é a área na qual se inspirou o desenvolvimento destes algoritmos. A dimensão das camadas escondidas determina a largura do modelo. Cada camada pode ser analisada como um vetor, onde cada elemento exerce um papel que simula o comportamento de um neurónio. Esta simulação deve-se ao facto de cada elemento do vetor receber informação de vários elementos da camada anterior, de modo a calcular o seu próprio valor (Goodfellow et al., 2016).

Por fim, o nome *feedforward* deriva do comportamento representativo de *deep networks*, onde a informação circula pelas diferentes camadas, sendo avaliada em x pelas diferentes computações que definem f (Goodfellow et al., 2016).

2.4.2 Convolutional Neural Network

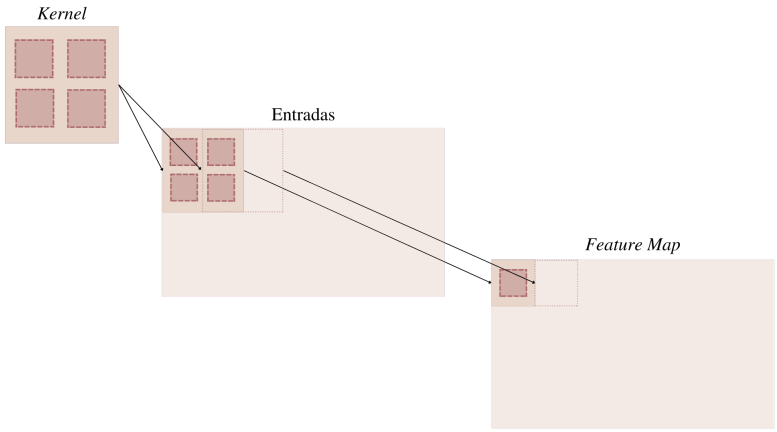
As CNN são uma família de *neural networks* especializadas em processar e analisar dados com uma estrutura em grelha, tal como imagens, representadas por uma grelha de píxeis. As CNN são descritas por (Goodfellow et al., 2016) como "*neural networks* onde, em pelo menos uma das suas camadas, utilizam convolução ao invés da multiplicação de matrizes".

A convolução consiste na aplicação de um filtro (*Kernel*), ao conjunto de entradas recebidas pelo modelo, de modo a originar um *feature map* que possui informação sobre características importantes das entradas. O *Kernel* é inferior às entradas e gera um *feature map* que é, por sua vez, também inferior (Figura 6). A aplicação recursiva do *Kernel* nas entradas traz vantagens como a procura, por toda a extensão de entradas, de características específicas que conseguem ser detetadas pelo *Kernel* e que podem facilitar a previsão sobre a entrada (Goodfellow et al., 2016).

O uso da técnica de convolução evidencia três vantagens major em comparação à utilização da multiplicação de matrizes.

A primeira vantagem está na escassez de interações, isto é, com a utilização do *Kernel*, é possível reduzir o número de interações, pelo que cada elemento do *input* não interage com todos os elementos do *output*, guardando os detalhes importantes no *feature map*. Isto providencia uma redução da memória

Figura 6: Aplicação recursiva do *kernel* nas entradas de modo a criar o *feature map*.



necessária para executar um modelo e do número de computações necessárias para construir o *output*.

A segunda vantagem é a partilha de parâmetros, em que cada peso dentro do *Kernel* é habitualmente aplicado em todas as posições do *input*, em oposição à multiplicação de matrizes onde cada valor da matriz de pesos só é utilizado uma vez. Esta característica permite ao modelo aprender apenas um par de parâmetros, reduzindo a quantidade de memória utilizada pelo modelo (Goodfellow et al., 2016) .

Por fim, a terceira vantagem é a equivariância. Com esta propriedade, a alteração do *input* implica a alteração do *output* do mesmo modo. Para uma função $f(x)$ ser equivariente a $g(x)$, é necessário que $f(g(x)) = g(f(x))$. No caso da convolução, para uma dada imagem, a aplicação da convolução ao *input* seguido da aplicação de uma escala, retorna o mesmo *output* que aplicar a escala ao *input* seguido do processo de convolução (Goodfellow et al., 2016).

2.4.3 Recurrent Neural Network

As RNN são um conjunto de *neural networks* utilizadas no processamento e análise de dados sequenciais, como por exemplo, séries temporais. Esta família de *neural networks* são especializadas no processamento de sequências de valores $x^{(1)}, ..., x^{(n)}$, permitindo uma melhor escalabilidade no tamanho máximo da sequência a analisar comparativamente a diferentes grupos de *neural networks*. Adicionalmente, a maioria das redes pertencentes a esta família são capazes de analisar sequências de comprimento variável dentro do mesmo *dataset*.

Esta família de *neural networks* distingue-se das restantes pela implementação de uma partilha de parâmetros através das diferentes partes do modelo. Esta partilha permite generalizar o modelo para sequências de dados de tamanho variável, mesmo quando este recebe como entrada uma sequência com um tamanho que não estava presente no *dataset* de treino. Como exemplo de um caso que é alvo do problema que esta partilha de parâmetros pretende resolver, apresenta-se o seguinte par de frases:

- Um carregador custa 50 euros.
- 50 euros corresponde ao preço de um carregador.

Nestas duas frases exemplo, pretende-se que o modelo consiga detetar o preço de **50 euros** como o fragmento de informação a identificar, independentemente do tamanho da frase fornecida, ou da posição relativa dessa informação na frase. Para o caso específico de uma DFN, onde esta partilha de parâmetros não existe, seria necessário que o modelo aprendesse todas as regras e possíveis estruturas da língua para ser capaz de retirar as mesmas conclusões do texto, o que não se verifica nas RNN. As DFN são descritas como um algoritmo capaz de mapear um vetor *input* para um vetor *output*. Por outro lado, as RNN são capazes, teoricamente, de mapear todo o histórico de *inputs* recebidos para cada *output* (Goodfellow et al., 2016; Graves, 2012).

2.4.3.1 Desdobramento de grafos computacionais

O desdobramento de grafos computacionais corresponde ao conceito base da partilha de parâmetros e da capacidade de generalização presentes nas RNN. Posto isto, será abordado o conceito de desdobramento de computações recorrentes, de forma a formalizá-las num grafo que representa a estrutura do conjunto de computações pertencentes.

Assumindo $s^{(t)}$ como o estado do sistema, tem-se a seguinte definição clássica de um sistema dinâmico:

$$s^{(t)} = f(s^{(t-1)}; \theta) \tag{9}$$

Nesta, verifica-se que s, no instante de tempo t, depende do estado do sistema no instante de tempo anterior. Assim, para um sistema onde existem $t = 3$ instantes de tempo, o desdobramento da função é dado por

$$s^{(3)} = f(s^{(2)}; \theta) \tag{10}$$

$$s^{(3)} = f(f(s^{(1)}; \theta); \theta) \tag{11}$$

Em razão do desdobramento aplicado, verifica-se na Equação 11 a inexistência de recursividade. Posto isto, a computação pode ser representada por um grafo computacional acíclico (Figura 7).

Utilizando a definição de um sistema dinâmico clássico (Equação 9) como base, pode ser adicionada uma influência exterior $x^{(t)}$, obtendo-se a Equação 12.

Figura 7: Representação do sistema descrito pela Equação 11 como um grafo computacional. Para todos os instantes de tempo t, os mesmos parâmetros θ são usados na parametrização de f. Adaptado de Goodfellow et al. (2016).

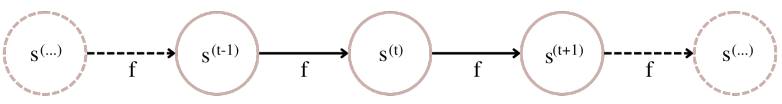
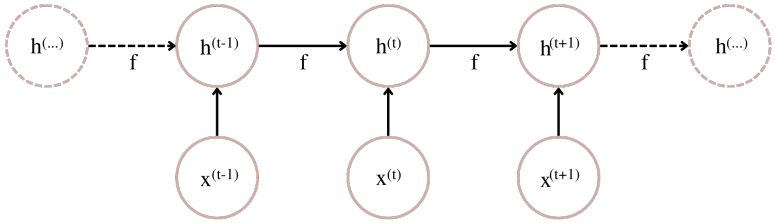


Figura 8: Uma RNN sem saídas. Representação da Equação 12 onde, a cada instante de tempo t, a informação proveniente de x é incorporada no estado h. Habitualmente, as RNN adicionam camadas de saídas para poderem realizar previsões. Adaptado de Goodfellow et al. (2016).



A Equação 12 é frequentemente utilizada em algoritmos de RNN para definir os valores das suas camadas escondidas. Desdobrando esta equação, obtém-se o grafo computacional apresentado (Figura 8).

$$h^{(t)} = f(h^{(t-1)}, x^{(t)}; \theta) \tag{12}$$

Com este desdobramento, é possível representar o estado atual após t instantes temporais, com auxílio de uma função $g^{(t)}$ (Equação 13).

$$h^{(t)} = g^{(t)}(x^{(t)}, x^{(t-1)}, \dots, x^{(2)}, x^{(1)}) \tag{13}$$

Esta função recebe toda a sequência de entradas x até o instante temporal t e produz o estado atual. O desdobramento permite fatorizar $g^{(t)}$ como uma aplicação repetida de uma função f, que permite aprender apenas um modelo f capaz de abranger entradas com um número variado de instantes temporais, e instantes temporais com sequências de diversos tamanhos. É através desta característica que é garantida a capacidade de generalização das RNN (Goodfellow et al., 2016).

2.4.3.2 Estruturas exemplo

Com a partilha de parâmetros e o desdobramento em grafos computacionais, é possível construir RNN com estruturas diversificadas, tendo sido exploradas duas estruturas relevantes de RNN.

Na Figura 9, observa-se a estrutura de uma RNN que possui ligações recorrentes entre as camadas escondidas. A cada instante temporal t, a RNN vai mapear uma sequência de entradas x para uma sequência de saídas o. A perda L será então calculada para aferir quão longe dos valores de treino y as previsões o se encontram. Por sua vez, na Figura 10, observa-se uma RNN onde apenas existem ligações recorrentes entre as previsões o do instante temporal $t-1$, com a camada escondida do instante temporal t. Estas RNN são, regra geral, menos potentes em consequência das previsões o observarem, habitualmente, uma escassez de informação importante sobre o passado. No entanto, esta adversidade possibilita uma maior facilidade em treinar esta estrutura.

Figura 9: Adaptado a partir de Goodfellow et al. (2016).

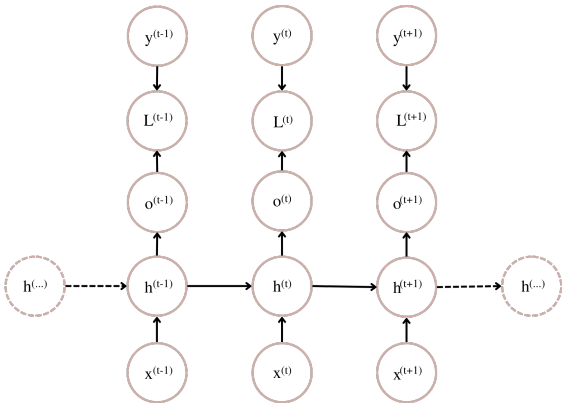
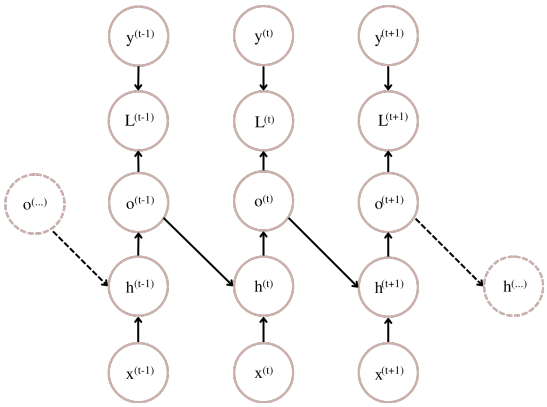


Figura 10: Adaptado a partir de Goodfellow et al. (2016).

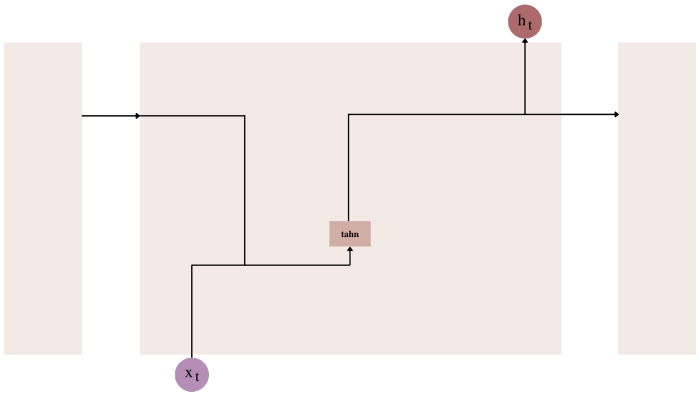


2.4.4 Long Short Term Memory Networks

As RNN foram desenvolvidas com o intuito de possibilitar a retenção de um histórico de informação passada de carácter importante (*hidden state*), de modo a fornecer contexto e facilitar previsões futuras. Contudo, as RNN originais são incapazes de salvaguardar informação relevante para um número elevado de passos temporais. Por exemplo, para uma frase como *"Na autoestrada, o limite máximo de velocidade é* **120***."*, uma RNN consegue deduzir o valor **120** baseando-se em palavras chave recentes como **autoestrada** e **limite máximo de velocidade**. No entanto, se entre as palavras chave e a informação a prever existir uma grande separação, uma RNN padrão será, provavelmente, incapaz de realizar uma previsão acertada. Tendo isto em mente, foram criadas as *Long Short Term Memory Network* (LSTM) (Olah, 2015; Irie et al., 2016).

As LSTM visam colmatar este problema através de uma reestruturação do funcionamento interno de uma RNN padrão (Figura 11). Para uma camada escondida, uma RNN padrão recebe o *hidden state* anterior, que possui informação sobre todas as entradas até ao instante temporal atual, e realiza uma concatenação do estado com a nova entrada x_t. O vetor resultante vai então passar por uma função *tahn*, cujo objetivo é regular os valores do vetor que viajam pela rede, possível através da compressão dos valores no vetor para números que se encontram no intervalo [-1,1]. Após regulados os valores, o *hidden*

Figura 11: Estrutura básica de um RNN. Adaptado de Olah (2015)



state atual pode ser utilizado para gerar uma saída.

As LSTM (Figura 12) aperfeiçoam a estrutura descrita através da adição de novos cálculos e da partilha de um parâmetro (*cell state* denotado por E na Figura 12). A informação relevante, previamente armazenada no *hidden state* (h na figura 12), é agora armazenada sob a responsabilidade de dois parâmetros. O *hidden state* vai continuar a possuir informação sobre o histórico de entradas até ao momento; e o *cell state* irá conter a informação relevante desde o início da sequência temporal. O *cell state* é modificado com recurso a portas que controlam o que alterar.

2.4.4.1 Forget Gate

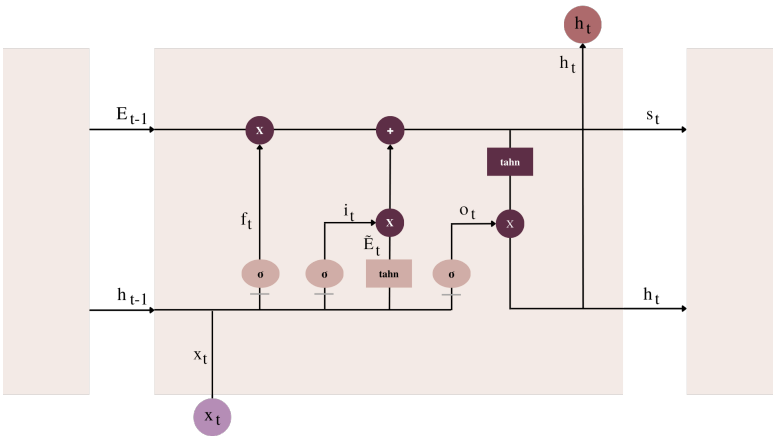
O primeiro passo numa LSTM passa por decidir a informação a excluir do *cell state* e calcular o f_t (Figura 12). Para este fim, é utilizada uma camada com uma função *sigmoid* que, através dos valores do *hidden state* e de x_t, e para cada valor no *cell state*, lhe atribui um número no intervalo [0,1]. Se lhe for atribuído o valor 0, a informação será descartada. Caso lhe seja atribuído o valor 1, a informação deve ser preservada (Olah, 2015; Irie et al., 2016).

Retomando ao exemplo previamente mencionado, o *cell state* poderá incluir o tipo de estrada a ser descrita para conseguir adivinhar o limite de velocidade. No entanto, se em x_t for mencionado um novo tipo de estrada, o *forget gate* vai ser utilizado para descartar a informação sobre se tratar de uma autoestrada, dado não ser relevante.

2.4.4.2 Input Gate

A etapa seguinte passa por decidir a informação a adicionar ao *cell state*. Este procedimento encontra-se divido em duas fases: numa primeira fase, são decididos os valores a adicionar do *cell state* através de uma camada com uma função *sigmoid* (*input gate layer*), obtendo-se i_t. Posteriormente, é calculado um vetor com novos valores candidatos (E'_t) que poderão ser adicionados ao *cell state* (Olah, 2015; Irie et al., 2016).

Figura 12: Estrutura padrão de uma LSTM composta por um *hidden state* h e um *cell state* E, que interagem através do uso de três portas. Adaptado de Olah (2015)



No exemplo descrito anteriormente, seria ideal adicionar ao *cell state* o tipo da nova estrada, substituindo o tipo previamente referenciado, que será eliminado.

2.4.4.3 Atualizar Cell State

O terceiro passo de uma LSTM passa por utilizar o f_t, i_t e E'_t para atualizar o *cell state* de E_{t-1} para E_t. Inicialmente, deve multiplicar-se E_{t-1} por f_t, de forma a eliminar a informação irrelevante, seguido da multiplicação de E'_t por i_t, para igualar os valores de E'_t que não são para adicionar a 0 e, por fim, realizado um somatório entre E_{t-1} e E'_t dando origem a E_t (Olah, 2015; Irie et al., 2016).

No exemplo anterior, seria nesta fase que a LSTM iria atualizar o *cell state* para obter a nova informação sobre o tipo de estrada.

2.4.4.4 Output Gate

O último passo numa LSTM consiste em decidir o *hidden state* seguinte. O *output gate*, que possui informação sobre as entradas anteriores, é utilizado para fazer as previsões. Inicialmente, h_{t-1} e x_t vão passar por uma camada com uma função *sigmoide* para decidir a informação a colocar na saída, originando o_t. Em seguida, é aplicada uma função *tahn* ao *cell state*, seguido de uma multiplicação com o_t para se obter o produto final h_t (Olah, 2015; Irie et al., 2016).

Seguindo o contexto dos exemplos anteriores, nesta fase a LSTM poderia decidir passar no *hidden state* informação sobre se o limite de velocidade a prever era o máximo ou o mínimo.

2.4.4.5 Variantes de LSTM

As LSTM permitiram um aumento significativo na fiabilidade das RNN. A estrutura previamente exposta corresponde à estrutura padrão originalmente proposta. No entanto, a estrutura das LSTM pode ser adaptada ao problema em questão. Alguns exemplos de LSTM com arquiteturas diferentes são: *Gated Recurrent Units* (GRU), propostas por Chung et al. (2014), *Depth Gated* RNN, propostas por Yao et al. (2015) e, por fim, as *Clockwork* RNN, por Koutník et al. (2014). Apesar das arquiteturas possuírem características distintas, o seu desempenho é semelhante. Um estudo realizado por Greff et al. (2017) comparou oito arquiteturas de LSTM diferentes com a original. Os resultados obtidos revelaram que as arquiteturas apresentavam um desempenho semelhante entre elas, com algumas a obter um desempenho ligeiramente superior em determinados problemas.

2.5 Aplicação de Machine e Deep Learning à infraestrutura de carregamento para veículos elétricos

O planeamento da infraestrutura de carregamento para veículos elétricos é uma componente essencial para possibilitar a adoção em massa de VE por parte da sociedade. Para um planeamento fiável, é

necessário considerar todas as vertentes que o constituem. Assim sendo, existem quatro aspetos fundamentais a avaliar:

- Previsão da procura de carregamento.
- Colocação dos postos de carga.
- Cálculo da utilização de postos de carga.
- Agendamento de postos de carga.

Uma vez que a infraestrutura de carregamento para VE é um assunto com crescente relevância, existem diversos estudos sobre a aplicação de ML e DL como solução para os aspetos previamente mencionados.

2.5.1 Previsão da procura de carregamento

A previsão da procura de carregamento consiste na previsão futura da demanda dos postos de carga durante o dia. Trata-se de uma área de extrema importância dado que a utilização VE influencia as horas onde existe um pico de procura de energia (Tan et al., 2017), podendo sobrecarregar a rede elétrica se esta não estiver preparada.

Num estudo conduzido por Buzna et al. (2019), pretendeu-se comparar os modelos de ML e modelos analíticos de análise de séries temporais, como *Seasonal AutoRegressive Integrated Moving Average model with external predictors* (SARIMAX). Para modelos de ML, foram aplicados dois modelos baseados em árvores de decisão, *Random Forests* (RF) e *Gradient Boosting Regression Trees* (GBRT). Recorreu-se a dados referentes ao uso de postos de carga na região Utrecht, dos Países Baixos, entre 3 de março de 2014 a 1 de março de 2015. O desempenho dos modelos foi avaliado através do *Mean Absolute Percentage Error* (MAPE). Foram treinados quatro modelos: dois modelos SARIMAX (um que tratava todos os dias como iguais e um modelo alternativo que fazia a distinção entre os dias da semana e os de fim-de-semana); o modelo RF e GBRT. Como forma de referência para o desempenho, foi utilizado o modelo *Persistence Method* (PM).

A análise dos resultados obtidos (Tabela 1) revela que o modelo SARIMAX apresentou melhor desempenho comparativamente aos modelos de ML, que pode ser explicado pelo conjunto de dados utilizados

Tabela 1: Resultados obtidos no estudo conduzido por Buzna et al. (2019)

Model MAPE (%)	**Horizonte de previsão**		
	7 dias	14 dias	28 dias
SARIMAX single	12.02	12.35	12.97
SARIMAX alternative	12.18	12.15	12.60
RF	12.58	12.55	13.28
GBRT	12.68	13.22	13.45
PM	16.40	16.86	17.98

para o treino ser relativamente pequeno. Além disso, o melhor desempenho de 14 a 28 dias do modelo alternativo SARIMAX demonstra a importância em diferenciar os dias da semana dos dias de fim-de-semana.

Num estudo realizado por Zhu et al. (2019), pretendeu-se comparar o desempenho entre diferentes modelos de DL na previsão de procura a curto termo. Foram comparados seis modelos diferentes, ANN, RNN, LSTM, GRU, SAEs e Bi-LSTM. O desempenho de cada modelo foi avaliado com recurso às métricas de erro RMSE e MAE. Os modelos foram treinados através de um conjunto de dados entre 1 de julho de 2017 a 30 de junho de 2018. Os resultados obtidos revelam que os modelos de DL podem ser utilizados para prever a curto termo de forma eficaz, destacando-se o modelo LSTM por ter apresentado o melhor desempenho.

2.5.2 Colocação dos postos de carga

A colocação dos postos de carga compreende a previsão das localizações adequadas para a colocação destes postos de carga e o tipo de carregadores a colocar, considerando fatores como a procura existente na área, a conveniência por parte dos utilizadores e ainda o apelo geográfico da localização.

Num estudo conduzido por Pevec et al. (2018), procurou-se resolver este problema com a utilização de ML. A colocação ótima de um posto de carga depende da entidade que o avalia. Neste estudo, realizaram-se duas abordagens diferentes: numa primeira abordagem, foi dada relevância ao dono do posto de carga e, consequentemente, ao lucro obtido. Numa segunda abordagem, consideraram-se os utilizadores de VE e as autarquias locais, que defendem que a colocação dos postos de carga deve ser realizada em localizações menos abrangidas, diminuindo a ansiedade de alcance. Recorreu-se a dados históricos sobre transações de carga de VE e informação geográfica constituída por localização de pontos de interesse e distância de condução entre os postos de carga.

Assim, os modelos de ML foram treinados nos dados de treino e foram utilizadas três funções objetivo para determinar os melhores locais para colocação dos postos de carga. A primeira função objetivo tinha como intuito maximizar a primeira abordagem; a segunda função objetivo pretendia maximizar a segunda abordagem; a terceira função objetivo tentava agrupar as duas abordagens numa só, e encontrar as melhores localizações que satisfaziam ambos os requisitos.

2.5.3 Cálculo da utilização de postos de carga

O cálculo da utilização de postos de carga consiste em calcular quantas vezes um posto de carga é utilizado e calcular estatísticas relevantes, como o tempo para além do necessário que os VE utilizam para carregar.

Uma abordagem para calcular o *idle time* de VE, (o tempo que um VE passa conectado ao posto de carga mesmo depois da bateria chegar aos 100%), é descrita em (Lucas et al., 2019). Neste estudo, foram utilizados modelos de ML de regressão, como o RF, *Gradient Boosting* (GB) e *Extreme Gradient Boosting* (XGBoost), e como métricas de performance foram utilizadas as RMSE e MAE. De entre os modelos

avaliados, o XGBoost foi o que apresentou melhores resultados. Com este estudo, verificou-se que a hora do dia em que o carregamento era efetuado é o parâmetro com maior influência no resultado, seguido da percentagem de bateria recarregada. Esta informação, apesar de aparentemente não ser relevante, permite uma melhor gestão da rede de carregamentos, e permite a um utilizador em espera para carregar o veículo avaliar se compensa esperar para que o carregador fique disponível.

2.5.4 Agendamento de postos de carga

O agendamento de postos de carga compreende a gestão dos postos de carga com base na demanda destes, de forma a que não se sobrecarregue a rede elétrica e que não se desperdice energia.

Um estudo realizado por Dang et al. (2019) utiliza uma abordagem baseada em *Q-Learning*, um algoritmo de *reinforcement learning*. O modelo desenvolvido neste estudo revelou-se capaz de resolver o problema apresentado. No entanto, o acréscimo da tabela Q provoca uma utilização intensiva dos recursos computacionais.

2.6 Síntese

O desenvolvimento da infraestrutura de carregamentos para VE em Portugal tem progredido de forma positiva, mas ainda se encontra longe dos valores recomendados pela União Europeia. O aprimoramento de modelos de ML e DL, nos últimos anos, tem aberto as portas a um número de problemas que estes conseguem resolver, nomeadamente modelos como LSTM, que são especializados na resolução de séries temporais, como se verifica no *dataset* a analisar nesta dissertação.

A importância do correto desenvolvimento da infraestrutura de carregamento, provocou a necessidade da construção de planos que assistam na implementação de novos postos de carga. Por este motivo, têm sido realizados diversos estudos em cidades onde se procurou implementar modelos de ML e DL para resolver uma de quatro vertentes que constituem o problema de desenvolver corretamente a infraestrutura. A grande maioria dos estudos realizados apresentaram resultados promissores quando aplicados a casos de estudo mais específicos que o proposto nesta dissertação, demonstrando contudo, uma perspetiva positiva para o desenvolvimento deste projeto.

Capítulo 3

Coleção, exploração e tratamento dos dados

3.1 Coleção dos dados

Para a construção do *dataset* de estudo, foram concretizadas cinco etapas evolutivas:

- Obtenção de informação referente aos EVSE em Portugal.
- Transformação dos dados de utilização de EVSE.
- Extração dos pontos de interesse por localização.
- Recolha de informação meteorológica.
- Junção de todos os dados obtidos.

3.1.1 Obtenção de informação técnica dos EVSE

Numa primeira fase, procurou-se obter informação técnica referente a todos os EVSE ativos em Portugal.

Os dados obtidos contêm informação referente a um total de 2779 EVSE. Nestes dados, encontram-se detalhados os conectores de cada EVSE e a sua localização (Consideração 1).

Consideração 1: Exemplo de uma entrada nos dados.

```
dictionary =
{
    # Id de um dos conectores (>=1) do posto elétrico;
    "evse_id": PRT-XXXXX-YY,
    # Id do posto elétrico;
    "location_id": PRT-XXXXX,
    # Cidade em que se encontra localizado;
    "city": Porto,
    # Latitude referente à localização representada pela "location_id";
    "latitude": 41.157225,
```

```
    # Longitude referente à localização representada pela "location_id";
    "longitude": -8.625865,
    # Estilo de estacionamento onde o posto se encontra colocado;
    "parking_type": PARKING_LOT,
    # Tensão elétrica do conector (V);
    "max_voltage": 400,
    # Amperagem do conector (A);
    "max_amperage": 32,
    # Velocidade de carga do conector (W);
    "max_electric_power": 22000
}
```

3.1.2 Obtenção da informação de utilização dos EVSE

Para a realização da segunda etapa, foram recolhidos dados referentes à utilização de EVSE ao longo de Portugal entre as datas *5 de maio de 2022* até *9 de novembro de 2022*. Os dados possuem informação referente às mudanças de estado dos EVSE (Consideração 2).

Consideração 2: Exemplo de uma entrada na informação obtida.

```
# Id do evse;
location_id: PRT-XXXXX,
# Id de um dos conectores (>=1) do posto elétrico;
uid: PRT-XXXXX-YY,
# Estado para em que o conector se encontra;
status: Available
# Data e hora em que o conector reportou o estado;
timestamp: 2021-02-03 05:23:20
```

Os conectores apresentam um total de três estados diferentes: *AVAILABLE*, *CHARGING* e *OUTOFORDER*. Visto que esta informação por si não indica as sessões de carga, foi necessário desenvolvê-la recorrendo às mudanças de estado. Para cada conector e quando detetado o estado *CHARGING*, foi guardada a *timestamp* como início da sessão de carga. A deteção de um estado *AVAILABLE* ou *OUTOFORDER* correspondeu à marcação do fim da sessão de carga, com o respetivo *timestamp*.

3.1.3 Extração dos pontos de interesse

Durante a revisão de literatura, verificou-se que os pontos de interesse na circunvizinhança dos EVSE eram frequentemente utilizados como um parâmetro durante o treino dos algoritmos, uma vez que a maior quantidade de pontos de interesse (restaurantes, escolas, atrações turísticas, entre outros) que rodeiam uma certa localização fomenta o maior apelo da mesma, levando a um maior número de pessoas a visitar a área. Consequentemente, pode verificar-se um crescimento no número de VE na área.

Posto isto, para cada uma das 2779 localizações, recolheram-se as respetivas coordenadas no formato de latitude e longitude. Recorrendo à *Places API* da *Google*, adquiriram-se os pontos de interesse num raio de 500m para cada uma das localizações.

Os dados obtidos através da *Google* possuem um total de 97 etiquetas diferentes. De modo a evitar conflitos de dimensão, selecionaram-se sete etiquetas para posterior inserção no *dataset*: as cinco etiquetas com mais registos (*store, food, lodging, health, restaurant*), e duas etiquetas de entre as primeiras vinte que representassem um ponto de interesse (*school, tourist_attraction*).

3.1.4 Recolha de informação meteorológica

De forma a enriquecer os dados do *dataset*, procurou-se obter informação meteorológica para os dias dos que se encontram em registo, com o intuito de avaliar se o estado meteorológico afeta, de algum modo, a utilização média dos EVSE, e ainda, se nestes dias, os EVSE localizados em áreas com proteção da chuva apresentavam uma utilização mais elevadas nos respetivos dias.

Através da *API OpenWeatherMap*, é possível obter dados meteorológicos para um dado dia e coordenada. A partir da informação obtida na primeira etapa, foram extraídas as cidades presentes no *dataset* e, consequentemente, atribuídas as respetivas coordenadas. O valor para as coordenadas foi obtido através de um *dataset* auxiliar.

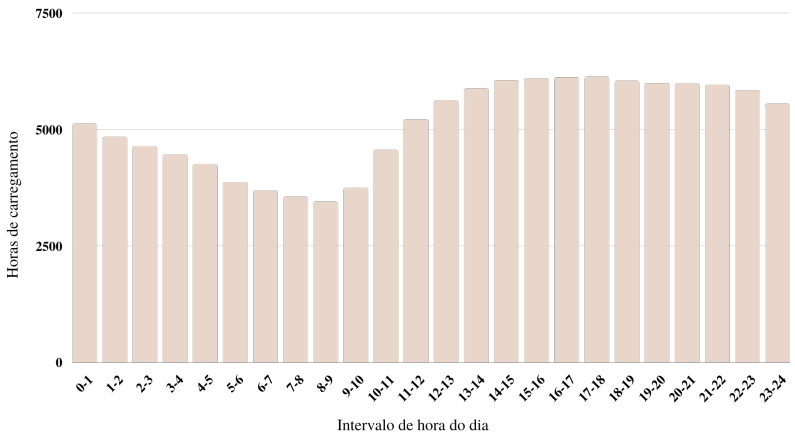
Os dados recebidos possuem informação relativa à meteorologia para cada hora do dia. De forma a evitar problemas de dimensão no *dataset* final, realizou-se um estudo sobre os dados, com o intuito de verificar as horas do dia com maior atividade de carregamentos (Figura 13).

De entre os intervalos de horas, foram selecionados os seis com uma maior utilização:

$$[14, 15[- [15, 16[- [16, 17[- [17, 18[- [18, 19[- [19, 20[.$$

De entre destes seis, foi escolhido o estado meteorológico mais frequente para ser inserido no *dataset*.

Figura 13: Uso coletivo das estações de carga em horas, por intervalos de hora por dia.



3.1.5 Junção dos dados

Na última etapa, foi realizada a junção de todos os dados anteriormente obtidos, de modo a conceber o *dataset* final.

Nas etapas anteriores, os dados encontravam-se separados por conector. Nesta etapa, todos os dados foram agrupados pela identificação da estação em vez do conector. Para cada conector de uma estação, foram contabilizadas as horas totais de cada sessão de carga que estes apresentaram durante o dia, e realizado um somatório para conhecer as horas totais que o correspondente EVSE esteve em uso. A esta variável atribui-se o nome de *dailyHours*, correspondendo ao *target* a medir pelos modelos a serem construídos.

3.2 Exploração dos dados

Foi conduzido um estudo sobre o *dataset* previamente obtido, de modo a observar as tendências por este apresentadas.

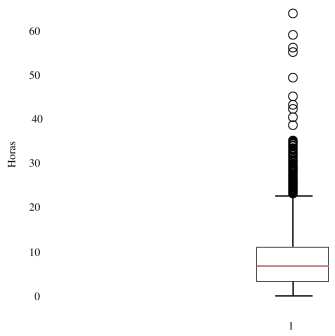
3.2.1 Remoção de outliers

Como primeiro passo, foram removidos todos os *outliers*. Estes valores foram removidos pela existência de alguns conectores defeituosos que apresentam uma taxa média de utilização de 24 horas por dia, que podiam influenciar a análise dos dados e o treino dos algoritmos. O método utilizado para a remoção dos *outliers* foi a amplitude interquartil (Figura 14). Com a remoção dos *outliers*, o *dataset* ficou composto por 476605 entradas e um total de 22 variáveis (estrutura disponível no Apêndice A - Consideração 7).

De um total de 2779 postos, 45 foram classificados postos como *outliers* e todos os seus registos foram removidos.

Depois de removidos os *outliers*, uma breve análise do *dataset* revelou: **2734** estações de carga com um total de **5815** conectores, através de **308** cidades, dentro de **20** regiões com um total de **7** tipos de estacionamento diferentes.

Figura 14: Diagrama de caixa da amplitude interquartil da média de horas dos postos.



3.2.2 Distribuição da potência elétrica

Procurou-se avaliar as tendências referentes à potência elétrica dos conectores e responder às seguintes questões:

- Quantos valores de potência diferentes existem?
- Qual é a distribuição destes valores?
- Qual é a utilização média por dia, para cada um dos diferentes valores de potência?
- Quantas sessões médias por dia os conectores com uma certa potência possuem?

Na Figura 15, são visíveis os valores de potência de cada conector (em watts) e respetiva distribuição. Com a análise deste gráfico, verifica-se a existência de uma discrepância significativa na distribuição dos conectores, em que 41% corresponde a conectores de 22 kilowatts e 22% a conectores de 50 kilowatts. Os restantes 33% estão distribuídos pelos ademais 21 níveis de potência de conectores.

Por sua vez, na Figura 16, verifica-se a utilização média, por dia, em percentagem, dos conectores para cada um dos diferentes níveis de potência (em watts). Prevê-se que a utilização média de um conector é tendencialmente inferior com o aumento da potência do respetivo conector. Esta previsão é confirmada pelo gráfico da figura apresentada.

Figura 15: Número de conectores em cada grupo de potência.

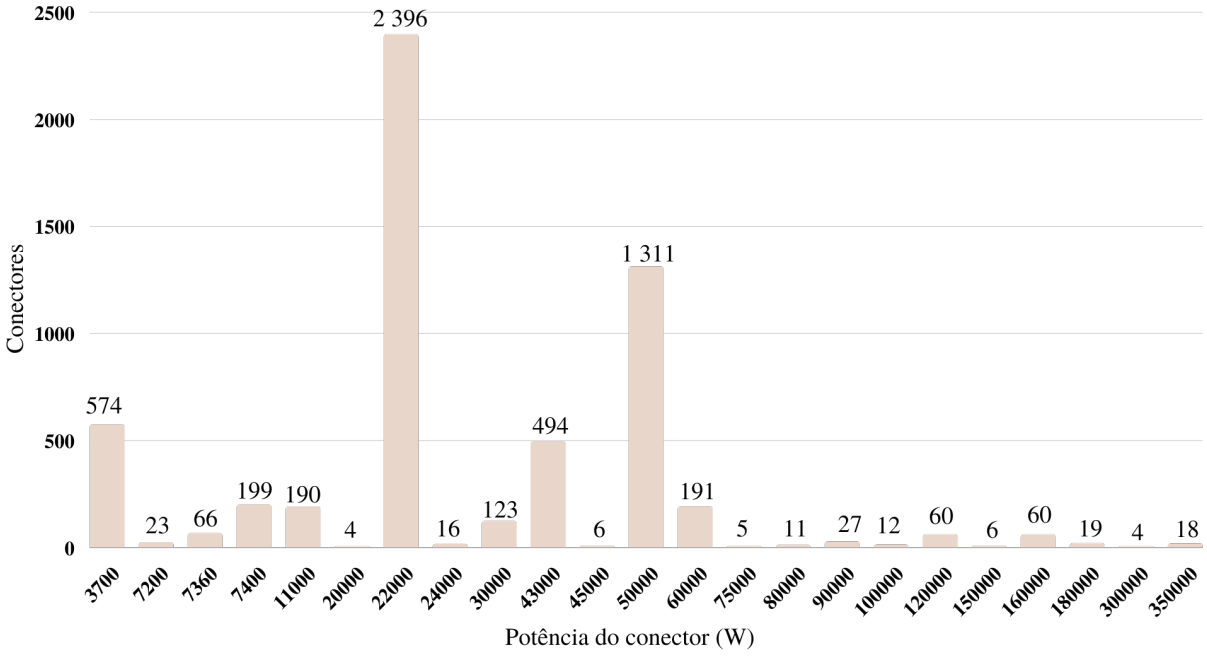
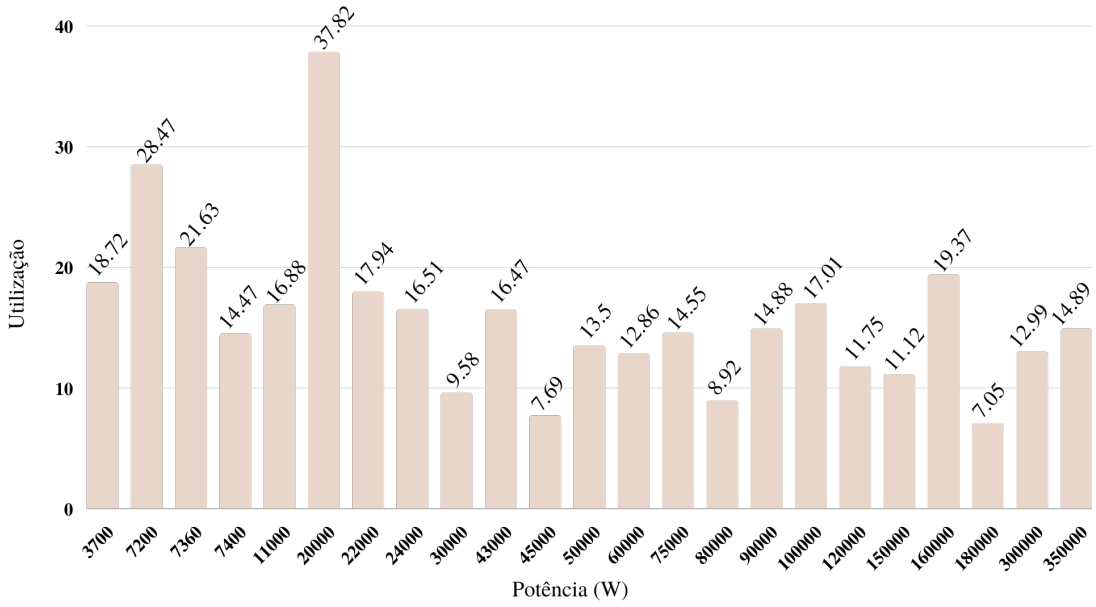


Figura 16: Utilização média dos conectores por dia. Y (utilização) está representado em percentagem total do dia.



Na Figura 17, observa-se o número médio de sessões por dia. Nesta figura, verifica-se que os conectores com um maior número de sessões diárias são conectores de *rapid charging*, frequentemente encontrados em lugares como autoestradas.

A breve análise destes três gráficos revela de imediato um possível contratempo. Visto que na presente dissertação se pretende a criação de um modelo que indique a configuração para um posto que obtém maior utilização, o modelo final vai possuir uma certa tendência para priorizar os conectores com menor potência.

Figura 17: Número médio de sessões diárias que cada grupo de conectores apresenta.

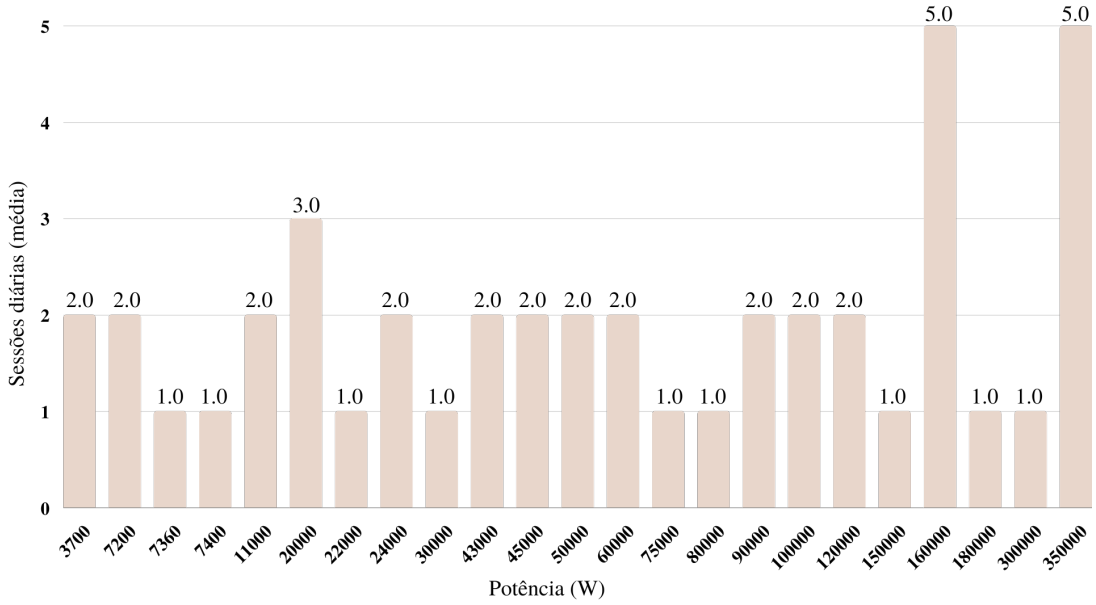


Figura 18: Número médio de pontos de interesse por posto de carga.

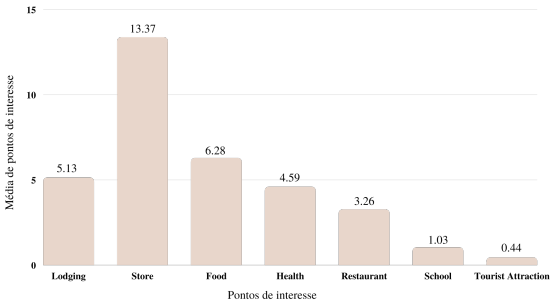
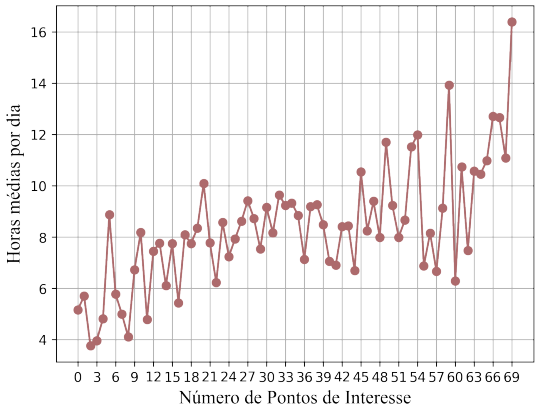


Figura 19: Média de horas diárias por número de pontos de interesse.



3.2.3 Influência dos pontos de interesse

Procedeu-se à verificação da distribuição dos pontos de interesse pelas localizações presentes no *dataset*. Na Figura 18, encontram-se representados os valores médios por localização para cada uma das categorias. A análise do gráfico revela que, tal como expectável, as primeiras cinco categorias apresentam uma presença muito mais notável por serem as cinco etiquetas mais frequentes obtidas na subsecção 3.1.3. Por sua vez, as últimas duas etiquetas apresentam uma presença significativamente mais reduzida.

De seguida, procurou-se verificar se o número de pontos de interesse que rodeavam um posto de carga, tem qualquer influência na média de horas diárias de um posto de carga (Figura 19). Da análise do gráfico, conclui-se que o aumento dos pontos de interesse, apesar de ser volátil, provoca um crescimento no número médio de horas diárias. Este comportamento encontra-se dentro do esperado, no entanto, pode não se dever integralmente ao aumento dos postos de interesse numa dada localização. Visto que as localizações ricas em pontos de interesse são grandes cidades, o valor da média de horas diárias pode dever-se à existência de uma quantidade mais notável de postos de carga de baixa potência.

3.2.4 Influência da meteorologia

Procurou-se averiguar os valores apresentados pela variável *weather* e, além disso, avaliar a influência que esta variável possui nas horas de carregamento (Figura 20), tal como a distribuição dos estados meteorológicos no *dataset* (Figura 21).

Os resultados obtidos demonstram que, regra geral, os dias mais limpos apresentam uma maior utilização dos postos de carga. No entanto, existe um *outlier* nos dados, *drizzle*, que apresenta um valor anormal comparativamente às restantes (12.41 horas).

Figura 20: Média de horas diárias por estado meteorológico.

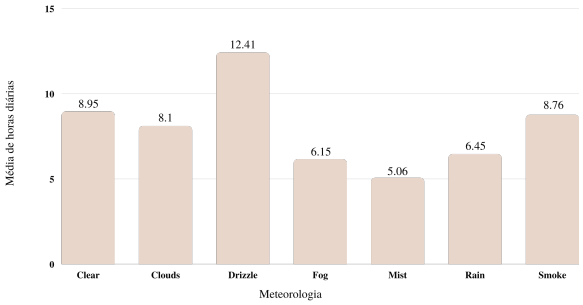
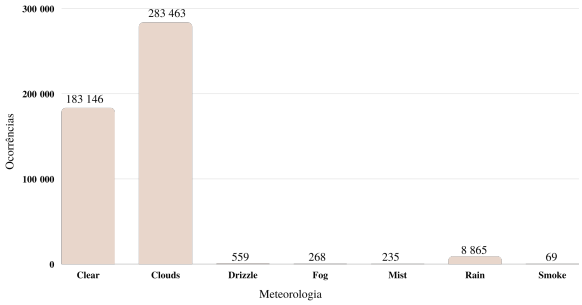


Figura 21: Distribuição dos estados meteorológicos.



3.2.5 Distribuição das estações

Dado que o modelo final vai procurar fornecer a configuração de uma estação, procurou-se conhecer a frequência do número de conectores, de modo a verificar qual o valor mais frequente (Figura 23). No que diz respeito à distribuição geral de todas as estações, estas seguem o padrão esperado, ou seja, encontram-se maioritariamente na zona costeira, com grandes concentrações em Lisboa, Porto e Faro (Figura 22).

Com a análise da Figura 23, verifica-se que 97% das estações de carga possuem entre um a três conectores, e que as estações pertencentes aos restantes 3% se encontram maioritariamente nas grandes cidades. Posteriormente, foram mapeadas as coordenadas de cada uma das estações, de modo a observar a distribuição das estações e a sua distribuição pelo número de conectores (Apêndice B).

Figura 22: Distribuição do número estações por distrito.

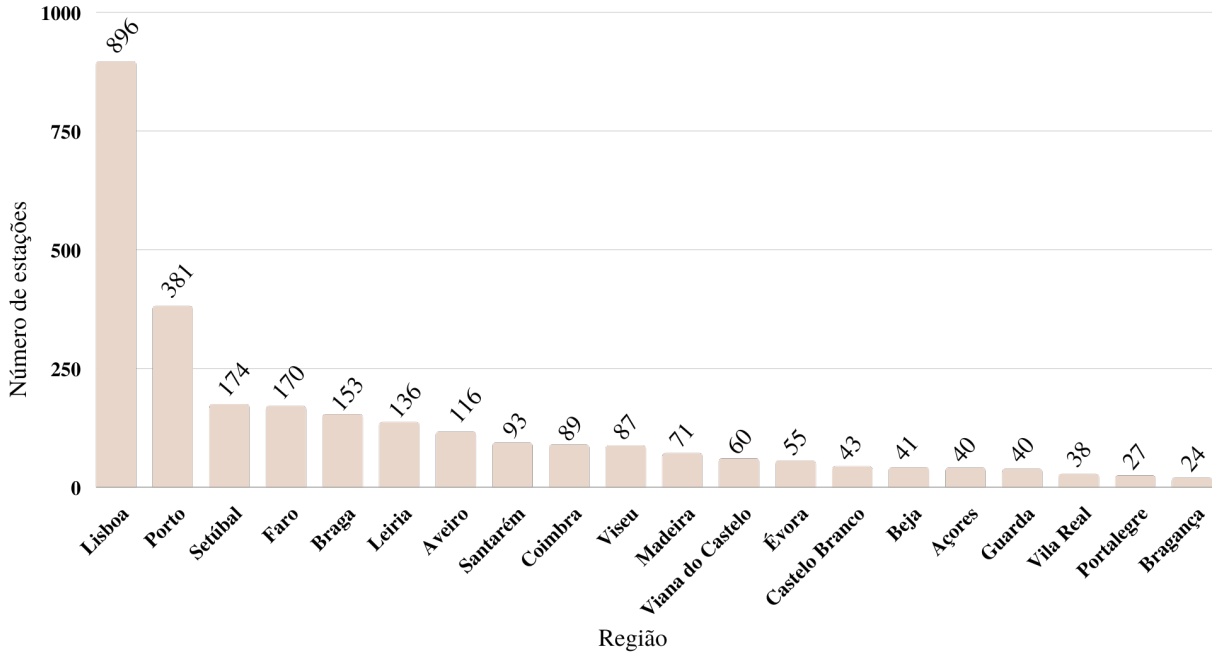


Figura 23: Distribuição do número de conectores por estações.

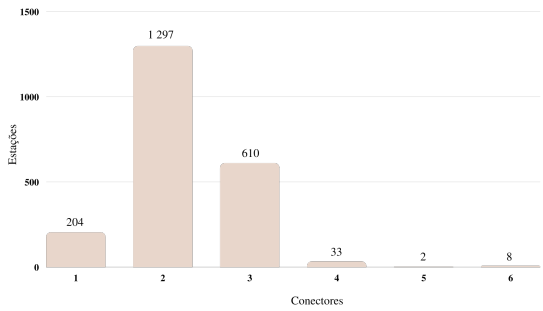
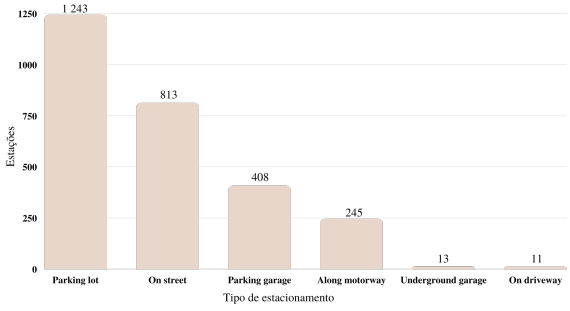


Figura 24: Distribuição do número de estações por tipo de estacionamento.



3.3 Pré-processamento do dataset

Para que o modelo possa ser treinado com os dados recolhidos, é fulcral que estes passem por uma *pipeline* de transformações para converter os dados num formato compreensível pelos modelos de ML.

3.3.1 Tratamento da latitude e longitude

Para o tratamento da latitude e longitude do *dataset*, foi aplicado o algoritmo de ML *K-Means clustering*. Quando a um algoritmo *K-Means* é fornecido um *dataset*, este algoritmo agrupa as entradas em **k** grupos (*clusters*), pelo que as entradas pertencentes aos mesmos grupos são similares.

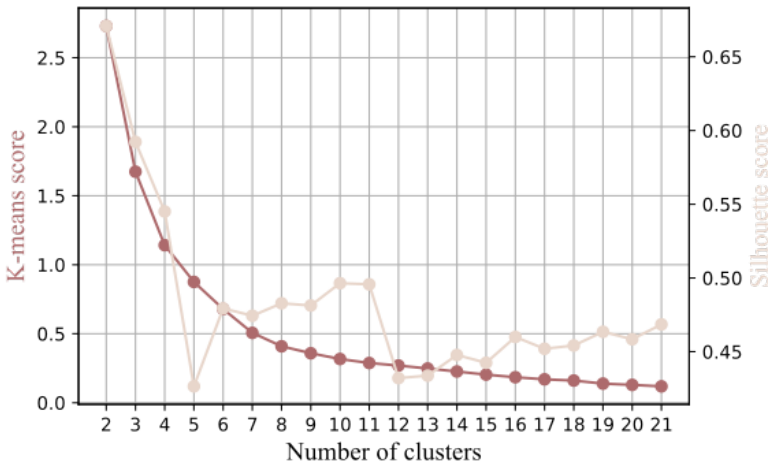
Dado que o modelo a ser desenvolvido receberá dados provenientes de todo o território português, foram criados 20 modelos *K-Means* (um para cada distrito, um para os Açores e outro para a Madeira). Estes modelos foram treinados na latitude/longitude de todos os postos pertencentes ao distrito/ilha que lhes corresponde. Quando fornecido um novo par latitude/longitude, o modelo retorna o grupo ao qual esse par pertence.

Para cada um dos modelos, foi necessário conhecer o valor ideal de **k**. Para este fim, recorreu-se a dois métodos: o método do cotovelo, que consiste em encontrar o ponto onde a linha que representa a pontuação atribuída pelo *K-Means* começa a ficar plana; e o método de pontuação de silhueta. Este último atribui um valor entre [-1,1], em que -1 significa que os pontos estão atribuídos de forma incorreta, 0 indica que existem grupos que se sobrepõem, e 1 indica que os pontos estão corretamente atribuídos e que os grupos são facilmente distinguíveis.

O objetivo tornou-se assim conseguir encontrar um equilíbrio entre estes dois métodos para determinar o melhor valor de **k**.

Como se pode observar na Figura 25, o ponto onde a curva do *K-means score* começa a ficar plana é ambíguo, pelo que se recorreu ao *Silhoutte score* para assistir na determinação do melhor valor para **k**. No caso de estudo, o valor de **k** selecionado foi 11.

Figura 25: Gráfico gerado para avaliar o valor de **k** para o distrito do Porto.



3.3.2 Tratamento de atributos temporais

O *dataset* possui de um atributo **data** que indica o dia de um dado registo. Uma vez que os dados utilizados são referentes a seis meses de registos, optou-se pela utilização da característica cíclica dos dias da semana para representar o atributo através das relações seno e cosseno (Consideração 3).

Consideração 3: Transformação do atributo data.

```
date = "2022-07-18"

  # Monday (0)
day = date.weekday() + 1

weekday_sin = np.sin(2 * np.pi * weekday/7)
weekday_cos = np.cos(2 * np.pi * weekday/7)
```

3.3.3 Tratamento dos arrays

O *dataset* construído para este estudo possui atributos em forma de array, necessitando de uma reestruturação. Estes atributos são:

- connectors_usage;
- connectors_amperage;
- connectors_voltage;
- connectors_electric_power.

De entre os quatro atributos supra mencionados, apenas se considerou o **connectors_electric_power**, tendo-se descartando os restantes.

Nos atributos **connectors_amperage** e **connectors_voltage**, a informação obtida é utilizada na área de previsão da carga na rede elétrica por parte do posto. No entanto, na presente dissertação, o modelo a ser construído pretende apenas prever a utilização dos postos.

Por sua vez, o atributo **connectors_usage**, apesar de possuir informação útil, induz o modelo em erro, uma vez que prioriza conectores de baixa potência, dado que estes demoram mais tempo a carregar, quando em comparação com os conectores de alta potência.

Assim sendo, para o *array* **connectors_electric_power**, foram consideradas duas abordagens no tratamento dos dados, sendo cada uma guardada num ficheiro diferente, de modo a serem comparadas aquando do treino do modelo.

A primeira abordagem consistiu em conhecer o maior número de conectores que um posto possui e adicionar uma coluna por cada conector ao *dataset*. Estas colunas foram posteriormente preenchidas pela potência dos conectores de cada posto por ordem crescente. Na ausência de um certo conector, o valor atribuído foi de zero (Consideração 4). Uma vez que se tratam de dados categóricos, foi aplicado um *label encoder*.

Consideração 4: Transformação do array da potência dos conectores.

```
# arrays connector_electric_power
posto1 = [25, 25, 25, 25]
posto2 = [150,350]

maxConnectors = 4

[...]

#entradas no dataset
connector1 || connector2 || connector3 || connector4
    25            25            25            25      #posto1
    150           350           0             0       #posto2
```

Por outro lado, a segunda abordagem teve como objetivo aplicar um *One Hot Encoding* a cada elemento do array e, de seguida, efetuar a soma dos vetores resultantes para colocar na entrada daquele posto no *dataset* (Consideração 5).

Consideração 5: Transformação do array da potência dos conectores.

```
# arrays connector_electric_power
posto1 = [25, 25, 25, 25]
posto2 = [150,350]

[...]
#One hot encoding e soma dos vetores
[...]

#entradas no dataset (correspondem aos valores de potência existentes no
   dataset)
P25 || P150 || P350
 4       0       0    #posto1
 0       1       1    #posto2
```

Uma vez que a primeira abordagem apresentou melhores resultados nos testes iniciais, optou-se pela sua utilização no modelo final.

3.3.4 Tratamento dos dados categóricos

Para o treino, serão considerados três atributos categóricos do *dataset*: *parking_type*, *weather* e *district*.

Sobre estes atributos, foram aplicadas as duas abordagens supra mencionadas, tendo-se selecionado a primeira abordagem para utilização no modelo final.

3.3.5 Separação entre treino e teste

Devido à natureza do problema em si, a separação dos dados entre treino e teste não pôde ser realizada de modo aleatório, uma vez que o objetivo do modelo é prever a utilização de postos não observados, pelo que esta divisão deve ser realizada entre os *locationId* únicos presentes no *dataset*. Recorreu-se a uma divisão de 80% para treino e 20% para teste (Consideração 6).

Consideração 6: Separação entre treino e teste.

```
import pandas as pd
import numpy as np

uniqueIds = pd.DataFrame( dataset['locationId'].unique() , columns=['ID'] )

np.random.seed(100)

uniqueIds['rand'] = (np.random.randint(0, 10000, uniqueIds.shape[0]))/10000

uniqueIds['GROUP'] = np.where(((uniqueIds.rand <= 0.8)),'TRAIN','TEST')
```

3.3.6 Escalonamento dos atributos

Por fim, consideraram-se duas abordagens para o escalonamento dos dados: *Standard Scaler* e *Min Max Scaler*. Assim sendo, realizaram-se testes com dados sem escalonamento, dados com o *Min Max Scaler* aplicado e dados com o *Standard Scaler* aplicado.

Dado que os melhores resultados obtidos foram os dados escalonados com o *Min Max Scaler*, considerou-se esta abordagem para o *dataset* final.

Capítulo 4

Desenvolvimento dos modelos

Neste capítulo, será exposto o processo de desenvolvimento do modelo de recomendação, desde o pré-processamento dos dados até à construção do modelo final, e descritos os resultados obtidos com os diversos modelos. Todos os testes nesta secção foram treinados com recurso a um computador com as seguintes especificações: CPU = Intel i7-6700k, GPU = GeForce GTX 1070, Ram = 16 GB.

4.1 Construção do modelo de previsão

A criação de um modelo capaz de recomendar configurações de postos elétricos que apresentam a maior utilização, em horas por dia, requer a criação de um modelo capaz de prever a utilização de uma estação dada a sua configuração. Como o valor a prever é um número (horas diárias de uso), trata-se de um problema de regressão.

4.1.1 Seleção dos algoritmos de ML

Inicialmente, previa-se a utilização de um algoritmo de DL. No entanto, estes algoritmos não revelam um bom desempenho em *datasets* de pequena escala (como o *dataset* utilizado neste estudo), pelo que se recorreu a algoritmos tradicionais de ML. Assim sendo, testaram-se seis algoritmos de regressão.

- *Support Vector Regression* (SVR)
- *eXtreme Gradient Boosting* (XGBoost)
- *Random Forest Regressor* (RFR)
- *Linear Regression* (LR)
- *Ridge Model* (RM)
- *Lasso Model* (LM)

Selecionaram-se estes seis algoritmos de acordo com a recomendação dada por Thakur (2020). Os algoritmos RM e LM, não foram abordados na revisão de literatura, são derivados da LR.

Estes algoritmos diferenciam-se do algoritmo de LR na adição de uma penalização na função de custo, que calcula o peso dos coeficientes descritos na subsecção 2.3.4.1. Esta penalização é distinta entre os diferentes algoritmos. No algoritmo RM, a penalização é baseada no quadrado dos coeficientes (Equação 14), tornando-os mais pequenos e reduzindo a complexidade do modelo. Por outro lado, o algoritmo LM adiciona uma penalização focada no valor absoluto do coeficiente (Equação 15), o que pode originar coeficientes com valor a 0, pelo que o algoritmo LM pode vir a ignorar variáveis do *dataset*. Ambos os modelos possuem λ como parâmetro de regularização da penalidade. Quanto maior a aproximação de λ de zero, maior a aproximação a um algoritmo de LR. Estas alterações visam reduzir a complexidade geral do modelo e respetivo *overfitting*, quando em comparação com LR (Marquardt and Snee, 1975; Ranstam and Cook, 2018).

$$Penalizacao\, RM = \lambda \sum_{i_0}^{n} b_i^2 \tag{14}$$

$$Penalizacao\, LM = \lambda \sum_{i_0}^{n} |b_i| \tag{15}$$

4.1.2 Testes iniciais

Na análise do *dataset*, nomeadamente nas subsecções 3.2.3 e 3.2.4, verificou-se quais as variáveis que aparentavam influenciar a utilização dos postos de carga. No entanto, para apurar a veracidade das conclusões obtidas, foi conduzido um teste sobre o *dataset*. Para este fim, foram treinados os seis algoritmos previamente expostos com os hiperparâmetros padrão para quatro *datasets* diferentes: um *dataset* sem informação referente aos pontos de interesse e meteorologia, outro *dataset* com informação sobre os pontos de interesse, um com informação sobre a meteorologia e, por fim, um *dataset* com toda a informação incluída. Todos os *datasets* utilizados apresentavam exclusivamente informação referente ao distrito de Braga (Tabela 2).

Os resultados revelam que, para todos os algoritmos, a inserção da meteorologia ou de pontos de interesse melhorou ligeiramente o desempenho. Além disso, de entre os *datasets* avaliados, aquele que continha toda a informação foi o que apresentou melhor desempenho em todos os algoritmos, validando a análise previamente realizada, e demonstrando que esta informação é útil para o *dataset*.

Tabela 2: Testes iniciais feito com os *datasets*.

Model	Dataset							
	Básico		Meteorologia		Pontos Interesse		Completo	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SVR	10.2401	6.5089	10.2225	6.4973	10.1438	6.3913	10.1212	6.3809
XGBoost	9.3193	6.7415	9.2859	6.7279	9.2242	6.6946	9.2137	6.6890
RFR	9.3193	6.7415	9.2859	6.7279	9.2242	6.6946	9.2216	6.6470
LR	9.6838	7.3277	9.6844	7.3238	9.4863	7.1262	9.4843	7.1177
RM	9.6838	7.3277	9.6844	7.3238	9.4863	7.1262	9.4843	7.1177
LM	9.4981	7.3062	9.4875	7.2885	9.4178	7.1467	9.4074	7.1313

4.1.3 Escolha preliminar e explicação dos hiperparâmetros

Selecionado o *dataset* a utilizar, foi necessário avaliar os algoritmos que apresentam o melhor desempenho para o problema em estudo. Assim sendo, para cada um destes modelos, foi aplicado o *Grid Search Cross-Validation* (GSCV). O GSCV é uma técnica que, ao receber um algoritmo de ML e uma lista de hiperparâmetros, efetua uma pesquisa exaustiva de todas as possíveis combinações de hiperparâmetros, de modo a encontrar a combinação que gera o modelo com a melhor performance (calculada através de *cross-validation* (Adnan et al., 2022)).

Antes de se aplicar o GSCV foi necessário compor uma lista de parâmetros a otimizar para todos os modelos (Tabelas 3, 4, 5). Em seguida, será brevemente descrito cada hiperparâmetro e respetivo propósito. Dar-se-á posterior descrição sobre os hiperparâmetros empregues nos algoritmos de RFR e XGBoost na secção que se segue.

Tabela 3: Hiperparâmetros utilizados no GSCV dos quatro modelos listados. Parâmetros recomendados por Thakur (2020).

Modelo	Hiperparâmetro	Gama de valores
SVR	C	[0.001, 0.1, 1, 10, 50, 100]
	gamma	['scale', 'auto']
LR	fit_intercept	['True', 'False']
RM	fit_intercept	['True', 'False']
	alpha	[1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100]
LM	fit_intercept	['True', 'False']
	alpha	[1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100]

Tabela 4: Hiperparâmetros utilizados no GSCV de XGBoost. Parâmetros recomendados por Thakur (2020).

Hiperparâmetros	Gama de valores
eta	[0.01, 0.015, 0.025, 0.05, 0.1]
gamma	[0.1, 0.5, 0.7, 1.0]
max_depth	[3, 5, 7, 9, 12, 15, 17, 25]

Como parâmetros para *Support Vector Regression* existem:

- **C**: Corresponde ao parâmetro de regularização das SVR. Este parâmetro permite indicar ao algoritmo o valor da penalização a atribuir aos pontos do *dataset* que se encontram dentro da margem (Figura 3). Quanto maior o valor de C, maior é a penalização (Scikit-learn, 2023e).

- **gamma**: Com o aumento deste parâmetro, mais próximos os pontos do *dataset* devem de estar, no hiperplano construído pelo *kernel*, para serem classificados com o mesmo *target*. Um valor baixo do parâmetro pode levar a *underfitting*, no entanto, um valor alto pode levar a *overfitting* (Scikit-learn, 2023e).

Os algoritmos de *Linear Regression*, *Ridge Model* e *Lasso Model* partilham alguns hiperparâmetros. São estes:

- **fit_intercept**: Este parâmetro indica se a linha de regressão (Figura 2) pode ou não intercetar o Y. Quando este parâmetro tem o valor *False*, o início da linha de regressão será em (0,0). No entanto, quando *True*, o início pode começar em qualquer valor de Y, sendo habitualmente aquele que melhor se adequa ao problema (Scikit-learn, 2023b; Scikit-learn, 2023d; Scikit-learn, 2023a).

- **alpha**: Corresponde ao parâmetro de regularização da penalização, tanto para RM e LM. Ou seja, este parâmetro é equivalente ao $lambda$ mencionado na subsecção 4.1.1 (Scikit-learn, 2023d; Scikit-learn, 2023a).

Tabela 5: Hiperparâmetros utilizados no GSCV de RFR. Parâmetros recomendados por Thakur (2020).

Hiperparâmetro	Gama de valores
n_estimators	[120, 300, 500, 800, 1200]
max_depth	[5, 8, 15, 25, 30, None]
min_samples_split	[1, 2, 5, 10, 15, 100]
min_samples_leaf	[1, 2, 5, 10]
max_features	['log2', 'sqrt', 'None']

4.1.4 Comparação preliminar dos algoritmos de ML

A Tabela 6 apresenta os resultados obtidos do desempenho dos modelos posteriormente à aplicação do GSCV. Para a obtenção destes resultados, foi utilizado um *dataset* mais restrito, por questões de limitações computacionais, que continha apenas estações de carga pertencentes ao distrito de Braga. Foi reduzido aquele *dataset* que continha os dados tratados com um *label encoder* e *Min Max Scaler*. Os restantes *datasets* apresentavam um *Standar Scaler* e/ou *One Hot Encoding* com resultados mais pobres, com um acréscimo no erro MAE num intervalo de $[1, 2]$ horas.

Apesar dos modelos SVR e XGBoost terem revelado melhores resultados, foram considerados os modelos XGBoost e RFR para receber um GSCV mais completo, por motivos de limitações computacionais, dado que o algoritmo SVR é computacionalmente mais pesado e o tempo necessário para treinar e otimizar é superior comparativamente aos restantes algoritmos.

4.1.5 Otimização dos melhores algoritmos

Depois de se terem averiguado os algoritmos de ML que apresentavam uma melhor capacidade para resolver o problema em mão, foi necessário conceber uma lista de hiperparâmetros mais completa de modo a obter o melhor desempenho possível dos algoritmos. Assim sendo, foi realizada uma pesquisa de hiperparâmetros para os algoritmos de XGBoost e RFR (Tabelas 7 e 8, respetivamente).

No caso do algoritmo de *eXtreme Gradient Boosting*, os parâmetros utilizados têm como propósito:

- **eta**: (ou *learning rate*) tem como propósito prevenir o *overfitting*. Esta prevenção é conseguida pela utilização do *eta* para reduzir o peso das variáveis, no final de cada interação do algoritmo (Dmlc, 2023).

- **gamma**: Um dos vários parâmetros que contribui na regularização da criação de novos nodos folha nas árvores do algoritmo. Especificamente, o *gamma*, permite indicar ao algoritmo que a criação de um novo nodo folha é vantajoso exclusivamente se a redução da perda originada dessa divisão for maior do que o valor do *gamma* (Dmlc, 2023).

- **max_depth**: Corresponde à profundidade máxima de cada árvore dentro do algoritmo (Dmlc, 2023).

Tabela 6: Resultados dos modelos posteriormente à aplicação do GSCV.

Modelo	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Error (MAE)
SVR	26.6018	5.1557	3.9558
XGBoost	26.3669	5.1348	3.7068
RFR	43.2484	6.5763	4.4383
LR	86.8755	9.3207	5.6945
RM	86.8655	9.3201	5.6864
LM	83.8840	9.1588	5.6435

Tabela 7: Hiperparâmetros utilizados no GSCV completo de XGBoost. Parâmetros recomendados por Thakur (2020).

Hiperparâmetro	Gama de valores
eta	[0.01,0.015,0.025,0.05,0.1]
gamma	[0.05,0.06,0.07,0.08,0.09,0.1,0.3,0.5,0.7,0.9,1.0]
max_depth	[3,5,7,9,12,15,17,25]
min_child_weight	[1,3,5,7]
subsample	[0.6,0.7,0.8,0.9,1.0]
colsample_bytree	[0.6,0.7,0.8,0.9,1.0]
lambda	[0.01,0.02,0.03,0.04,0.05,0.06,0.07,0.08,0.09,0.1,1]
alpha	[0,0.1,0.5,1.0]

- **min_child_weight**: Parâmetro que permite controlar o crescimento das árvores. Este parâmetro indica o número mínimo de instâncias do *dataset* que precisam de estar num nodo folha da árvore para ser permitida a sua divisão em dois novos nodos (Dmlc, 2023).

- **subsample**: A cada iteração do algoritmo, o algoritmo seleciona uma amostra do *dataset* de treino, equivalente ao valor deste parâmetro em percentagem, para ser utilizada para treinar as árvores naquela iteração. Desta forma, a cada iteração e para um valor *subsample* de 0.6, o algoritmo seleciona uma amostragem aleatória equivalente a 60% do *dataset* de treino (Dmlc, 2023).

- **colsample_bytree**: Este parâmetro funciona do mesmo modo que o parâmetro de *subsample*. Distingue-se por o valor ser aplicado às variáveis do *dataset* de treino e cada árvore calcula a sua amostragem a cada iteração (Dmlc, 2023).

- **lambda**: Parâmetro de regularização L2. Este parâmetro controla uma das penalizações colocadas nas folhas das árvores, baseada na penalização exposta na Equação 14 (Dmlc, 2023).

- **alpha**: Parâmetro de regularização L1. Este parâmetro controla uma das penalizações colocadas nas folhas das árvores, baseada na penalização exposta na Equação 15 (Dmlc, 2023).

Tabela 8: Hiperparâmetros utilizados no GSCV completo de RFR. Parâmetros recomendados por Thakur (2020).

Hiperparâmetros	**Gama de valores**
n_estimators	[120, 300, 500, 800, 1200]
max_depth	[5, 8, 15, 25, 30, None]
min_samples_split	[1, 2, 5, 10, 15, 100]
min_samples_leaf	[1, 2, 5, 10]
max_features	['log2', 'sqrt', None]

No que respeita ao *Random Forest Regressor*, têm-se os seguintes parâmetros:

- **n_estimators**: Representa o número de árvores que o algoritmo vai utilizar (Scikit-learn, 2023c).
- **max_depth**: Corresponde à profundidade máxima de cada árvore dentro do algoritmo (Scikit-learn, 2023c).
- **min_samples_split**: Parâmetro que controla o crescimento das árvores. Este parâmetro indica o número mínimo de instâncias do *dataset* que precisam de estar num nodo folha da árvore para ser permitida a sua divisão em dois novos nodos (Dmlc, 2023) (Scikit-learn, 2023c).
- **min_samples_leaf**: Parâmetro que controla o crescimento das árvores. Neste caso, o parâmetro indica o valor mínimo de instâncias do *dataset* que precisam de vir a estar num nodo folha hipotético antes da sua criação, isto é, antes do nodo que lhe vai dar origem se dividir (Scikit-learn, 2023c).
- **max_features**: Número máximo de variáveis que são utilizadas para treinar cada árvore. Por exemplo, se o valor é "sqrt", então o número máximo será sqrt(número_total_variáveis) (Scikit-learn, 2023c).

Posteriormente à recolha de lista de hiperparâmetros, realizou-se o GSCV para cada um dos algoritmos, com o *dataset* de treino na sua integra, obtendo-se os parâmetros listados na Tabela 9. Em seguida, foram treinados os algoritmos nestes parâmetros, e avaliados os resultados obtidos.

Tabela 9: Resultados do GSCV completo para cada um dos algoritmos.

Model	**Eta**	**Gamma**	**Max_Depth**	**Min_Child_Weight**	**Subsample**	**Colsample_Bytree**	**Lambda**	**Alpha**
XGBoost	0.01	1.0	9	1	0.8	0.8	1	0
	N_Estimators	**Max_Depth**	**Min_Samples_Split**		**Min_Samples_Leaf**		**Max_Features**	
RFR	500	None	2		10		sqrt	

4.2 Construção do modelo de recomendação

Uma vez que se pretende realizar uma recomendação de configurações capazes de obter uma utilização diária elevada, desenvolveu-se um modelo que, para dado *input*, obtém as configurações que apresentam o melhor tempo produtivo médio sob as situações avaliadas.

Como *input*, o modelo recebe:

- **Par latitude/longitude**: localização pretendida para a nova estação;
- **Tipo de estacionamento**: possíveis valores (expostos na figura 24);
- **Região**: define o modelo de *clustering* a utilizar nas coordenadas.

Com um dado *input*, o modelo obtém a informação dos pontos de interesse num raio de 500 metros das coordenadas fornecidas, através da *Places API* da *Google*. A restante informação necessária para criar uma entrada para o modelo de previsão é obtida (por diferentes métodos) e são-lhe aplicadas as transformações descritas na Secção 3.3.

Para gerar as configurações a testar, são utilizados os valores de potência expostos na Figura 15, e é realizada uma procura exaustiva para obter todas as combinações possíveis para postos com um, dois e três conectores. Atribuiu-se maior relevância a estes valores do número de conectores, uma vez que são os valores mais frequentes (Secção 3.2.5).

Para avaliar as configurações, executou-se o modelo de previsão para cada dia da semana, e para cada um dos diferentes estados meteorológicos, resultando num total de 49 previsões para cada configuração, sendo posteriormente calculada a média de todas as previsões. As configurações foram agrupadas em três grupos: as configurações que apresentam um conector; as configurações que apresentam dois conectores; e as configurações que apresentam três conectores. Cada grupo foi avaliado separadamente de modo a ser possível fornecer como *output* uma configuração ideal para cada valor de conectores.

Capítulo 5

Resultados e discussão

Os resultados obtidos pelos algoritmos XGBoost e RFR, posteriormente ao treino com os parâmetros da Tabela 9, encontram-se listados na Tabela 10. A análise dos resultados, obtidos revelam uma semelhança entre estes. Os dois modelos apresentam um MAE similar, indicando que a média do erro absoluto entre os dois modelos é a mesma. Por outro lado, o valor do RMSE para o modelo XGBoost é superior em relação ao modelo RFR. Estes valores indicam que, regra geral, os erros dados pelo modelo XGBoost são maiores do que os erros do modelo RFR, visto que o RMSE é uma métrica que penaliza, de forma mais acentuada, os erros de maior magnitude.

Tabela 10: Resultados obtidos pelos modelos depois de aplicado os resultados do GSCV

Modelo	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Error (MAE)
XGBoost	92.3012	9.6074	6.6310
RFR	82.6820	9.0929	6.6220

Com base no desempenho na métricas de erro, o melhor modelo aparente ser o RFR. No entanto, optou-se por realizar uma comparação entre os tempos de execução de ambos os modelos (Tabela 11). Da análise da tabela, conclui-se que o modelo XGBoost é consideravelmente mais rápido na realização de previsões que o modelo RFR.

No contexto do modelo de recomendação desenvolvido nesta dissertação, o tempo de execução não é um problema, pelo que o modelo RFR foi determinado como sendo o melhor. No entanto, para um modelo de recomendação com maior alcance que o atualmente desenvolvido, ou para um modelo onde seja relevante considerar o tempo de execução, o modelo XGBoost é a melhor opção, uma vez que o tempo

Tabela 11: Comparação de tempos de execução

Modelo	Tempo de treino (s)	Tempo de previsão (s)
XGBoost	15.97	0.0019
RFR	123.84	0.031

de execução é significativamente mais reduzido, e os resultados das métricas não são muito díspares entre si.

Posto isto, os resultados obtidos demonstram um baixo desempenho. Porém encontram-se dentro do objetivo definido, que consistia em obter um modelo cujo MAE fosse inferior ao valor médio do *target* do *dataset* empregado (7.4331). Avaliando o modelo RFR, este apresentou um valor MAE de 6.6220. Considerando que o MAE possui as mesmas unidades do *target*, este valor indica que, para qualquer previsão realizada pelo modelo, poderá existir um erro de ±6.6 horas face ao valor real, por exemplo, para uma previsão que devolva o valor 12, regra geral, o valor real pode pertencer ao intervalo [5.4, 18.6]. Por conseguinte, selecionaram-se de forma aleatória dez estações do *dataset* de treino, de forma a averiguar o impacto que este valor poderá ter sobre os resultados finais. Sobre estas estações, calculou-se a média semanal apresentadas em registo. Estas estações foram então inseridas no modelo de previsão para analogia dos valores (Tabela 12).

Tabela 12: Comparação entre valores do dataset e valores previstos pelo modelo. As médias representam o número de horas diárias de utilização do posto.

Estação	**E_1**	**E_2**	**E_3**	**E_4**	**E_5**	**E_6**	**E_7**	**E_8**	**E_9**	**E_10**
Média real	25.12	0.96	2.63	10.67	8.52	17.25	3.49	0.23	14.77	2.20
Média prevista	19.43	0.94	7.54	12.71	3.87	15.07	3.86	0.32	9.46	2.69

A análise da tabela evidencia que os valores se encontram dentro do intervalo da taxa de erro, existindo valores com uma elevada disparidade do valor real, como nas Estações 1, 5 e 9. Não obstante, o modelo de previsão foi capaz de, num âmbito geral, preservar o ranking de utilização entre as diferentes estações, com exceção das Estações 3, 4 e 9, sendo que, as posições das Estações 4 e 9 encontram-se invertidas. A realização deste teste permitiu verificar que, apesar do erro ser consideravelmente elevado, o modelo é ainda capaz de comparar diferentes configurações e avaliá-las corretamente.

Analisados os modelos, decidiu-se explorar uma nova abordagem, que consistiu em dividir o *dataset* em vinte partições, cada um correspondente às estações pertencentes a um certo distrito ou arquipélago. Com esta divisão, pretendeu-se observar se modelos especializados para um certo distrito seriam capazes de atingir uma performance mais satisfatória em relação a um modelo geral. Dos modelos treinados, 11 obtiveram um MAE superior ao valor médio do *target*, e os restantes 9 obtiveram um valor de MAE inferior ao valor médio do *target*, não representando um diferença acentuada (Tabela 13). Considerando a diferença entre os valores do MAE e a média do *target*, destacam-se os modelos referentes às regiões do Porto, Aveiro, Lisboa e Setúbal. Com base na Figura 22, verifica-se que, de modo geral, as regiões que apresentam melhor desempenho correspondem as que possuem um maior número de estações de carga a estudar. De notar, no entanto, que a região de Aveiro não se aplica à conclusão retirada, sendo um *outlier*. Dado o pobre desempenho na maioria das regiões, descartou-se a atual abordagem. No entanto, num futuro em que as regiões com menor representação apresentem um número significativo de estações de carga, esta abordagem pode revelar-se mais benéfica do que um modelo que englobe a totalidade das regiões.

Tabela 13: Resultados dos testes realizados por região

Região	Valor médio do target	Mean Absolute Error (MAE)
Viana do Castelo	4.50	5.11
Braga	5.60	6.05
Porto	7.70	6.80
Vila Real	4.18	4.64
Bragança	3.97	5.85
Aveiro	7.50	6.19
Viseu	4.50	5.27
Guarda	3.64	4.84
Coimbra	5.40	6.02
Castelo Branco	5.37	4.88
Leiria	7.21	7.70
Santarém	6.73	6.43
Portalegre	3.52	4.40
Lisboa	8.54	6.97
Évora	4.26	5.87
Setúbal	9.09	7.60
Beja	4.55	4.36
Faro	6.68	6.30
Madeira	7.20	6.83
Açores	1.28	2.91

No desenvolvimento da atual dissertação, deparou-se com algumas dificuldades, como o desbalanceamento do *dataset* (nomeadamente nas variáveis referentes à potência dos conectores e à meteorologia). Este desbalanceamento provoca o desenvolvimento de tendências que afetam negativamente o treino do algoritmo. Não foram, no entanto, utilizados métodos para colmatar este desbalanceamento, graças à natureza do *dataset*. O desempenho dos modelos poderá melhorar com a remoção deste desbalanceamento. A principal limitação foi o tempo dispendido a otimizar os dados e os modelos, nomeadamente na otimização através de GSCV que, por vezes, demorava vários dias a completar, sendo que limitou o número de algoritmos a que se pode aplicar.

Com isto em mente, conclui-se que o modelo de previsão *Random Forest Regressor* é aquele que reuniu os melhores resultados num âmbito geral, tendo sido o algoritmo utilizado na construção do modelo de recomendação. Para este fim, incorporaram-se os *datasets* de treino e teste e, utilizando os melhores parâmetros para RFR da Tabela 9, o algoritmo foi treinado uma última vez, de modo a obter um modelo que incorpora todos os dados recolhidos.

Na Figura 29 (Apêndice C), evidencia-se um exemplo da utilização do modelo de recomendação desenvolvido. Nesta, constata-se uma adversidade previamente mencionada no capítulo 3 (subsecção 3.2.2), onde o modelo priorizou conectores de reduzida potência no seu ranking, por apresentarem (em média)

uma utilização mais elevada. De entre as diversas abordagens para colmatar este problema, o enriquecimento dos dados com diferentes variáveis que permitam aumentar o peso dos postos de potência mais elevado é a abordagem mais proeminente, existindo ainda a possibilidade da alteração da variável *target* para uma de maior flexibilidade.

Capítulo 6

Conclusão e considerações futuras

6.1 Conclusões

A presente dissertação permitiu adquirir uma visão geral do estado atual da mobilidade elétrica em Portugal. Além disso, foi possível estudar diferentes algoritmos de ML de forma a averiguar os mais aptos na resolução de problemas de previsão.

A mobilidade elétrica em Portugal está a desenvolver de forma promissora, no entanto, como se verificou na revisão de literatura, Portugal possui atualmente em média 22 VE por posto de carga, quando o recomendado pela União Europeia é de 10 VE por posto de carga. Com a análise dos dados fornecidos, observou-se na distribuição dos postos de carga uma grande concentração de postos na zona costeira, maioritariamente em Lisboa, Porto e Faro. Este desequilíbrio na distribuição dos postos de carga pode desencorajar os habitantes do interior de Portugal a considerarem a compra de um VE.

Um dos passos essenciais na criação de modelos de ML fiáveis é a aquisição de um *dataset* de boa qualidade. Para este meio, foram utilizadas diversas fontes de dados: os dados dos postos e respetiva atividade proveniente da We Can Charge; os pontos de interesse que rodeiam cada posto adquiridos com a *API Places* da *Google*; e por fim, os dados meteorológicos obtidos pela *API OpenWeatherMap*. Ao *dataset* adquirido, foi ainda realizado um pré-processamento para os preparar para algoritmos de ML.

O membro mais importante do modelo final que recomenda configurações de postos é um modelo de ML que consegue prever uma utilização média dessas configurações para certas coordenadas. Para este fim, foram avaliados seis algoritmos de ML diferentes: *Support Vector Regression*, *eXtreme Gradient Boosting*, *Random Forest Regressor*, *Linear Regression*, *Ridge Model*, *Lasso Model*. Os resultados obtidos revelam que o algoritmo *Random Forest Regressor* foi aquele que apresentou melhor desempenho, com um MAE de 6.6220. De notar, no entanto, que os resultados não foram muito satisfatórios, visto que o erro apresentado pelos modelos criados é relativamente elevado.

Foi posteriormente construído o modelo de recomendação que, através do modelo de ML desenvolvido (RFR), recomenda as melhores configurações encontradas para um posto de carga nas coordenadas fornecidas. Este modelo não deve ser utilizado como uma indicação do que deve ser feito, mas como um método auxiliar para conhecer possíveis configurações que apresentem um bom desempenho.

6.2 Resposta as questões de investigação

6.2.1 Primeira questão

A primeira questão de investigação consistia em averiguar o estado atual da evolução da infraestrutura de carregamento em Portugal. Através da revisão de literatura, foi possível observar que, no que diz respeito a Portugal, a infraestrutura de carregamento está a progredir de forma positiva, mas ainda longe do recomendado. Atualmente, existem 3268 postos de carga para um total de 73 mil VE, resultando em cerca de 22 VE por posto, um valor superior ao recomendado pela União Europeia (10 VE por posto). O desenvolvimento da mobilidade elétrica em Portugal tem sido, no entanto, visível através da adoção de vários incentivos (dedução do iva, isenção de taxas, entre outros).

6.2.2 Segunda questão

A segunda questão debruçava-se sobre como utilizar a IA para prever a utilização futura das postos de carga. A revisão da literatura revelou que existem diferentes abordagens para a utilização futura. Tratando-se de um problema de *forecasting*, pode-se utilizar algoritmos de ML (RFR, SVR, XGBoost, etc) ou, mais adequadamente, algoritmos de DL, como as *Long Short Term Memory Networks*, que foram desenvolvidas especificamente para lidar com problemas de *forecasting*, pelo que se prevê um melhor desempenho com a sua utilização.

6.2.3 Terceira questão

A terceira e última questão compreendia conhecer como evoluir a infraestrutura de carregamento através da utilização de IA. Esta questão revelou-se a mais complexa de responder, no entanto, com a revisão da literatura, encontrou-se alguns métodos que podem ser empregues na resolução deste problema. Um deles corresponde ao método utilizado nesta dissertação, apesar de ser mais limitado, uma vez que exige que o utilizador forneça a localização a avaliar. Além disso, como a revisão de literatura esclareceu, é possível utilizar algoritmos de ML em conjunto com funções objetivo que maximizem diferentes paradigmas, como forma de determinar novas localizações para a colocação de postos de carga.

No entanto, este tópico não compreende apenas a colocação de novos postos de carga. Áreas como a previsão da procura de carregamento e o agendamento de postos de carga também beneficiaram da aplicação de modelos de ML e, apesar de não tratarem exclusivamente da expansão da rede, continuam a ser essenciais, pois fornecem informação sobre a demanda futura de postos de carga, útil para uma melhor gestão da rede de energia elétrica.

6.3 Limitações e considerações futuras

As principais limitações no desenvolvimento desta dissertação foram a qualidade dos dados obtidos e escassez de tempo. O *dataset* utilizado para a realização da dissertação apresentava apenas seis meses de sessões de carga. Ainda mais, os detalhes das sessões de carga não eram ricos em informação, apresentavam apenas o início e o fim da sessão, omitindo qualquer outro pedaço de informação que podia vir a ser útil, por exemplo: o preço da sessão de carga; quanto tempo o veículo ocupou o posto mesmo depois ter a bateria a 100%; entre outros. Apesar das limitações, averiguou-se que a utilização de um *dataset* mais detalhado possibilitaria a criação de modelos de previsão com elevada fiabilidade.

A presente dissertação permitiu conhecer as etapas ideais que se deve seguir para preparar dados para *Machine Learning*. Como forma de estender o alcance da dissertação, seria interessante avaliar o desempenho de algoritmos de *Deep Learning* na resolução do problema proposto, com um *dataset* mais completo e amplo, e comparar com os resultados obtidos neste projeto.

Bibliografia

Muhammad Adnan, Alaa Abdul Salam Alarood, M Irfan Uddin, and Izaz ur Rehman. Utilizing grid search cross-validation with adaptive boosting for augmenting performance of machine learning models. *PeerJ Computer Science*, 8:e803, 2022.

Leo Breiman. *Machine Learning*, 45(1):5–32, 2001. doi: 10.1023/a:1010933404324. URL `https://doi.org/10.1023/a:1010933404324`.

Lubos Buzna, Pasquale De Falco, Shahab Khormali, Daniela Proto, and Milan Straka. Electric vehicle load forecasting: A comparison between time series and machine learning approaches. pages 1–5, 05 2019. doi: 10.1109/SyNERGY-MED.2019.8764110.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. 2016. doi: 10.1145/2939672.2939785.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014.

European Commision. Regulation of the european parliament and of the council on the deployment of alternative fuels infrastructure, and repealing directive 2014/94/eu of the european parliament and of the council. 2021. Page 22, 1st paragraph.

Nello Cristianini, John Shawe-Taylor, et al. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

Qiyun Dang, Di Wu, and Benoit Boulet. A q-learning based charging scheduling scheme for electric vehicles. In *2019 IEEE Transportation Electrification Conference and Expo (ITEC)*, pages 1–5, 2019. doi: 10.1109/ITEC.2019.8790603.

Sanchari Deb. Machine learning for solving charging infrastructure planning problems: A comprehensive review. *Energies*, 14(23), 2021. ISSN 1996-1073. doi: 10.3390/en14237833. URL `https://www.mdpi.com/1996-1073/14/23/7833`.

Dmlc. Xgboost parameters. https://xgboost.readthedocs.io/en/latest/parameter.html, 2023. Acedido a 25/01/2023.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer, Berlin, Heidelberg, 2012. https://doi.org/10.1007/978-3-642-24797-2.

Klaus Greff, Rupesh K. Srivastava, Jan Koutnik, Bas R. Steunebrink, and Jurgen Schmidhuber. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, Oct 2017. ISSN 2162-2388. doi: 10.1109/tnnls.2016.2582924. URL http://dx.doi.org/10.1109/TNNLS.2016.2582924.

Roger Harrabin. Electric car emissions myth 'busted'. https://www.bbc.com/news/science-environment-51977625, March 2020. Acedido a 10/11/2021.

Ned Horning et al. Random forests: An algorithm for image classification and generation of continuous fields data sets. In *Proceedings of the International Conference on Geoinformatics for Spatial Infrastructure Development in Earth and Allied Sciences, Osaka, Japan*, volume 911, pages 1–6, 2010.

Kazuki Irie, Zoltán Tüske, Tamer Alkhouli, Ralf Schlüter, Hermann Ney, et al. Lstm, gru, highway and a bit of attention: An empirical overview for language modeling in speech recognition. pages 3519–3523, 2016.

Jan Koutník, Klaus Greff, Faustino Gomez, and Jürgen Schmidhuber. A clockwork rnn, 2014.

Alexandre Lucas, Ricardo Barranco, and Nazir Refa. Ev idle time estimation on charging infrastructure, comparing supervised machine learning regressions. *Energies*, 12(2), 2019. ISSN 1996-1073. doi: 10.3390/en12020269. URL https://www.mdpi.com/1996-1073/12/2/269.

Donald W. Marquardt and Ronald D. Snee. Ridge regression in practice. *The American Statistician*, 29(1):3–20, 1975. doi: 10.1080/00031305.1975.10479105. URL https://www.tandfonline.com/doi/abs/10.1080/00031305.1975.10479105.

Lucien Mathieu. Roll-out of public ev charging infrastructure in the eu. 2018.

Dastan Maulud and Adnan M Abdulazeez. A review on linear regression comprehensive in machine learning. *Journal of Applied Science and Technology Trends*, 1(4):140–147, 2020.

MOBI.E. Mapa dos postos elétricos em portugal. https://www.mobie.pt/en/redemobie/procurar-posto, 2021. Acedido a 21/11/2021.

Vladimir Nasteski. An overview of the supervised machine learning methods. *Horizons. b*, 4:51–62, 2017.

European Court of Auditors. Infrastructure for charging electric vehicles: more charging stations but uneven deployment makes travel across the eu complicated. 2021.

Christopher Olah. Understanding lstm networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/, Agosto 2015. Acedido a 28/12/2021.

Martijn Otterlo and Marco Wiering. Reinforcement learning and markov decision processes. *Reinforcement Learning: State of the Art*, pages 3–42, 01 2012. doi: 10.1007/978-3-642-27645-3_1.

European Parliament and the council of the European Union. Directive 2014/94/eu of the european parliament and of the council of 22 october 2014 on the deployment of alternative fuels infrastructure. *Official Journal of the European Union*, 2014. Point (23).

Dario Pevec, Jurica Babic, Martin A. Kayser, Arthur Carvalho, Yashar Ghiassi-Farrokhfal, and Vedran Podobnik. A data-driven statistical approach for extending electric vehicle charging infrastructure. *International Journal of Energy Research*, 42:3102–3120, 02 2018. doi: 10.1002/er.3978.

Kevin P.Murphy. *Deep Learning*. MIT Press, 2012.

DECO PROTESTE. Carros elétricos: apoios à compra e condições. https://www.deco.proteste.pt/auto/automoveis/noticias/carros-eletricos-apoios-compra-condicoes?_sm_au_=iVVn5MD7jP1WnDZQBBQNvKssLRJF6, Agosto 2021. Acedido a 20/11/2021.

DECO PROTESTE. Incentivos: quais os apoios do estado para a compra de carros elétricos? https://www.deco.proteste.pt/auto/carros-eletricos/noticias/incentivos-quais-apoios-estado-compra-carros-eletricos/incentivo-do-governo, Abril 2022. Acedido a 23/01/2023.

J Ranstam and J A Cook. LASSO regression. *British Journal of Surgery*, 105(10):1348–1348, 08 2018. ISSN 0007-1323. doi: 10.1002/bjs.10895. URL https://doi.org/10.1002/bjs.10895.

Armin Razmjoo, Arezoo Ghazanfari, Mehdi Jahangiri, Evan Franklin, Mouloud Denai, Mousa Marzband, Davide Astiaso Garcia, and Alireza Maheri. A comprehensive study on the expansion of electric vehicles in europe. *Applied Sciences*, 12(22):11656, November 2022. doi: 10.3390/app122211656. URL https://doi.org/10.3390/app122211656.

Scikit-learn. Linear model trained with l1 prior as regularizer (aka the lasso). https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html#sklearn.linear_model.Lasso, 2023a. Acedido a 25/01/2023.

Scikit-learn. Ordinary least squares linear regression. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html, 2023b. Acedido a 25/01/2023.

Scikit-learn. A random forest regressor. `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html`, 2023c. Acedido a 25/01/2023.

Scikit-learn. Linear least squares with l2 regularization. `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html#sklearn.linear_model.Ridge`, 2023d. Acedido a 25/01/2023.

Scikit-learn. Epsilon-support vector regression. `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html`, 2023e. Acedido a 25/01/2023.

Kang Miao Tan, Vigna K. Ramachandaramurthy, Jia Ying Yong, Sanjeevikumar Padmanaban, Lucian Mihet-Popa, and Frede Blaabjerg. Minimization of load variance in power grids—investigation on optimal vehicle-to-grid scheduling. *Energies*, 10(11), 2017. ISSN 1996-1073. doi: 10.3390/en10111880. URL `https://www.mdpi.com/1996-1073/10/11/1880`.

Abhishek Thakur. *Approaching (almost) any machine learning problem*. Abhishek Thakur, 2020.

Kaisheng Yao, Trevor Cohn, Katerina Vylomova, Kevin Duh, and Chris Dyer. Depth-gated lstm, 2015.

Juncheng Zhu, Zhile Yang, Monjur Mourshed, Yuanjun Guo, Yimin Zhou, Yan Chang, Yanjie Wei, and Shengzhong Feng. Electric vehicle charging load forecasting: A comparative study of deep learning approaches. *Energies*, 12(14), 2019. ISSN 1996-1073. doi: 10.3390/en12142692. URL `https://www.mdpi.com/1996-1073/12/14/2692`.

Apêndice A

Constituição do dataset final

Consideração 7: Exemplo de uma entrada no *dataset*.

```
# A data em que foi medida a utilização do posto de carga
date: 2022-06-07
# Id do posto elétrico;
locationId: PRT-XXXXX
# Número de horas de carregamento que o posto apresentou;
dailyHours: 11.10
# Número de sessões de carregamento que o posto apresentou;
dailySessions: 9
# Cidade em que se encontra localizado;
city: Porto
# Latitude referente à localização representada pela "location_id";
latitude: 41.176276
# Longitude referente à localização representada pela "location_id";
longitude: -8.668786
# Estilo de estacionamento onde o posto se encontra colocado;
parking_type: PARKING_LOT
# Número de conectores neste posto;
number_of_connectors: 2
# A percentagem de utilização dos conetores durante o dia;
connectors_usage: [0.1782407407407407, 5.811342592592593]
# A amperagem(A) de cada um dos conetores;
connectors_amperage: [32, 32]
# A tensão elétrica(T) de cada um dos conetores;
connectors_voltage: [400, 400]
# A velocidade de carga(W) de cada um dos conetores;
connectors_electric_power: [22000, 22000]
# O clima existente no dia denotado pelo valor "date";
weather: Clear
# Componente Lodging dos Pontos de Interesse de uma localização;
lodging: 3
# Componente Store dos Pontos de Interesse de uma localização;
```

```
store: 3
# Componente Food dos Pontos de Interesse de uma localização;
food: 3
# Componente Health dos Pontos de Interesse de uma localização;
health: 24
# Componente Restaurant dos Pontos de Interesse de uma localização;
restaurant: 1
# Componente School dos Pontos de Interesse de uma localização;
school: 1
# Componente Tourist Attraction dos Pontos de Interesse de uma localização;
tourist_attraction: 0
# O distrito onde o posto se encontra localizado.
region: Porto
```

Apêndice B

Distribuição das estações

Figura 26: Distribuição das estações em Portugal Continental

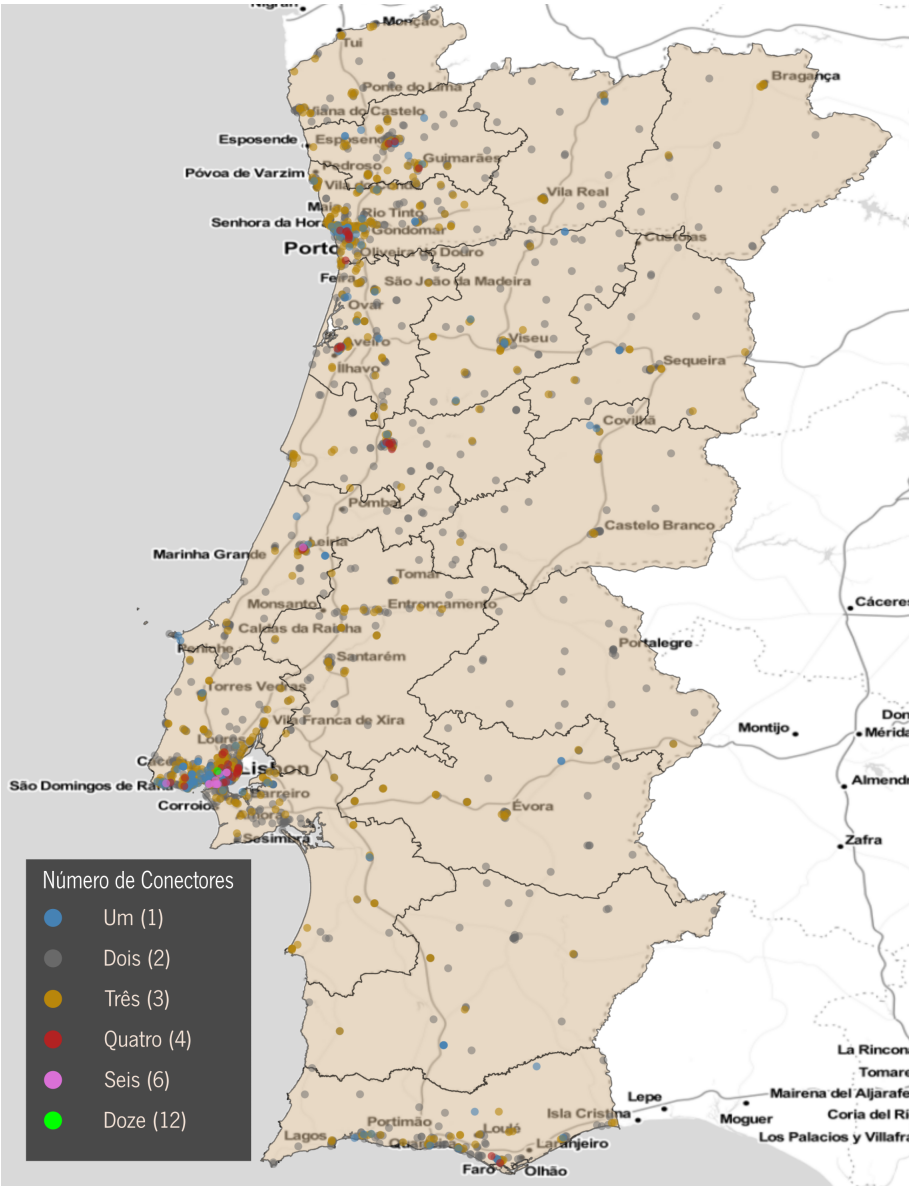
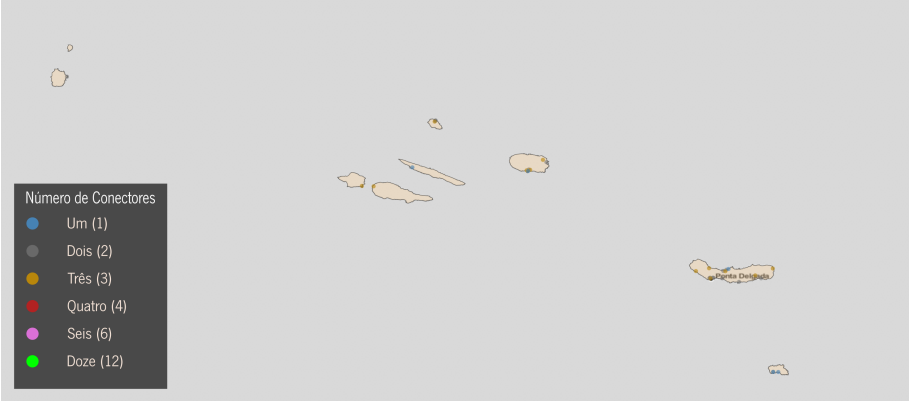


Figura 27: Distribuição das estações na Madeira

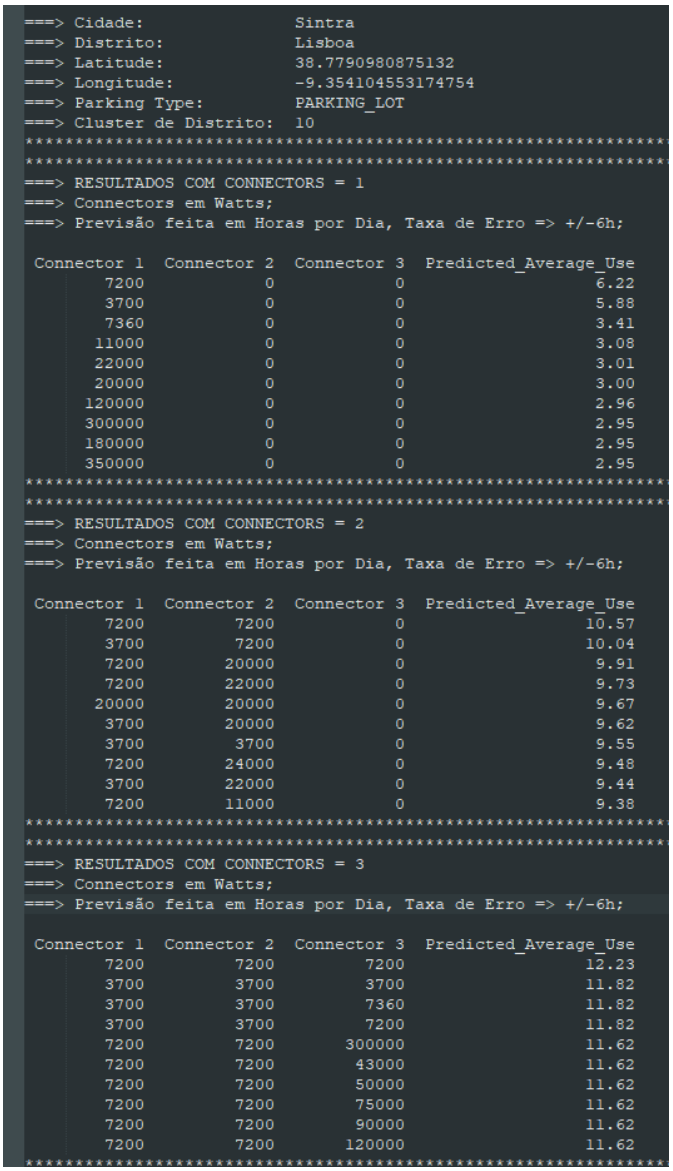
Figura 28: Distribuição das estações nos Açores



Apêndice C

Output do modelo final

Figura 29: *Output* do modelo.



```
===> Cidade:              Sintra
===> Distrito:            Lisboa
===> Latitude:            38.7790980875132
===> Longitude:           -9.354104553174754
===> Parking Type:        PARKING_LOT
===> Cluster de Distrito: 10
**********************************************************************
**********************************************************************
===> RESULTADOS COM CONNECTORS = 1
===> Connectors em Watts;
===> Previsão feita em Horas por Dia, Taxa de Erro => +/-6h;

 Connector 1  Connector 2  Connector 3  Predicted_Average_Use
        7200           0           0                   6.22
        3700           0           0                   5.88
        7360           0           0                   3.41
       11000           0           0                   3.08
       22000           0           0                   3.01
       20000           0           0                   3.00
      120000           0           0                   2.96
      300000           0           0                   2.95
      180000           0           0                   2.95
      350000           0           0                   2.95
**********************************************************************
**********************************************************************
===> RESULTADOS COM CONNECTORS = 2
===> Connectors em Watts;
===> Previsão feita em Horas por Dia, Taxa de Erro => +/-6h;

 Connector 1  Connector 2  Connector 3  Predicted_Average_Use
        7200         7200           0                  10.57
        3700         7200           0                  10.04
        7200        20000           0                   9.91
        7200        22000           0                   9.73
       20000        20000           0                   9.67
        3700        20000           0                   9.62
        3700         3700           0                   9.55
        7200        24000           0                   9.48
        3700        22000           0                   9.44
        7200        11000           0                   9.38
**********************************************************************
**********************************************************************
===> RESULTADOS COM CONNECTORS = 3
===> Connectors em Watts;
===> Previsão feita em Horas por Dia, Taxa de Erro => +/-6h;

 Connector 1  Connector 2  Connector 3  Predicted_Average_Use
        7200         7200         7200                  12.23
        3700         3700         3700                  11.82
        3700         3700         7360                  11.82
        3700         3700         7200                  11.82
        7200         7200       300000                  11.62
        7200         7200        43000                  11.62
        7200         7200        50000                  11.62
        7200         7200        75000                  11.62
        7200         7200        90000                  11.62
        7200         7200       120000                  11.62
**********************************************************************
```