Previsão de Consumos Energéticos em Edifícios não Residenciais com Recurso a Métodos de *Machine Learning*

Dissertação

Mestrado em Engenharia Eletrotécnica

Francisco Rafael Ladeira Fernandes

Leiria, março de 2022

escola superior de tecnologia e gestão
instituto politécnico de leiria

Previsão de Consumos Energéticos em Edifícios não Residenciais com Recurso a Métodos de *Machine Learning*

Dissertação

Mestrado em Engenharia Eletrotécnica

Francisco Rafael Ladeira Fernandes

Dissertação realizada sob a orientação dos Professores João Sousa e Hermano Bernardo

Leiria, março de 2022

Este trabalho foi parcialmente apoiado através de uma Bolsa de Investigação com a referência UI0308E.Energética/Edifícios.1/2020, no âmbito do Financiamento Plurianual de Unidades de I&D 2020-2023 (UIDB/00308/2020).

Originalidade e Direitos de Autor

A presente dissertação é original, elaborada unicamente para este fim, tendo sido devidamente citados todos os autores cujos estudos e publicações contribuíram para o elaborar.

Reproduções parciais deste documento serão autorizadas na condição de que seja mencionado o Autor e feita referência ao ciclo de estudos no âmbito do qual o mesmo foi realizado, a saber, Curso de Mestrado em Engenharia Eletrotécnica, no ano letivo 2020/2021, da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria, Portugal, e, bem assim, à data das provas públicas que visaram a avaliação destes trabalhos.

Agradecimentos

Começo por expressar um agradecimento especial aos meus orientadores, os Professores João Sousa e Hermano Bernardo, cuja disponibilidade, dedicação e colaboração demonstradas tiveram um papel determinante na elaboração da presente dissertação. Por tudo isto e por todos conhecimentos partilhados o meu muito obrigado.

Ao Instituto de Engenharia de Sistemas e Computadores de Coimbra, por ter providenciado uma bolsa de investigação no âmbito da presente dissertação.

Ao Instituto Politécnico de Leiria, em especial à Escola Superior de Tecnologia e Gestão, e respetivos colaboradores, pela qualidade da formação que me foi prestada no Mestrado e na Licenciatura.

À Direção dos Serviços Técnicos do Instituto Politécnico de Leiria, pela disponibilidade e empenho na cedência de dados de consumo solicitados.

Aos docentes e colegas com quem me cruzei nos cursos de Licenciatura em Engenharia Eletrotécnica e de Computadores e de Mestrado em Engenharia Eletrotécnica, os quais contribuíram para o desenvolvimento dos meus conhecimentos.

À minha família, em especial aos meus pais e irmã, à minha namorada e aos meus amigos, pelo apoio incondicional que sempre prestaram.

Resumo

O crescimento descontrolado a nível mundial dos consumos de energia, muitas vezes produzida a partir de combustíveis fósseis, tem vindo a revelar-se um grande problema para a humanidade, sendo das maiores causas para as preocupações ambientais que se vivem atualmente, nomeadamente o aquecimento global. A par do setor da indústria e dos transportes, os maiores consumidores de energia são os edifícios, a que inclusive estão associados diferentes vetores energéticos (como a eletricidade e o gás natural) para diferentes utilizações finais de energia como iluminação, climatização, produção de águas quentes sanitárias ou equipamentos de tecnologia de informação. A previsão eficiente e atempada de consumos energéticos em edifícios, possibilitada pela facilidade de acesso que existe atualmente a grandes quantidades de dados, fornecidos por equipamentos modernos como os *smart meters*, tem então um papel muito relevante na procura por uma boa gestão de energia. As técnicas de *machine learning* têm vindo a revelar-se como promissoras na construção desse tipo de modelos de previsão, de tal forma que nos últimos anos o número de artigos publicados nessa área tem sido significativo.

Esta dissertação consiste na criação de modelos de *machine learning* recorrendo a vários algoritmos e com vista à previsão de curto prazo de consumos energéticos em edifícios não residenciais, mais concretamente no Campus 2 do Instituto Politécnico de Leiria. São abordadas as diferentes fases envolvidas na criação desses modelos. Inicialmente é feita uma análise e tratamento dos dados, seguindo-se uma seleção e combinação de *features.* De seguida são aplicados vários algoritmos com diferentes parametrizações. Por fim são selecionados e testados os melhores modelos e é feita uma análise dos resultados com recurso a várias técnicas de benchmarking.

Palavras-chave: Aprendizagem automática, Modelos de previsão de consumos energéticos, Edifícios não residências, Análise e pré-processamento de dados, Seleção e extração de *features*, Benchmarking entre modelos

Abstract

The uncontrolled growth of energy consumption worldwide, often derived from fossil fuels, has proven to be one of the main problems that humanity has been facing, being a major cause for today's environmental concerns, namely global warming. Alongside industry and transport sectors, the biggest energy consumers are buildings, which are also associated with different energy vectors (such as electricity and natural gas) for different energy end uses such as lighting, air conditioning, production of domestic hot water or information technology equipment. The efficient and timely forecasting of energy consumption in buildings, made possible nowadays by the ease of access to large amounts of data, provided by modern equipment such as smart meters, has a relevant role in the search for a good energy management. Machine learning techniques have been shown to be promising in the construction of this type of forecasting models, which explains why the number of articles published is this subject has been significant.

This dissertation consists of the creation of machine learning models using various algorithms with the goal of predicting short-term energy consumption in non-residential buildings, specifically on the Campus 2 of the Polytechnic Institute of Leiria. The different phases involved on the creation of these models are discussed. Initially, an analysis and treatment of the data is carried out, followed by a selection and combination of features. Then, several algorithms with different parameterizations are applied. Finally, the best models are selected, tested and the results are analyzed using several benchmarking techniques.

Keywords: Machine Learning, Energy consumption forecasting models, Non-residential buildings, Data analysis and pre-processing, Feature selection and extraction, Forecasting Models Benchmark.

Índice

Lista de Figuras

Lista de tabelas

Lista de siglas e acrónimos

ANN	*Artificial Neural Networks*
ESTG	Escola Superior de Tecnologia e Gestão
IQR	*Interquartile Range*
IPLEIRIA	Instituto Politécnico de Leiria
LSVM	*Linear Support Vector Machines*
MAE	*Mean Absolute Error*
MAPE	*Mean Absolute Percentage Error*
MARS	*Multivariate Adaptive Regression Splines*
MAD	*Median Absolute Deviation*
MSE	*Mean Squared Error*
RAM	*Random Access Memory*
RBF	*Radial Basis Function*
REN	Redes Energéticas Nacionais
RNN	*Recurrent Neural Network*
RMSE	*Root Mean Squared Error*
SVM	*Support Vector Machines*
PCA	*Principal Component Analysis*
UTC	*Coordinated Universal Time*

1. Introdução

Pretende-se neste capítulo descrever a motivação por detrás da presente dissertação, bem como os objetivos e estrutura gerais da mesma.

1.1.Motivação

Com o desenvolvimento da humanidade e o constante crescimento da população mundial, os consumos energéticos têm também verificado um constante aumento nas últimas décadas, o que constitui um problema grave devido aos problemas ambientais causados pela utilização dos combustíveis fósseis, os quais continuam a ser a mais utilizada fonte de energia e a contribuir para o aumento das emissões globais de CO_2. Estes factos são comprovados pelos gráficos apresentados na Figura 1-1 e Figura 1-3, retirados do "Statistical Review of World Energy 2021" da BP [1].

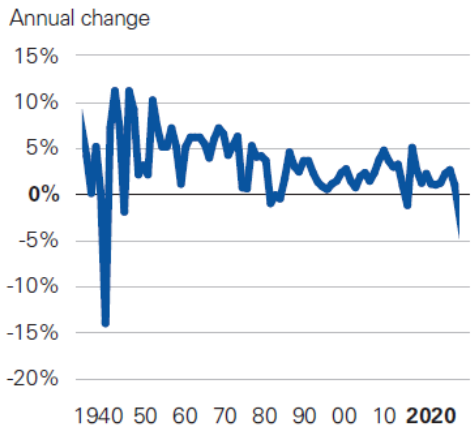


Figura 1-1:Evolução do consumo global de energia primária [1]

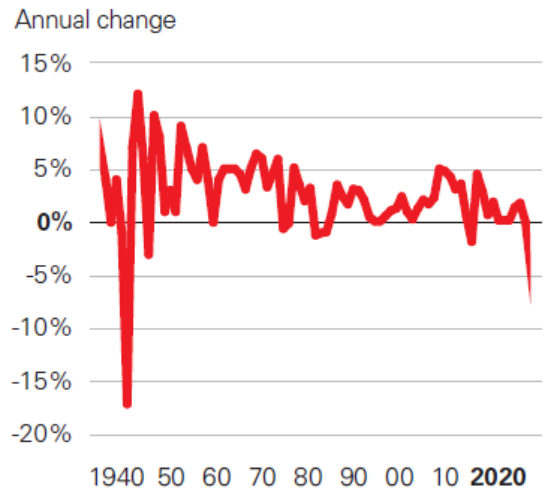


Figura 1-2-Evolução das emissões globais de CO_2 [1]

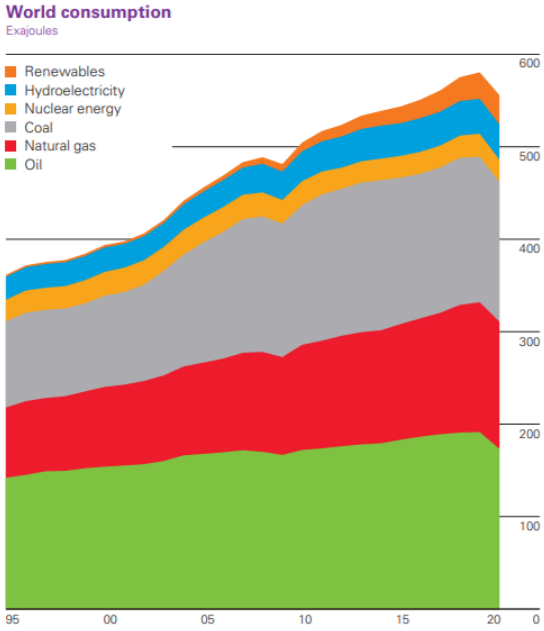


Figura 1-3-Consumo global de energia primária por fonte [1]

Posto isto, o desenvolvimento de investigação com vista à implementação de medidas que possam ajudar à melhor gestão dos recursos energéticos é essencial. Uma das áreas que tem despertado interesse nos últimos anos é a da aprendizagem automática, mais comumente denominada de *machine learning*, nomeadamente para previsão de consumos energéticos de diversos tipos (eletricidade, aquecimento, arrefecimento…).

De facto, a previsão de consumos energéticos constitui uma mais-valia na gestão energética pelas mais variadíssimas razões. Por exemplo, o despacho económico, o qual funciona procurando satisfazer em cada instante a premissa da produção satisfazer o consumo de energia, tem visto nos últimos anos a sua tarefa mais dificultada devido à integração de fontes de energia bastante inconstantes do ponto de vista da produção. O caso mais evidente é o de integração de energia eólica, pelo que uma estimativa fiável do consumo, e nesse caso também da produção, são uma ferramenta útil neste problema. Na Figura 1-4 e na Figura 1-5 são apresentadas, para o dia 30 de outubro de 2021, a repartição da produção total e da produção renovável não hídrica, respetivamente, retirados do Data Hub da REN (Redes Energéticas Nacionais) [2] e que mostram, ao retratarem uma queda abrupta da produção eólica, o quão inconstante é esta forma de energia.

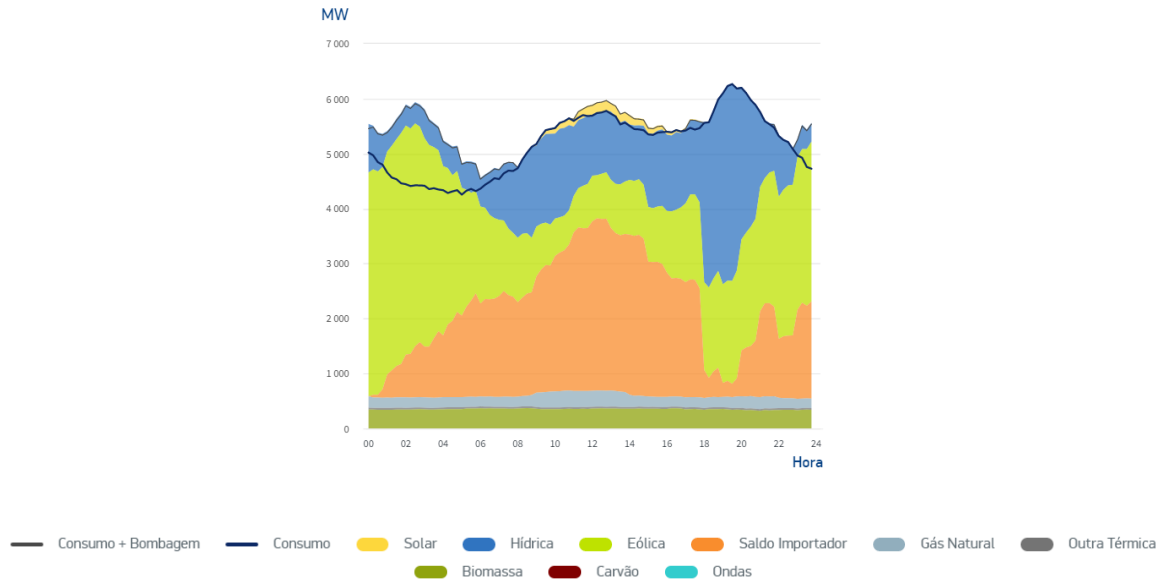


Figura 1-4-Repartição da produção total (dia 30/10/2021)[2]

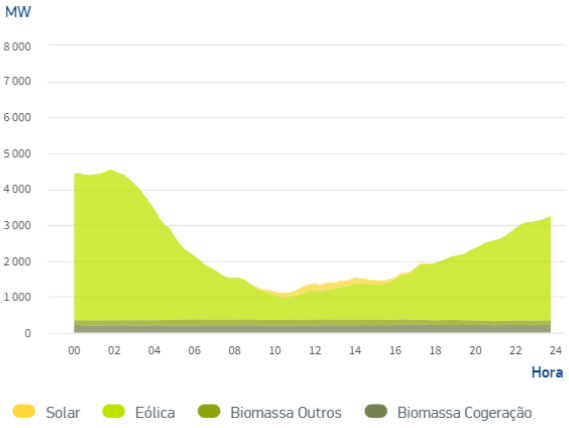


Figura 1-5-Repartição da produção renovável não hídrica (dia 30/10/2021)[2]

Outro exemplo é que, sabendo os consumos energéticos antecipadamente, uma entidade pode obter poupanças na sua fatura energética, ao procurar controlar a sua máxima potência ativa média, a qual irá ditar a potência contratada a ser faturada, para o caso das instalações que não sejam de baixa tensão normal, o que é em muitas vezes o caso em edifícios não residenciais. Também a deteção a priori de valores de consumos anormalmente altos permite agir atempadamente, evitando desta forma encargos com consumos desnecessários.

A utilização de modelos de previsão de consumos energéticos é então importante em diferentes setores. No caso dos edifícios, estes são responsáveis por uma parte significativa dos consumos energéticos mundiais com 30% em média, bem como um terço das emissões mundiais de CO_2[1], pelo que a pertinência destes modelos tende a merecer especial destaque. O gráfico apresentado na Figura 1-6, retirado da "Energy Outlook 2019" da BP [3] mostra o peso relativo dos edifícios no consumo total em termos do consumo de energia final por setor, projetando também o peso esperado até 2040. Como se pode ver a importância da energia usada nos edifícios tem vindo a aumentar como resultado do crescimento da prosperidade nos países desenvolvidos, o qual leva a um aumento significativo das necessidades energéticas, nomeadamente para arrefecimento de espaços, iluminação e aparelhos elétricos[2].

[1] "Buildings account for a significant part of the global energy consumption with 30% in average and a third of the associated CO2 emissions (International Energy Agency, 2016)."[66]
[2] "The importance of energy used within buildings expands over the Outlook, as growing prosperity in developing economies leads to significant increases in power demand, for space cooling, lighting and electrical appliances" [3].

O “Energy Outlook 2019” refere no entanto que esse crescimento poderá ser muito menor num cenário em que sejam tomadas medidas mais notórias, nomeadamente de eficiência energética.

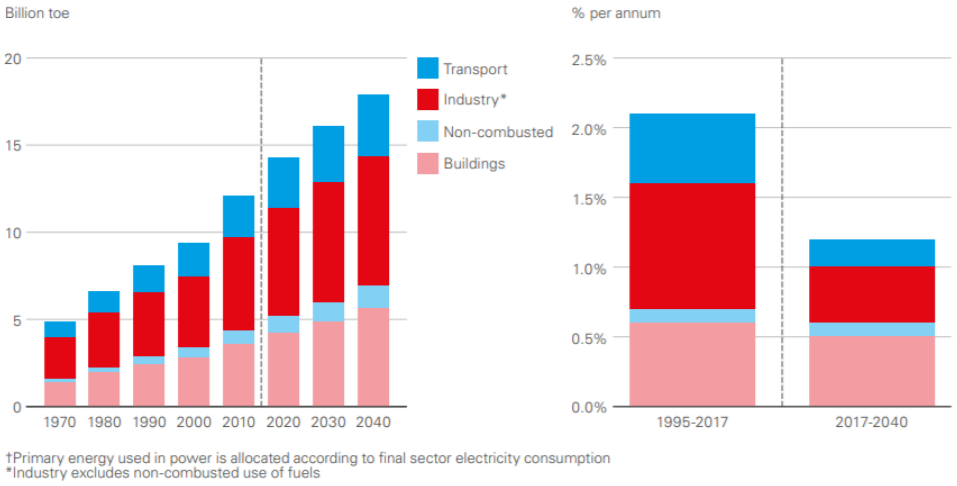


Figura 1-6-Consumo de energia primária por setor e aumento anual da procura por setor[3]

O peso dos consumos dos edifícios em Portugal é também elevado, sendo igual a, segundo a edição 2021 da publicação “Energia em Números” [4], 18% para o setor doméstico e de 12,1% para o setor dos serviços, estando por isso de acordo com os consumos globais.

Com vista ao combate às alterações climáticas vários países, incluindo Portugal, comprometeram-se a atingir a neutralidade carbónica até 2050. Deste compromisso resultou o roteiro para a neutralidade carbónica, o qual “estabelece, de forma sustentada, a trajetória para atingir a neutralidade carbónica em 2050, define as principais linhas de orientação e identifica as opções economicamente viáveis para atingir aquele fim, em diferentes cenários de desenvolvimento socioeconómico.” [5]. Analisando o referido documento é possível perceber a importância dos edifícios para atingir o compromisso assumido uma vez que este é dos setores com mais potencial de redução de emissões, como mostra a Tabela 1-1, onde é visível que a redução em 2050 face a 2005 possa ser de até 85%.

Tabela 1-1-Potencial de redução de emissões em relação a 2005 resultante do exercício de modelação[5].

SECTORES	2030	2040	2050
Energia	80% I 81%	92%	96%
Indústria	52% I 48%	59% I 60%	73% I 72%
Edifícios	48% I 49%	73% I 74%	85%
Transportes	43% I 46%	84% I 85%	98%
Agricultura e usos solo	36% I 39%	37% I 49%	38% I 60%
Resíduos e Águas residuais	57% I 58%	69% I 71%	77% I 80%

O roteiro prevê ainda que os principais *drivers* da descarbonização dos setores residencial e serviços sejam a eficiência energética, a eletrificação, o isolamento e reabilitação e o solar térmico e a instalação de bombas de calor. É de destacar o facto de, para o caso do setor dos serviços no qual esta dissertação se enquadra, ser prevista uma redução de 100% dos gases com efeito de estufa. Na Figura 1-7 é apresentada a narrativa de neutralidade carbónica até 2050 do setor dos serviços, a qual permite visualizar que medidas permitirão que essa redução de emissões seja atingida. A nível de custos, o roteiro estima que, no que respeita a habitação e serviços, "a maioria do investimento estará relacionado com a renovação e substituição de equipamentos elétricos por equipamentos mais eficientes" [5], sendo também feita referência a despesas com isolamento de edifícios e instalação de bombas de calor.

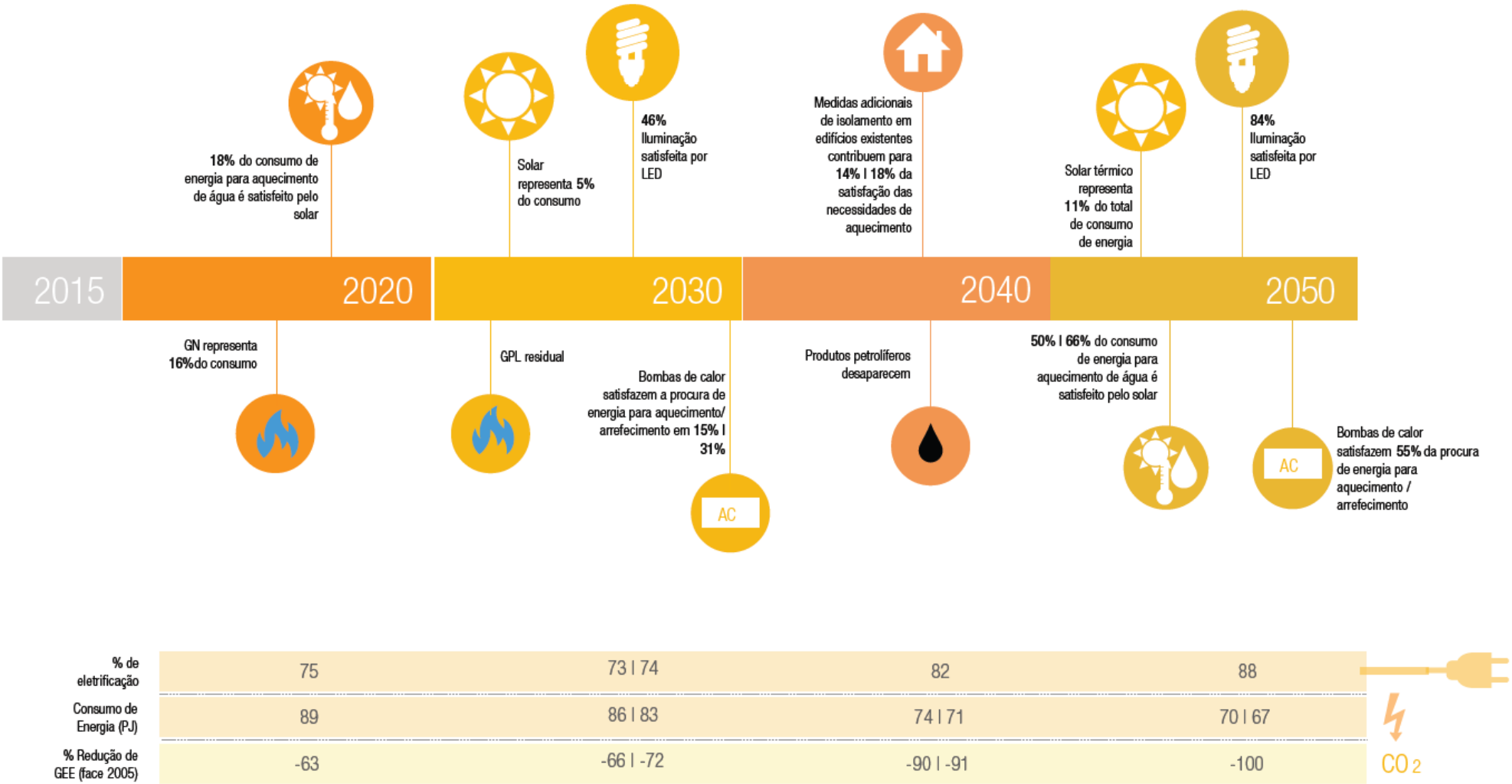


Figura 1-7-Narrativa de neutralidade carbónica até 2050 do setor dos serviços [5]

1.2.Objetivos e estrutura

A presente dissertação consiste no desenvolvimento de modelos de *machine learning* com o objetivo de prever os valores de potência ativa consumidos pelo Campus 2 do Instituto Politécnico de Leiria (visível na Figura 1-8, através de imagem retirada do Google Earth [6]). Pretende-se abordar as diferentes fases inerentes à criação e avaliação deste tipo de modelos, que neste caso serão aplicados a horizontes de previsão de curto prazo (para o dia seguinte), de maneira a atingir previsões confiáveis.

Figura 1-8-Campus 2 do Instituto Politécnico de Leiria [7]

No primeiro capítulo é feita uma descrição da motivação para este trabalho, tendo em conta o contexto energético mundial e nacional atual, bem como uma breve descrição dos objetivos e estrutura da presente dissertação.

No segundo capítulo é feita uma breve revisão teórica dos algoritmos de *machine learning* utilizados.

No terceiro capítulo realiza-se uma análise geral do conjunto de dados, bem como a divisão destes em períodos de treino e de teste e ainda a escolha da técnica de validação utilizada.

No quarto capítulo é feito o pré-processamento dos dados, essencial para o bom desempenho dos modelos.

No quinto capítulo é feita a seleção e combinação das entradas com recurso a diversas técnicas, bem como uma introdução às métricas de erro utilizadas.

No sexto capítulo é feita a parametrização dos modelos e os testes finais no conjunto de teste, até então deixado de lado por forma a fornecer um resultado que não seja tendencioso, evitando assim problemas de *data leakage*. É também feita neste capítulo a avaliação e comparação do desempenho dos diversos modelos, com base em diferentes técnicas de benchmarking.

Por fim, no sétimo capítulo, são enumeradas as principais conclusões em relação ao trabalho desenvolvido e descritas pistas para investigação futura com base no trabalho que ficou por desenvolver.

2. Revisão teórica dos algoritmos de *Machine Learning* utilizados

Pretende-se neste capítulo efetuar uma breve explicação introdutória dos princípios por detrás dos diferentes algoritmos utilizados no que respeita a algoritmos de seleção e extração de variáveis, algoritmos de previsão e algoritmos de otimização usados na escolha dos parâmetros dos modelos.

2.1. Algoritmos de previsão

2.1.1. Modelo de Persistência

Uma boa prática em *machine learning* é a utilização de um modelo de persistência, o qual serve de *baseline* para o problema em causa, pelo que se um determinado modelo revelar uma performance igual ou pior que o *baseline*, esse modelo deverá ser corrigido ou descartado[3], já que o objetivo desta abordagem não é o de obter um desempenho elevado mas sim o de obter metas mínimas de referência para os restantes algoritmos. De acordo com a bibliografia [8], um modelo que sirva a este propósito deve ser simples, rápido e repetível. Um modelo que satisfaz estas premissas é o modelo de persistência que consiste basicamente em assumir que o valor previsto para uma dada variável é igual a um valor anterior dessa mesma variável. No caso da presente dissertação optou-se então, com base na análise de autocorrelação, por assumir que o valor previsto é igual ao valor registado anteriormente com um desfasamento de uma semana. Este modelo foi então aplicado apenas no conjunto de teste, já que não necessita de qualquer parametrização e funcionará apenas como técnica de benchmarking.

[3] "It is a point of reference for all other modeling techniques on your problem. If a model achieves performance at or below the baseline, the technique should be fixed or abandoned."[8]

2.1.2. Redes Neuronais Artificiais

As redes neuronais artificiais não se enquadram no grupo dos modelos convencionais, sendo consideradas um modelo não linear no caso de se usarem funções de ativação não lineares. Este método procura simular o comportamento do cérebro humano, o que pode ser visto analisando a estrutura das mesmas, a qual consiste em pelo menos três camadas: a de entrada, a/as oculta/as (o número de camadas ocultas é parametrizável) e a de saída. A primeira possui o conjunto das variáveis de entrada. Na camada oculta essas entradas são sujeitas a uma ponderação, sendo posteriormente somadas entre si e a um termo constante (*offset*). O resultado dessa soma é sujeito a uma função de ativação (função identidade, *ReLu*, ou uma função sigmoide para permitir evidenciar relações não lineares entre entradas e saídas). Esta estrutura pode ser vista na Figura 2-1 e na Figura 2-2, onde são apresentadas ilustrações de um neurónio e de uma rede neuronal, neste caso com várias ligações e mais do que um neurónio. A estrutura da rede poderá ser mais complexa quanto mais neurónios e camadas ocultas estiverem presentes na rede. A escolha cuidada destes parâmetros tem reflexos na performance do modelo ANN ("Artificial Neural Networks"), daí que seja normalmente feita, como acontece na referência [9], a comparação das métricas de erro com diferentes parametrizações. O algoritmo possui ainda muitos outros parâmetros, o que constitui uma das principais desvantagens do mesmo, já que necessita de uma parametrização complexa e morosa, no entanto é um dos métodos mais populares na aprendizagem automática atualmente, produzindo geralmente bons resultados em problemas mais complexos e com relações não lineares, razão pela qual se optou pela sua utilização.

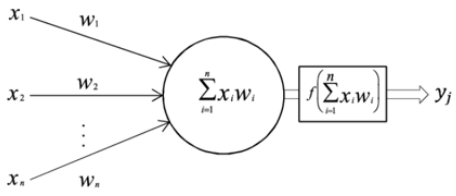


Figura 2-1-Exemplo de estrutura de um neurónio numa rede neuronal [10]

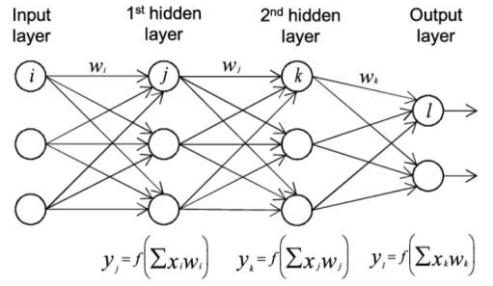


Figura 2-2-Exemplo de estrutura de uma rede neuronal [10]

Para implementar este algoritmo recorreu-se ao modelo "*MLPRegressor*" da biblioteca *scikit-learn* , cuja documentação pode ser consultada em [11], e os parâmetros desse modelo a que será dado enfase, nomeadamente na fase de otimização de parâmetros, são os exibidos na Tabela 2-1.

Tabela 2-1-Parâmetros do algoritmo ANN considerados no processo de otimização (baseada na documentação [11])

Nome na biblioteca sklearn	Significado	Valor/Tipo de Variável
hidden_layer_sizes	Número de camadas ocultas e de neurónios em cada camada	Tuple
activation	Função de ativação para a(s) camada(s) oculta(s)	Identidade, Relu, Tanh, Logistic (categorical)
solver	O solver a utilizar na otimização dos pesos	Lbfgs, Sgd, Adam (categorical)
alpha	Termo de regularização	Float
learning_rate	Taxa de aprendizagem para a atualização dos pesos	Constant, Invscalling, Adaptive (categorical)
learning_rate_init	Taxa de aprendizagem usada inicialmente	Float

2.1.3. Support Vector Machines

O princípio de funcionamento deste algoritmo, que foi inicialmente proposto para fins de classificação, foi desenvolvido para fins de problemas de regressão em 1995 na referência [12] e consiste num mapeamento através de uma função de *kernel* (função linear, polinomial, rbf ("*Radial Basis Function*")…) com vista a aprender uma função que represente uma adequada aproximação à série temporal associada à variável a prever.

É então definido um hiperplano em que os pontos adjacentes ao mesmo se denominam de vetores de suporte. Para além da função kernel, é necessária a especificação de mais alguns parâmetros dos quais se destacam a margem de erro e o custo, os quais irão refletir a tolerância ao erro admitido. Na Figura 2-3 é apresentada uma imagem que ilustra este método.

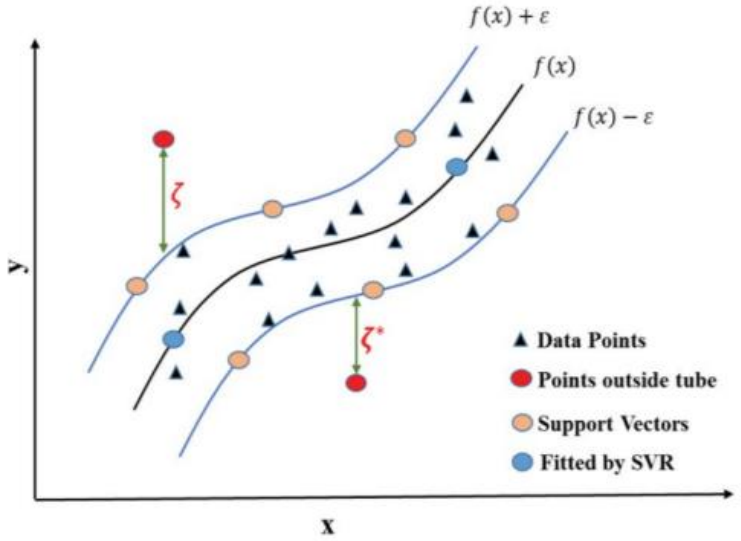


Figura 2-3-Ilustração das Support Vector Machines [13]

Este algoritmo foi implementado inicialmente com o modelo "SVR" da biblioteca *scikit-learn* e a sua documentação pode ser consultada em [14]. Por razões descritas no capítulo 6 utilizou-se no final o modelo "LinearSVR" da mesma biblioteca, cuja documentação se encontra em [15]. Na Tabela 2-2 são apresentados os parâmetros que de acordo com a bibliografia são mais relevantes neste algoritmo.

Tabela 2-2-Parâmetros do algoritmo SVM considerados no processo de otimização (baseada na documentação [15])

Nome na biblioteca sklearn	Significado	Valor/Tipo de Variável
epsilon	Parâmetro epsilon na epsilon *insensitive-loss function* (erros menores que epsilon são igonorados)	*Float*
C	Termo de regularização	*Float*

2.1.4. Métodos *ensemble, Decision trees* e *Random Forests*

Outro algoritmo de previsão que se utilizou foram as *Random Forests*. Para perceber o princípio por detrás deste algoritmo é importante abordar dois conceitos, o dos modelos "ensemble" e o do algoritmo das *Decision Trees*.

Os modelos *ensemble* procuram efetuar previsões com uma abordagem diferente. Em vez de procurarem a melhor hipótese para explicar os dados, constroem um conjunto de hipóteses (por vezes chamados de "ensemble") procedendo depois, de uma determinada maneira, a um processo de "votação" para efetuar a previsão.[4] Existem dois tipos principais de modelos *ensemble*, chamados de *Bagging* e *Boosting*. No primeiro, são geradas múltiplas versões de um modelo de previsão, as quais são depois usadas para ter um modelo agregado. Essa agregação é feita realizando a média entre as versões, no caso de um problema de regressão, e uma votação no caso de um problema de classificação. [5] O tipo *Boosting* funciona repetindo a execução de um dado algoritmo fraco em várias distribuições sobre os dados de treino, combinando os modelos de previsão produzidos num único modelo.[6] Estes dois processos estão representados na Figura 2-4, retirada da bibliografia.

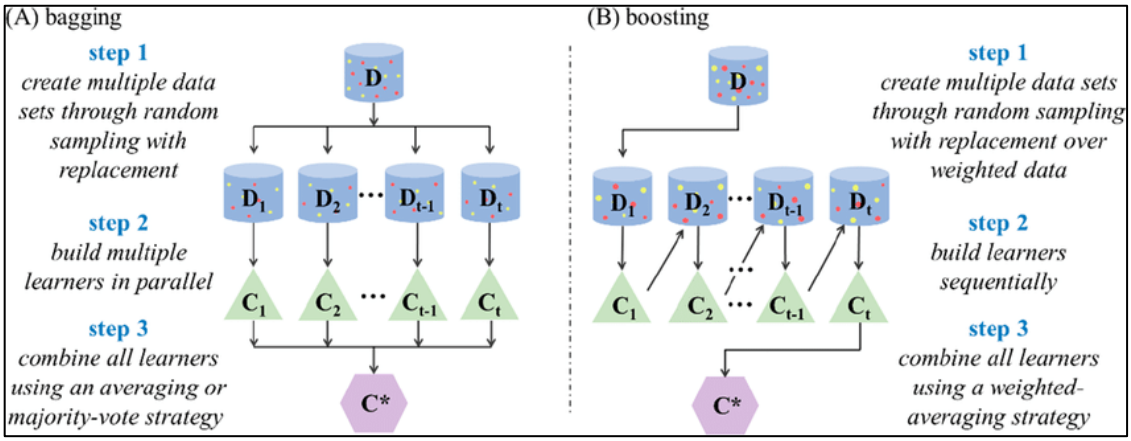


Figura 2-4-Exemplo de ilustração dos processos Bagging e Boosting [16]

[4] "Ensemble learning algorithms take a different approach. Rather than finding one best hypothesis to explain the data, they construct a set of hypotheses (sometimes called a "committee" or "ensemble") and then have those hypotheses "vote" in some fashion to predict the label of new data points."[67]

[5] "Bagging predictors is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. The aggregation averages over the versions when predicting a numerical outcome and does a plurality vote when predicting a class."[68]

[6] "Boosting works by repeatedly running a given weak learning algorithm on various distributions over the training data, and then combining the classifiers produced by the weak learner into a single composite classifier."[69]

No algoritmo das *Decision Trees ou* árvores de decisão, uma árvore representa uma segmentação dos dados criada ao aplicar uma série de regras simples. Estes modelos geram um conjunto de regras que podem ser usadas na previsão a partir de um processo repetitivo de divisão. [7] Uma árvore consiste então num conjunto de nós formados a partir de um nó inicial denominado "raiz", o qual não possui arestas de chegada, seguindo-se nós internos ou de teste com arestas de chegada e de saída, terminando em nós denominados "*leaves*" ou "folhas", os quais possuem apenas arestas de chegada.[8] Um exemplo de uma *Decison Tree*, retirada da bibliografia, é exibido na Figura 2-5.

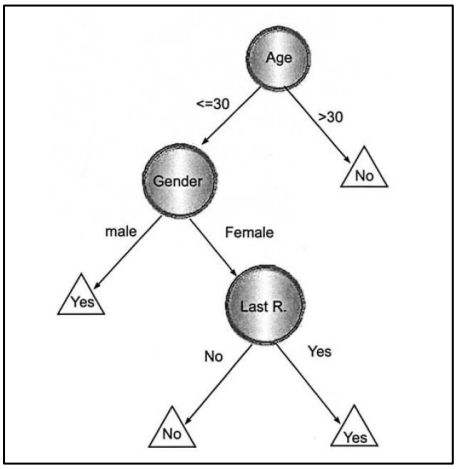


Figura 2-5-Exemplo de Decision Tree [17]

O algoritmo *Random Forests* consiste num modelo *ensemble* do tipo *bagging*, de *Decision Trees*, daí a importância dos conceitos anteriores, podendo ser visto, como o nome indica, como um conjunto de árvores de decisão, em que várias árvores são corridas em paralelo com subconjuntos aleatórios do *dataset* e das *features*, sendo no final incluídos os resultados numa operação de média a qual permite chegar a um resultado final de previsão. O funcionamento deste algoritmo encontra-se ilustrado na Figura 2-6, retirada da bibliografia.

[7] "In decision tree modeling, an empirical tree represents a segmentation of the data that is created by applying a series of simple rules. These models generate a set of rules which can be used for prediction through the repetitive process of splitting."[70]

[8] "The decision tree consists of nodes that form a rooted tree, meaning it is a directed tree with a node called "root" that has no incoming edges. All other nodes have exactly one incoming edge. A node with outgoing edges is called an internal or test node. All other nodes are called leaves (also known as terminal or decision nodes)." [17]

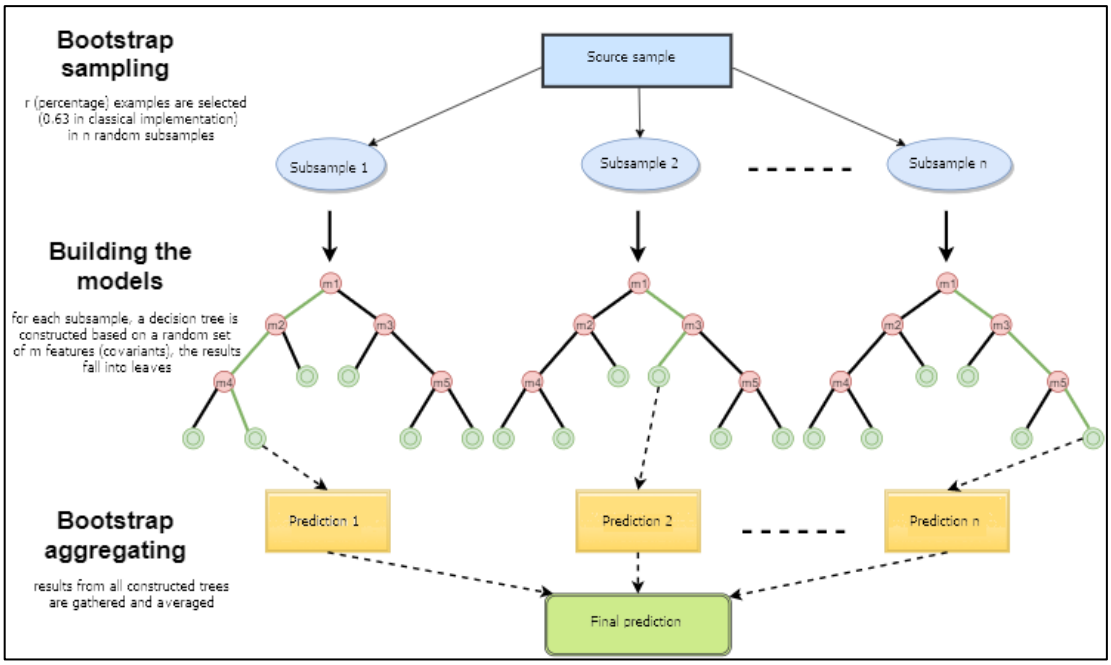


Figura 2-6-Ilustração do funcionamento do algoritmo Random Forests [18]

Quando comparado com as *Decision Trees* as *Random Forests* possuem várias vantagens, nomeadamente um risco menor de *overfit*, ou seja uma melhor capacidade de generalização sem estar demasiado fiel aos dados de treino, um funcionamento melhor com dados não lineares e uma maior robustez a registos anómalos, necessitando no entanto de tempos de treino maiores e não possuindo uma interpretação visual tão simples [19].

As *Random Forests* efetuam também uma *feature selection* automática, fornecendo coeficientes que retratam a importância de cada variável, tendo esse facto sido também importante na escolha deste algoritmo, que foi então também usado nesta dissertação para efeitos de seleção de variáveis. Para além disso, o facto de existirem experiências que provam que os métodos ensemble são mais precisos que qualquer hipótese singular [9] motivou a utilização de um algoritmo como o *Random Forests*.

Este algoritmo foi implementado com a biblioteca *scikit-learn* e a sua documentação pode ser consultada em [20]. Na Tabela 2-3 são apresentados os parâmetros deste algoritmo que se otimizaram no capítulo 6, bem como o seu significado.

[9] "Experimental evidence has shown that ensemble methods are often much more accurate than any single hypothesis."[67]

Tabela 2-3-Parâmetros do algoritmo Random Forests considerados no processo de otimização (baseada na documentação [20])

Nome na biblioteca sklearn	Significado	Valor/Tipo de Variável
n_estimators	Número de árvores utilizadas na floresta	Int
max_depth	Profundidade (=número de divisões) máxima de cada árvore	Int
min_samples_split	Número mínimo de amostras necessário para dividir um nó interno	Int
min_samples_leaf	Número mínimo de amostras mínimo necessário num nó leaf	Int
max_features	Número de variáveis consideradas na procura da melhor divisão	Int
max_samples	Número (ou percentagem) de amostras a utilizar no treino de cada árvore	Int ou Float

2.1.5. *Linear Regression* e MARS (*Multivariate Adaptive Regression Splines*)

O algoritmo MARS, introduzido por Jerome H. Friedman em [21], consiste basicamente numa melhoria da regressão linear, a qual efetua a previsão através de uma equação linear que representa a relação entre a variável a prever e as entradas, assumindo por isso uma relação linear entre as variáveis. No entanto, uma vez que em muitos problemas do mundo real as relações entre as variáveis são não lineares, uma simples regressão linear acaba por ter um mau desempenho em problemas mais complexos[10], razão pela qual só foi utilizada nesta dissertação como algoritmo de previsão inerente à *Sequential Feature Selection*, como se descreverá na secção 2.3.1.

O algoritmo MARS combate este problema uma vez que procura agregar diferentes equações lineares podendo, portanto, ser visto como um tipo de algoritmo *ensemble*.

Uma das desvantagens deste algoritmo é a sua difícil interpretação, no entanto o seu princípio de funcionamento pode ser resumido essencialmente em duas fases, a fase *forward* e a fase *backward*.

[10] "However, the interaction between metrics in the real-world is often non-linear, which means that simple linear regression cannot always give us a good approximation of outputs given the inputs."[71]

Na fase *forward* o intervalo de valores do modelo de previsão é dividido em vários grupos[11] separados por nós, criando o algoritmo duas funções de base para cada nó, as quais relacionam a variável a prever com as entradas. A Figura 2-7 ilustra este processo.

Na fase *backward* as diferentes funções criadas são percorridas, sendo eliminadas aquelas que não contribuem para uma melhoria na performance do modelo.

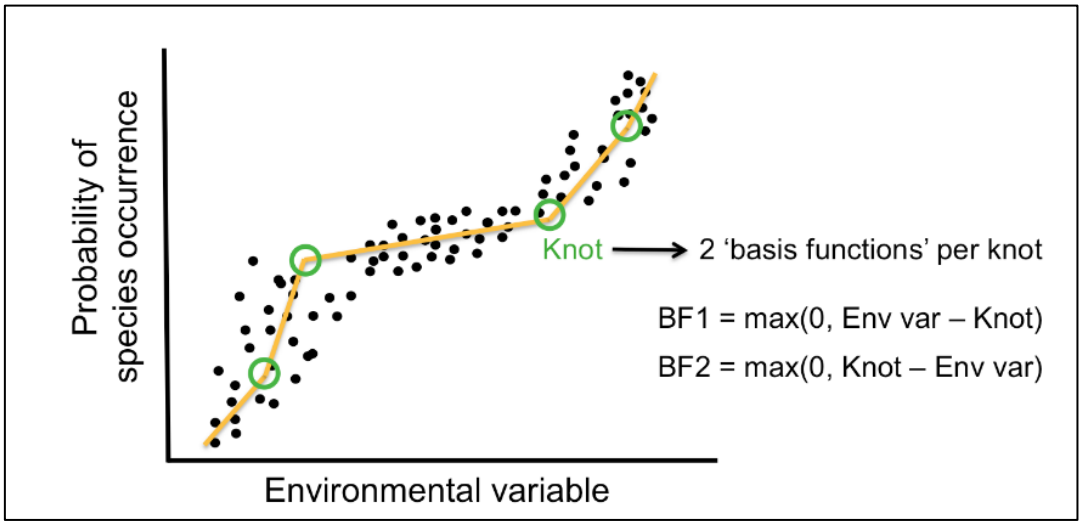


Figura 2-7-Exemplo de ilustração do algoritmo MARS [22]

Optou-se pela utilização deste algoritmo pois o mesmo deteta automaticamente relações entre as variáveis, ou seja, pode ser usado como um método de *feature selection.* Este algoritmo foi implementado com recurso ao modelo "Earth" da biblioteca *pyearth* e a sua documentação pode ser consultada em [23]. No que respeita a este algoritmo, os parâmetros que se otimizaram são os expostos na Tabela 2-4.

Tabela 2-4-Parâmetros do algoritmo MARS considerados no processo de otimização (baseada na documentação [23])

Nome na biblioteca pyearth	Significado	Valor/Tipo de Variável
max_terms	Número máximo de termos gerado no processo *forward*	Int
max_degree	Grau máximo dos termos gerado no processo *forward*	Int
penalty	Parâmetro de suavização (equação 32 em [21])	Float

[11] "In this procedure, the range of predictor values is partitioned in several groups"[22]

2.2.Algoritmos de *dimensionality reduction*

2.2.1. *Principal Component Analysis* (PCA)

O algoritmo *Principal Component Analysis* (PCA) é um dos algoritmos mais utilizados na redução da dimensionalidade de grandes *datasets*, procurando aumentar a facilidade de compreensão dos problemas e ao mesmo tempo evitar a perda de informação. Para isso, o PCA cria novas variáveis não correlacionadas que levem a um aumento da variância [12]. Este algoritmo compreende cinco fases [24]:

- normalização das variáveis;
- cálculo da matriz de covariância;
- cálculo dos valores e vetores próprios da matriz de covariância, de maneira a identificar os componentes principais, ou seja, os componentes que mais contribuem para a variância;
- construção do vetor de *features*, o qual contém os valores próprios calculados contendo informação sobre a importância de cada componente. Com base no número de componentes *n* desejado pelo utilizador, são mantidas as *n* componentes principais.
- reorientar os dados dos eixos originais para os eixos definidos pelas componentes principais.

Este algoritmo foi implementado com as bibliotecas *scikit-learn* e *pca* e as suas documentações podem ser consultadas em [25] e [26], respetivamente.

2.3. Algoritmos de *feature selection*

2.3.1. *Sequencial Feature Selection* (*forward* e *backward*)

A *sequential feature selection* consiste em, tal como o nome indica, um processo sequencial de seleção de variáveis, sendo que existem dois tipos de implementação deste algoritmo: a *forward* e a *backward*.

[12] "…Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. It does so by creating new uncorrelated variables that successively maximize variance."[72]

No caso do tipo *forward* começa-se com um conjunto vazio de variáveis, sendo adicionada ao subconjunto a variável que mais contribuir para a função objetiva. No segundo passo em diante as restantes variáveis são adicionadas individualmente ao subconjunto atual e o novo subconjunto é avaliado, ficando a variável permanentemente no subconjunto caso leve a uma métrica máxima. Este processo é repetido até se atingir o número exigido de *features*.[13]

No caso do tipo backward, este funciona com um princípio semelhante mas inverso, isto é, o algoritmo inicia com um conjunto com todas as variáveis, sendo de seguida removida uma de cada vez aquela cuja exclusão levar a uma diminuição menor de desempenho [14].
Na Figura 2-8 , retirada da bibliografia, é mostrada uma ilustração que exemplifica bem o funcionamento da *Sequential Backward Selection*.

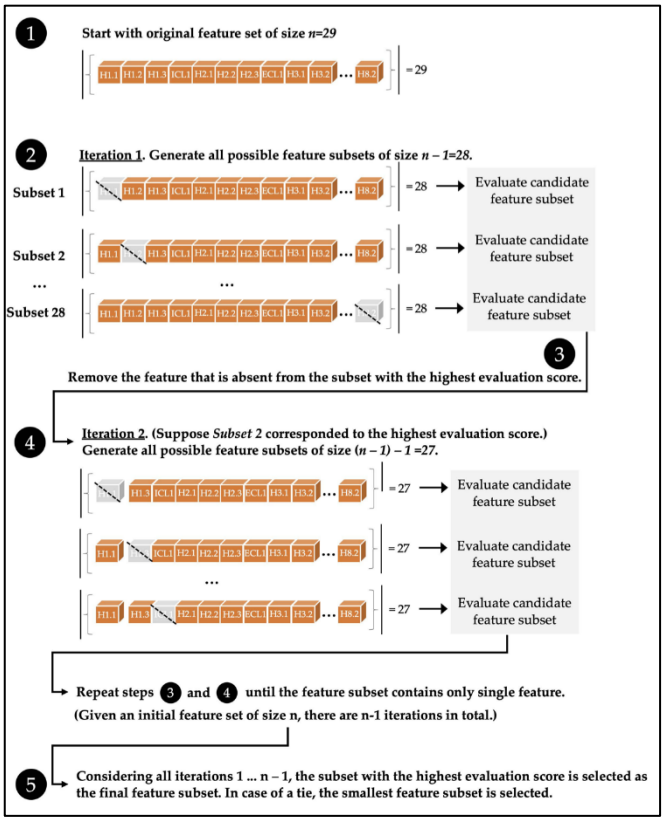


Figura 2-8- Exemplo de ilustração do algoritmo Sequential Backward Selection [27]

[13] "The Sequential Feature Selection (SFS) algorithm [33,34] starts with an empty set and adds one feature for the first step which gives the highest value for the objective function. From the second step onwards the remaining features are added individually to the current subset and the new subset is evaluated. The individual feature is permanently included in the subset if it gives the maximum classification accuracy. The process is repeated until the required number of features are added." [73]

[14] "A Sequential Backward Selection (SBS) algorithm can also be constructed which is similar to SFS but the algorithm starts from the complete set of variables and removes one feature at a time whose removal gives the lowest decrease in predictor performance." [73]

O algoritmo *Sequential Feature Selection* foi implementado nas suas duas vertentes com o modelo "*SequentialFeatureSelector*" da biblioteca *mlxtend* e a sua documentação pode ser consultada em [28].

2.4.Algoritmos de otimização

2.4.1. *Grid Search* e *Random Search*

Uma parte importante no desenvolvimento de modelos de previsão é a afinação ou *tuning* dos parâmetros dos modelos. Os algoritmos mais comuns para este fim são a *Grid Search* e a *Random Search*. O funcionamento destes algoritmos é bastante trivial, sendo que em ambos o utilizador necessita de fornecer uma grelha contendo os intervalos de valores para os parâmetros que pretende otimizar. O algoritmo *Grid Search* limita-se a testar todas as combinações possíveis, enquanto o *Random Search* executa um número aleatório de combinações, sendo o número de iterações definido pelo utilizador. No final, ambos os algoritmos devolvem a combinação de valores que levou à melhor performance. O funcionamento de ambos os algoritmos é ilustrado na Figura 2-9, adaptada da bibliografia.

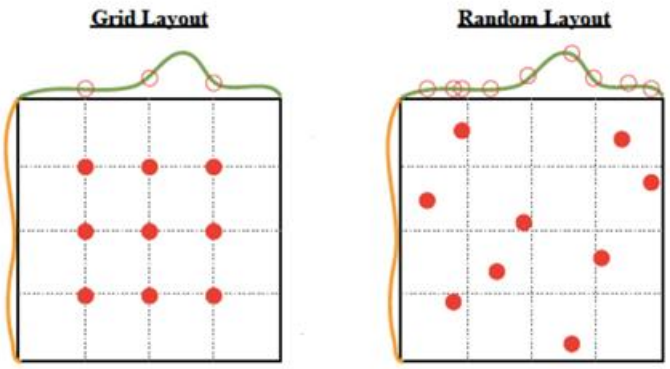


Figura 2-9-Ilustração dos algoritmos Grid Search e Random Search (Adaptado de [29])

A *Random Search* possui naturalmente a vantagem de levar a tempos de processamento bastante menores uma vez que não são percorridas todas as combinações possíveis no entanto possui a desvantagem de não garantir que a melhor solução seja encontrada.

A *Random Search* é também mais eficiente porque nem todos os parâmetros possuem a mesma relevância, sendo que a *Grid Search* dedica muitas tentativas na exploração de dimensões que não importam e sofre de pouca cobertura em dimensões mais importantes[15].

Estes algoritmos foram implementados com a biblioteca *scikit-learn* e as documentações podem ser consultadas em [30] e [31].

[15] "Our analysis of the hyper-parameter response surface (Y) suggests that random experiments are more efficient because not all hyperparameters are equally important to tune. Grid search experiments allocate too many trials to the exploration of dimensions that do not matter and suffer from poor coverage in dimensions that are important."[74]

3. Descrição e divisão dos dados

Os dados utilizados nesta dissertação são essencialmente os que farão parte das entradas/*features* dos modelos de previsão, as quais podem ser divididas em variáveis endógenas e exógenas. As variáveis endógenas em causa são os dados relativos à potência elétrica ativa, variável que se pretende prever, sendo que o histórico de valores destas variáveis constituem geralmente as *features* mais relevantes nos modelos de previsão, relevância essa que merece ser avaliada para todas as variáveis, na fase de *feature selection*. Quanto às variáveis exógenas são variáveis que, sendo diferentes da variável que se pretende estimar, prevê-se que tenham influência na saída (potência ativa), como é o caso de variáveis climáticas ou relacionadas com perfis de ocupação e horários dos espaços.

3.1. Variáveis endógenas/dados de consumo

Os dados de consumo disponíveis no âmbito desta dissertação dizem respeito ao Campus 2 do Instituto Politécnico de Leiria, no período compreendido entre finais de outubro de 2015 e março de 2021, contendo os registos de potência ativa (kW), reativa indutiva (kvar) e reativa capacitiva (kvar) em intervalos de 15 minutos, bem como a data e hora referente aos mesmos registos.

"Na Tabela 1 é apresentado um resumo do conjunto dos dados disponível, antevendo-se que relativamente aos anos de 2016 e de 2018 será necessário algum tipo de abordagem para lidar com os dados em falta. Por fim são também apresentadas as datas das mudanças de hora em cada ano, uma vez que tal acontecimento tem impacto na série temporal e irá obrigar também a um tratamento adicional dos dados.

Tabela 3-1-Tabela resumo dos dados em bruto da potência ativa

Ano	1º Registo	Último Registo	Registos esperados	Nº de Registos (dados em bruto)	Dados em falta	Mudança de Hora (Março)	Mudança de Hora (Outubro)
2015	27/10/2015 11h:30 [16]	31/12/2015	6291	6291	0	29 de Março (sem dados)	25 de Outubro (sem dados)
2016	01/01/2016	31/12/2016	35136	35118	18	27 de Março	30 de Outubro
2017	01/01/2017	31/12/2017	35040	35040	0	26 de Março	29 de Outubro
2018	01/01/2018	31/12/2018	35040	35036	4	25 de Março	28 de Outubro
2019	01/01/2019	31/12/2019	35040	35040	0	31 de Março	27 de Outubro
2020	01/01/2020	31/12/2020	35136	35136	0	29 de Março	25 de Outubro
2021	01/01/2021	28/02/2021	5664	5664	0	28 de Março (sem dados)	31 de Outubro (sem dados)

Um ponto importante no conhecimento da série temporal é o perfil de consumo do Campus, o qual foi fortemente influenciado pela pandemia de covid-19. Na Figura 3-1 são apresentados os diagramas de carga em 2019 para quatro semanas em tempo de aulas em maio e novembro e em semanas com regimes letivos diferentes.

[16] Registo Incompleto.

Como se pode ver analisando o regime presencial, é facilmente percetível o efeito dos fins-de-semana, onde a atividade no campus é muito mais reduzida e consequentemente o consumo. Já nos dias de semana conseguem-se identificar os períodos de maior atividade no campus, que coincidem naturalmente com as atividade letivas, que começam às 8h e se prolongam até às 24h, tendo, no entanto, maior atividade no regime diurno, o qual decorre até às 20h. É também possível visualizar que os valores máximos de potência se situam entre os 600 e 700 kW, e que a base dos diagramas se situa geralmente perto dos 200 kW, sendo um pouco menor nos domingos, o que se deve à inexistência de atividade no Campus neste dia da semana. Outro ponto interessante, é o facto dos valores máximos e de base dos diagramas não diferirem muito de maio a novembro. Este facto pode também comprovar-se ao consultar os consumos mensais apresentados na Tabela 3-2.

Tabela 3-2-Consumos mensais de energia ativa [MWh]

	2016	2017	2018	2019	2020	2021
Janeiro	244,07	273,30	238,28	268,36	232,58	194,21
Fevereiro	222,85	223,31	219,93	213,76	191,67	98,07
Março	245,88	254,35	254,89	222,22	174,24	-
Abril	240,30	210,99	224,64	198,41	125,66	-
Maio	228,96	242,97	207,37	235,13	114,29	-
Junho	236,06	243,48	202,85	221,53	107,15	-
Julho	255,39	219,80	201,45	214,70	127,38	-
Agosto	202,93	192,86	175,92	179,77	128,86	-
Setembro	239,96	215,93	218,32	223,77	162,56	-
Outubro	241,67	243,04	251,84	232,76	179,90	-
Novembro	253,07	235,01	246,20	229,11	183,44	-
Dezembro	235,66	215,91	220,83	223,04	177,96	-

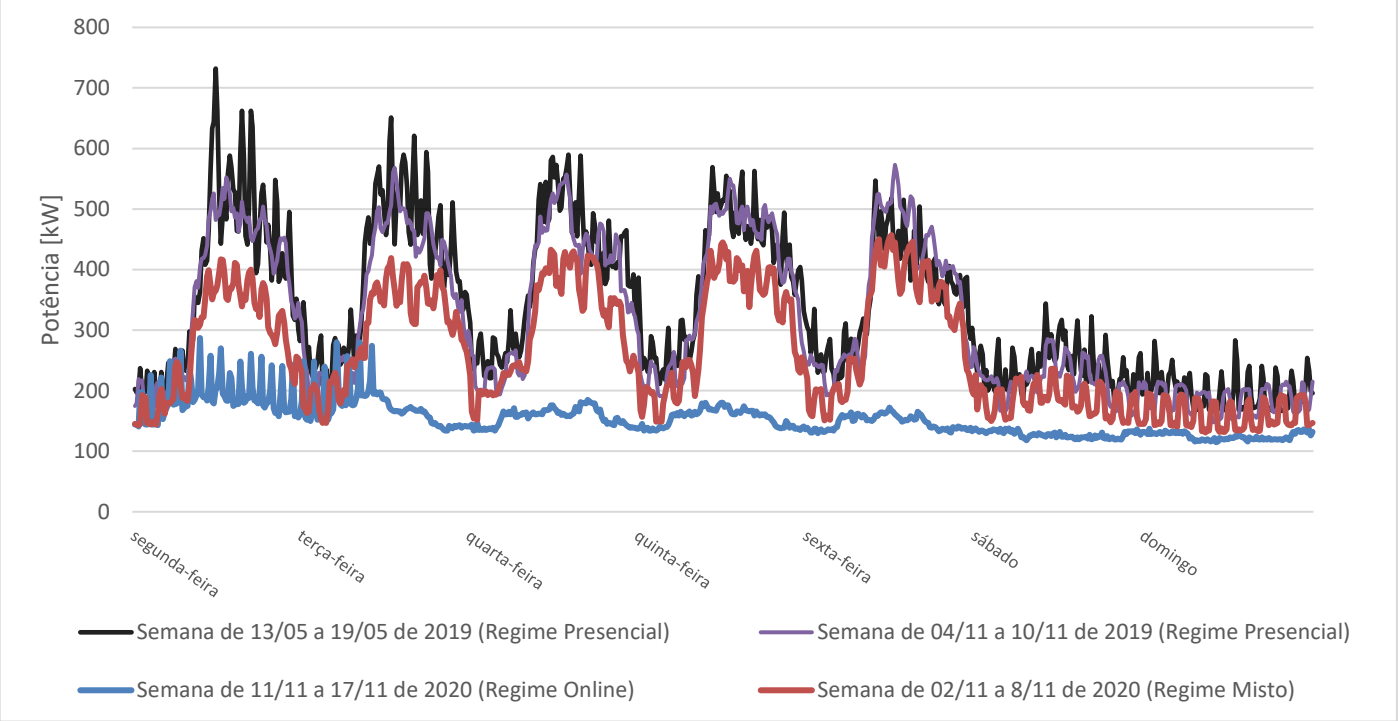


Figura 3-1-Diagramas de carga em diferentes estações do ano e com diferentes regimes letivos

3.1.1. Impacto da pandemia Covid-19 no perfil de consumo

Para se perceber o quanto o consumo tem sido influenciado pela pandemia Covid-19 e pelas medidas restritivas associadas à mesma, podem comparar-se os gráficos da Figura 3-1 em regime presencial com os gráficos da mesma figura, mas referentes ao período em que vigorou o regime online e misto. Como se pode ver, quando todas as atividades letivas ocorreram em regime online, como é o caso da semana apresentada de maio de 2020 o consumo foi muito mais constante e baixo, com valor máximos inferiores a 200 kW, valor que antes correspondia à base do diagrama. Também na semana de novembro apresentada, onde as atividades letivas se repartiram pelos regimes online e presencial (regime misto), o consumo foi mais baixo, com valores máximos na ordem dos 450 kW. Estas diferenças também são visíveis nos consumos mensais da Tabela 3-2, onde se apresentam a sombreado amarelo os períodos durante o período de confinamento, no qual por exemplo o consumo em maio de 2020 corresponde a cerca de 48,6% do consumo em 2019 no mesmo mês. Por fim, o efeito da pandemia é bastante evidente também na Figura 3-2, onde são apresentados

os consumos diários de energia e a altura em que o regime online entrou em vigor, verificando-se de seguida uma queda abrupta nos consumos.

Esta grande diferença obrigou a ter cuidados ao longo do desenvolvimento do trabalho, pois os perfis e amplitudes do consumo antes e durante a pandemia são completamente diferentes.

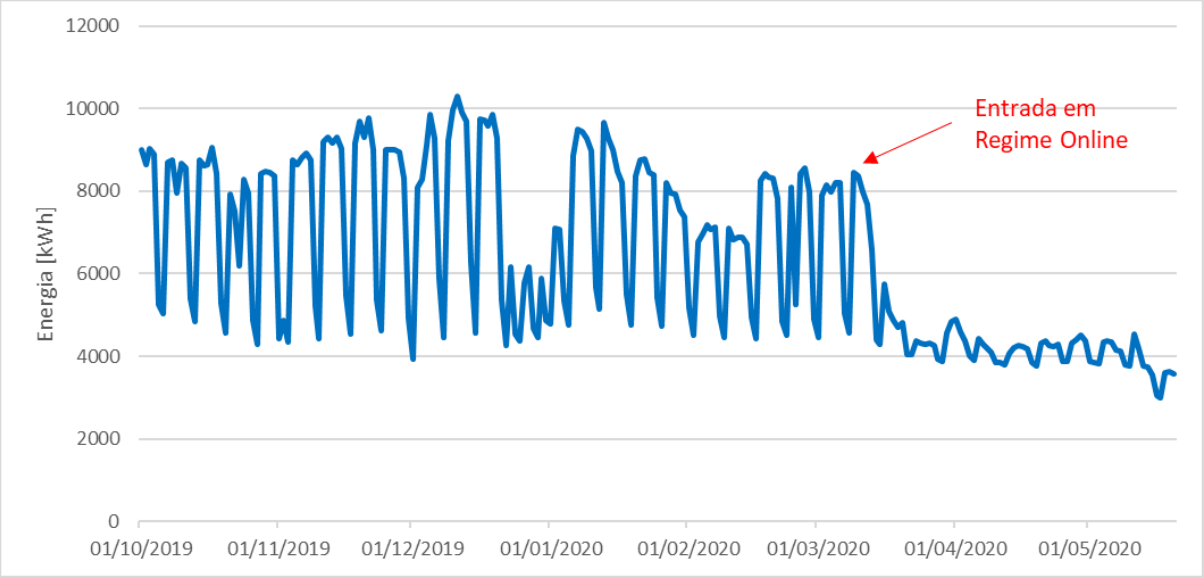


Figura 3-2-Efeito da pandemia Covid-19 no consumo diário de energia ativa

3.2.Variáveis exógenas

As variáveis exógenas analisadas encontram-se descritas na Tabela 2. Todas as variáveis climáticas, com exceção dos graus dia de aquecimento e arrefecimento, foram obtidos no site *WeatherUnderground* [32], sendo provenientes de três estações meteorológicas perto do Campus 2 do Instituto Politécnico de Leiria, mais concretamente nos Parceiros, Leiria e Barosa. A utilização de dados provenientes de três estações diferentes permitiu garantir a não existência de valores em falta no conjunto de dados destas variáveis, uma vez que, para cada uma das estações, havia períodos sem dados disponíveis. Os registos referentes aos graus dia foram retirados do site da Eurostat [33]. Considerou-se também os registos da potência reativa com um *lag* de uma semana como *feature*[17], de maneira a tentar aproveitar os dados relacionados com o comportamento desta variável.

[17] Uma vez que a variável a prever é a potência ativa, considerou-se a potência reativa como variável exógena.

Tabela 3-3-Variáveis exógenas utilizadas

Variável	Tipo	Unidade	Valor Máximo	Valor Mínimo	Resolução
Temperatura Máxima	Float	°C	47,22	8,50	Horária
Temperatura Média	Float	°C	35,11	2,17	
Temperatura Mínima	Float	°C	23,72	-3,89	
Temperatura de Ponto de Orvalho Máxima	Float	°C	25,5	-6,89	
Temperatura de Ponto de Orvalho Média	Float	°C	20,94	-11	
Temperatura de Ponto de Orvalho Mínima	Float	°C	18,72	-23,33	
Humidade Máxima	Float	%	0,99	0,19	
Humidade Média	Float	%	0,98	0,14	
Humidade Mínima	Float	%	0,98	0	
Velocidade do Vento Máxima	Float	m/s	18,69	0	
Velocidade do Vento Média	Float	m/s	4,56	0	
Velocidade do Vento Mínima	Float	m/s	2,24	0	
Pressão Atmosférica Máxima	Float	m	105,64	70,43	
Pressão Atmosférica Mínima	Float	m	105,64	70,43	
Precipitação Acumulada	Float	m	779,78	0	
Tipo de Dia	Int	-	1	8	
Dia	Int	-	1	7	
Função Seno	Float	-	0,98	-0,98	
Função Cosseno	Float	-	1	-0,90	
Graus dia de Aquecimento	Float	-	259,05	0	Mensal
Graus dia de Arrefecimento	Float	-	108,61	0	
Minuto	Int	-	1	96	Quarta-Horária
Reativa	Float	kvar	404	12	

Algo característico dos registos de potência elétrica ativa no Campus é a periodicidade dos mesmos.

Em [34] criaram-se duas variáveis cíclicas com vista a passar esta informação para os modelos. Essas variáveis são calculadas como apresentado na (Equação 3.1) e na (Equação 3.2), onde "d" representa um valor de 1 a 7 consoante o dia da semana, e ajudaram a atingir boas métricas de erro nesse caso de estudo, pelo que se tentou obter o mesmo efeito na previsão dos consumos do Campus 2 do Politécnico de Leiria, tendo-se chamado a essas duas variáveis "Função Seno" e "Função Cosseno".

$$Função\ Seno = sen(\frac{2\pi \times d}{7})$$

(Equação 3.1)

$$Função\ Cosseno = cos(\frac{2\pi \times d}{7})$$

(Equação 3.2)

Outra variável que se criou para tentar replicar esse efeito de periodicidade foi a variável "Dia", que consiste apenas numa variável cíclica com um valor de 1 a 7, consoante o dia da semana.

As variáveis anteriores apenas acautelam a periodicidade semanal do consumo, no entanto este comportamento periódico é afetado por outros fatores, pois por exemplo numa segunda-feira com atividades letivas atingem-se consumos muito diferentes de uma segunda-feira de feriado, onde o perfil de consumo passa então a ser mais semelhante ao de um domingo, altura em que também não existem atividades letivas. Assim sendo, criou-se mais uma variável com vista a ter em conta o comportamento cíclico do consumo, não apenas em função do dia da semana, mas também da época do ano letivo (época letiva, de exames, de férias de natal,…), tendo-se para isso recorrido aos calendários de avaliação da Escola Superior de Tecnologia e Gestão e à Aplicação de Gestão Científica e Pedagógica disponível no site AGCP [35], onde se podem consultar horários e datas de avaliações de anos anteriores. Na Tabela 3-4 são apresentados os diferentes valores que a variável "Tipo de Dia" pode tomar.

Tabela 3-4-Valores tomados pela variável "Tipo de Dia"

Valor	Descrição
1	Dias de aulas
2	Sábados
3	Domingos e feriados
4	Épocas de exames
5	Férias de Natal
6	Férias da Páscoa
7	Férias de Verão
8	Outras interrupções

Por fim criou-se uma última variável cíclica denominada “Minuto” com o intuito de obter uma variável com a mesma resolução da potência que se pretende prever e que consiste apenas num valor de 1 a 96 representando os 96 intervalos de 15 minutos que existem num dia.

3.3. Divisão dos dados e estratégia de validação adotada

Um ponto importante no desenvolvimento de um modelo de *machine learning* é a divisão dos dados. Embora por vezes se considerem apenas dois períodos, denominados por período de treino e período de teste, algo que pode ajudar os modelos a generalizar melhor por evitar problemas de *overfit*, ou seja a manter um bom desempenho quando os modelos são confrontados com novos dados, é a utilização de um conjunto de dados de validação. A ideia passa por treinar os modelos no conjunto de treino, validar/melhorar os modelos através do conjunto de validação e testar apenas os modelos finais no conjunto de teste. Assim sendo os testes que envolvem a avaliação de desempenhos com diferentes parametrizações ou diferentes *features* deverão ser feitos no subconjunto de validação e apenas os modelos finais com cada algoritmo deverão ser testados no subconjunto de teste, permitindo assim efetuar um teste final que seja independente, uma vez que os dados de teste não foram utilizados nos processos de treino e de validação.

Esta abordagem pretende então evitar o problema de *data leakage*, o qual acontece quando o modelo teve já contato com alguma parte dos dados de teste depois do processo de treino[18]. Em relação à estratégia utilizada na separação em conjunto de treino e de validação existem várias técnicas, sendo as mais comuns as da validação *Hold-out* e a validação cruzada com as técnicas *K-fold* ou *Leave-One-Out*.

Na técnica de validação *Hold-out* os dados são simplesmente divididos manualmente em duas partes, sendo que numa é efetuado o treino e noutra o modelo treinado é validado[19].

Na técnica *K-fold* os dados de treino são separados em K conjuntos ou *folds* iguais, sendo o treino feito nos K-1 *folds* e a validação feita no último. O processo anterior é repetido K vezes usando como conjunto de validação cada um dos K conjuntos até a validação ter sido feita em todos eles [20], como ilustrado na Figura 3-3. No caso particular de K ser igual ao número de amostras a técnica denomina-se de validação cruzada *Leave-One-Out*.

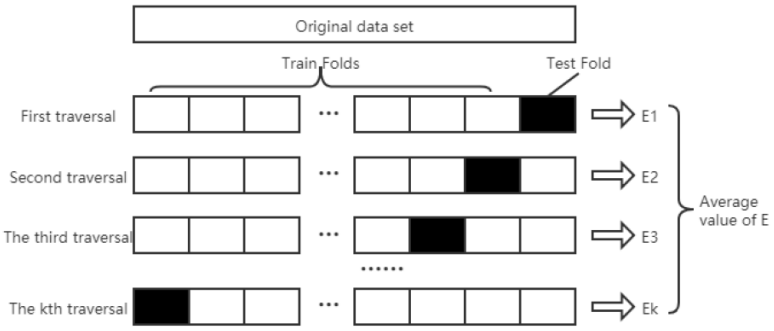


Figura 3-3-Validação cruzada K-Fold [36]

Na validação *Hold-Out* existe a grande desvantagem de não ser feito um uso eficiente dos dados[21], uma vez que o conjunto de validação é fixo, o que torna as medições de desempenho mais propícias a problemas de *overfit*. Com a validação cruzada consegue-se evitar melhor o impacto direto da divisão dos dados nos resultados[22], já que são testadas diferentes partes dos dados no processo de validação, sendo estes também aproveitados para treinar o modelo, o que não acontece no método *Hold-Out*.

[18] "Data Leakage is the scenario where the Machine Learning Model is already aware of some part of test data after training".[75]
[19] "…in hold-out validation, we split the data into two parts, training on one while testing on other with the trained model."[76]
[20] "…data is split into k-equal parts. Training of the model is done on k-1 parts and one part is left out for testing. The process does not end here. The above is repeated k times while changing the test part one-by-one until testing has been done on all the k parts."[76]
[21] "The holdout method makes inefficient use of the data…"[77]
[22] "The k-fold cross validation was used to avoid the direct impact of data set partition on the results".[36]

A validação cruzada *Leave-One-Out* possui como grande desvantagem uma exigência computacional elevadíssima, razão pela qual é geralmente apenas utilizada em *datasets* pequenos, o que invalida o seu uso nesta dissertação devido ao elevado número de registos, como se pode constatar através de consulta à Tabela 3-1.

Optou-se então pela utilização do método *k-fold*, uma vez que dos três métodos mencionados é o que permite um melhor compromisso entre um uso eficiente dos dados disponíveis e um esforço computacional aceitável, o qual será naturalmente proporcional ao valor definido para o parâmetro K. De facto, na bibliografia, a utilização do método *K-fold* aparenta ser o mais popular, tendo por exemplo em [36] sido testados vários métodos de validação e foi o *K-Fold* que apresentou melhores resultados[23].

Em relação aos períodos de treino e de teste, a divisão foi feita de acordo com o apresentado na Tabela 3-5 e na Tabela 3-6.

Tabela 3-5- Divisão em períodos de treino e de teste (1)

Features		Dados de Treino																																			
11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2
2015		2016												2017												2018												2019	

Tabela 3-6-Divisão em períodos de treino e de teste (2)

Dados de Teste											
3	4	5	6	7	8	9	10	11	12	1	2
2019										2020	

Apesar de se possuírem sensivelmente 2 meses de dados de 2015, optou-se por não se incluir estes dados no período de treino de maneira a garantir que se possui histórico de dados para as primeiras semanas do período de treino. Assim sendo, dependendo do atraso de valores históricos que seja utilizado, os valores de 2015 serão apenas usados como *features* dos modelos e não como valores alvo a prever (saídas dos modelos). Quanto à definição do mês de fevereiro de 2020 como mês final do período de teste, deve-se ao facto do mês de março deste mesmo ano corresponder ao momento em que foi decretado em Portugal o fecho de estabelecimentos de Ensino devido à pandemia de Covid-19. Devido às substanciais diferenças entre os perfis de consumo antes e durante a pandemia, tomou-se a decisão de não incluir estes dados no período de teste.

[23] "The RMSE values of the four algorithms under k-fold cross validation method are almost unchanged, and the results are very stable and objective, which reflects the advantages of kfold cross validation method. Compared with the two methods, k-fold cross validation is better."[36]

Estes dados serão então utilizados posteriormente para fazer uma pequena análise do efeito da pandemia no consumo de potência ativa. Como se pretende aferir o desempenho dos modelos durante os diferentes períodos do ano, avaliando por exemplo se o modelo consegue efetuar boas previsões fora do período de aulas, considerou-se um ano como período de teste, totalizando então aproximadamente 25% dos dados como dados de teste e os restantes como dados de treino.

Tal como no período de teste é também importante garantir que os subconjuntos do processo de validação cruzada permitam avaliar os modelos durante os diferentes períodos ao longo do ano. Assim sendo decidiu-se usar um valor de 3 para o parâmetro K, fazendo com que cada *fold* do processo de validação cruzada tenha 385 dias.

4. Pré-processamento dos dados

A qualidade dos resultados dos modelos de previsão está fortemente ligada à qualidade dos dados utilizados. Tipicamente os dados em bruto possuem bastantes problemas como a existência de dados atípicos ou dados em falta. É então essencial tratar estes e outros problemas no conjunto de dados antes de proceder à aplicação e treino de modelos.

4.1. Mudanças de hora

Um aspeto bastante importante no que respeita à preparação da série temporal dos dados de consumo e, pouco explorado na bibliografia consultada, apesar de ser um procedimento comum, é a questão das mudanças de hora, que poderá ter diferentes implicações no conjunto de dados. Geralmente, e também no presente trabalho, as mudanças de hora traduzem-se em: horas em falta, no caso da mudança de hora de Inverno para hora de Verão, a qual ocorre na última semana de março; horas duplicadas, no caso da mudança de hora de Verão para hora de Inverno, a qual ocorre na última semana de outubro. Assim sendo, estas mudanças irão fazer com que um dia de março seja representado como tendo 23 horas e com que um dia de outubro seja representado como tendo 25 horas, o que traz problemas, por exemplo quando se pretendem visualizar os dados num gráfico como o diagrama de carga (que representa a potência ativa em função do tempo). Estes efeitos podem ser vistos na Tabela 4-1 e na Tabela 4-2 , onde são apresentados os dados em bruto do ano 2016 do Campus 2, na altura em que ocorrem as mudanças de hora.

Tabela 4-1-Mudança de hora em março de 2016(Campus 2)

Data e hora	P[24](kW)	Qi[25] (kvar)	Qc[26] (kvar)
27/03/2016 00:30	167	61	0
27/03/2016 00:45	165	60	0
27/03/2016 02:00	164	64	0
27/03/2016 02:15	169	51	0

[24] Potência ativa.
[25] Potência reativa indutiva.
[26] Potência reativa capacitiva.

Tabela 4-2-Mudança de Hora em outubro de 2016(Campus 2)

Data e hora	P (kW)	Qi (kvar)	Qc (kvar)
30/10/2016 00:30	197	75	0
30/10/2016 00:45	196	71	0
30/10/2016 01:00	197	75	0
30/10/2016 01:15	224	118	0
30/10/2016 01:30	205	85	0
30/10/2016 01:45	207	85	0
30/10/2016 01:00	347	188	0
30/10/2016 01:15	304	180	0
30/10/2016 01:30	228	118	0
30/10/2016 01:45	200	77	0
30/10/2016 02:00	195	70	0
30/10/2016 02:15	192	70	0
30/10/2016 02:30	212	103	0

Existem várias possibilidades no que toca à abordagem a ser seguida para lidar com este problema. Na referência [37], por exemplo, optou-se por substituir os valores em falta resultantes da mudança de hora em março por uma média de valores anteriores e em descartar a hora duplicada no caso da mudança em outubro, com o argumento de que neste caso os consumidores encaram a primeira hora como fazem habitualmente e a hora duplicada como hora de sono extra.

Outra estratégia, usada em [38], consiste em utilizar duas colunas de dados, uma com os dados do fuso horário local, nesse caso o do Reino Unido, o qual está sujeito a mudanças de hora, e outra com o tempo universal coordenado (UTC), não sujeito a essa alteração. Esta abordagem teria a vantagem de se poderem utilizar os dados originais, sem adição por exemplo de valores provenientes de operações de média.

Outra possibilidade seria realizar um deslocamento (ou *shift*) nos dados, transformando a Tabela 4-1 na Tabela 4-3 e a Tabela 4-2 na Tabela 4-4, o que teria também a vantagem já descrita anteriormente, e ao mesmo tempo possuir dados para todos os intervalos de 15 minutos.

Tabela 4-3-Mudança de hora em março de 2016(Campus 2) com shift

Data e hora	P (kW)	Qi (kvar)	Qc (kvar)
27/03/2016 00:30	167	61	0
27/03/2016 00:45	165	60	0
27/03/2016 01:00	164	64	0
27/03/2016 01:15	169	51	0

Tabela 4-4-Mudança de hora em outubro de 2016(Campus 2) com shift

Data e hora	P (kW)	Qi (kvar)	Qc (kvar)
30/10/2016 00:30	197	75	0
30/10/2016 00:45	196	71	0
30/10/2016 01:00	197	75	0
30/10/2016 01:15	224	118	0
30/10/2016 01:30	205	85	0
30/10/2016 01:45	207	85	0
30/10/2016 02:00	347	188	0
30/10/2016 02:15	304	180	0
30/10/2016 02:30	228	118	0
30/10/2016 02:45	200	77	0
30/10/2016 03:00	195	70	0
30/10/2016 03:15	192	70	0
30/10/2016 03:30	212	103	0

Quanto à possibilidade de utilizar duas colunas de dados, uma com o fuso horário local não alterado, e outra com o UTC, tomou-se a decisão de não recorrer a tal abordagem, pois a dimensão do problema não justifica a criação de mais uma coluna de dados, uma vez que as mudanças de hora ocorrem apenas em 2 horas num universo de 8760 horas anuais.

Em relação à opção do *shift* dos dados, optou-se por não recorrer a esta estratégia por esta levar a uma perda de informação no que respeita à caracterização dos perfis de consumo, pois eventos importantes (valor máximos de consumo, quedas abruptas de consumo devido ao fecho de edifícios,…) virem a ser representados como acontecendo 1 hora mais cedo, no caso do período onde vigora a hora de Verão, ou 1 hora mais tarde, no caso do período onde vigora a hora de Inverno.

Assim sendo decidiu-se optar por uma estratégia semelhante à utilizada em [37], no entanto com algumas diferenças. No caso da mudança em março, utilizou-se em vez da média uma interpolação linear, uma vez que consiste num método numérico, aproximando-se mais a um algoritmo de previsão uma vez que assume uma certa tendência dos dados. Tendo em conta que os dados são sucessivos, o termo y1 corresponde a um valor ele próprio resultante de interpolação, com exceção da primeira linha a calcular. Essa interpolação foi feita com recurso ao valor anterior e ao valor das 2h:00 para preencher os valores referentes às horas em falta, chegando-se à Tabela 4-5, onde é apresentada a tabela já corrigida.

Na (Equação 4.1) pode ser visto o cálculo do primeiro valor em falta, ou seja, o registo para a 1h:00, sendo:

-y_2 o valor da potência ativa consumida às 2h:00;

- y_1 o valor da potência ativa consumida às 00h:45;

- x_2 o valor do índice do registo às 02h:00;

- x_1 o valor do índice do registo às 00h:45;

- x_0 o valor d0 índice do registo à 01h:00;

$$y = y1 + (x - x1)\frac{(y2 - y1)}{(x2 - x1)} = 165 + (8260 - 8259) \times \frac{(164 - 165)}{(8264 - 8259)} = 164{,}8\ kW$$

(Equação 4.1)

Tabela 4-5-Mudança de hora em março de 2016(Campus 2) corrigida

Data	Hora	P (kW)	Qi (kvar)	Qc (kvar)
27/03/2016	00:45	165	60	0
27/03/2016	01:00	164,80	60,80	0
27/03/2016	01:15	164,60	61,60	0
27/03/2016	01:30	164,40	62,40	0
27/03/2016	01:45	164,20	63,20	0
27/03/2016	02:00	164	64	0

No que toca à mudança de hora em outubro onde há horas duplicadas, em vez de se eliminar o segundo valor como sugerido em [37], optou-se por fazer uma média dos dois valores, contribuindo assim para aproveitar melhor a informação disponível, uma vez que o resultado corrigido terá um impacto de ambos os valores. Partindo da Tabela 4-2 e aplicando o raciocínio explicado anteriormente chega-se à Tabela 4-6. A título de exemplo, é apresentado na (Equação 4.2) o cálculo do primeiro valor corrigido, referente ao intervalo da 1h:00, onde:

-y representa o valor da potência ativa consumida corrigido à 1h:00 a calcular;

- y_1 representa o primeiro valor da potência ativa consumida à 1h:00;

-y_2 representa o segundo valor da potência ativa consumida à 2h:00;

$$y = \frac{y1 + y2}{2} = \frac{197 + 347}{2} = 272\ kW$$

(Equação 4.2)

Tabela 4-6-Mudança de hora em Outubro de 2016(Campus 2) corrigida

Data	Hora	P (kW)	Qi (kvar)	Qc (kvar)
30/10/2016	00:45	196	71	0
30/10/2016	01:00	272	131,5	0
30/10/2016	01:15	264	149	0
30/10/2016	01:30	216,5	101,5	0
30/10/2016	01:45	203,5	81	0
30/10/2016	02:00	195	70	0

4.2.Valores em falta

É comum existirem valores em falta nas medições de consumo utilizadas. Por exemplo, a informação recolhida por sensores possui tipicamente ruído e pode estar sujeita a interferência, contribuindo então para um *dataset* incompleto. Eventuais falhas de energia ou falhas de equipamentos de medição podem também levar à ocorrência deste problema. No caso do presente trabalho, como já foi descrito este problema no que respeita às *features* climáticas solucionou-se utilizando dados de estações meteorológicas próximas sempre que se verificavam registos em falta. Por fim, as *features* cíclicas foram variáveis que foram construídas ou calculadas, pelo que este problema não se coloca. Assim sendo a questão de registos em falta só tem de ser analisada na série temporal do histórico dos consumos. Como se pode ver na Tabela 4-7, existem um total de 22 registos em falta, 18 deles em 2016 e 4 deles em 2018 o que, considerando todos os dados disponíveis (de outubro de 2015 a março 2021), corresponde a cerca de 0,0117% do conjunto total de dados e a 0,0167% do conjunto de dados de treino, sendo a percentagem menor devido ao facto dos dados em falta pertencerem todos ao conjunto de treino. Quer na série de 2016, quer na de 2018, os dados em falta dizem respeito a um único período, ou seja, ocorrem sucessivamente.

Tabela 4-7-Registos em falta no histórico de potência ativa

2016	2018
20/05/2016 01:30	13/09/2018 06:45
20/05/2016 01:45	13/09/2018 07:00
20/05/2016 02:00	13/09/2018 07:15
20/05/2016 02:15	13/09/2018 07:30
20/05/2016 02:30	
20/05/2016 02:45	
20/05/2016 03:00	
20/05/2016 03:15	
20/05/2016 03:30	
20/05/2016 03:45	
20/05/2016 04:00	
20/05/2016 04:15	
20/05/2016 04:30	
20/05/2016 04:45	
20/05/2016 05:00	
20/05/2016 05:15	
20/05/2016 05:30	
20/05/2016 05:45	

São encontradas várias abordagens na bibliografia para lidar com este problema.

A primeira abordagem a considerar é a eliminação dos dados em falta, tal como foi feito em [39], ou seja não proceder ao preenchimento dos mesmos. Isto poderia ser feito excluindo os registos em si ou os dias em que existem registos em falta. Esta abordagem foi descartada pois o desenvolvimento do código em *python* é bastante facilitado se puder ser assumido que existe uma homogeneidade nos intervalos dos registos, ou seja poder assumir-se que existe o mesmo número de registos para o mesmo intervalo de tempo, o que no caso de registos de 15 minutos se traduz em ter sempre 96 registos diários, 672 registos semanais, …

Outra abordagem simples passa por utilizar um registo anterior ou seguinte para substituir os registos em falta. Foi o que foi feito na referência [40], em que se optou pelo preenchimento por valores do mesmo dia e hora da semana anterior.

Esta abordagem, apesar de ter em conta o efeito cíclico do consumo, provoca naturalmente resultados tendenciosos, pois assume que um registo é exatamente igual a um registo anterior. Para melhorar um pouco essa questão, pode optar-se por preencher com um valor resultante de uma operação que tem em conta vários registos do histórico de dados, por exemplo uma média ou mediana de registos existentes, sendo que a utilização da média tem a desvantagem de reduzir a variação no conjunto de dados, problema este que se agrava quanto maior for o número de registos omissos. Esta técnica foi usada por exemplo em [41], onde, nos casos em que os registos em falta sucessivos não são superiores a 3, preenchem-se os registos em falta com a média do registo anterior e posterior e, nos casos onde ocorrem mais de três registos omissos sucessivos, optou-se pelo preenchimento através de uma média entre o valor da semana anterior (ou da semana seguinte, no caso de ter sido feriado na semana anterior) e os valores na fronteira de dados.

Uma técnica ligeiramente melhor que a da média é a da interpolação linear, pelas razões já descritas aquando da sua utilização na questão das mudanças de hora.

Outras abordagens mais morosas implicam a utilização de métodos de previsão ou de algoritmos mais complexos dedicados apenas ao preenchimento de valores em falta. Em [42] recorreu-se ao algoritmo AMELIA 2, o qual fornece 5 valores possíveis para cada valor em falta, sendo nessa referência feita depois uma média desses valores para decidir o valor a usar no preenchimento. Em [43], onde se menciona que a percentagem de dados em falta relativamente ao conjunto de treino é inferior a 3% e que os intervalos em falta variam desde 1h a 7 dias, utilizou-se uma interpolação em cujo cálculo entram, para além de valores do histórico de dados, previsões resultantes de um modelo de previsão do tipo *deep* RNN (“*Recurrent Neural Network”*).

No que respeita às abordagens escolhidas para o presente trabalho, optou-se por descartar a utilização de abordagens mais complexas. Esta escolha prendeu-se mais uma vez com o facto da dimensão do problema em causa não justificar a utilização dos métodos mais complexos encontrados na bibliografia, uma vez que os dados em falta representam cerca de 0,0117% do conjunto total de dados disponíveis e aproximadamente 0,0167% do conjunto de dados de treino, bastante inferiores, por exemplo, ao descrito em [43], onde, como já foi descrito, se utilizou um procedimento mais complexo.

Relativamente aos registos em falta de 2016, recorreu-se então à estratégia da interpolação linear, já utilizada na questão das mudanças de hora e cuja fórmula é apresentada na (Equação 4.1), no entanto desta vez utilizaram-se para efeitos de cálculo os valores do dia anterior e seguinte à mesma hora, procurando aproveitar o efeito cíclico do consumo.

Na Figura 4-1 pode ser visto o preenchimento efetuado, bem como os dados em bruto e o perfil do dia anterior à mesma hora, o qual permite ter uma perceção do perfil esperado após a correção. Como se pode ver pela análise do gráfico dos dados em bruto, existe uma queda abrupta do consumo imediatamente antes da anomalia dos registos em falta, havendo inclusive um registo de 0 kW. Por este motivo incluíram-se na correção os três valores imediatamente antes do preenchimento, pois constituem claramente registos anómalos ou *outliers*. Após o preenchimento, verifica-se que, apesar do perfil seguido não ser exatamente igual ao do dia anterior, o intervalo de valores seguido é já dentro do esperado.

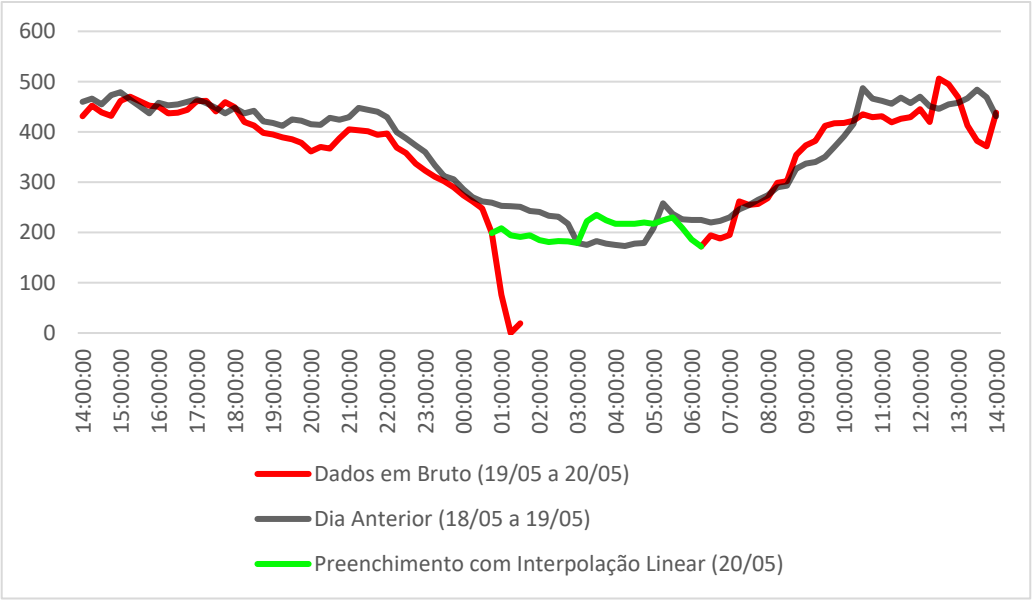


Figura 4-1-Correção de valores em falta no dia 20/05/2016

No que diz respeito à correção de valores em falta em setembro de 2018, uma vez que apenas existem quatro registos seguidos em falta, decidiu-se usar uma operação de média com base em registos históricos para completar o *dataset.*

A ideia inicial passava por fazer uma média entre o valor da semana anterior e o da semana seguinte, no entanto como no dia em causa ainda não existiam aulas, tendo as mesmas iniciado na semana seguinte, optou-se em alternativa por utilizar os registos da semana anterior com um peso de 70% e os do dia anterior com um peso de 30%, procurando assim aproveitar ao mesmo tempo o efeito cíclico semanal do consumo e o efeito de aproximação do início do ano letivo. Os resultados desta correção são apresentados na Figura 4-2. Tal como nos dados de 2016, incluíram-se mais dados na correção por se conseguirem identificar valores claramente anómalos, neste caso não só imediatamente antes da anomalia como também imediatamente depois. Assim sendo incluíram-se mais dois registos na correção, o anterior ao período em falta e o posterior.

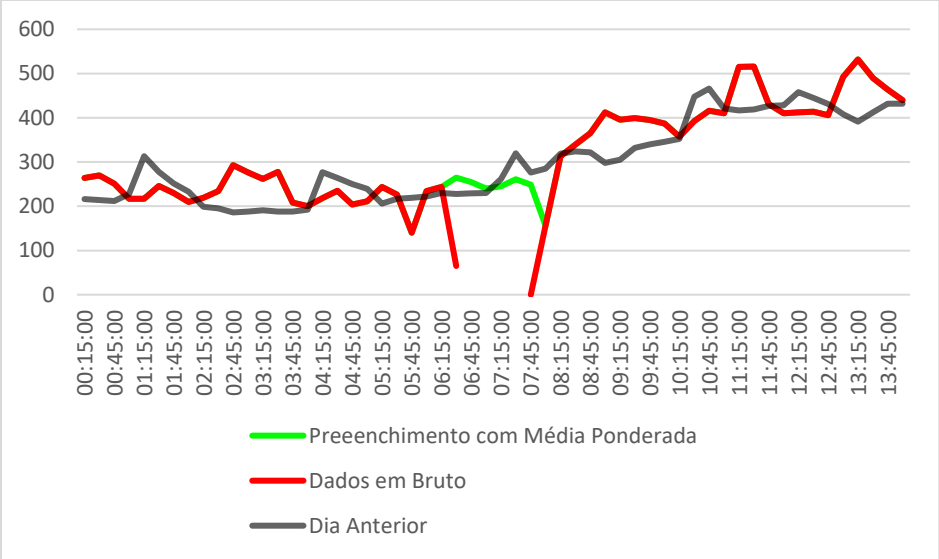


Figura 4-2-Correção de valores em falta no dia 13/09/2018

4.3.Valores anómalos/outliers

4.3.1. Variável endógena

Uma questão importante que pode também prejudicar a qualidade dos dados é a presença de valores anómalos (*outliers*). Estes registos contribuem para aumentar a variabilidade dos dados, reduzindo consequentemente a eficácia dos métodos estatísticos, o que é facilmente percetível por exemplo num cálculo de média, onde um valor muito distante dos restantes irá ter um forte impacto no resultado. Estes valores terão também uma influência negativa após o processo de normalização dos dados uma vez que esta, dependendo do método utilizado, poderá implicar a realização de cálculos com a média e desvio padrão ou cálculos com valores máximos e mínimos.

Existem várias estratégias possíveis para lidar com este problema, tendo sido feita em [44] uma revisão pormenorizada de alguns dos métodos mais utilizados, onde se recomenda, para conjuntos de dados cuja distribuição não é simétrica, como é o caso da série temporal de potência ativa do presente trabalho, um dos seguintes métodos: boxplot (ou método de Tukey); boxplot ajustado (semelhante ao Tukey mas que tem em consideração a assimetria dos dados); *Median Rule*; MAD("*Median Absolute Deviation*"), o qual usa a mediana e o desvio médio absoluto. A assimetria dos dados de consumo do Campus 2 do IPLEIRIA pode ser vista nos histogramas apresentados na Figura 4-3 e na Figura 4-4, em que um diz respeito a todas as horas e outro a apenas um intervalo específico, sendo que os restantes intervalos horários apresentam uma distribuição semelhante. As classes nestes gráficos foram definidas automaticamente.

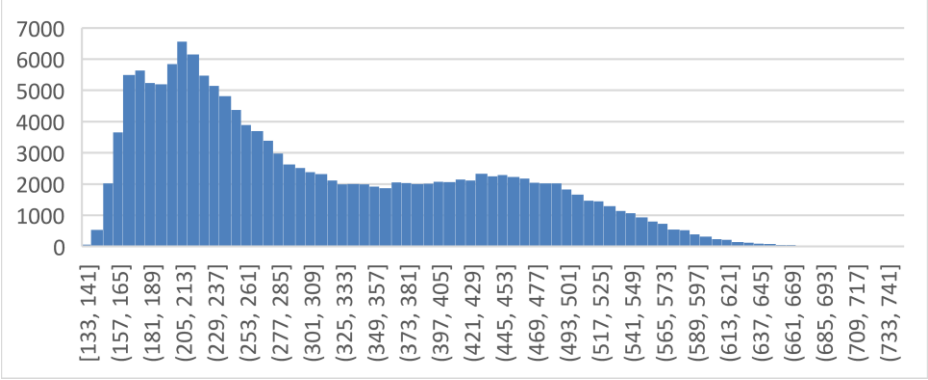


Figura 4-3-Histograma dos dados de potência ativa

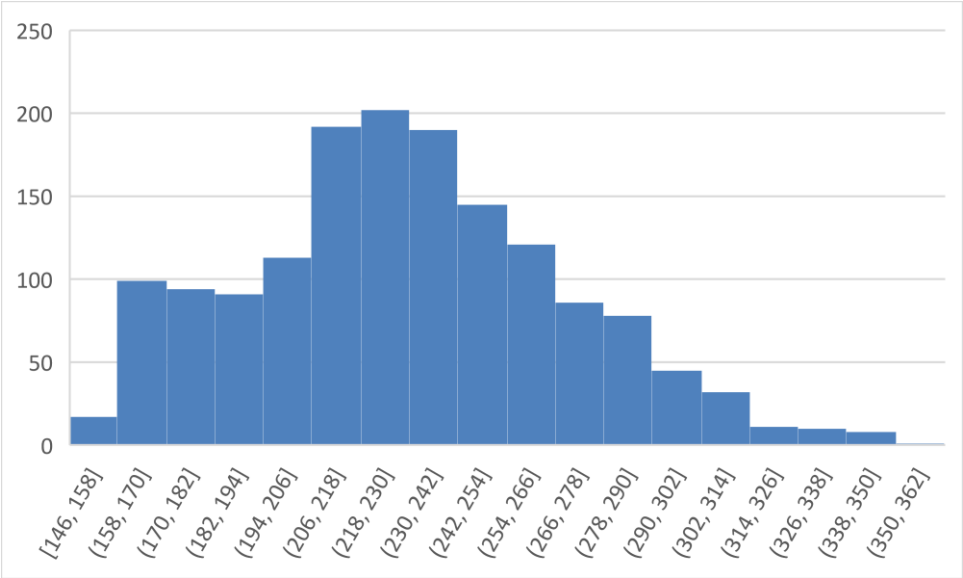


Figura 4-4-Histograma dos dados de potência ativa do intervalo das 00h:15

Outras técnicas mais avançadas implicam, à semelhança da deteção de valores em falta, a utilização de algoritmos e modelos de *machine learning* mais complexos destinados apenas à deteção e correção de *outliers*.

Um dos métodos mais utilizados e pioneiros nesta matéria é no entanto o método dos *boxplots*, introduzido pela primeira vez por John Tukey em [45], onde se definem como prováveis *outliers* os valores acima ou abaixo dos limites correspondentes a 1,5 vezes o valor do intervalo interquartis (IQR) e como valores extremos os valores acima ou abaixo dos limites correspondentes a 3 vezes o intervalo inter-quartis. É importante referir que não existe uma base estatística para o facto de Tukey ter usado 1,5 e 3 no que respeita ao IQR para fazer as fronteiras interior e exterior, sendo esta regra portanto uma *rule of thumb*, devendo por isso os *outliers* resultantes deste método ser analisados para cada caso. De facto, diversos autores sugeriram alterações ao método de Tukey que passaram também pela alteração do cálculo dos limites superior e inferior.

Em [46] os limites foram calculados com uma abordagem sequencial ajustada e em [47] criou-se um método denominado *Median Rule* onde o primeiro e terceiro quartis foram substituídos pela mediana, enquanto que a constante 1,5 foi substituída por uma fórmula para regular os limites.

A alteração do valor a multiplicar pelo IQR é algo que vale então a pena testar e irá então alterar a tolerância do método na deteção de valores anómalos, sendo esta tanto menor quanto maior for este valor, sendo por isso este valor designado de tolerância no presente trabalho. É então de esperar que um valor demasiado baixo provoque a classificação de valores normais como valores anómalos e que um valor demasiado alto provoque o efeito contrário. Com base neste princípio foram efetuados vários testes alterando a tal tolerância, procurando a adequada para o presente trabalho, resultando na Tabela 4-8 para o caso dos dados anteriores à pandemia, pois como já foi mencionado os perfis de consumo foram muito afetados pela mesma, pelo que faz sentido analisar ambas as situações separadamente.

De realçar também que o valor do intervalo interquartis será diferente para cada hora, apesar do valor de tolerância ser igual para todas as horas. Tomou-se esta decisão porque um *outlier* numa hora de maior consumo, o que numa instituição de ensino irá corresponder a um período de manhã ou de tarde, é naturalmente diferente de um valor anómalo numa hora de menor consumo, no presente caso por exemplo a meio da noite. Incluiu-se ainda no teste o método *Median Rule*, com um valor de tolerância de 2,3, por este ser o valor por omissão sugerido em [47].

Tabela 4-8-Nº de outliers detetados em função de valor de tolerância

Método	Tolerância	Número de *Outliers* Detetados
Tukey	0,5	11630
Tukey	1	2368
Tukey	1,5	774
Tukey	2	271
Median Rule	2,3	450
Tukey	2,5	96
Tukey	3	36

Para o valor de 1,5, obteve-se o *boxplot* apresentado na Figura 4-5. Como se pode ver todos os valores anómalos são registados nos intervalos entre as 23:15 e as 9:00, o que é de esperar visto que durante a manhã e tarde existe maior variabilidade, uma vez que as atividades letivas são predominantes neste período, o que significa que as diferenças de consumo entre, por exemplo, épocas letivas e épocas de férias sejam significativas, deixando a distância interquartis bastante maior. Uma vez que existe confiança na qualidade geral dos dados e que não se pretende correr o risco de perder informação ao alterar valores não anómalos, optou-se por escolher o método de Tukey com tolerância de 2,5.

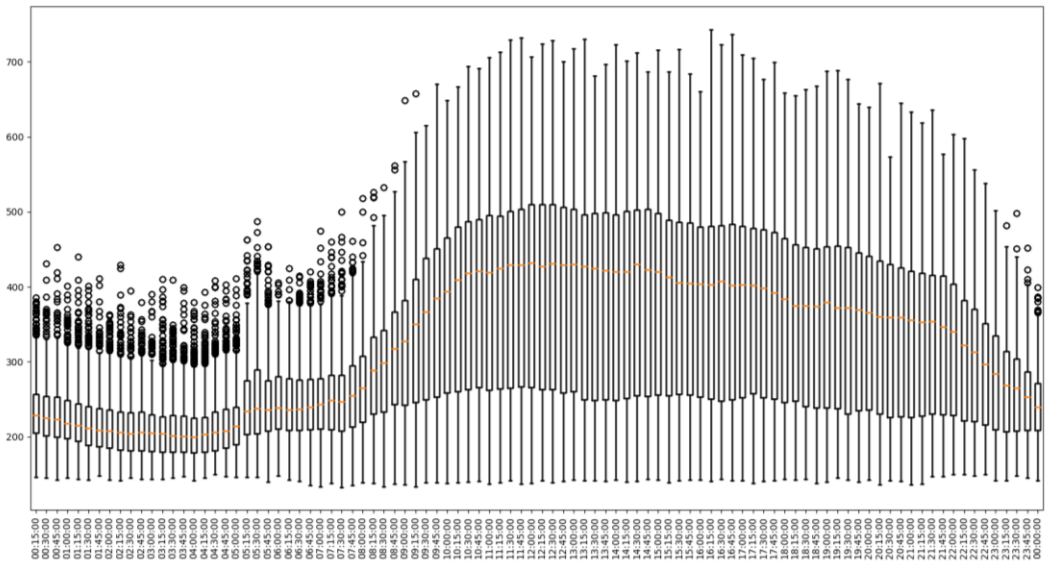


Figura 4-5- Boxplot para deteção de outliers (método Tukey)

Foram então corrigidos 96 registos, utilizando para o efeito a média entre os registos à mesma hora do dia anterior, dia seguinte, semana anterior e semana seguinte. Nos poucos casos em que se verificou que os valores à mesma hora do dia ou semana seguinte eram também *outliers*, utilizaram-se os registos de 2 dias ou 2 semanas depois, respetivamente.

Estes casos ocorreram bastante mais vezes para tolerâncias inferiores, sendo que por vezes vários valores à mesma hora de vários dias ou semanas consecutivos foram considerados como *outliers,* facto este que contribuiu então para optar por um método menos sensível a valores anómalos.

4.3.2. Variáveis exógenas

Em relação à correção de *outliers* nas variáveis exógenas, foi feita uma análise semelhante chegando-se à Tabela 4-9.

Tabela 4-9-Análise de outliers das variáveis exógenas

Tolerância	0,5	1	1,5	2	2,5	3	Percentagem com 3	2.3(Median Rule)
Graus dia Aq.	5	0	0	0	0	0	0%	0
Graus dia Arr.	11	7	5	4	2	2	3,44%	185
Temp High	185	57	9	3	2	2	0,11%	5
Temp Avg	220	28	3	3	2	2	0,11%	3
Temp Low	214	32	4	4	4	4	0,21%	4
D.Point High	311	165	113	54	9	3	0,16%	75
D.Point Avg	263	165	83	13	3	3	0,16%	41
D.Point Low	263	154	57	30	21	8	0,42%	31
Humidity High	378	259	204	184	169	165	8,75%	204
Humidity Avg	354	215	176	162	142	98	5,20%	170

Continua na próxima página

Tolerância	0,5	1	1,5	2	2,5	3	Percentagem com 3	2.3(Median Rule)
Humidity Low	484	219	25	0	0	0	0%	0
W.Speed High	305	114	23	7	2	2	0,11%	10
W.Speed Avg	392	146	70	30	10	6	0,32%	50
W.Speed Low	210	210	210	210	210	210	11,13%	210
Pressure High	504	360	313	311	251	239	12,67%	311
Pressure Low	493	395	320	311	311	298	15,80%	312
Prec.Accum.	455	400	386	376	373	346	18,35%	400

A tabela mostra que, inclusive para uma tolerância maior com o valor de 3, são detetadas centenas de *outliers* nalgumas variáveis, chegando no caso mais severo, correspondente à variável da precipitação acumulada, a serem detetados 346 valores anómalos. Uma vez que, ao contrário do consumo estas variáveis possuem uma resolução diária, o que significa que se possuem 1886 registos de cada variável exógena entre 2016 e fevereiro de 2021, inclusive, esse valor de outliers irá corresponder a cerca de 18,35% dos registos totais. No caso dos graus dia, onde a resolução é mensal, possuem-se 60 registos desde 2016 até 2020, sendo que não existem dados de 2021.

Não sendo desejável alterar uma percentagem tão significativa dos dados, pois correr-se-ia o risco de se adulterar os dados perdendo informação, tomou-se a decisão de não se proceder à correção de *outliers* quando a percentagem de *outliers* detetados fosse superior a 5%. O facto de não se corrigirem os outliers nestes caso pode ter-se refletido, por exemplo, nas temperaturas mínimas das variáveis das temperaturas de ponto de orvalho apresentadas na Tabela 3-3, as quais são claramente demasiado baixas.

De facto uma consulta à bibliografia permite concluir que tais percentagens não são comuns, por exemplo a referência onde são analisados diferentes métodos mostra que nunca foram atingidas tais percentagens, tendo a percentagem maior sido de 10,32% e para um método onde era expectável uma maior deteção de *outliers*, o que não é o caso do método de Tukey com tolerância de 3. Em relação às restantes variáveis procedeu-se à correção dos *outliers* detetados com a tolerância de 3, totalizando 32 *outliers* detetados. Optou-se ainda por não se corrigir os *outliers* referentes aos graus dia de arrefecimento devido a esta variável ser mensal e aos valores em causa, apesar de serem elevados, não estarem tão distantes dos restantes. A correção foi feita trocando os registos em causa por registos de uma estação meteorológica próxima, o que traz mais veracidade aos resultados do que se estes fossem substituídos por uma média de outros dias. Ao comparar os resultados que se tinham previamente com os da estação meteorológica próxima, verificou-se que alguns dos registos de velocidade do vento considerados como *outliers* eram semelhantes entre as duas estações, o que significa que os dados não foram medidos incorretamente, como aparenta ter acontecido com os dados de temperatura, mas que se trataram de dias particularmente anómalos. Nos casos em que tal aconteceu estes valores não foram corrigidos.

Em relação à potência reativa, optou-se por não corrigir os seus *outliers*, pois tal seria um processo tão moroso como o da potência ativa e verificou-se que essa variável não era muito relevante nesta dissertação, como será comprovado no capítulo da *feature selection*.

4.4. Normalização

Uma medida que é necessária em quase todos os modelos é a normalização dos dados, que visa garantir que todas as entradas/*features* se encontram na mesma escala. As duas técnicas de normalização mais usadas são a *min-max* e a standard. A primeira é fortemente influenciada por *outliers*, uma vez que a sua fórmula de cálculo se baseia nos valores máximo e mínimo, sendo por este motivo a menos utilizada.

Na segunda, os valores normalizados ficam numa gama não limitada, o que pode ser um problema com alguns algoritmos e que não acontece com a normalização *min-max*, já que os valores normalizados se encontram sempre entre -1 e 1.

$$x_{i\ normalizados} = \frac{x_i - \min(\boldsymbol{X})}{\max(\boldsymbol{X}) - \min(\boldsymbol{X})}$$

(Equação 4.3)

$$x_{i\ normalized} = \frac{x_i - \text{média}\ (\boldsymbol{X})}{desvio\ padrão\ (\boldsymbol{X})}$$

(Equação 4.4)

Ambas as técnicas são encontradas facilmente na bibliografia, por exemplo na referência [48] foi implementada a normalização *standard* e na referência [49] a normalização *min-max*.

Foram testados ambos os tipos de normalização com diversos modelos com diferentes parametrizações e em todos eles a normalização standard produziu resultados claramente melhores, pelo que se optou por esta estratégia. De maneira a evitar problemas de *data leakage*, apenas o comando "*scaler.transform*" foi aplicado a todo o *dataset*, tendo o comando "*scaler.fit*" apenas sido aplicado aos dados de treino.

5. Métricas de erro e seleção e extração de *features*

O uso de um maior número de *features* não significa necessariamente que o modelo seja mais eficiente. De facto, uma análise da importância de cada uma das *features* e eventual remoção de *features* menos relevantes poderá ser bastante vantajosa, levando a uma redução dos dados envolvidos, e consequentemente diminuindo bastante os tempos de processamento. Também as métricas de erro poderão melhorar com esta medida, como aconteceu na referência [50], onde a remoção de três *features* irrelevantes num número inicial de vinte levou a melhorias nas duas métricas testadas (MAE- "*Mean Absolute Error"* e MAPE– "*Mean Absolute Percentage Error"*) na previsão de consumos de um edifício não residencial. Nessa referência foi usada uma seleção de *features* baseada em correlação por software. Na referência [48], onde se aplica um método de previsão a dois edifícios (um administrativo e um académico), a remoção de três *features* não afetou a métrica R^2 obtida antes da remoção, tendo essa remoção sido baseada numa análise de correlação. Essa análise conclui que a variável da velocidade do vento não tinha impacto no consumo de eletricidade, e que os pares de *features* irradiação global/temperatura exterior e irradiação global/humidade possuíam grande correlação entre si, pelo que se comportam de maneira semelhante e se puderam eliminar as duas últimas sem comprometer a eficiência da previsão. Na referência [51] também se conseguiram eliminar *features* irrelevantes, o que levou também a melhores resultados.

Para além da seleção das entradas, algo que também poderá ajudar a diminuir a dimensionalidade do *dataset* é a extração de *features*, a qual consiste em combinar variáveis que tenham uma correlação elevada entre si. Esta técnica possui, em relação aos exemplos já mencionados onde se removeram variáveis com uma elevada correlação, a vantagem de eventualmente se perder menos informação, pois nenhuma *feature* é eliminada, mas sim combinada com outra de alguma forma, pelo que antes de remover alguma *feature*, é uma boa prática avaliar se ao combiná-la com outra trará alguma vantagem aos resultados da previsão. Na referência [42] foram utilizados os algoritmos PCA e *Factor Analysis* no que à extração de *features* diz respeito, tendo o PCA obtido melhores resultados.

5.1.Métricas de erro

A prática mais utilizada para avaliar e comparar o desempenho dos modelos de previsão é a utilização de métricas de erro. Estas métricas são definidas como construções matemáticas e/ou lógicas que se destinam a medir o quão próximo as observações reais estão das previstas, relacionando muitas vezes a variação entre as observações reais e medidas em termos de erros [52].

Existem diversas métricas de erro, sendo que a referência [52] enumera um total de vinte e nove para o caso dos problemas de regressão, da qual a previsão de consumos faz parte, sendo que nessa referência também são enumeradas a definição, fórmula de cálculo e algumas observações para cada métrica de erro. Ainda que o número de métricas disponível seja elevado, a grande maioria dos artigos analisados continua a recorrer a um número reduzido de métricas, as quais continuam a prevalecer como as mais populares. Para este trabalho as métricas utilizadas foram o erro médio absoluto (MAE), o erro médio quadrático (MSE – "*Mean Square Error"*), a raiz do erro médio quadrático (RMSE – "*Root Mean Squared Error"*) e o erro percentual médio absoluto (MAPE). Utilizou-se ainda o coeficiente de determinação (R^2) na aplicação da *Sequential Feature Selection* e numa parte do teste final. As fórmulas destas métricas são apresentadas nas equações seguintes.

$$MAE = \frac{\sum_{i=1}^{Número\ de\ observações} |y_i - \hat{y}_i|}{Número\ de\ observações}$$

(Equação 5.1)

$$MSE = \frac{\sum_{i=1}^{Número\ de\ observações} (y_i - \hat{y}_i)^2}{Número\ de\ observações}$$

(Equação 5.2)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{Número\ de\ observações} (y_i - \hat{y}_i)^2}{Número\ de\ observações}}$$

(Equação 5.3)

$$MAPE = \frac{\sum_{i=1}^{Número\ de\ observações} \left[\frac{(y_i - \widehat{y_i})}{y_i}\right]}{Número\ de\ observações} \quad \textit{(Equação 5.4)}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{Número\ de\ observações} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{Número\ de\ observações} (y_i - \bar{y}_i)^2} \quad \textit{(Equação 5.5)}$$

Nas equações anteriores:

- yi representa o valor de referência;
- ŷ representa o valor previsto;
- $\bar{y}$ representa a média dos valores de referência.

Cada uma das métricas anteriores possui vantagens e desvantagens, pelo que a utilização de várias métricas é útil para ter uma melhor perceção da qualidade global do modelo. Em termos de unidades, as do MAE, MSE e RMSE correspondem às dos dados de saída (ou dados de saída ao quadrado no caso do MSE) uma vez que são métricas de erro absoluto, o que pode ser uma desvantagem no caso de se pretenderem comparar modelos aplicados em casos de estudo com características muito diferentes, nomeadamente a grandeza dos consumos envolvidos. Neste aspeto em particular o MAPE e o R^2 são mais vantajosos pois correspondem a uma percentagem, sendo portanto métricas de erro relativo.

5.2.Análise de autocorrelação

Em grande parte dos modelos de *machine learning*, as *features* de maior importância são geralmente registos anteriores da variável que se pretende prever, pelo que faz sentido definir esta *feature* antes das restantes. Assim sendo, uma boa forma de definir qual o histórico de dados a ser utilizado e de ver qual a periodicidade do consumo no Campus 2 do IPLEIRIA é a realização de uma análise de autocorrelação, a qual permite saber a relação existente entre uma variável e os valores anteriores dessa mesma variável, como no caso da referência [53], onde foi feita uma análise de autocorrelação na previsão de consumos em edifícios residenciais, podendo a mesma lógica ser seguida num edifício não residencial.

Na Figura 5-1 é apresentado o gráfico de autocorrelação, o qual apresenta no eixo das ordenadas o coeficiente de correlação e no das abcissas o *lag* com o qual se está a comparar. De salientar que neste caso 1 *lag* equivale a um intervalo de 15 minutos, pelo que 2976 *lags*, o máximo apresentado na Figura 5-1, irá equivaler a um intervalo de 1 mês com 31 dias.

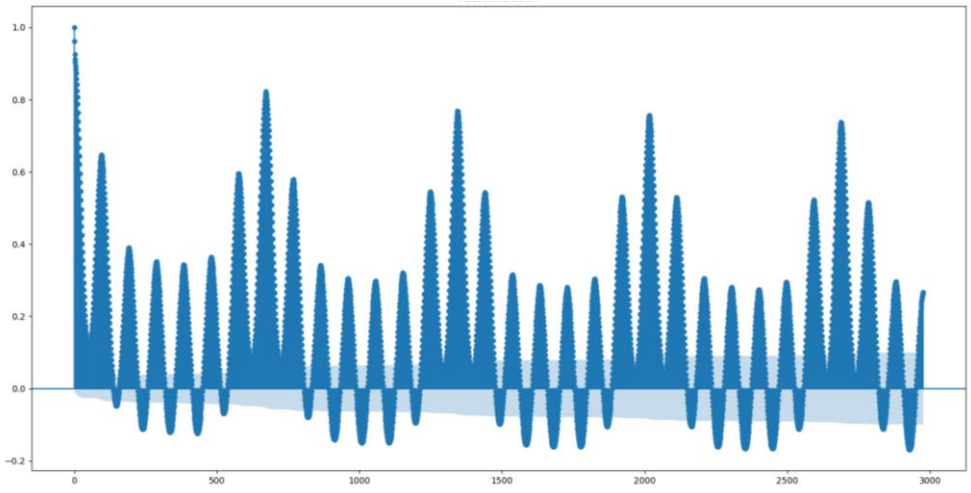


Figura 5-1-Análise de autocorrelação

Como se pode ver existe uma muito alta correlação com os valores dos intervalos imediatamente antes da variável, sendo a partir daí a correlação maior para os valores de semanas anteriores à mesma hora e de horas adjacentes. A variável apresenta também valores de correlação altos para valores à mesma hora e horas adjacentes de dias anteriores. A análise de autocorrelação comprovou então o que seria espectável, uma vez que é de esperar que o consumo no Campus 2 do IPLEIRIA, por exemplo de uma segunda-feira, seja parecido ao da segunda-feira seguinte, devido à semelhança de horários das atividades letivas, no entanto em termos dos *lags* diários o perfil de consumo já não será tão semelhante. Esta constatação deve-se principalmente por os fins de semana possuírem consumos totalmente diferentes dos dias úteis, diferindo estes também um pouco entre si devido aos diferentes horários das atividades letivas.

5.3.Extração com PCA das *features* referentes a registos anteriores

Com base nos resultados da análise de autocorrelação anterior decidiu-se então testar 63 variáveis correspondentes a diferentes *lags,* as quais são enumeradas no Anexo A. Evidentemente, tal representa um número muito elevado de *features*, pelo que é importante reduzir a dimensionalidade do problema. Para tal recorreu-se ao algoritmo PCA para proceder à extração/combinação de *features,* tendo-se testado duas abordagens. A primeira foi baseada no princípio de combinar *features* referentes a um *lag* do mesmo dia, ou seja, forçando o algoritmo PCA a combinar por exemplo as cinco variáveis referentes a registos do dia anterior uma vez que estas terão naturalmente uma correlação elevada entre si por serem referentes a registos de horas adjacentes. A segunda consistiu em simplesmente alimentar o algoritmo com todas as 63 variáveis.

5.3.1. Escolha do número de componentes principais

O principal parâmetro a definir neste algoritmo é naturalmente o número de componentes que se pretende manter no final do processo. A escolha deste parâmetro é importante pois, por um lado, um valor muito pequeno deste valor seria desejável pois levaria a uma redução do *dataset* mas por outro lado se demasiadas componentes forem removidas poderão perder-se detalhes importantes[27]. Para definir esse parâmetro pode então recorrer-se a um gráfico da *cumulative explained variance* (variância cumulativa explicada). Um dos atributos fornecidos pelo algoritmo PCA quando executado com recurso à biblioteca *Scikit-Learn* é a *explained variance* (variância explicada), a qual indica a quantidade de informação (variância) que pode ser atribuída a cada uma das componentes principais [28],pelo que realizando um *plot* da variância explicada cumulativa é possível visualizar a proporção de variância se incluirmos um número de componentes principais até um dado valor.[29] Geralmente é útil escolher um número de componentes principais tal que 85% a 99% da variância total seja explicada, o que está de acordo com a bibliografia nesta área[30].

[27] "On the one hand, a very small K would be desirable because it would reduce the amount of data, but on the other hand, if too many dimensions are removed, the data may not capture important details."[78]
[28] "The explained variance tells us how much information (variance) can be attributed to each of the principal components".[79]
[29] "This is where the yellow line comes in; the yellow line indicates the cumulative proportion of variance explained if you included all principal components up to that point."[80]
[30] "In general, we would like to choose the smallest K such that 0.85 to 0.99 (equivalently, 85% to 95%) of the total variance is explained, where these values follow from PCA best practices. In this paper we use the value of 0.95."[78]

No Anexo B são apresentados os gráficos com a variância explicada cumulativa para as 13 extrações efetuadas no teste 1, onde se definiu um *threshold* de 90%. Como se pode ver nos gráficos em todos os casos conseguiu superar-se o *threshold* mantendo apenas uma componente, pelo que após se aplicar este algoritmo passaram-se das 63 variáveis iniciais para 13. Este processo de *feature extraction* das variáveis referentes ao histórico de dados encontra-se ilustrado no Anexo C.

Já na segunda abordagem, onde se alimentou o algoritmo com as 62 variáveis de uma só vez, testaram-se dois *thresholds*, um de 95% e um de 90%, correspondendo estes testes ao 2 e 3, respetivamente. Os resultados da *cumulative explained variance*, bem como da *explained variance* de cada componente, são apresentados na Figura 5-2 e Figura 5-3. A razão pela qual se testaram dois *thresholds* foi devido ao facto de com um *threshold* 5 % mais pequeno se ter diminuído o número de componentes principais necessárias de 15 para 9, como se pode ver nas figuras.

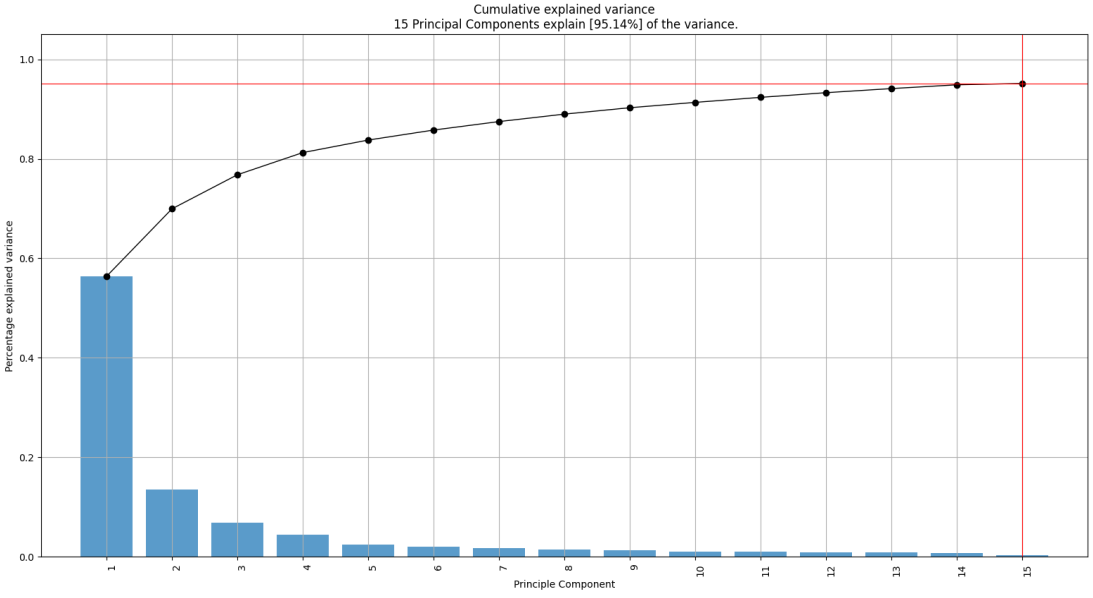


Figura 5-2-Cumulative explained variance para o teste 2 da extração de registos anteriores (95%)

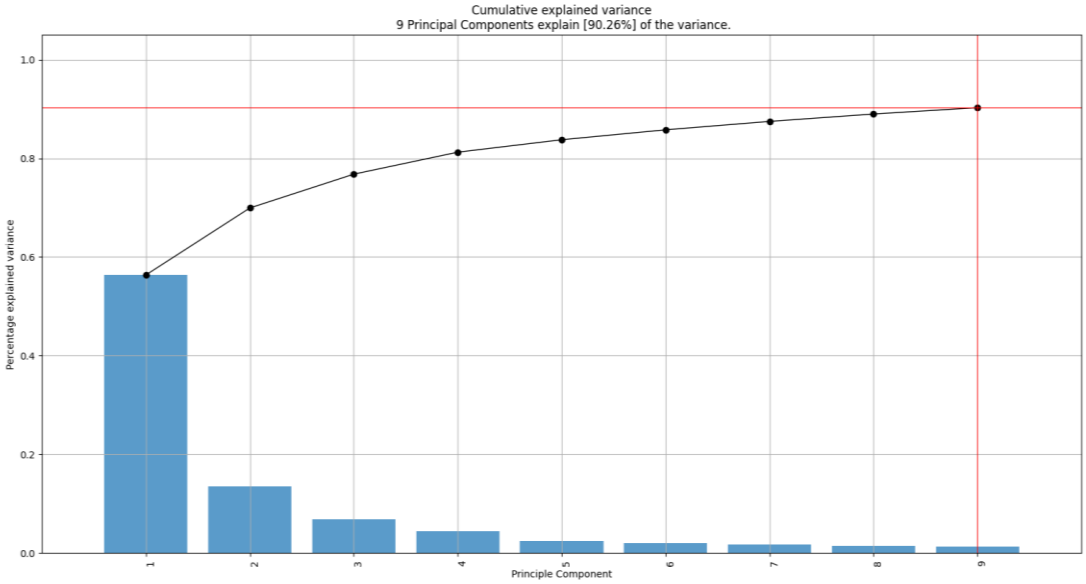


Figura 5-3-Cumulative explained variance para o teste 3 da e extração de registos anteriores (90%)

5.3.2. Impacto nos resultados

A fim de avaliar o efeito deste processo foram feitas simulações com os algoritmos Random Forests e MARS, já que os mesmos possuem tempos de processamento baixos e efetuam uma feature selection automática, fornecendo como outputs coeficientes que permitem auxiliar na escolha das features. Realça-se que, não tendo nesta fase sido ainda feita a otimização de parâmetros dos algoritmos MARS e Random Forests, utilizaram-se neste capítulo os valores predefinidos para os mesmos. A exceção a isto é, no algoritmo MARS, o parâmetro "feature_importance_type", que foi alterado para ('rss', 'gcv', 'nb_subsets') de maneira a ser possível aceder a estes valores que refletem a importância de cada variável. Já no algoritmo Random Forests, apenas se alterou o número de "n_estimators" para 50. Na Tabela 5-1 e na Tabela 5-2 apresentam-se os resultados de previsão com os resultados de cada teste efetuado, bem como utilizando todas as 63 variáveis. Tendo em conta que na presente dissertação é feita várias vezes referência aos tempos de processamento, é importante realçar que as simulações foram efetuadas num computador com um processador i7-9750H e 8 GB de RAM. Foram assinaladas a verde as melhores métricas obtidas de entre os 3 testes. Como se pode ver os tempos de processamento são fortemente reduzidos com a aplicação de um número menor de features e, apesar desta redução ser significativa em todos os testes face ao teste inicial com todas as 63 variáveis, houve apenas um ligeiro aumento das métricas da maioria das métricas de erro, pelo que se conseguiu diminuir a dimensionalidade dos dados sem uma degradação forte das métricas de erro.

Para além disso, houve até algumas métricas que melhoraram em relação ao teste inicial. Em ambos os algoritmos, o teste 3 obteve métricas piores que os restantes. Em relação aos restantes testes, o teste 1 foi o que produziu melhores métricas nas *Random Forests* e o teste 2 no algoritmo MARS.

Posto isto decidiu-se escolher o teste 1 e utilizar as 13 variáveis criadas nessa simulação, uma vez que esta produziu menos variáveis e consequentemente melhores tempos de processamento, bem como pelo facto de o algoritmo *Random Forests* ter sido o que produziu melhores métricas de entre os 2 testados.

Tabela 5-1-Resultados dos testes PCA com Random Forests (extração de registos anteriores)

		Validação				Treino				
Simulação	**Nº de *Features***	**MAE [kW]**	**MSE [kW²]**	**RMSE [kW]**	**MAPE [%]**	**MAE [kW]**	**MSE [kW²]**	**RMSE [kW]**	**MAPE [%]**	**Tempo Processo Treino [s]**
Todas as Variáveis	63	37,76	2868,2	53,49	12,94	11,45	274,2	16,55	3,92	252,98
Teste 1	13	38,54	2988,6	54,59	13,19	10,18	211,0	14,51	3,49	70,14
Teste 2	15	39,69	3150,0	56,07	13,49	10,37	213,0	14,58	3,58	85,11
Teste 3	9	41,37	3386,5	58,16	14,06	10,60	219,5	14,80	3,68	55,29

Tabela 5-2- Resultados dos testes PCA com MARS (extração de registos anteriores)

		Validação				Treino				
Simulação	**Nº de *Features***	**MAE [kW]**	**MSE [kW²]**	**RMSE [kW]**	**MAPE [%]**	**MAE [kW]**	**MSE [kW²]**	**RMSE [kW]**	**MAPE [%]**	**Tempo Processo Treino [s]**
Todas as Variáveis	63	41,90	3462,9	58,82	14,10	41,51	3396,4	58,28	13,98	59,36
Teste 1	13	41,73	3463,2	58,85	14,15	40,74	3227,2	56,80	13,82	21,80
Teste 2	15	41,37	3333,7	57,72	13,94	40,28	3133,8	55,97	13,63	46,44
Teste 3	9	44,34	3720,5	60,98	15,09	43,48	3556,7	59,63	14,83	28,02

5.4.Análise da variância e desvio padrão

Uma análise inicial que se pode fazer em termos de *feature selection* é a da variância e do desvio padrão de cada *feature*, as quais permitem avaliar a variabilidade de uma variável. Tipicamente o desvio padrão é mais utilizado, pois possui as mesmas unidades da variável para o qual foi calculado, enquanto a variância resulta no quadrado dessa unidade.

A ideia passa por, se uma variável possuir uma variância ou desvio padrão muito próxima de zero, tal significa que possui muito pouca variabilidade, ou seja, que é praticamente constante, e como tal apenas trará complexidade ao modelo sem qualquer benefício associado.

A Tabela 5-3 mostra a variância e desvio padrão das features analisadas, estando assinaladas a vermelho as que estão abaixo do *threshold* de 0,1 para a variância e de 1 para o desvio padrão. Estes *thresholds* foram definidos simplesmente tendo em conta a amostra em estudo. De facto, uma análise visual das *features* assinaladas nas duas colunas permite constatar que a velocidade mínima do vento é praticamente sempre igual a 0, pelo que pode ser descartada. Em relação às *features* relacionadas com a humidade, convém realçar que a sua unidade é em percentagem, ou seja, varia de 0 a 1, pelo que os baixos resultados poderão ser enganadores e as variáveis não deverão ser ainda descartadas. Por fim, os baixos valores de desvio padrão das variáveis “Função Seno” e “Função Cosseno” não devem ser considerados pois estas variáveis foram construídas para replicar o efeito de periodicidade do consumo.

Tabela 5-3-Variância e desvio padrão

Feature	Unidade	Variância [X²]	Desvio Padrão [X]
Temp High	°C	43,686	6,607
Temp Avg	°C	23,586	4,855
Temp Low	°C	26,027	5,100
D.Point High	°C	35,797	5,981
D.Point Avg	°C	41,087	6,408
D.Point Low	°C	56,831	7,536
Humidity High	%	0,037	0,192
Humidity Avg	%	0,038	0,196
Humidity Low	%	0,044	0,211
W.Speed High	m/s	6,377	2,524
W.Speed Avg	m/s	0,770	0,877
W.Speed Low	m/s	0,024	0,155
Pressure High	kPa	98,935	9,943
Pressure Low	kPa	141,460	11,890
Prec.Accum.	mm	1586,676	39,819
Graus dia Aq.	-	6981,692	83,528
Graus dia Arr.	-	700,548	26,459
Tipo Dia	-	4,010	2,002
Seno	-	0,501	0,707
Cosseno	-	0,500	0,707
Dia	-	4,008	2,001
Minuto	-	767,917	27,711
Reativa	kvar	4032,550	63,503

5.5.Análise da correlação entre *features* e potência

Uma análise tipicamente feita em termos de *feature selection* é a da correlação. Esta análise permite perceber o quanto cada uma das *features* se relaciona com a variável que se pretende prever, neste caso a potência ativa. Para este efeito foram calculados dois tipos de correlação, a de Pierson e a de Spearman., sendo que a correlação de Spearman possui a particularidade de conseguir captar melhor relações não lineares que a de Pearson. Como a maioria destas variáveis são diárias ou mensais, no caso dos graus-dia, enquanto a potência ativa possui uma resolução quarta-horária, utilizou-se na análise o consumo diário e mensal em vez da potência. A exceção foram as variáveis “Minuto” e “Reativa”, onde se utilizou a potência ativa visto ter a mesma resolução que estas variáveis.

Como algumas das variáveis são claramente sazonais, foi feita também a análise dividindo os dados em período de Verão e de Inverno, baseando essa separação nas mudanças de hora.

Foi também feita a análise com e sem os dados durante a pandemia de covid-19. Os resultados desta análise são apresentados no Anexo D, sendo assinaladas a verde as *features* que ultrapassam um *threshold* de 0,2. Tal como na análise da variância e desvio padrão, estes *thresholds* foram definidos tendo em conta a amostra em estudo. A principal conclusão que se pode retirar desta análise é a de que as variáveis "Tipo Dia", "Função Cosseno", "Minuto" e "Reativa" possuem uma correlação com o consumo mais forte que as outras variáveis, já que foram as únicas que ultrapassaram o *threshold* definido em todos os testes. Também as variáveis "Reativa", "Minuto" e as variáveis referentes aos Graus-Dia apresentam bons resultados nalguns testes. Em relação às restantes variáveis, apresentam de maneira geral correlações bastante baixas, sendo os piores resultados os das variáveis da pressão acumulada, das temperaturas de ponto de orvalho, a variável "Dia" e a variável "Função Seno", o que indica estas variáveis deverão estar entre as descartadas.

5.6. Análise da relevância das *features* com base nos resultados dos algoritmos Random Forests e MARS

Como já se referiu anteriormente, os algoritmos Random Forests e MARS fornecem coeficientes que indicam o quanto uma variável foi relevante para uma dada previsão, podendo esses valores ser então utilizados para efeitos de *feature selection*. No Anexo F são então apresentados esses valores, que foram obtidos fazendo a média entre os três valores referentes a cada um dos subconjuntos de validação para cada variável.

No caso do algoritmo MARS, que fornece três critérios para efeitos de ordenação da importância das *features*, considerou-se como coeficiente final a média entre os três critérios visíveis na tabela.

Pela análise do Anexo E é então possível tirar algumas conclusões importantes. A que salta à vista é o facto de a variável "PCA -1 semana" possuir coeficientes muito mais altos que as restantes variáveis nos dois algoritmos, pelo que se pode concluir que é a *feature* mais relevante. Depois dessa variável, as mais relevantes são as referentes aos valores com 1 dia e 3 semanas de atraso. É também possível corroborar os resultados da análise de correlação feita anteriormente no que diz respeito às variáveis exógenas, uma vez que praticamente todas apresentam coeficientes muito baixos, com exceção da variável "Tipo Dia" nas *Random Forests* e "Cosseno" no MARS. A variável "Minuto" conseguiu também ter um resultado razoável nas *Random Forests*, sendo considerada a 11ª mais relevante. Fica

também evidente que algumas variáveis referentes a registos anteriores poderão não ser relevantes, como as que têm um atraso de 6 e 7 semanas, uma vez que não entram nas 10 mais relevantes em nenhum dos algoritmos.

5.7. Análise dos resultados da *Sequential Feature Selection*

5.7.1. Escolha do número final de *features*

Como último método de seleção de variáveis recorreu-se à *sequential feature selection* com o algoritmo da regressão linear uma vez que este possui tempos de processamento baixos. A grande vantagem deste método quando comparado com os anteriores é que fornece um critério sobre o número ideal de *features* a utilizar.

Da Figura 5-4 à Figura 5-6 são apresentados os gráficos que mostram a evolução da métrica R^2 consoante o número de *features* selecionadas, para o caso do processo *forward* nas duas primeiras figuras e do processo *backward* nas duas últimas.

No caso do processo *forward*, o pico de performance ocorre com um número de *features* entre as 19 e as 23, no entanto analisando a Figura 5-4 conclui-se que a partir das sete as melhorias de performance com o aumento do número de *features* são ligeiros e a partir das 13 o aumento é praticamente desprezável. Já no processo *backward* o pico localiza-se nas 23 *features*, no entanto pela análise da Figura 5-7 verifica-se que os aumentos a partir de um número de *features* igual a 13 são muito ligeiros.

Com base nas análises anteriores, conclui-se que um número de *features* igual a 13 é adequado e oferece um bom compromisso entre descartar as variáveis menos relevantes, contribuindo para a rapidez dos algoritmos, e ao mesmo tempo não prejudica a performance dos modelos de previsão com perdas de informação relevante.

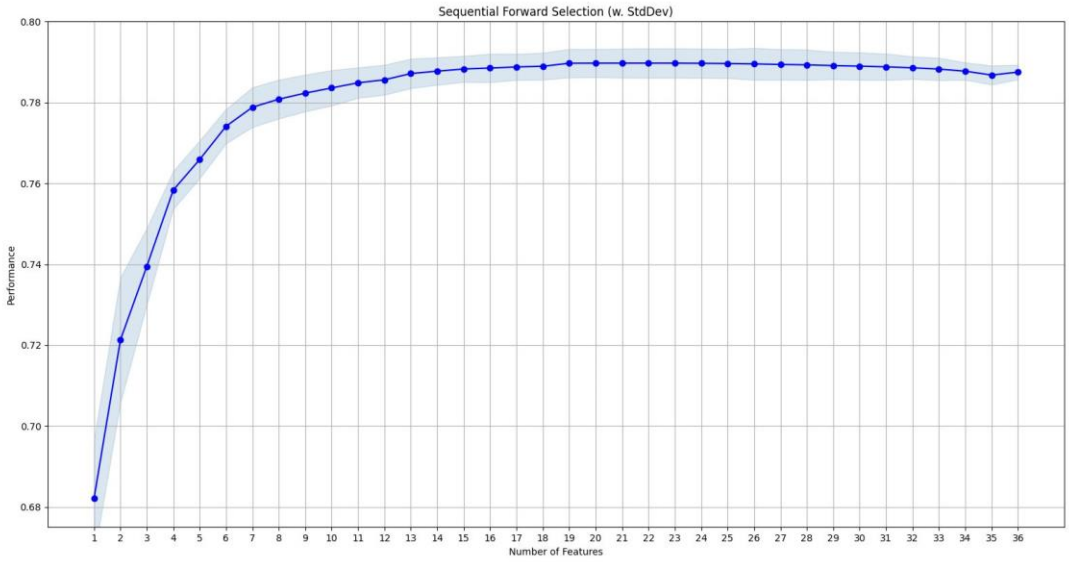


Figura 5-4-Evolução da métrica R^2 com número de features (Sequential Forward Selection)

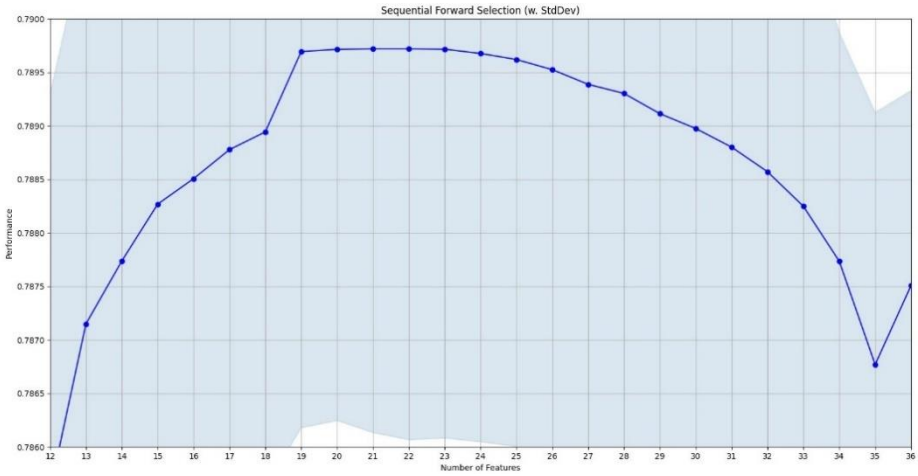


Figura 5-5-Evolução da métrica R^2 com número de features com zoom (Sequential Forward Selection)

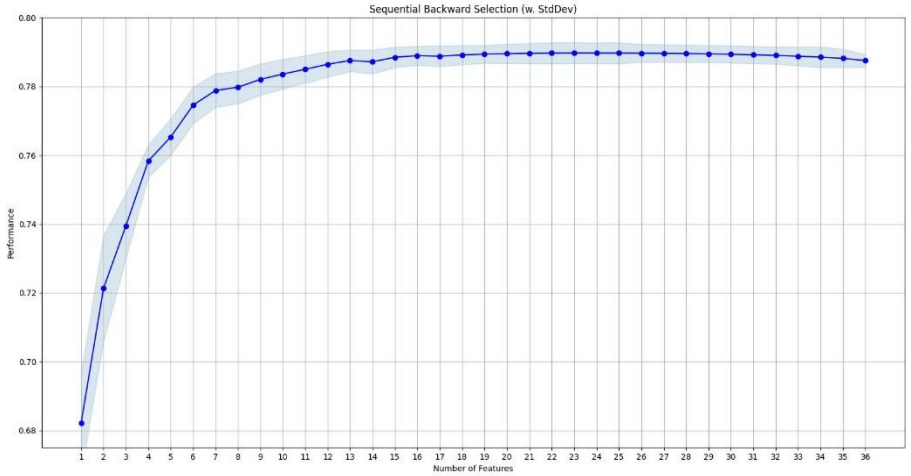


Figura 5-7-Evolução da métrica R^2 com número de features (Sequential Backward Selection)

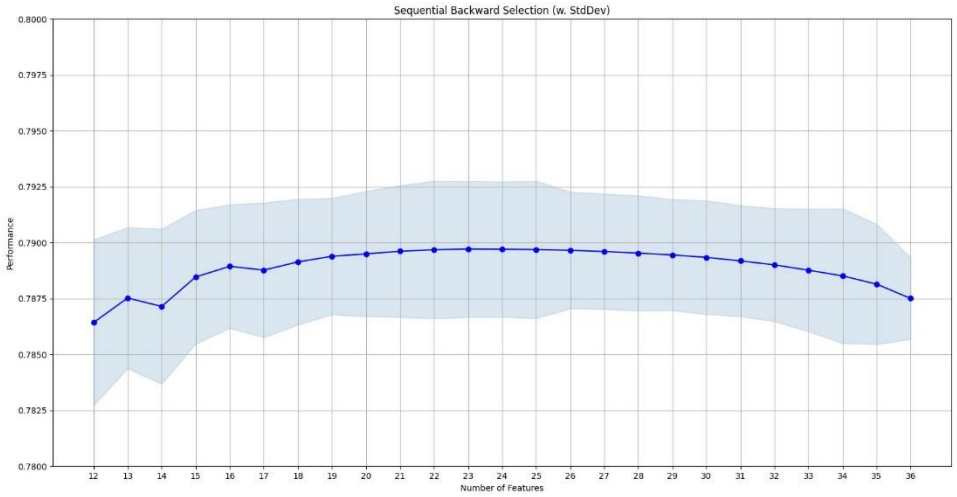


Figura 5-6-Evolução da métrica R^2 com número de features com Zoom (Sequential Backward Selection)

5.7.2. Análise da relevância das *features*

Em relação à importância de cada *feature*, a ordem fornecida pelos processos *forward* e *backward* é apresentada também no Anexo E e a mesma permite fortalecer a ideia de que a variável "PCA -1semana" é a mais importante, tendo também as variáveis "PCA -1Dia", "PCA -3semanas" e "Tipo Dia" tido resultados muito bons à semelhança do que se verificou nos testes anteriores. Em relação às variáveis exógenas, com exceção da variável "Tipo Dia" apenas a variável "Cosseno" apresentou bons resultados em ambos os processos *forward* e *backward*.

5.8. Seleção final de *features*

Com base nos resultados visíveis no Anexo E selecionaram-se, de acordo com a ordenação média entre os quatro algoritmos utilizados, as 13 variáveis mais relevantes, as quais são apresentadas na Tabela 5-4 sem nenhuma ordem de relevância em particular.

Tabela 5-4-Variáveis selecionadas após processo de feature selection

Variável	
Cosseno	PCA -4semanas
Tipo Dia	PCA -5semanas
PCA -1Dia	PCA -6Dias
PCA -1semana	PCA -7semanas
PCA -2semanas	PCA -8Dias
PCA -3Dias	PCA -8semanas
PCA -3semanas	

O efeito do processo de *feature selection* nas métricas de erro e nos tempo de processamento obtidos com os algoritmos *Random Forests* e MARS pode ser visto na Tabela 5-5. Como se pode ver, quando se passam de 36 para 13 variáveis as métricas do conjunto de validação são muito pouco afetadas, havendo inclusive algumas que melhoraram, o que indica que o processo de seleção de variáveis foi feito corretamente. Para além disso, conseguiu-se também que os tempos de processamento fossem reduzidos significativamente, o que se pode comprovar pelos tempos obtidos durante o processo de treino antes e depois da seleção de variáveis.

Tabela 5-5-Efeito da seleção de features nas métricas de erro com Random Forests e MARS

		Validação				Treino				
Simulação	**Nº de *Features***	**MAE [kW]**	**MSE [kW^2]**	**RMSE [kW]**	**MAPE [%]**	**MAE [kW]**	**MSE [kW^2]**	**RMSE [kW]**	**MAPE [%]**	**Tempo Processo Treino [s]**
Antes de Feature Selection	36	34,37	2061,6	45,34	11,87	8,58	149,2	12,21	2,9	116,9
		41,37	3339,3	57,78	14,11	40,09	3077,7	55,47	13,71	34,26
Após Feature Selection	13	34,76	2122,2	46	11,84	9,57	176,7	13,28	3,26	58,68
		41,38	3335,8	57,75	14,14	40,38	3113,1	55,79	13,81	18,78

5.9. Correlação entre features e extração de features exógenas com PCA

5.9.1. Escolha do número de componentes principais

Uma vez que com exceção das variáveis "Tipo Dia" e "Cosseno" todas as variáveis exógenas foram excluídas no processo de seleção de *features* avaliou-se a possibilidade de proceder a uma extração de *features* com o algoritmo PCA nas variáveis exógenas, tal como feito para as *features* referentes a registos anteriores. Para tal foi feita uma análise de correlação entre as variáveis com os coeficientes de *Pearson* e *Spearman*, a qual é visível nos mapas térmicos apresentados no Anexo F. Estes mapas foram feitos com recurso à função "corrplot" da biblioteca "heatmap"[54] e optou-se por recorrer a estes mapas porque ao contrário da correlação feita no capítulo 5.4, pretende-se agora analisar a relação entre diferentes pares de variáveis e não apenas em relação à potência ativa, pelo que estes mapas tornam-se muito mais fáceis de analisar visualmente do que uma tradicional tabela com valores numéricos.

Quer no mapa referente à correlação de Pearson quer da correlação de Spearman verifica-se que como seria de esperar, as variáveis referentes à mesma grandeza, por exemplo as temperaturas máxima, média e mínima do ar, apresentam valores elevados de correlação entre si, sendo as únicas exceções as variáveis *"W. Speed Low"* nos dois tipos de correlação e a *"Humidity High"* na correlação de Spearman.

Para além disso verifica-se ainda mais quatro conjuntos de variáveis com bons níveis de correlação entre si, sendo eles:

- As variáveis "Seno" e "Dia" em ambos os mapas;
- Todas as 8 variáveis referentes a temperaturas, ou seja graus dia, temperaturas do ar e temperaturas de ponto de orvalho, no mapa da correlação de Spearman;
- As 5 vaiáveis referentes aos graus dia e às temperaturas do ar no mapa da correlação de Pearson;
- As 6 variáveis referentes a humidades e a temperaturas de ponto de orvalho, no mapa da correlação de Pearson.

À semelhança do que foi feito no capítulo 1.1 testaram-se diferentes abordagens, a primeira consistiu em alimentar o algoritmo PCA com todas as 23 variáveis exógenas e o segundo a

forçar a combinação de certas variáveis alimentando o algoritmo com as variáveis que apresentaram boa correlação de acordo com a análise anterior. Por fim realizou-se um teste com os algoritmos MARS e *Random Forests* para visualizar se as PCA são selecionadas como relevantes quando utilizadas com as 13 selecionadas no processo de *feature selection*.

No primeiro teste utilizou-se como critério para definir o número de componentes principais um *threshold* de *cumulative explained variance* de 95% e no segundo uma percentagem de 90%. O gráfico que mostra a *cumulative explained variance* em relação ao número de componentes principais para o primeiro teste é apresentado na Figura 5-8 e pela análise do mesmo é possível concluir que cinco componentes principais são suficientes para explicar 97,45% da variância nesse caso.

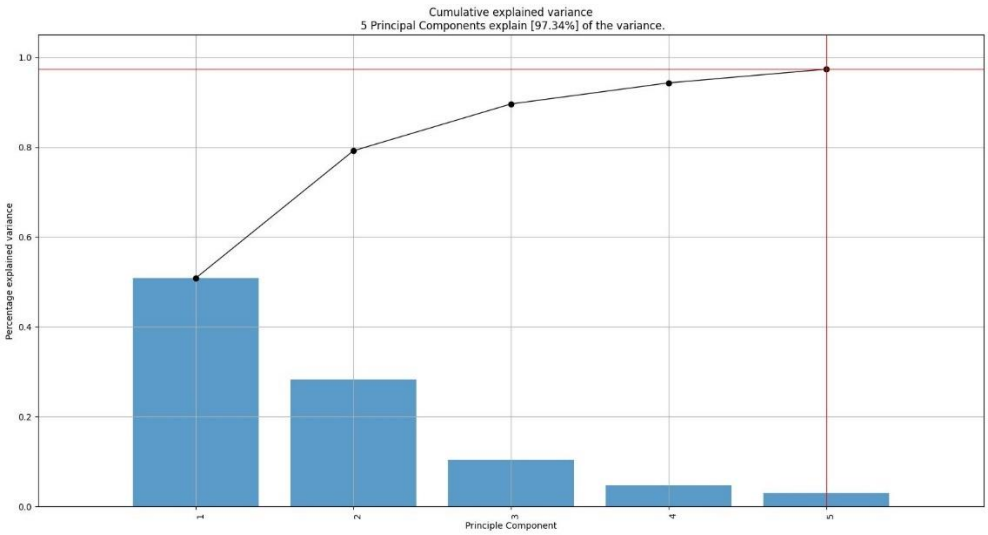


Figura 5-8-Cumulative explained variance para o teste 1 da extração de variáveis exógenas

Em relação ao segundo teste, a ilustração do processo seguido é visível no Anexo H e os gráficos da *cumulative explained variance* nos Anexo G. Tirando a extração das variáveis da humidade, que necessitou de duas componentes principais, todas as restantes necessitaram de apenas uma componente principal para explicar mais de 90% da variância.

5.9.2. Impacto nos resultados

Os resultados das métricas no conjunto de validação, quando aplicados os dois testes mencionados são apresentados na Tabela 5-6. Pela sua análise conclui-se que, comparativamente aos resultados da Tabela 5-5, algumas métricas pioraram e outras melhoraram ligeiramente, pelo que se confirma que apenas se considerarão no capítulo seguinte as variáveis da Tabela 5-4.

Tabela 5-6-Efeito da extração de features exógenas nas métricas de erro com Random Forests e MARS

		Validação				Treino				
Simulação	**Algoritmo**	**MAE [kW]**	**MSE [kW²]**	**RMSE [kW]**	**MAPE [%]**	**MAE [kW]**	**MSE [kW²]**	**RMSE [kW]**	**MAPE [%]**	**Tempo Processo Treino [s]**
Teste 1	Random Forests	34,96	2113,9	45,92	12,09	9,08	163,2	12,76	3,08	124,2
	MARS	47,12	4428,1	66,19	16,63	39,12	2923,1	54,06	13,47	101,9
Teste 2	Random Forests	34,61	2090,7	45,65	11,91	8,80	155,1	12,44	2,97	79,5
	MARS	41,21	3318,8	57,61	14,11	40,26	3105,2	55,72	13,77	37,7

6. Parametrização, *benchmarking* e análise de resultados

Neste capítulo, possuindo-se nesta fase os dados tratados e as variáveis mais relevantes selecionadas, passou-se de seguida à fase de afinação/*tuning* dos parâmetros dos diferentes algoritmos. Com os parâmetros dos algoritmos definidos efetuaram-se as simulações no conjunto de teste e fizeram-se as análises final de resultados com vista a definir o melhor modelo encontrado. Por fim efetuou-se um teste com o melhor modelo encontrado com os dados de potência ativa durante a pandemia Covid-19, com vista a avaliar a performance do mesmo nesse contexto. É importante realçar que na bibliografia é tipicamente feita a distinção entre o conceito de parâmetro e hiperparâmetro, sendo o parâmetro entendido como alvo de ajuste automático pelos algoritmos, como por exemplo os pesos numa rede neuronal, enquanto um hiperparâmetro é algo definido *a priori* pelo utilizador, como é o caso do número de camadas ocultas e de neurónios em cada camada. Assim sendo, embora se faça referência a parâmetros por questões de simplicidade, o que se está de facto a otimizar são os hiperparâmetros dos algoritmos.

6.1. Parametrização do modelo *Random Forests*

6.1.1. Descrição do processo seguido

Devido ao tamanho elevado do *dataset*, nomeadamente do conjunto de treino, bem como ao facto de se pretender otimizar os seis parâmetros apresentados na Tabela 2-3 no modelo *Random Forests*, torna-se muito pouco prática a utilização da técnica de *Grid Search*, uma vez que a mesma consiste em testar todas as combinações possíveis da grelha de parâmetros definida o que levaria a tempos de processamento incomportáveis. Optou-se então pela opção de recorrer em primeiro lugar à técnica de *Random Search*, uma vez que, com um número de iterações adequado mas mesmo assim muito inferior ao *Grid Search*, o mesmo garante uma boa solução, apesar de não encontrar a solução ótima. A técnica de *Grid Search* será utilizada após a otimização inicial com *Random Search* para definir parâmetros concretos. Em relação ao número de iterações a utilizar na *Random Search* encontra-se na bibliografia informação que para qualquer distribuição com um máximo finito, o máximo de 60 observações aleatórias está dentro dos 5% superiores do máximo verdadeiro, com 95%

de probabilidade[31], pelo que se optou por esse valor de todas as vezes que se usar Random Search nesta dissertação e que na realidade irá equivaler a 180 simulações, já que usou uma técnica de validação cruzada com três *folds*.

É importante realçar dois parâmetros que requerem um especial cuidado pois quando definidos de certa forma contribuem para o aumento significativo dos tempos de processamento, sendo eles o *"n_estimators"* e o *"max_samples"*.

O aumento do parâmetro "n_*estimators",* de acordo com alguma bibliografia, contribui para a melhoria das métricas de erro sem no entanto existir risco de *overfit* com um valor demasiado elevado. Por esta razão existem inclusive estudos, como o da referência [55], onde se defende que este valor não deve ser ajustado como os restantes e que seja simplesmente o mais elevado possível de acordo com o poder computacional disponível. Ainda assim nesse mesmo estudo também é dito que embora se defenda que o aumento do número de árvores não possa ser prejudicial, os resultados mostram que para a grande maioria dos *datasets* analisados, o maior ganho de performance foi obtido ao treinar as primeiras 100 árvores, no entanto, essa taxa de convergência pode ser influenciada por outros parâmetros da *Random Forest.*[32] Assim sendo existe um determinado valor para o qual o parâmetro *"n_estimators"* oferecerá um melhor compromisso entre boa performance e tempos de processamento aceitáveis, pelo que se decidiu optar por um valor relativamente elevado igual a 500 inicialmente, sendo depois feita uma procura detalhada com *Grid Search* para encontrar o número onde existe convergência. Escolheu-se o valor de 500 pois a referência [56] menciona os valores 500 e 1000 como valores mais comuns ou *default* para esse parâmetro.

Já em relação ao parâmetro *"max_samples"* este apresenta um comportamento semelhante ao do "n_estimators", no sentido em que se for utilizado um valor alto para o mesmo isso não trará riscos de *overfit*, podendo no entanto um valor mais baixo ser suficiente e oferecer tempos de processamento significativamente mais baixos.

[31] "In hindsight, there is a simple probabilistic explanation for the result: for any distribution over a sample space with a finite maximum, the maximum of 60 random observations lies within the top 5% of the true maximum, with 95% probability."[81]

[32] "Although we claim that increasing the number of trees cannot harm, our empirical results show that, for most of the examined datasets, the biggest performance gain is achieved when training the first 100 trees. However, the rate of convergence may be influenced by other hyperparameters of the RF."[55]

Assim sendo utilizou-se inicialmente o valor *default* (None) para esse parâmetro, em que em cada *split* é utilizado todo o *dataset* e foi feita no final uma segunda procura com *Grid Search* para encontrar a percentagem do *dataset* a utilizar e na qual se verifica convergência.

Em relação aos restantes quatro parâmetros, exemplos de gráficos com a sua influência podem ser vistos também na referência [57]. Os valores a utilizar nestes parâmetros serão definidos após a procura inicial com *Random Search*.

6.1.2. Resultados da otimização inicial com *Random Search*

A grelha utilizada no processo de *Random Search* é visível na Tabela 6-1.

Tabela 6-1-Grelha utilizada na Random Search para o modelo Random Forests

Parâmetro	Valores Testados
n_estimators	500
max_features	1;2;3;4;5;6;7;8;9;10;11;12;13
max_depth	None;2;6;10;15;20;30;40;50;60;70;80;90;100;125;150
min_samples_split	2;4;8;10;25;50;75;100;500;1000;5000;10000
min_samples_leaf	2;4;8;10;25;50;75;100;500;1000;5000;10000
max_samples	None

Os resultados do processo de *Random Search* são apresentados na Tabela 6-2, em termos de valores para os parâmetros e na Tabela 6-3, em termos de tempos de processamento e *score*, o qual corresponde à métrica R^2.

Tabela 6-2-Resultados do processo de Random Search para o modelo Random Forests (parâmetros)

n_estimators	min_samples_split	min_samples_leaf	max_samples	max_features	max_depth
500	50	10	None	8	70

Tabela 6-3-Resultados do processo de Random Search para o modelo Random Forests (tempos de processamento e scoring)

mean_fit_time	std_fit_time	mean_test_score	std_test_score	mean_train_score	std_train_score
156,855	5,598	0,868	0,009	0,929	0,004

6.1.3. Resultados da otimização do parâmetro *"n_estimators"* com *Grid Search*

Utilizando os resultados do processo de *Random Search* para os valores dos restantes parâmetros, efetuou-se de seguida um processo de *Grid Search* para definir o valor de "n_estimators". Os valores testados são apresentados na Tabela 6-4.

Tabela 6-4-Valores testados para n_estimators na 1ª GridSearch do modelo Random Forests

Parâmetro	Valores Testados
n_estimators	10;50;100;250;500;750;1000;1250;1500;2000;2500;3000;4000;5000

Os gráficos obtidos no que respeita à variação da métrica de erro R^2 consoante o número de árvores utilizadas para os processos de treino e validação são visíveis na Figura 6-1 e na Figura 6-2, respetivamente. Ressalva-se que se optou por apresentar estes resultados em dois gráficos diferentes de maneira a facilitar a visualização devido à diferença nas escalas verticais. Como se pode com não aparentam existir de problemas de *overfit* por muito elevados os valores escolhidos, bem como com um ganho de performance maior nas 100 primeiras árvores. No que respeita aos resultados da validação cruzada a convergência aparenta estar próxima do valor de 500 árvores, valor a partir do qual a performance parece estabilizar, pelo que se manteve a utilização deste valor.

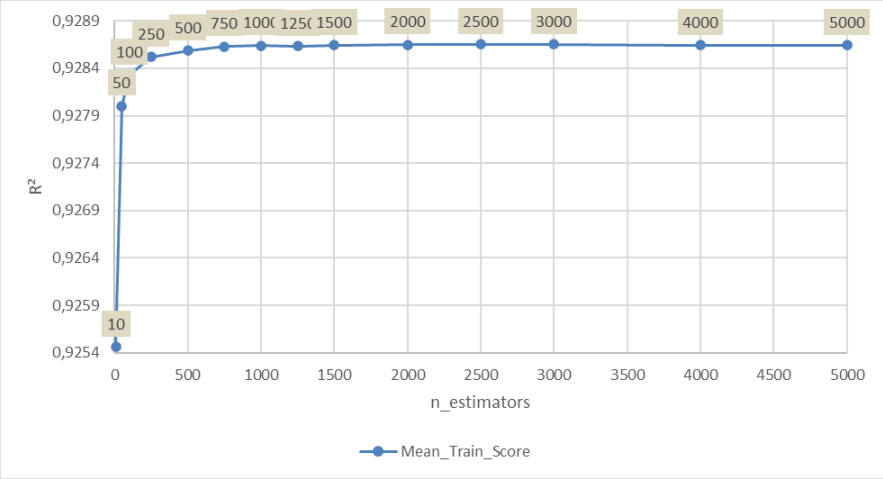


Figura 6-1-Variação do "Mean_Train_Score" com o aumento do número de árvores

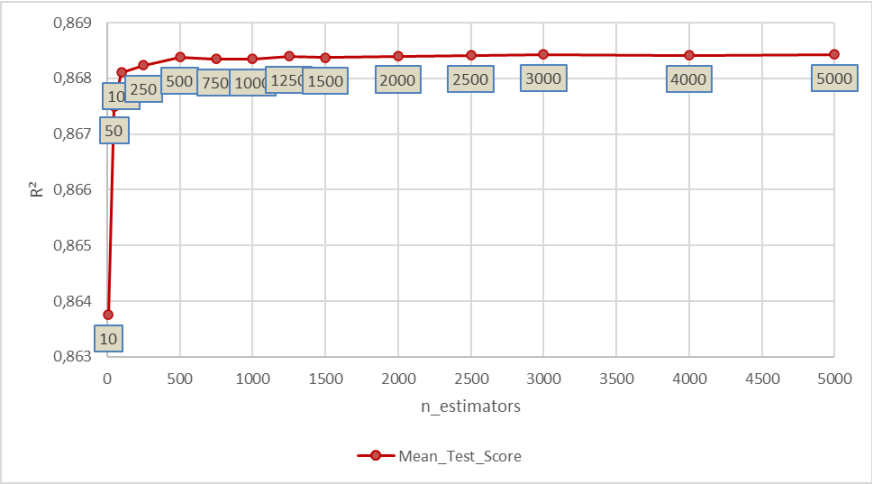


Figura 6-2-Variação do "Mean_Test_Score" com o aumento do número de árvores

Apesar de se ter optado pelo valor de 500 este não foi o valor escolhido pelo processo de *Grid Search*, como se pode ver na Tabela 6-5, onde são apresentados os resultados dos tempos de processamento e do *score* (métrica R^2) obtidos para os diferentes valores do número de árvores. Pode concluir-se que, apesar do valor de 500 ser apenas o sétimo melhor em termos de *score* de entre os 13 testados, este possui um *scoring* do processo de validação cruzada *("mean_test_score")* muito próximo do melhor resultado encontrado, o qual corresponde a 5000 árvores. No entanto, em termos de tempo de processamento gasto no processo de treino *("mean_fit_time")* este foi reduzido de forma muito significativa, pelo que não se justifica a utilização de um valor mais elevado e se confirma que se procedeu corretamente ao não afinar este parâmetro juntamente com os restantes na *Random Search*.

Tabela 6-5-Resultados do 1º processo de Grid Search para o modelo Random Forests (tempos de processamento e scoring)

param_model__n_estimators	rank_test_score	mean_fit_time	std_fit_time	mean_test_score	std_test_score	mean_train_score	std_train_score
10	14	3,274	0,091	0,86375	0,00851	0,92546	0,00441
50	13	15,582	0,512	0,86749	0,00919	0,92800	0,00407
100	12	30,293	0,262	0,86811	0,00918	0,92833	0,00407
250	11	76,603	0,709	0,86824	0,00914	0,92852	0,00412
500	7	154,932	1,702	0,86838	0,00924	0,92859	0,00409
750	9	228,473	2,201	0,86835	0,00926	0,92863	0,00407
1000	10	306,478	2,958	0,86835	0,00922	0,92864	0,00406
1250	6	395,712	5,734	0,86839	0,00924	0,92864	0,00405
1500	8	443,528	14,696	0,86837	0,00925	0,92865	0,00405
2000	5	572,040	2,396	0,86840	0,00928	0,92865	0,00404
2500	3	712,329	2,485	0,86842	0,00928	0,92865	0,00404
3000	2	853,085	4,497	0,86843	0,00927	0,92865	0,00404
4000	4	1139,819	7,653	0,86841	0,00927	0,92864	0,00405
5000	1	1429,505	8,705	0,86843	0,00927	0,92865	0,00405

6.1.4. Resultados da otimização do parâmetro *"max_samples"* com *Grid Search* e modelo final

A última otimização realizada nos parâmetros do modelo *Random Forests* consistiu num processo de *Grid Search* em que se testaram os valores apresentados na Tabela 6-6 para a variável *"max_samples"*.

Tabela 6-6-Valores testados para max_samples na 2ª GridSearch do modelo Random Forests

Parâmetro	Valores Testados
max_samples	0,1;0,2;0,3;0,4;0,5;0,6;0,7;0,8;0,9;None

Os gráficos obtidos no que respeita à variação da métrica de erro R^2 consoante o número de amostras em cada *split*, para os processos de treino e validação, são visíveis na Figura 6-3 e na Figura 6-4, respetivamente. A análise das mesmas permite concluir que existe um aumento constante das métrica referentes ao conjunto de treino enquanto que no caso do conjunto de validação existe uma convergência por volta de um número máximo de amostras de 40%, com um ligeiro pico nos 70%.

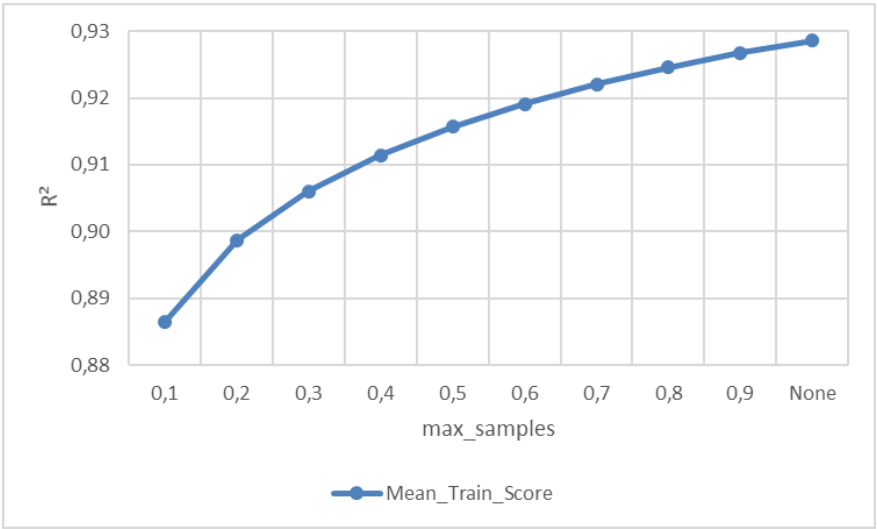


Figura 6-3-Variação da "Mean_Train_Score" com o aumento do número de "max_samples"

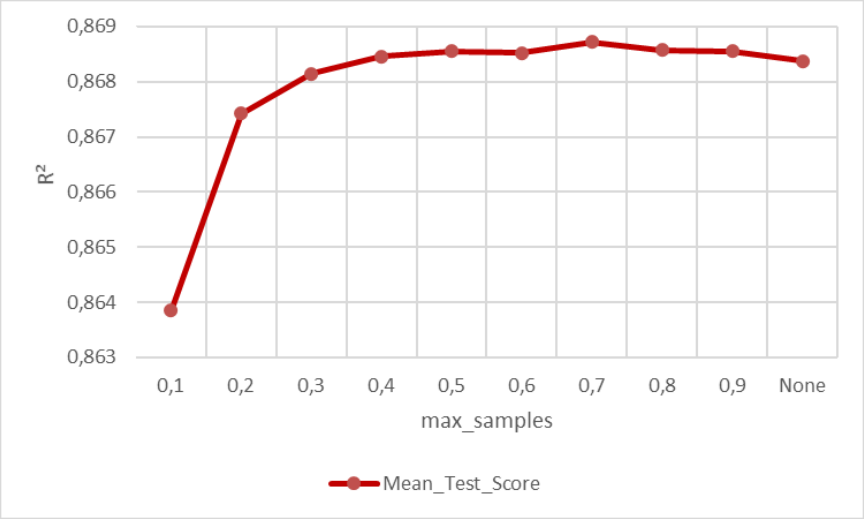


Figura 6-4-Variação da "Mean_Test_Score" com o aumento do número de "max_samples"

Os resultados deste segundo processo de *Grid Search* são apresentados na Tabela 6-7 . Como se pode ver, apesar dos tempos do processo de treino aumentarem com o aumento da percentagem do *dataset* a utilizar em cada *split*, a performance do algoritmo no conjunto de validação visível na coluna *"mean_test_score"* aumenta apenas ligeiramente, pelo que se optou então pelo valor de 40% ou 0,4 por, como foi explicado anteriormente, se verificar um menor aumento de desempenho a partir desse valor.

Tabela 6-7-Resultados do 2º processo de Grid Search para o modelo Random Forests (tempos de processamento e scoring)

param_model_ _max_samples	rank_test_score	mean_fit_time	std_fit_time	mean_test_score	std_test_score	mean_train_score	std_train_score
0,1	10	15,3691	0,0589	0,86385	0,00740	0,88654	0,00580
0,2	9	33,4118	0,1651	0,86742	0,00818	0,89871	0,00551
0,3	8	50,7066	0,6767	0,86815	0,00843	0,90603	0,00517
0,4	6	66,3810	0,3035	0,86846	0,00861	0,91145	0,00492
0,5	4	81,5224	0,2569	0,86855	0,00888	0,91568	0,00474
0,6	5	96,0897	0,8302	0,86852	0,00903	0,91914	0,00456
0,7	1	109,0014	0,6265	0,86872	0,00891	0,92207	0,00443
0,8	2	121,1722	0,6222	0,86857	0,00909	0,92457	0,00430
0,9	3	132,4573	0,6096	0,86856	0,00913	0,92673	0,00416
None	7	142,7404	0,4580	0,86838	0,00924	0,92859	0,00409

6.1.5. Modelo Random Forests final

O modelo *Random Forest* a utilizar na fase de teste terá então os parâmetros da Tabela 6-8 e produziu as métricas e erro exibidas na Tabela 6-9.

Tabela 6-8-Parâmetros do modelo final com Random Forests

n_estimators	min_samples_split	min_samples_leaf	max_samples	max_features	max_depth
500	50	10	0,4	8	70

Tabela 6-9-Métricas de erro obtidas após afinação do modelo Random Forests

Validação				Treino					
MAE [kW]	MSE [kW^2]	RMSE [kW]	MAPE [%]	MAE [kW]	MSE [kW^2]	RMSE [kW]	MAPE [%]	Tempo Treino [s]	Tempo Previsão Treino [s]
33,54	1982,9	44,46	11,37	25,58	1179,4	34,31	8,68	212,29	6,0724

6.2. Parametrização do modelo MARS

6.2.1. Descrição do processo seguido

O algoritmo MARS possui dois parâmetros mais importantes que necessitam de ser afinados: o grau das *features* que são adicionadas ao modelo e o número de termos retidos[33], possuindo por isso uma otimização de parâmetros mais simples do que por exemplo os algoritmos *Random Forests* e Redes Neuronais. Uma vez que em alguma bibliografia, como por exemplo o guia presente na referência [58], o parâmetro "penalty" foi considerado como relevante, o mesmo foi incluído no processo de otimização. Estes parâmetros que serão otimizados são então os apresentados na Tabela 2-4.

Em relação ao parâmetro *"penalty",* Jerome H. Friedman que apresentou pela primeira vez o algoritmo, afirma que para todos os casos estudados o melhor valor para este parâmetro se encontra no intervalo $2 \leq d \leq 4$.[34]

[33] "To summarize, there are two tuning parameters associated with the MARS model: the degree of the features that are added to the model and the number of retained terms." [82]
[34] "Over all situations studied, the best value for d is in the range $2 \leq d \leq 4$."[21]

Quanto ao parâmetro *"degree",* raramente existem benefícios em considerar uma valor superior a 3 [35], pelo que se considerou um valor máximo possível imediatamente acima desse valor, ou seja 4.

Por fim, para o parâmetro *"max_terms"* testaram-se 15 valores diferentes.

Uma vez que este algoritmo possui apenas três parâmetros que necessitam de ser otimizados optou-se por se realizar dois processos de *Grid Search*, o primeiro testando os valores para *"max_terms"* e *"max_degree"* e o segundo, mais curto e apenas para avaliar a influência do parâmetro "penalty".

6.2.2. Resultados da 1ª otimização dos parâmetros *"max_terms"* e *"max_degree"* com *Grid Search*

Os valores testados neste primeiro processo de *Grid Search* são apresentados na Tabela 6-10.

Tabela 6-10-Grelha utilizada na 1ª Grid Search para o modelo MARS

Parâmetro	Valores Testados
max_terms	10;25;50;75;100;150;200;250;300;350;400;450;500;750;1000
max_degree	1;2;3;4
penalty	3

Após correr o primeiro processo obtiveram-se o resultados representados na Figura 6 5. Como se pode ver o valor de quatro para o parâmetro "max_degree" é claramente o que fornece melhores resultados. Já o parâmetro "max_terms", a partir de um valor próximo de 50 deixa de ter influência na métrica R^2, já que os melhores resultados encontrados dizem respeito aos vários valores deste parâmetro a partir do valor de 50, isto é, todas essas simulações atingiram o mesmo de score e o "rank_test_score" de um. Devido a estes resultados realizou-se um novo Grid Search, com o intuito de definir os valores finais para "max_terms" e "max_degree".

[35] "Rarely is there any benefit in assessing greater than 3-rd degree interactions."[83]

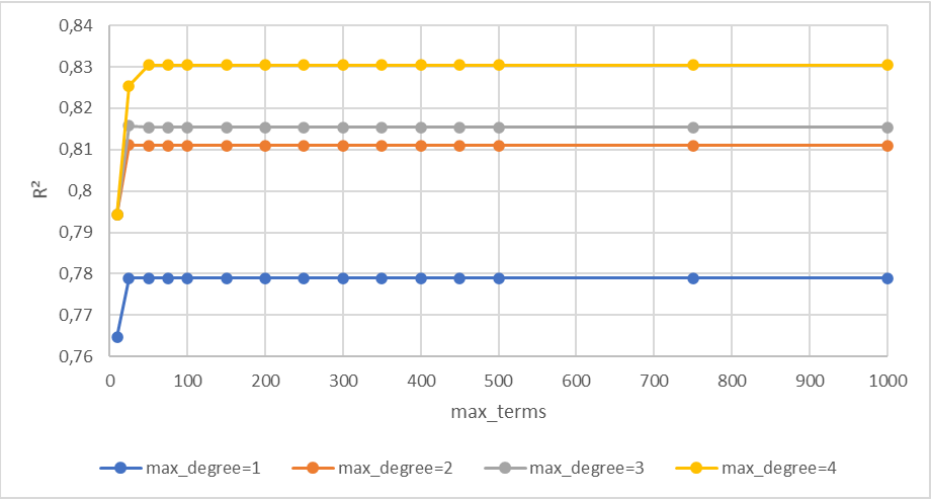


Figura 6-5-Variação da "Mean_Test_Score" com o aumento de "max_terms" e "max_degree"

6.2.3. Resultados da 2ª otimização dos parâmetros *"max_terms"* e *"max_degree"* com *Grid Search*

Os valores testados no segundo processo de otimização dos parâmetros do modelo MARS com *Grid Search* são apresentados na Tabela 6-11.

Tabela 6-11-Grelha utilizada na 2ª Grid Search para o modelo MARS

Parâmetro	Valores Testados
max_terms	27,5;30;32,5;35;37,5;40;42,5;45;47,5;50
max_degree	4;5;6
penalty	3

Os resultados desta simulação são mostrados no gráfico da Figura 6-6. Como se pode ver analisando o gráfico, a partir do valor de 40 para o parâmetro *"max_terms"* deixam de existir aumentos de performance com o aumento do valor deste parâmetro, pelo que se optou por esse valor. Em relação ao parâmetro *"max_degree"* não é ainda claro qual o valor a utilizar uma vez que a performance continuou a aumentar com o aumento deste parâmetro, sendo claramente superior com o valor de seis, pelo que se tomou a decisão de realizar uma terceira procura com *Grid Search*, mais curta e apenas com o objetivo de definir o valor final de *"max_degree"*.

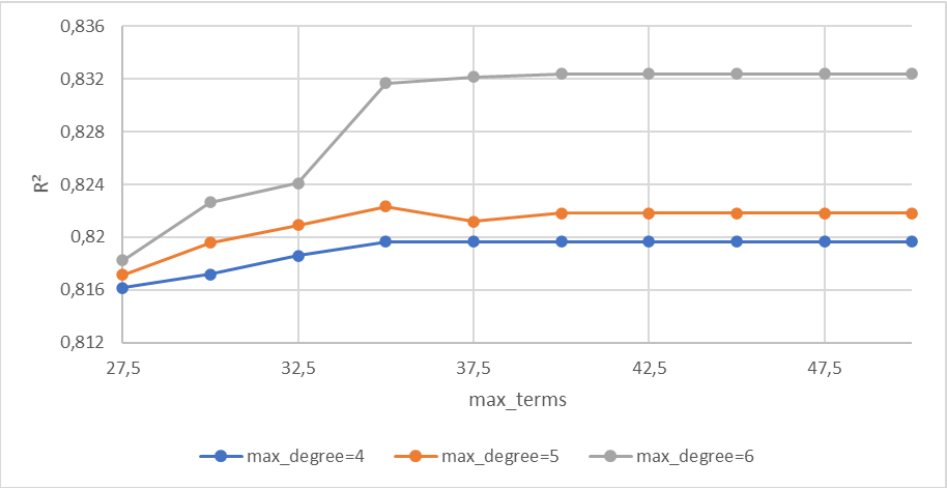


Figura 6-6-Variação da "Mean_Test_Score" com o aumento de "max_terms" e "max_degree" (2)

6.2.4. Resultados da otimização final do parâmetro "*max_degree*" com *Grid Search*

De seguida, com o valor definido anteriormente para *"max_terms"* e continuando a usar o valor predefinido para *"penalty"* realizou-se uma curta procura com *Grid Search* em que se testaram mais dois valores diferentes para *"max_degree"* apresentados na Tabela 6-12.

Tabela 6-12-Grelha utilizada na 3ª Grid Search para o modelo MARS

Parâmetro	Valores Testados
max_terms	40
max_degree	6;7;8
penalty	3

Como se pode ver no gráfico da Figura 6-7 onde são apresentados os resultados do processo, a partir do valor de sete deixa de haver um aumento da métrica R^2 com aumentos de *"max_degree"* pelo que se optou pela utilização desse valor. Apesar de se esperar que um valor de sete para *"max_degree"* leve a grandes aumentos dos tempos de processamento deste modelo decidiu-se na mesma prosseguir com esse valor pois o mesmo aparenta ter um impacto forte na performance do modelo de previsão.

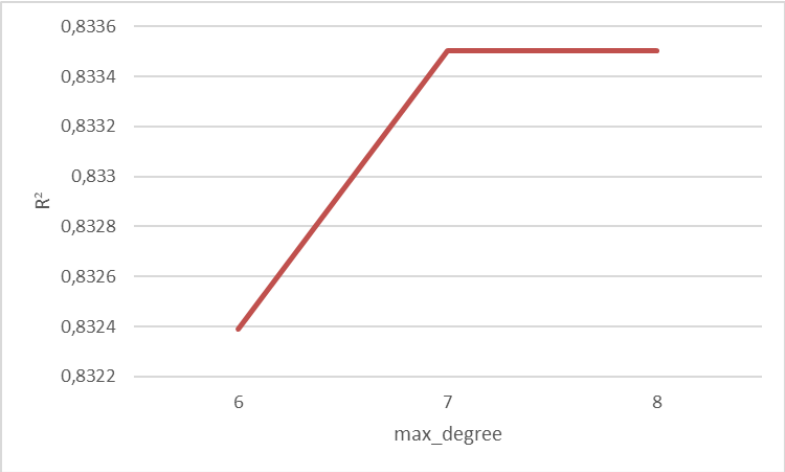


Figura 6-7-Variação da "Mean_Test_Score" com o aumento de "max_degree"

6.2.5. Resultados da otimização do parâmetro "penalty" com Grid Search

Como último processo de otimização de parâmetros do modelo MARS realizou-se uma procura com Grid Search com os valores visíveis na Tabela 6-13 com o objetivo de avaliar o impacto do parâmetro "penalty" e de definir um valor para o mesmo.

Tabela 6-13-Grelha utilizada na 4ª Grid Search para o modelo MARS

Parâmetro	Valores Testados
max_terms	40
max_degree	7
penalty	2;2,5;3;3,5;4

Na Figura 6-8 é apresentado um gráfico que representa os resultados do processo. Pela sua análise pode constatar-se que o impacto de "penalty" na métrica R^2 é reduzido mas que, ainda assim, é preferível selecionar um valor de 2,5 para o mesmo, já que se verifica um ligeiro declínio de performance a partir desse valor.

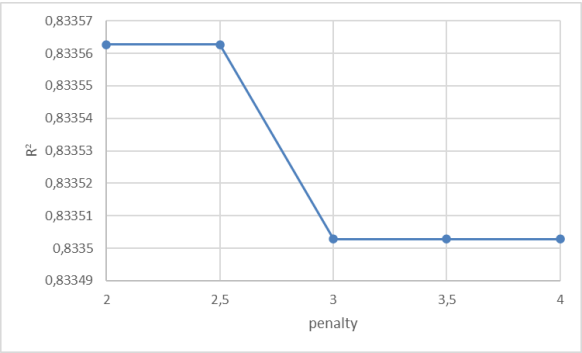


Figura 6-8-Variação da Mean_Test_Score com o aumento de "penalty"

6.2.6. Modelo MARS final

O modelo MARS a utilizar na fase de teste terá os parâmetros expostos na Tabela 6-14 e o mesmo resultou nas métricas e tempos apresentados na Tabela 6-15.

Tabela 6-14-Parâmetros do modelo final com MARS

max_terms	max_degree	penalty
40	7	2,5

Tabela 6-15-Métricas de erro obtidas após afinação do modelo MARS

Validação				Treino					
MAE [kW]	MSE [kW^2]	RMSE [kW]	MAPE [%]	MAE [kW]	MSE [kW^2]	RMSE [kW]	MAPE [%]	Tempo Treino [s]	Tempo Previsão Treino [s]
36,37	2507,1	50,05	12,35	35,45	2343,9	48,35	12,06	995,84	0,2156

6.3. Parametrização do modelo SVM

Um problema do algoritmo SVM é o facto da complexidade do tempo de treino deste algoritmo ser mais de quadrática com o número de amostras, o que torna difícil aplicar o algoritmo a conjuntos de dados com mais de algumas dezenas de milhar de amostras. Para estes casos, no qual o *dataset* utilizado nesta dissertação se enquadra, a documentação do *scikit-learn* recomenda antes a utilização dos modelos "*LinearSVR"* e "*SGDRegressor"* [36].

Optou-se então por recorrer ao modelo "*LinearSVR"* uma vez que ao tentar utilizar o modelo *"SVR"* este levou a tempos de processamento incomportáveis, impossibilitando a otimização dos parâmetros e comprovando o que foi descrito anteriormente. A otimização foi então focada nos parâmetros C, o qual consiste num termo de regularização, e o parâmetro épsilon, relacionado com a tolerância do modelo aos erros, utilizando um processo de *Random Search*, já que os restantes parâmetros visíveis na Tabela 2-2 não estão presentes no modelo "*LinearSVR*". Este modelo é mais rápido devido a utilizar um *kernel* linear, pelo que é de esperar que não apresente resultados tão positivos, uma vez que não se adaptará tão bem a não linearidades.

[36] "The fit time complexity is more than quadratic with the number of samples which makes it hard to scale to datasets with more than a couple of 10000 samples. For large datasets consider using LinearSVR or SGDRegressor instead, possibly after a Nystroem transformer."[14]

Os valores utilizados na grelha utilizada na otimização são apresentados na Tabela 6-16. Estes valores foram escolhidos tendo em conta valores utilizados na bibliografia, mais concretamente na referência [59].

Utilizou-se também um número máximo de iterações mais elevado e igual a 1000000, pois o valor *default* não era suficiente para se verificar convergência.

Tabela 6-16-Grelha utilizada na otimização do modelo LinearSVR com Random Search

Parâmetro	Valores Testados
C	1;5;30;50;100;150;200;250;300;350;400;450;500;1000…15000
epsilon	$10^x, x = \{-3, -2.5, -2, ..., -0.5\}$

Os melhores valores encontrados são apresentados na Tabela 6-17 e produziram as métricas mostradas na Tabela 6-18.

Tabela 6-17-Parâmetros do modelo final com Linear SVR

Parâmetro	Valores Testados
C	450
epsilon	0,031623

Tabela 6-18-Métricas de erro obtidas após afinação do modelo LSVM

Validação				Treino					
MAE [kW]	MSE [kW^2]	RMSE [kW]	MAPE [%]	MAE [kW]	MSE [kW^2]	RMSE [kW]	MAPE [%]	Tempo Treino [s]	Tempo Previsão Treino [s]
40,50	3278,0	57,23	13,50	40,23	3248,0	56,99	13,41	2859,22	0,1095

6.4.Parametrização do modelo ANN

Como já foi mencionado no capítulo 2, o algoritmo das redes neuronais possui muitos parâmetros que necessitam de ser otimizados, tendo-se optado por dar ênfase aos parâmetros apresentados na Tabela 2-1, pelo que se decidiu realizar essa otimização com o algoritmo *Random Search*.

6.4.1. Número de camadas ocultas e número de neurónios

Em relação ao intervalo de valores considerado para cada parâmetro, o caso mais particular é o do número de camadas ocultas e de neurónios em cada camada. É possível encontrar na bibliografia diversas *rules of thumb*, bem como técnicas mais complexas, utilizadas para otimizar estes parâmetros, no entanto nenhum autor conseguiu ainda encontrar uma fórmula ideal para tal, levando à diminuição do tempo de treino e aumento de performance das redes neuronais[37]. No que diz respeito ao número de camadas ocultas, é referido em [60] que duas ou menos camadas geralmente são suficientes com conjuntos de dados simples, no entanto para *datasets* complexos envolvendo séries temporais ou visão computacional, mais camadas podem ser úteis[38]. Por outro lado, em [61], é dedicado um capítulo a detalhar o porquê de não se utilizarem quatro camadas ocultas. Tendo isto em conta considerou-se a utilização de até três camadas ocultas, uma vez que o *dataset* em causa utiliza séries temporais longas, pelo que a utilização de uma terceira camada oculta deverá ser avaliada. De entre as *rules of thumb* mais comuns na bibliografia para definir o número de neurónios nas camadas ocultas, existem três que são mais vezes encontradas na bibliografia, por exemplo nas referências [61] e [62], as quais abordam esta temática com algum detalhe. sendo elas*:*

- O número de neurónios nas camadas ocultas deve ser entre o tamanho da camada de entrada e o da camada de saída[39];
- O número de neurónios nas camadas ocultas deve ser $\frac{2}{3}$ o tamanho da camada de entrada, mais o tamanho da camada de saída[40];
- O número de neurónios nas camadas ocultas não deve ser mais de duas vezes o tamanho da camada de entrada[41].

[37] "…unfortunately nobody succeed in finding the optimal formula for calculating the number of neurons that should be kept in the hidden layer so that the neural network training time can be reduced and also accuracy in determining target output can be increased."[61]
[38] "Two or fewer layers will often suffice with simple data sets. However, with complex datasets involving time-series or computer vision, additional layers can be helpful."[60]
[39] "The number of hidden neurons should be between the size of the input layer and the size of the output layer".[84]
[40] "The number of hidden neurons should be 2/3 the size of the input layer, plus the size of the output layer."[60]
[41] ""How large should the hidden layer be? One rule of thumb is that it should never be more than twice as large as the input layer.."[85]

Estas regras apesar de poderem servir de ponto de partida não devem ser seguidas cegamente, devendo sempre a escolha ser feita a partir de tentativa e erro, já que estas *rules of thumb* são muito controversas e não estão de todo comprovadas. O artigo da referência [63] apresenta várias outras regras e fórmulas encontradas na bibliografia para definir este parâmetro. Outro ponto a ter em conta, relacionada com a primeira *rule of thumb* apresentada anteriormente, é o facto de tendencialmente nos modelos de redes neuronais encontrados na bibliografia o número de neurónios diminuir ou pelo menos ser igual ao longo das camadas, sendo na referência [64] apresentadas várias razões para tal. Assim sendo testaram-se para o parâmetro os "hidden_layer_sizes" os valores 5,10,25,50,75,100,150 e 200 para até três camadas, com a restrição de a camada em causa não ter mais neurónios que a anterior.

Para o caso de se utilizar apenas uma camada oculta, testou-se também os valores 10 e 28, indo ao encontro da segunda e terceira *rules of thumb* apresentadas anteriormente, uma vez que a camada de entrada possui 14 neurónios (um para cada *feature* mais um para o bias) e a de saída um. Incluiu-se também na grelha a possibilidade de não existência de camada oculta, que na prática equivale a utilização do algoritmo de regressão logística no caso de se utilizar o solver *logistic*.

6.4.2. Restantes parâmetros

Quanto ao parâmetro solver, incluíram-se na grelha o "sgd" e o "adam", tendo-se optado, por deixar de fora o "L-BFGS" uma vez que, de acordo com a documentação, este converge mais rápido e com melhores soluções em conjuntos de dados pequenos[42], o que não é o caso do *dataset* utilizado na presente dissertação. Em relação os parâmetros "*activation*" e "*learning_rate*" incluíram-se na grelha todos os valores possíveis que esses parâmetros podem tomar, no entanto é necessário realçar que este último é apenas relevante no caso de se usar o solver "sgd".

Um parâmetro particularmente importante é o "*alpha*", o qual constitui um termo de regularização, que combate os problema de *overfit* e *underfit* restringindo o tamanho dos pesos. Aumentar o seu valor pode corrigir a variância elevada, relacionada com *overfit*, ao encorajar pesos mais pequenos que resultam em fronteiras de decisão com menos curvaturas.

[42] "Empirically, we observed that L-BFGS converges faster and with better solutions on small datasets."[86]

Por outro lado, a sua diminuição pode corrigir o *bias* elevado, relacionado com *underfit*, ao encorajar pesos maiores, que resultam em fronteiras de decisão mais complicadas.[43] A documentação aconselha que este parâmetro seja otimizado com um processo de GridSearch, normalmente no intervalo de 10^{-x}, com X de um a sete[44].

Finalmente para o parâmetro "learning_rate_init", encontrou-se na bibliografia informação de que valores típicos no caso de redes neuronais com entradas normalizadas são inferiores a 1 e maiores que 10^{-6}, pelo que se optou pela utilização deste intervalo de valores.[45]

Como o parâmetro da taxa de aprendizagem inicial, de acordo com a bibliografia[46], é particularmente importante, e como a documentação, como já foi referido, aconselha a otimização com *Grid Search* do parâmetro "alpha", optou-se por, em primeiro lugar, aplicar o método de *Random Search*, com a grelha visível na Tabela 6-19, para encontrar os valores a adotar nos restantes parâmetros, efetuando-se de seguida uma *Grid Search*, cuja grelha é apresentada na Tabela 6-20, para definir então os parâmetros "alpha" e "learning_rate_init".

Tabela 6-19-Grelha utilizada na Random Search para o modelo ANN

Parâmetro	Valores Testados
solver	Sgd; adam
hidden_layer_sizes	(Valores descritos anteriormente, totalizando 148 combinações)
activation	Logistic; tanh; relu; identity
learning_rate	Constant; invscaling; adaptive

Tabela 6-20-Grelha utilizada na Grid Search para o modelo ANN

Parâmetro	Valores Testados
learning_rate_init	$[10^{0};10^{-1};10^{-2};10^{-3};10^{-4};10^{-5};10^{-6}]$
alpha	$[10^{0};10^{-1};10^{-2};10^{-3};10^{-4};10^{-5};10^{-6};10^{-7}]$

[43] "Alpha is a parameter for regularization term, aka penalty term, that combats overfitting by constraining the size of the weights. Increasing alpha may fix high variance (a sign of overfitting) by encouraging smaller weights, resulting in a decision boundary plot that appears with lesser curvatures. Similarly, decreasing alpha may fix high bias (a sign of underfitting) by encouraging larger weights, potentially resulting in a more complicated decision boundary."[87]

[44] "Finding a reasonable regularization parameter alpha is best done using GridSearchCV, usually in the range 10.0 ** -np.arange(1, 7)."[86]

[45] "Typical values for a neural network with standardized inputs (or inputs mapped to the (0,1) interval) are less than 1 and greater than 10−6."[88]

[46] "This is often the single most important hyperparameter and one should always make sure that it has been tuned."[88]

Os melhores valores encontrados durante os dois processos de otimização são mostrados na Tabela 6-21 e resultaram nas métricas da Tabela 6-22.

Tabela 6-21-Parâmetros do modelo final com ANN

Parâmetro	Valor
solver	adam
hidden_layer_sizes	(200, 200, 50)
activation	logistic
learning_rate	adaptive
learning_rate_init	0,01
alpha	0,01

Tabela 6-22-Métricas de erro obtidas após afinação do modelo ANN

Validação				Treino					
MAE [kW]	MSE [kW^2]	RMSE [kW]	MAPE [%]	MAE [kW]	MSE [kW^2]	RMSE [kW]	MAPE [%]	Tempo Treino [s]	Tempo Previsão Treino [s]
37,46	2524,5	50,16	12,57	24,42	1079,3	32,81	8,27	200,06	0,9987

6.5.Análise de resultados final

Tendo os modelos afinados, procedeu-se de seguida à simulação dos mesmos no conjunto de teste, que como já foi referido anteriormente foi deixado de parte até aqui, por forma a não levar a resultados tendenciosos, ou seja a problemas de *data leakage*, como é recomendado pelas boas práticas de *machine learning*.

6.5.1. Métricas de erro finais

Uma análise importante que se deve fazer é a da qualidade das previsões nos diferentes períodos, isto é, nos diferentes valores que a variável "Tipo Dia" pode tomar. Para tal calcularam-se as métricas de erro, não apenas para todo o período de teste, mas também em função dos períodos definidos para essa variável, tendo-se chegado à Tabela 6-23, que para além disso destaca também os melhores resultados encontrados. Adicionou-se formatação condicional por linha para facilitar a visualização dos resultados.

Tabela 6-23-Métricas de erro obtidas no teste final para os diferentes valores da variável "Tipo Dia"

		Tipo Dia								
Modelo	**Métrica**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**1-8**
		Dias de Aulas	**Sábados**	**Domingos e Feriados**	**Épocas de Exames**	**Férias de Natal**	**Férias da Páscoa**	**Féria de Verão**	**Outras Interrupções**	**Todos os Períodos**
Total de dias →		140	52	65	59	8	4	34	4	366
Persistência	MAE [kW]	48,48	34,37	48,09	53,47	102,40	98,74	46,75	31,80	48,59
	MSE [kW²]	4909,1	2179,3	6771,3	6169,5	21975,7	13573,4	4206,4	1566,8	5421,1
	RMSE [kW]	70,07	46,68	82,29	78,55	148,24	116,50	64,86	39,58	73,63
	MAPE [%]	13,38	15,50	24,61	14,77	43,18	37,88	17,39	11,95	17,18
Random Forests	MAE [kW]	31,13	27,53	27,32	31,66	39,53	41,93	33,60	24,11	30,48
	MSE [kW²]	1646,2	1280,9	1253,5	1875,9	3195,0	2355,3	1734,4	908,4	1603,3
	RMSE [kW]	40,57	35,79	35,40	43,31	56,52	48,53	41,65	30,14	40,04
	MAPE [%]	8,92	12,58	14,18	8,92	18,23	17,19	13,02	9,08	11,05
MARS	MAE [kW]	32,97	28,71	35,61	31,50	70,15	55,62	32,29	42,56	33,70
	MSE [kW²]	1843,2	1411,8	2719,1	1854,4	9132,8	4008,6	1716,7	2575,9	2118,6
	RMSE [kW]	42,93	37,57	52,14	43,06	95,57	63,31	41,43	50,75	46,03
	MAPE [%]	9,57	12,54	18,70	8,85	31,20	22,36	12,10	15,73	12,41
ANN	MAE [kW]	32,04	27,21	26,62	29,93	47,48	40,79	34,04	28,97	30,64
	MSE [kW²]	1717,5	1218,5	1142,9	1628,1	4997,1	2228,4	1769,7	1386,8	1608,8
	RMSE [kW]	41,44	34,91	33,81	40,35	70,69	47,21	42,07	37,24	40,11
	MAPE [%]	9,23	12,45	13,89	8,81	21,63	16,36	13,26	10,79	11,19
LSVM	MAE [kW]	37,42	28,04	42,77	37,73	70,50	60,91	35,45	24,69	37,75
	MSE [kW²]	2571,7	1323,7	5042,0	2616,2	9594,5	5309,6	2098,0	913,8	2961,6
	RMSE [kW]	50,71	36,38	71,01	51,15	97,95	72,87	45,80	30,23	54,42
	MAPE [%]	10,43	12,68	21,90	10,46	31,44	23,58	13,19	9,14	13,64
Melhor Resultado	MAE [kW]	31,13	27,21	26,62	29,93	39,53	40,79	32,29	24,11	30,48
	MSE [kW²]	1646,16	1218,47	1142,94	1628,13	3195,01	2228,45	1716,68	908,37	1603,30
	RMSE [kW]	40,57	34,91	33,81	40,35	56,52	47,21	41,43	30,14	40,04
	MAPE [%]	8,92	12,45	13,89	8,81	18,23	16,36	12,10	9,08	11,05
Pior Resultado	MAE [kW]	48,48	34,37	48,09	53,47	102,40	98,74	46,75	42,56	48,59
	MSE [kW²]	4909,12	2179,27	6771,35	6169,51	21975,73	13573,36	4206,44	2575,88	5421,10
	RMSE [kW]	70,07	46,68	82,29	78,55	148,24	116,50	64,86	50,75	73,63
	MAPE [%]	13,38	15,50	24,61	14,77	43,18	37,88	17,39	15,73	17,18
Melhor Modelo	MAE [kW]	Random Forests	ANN	ANN	ANN	Random Forests	ANN	MARS	Random Forests	Random Forests
	MSE [kW²]	Random Forests	ANN	ANN	ANN	Random Forests	ANN	MARS	Random Forests	Random Forests
	RMSE [kW]	Random Forests	ANN	ANN	ANN	Random Forests	ANN	MARS	Random Forests	Random Forests
	MAPE [%]	Random Forests	ANN	ANN	ANN	Random Forests	ANN	MARS	Random Forests	Random Forests
Pior Modelo	MAE [kW]	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	MARS	Persistencia
	MSE [kW²]	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	MARS	Persistencia
	RMSE [kW]	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	MARS	Persistencia
	MAPE [%]	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	Persistencia	MARS	Persistencia

Várias conclusões importantes podem ser retiradas da tabela anterior. A que salta à vista é a de que os todos os modelos apresentam uma pior performance no período das férias de Natal e da Páscoa de acordo com todas as métricas.

Em relação aos dias de aulas e épocas de exames, que acabam por ser os mais importantes já que representam os períodos de maior atividade do Campus, a métrica MAPE é bastante boa em todos os modelos, estando inclusive abaixo dos 10% e mesmo dos 9% em alguns casos, mas tal não acontece com as restantes métricas.

Tal pode ser explicado pelo facto de, durante os dias de aulas e exames a potência ativa tomar valores muito mais altos que nos restantes períodos, pelo que os erros nos períodos de maior atividade irão afetar mais as métricas de erro absoluto (MAE, MSE e RMSE), razão

pela qual a métrica MAPE é mais fiável no que toca a comparar o desempenho dos modelos em diferentes períodos.

Os diagramas de carga que serão apresentados de seguida irão demonstrar isto mesmo, que ao contrário do que as métricas de erro absoluto fazem parecer, os modelos apresentam boas previsões nesses períodos.

Olhando então para o MAPE, verifica-se que, para além dos dias de aulas e épocas de exames, os modelos, com exceção do MARS, apresentam bons resultados nos quatro dias correspondentes a outras interrupções. Os sábados apresentam um MAPE aceitável e sempre melhor que o dos domingos e Feriados, tendo o MARS e LSVM neste último caso um MAPE já bastante elevado de mais de 18%. Por fim para o período das Férias de Verão os valores do MAPE atingidos são razoáveis, não chegando a 14% em nenhum dos modelos.

Quanto às métricas considerando todo o período de teste, é importante realçar que os modelos não aparentam ter problemas de *overfit*, pois obtiveram-se métricas semelhantes às que tinham sido obtidas no final dos processos de otimização de parâmetros, o que indica que possuem boa capacidade de generalização. Também é visível que as melhores métricas foram as dos modelos *Random Forests*, com as ANN com valores ligeiramente inferiores. Pode ver-se ainda que os modelos *Random Forests* e ANN foram o melhor por 16 vezes cada um e o MARS por quatro, tendo todas elas sido no período das férias de Verão. Em relação às piores performances e utilizando como *baseline* o modelo de persistência, apenas por quatro vezes este não foi o que apresentou as piores métricas, tendo tal acontecido no período de outras interrupções, em que o modelo MARS foi o pior.

6.5.2. Diagramas de carga real *vs* previsto

Outra ferramenta importante na análise de performance de modelos de previsão são os diagramas de carga em que se comparam as previsões com os resultados reais, permitindo aferir sobre as semelhanças entre ambos. Nos artigos analisados este é um dos métodos de avaliação mais presentes e comuns, sendo por exemplo utilizado na referência [51] no contexto de uma previsão num edifício de escritórios.

Em relação à presente dissertação, na Figura 6-9 é apresentado o diagrama de carga com a potência ativa real e a prevista pelos diferentes modelos, para a semana de 1 a 7 de Novembro de 2019, tendo-se escolhido esta semana para se poder visualizar o comportamento dos modelos num feriado.

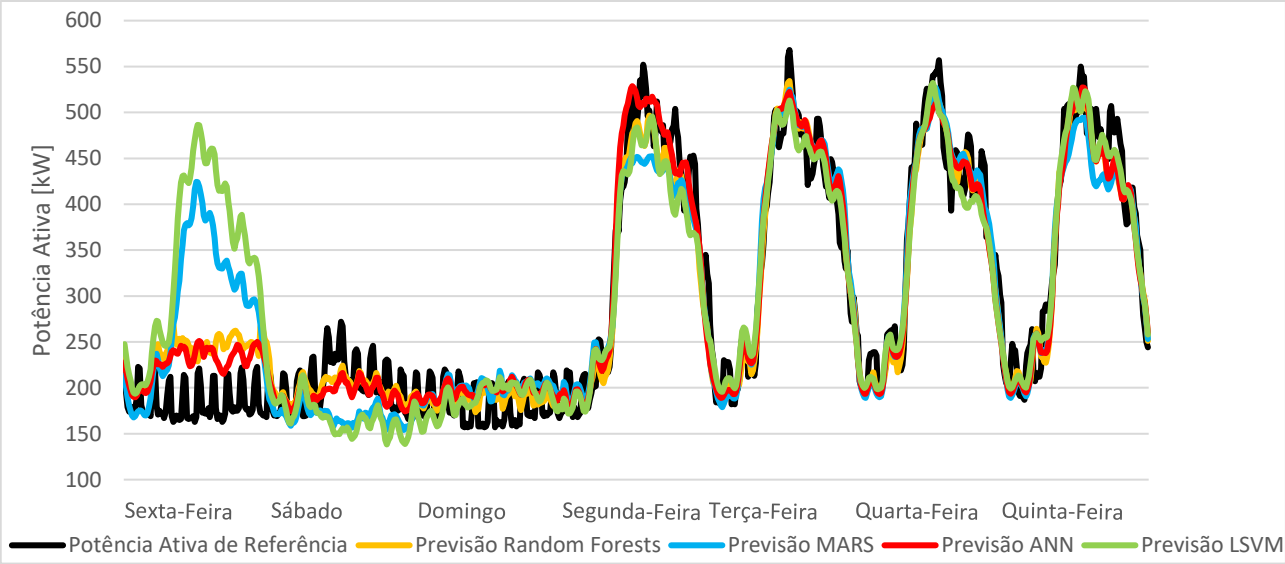


Figura 6-9-Diagrama de carga potência ativa real vs prevista (semana 1/11 a 7/11 de 2019)

Como se pode ver os modelos aparentam conseguir prever com bastante rigor as potências dos quatro dias de aulas apresentados. Em relação ao fim de semana a grande variabilidade da potência real aparenta ter dificultado as previsões, que no entanto ainda assim conseguem, principalmente no Domingo, prever valores dentro daqueles que a potência real tomou nesse dia. No sábado os modelos MARS e LSVM não conseguiram "seguir" os valores reais quando estes tomaram valores mais elevados. Por fim, em relação ao feriado apresentado, correspondeu ao dia exibido onde os modelos apresentaram pior performance, prevendo valores muito superiores aos reais, ainda que menores do que num dia normal de aulas, o que mostra a influência da variável "Tipo Dia", que conseguiu transmitir essa informação aos modelos. Neste último caso os modelos MARS e LSVM apresentaram uma performance bastante negativa.

Na Figura 6-10 e Figura 6-11 são apresentados os diagramas de carga para a semana do Natal e da Páscoa, respetivamente, sendo que os quatro dias referentes ao período das férias da Páscoa no que a variável "Tipo Dia" diz respeito são os quatro primeiros, visto que sexta-feira foi feriado. Os resultados dos gráficos vão ao encontro do que foi obtido na Tabela 6-23 no sentido em que os modelos obtiveram uma performance fraca nestes períodos. No entanto os resultados referentes às férias da Páscoa não são tão negativos quanto as métricas faziam transparecer, já que as previsões apesar de não seguirem com exatidão as potências de referência aparentam acompanhar melhor os picos desses valores.

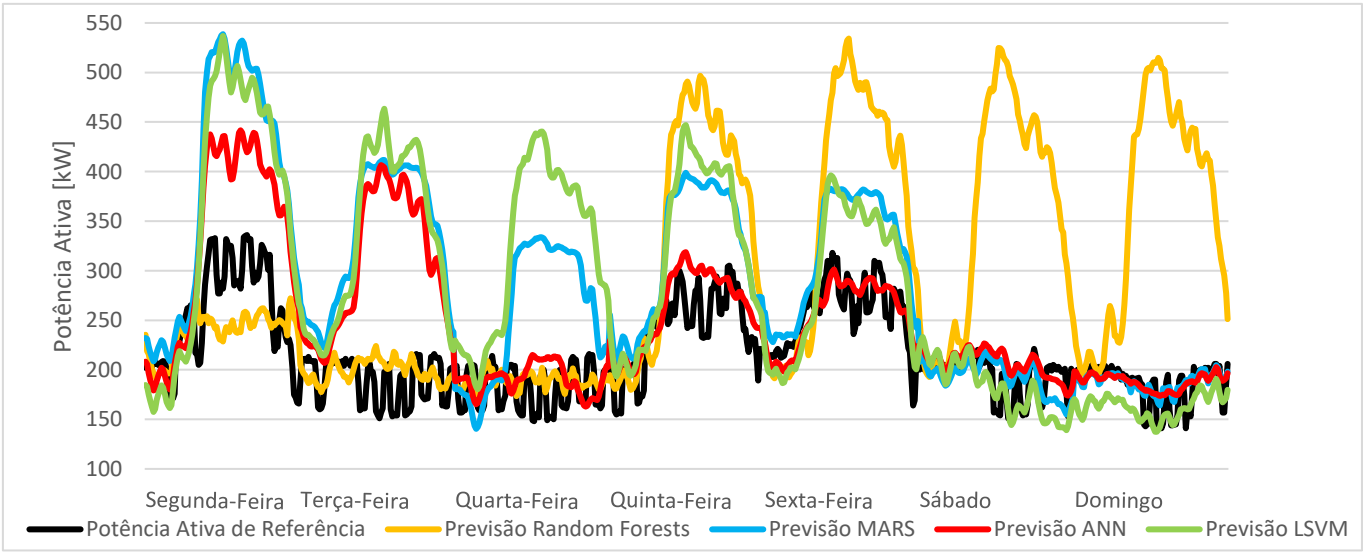


Figura 6-10-Diagrama de carga potência ativa real vs prevista (semana 23/12 a 29/12 de 2019)

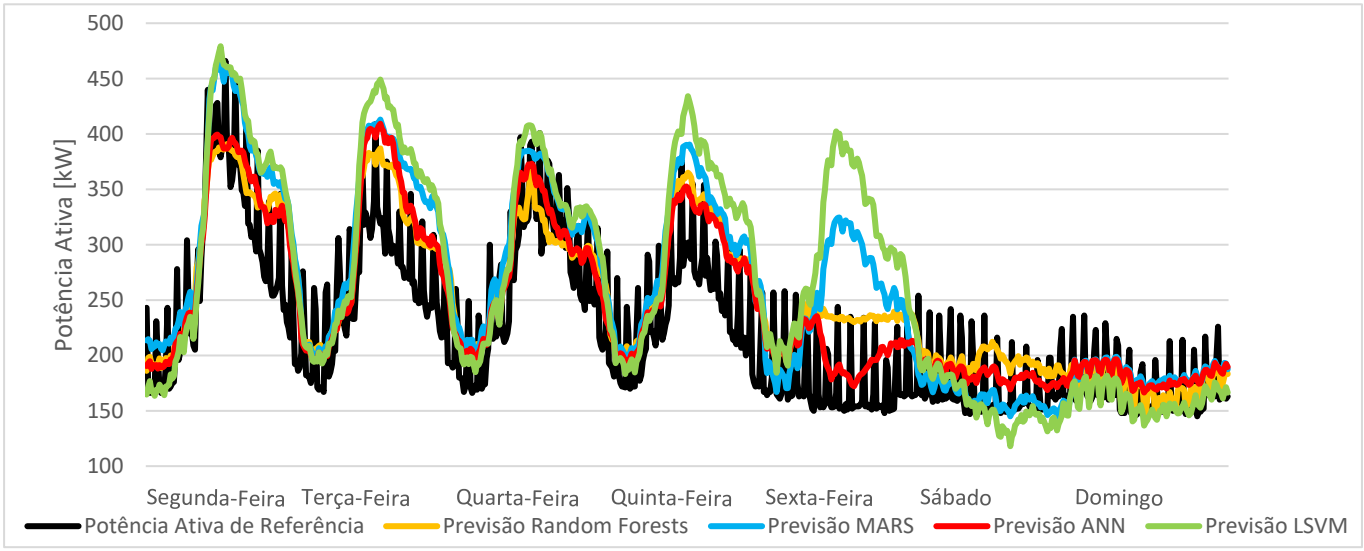


Figura 6-11-Diagrama de carga potência ativa real vs prevista (semana 15/04 a 21/04 de 2019)

Finalmente, em relação ao desempenho dos modelos nos períodos de avaliação e durante as férias de Verão, este pode ser visto na Figura 6-12 e na Figura 6-13, respetivamente. Os resultados são bons, tal como tinha sido indiciado pela Tabela 6-23, no entanto a performance nos períodos de avaliação não é tão elevada como as métricas indicavam, já que os constantes picos de potência atingidos não são detetados pelos modelos, o que indica que o melhor desempenho dos modelos poderá ter sido atingido nos períodos de aulas e não nos de avaliações. Por outro lado, o desempenho durante as férias de Verão aparenta ser um pouco melhor que aquele indicado pelas métricas, já que também aqui apenas esses valores máximos não foram detetados. No entanto, é preciso ter em conta que apenas se está a mostrar uma semana no gráfico.

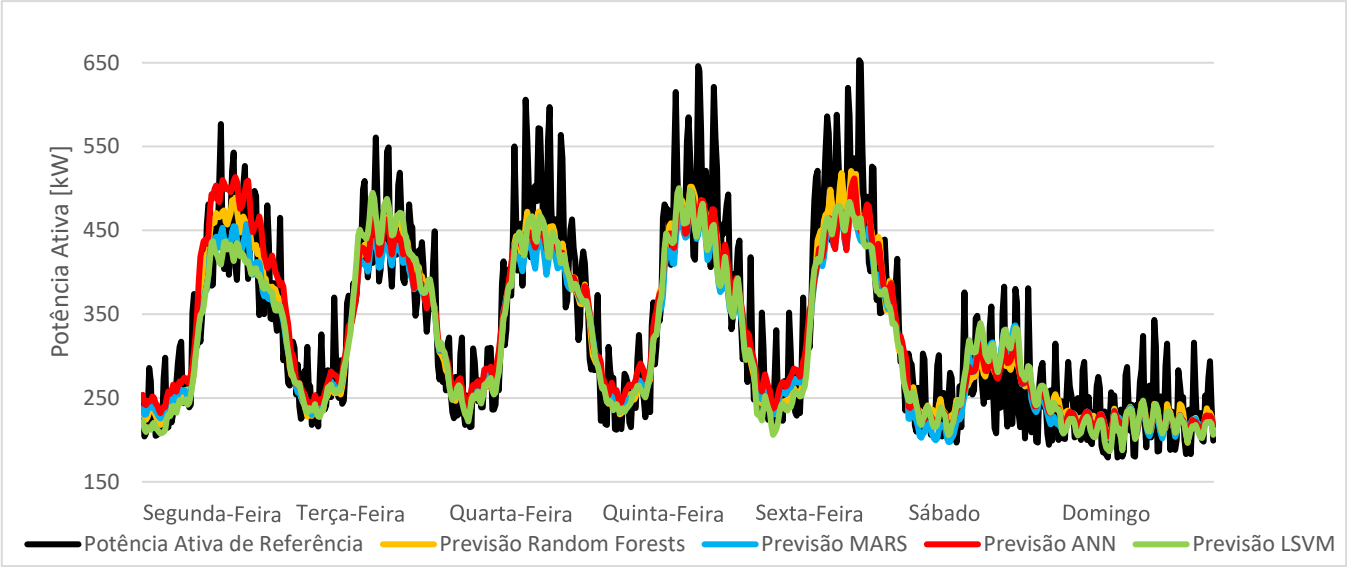


Figura 6-12-Diagrama de carga potência ativa real vs prevista (semana 08/07 a 14/07 de 2019)

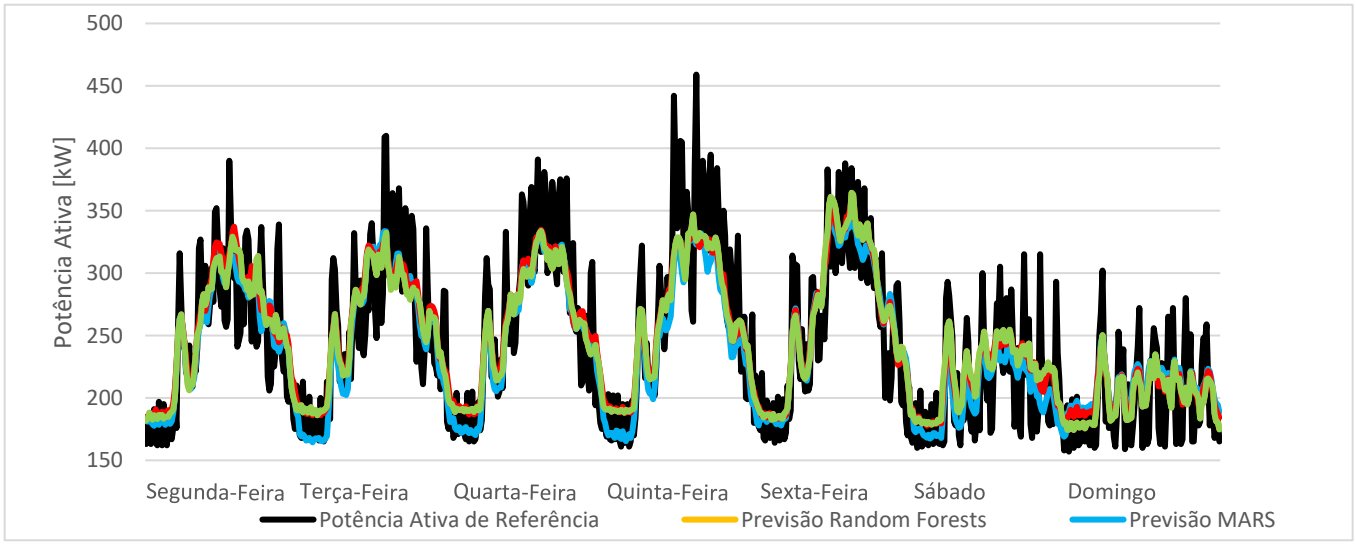


Figura 6-13-Diagrama de carga potência ativa real vs prevista (semana 19/08 a 25/08 de 2019)

6.5.3. Mapas térmicos do consumo diário de energia

Os diagramas de carga apresentados anteriormente, apesar de serem uma ferramenta importante na avaliação da qualidade dos modelos, possuem o problema de permitirem apenas a visualização de alguns dias de cada vez. Posto isto, com o intuito de visualizar a performance dos modelos em todo o período de teste de uma só vez, criaram-se os mapas térmicos apresentados no Anexo I, indo ao encontro do que foi encontrado na bibliografia, mais concretamente no artigo da referência [62].

A unidade representada nos mapas apresentados no anexo é o consumo diário de energia calculado com base nos valores de potência ativa reais e previstos.

A visualização destes mapas permite reforçar a ideia de que os modelos conseguiram captar os pontos essenciais do perfil de consumo do Campus, já que em todos eles pode ver-se que, tal como acontece no mapa feito com os dados reais, os consumos são claramente menores aos fins de semana e, em termos mensais, no mês de agosto seguido de fevereiro, com os maiores consumos a acontecerem principalmente nos últimos meses do ano.

6.5.4. Gráficos de dispersão

Outra ferramenta útil para a avaliação da qualidade de um modelo de *machine learning* são os gráficos de dispersão, os quais na representação gráfica de pares de dados permitem determinar visualmente com facilidade o tipo de correlação existente entre variáveis. Em termos de modelos de previsão de consumos, estes gráficos são habitualmente usados para analisar a relação entre os consumos previstos e os reais, podendo assim ter-se noção da performance do modelo em causa, como na referência [63], onde se analisa a similaridade entre os valores de consumos previstos e os reais num edifício comercial em Chicago. Para além disso podem ter outras aplicações como na referência [9], onde se utilizaram gráficos de dispersão para visualizar a distribuição probabilística do erro, ou na referência [64], onde os mesmos foram usados para comparar as previsões obtidas com diferentes modelos.

Os gráficos de dispersão criados para os modelos de previsão *Random Forests*, MARS, ANN e LSVM são apresentados da Figura 6-14 à Figura 6-17.

A análise dos mesmos permite concluir que os modelo Random Forests e ANN são os que aparentam ter melhor performance, já que a distribuição dos dados é a que se aproxima mais da linha de tendência, bem como pela métrica R^2 apresentada nos gráficos.

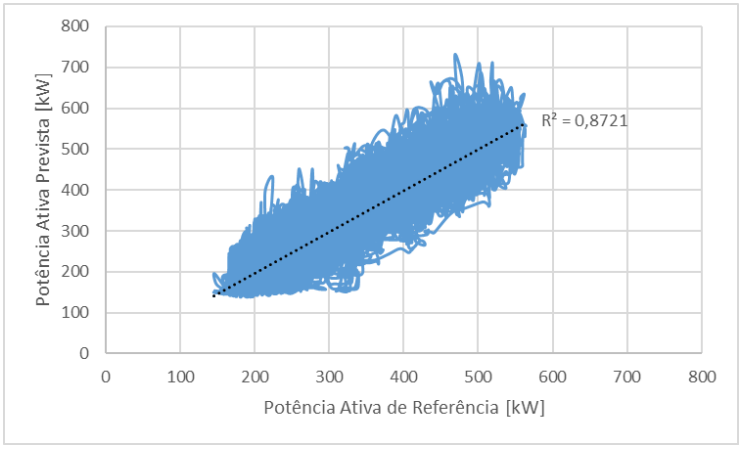


Figura 6-14- Gráfico de dispersão real vs previsto (modelo Random Forests)

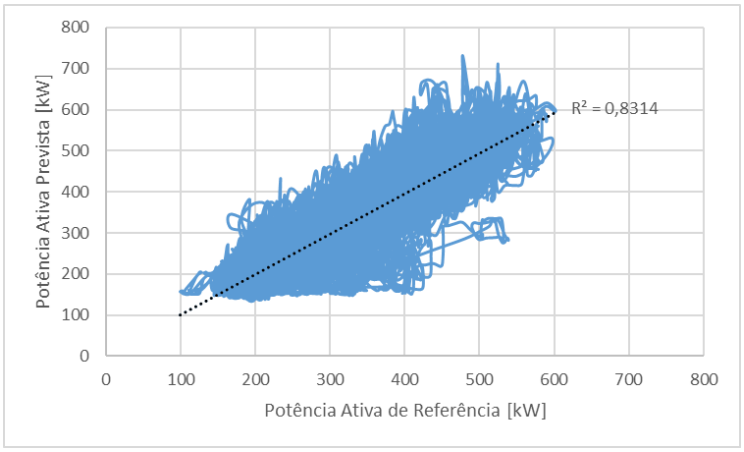


Figura 6-15-Gráfico de dispersão real vs previsto (modelo MARS)

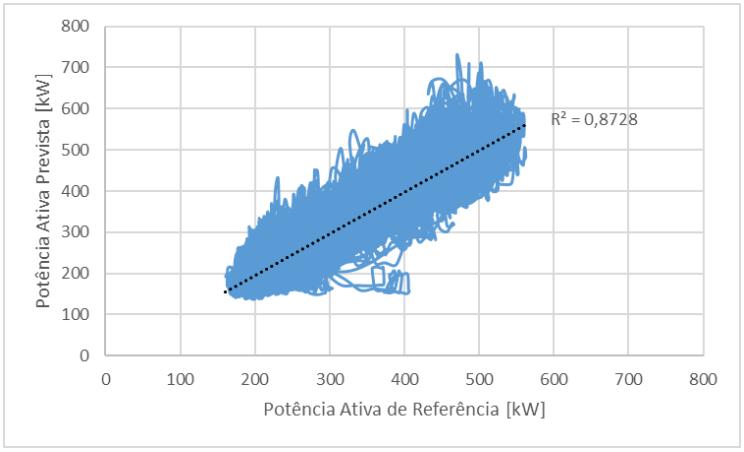


Figura 6-16-Gráfico de dispersão real vs previsto (modelo ANN)

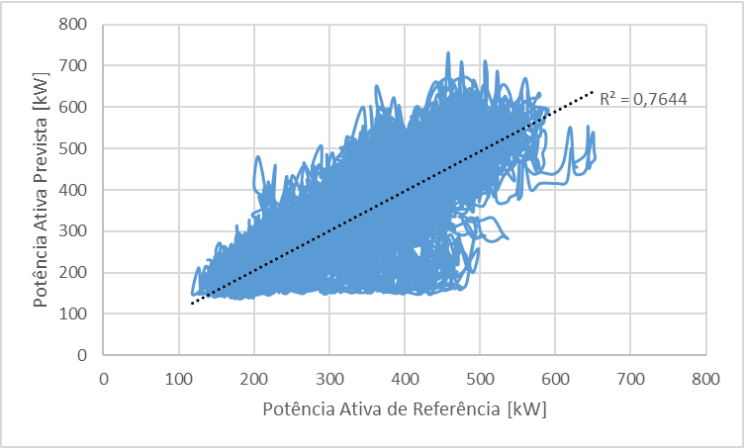


Figura 6-17-Gráfico de dispersão real vs previsto (modelo LSVM)

6.5.5. Gráfico boxplot

Outra ferramenta gráfica que permite avaliar a dispersão dos dados são os gráficos *boxplot*. Na bibliografia analisada estes gráficos foram aplicados para diferentes fins. Por exemplo, a título de exemplo, na referência [48]foram usados para comparar os consumos reais com os previstos; na referência [40] para detetar *outlier*s e na referência [65] para análise da dispersão dos erros de um modelo proposto. Na presente dissertação os *boxplots* desenvolvidos dizem também respeito à dispersão dos erros. Foram criados três gráficos deste tipo, no primeiro a divisão dos erros foi feita por períodos horários de 4 horas, no segundo uma divisão mensal e no terceiro uma divisão com base no valor tomado pela variável "Tipo de Dia". Os dois últimos são apresentados no Anexo J enquanto que o primeiro é mostrado na Figura 6-18.

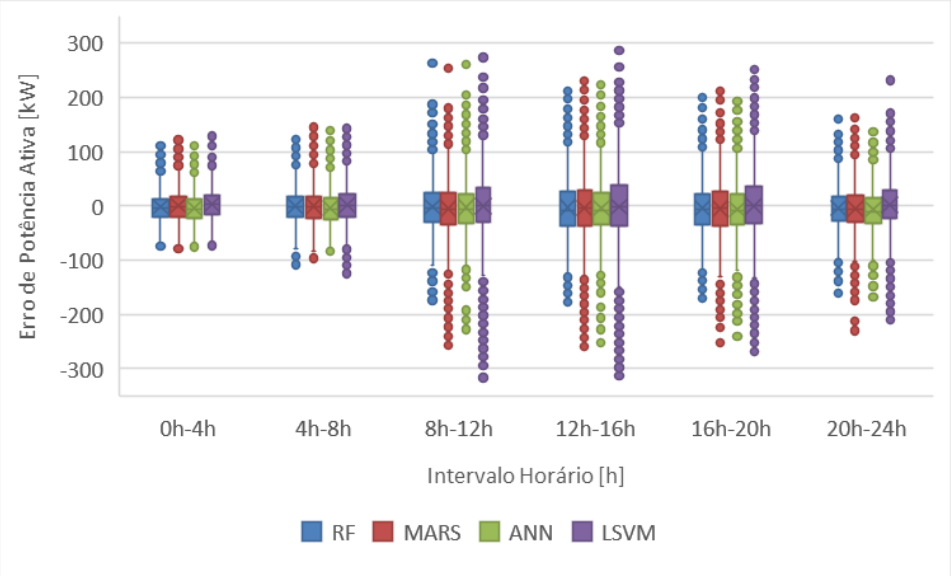


Figura 6-18-Gráfico boxplot por intervalos horários de 4h

Uma conclusão importante que se pode retirar dos três gráficos é a de que as previsões efetuadas com o modelo LSVM aparentam ser as piores, já que os seus *boxplots* são na maioria dos casos mais compridos que os dos restantes modelos, seguindo-se geralmente os *boxplots* do modelo MARS. No que respeita ao gráfico da Figura 6-18 pode ver-se que a dispersão dos erros é maior entre as 8 e as 20 horas, seguindo-se o período entre as 20 e as 24 horas, o que se explica por estes serem geralmente os períodos de maior consumo de potência ativa no Campus. Em relação ao *boxplot* mensal, verificam-se tendências diferentes em diferentes meses. Por exemplo em março e abril as previsões são tendencialmente mais elevadas que a referência, uma vez que a mediana e grande parte do intervalo interquartis estão abaixo de 0. Por outro lado, em Setembro verifica-se o contrário, pelo que os erros neste período estarão mais vezes relacionados com previsões demasiado baixas. Por fim, no gráfico onde a divisão é feita tendo em conta a variável "Tipo de Dia", o que se destaca é a maior dispersão dos erros nos períodos das férias e principalmente de Natal e da Páscoa, o que reforça o que foi visto aquando da análise das métricas de erro onde se concluiu que nestes períodos o desempenho dos modelos é mais baixo.

6.6.Análise de resultados de previsão durante pandemia de Covid-19

A fim de analisar a performance dos modelos durante a pandemia de Covid-19 aplicou-se o modelo *Random Forests* aos dados de potência ativa entre 1 de março de 2020 e 28 de fevereiro de 2021, que foram excluídos do teste final. Optou-se pelo modelo *Random Forests* por este modelo por ter sido o que apresentou melhores métricas de erro quando considerados todos os períodos da variável "Tipo de Dia". Na Tabela 6-24 são apresentadas as métricas de erro para o intervalo descrito anteriormente e no Anexo K o diagrama de carga Real vs Previsto para as primeiras quatro semanas imediatamente após o fim do período de teste.

Tabela 6-24-Métricas de erro obtidas durante período de pandemia

MAE Teste [kW]	MSE Teste [kW²]	RMSE Teste [kW]	MAPE Teste [%]	Tempo Previsão Teste [s]
40,08	2883,9	53,70	22,95	0,30

Como se pode ver as métricas deterioraram-se consideravelmente face às obtidas durante o período de teste, o que vem comprovar que se decidiu corretamente ao excluir estes dados do período de teste. É então evidente que, devido à substancial diferença nos perfis de consumo entre o período pandemia e pré-pandemia os modelos não estão preparados para efetuar previsões de qualidade neste período pois os dados utilizados para treinar esses mesmos modelos não incluíram essa situação atípica. Analisando o gráfico do Anexo K pode ver-se que, a partir do momento em que os efeitos da pandemia se manifestam nos valores de potência, o que acontece no dia 16 de março de 2020, o modelo demora cerca de 3 semanas a conseguir acompanhar os valores de referência, apesar de a partir daí não o conseguir fazer com rigor como as métricas evidenciaram. Esta demora está naturalmente relacionada com as 13 *features* referentes a registos anteriores utilizadas, que recorde-se vão desde um dia até oito semanas de atraso.

7. Conclusões e trabalho futuro

7.1. Conclusões

A presente dissertação consistiu na criação de quatro modelos de previsão de curto prazo da potência ativa do Campus 2 do Politécnico de Leiria, mais concretamente até 24 horas, uma vez que uma das *features* utilizadas foram registos de potência ativa do dia anterior.

Começou por ser feita uma descrição e tratamento criteriosos dos dados disponíveis, envolvendo preenchimento de valores em falta, correção de *outliers* e correção de mudanças de hora, procurando justificar as diferentes decisões tomadas com base em bibliografia na área. O mesmo foi feito de seguida na secção da seleção e extração das *features*, onde se aplicaram diferentes técnicas para avaliar a relevância de cada variável e a técnica de PCA para combinar *features* e combater o problema da dimensionalidade. O detalhe dado às diferentes fases inerentes à criação de um modelo de *machine learning* e que antecedem a utilização dos modelos propriamente ditos constitui um dos pontos fortes deste trabalho quando comparado com a maioria da bibliografia estudada. No final foram criados quatro modelos recorrendo às técnicas *Random Forests*, ANN, SVM e MARS, os quais foram otimizados com as técnicas de *Grid Search* e *Random Search*. Estes modelos foram depois comparados com recurso a métricas de erro e diferentes ferramentas de visualização, mais concretamente diagramas de carga, mapas térmicos, gráficos de dispersão e boxplots. Essa análise revelou que os modelos *Random Forests e ANN* destacaram-se dos restantes e apresentaram bom desempenho na maioria dos períodos, principalmente nos dias de aulas e épocas de exames, que representam a maior parte do funcionamento do Campus. Por outro lado, a qualidade das previsões nos domingos e feriados, mas especialmente nas férias de Natal e da Páscoa foram piores. Testou-se ainda o modelo *Random Forests* durante a pandemia Covid-19, cujos resultados evidenciaram os problemas de aplicar modelos de *machine learning* em situações totalmente distintas daquelas em que foram treinados. Considera-se então que se atingiram os objetivos propostos inicialmente, uma vez que foram atingidas previsões confiáveis na maioria das situações.

7.2.Trabalho futuro

Como trabalho futuro poderia procurar-se afinar o comportamento dos modelos nos intervalos onde estes apresentaram pior desempenho, por exemplo através de modelos mais complexos, nomeadamente modelos *ensemble* que utilizem técnicas distintas em diversos períodos, já que a Tabela 6-23 evidencia que diferentes modelos se destacaram em diferentes valores da variável "Tipo de Dia", o que poderia ser aproveitado.

Na secção de *features* disponíveis concluiu-se, entre outras coisas, que as variáveis meteorológicas não eram relevantes. Algo que poderia ajudar a melhorar a qualidade dos modelos seria precisamente a obtenção de variáveis exógenas com uma resolução mais pequena e mais próxima à dos dados de potência ativa, os quais representam intervalos de 15 minutos, enquanto os dados meteorológicos utilizados possuem uma resolução diária, o que poderá explicar o porquê de não terem contribuído para as previsões, ao contrário do que se verificou em muita da bibliografia analisada, que recorreu a este tipo de *features*. Algo que poderia também ajudar a resolver esse problema seria a existência de variáveis desse tipo mas resultantes de medições feitas diretamente no Campus e não apenas nas proximidades.

A obtenção de dados de ocupação do Campus, por exemplo o número de ocupantes em cada edifício ou o nível de ocupação dos parques de estacionamento, poderia também ser uma mais-valia e constituir mais um entrada relevante para os modelos de previsão. Tal poderia ser feito, por exemplo, com recurso a contadores na entrada dos edifícios ou parques de estacionamento, os quais se tornaram comuns nos espaços comerciais durante a pandemia e recorrem, por exemplo, a técnicas de visão computacional, também elas lecionadas no mestrado no qual esta dissertação se enquadra.

Por fim, poderiam aplicar-se em tempo real os modelos desenvolvidos e ser criada uma plataforma que permitisse consultar as previsões efetuadas. Essa plataforma poderia ser fundamental no apoio à gestão dos edifícios, podendo, entre outras coisas, emitir alertas no caso da existência de consumos anómalos que divirjam significativamente das previsões, os quais permitiriam uma ação célere sobre eventuais problemas.

Referências bibliográficas

[1] BP, “Statistical Review of World Energy globally consistent data on world energy markets . and authoritative publications in the field of energy,” *BP Energy Outlook 2021*, vol. 70, pp. 8–20, 2021.

[2] “Data Hub da Ren (Dia 30/10/2021).” https://datahub.ren.pt/pt/eletricidade/balanco-diario/?date=2021-10-30.

[3] BP p.l.c., “BP Energy Outlook 2019 edition The Energy Outlook explores the forces shaping the global energy transition out to 2040 and the key uncertainties surrounding that,” *BP Energy Outlook 2019*, 2019.

[4] Observatório da Energia, D. de S. de P. E. e E. DGEG – Direção Geral de Energia e Geologia, and U. de I. ADENE – Agência para a Energia, “Energia em Números (Edição 2021),” no. EDIÇÃO, 2021.

[5] República Portuguesa, “Roteiro para a Neutralidade Carbónica 2050,” *Estratégia longo prazo para a neutralidade carbónica da Econ. Port. em 2050*, vol. 2050, pp. 9–24, 2019.

[6] “Google Earth (Campus 2 do IPL).” https://earth.google.com/web/@39.73401783,-8.82136646,59.09666471a,359.15263967d,35y,73.67258129h,54.70635673t,0.0000017r.

[7] A. Letivo, “Guia de Integração na ESTG,” 2012.

[8] J. Brownlee, “How to Make Baseline Predictions for Time Series Forecasting with Python,” 2016. https://machinelearningmastery.com/persistence-time-series-forecasting-with-python/.

[9] A. Bagnasco, F. Fresi, M. Saviozzi, F. Silvestro, and A. Vinci, “Electrical consumption forecasting in hospital facilities : An application case,” vol. 103, pp. 261–270, 2015.

[10] S. Vieira, W. H. L. Pinaya, and A. Mechelli, “Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and

applications," *Neurosci. Biobehav. Rev.*, vol. 74, no. January, pp. 58–75, 2017, doi: 10.1016/j.neubiorev.2017.01.002.

[11] Scikit learn Developers, "sklearn.neural_network.MLPRegressor." https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html#sklearn.neural_network.MLPRegressor.

[12] C. CORTES and V. VAPNIK, "Support-Vector Networks," 1995, doi: 10.1109/64.163674.

[13] R. Taghizadeh-Mehrjardi, R. Neupane, K. Sood, and S. Kumar, "Artificial bee colony feature selection algorithm combined with machine learning algorithms to predict vertical and lateral distribution of soil organic matter in South Dakota, USA," *Carbon Manag.*, vol. 8, no. 3, pp. 277–291, 2017, doi: 10.1080/17583004.2017.1330593.

[14] Scikit learn Developers, "sklearn.svm.SVR." https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html.

[15] Scikit learn Developers, "sklearn.svm.LinearSVR." https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html.

[16] X. Yang, Y. Wang, R. Byrne, G. Schneider, and S. Yang, "Concepts of Artificial Intelligence for Computer-Assisted Drug Discovery," *Chem. Rev.*, vol. 119, no. 18, pp. 10520–10594, 2019, doi: 10.1021/acs.chemrev.8b00728.

[17] L. Rokach and O. Maimon, "Decision Trees," in *DATA MINING AND KNOWLEDGE DISCOVERY HANDBOOK*, Springer, Boston, MA, 2005.

[18] M. Dmitrievsky, "RANDOM DECISION FOREST IN REINFORCEMENT LEARNING," 2018. https://www.mql5.com/en/articles/3856.

[19] D. Trehan, "Why Choose Random Forest and Not Decision Trees," 2020. https://towardsai.net/p/machine-learning/why-choose-random-forest-and-not-decision-trees.

[20] Scikit learn Developers, "sklearn.ensemble.RandomForestRegressor," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html.

[21] J. Friedman, “MULTIVARIATE ADAPTIVE REGRESSION SPLINES,” *Ann. Stat.*, 1990.

[22] “MULTIVARIATE ADAPTIVE REGRESSION SPLINES,” *Biodiversity and Climate Change Virtual Laboratory*, 2019. https://support.bccvl.org.au/support/solutions/articles/6000118097-multivariate-adaptive-regression-splines.

[23] J. Rudy, “pyearth.Earth.” https://contrib.scikit-learn.org/py-earth/content.html#multivariate-adaptive-regression-splines.

[24] Z. Jaadi, “A Step-by-Step Explanation of Principal Component Analysis (PCA).” https://builtin.com/data-science/step-step-explanation-principal-component-analysis.

[25] Scikit learn Developers, “sklearn.decomposition.PCA.” https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html.

[26] E. Taskesen, “pca,” 2019. https://github.com/erdogant/pca.

[27] J. Bemister-Buffington, A. J. Wolf, S. Raschka, and L. A. Kuhn, “Machine learning to identify flexibility signatures of class a GPCR inhibition,” *Biomolecules*, vol. 10, no. 3, pp. 1–22, 2020, doi: 10.3390/biom10030454.

[28] S. Raschka, “Sequential Feature Selector from mlxtend library.” http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/.

[29] S. ES, “Hyperparameter Tuning in Python: a Complete Guide,” 2022. https://nanonets.com/blog/hyperparameter-optimization/.

[30] Scikit learn Developers, “sklearn.model_selection.GridSearchCV,” [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[31] Scikit learn Developers, “sklearn..model_selection.RandomizedSearchCV.” https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html.

[32] “Weather Underground.” https://www.wunderground.com/.

[33] “Eurostat.” https://ec.europa.eu/eurostat/databrowser/view/nrg_chdd_a/default/table?lang=en.

[34] J. N. Fidalgo and M. A. Matos, “Forecasting Portugal Global Load with Artificial Neural Networks,” *Int. Conf. Artif. Neural Networks – ICANN2007*, 2007.

[35] “Site AGCP do IPL.” https://publico.agcp.ipleiria.pt/paginas/default.aspx.

[36] N. Li, T. Zong, and Z. Zhang, “Prediction of the Electronic Work Function by Regression Algorithm in Machine Learning,” *2021 IEEE 6th Int. Conf. Big Data Anal. ICBDA 2021*, pp. 87–91, 2021, doi: 10.1109/ICBDA51983.2021.9403202.

[37] J. Hinman and E. Hickey, “Modeling and Forecasting Short-Term Electricity Load Using Regression Analysis,” *Illinois State Univ.*, pp. 1–51, 2009, [Online]. Available: http://www.irps.ilstu.edu/research/documents/LoadForecastingHinman-HickeyFall2009.pdf.

[38] G. Wilson, S. Sharma, J. Day, and N. Godfrey, “A method to calculate Great Britain’s half-hourly electrical demand from publicly available data.,” pp. 1–8, 2020, [Online]. Available: https://www.researchgate.net/profile/Juan_Banda/publication/340523391_A_large-scale_COVID-19_Twitter_chatter_dataset_for_open_scientific_research_--_an_international_collaboration/links/5e973c7192851c2f52a61ebf/A-large-scale-COVID-19-Twitter-chatter-datas.

[39] N. Kim, M. Kim, and J. K. Choi, “LSTM Based Short-term Electricity Consumption Forecast with Daily Load Profile Sequences,” *2018 IEEE 7th Glob. Conf. Consum. Electron. GCCE 2018*, pp. 834–835, 2018, doi: 10.1109/GCCE.2018.8574484.

[40] S. Shan, “Forecasting the Short-Term Electricity Consumption of Building Using a Novel Ensemble Model,” *IEEE Access*, vol. 7, pp. 88093–88106, 2019, doi: 10.1109/ACCESS.2019.2925740.

[41] S. Chemetova, “Previsão de consumo de energia eléctrica nos principais pontos injectores da rede de transporte na rede de distribuição,” 2018.

[42] J. Moon, J. Park, E. Hwang, and S. Jun, “Forecasting power consumption for higher

educational institutions based on machine learning," *J. Supercomput.*, vol. 74, no. 8, pp. 3778–3800, 2018, doi: 10.1007/s11227-017-2022-x.

[43] A. Rahman, V. Srikumar, and A. D. Smith, "Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks," *Appl. Energy*, vol. 212, no. October 2017, pp. 372–385, 2018, doi: 10.1016/j.apenergy.2017.12.051.

[44] S. Seo, "A review and comparison of methods for detecting outliersin univariate data sets," *Dep. Biostat. Grad. Sch. Public Heal.*, pp. 1–53, 2006, [Online]. Available: http://d-scholarship.pitt.edu/7948/.

[45] J. W. Tukey, *Exploratory Data Analysis*. Addison-Wesley Publishing Company, 1977.

[46] H. S. Wong and A. Fitrianto, "Adjusted sequential fences for detecting univariate outliers in skewed distributions," *ASM Sci. J.*, vol. 12, no. SpecialIssue5, pp. 107–115, 2019.

[47] K. Carling, "Resistant outlier rules and the non-Gaussian case," *Comput. Stat. Data Anal.*, vol. 33, no. 3, pp. 249–258, 2000, doi: 10.1016/S0167-9473(99)00057-2.

[48] K. P. Amber *et al.*, "Energy Consumption Forecasting for University Sector Buildings," pp. 1–18, 2017, doi: 10.3390/en10101579.

[49] Z. Tan *et al.*, "Combined electricity-heat-cooling-gas load forecasting model for integrated energy system based on multi-task learning and least square support vector machine," *J. Clean. Prod.*, vol. 248, p. 119252, 2020, doi: 10.1016/j.jclepro.2019.119252.

[50] M. Sarhani, "Electric load forecasting using hybrid machine learning approach incorporating feature selection," no. 1992, pp. 1–7, 2015.

[51] A. T. Eseye and M. Lehtonen, "Short - term Forecasting of Heat Demand of Buildings for Efficient and Optimal Energy Management Based on Integrated Machine Learning Models," 2020, doi: 10.1109/TII.2020.2970165.

[52] A. A. Naser M.Z., "Insights into Performance Fitness and Error Metrics for Machine Learning," p. 25, 2020.

[53] P. Amin, L. Cherkasova, R. Aitken, and V. Kache, “Automating energy demand modeling and forecasting using smart meter data,” *Proc. - 2019 IEEE Int. Congr. Internet Things, ICIOT 2019 - Part 2019 IEEE World Congr. Serv.*, pp. 133–137, 2019, doi: 10.1109/ICIOT.2019.00032.

[54] “Biblioteca Heatmap,” 2020. https://pypi.org/project/heatmapz/.

[55] P. Probst and A. L. Boulesteix, “To tune or not to tune the number of trees in random forest,” *J. Mach. Learn. Res.*, vol. 18, no. 2001, pp. 1–8, 2018.

[56] P. Probst, M. N. Wright, and A. L. Boulesteix, “Hyperparameters and tuning strategies for random forest,” *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 9, no. 3, pp. 1–19, 2019, doi: 10.1002/widm.1301.

[57] S. Saxena, “A Beginner’s Guide to Random Forest Hyperparameter Tuning,” 2020. https://www.analyticsvidhya.com/blog/2020/03/beginners-guide-random-forest-hyperparameter-tuning/#h2_7.

[58] T. Sterbak, “Getting started with Multivariate Adaptive Regression Splines,” 2013. https://www.depends-on-the-definition.com/getting-started-with-multivariate-adaptive-regression-spline/.

[59] L. Villalobos-Arias and C. Quesada-López, *Comparative study of random search hyper-parameter tuning for software effort estimation*, vol. 1, no. 1. Association for Computing Machinery, 2021.

[60] J. Heaton, “Heaton Research The Number of Hidden Layers,” 2017. https://www.heatonresearch.com/2017/06/01/hidden-layers.html.

[61] S. Karsoliya, “Approximating Number of Hidden layer neurons in Multiple Hidden Layer BPNN Architecture,” *Int. J. Eng. Trends Technol.*, vol. 3, no. 6, pp. 714–717, 2012.

[62] (KW Engineering), “How a Heat Map Can Lower Your Energy Bill.” https://www.kw-engineering.com/energy-savings-calendar-heat-map/.

[63] A. Haque *et al.*, “An SVR-based Building-level Load Forecasting Method Considering Impact of HVAC Set Points,” *2019 IEEE Power Energy Soc. Innov. Smart Grid Technol. Conf. ISGT 2019*, pp. 9–13, 2019, doi:

10.1109/ISGT.2019.8791649.

[64] C. Robinson *et al.*, “Machine learning approaches for estimating commercial building energy consumption,” *Appl. Energy*, vol. 208, no. September, pp. 889–904, 2017, doi: 10.1016/j.apenergy.2017.09.060.

[65] X. Godinho, H. Bernardo, F. T. Oliveira, and J. C. Sousa, “Forecasting Heating and Cooling Energy Demand in an Office Building using Machine Learning Methods,” pp. 1–6, 2020, doi: 10.1109/yef-ece49388.2020.9171807.

[66] M. Bourdeau, E. Nefzaoui, X. Guo, and P. Chatellier, “Modeling and forecasting building energy consumption : A review of data- driven techniques,” *Sustain. Cities Soc.*, vol. 48, no. April, p. 101533, 2019, doi: 10.1016/j.scs.2019.101533.

[67] T. G. (Oregon S. U. Dietterich, “Ensemble Learning,” 2002, [Online]. Available: https://home.cs.colorado.edu/~mozer/Teaching/syllabi/6622/papers/Dietterich2002.pdf.

[68] R. Richman and M. V. Wüthrich, “Bagging predictors,” *Risks*, vol. 8, no. 3, pp. 1–26, 2020, doi: 10.3390/risks8030083.

[69] Y. Freund and R. E. Schapire, “Experiments with a New Boosting Algorithm,” *Proc. 13th Int. Conf. Mach. Learn.*, pp. 148–156, 1996, doi: 10.1.1.133.1040.

[70] G. K. F. Tso and K. K. W. Yau, “Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks,” *Energy*, vol. 32, no. 9, pp. 1761–1768, 2007, doi: 10.1016/j.energy.2006.11.010.

[71] S. Dobilas, “MARS: Multivariate Adaptive Regression Splines — How to Improve on Linear Regression? | by Saul Dobilas | Towards Data Science,” 2020. https://towardsdatascience.com/mars-multivariate-adaptive-regression-splines-how-to-improve-on-linear-regression-e1e7a63c5eae.

[72] I. T. Jollife and J. Cadima, “Principal component analysis: A review and recent developments,” *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 374, no. 2065, 2016, doi: 10.1098/rsta.2015.0202.

[73] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Comput. Electr. Eng.*, vol. 40, no. 1, pp. 16–28, 2014, doi:

10.1016/j.compeleceng.2013.11.024.

[74] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.

[75] C. GOYAL, “Data Leakage And Its Effect On The Performance of An ML Model,” 2021. https://www.analyticsvidhya.com/blog/2021/07/data-leakage-and-its-effect-on-the-performance-of-an-ml-model/.

[76] S. Yadav and S. Shukla, “Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification,” *Proc. - 6th Int. Adv. Comput. Conf. IACC 2016*, no. Cv, pp. 78–83, 2016, doi: 10.1109/IACC.2016.25.

[77] R. Kohavi and S. Edu, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” *Proc. 14th Int. Jt. Conf. Artif. Intell.*, vol. 2, pp. 1137–1143, 1993.

[78] T. Phan, “An introduction to Principal Component Analysis with examples in R,” pp. 1–14, 2016.

[79] A. Tripathi, “A Complete Guide to Principal Component Analysis — PCA in Machine Learning,” 2019. https://towardsdatascience.com/a-complete-guide-to-principal-component-analysis-pca-in-machine-learning-664f34fc3e5a.

[80] M. Brems, “A One-Stop Shop for Principal Component Analysis,” 2017. https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c.

[81] A. Zheng, “How to Evaluate Machine Learning Models: Hyperparameter Tuning,” 2015. https://web.archive.org/web/20160701182750/http://blog.dato.com/how-to-evaluate-machine-learning-models-part-4-hyperparameter-tuning.

[82] M. Kuhn and K. Johnson, *Applied Predictive Modeling with Applications in R*, vol. 26. 2013.

[83] B. Boehmke and B. Greenwell, “Chapter 7 Multivariate Adaptive Regression Splines,” in *Hands-On Machine Learning with R*, 2020.

[84] A. Blum, *Neural networks in C++ : an object-oriented framework for building*

connectionist systems. NY:Willey, 1992.

[85] G. S. Linoff and M. J. A. Berry, *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management, 3rd Edition*. NY:Willey.

[86] Scikit learn Developers, “Neural network models (supervised),” 2022. https://scikit-learn.org/stable/modules/neural_networks_supervised.html.

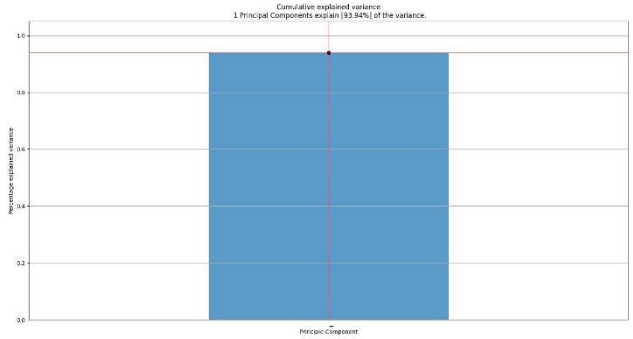
[87] Scikit learn Developers, “Varying regularization in Multi-layer Perceptron,” 2022. https://scikit-learn.org/stable/auto_examples/neural_networks/plot_mlp_alpha.html#sphx-glr-auto-examples-neural-networks-plot-mlp-alpha-py.

[88] Y. Bengio, “Practical recommendations for gradient-based training of deep architectures,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7700 LECTU, pp. 437–478, 2012, doi: 10.1007/978-3-642-35289-8_26.

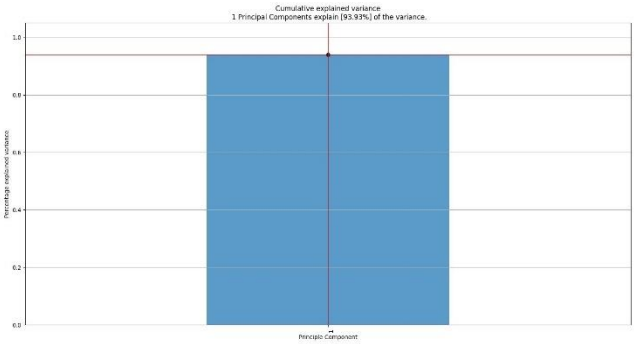
Anexo A- *Features* referentes a registos anteriores

Variável	Lag/Atraso	Variável	Lag/Atraso
-1 Dia	1 Dias	-2 semanas2	2 Semanas - 30 minutos
-1 Dia1	1 Dias - 15 minutos	-2 semanas3	2 Semanas + 15 minutos
-1 Dia2	1 Dias - 30 minutos	-2 semanas4	2 Semanas + 30 minutos
-1 Dia3	1 Dias + 15 minutos	-3 semanas	3 Semanas
-1 Dia4	1 Dias + 30 minutos	-3 semanas1	3 Semanas - 15 minutos
-2 Dias	2 Dias	-3 semanas2	3 Semanas - 30 minutos
-2 Dias1	2 Dias - 15 minutos	-3 semanas3	3 Semanas + 15 minutos
-2 Dias2	2 Dias - 30 minutos	-3 semanas4	3 Semanas + 30 minutos
-2 Dias3	2 Dias + 15 minutos	-4 semanas	4 Semanas
-2 Dias4	2 Dias + 30 minutos	-4 semanas1	4 Semanas - 15 minutos
-3 Dias	3 Dias	-4 semanas2	4 Semanas - 30 minutos
-3 Dias1	3 Dias - 15 minutos	-4 semanas3	4 Semanas + 15 minutos
-3 Dias2	3 Dias - 30 minutos	-4 semanas4	4 Semanas + 30 minutos
-3 Dias3	3 Dias + 15 minutos	-5 semanas	5 Semanas
-3 Dias4	3 Dias + 30 minutos	-5 semanas1	5 Semanas - 15 minutos
-6 Dias	6 Dias	-5 semanas2	5 Semanas - 30 minutos
-6 Dias1	6 Dias - 15 minutos	-5 semanas3	5 Semanas + 15 minutos
-6 Dias2	6 Dias - 30 minutos	-5 semanas4	5 Semanas + 30 minutos
-6 Dias3	6 Dias + 15 minutos	-6 semanas	6 Semanas
-6 Dias4	6 Dias + 30 minutos	-6 semanas1	6 Semanas - 15 minutos
-8 Dias	8 Dias	-6 semanas2	6 Semanas - 30 minutos
-8 Dias1	8 Dias - 15 minutos	-6 semanas3	6 Semanas + 15 minutos
-8 Dias2	8 Dias - 30 minutos	-6 semanas4	6 Semanas + 30 minutos
-8 Dias3	8 Dias + 15 minutos	-7 semana	7 Semanas
-8 Dias4	8 Dias + 30 minutos	-7 semanas1	7 Semanas - 15 minutos
-1 semana	1 Semana	-7 semanas2	7 Semanas - 30 minutos
-1 semana1	1 Semana - 15 minutos	-7 semanas3	7 Semanas + 15 minutos
-1 semana2	1 Semana - 30 minutos	-7 semanas4	7 Semanas + 30 minutos
-1 semana3	1 Semana + 15 minutos	-8 semanas	8 Semanas
-1 semana4	1 Semana + 30 minutos	-8 semanas3	8 Semanas + 15 minutos
-2 semanas	2 Semanas	-8 semanas4	8 Semanas + 30 minutos
-2 semanas1	2 Semanas - 15 minutos		

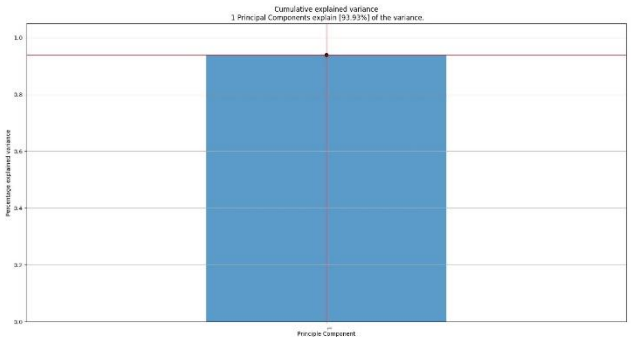
Anexo B- Gráficos de *cumulative explained variance* para a extração das *features* referentes a registos anteriores



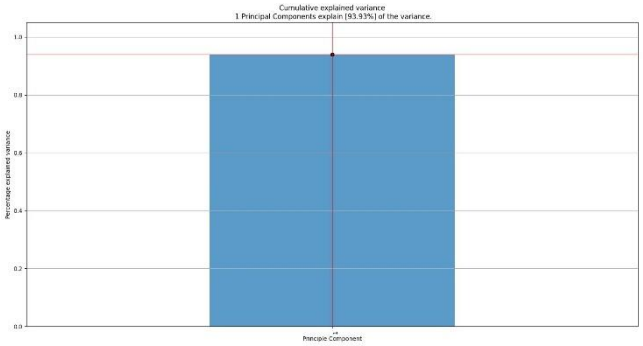
1 Dia



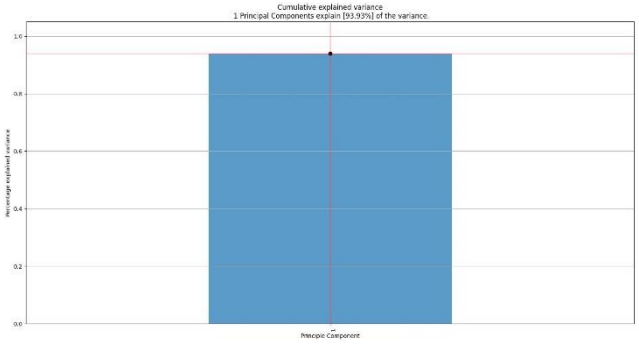
2 Dias



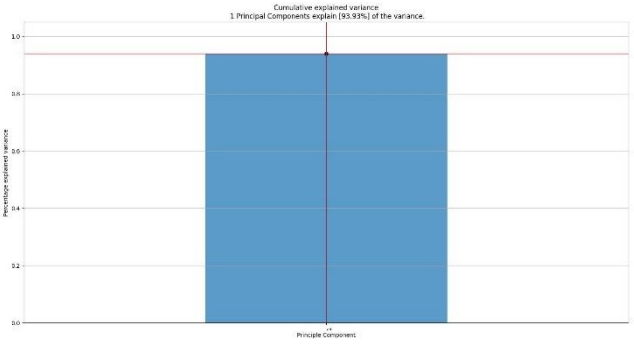
3 Dias



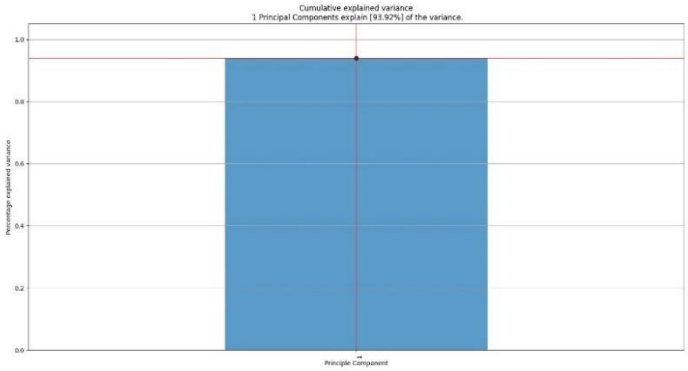
6 Dias



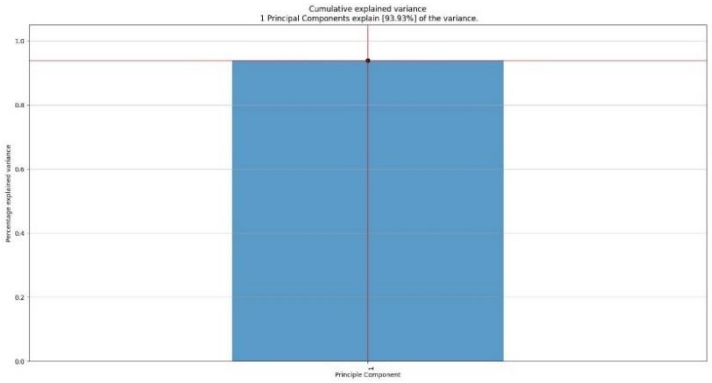
8 Dias



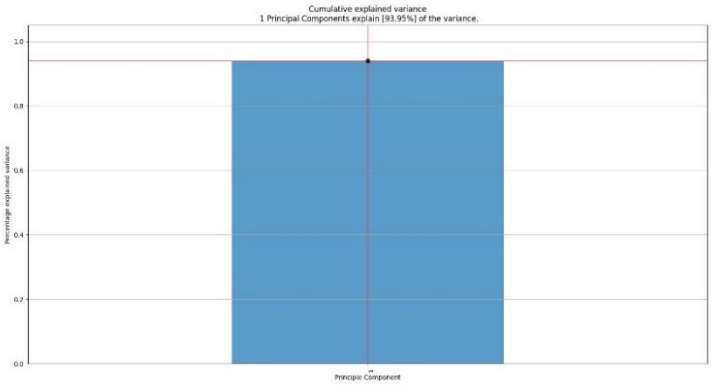
1 Semana



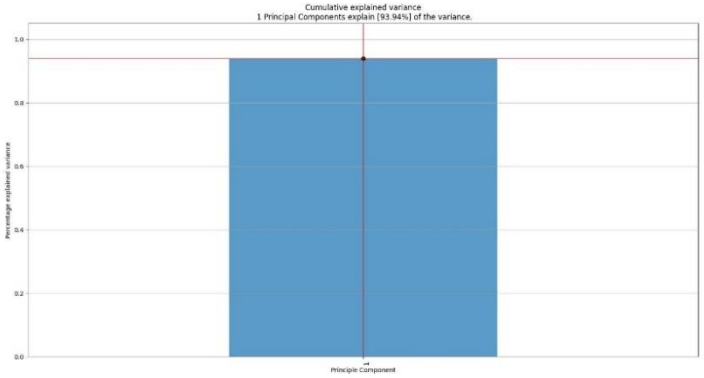
2 Semanas



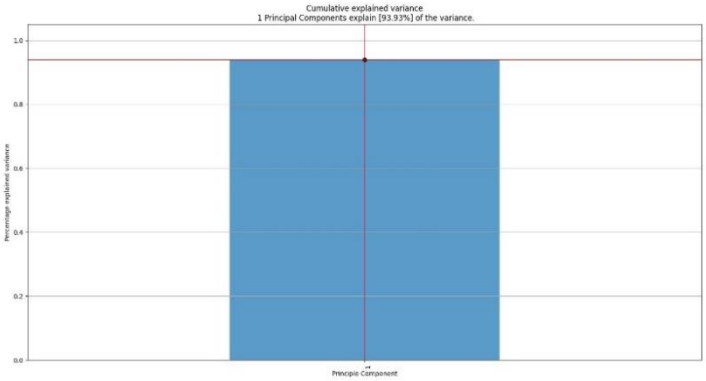
3 Semanas



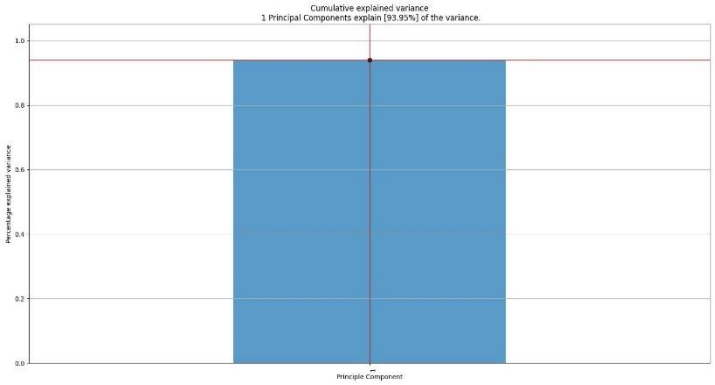
4 Semanas



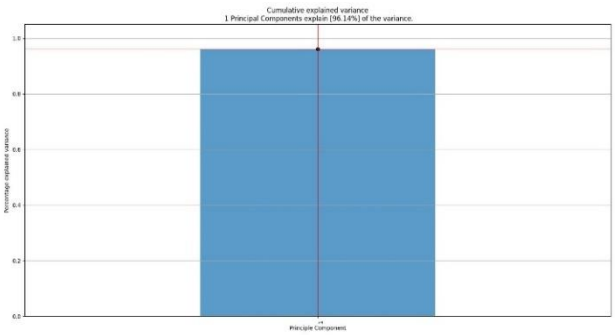
5 Semanas



6 Semanas

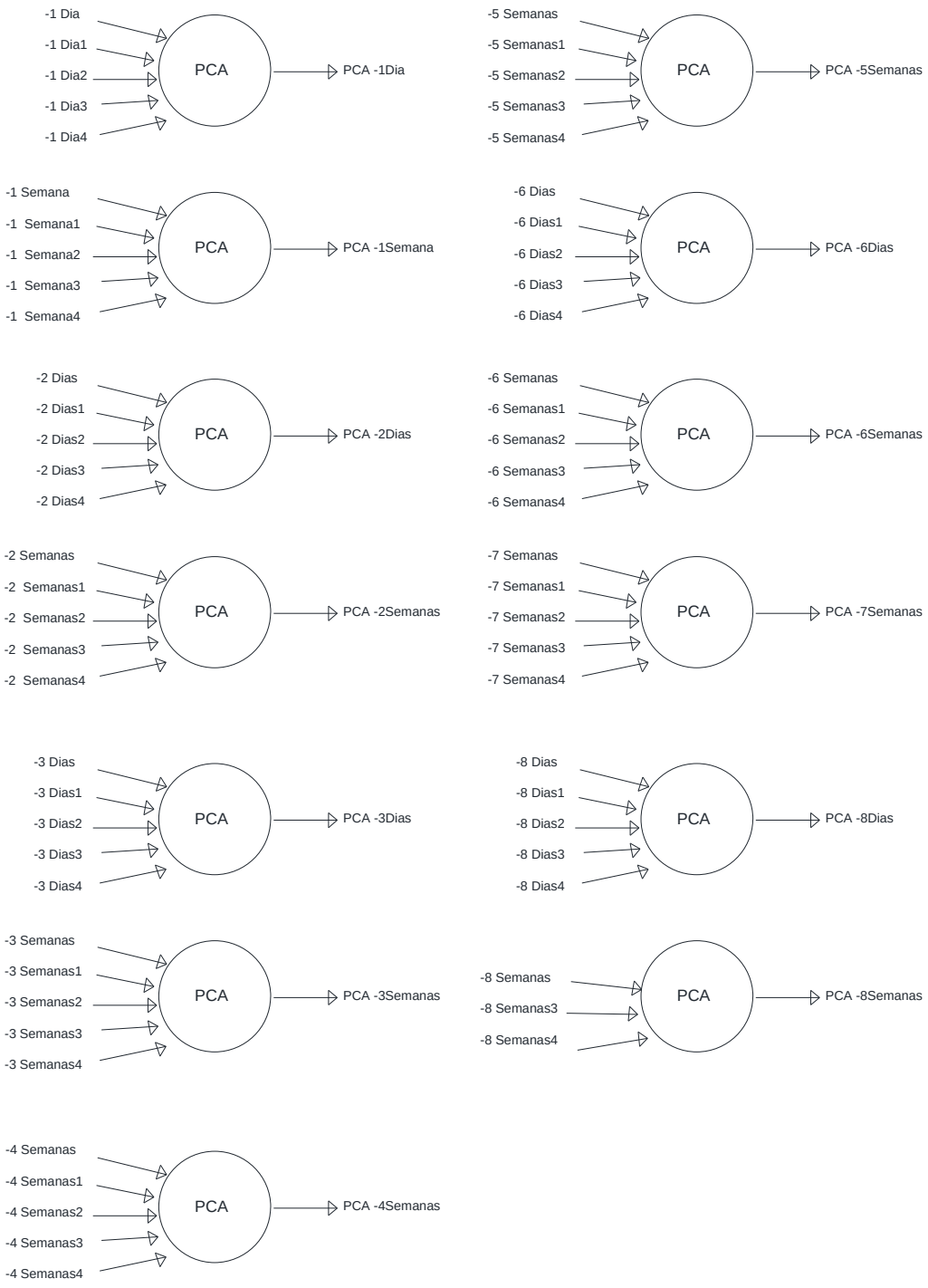


7 Semanas



8 Semanas

Anexo C- Processo de *feature extraction* das *features* referentes a registos anteriores



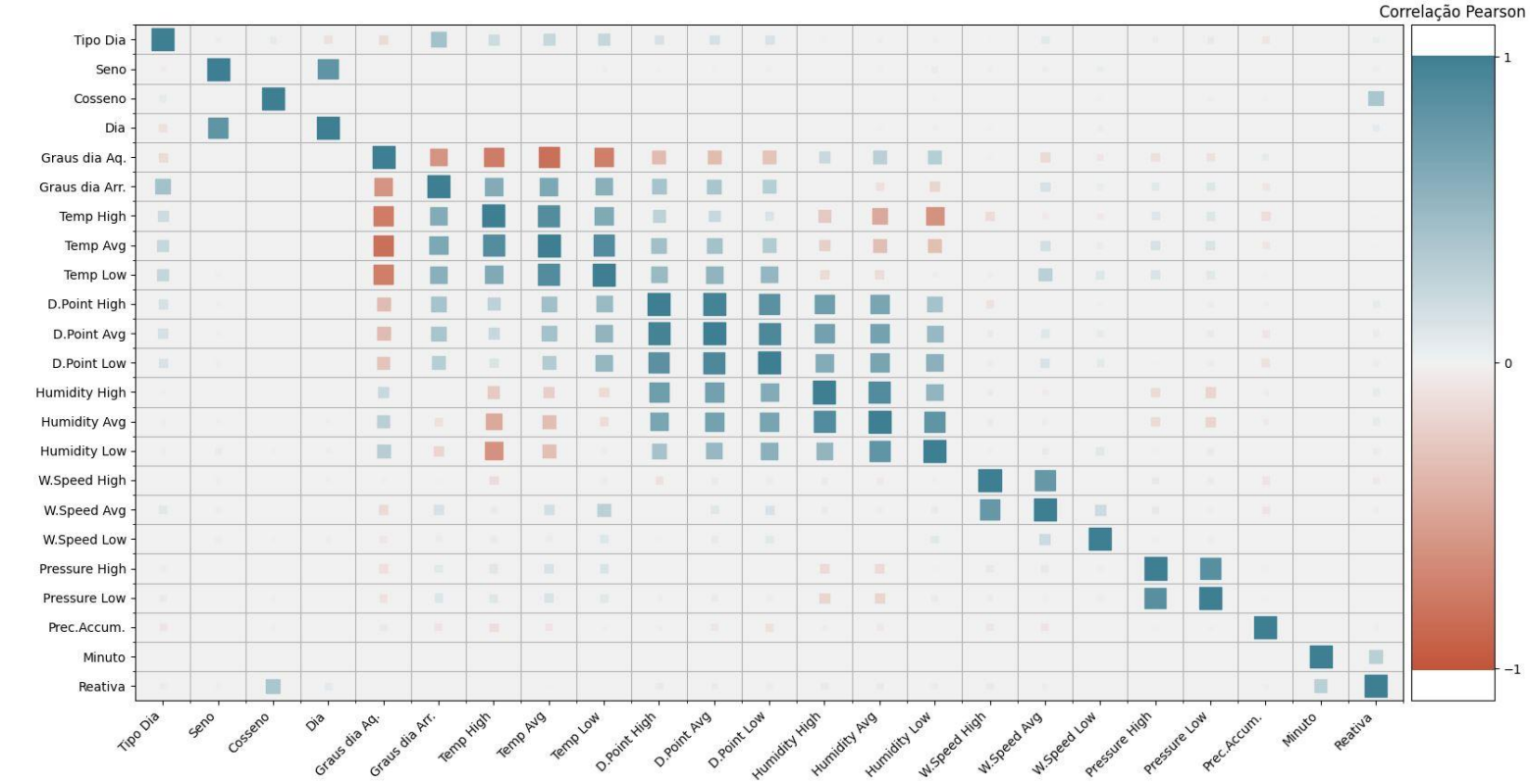
Anexo D- Análise de correlação *features-output*

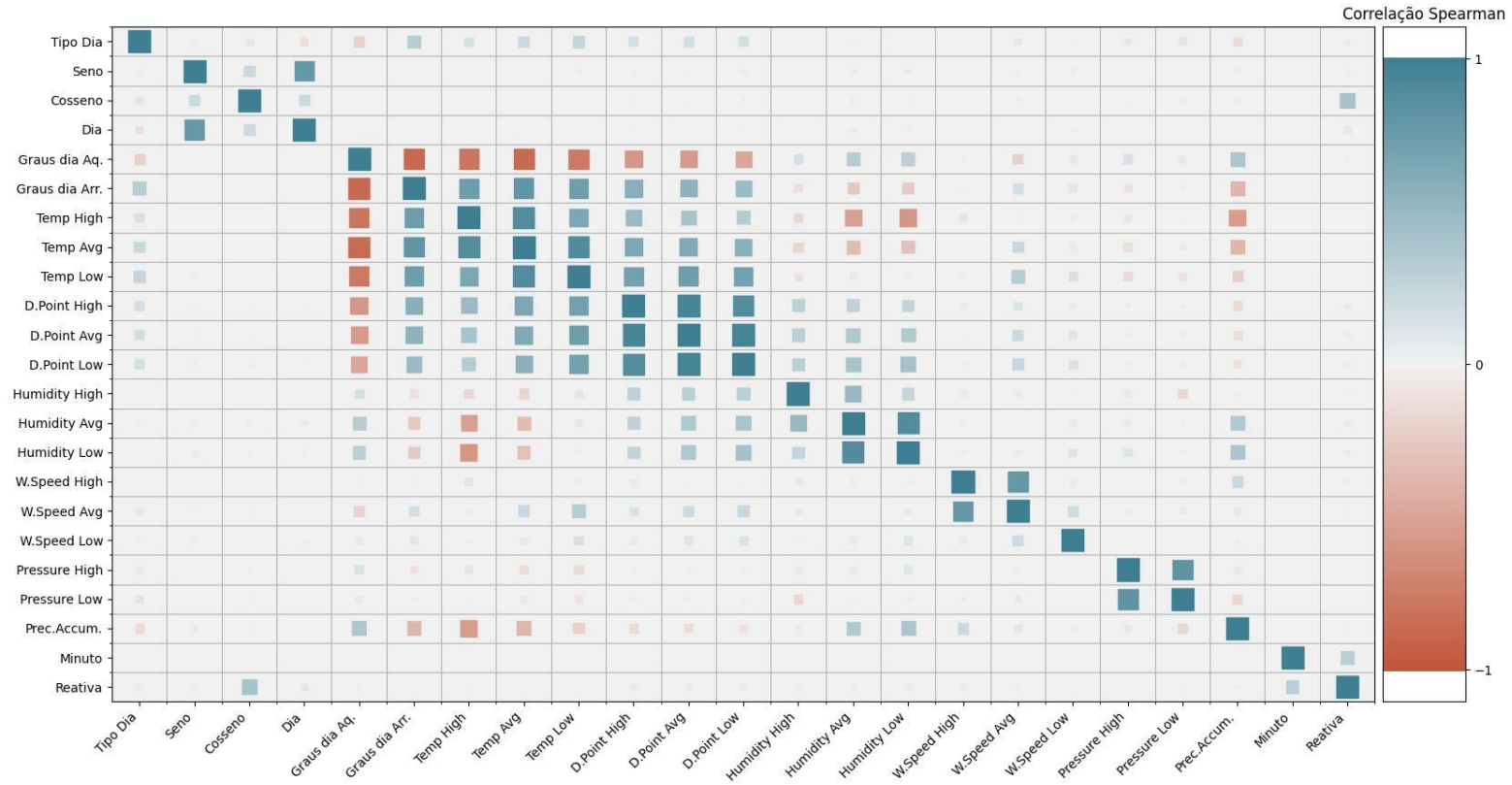
	Todos os Dados						Dados Pré-Pandemia					
Feature	Pearson	Spearman	Pearson Verao	Spearman Verao	Pearson Inverno	Spearman Inverno	Pearson	Spearman	Pearson Verao	Spearman Verao	Pearson Inverno	Spearman Inverno
Temp High	-0,139	-0,179	0,033	0,030	-0,075	-0,126	-0,121	-0,172	-0,005	-0,019	-0,065	-0,137
Temp Avg	-0,212	-0,225	-0,071	-0,070	-0,136	-0,166	-0,155	-0,186	-0,038	-0,052	-0,105	-0,141
Temp Low	-0,217	-0,222	-0,121	-0,123	-0,102	-0,098	-0,146	-0,166	-0,052	-0,061	-0,072	-0,066
D.Point High	-0,053	-0,086	0,011	0,065	-0,082	-0,080	-0,041	-0,090	0,009	0,004	-0,061	-0,055
D.Point Avg	-0,074	-0,110	-0,005	0,011	-0,104	-0,101	-0,053	-0,093	0,004	-0,010	-0,081	-0,067
D.Point Low	-0,087	-0,124	-0,035	-0,041	-0,084	-0,089	-0,039	-0,084	0,013	-0,014	-0,060	-0,061
Humidity High	0,141	0,289	0,094	0,256	0,149	0,248	0,062	0,088	0,022	0,015	0,062	0,125
Humidity Avg	0,138	0,213	0,081	0,174	0,037	0,095	0,073	0,114	0,026	0,044	0,012	0,071
Humidity Low	0,046	0,054	-0,027	-0,033	-0,055	-0,032	0,051	0,069	0,012	0,006	-0,031	0,015
W.Speed High	0,141	0,160	0,220	0,255	0,062	0,072	-0,001	-0,010	0,014	0,010	-0,002	0,001
W.Speed Avg	0,020	0,002	0,081	0,076	0,033	0,048	-0,035	-0,049	0,000	-0,018	-0,012	0,009
W.Speed Low	-0,121	-0,263	-0,140	-0,315	-0,023	-0,083	0,029	0,025	0,070	0,075	-0,023	-0,018
Pressure High	-0,077	-0,192	-0,036	-0,256	-0,086	-0,176	-0,025	-0,004	0,015	0,008	-0,018	-0,078
Pressure Low	-0,076	-0,204	-0,062	-0,265	-0,073	-0,182	-0,002	0,003	0,019	0,033	0,004	-0,076
Prec.Accum	0,035	0,116	0,052	0,033	0,010	0,080	0,015	0,118	0,078	0,062	-0,013	0,076
Graus dia Ag.	0,238	0,248	0,072	0,083	0,163	0,202	0,173	0,223	0,047	0,131	0,103	0,155
Graus dia Arr.	-0,187	-0,183	-0,129	-0,090	0,006	0,001	-0,159	-0,163	-0,131	-0,094	-0,007	-0,033
Tipo Dia	-0,237	-0,360	-0,210	-0,315	-0,216	-0,381	-0,270	-0,434	-0,251	-0,399	-0,265	-0,468
Seno	0,021	0,013	0,018	0,012	0,025	0,007	0,021	0,008	0,022	0,010	0,020	-0,002
Cosseno	0,545	0,528	0,489	0,478	0,650	0,608	0,658	0,626	0,638	0,619	0,707	0,643
Dia	0,058	0,061	0,057	0,061	0,062	0,056	0,070	0,065	0,074	0,069	0,067	0,055
Minuto	0,282	0,269	0,273	0,263	0,298	0,280	0,333	0,338	0,333	0,349	0,337	0,326
Reativa	0,623	0,625	0,594	0,592	0,650	0,634	0,602	0,620	0,549	0,586	0,656	0,659

Anexo E- Ordenação da importância das *features* com os algoritmos *Random Forests*, MARS e *Sequential Feature Selection*

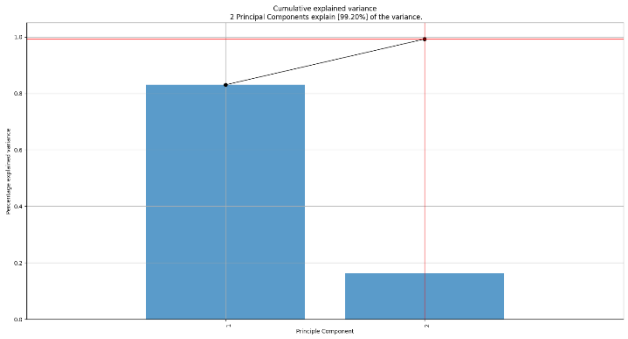
Feature	Coeficiente Random Forest	Ordem Coeficiente R.F.	Coeficientes MARS				Ordem Coeficiente MARS	Ordem Sequential Forward Selection	Ordem Sequential Backward Selection	Ordem Média
			rss	gsv	nb_subsets	Média				
Cosseno	0,00352	19	0,010	0,010	0,047	0,022	9	12	8	12
D.Point Avg	0,00192	29	0,000	0,000	0,000	0,000	18	34	32	28,25
D.Point High	0,00269	23	0,000	0,000	0,013	0,004	15	31	28	24,25
D.Point Low	0,00214	28	0,000	0,000	0,000	0,000	18	28	25	24,75
Dia	0,00152	31	0,000	0,000	0,013	0,004	15	20	10	19
Graus dia Aq.	0,00233	26	0,000	0,000	0,017	0,006	14	19	34	23,25
Graus dia Arr.	0,00137	32	0,000	0,000	0,000	0,000	18	10	12	18
Humidity Avg	0,00264	24	0,000	0,000	0,000	0,000	18	26	36	26
Humidity High	0,00117	33	0,000	0,000	0,000	0,000	18	22	30	25,75
Humidity Low	0,00290	21	0,000	0,000	0,000	0,000	18	27	29	23,75
Minuto	0,00668	11	0,000	0,000	0,000	0,000	18	23	24	19
Prec.Accum.	0,00110	34	0,000	0,000	0,000	0,000	18	15	16	20,75
Pressure High	0,00233	25	0,000	0,000	0,000	0,000	18	35	14	23
Pressure Low	0,00174	30	0,000	0,000	0,000	0,000	18	36	15	24,75
Reativa	0,00498	17	0,000	0,000	0,000	0,000	18	24	27	21,5
Seno	0,00081	35	0,000	0,000	0,000	0,000	18	14	23	22,5
Temp Avg	0,00393	18	0,000	0,000	0,000	0,000	18	30	17	20,75
Temp High	0,00502	16	0,000	0,000	0,000	0,000	18	18	35	21,75
Temp Low	0,00270	22	0,000	0,000	0,013	0,004	15	29	18	21
Tipo Dia	0,06300	4	0,000	0,000	0,000	0,000	18	6	6	8,5
W.Speed Avg	0,00327	20	0,000	0,000	0,000	0,000	18	16	33	21,75
W.Speed High	0,00223	27	0,000	0,000	0,000	0,000	18	25	22	23
W.Speed Low	0,00011	36	0,000	0,000	0,000	0,000	18	33	31	29,5
PCA -1Dia	0,06561	3	0,047	0,047	0,110	0,068	2	3	3	2,75
PCA -1semana	0,62818	1	0,867	0,867	0,093	0,609	1	1	1	1
PCA -2Dias	0,00657	12	0,000	0,000	0,063	0,021	10	21	21	16
PCA -2semanas	0,01076	7	0,000	0,000	0,077	0,026	8	9	11	8,75
PCA -3Dias	0,02029	5	0,000	0,000	0,030	0,010	12	13	9	9,75
PCA -3semanas	0,08984	2	0,040	0,040	0,077	0,052	3	2	2	2,25
PCA -4semanas	0,00729	8	0,000	0,000	0,000	0,000	18	17	19	15,5
PCA -5semanas	0,00674	10	0,003	0,003	0,093	0,033	6	5	7	7
PCA -6Dias	0,01900	6	0,000	0,000	0,093	0,031	7	8	20	10,25
PCA -6semanas	0,00607	15	0,003	0,003	0,033	0,013	11	32	26	21
PCA -7semanas	0,00639	13	0,000	0,000	0,030	0,010	12	11	13	12,25
PCA -8Dias	0,00707	9	0,010	0,010	0,123	0,048	4	4	4	5,25
PCA -8semanas	0,00610	14	0,010	0,010	0,093	0,038	5	7	5	7,75

Anexo F- Análise de correlação entre *features*

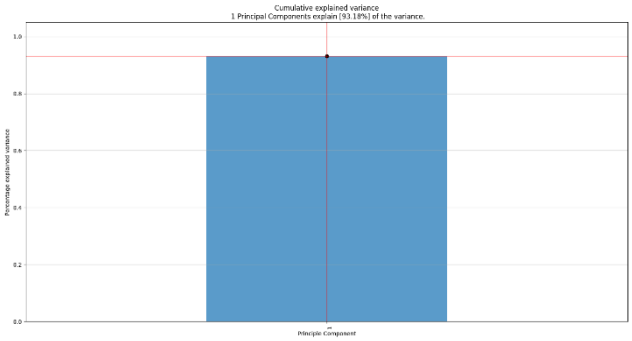




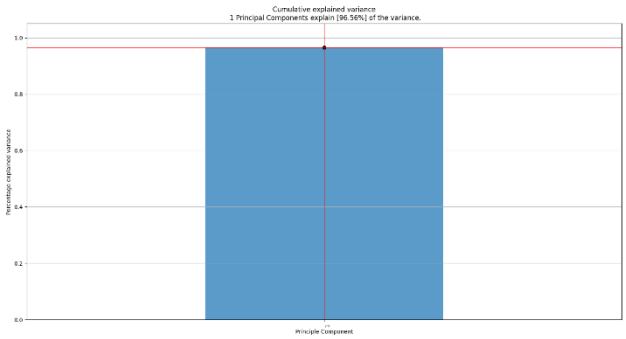
Anexo G- Gráficos de *cumulative explained variance* para a extração das *features* exógenas



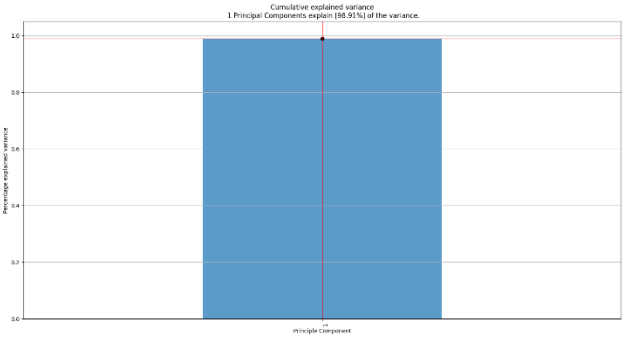
Humidades



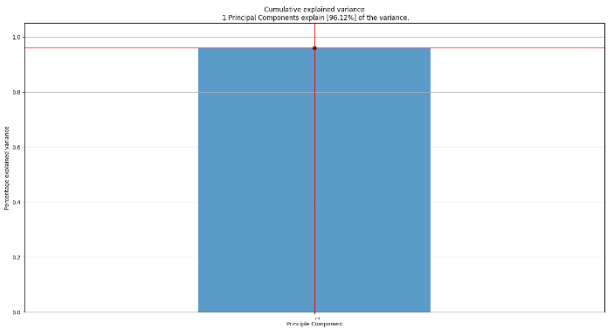
Pressão



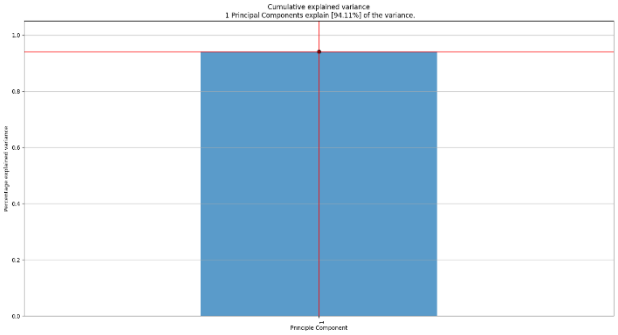
Seno e Dia



Temperaturas

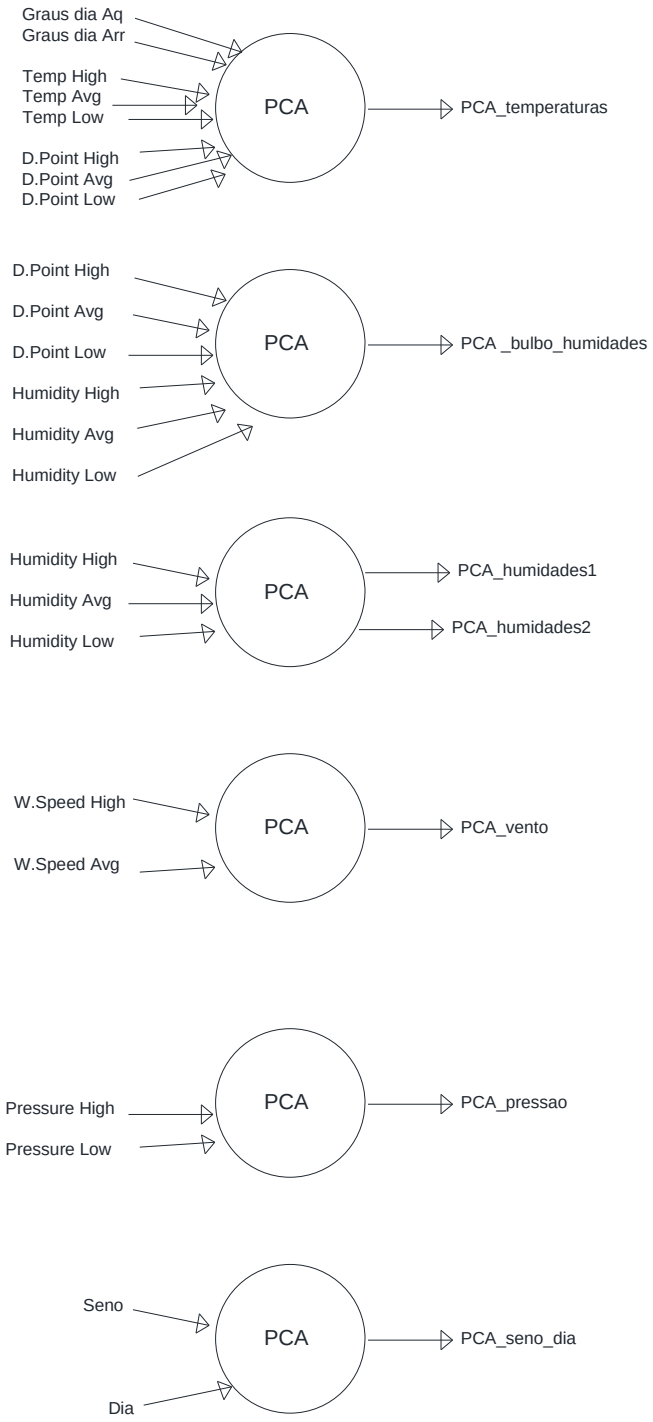


Velocidade do Vento

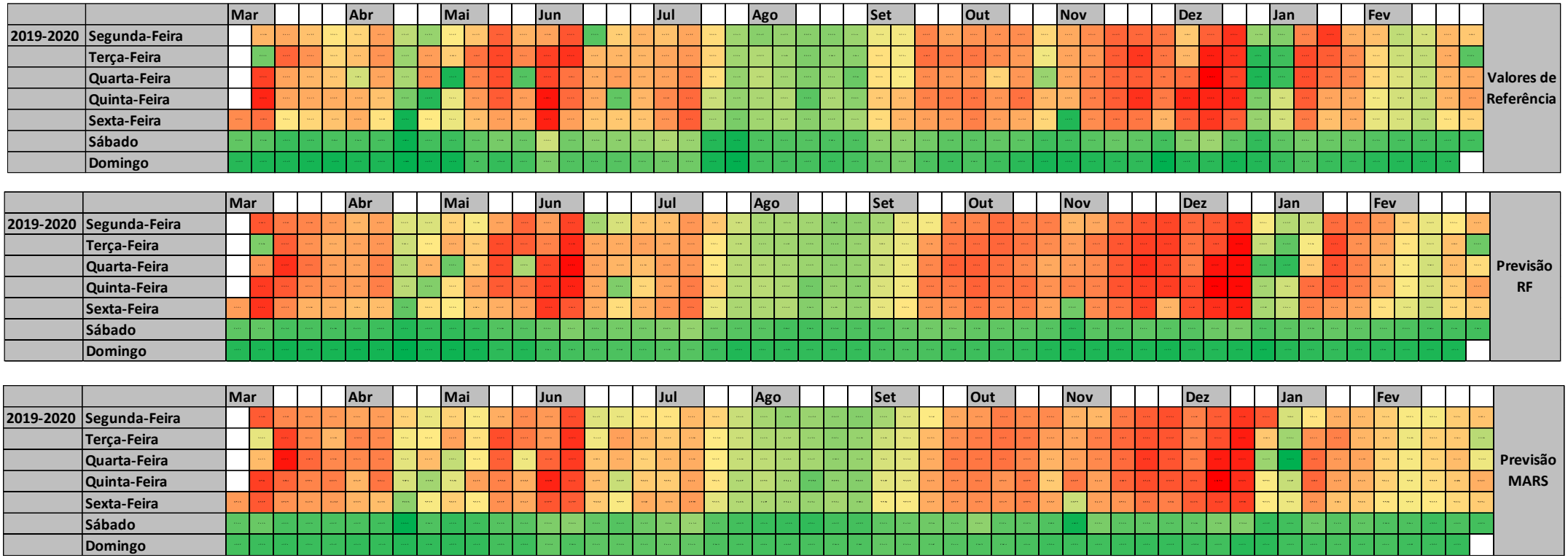


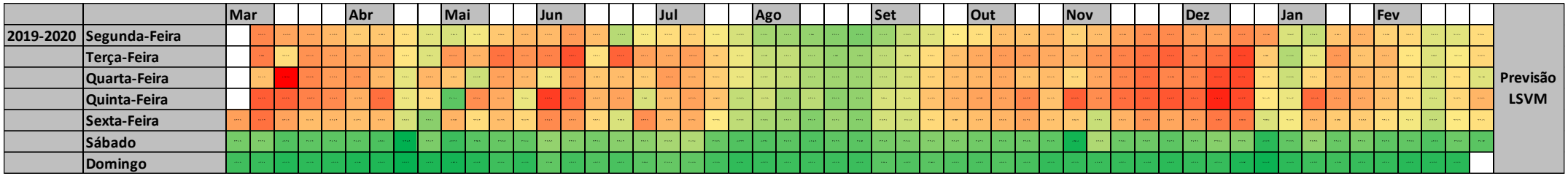
Temperaturas de Ponto de Orvalho e Humidades

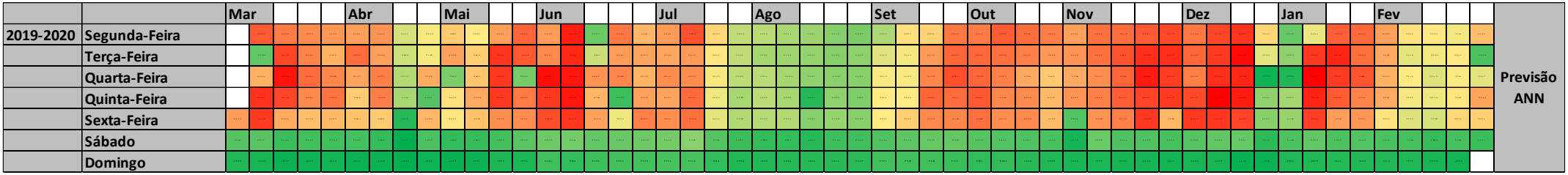
Anexo H- Processo de *feature extraction* das *features* exógenas



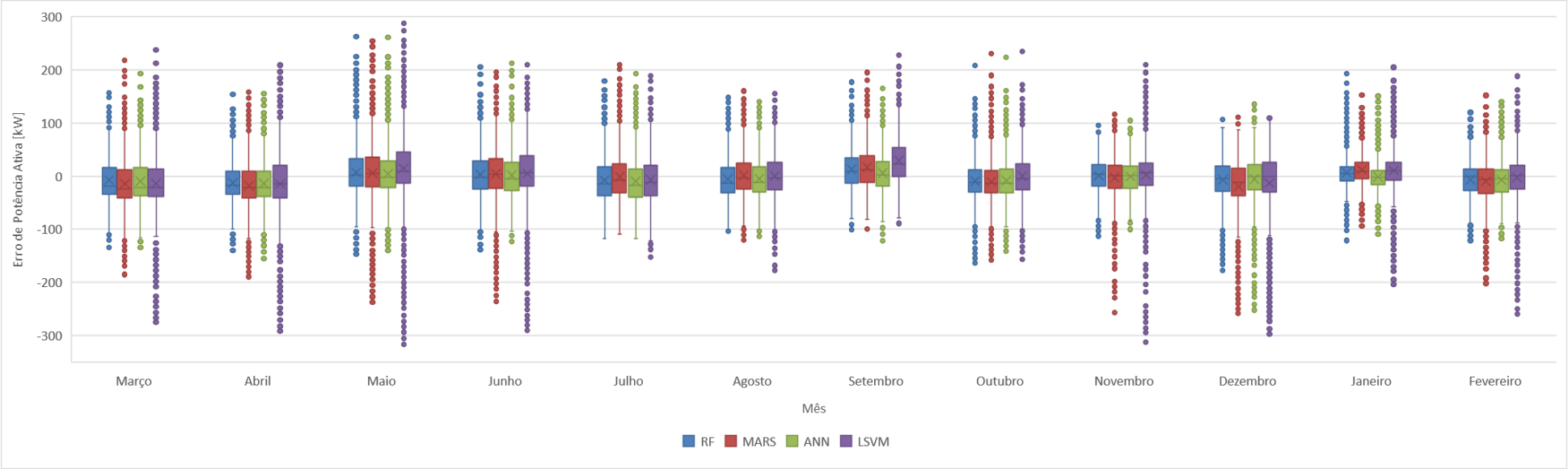
Anexo I- Mapas térmicos do consumo diário de energia

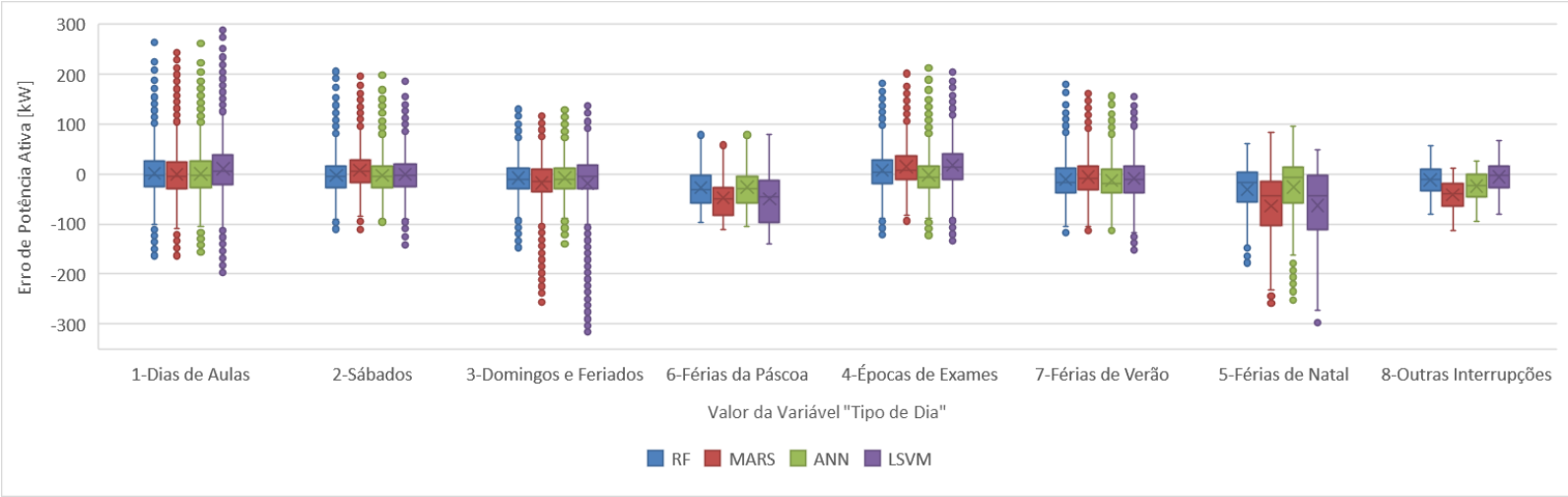



Mar
Abr
Mai
Jun
Jul
Ago
Set
Out
Nov
Dez
Jan
Fev
2019-2020
Segunda-Feira
Terça-Feira
Quarta-Feira
Quinta-Feira
Sexta-Feira
Sábado
Domingo
Previsão
LSVM

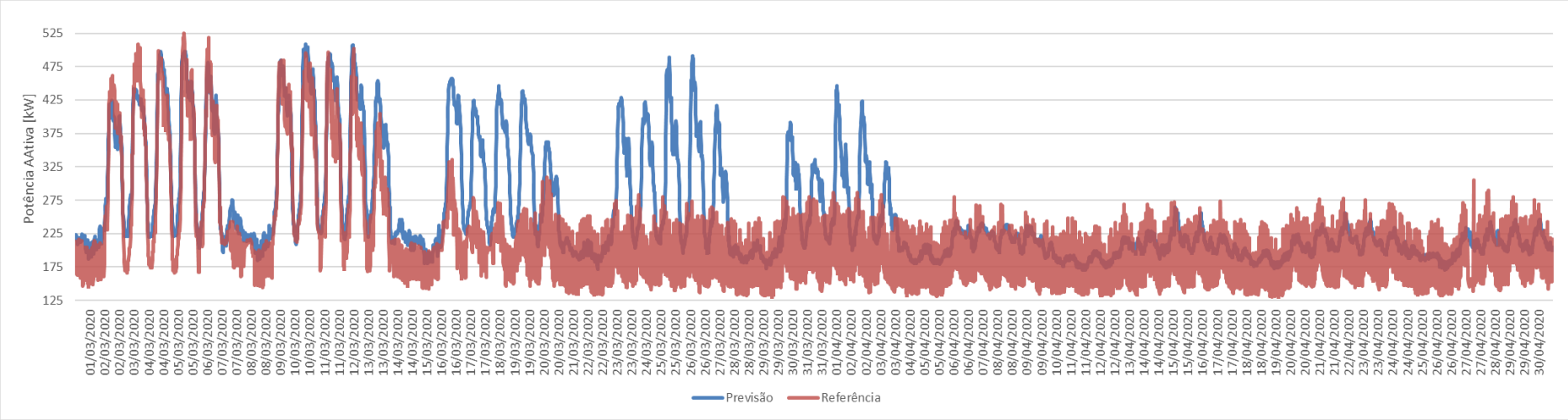

Mar
Abr
Mai
Jun
Jul
Ago
Set
Out
Nov
Dez
Jan
Fev
2019-2020
Segunda-Feira
Terça-Feira
Quarta-Feira
Quinta-Feira
Sexta-Feira
Sábado
Domingo
Previsão
ANN

Anexo J- Gráficos boxplot





Anexo K- Diagrama de carga real vs previsto em pandemia



Anexo L- Exemplo de código 1- Análise de correlação entre *features* e potência

```
# -*- coding: utf-8 -*-

"""
Created on Thu Mar 18 13:37:54 2021
@author: 2192592
"""
#Importação das bibliotecas necessárias
import numpy as np
import pandas as pd
from scipy.stats import pearsonr
from scipy.stats import spearmanr

#Importar dados totais de 2016
features_2016=pd.read_excel('C2-VariaveisClima.xlsx', sheet_name='2016', header=[0,1])
features_2016_array1=np.array(features_2016)
consumo_2016_verao=features_2016_array1[86:303,22:23]
consumo_2016_array1=features_2016_array1[:86,22:23]
consumo_2016_array2=features_2016_array1[303:,22:23]
consumo_2016_inverno=np.concatenate((consumo_2016_array1,consumo_2016_array2))
#Segmentação dos dados de 2016 em Verão/Inverno
features_2016_array2=features_2016_array1[:86,1:22]
features_2016_verao=features_2016_array1[86:303,1:22]
features_2016_array3=features_2016_array1[303:,1:22]
features_2016_inverno=np.concatenate((features_2016_array2,features_2016_array3))
features_2016_verao=pd.DataFrame(features_2016_verao)
features_2016_inverno=pd.DataFrame(features_2016_inverno)
consumo_2016_verao=pd.DataFrame(consumo_2016_verao)
consumo_2016_inverno=pd.DataFrame(consumo_2016_inverno)

#Importar dados totais de 2017
features_2017=pd.read_excel('C2-VariaveisClima.xlsx', sheet_name='2017', header=[0,1])
features_2017_array1=np.array(features_2017)
consumo_2017_verao=features_2017_array1[84:301,22:23]
consumo_2017_array1=features_2017_array1[:84,22:23]
consumo_2017_array2=features_2017_array1[301:,22:23]
consumo_2017_inverno=np.concatenate((consumo_2017_array1,consumo_2017_array2))
#Segmentação dos dados de 2017 em Verão/Inverno
features_2017_array2=features_2017_array1[:84,1:22]
features_2017_verao=features_2017_array1[84:301,1:22]
features_2017_array3=features_2017_array1[301:,1:22]
features_2017_inverno=np.concatenate((features_2017_array2,features_2017_array3))
features_2017_verao=pd.DataFrame(features_2017_verao)
features_2017_inverno=pd.DataFrame(features_2017_inverno)
consumo_2017_verao=pd.DataFrame(consumo_2017_verao)
consumo_2017_inverno=pd.DataFrame(consumo_2017_inverno)

#Importar dados totais de 2018
features_2018=pd.read_excel('C2-VariaveisClima.xlsx', sheet_name='2018', header=[0,1])
features_2018_array1=np.array(features_2018)
consumo_2018_verao=features_2018_array1[83:300,22:23]
consumo_2018_array1=features_2018_array1[:83,22:23]
consumo_2018_array2=features_2018_array1[300:,22:23]
consumo_2018_inverno=np.concatenate((consumo_2018_array1,consumo_2018_array2))
```

```
#Segmentação dos dados de 2018 em Verão/Inverno
features_2018_array2=features_2018_array1[:83,1:22]
features_2018_verao=features_2018_array1[83:300,1:22]
features_2018_array3=features_2018_array1[300:,1:22]
features_2018_inverno=np.concatenate((features_2018_array2,features_2018_array3))
features_2018_verao=pd.DataFrame(features_2018_verao)
features_2018_inverno=pd.DataFrame(features_2018_inverno)
consumo_2018_verao=pd.DataFrame(consumo_2018_verao)
consumo_2018_inverno=pd.DataFrame(consumo_2018_inverno)
#Importar dados totais de 2019
features_2019=pd.read_excel('C2-VariaveisClima.xlsx', sheet_name='2019', header=[0,1])
features_2019_array1=np.array(features_2019)
consumo_2019_verao=features_2019_array1[89:299,22:23]
consumo_2019_array1=features_2019_array1[:89,22:23]
consumo_2019_array2=features_2019_array1[299:,22:23]
consumo_2019_inverno=np.concatenate((consumo_2019_array1,consumo_2019_array2))
#Segmentação dos dados de 2019 em Verão/Inverno
features_2019_array2=features_2019_array1[:89,1:22]
features_2019_verao=features_2019_array1[89:299,1:22]
features_2019_array3=features_2019_array1[299:,1:22]
features_2019_inverno=np.concatenate((features_2019_array2,features_2019_array3))
features_2019_verao=pd.DataFrame(features_2019_verao)
features_2019_inverno=pd.DataFrame(features_2019_inverno)
consumo_2019_verao=pd.DataFrame(consumo_2019_verao)
consumo_2019_inverno=pd.DataFrame(consumo_2019_inverno)

#Juntar dados dos 4 anos
features_list_verao = [features_2016_verao, features_2017_verao,
features_2018_verao,features_2019_features_list_inverno = [features_2016_inverno, features_2017_inverno,
features_2018_inverno,features_consumo_list_verao=[consumo_2016_verao, consumo_2017_verao,
consumo_2018_verao,consumo_2019_verao]
consumo_list_inverno=[consumo_2016_inverno, consumo_2017_inverno,
consumo_2018_inverno,consumo_2019_features_verao = pd.concat(features_list_verao)
features_inverno = pd.concat(features_list_inverno)
consumo_verao = pd.concat(consumo_list_verao)
consumo_inverno = pd.concat(consumo_list_inverno)

#Correlação com coeficiente de Pearson para o Período de Verão
correlacao = pd.DataFrame(columns=['Feature','Pearson Verao','Spearman Verao','Pearson Inverno',
correlacao['Feature']=['Temp High','Temp Avg','Temp Low','D.Point High','D.Point Avg','D.Point Low'
correlacao.iloc[0,1], _ = pearsonr(features_verao.iloc[:,0], consumo_verao)
correlacao.iloc[1,1], _ = pearsonr(features_verao.iloc[:,1], consumo_verao)
correlacao.iloc[2,1], _ = pearsonr(features_verao.iloc[:,2], consumo_verao)
correlacao.iloc[3,1], _ = pearsonr(features_verao.iloc[:,3], consumo_verao)
correlacao.iloc[4,1], _ = pearsonr(features_verao.iloc[:,4], consumo_verao)
correlacao.iloc[5,1], _ = pearsonr(features_verao.iloc[:,5], consumo_verao)
correlacao.iloc[6,1], _ = pearsonr(features_verao.iloc[:,6], consumo_verao)
correlacao.iloc[7,1], _ = pearsonr(features_verao.iloc[:,7], consumo_verao)
correlacao.iloc[8,1], _ = pearsonr(features_verao.iloc[:,8], consumo_verao)
correlacao.iloc[9,1], _ = pearsonr(features_verao.iloc[:,9], consumo_verao)
correlacao.iloc[10,1], _ = pearsonr(features_verao.iloc[:,10], consumo_verao)
correlacao.iloc[11,1], _ = pearsonr(features_verao.iloc[:,11], consumo_verao)
correlacao.iloc[12,1], _ = pearsonr(features_verao.iloc[:,12], consumo_verao)
correlacao.iloc[13,1], _ = pearsonr(features_verao.iloc[:,13], consumo_verao)
correlacao.iloc[14,1], _ = pearsonr(features_verao.iloc[:,14], consumo_verao)
correlacao.iloc[15,1], _ = pearsonr(features_verao.iloc[:,15], consumo_verao)
correlacao.iloc[16,1], _ = pearsonr(features_verao.iloc[:,16], consumo_verao)
correlacao.iloc[17,1], _ = pearsonr(features_verao.iloc[:,17], consumo_verao)
correlacao.iloc[18,1], _ = pearsonr(features_verao.iloc[:,18], consumo_verao)
correlacao.iloc[19,1], _ = pearsonr(features_verao.iloc[:,19], consumo_verao)
correlacao.iloc[20,1], _ = pearsonr(features_verao.iloc[:,20], consumo_verao)
```

```
#Correlacao com coeficiente de Spierman para o Período de Verão
correlacao.iloc[0,2], _ = spearmanr(features_verao.iloc[:,0], consumo_verao)
correlacao.iloc[1,2], _ = spearmanr(features_verao.iloc[:,1], consumo_verao)
correlacao.iloc[2,2], _ = spearmanr(features_verao.iloc[:,2], consumo_verao)
correlacao.iloc[3,2], _ = spearmanr(features_verao.iloc[:,3], consumo_verao)
correlacao.iloc[4,2], _ = spearmanr(features_verao.iloc[:,4], consumo_verao)
correlacao.iloc[5,2], _ = spearmanr(features_verao.iloc[:,5], consumo_verao)
correlacao.iloc[6,2], _ = spearmanr(features_verao.iloc[:,6], consumo_verao)
correlacao.iloc[7,2], _ = spearmanr(features_verao.iloc[:,7], consumo_verao)
correlacao.iloc[8,2], _ = spearmanr(features_verao.iloc[:,8], consumo_verao)
correlacao.iloc[9,2], _ = spearmanr(features_verao.iloc[:,9], consumo_verao)
correlacao.iloc[10,2], _ = spearmanr(features_verao.iloc[:,10], consumo_verao)
correlacao.iloc[11,2], _ = spearmanr(features_verao.iloc[:,11], consumo_verao)
correlacao.iloc[12,2], _ = spearmanr(features_verao.iloc[:,12], consumo_verao)
correlacao.iloc[13,2], _ = spearmanr(features_verao.iloc[:,13], consumo_verao)
correlacao.iloc[14,2], _ = spearmanr(features_verao.iloc[:,14], consumo_verao)
correlacao.iloc[15,2], _ = spearmanr(features_verao.iloc[:,15], consumo_verao)
correlacao.iloc[16,2], _ = spearmanr(features_verao.iloc[:,16], consumo_verao)
correlacao.iloc[17,2], _ = spearmanr(features_verao.iloc[:,17], consumo_verao)
correlacao.iloc[18,2], _ = spearmanr(features_verao.iloc[:,18], consumo_verao)
correlacao.iloc[19,2], _ = spearmanr(features_verao.iloc[:,19], consumo_verao)
correlacao.iloc[20,2], _ = spearmanr(features_verao.iloc[:,20], consumo_verao)

#Correlação com coeficiente de Pearson para o Período de Inverno
correlacao.iloc[0,3], _ = pearsonr(features_inverno.iloc[:,0], consumo_inverno)
correlacao.iloc[1,3], _ = pearsonr(features_inverno.iloc[:,1], consumo_inverno)
correlacao.iloc[2,3], _ = pearsonr(features_inverno.iloc[:,2], consumo_inverno)
correlacao.iloc[3,3], _ = pearsonr(features_inverno.iloc[:,3], consumo_inverno)
correlacao.iloc[4,3], _ = pearsonr(features_inverno.iloc[:,4], consumo_inverno)
correlacao.iloc[5,3], _ = pearsonr(features_inverno.iloc[:,5], consumo_inverno)
correlacao.iloc[6,3], _ = pearsonr(features_inverno.iloc[:,6], consumo_inverno)
correlacao.iloc[7,3], _ = pearsonr(features_inverno.iloc[:,7], consumo_inverno)
correlacao.iloc[8,3], _ = pearsonr(features_inverno.iloc[:,8], consumo_inverno)
correlacao.iloc[9,3], _ = pearsonr(features_inverno.iloc[:,9], consumo_inverno)
correlacao.iloc[10,3], _ = pearsonr(features_inverno.iloc[:,10], consumo_inverno)
correlacao.iloc[11,3], _ = pearsonr(features_inverno.iloc[:,11], consumo_inverno)
correlacao.iloc[12,3], _ = pearsonr(features_inverno.iloc[:,12], consumo_inverno)
correlacao.iloc[13,3], _ = pearsonr(features_inverno.iloc[:,13], consumo_inverno)
correlacao.iloc[14,3], _ = pearsonr(features_inverno.iloc[:,14], consumo_inverno)
correlacao.iloc[15,3], _ = pearsonr(features_inverno.iloc[:,15], consumo_inverno)
correlacao.iloc[16,3], _ = pearsonr(features_inverno.iloc[:,16], consumo_inverno)
correlacao.iloc[20,3], _ = pearsonr(features_inverno.iloc[:,20], consumo_inverno)
```

```
#Correlacao com coeficiente de Spierman para o Período de Inverno
correlacao.iloc[0,4], _ = spearmanr(features_inverno.iloc[:,0], consumo_inverno)
correlacao.iloc[1,4], _ = spearmanr(features_inverno.iloc[:,1], consumo_inverno)
correlacao.iloc[2,4], _ = spearmanr(features_inverno.iloc[:,2], consumo_inverno)
correlacao.iloc[3,4], _ = spearmanr(features_inverno.iloc[:,3], consumo_inverno)
correlacao.iloc[4,4], _ = spearmanr(features_inverno.iloc[:,4], consumo_inverno)
correlacao.iloc[5,4], _ = spearmanr(features_inverno.iloc[:,5], consumo_inverno)
correlacao.iloc[6,4], _ = spearmanr(features_inverno.iloc[:,6], consumo_inverno)
correlacao.iloc[7,4], _ = spearmanr(features_inverno.iloc[:,7], consumo_inverno)
correlacao.iloc[8,4], _ = spearmanr(features_inverno.iloc[:,8], consumo_inverno)
correlacao.iloc[9,4], _ = spearmanr(features_inverno.iloc[:,9], consumo_inverno)
correlacao.iloc[10,4], _ = spearmanr(features_inverno.iloc[:,10], consumo_inverno)
correlacao.iloc[11,4], _ = spearmanr(features_inverno.iloc[:,11], consumo_inverno)
correlacao.iloc[12,4], _ = spearmanr(features_inverno.iloc[:,12], consumo_inverno)
correlacao.iloc[13,4], _ = spearmanr(features_inverno.iloc[:,13], consumo_inverno)
correlacao.iloc[14,4], _ = spearmanr(features_inverno.iloc[:,14], consumo_inverno)
correlacao.iloc[15,4], _ = spearmanr(features_inverno.iloc[:,15], consumo_inverno)
correlacao.iloc[16,4], _ = spearmanr(features_inverno.iloc[:,16], consumo_inverno)
correlacao.iloc[17,4], _ = spearmanr(features_inverno.iloc[:,17], consumo_inverno)
correlacao.iloc[18,4], _ = spearmanr(features_inverno.iloc[:,18], consumo_inverno)
correlacao.iloc[19,4], _ = spearmanr(features_inverno.iloc[:,19], consumo_inverno)
correlacao.iloc[20,4], _ = spearmanr(features_inverno.iloc[:,20], consumo_inverno)

"Guardar DataFrame em excel"
with pd.ExcelWriter('Correlacao.xlsx', engine='openpyxl', mode='a') as writer:
        correlacao.to_excel(writer,sheet_name='4 anos com Periodos')
```

Anexo M- Exemplo de código 2- Modelo MARS após *feature selection*

```
# -*- coding: utf-8 -*-
"""
Created on Mon Oct 11 16:45:17 2021
@author: 2192592
"""
#Importação das bibliotecas necessárias
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler
import pickle
import time
from pyearth import Earth
from sklearn.model_selection import cross_validate
from statistics import mean
from sklearn.model_selection import KFold
from sklearn.pipeline import Pipeline

#Definição de variável usada para contar tempo do script
t = time.process_time()
#Definição dos períodos de treino, validação e teste (em dias)
train_size=1155
validation_size=0 #Zero porque se usou validação cruzada
test_size=366

#Seleção da normalizacao pretendida
normalizacao=None
while normalizacao !='1' and normalizacao !='2':
        normalizacao=input('Insira a normalização pretendidoa (1 (Min-Max) ou 2 (Standard)):')
if (normalizacao=='1'):
        normalizacao='min-max'
elif (normalizacao=='2'):
        normalizacao='standard'
if normalizacao!='min-max' and normalizacao!='standard':
        print('A normalização inserida tem de ser 1 ou 2')
        print('Normalização Selecionado: ' + str(normalizacao))

#Normalização dos dados
normalizacao='standard'
if(normalizacao=='min-max'):
        scaler = MinMaxScaler()
elif(normalizacao=='standard'):
        scaler = StandardScaler()
else:
        print('Erro na normalização')

#Importação dos dados de potência ativa
consumo_no_norm=pd.read_excel('Dados3.xlsx',sheet_name='C2-Ativa-1D')
consumo_no_norm=np.array(consumo_no_norm)
consumo_no_norm=consumo_no_norm[:(56+train_size+validation_size+test_size)*96,3:4]

#Definição das saídas não normalizadas
Outputs_no_norm=consumo_no_norm[56*96:,:]
Outputs_Train_no_norm=Outputs_no_norm[0:train_size*96,:]
Outputs_Test_no_norm=Outputs_no_norm[(train_size+validation_size)*96:,:]
```

```
#Importação dos dados da potência reativa
reativa_no_norm=pd.read_excel('Dados3.xlsx',sheet_name='C2-Reativa-1D')
reativa_no_norm=np.array(reativa_no_norm)
reativa_no_norm=reativa_no_norm[:(56+train_size+validation_size+test_size)*96,3:4]
#Importação das restantes variaveis exogenas
features_no_norm=pd.read_excel('Features2.xlsx',sheet_name='Folha2')
features_no_norm=np.array(features_no_norm)
features_no_norm=features_no_norm[:(train_size+validation_size+test_size)*96,4:]

# Importação de variáveis referentes a registos anteriores
pca_dia=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_1dia')
pca_dia=np.array(pca_dia)
pca_dia=pd.DataFrame(pca_dia)
pca_semana=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_1semana')
pca_semana=np.array(pca_semana)
pca_semana=pd.DataFrame(pca_semana)
pca_2dias=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_2dias')
pca_2dias=np.array(pca_2dias)
pca_2dias=pd.DataFrame(pca_2dias)
pca_2semanas=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_2semanas')
pca_2semanas=np.array(pca_2semanas)
pca_2semanas=pd.DataFrame(pca_2semanas)
pca_3dias=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_3dias')
pca_3dias=np.array(pca_3dias)
pca_3dias=pd.DataFrame(pca_3dias)
pca_3semanas=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_3semanas')
pca_3semanas=np.array(pca_3semanas)
pca_3semanas=pd.DataFrame(pca_3semanas)
pca_4semanas=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_4semanas')
pca_4semanas=np.array(pca_4semanas)
pca_4semanas=pd.DataFrame(pca_4semanas)
pca_5semanas=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_5semanas')
pca_5semanas=np.array(pca_5semanas)
pca_5semanas=pd.DataFrame(pca_5semanas)
pca_6dias=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_6dias')
pca_6dias=np.array(pca_6dias)
pca_6dias=pd.DataFrame(pca_6dias)
pca_6semanas=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_6semanas')
pca_6semanas=np.array(pca_6semanas)
pca_6semanas=pd.DataFrame(pca_6semanas)
pca_7semanas=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_7semanas')
pca_7semanas=np.array(pca_7semanas)
pca_7semanas=pd.DataFrame(pca_7semanas)
pca_8dias=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_8dias')
pca_8dias=np.array(pca_8dias)
pca_8dias=pd.DataFrame(pca_8dias)
pca_8semanas=pd.read_excel('PCA_historico2.xlsx',sheet_name='pca_8semanas')
pca_8semanas=np.array(pca_8semanas)
pca_8semanas=pd.DataFrame(pca_8semanas)

#Concatenar variáveis referentes a registos anteriores
pcs_no_norm=pd.concat([pca_dia,pca_semana,pca_2dias,pca_2semanas,pca_3dias,pca_3semanas,pca_4semanas,pca_
5semanas,pca_6dias,pca_6semanas,pca_7semanas,pca_8dias,pca_8semanas],axis=1,ignore_index=True)
#Definição dos nomes de todas as features
Nomes=['Tipo Dia','Seno','Cosseno','Dia', 'Graus dia Aq.','Graus dia Arr.','Temp High','Temp Avg','Temp Low' ,'D.Point
High','D.Point Avg','D.Point Low','Humidity High','Humidity Avg','Humidity Low','W.Speed High','W.Speed
Avg','W.Speed Low','Pressure High','Pressure Low', 'Prec.Accum.','Minuto','Reativa','PCA -1Dia','PCA -
1semana','PCA -2Dias','PCA -2semanas','PCA -3Dias','PCA -3semanas','PCA -4semanas','PCA -5semanas','PCA -
6Dias','PCA -6semanas','PCA -7semanas','PCA -8Dias','PCA -8semanas']
#Definição das entradas não normalizadas
Inputs_no_norm=features_no_norm
Inputs_no_norm=np.array(Inputs_no_norm)
Inputs_no_norm=pd.DataFrame(Inputs_no_norm)
```

```
#Definição das variáveis referentes à potência reativa
reativa_feature_no_norm=reativa_no_norm[(49*96):len(reativa_no_norm)-(7*96),:]
reativa_feature_no_norm=pd.DataFrame(reativa_feature_no_norm)
#Concatenar reativa com restantes features
Inputs_no_norm=pd.concat([Inputs_no_norm,reativa_feature_no_norm,pcs_no_norm], axis = 1)
Inputs_no_norm=np.array(Inputs_no_norm)
Inputs_no_norm=pd.DataFrame(Inputs_no_norm,columns=Nomes)

#Descarte de variáveis não normalizadas de acordo com as conclusões da feature selection
Inputs_no_norm=Inputs_no_norm.drop(['PCA -2Dias','PCA -6semanas','Minuto','Graus dia Arr.','Graus dia Aq.',
,'Reativa','Seno','Dia','Temp High','Temp Avg','Temp Low','D.Point High','D.Point Avg','D.Point Low','Humidity
High','Humidity Avg','Humidity Low','W.Speed High','W.Speed Avg','W.Speed Low','Pressure High','Pressure Low',
'Prec.Accum.'],axis=1)

#Segmentação das entradas não normalizadas em treino/teste
Inputs_no_norm_df=Inputs_no_norm
Inputs_no_norm=np.array(Inputs_no_norm)
Inputs_no_norm_Train=Inputs_no_norm[0:train_size*96,:]
Inputs_no_norm_Val=Inputs_no_norm[train_size*96:(train_size+validation_size)*96,:]
Inputs_no_norm_Test=Inputs_no_norm[(train_size+validation_size)*96:,:]

#Inicialização do modelo MARS
criteria = ('rss', 'gcv', 'nb_subsets')
model = Earth(feature_importance_type=criteria)

#Criação do pipeline com modelo e normalização pretendida
pipeline = Pipeline([('transformer', scaler), ('estimator', model)])

#Escolha do K da validação cruzada K-Fold
numero_folds=3

#Definição das métricsa de erro pretendidas
my_scoring={'MAE': 'neg_mean_absolute_error',
            'MSE': 'neg_mean_squared_error',
            'RMSE': 'neg_root_mean_squared_error',
            'MAPE':'neg_mean_absolute_percentage_error'}
#Definição da estratégia de validação cruzada K-Fold
meu_cv = KFold(n_splits=numero_folds, random_state=None, shuffle=False)
#Obtenção das métricas de validação
scores = cross_validate(pipeline, Inputs_no_norm_Train, Outputs_Train_no_norm.ravel(), scoring=my_scoring,
                        cv=meu_cv, return_train_score=True,return_estimator=True)

#Obtenção dos coeficientes das relevâncias de cada feature
estimator=scores.get('estimator')
estimator_fold1=estimator[0]
estimator_fold1=estimator_fold1.named_steps['estimator']
estimator_fold2=estimator[1]
estimator_fold2=estimator_fold2.named_steps['estimator']
estimator_fold3=estimator[2]
estimator_fold3=estimator_fold3.named_steps['estimator']
sumario1=estimator_fold1.summary_feature_importances(sort_by='gcv')
sumario2=estimator_fold2.summary_feature_importances(sort_by='gcv')
sumario3=estimator_fold3.summary_feature_importances(sort_by='gcv')
importancia1=sumario1.split('\n')
importancia2=sumario2.split('\n')
importancia3=sumario3.split('\n')
importancias1=pd.DataFrame(importancia1)
importancias2=pd.DataFrame(importancia2)
importancias3=pd.DataFrame(importancia3)
```

```
#Cálculo de métricas de erro de validação
mape_validacao=mean(scores.get('test_MAPE'))
mae_validacao=mean(scores.get('test_MAE'))
mse_validacao=mean(scores.get('test_MSE'))
rmse_validacao=mean(scores.get('test_RMSE'))
mape_treino=mean(scores.get('train_MAPE'))
mae_treino=mean(scores.get('train_MAE'))
mse_treino=mean(scores.get('train_MSE'))
rmse_treino=mean(scores.get('train_RMSE'))
tempo_processo=sum(scores.get('fit_time'))

#Criação e preenchimento de lista com resultados de métricas de erro
metricas=[]
metricas.append(mae_validacao)
metricas.append(mse_validacao)
metricas.append(rmse_validacao)
metricas.append(mape_validacao)
metricas.append(mae_treino)
metricas.append(mse_treino)
metricas.append(rmse_treino)
metricas.append(mape_treino)
metricas.append(tempo_processo)

#Criação e preenchimento de DataFrame com resultados das métricas de erro
metricas_df= pd.DataFrame(columns=['Simulacao','MAE Validação','MSE Validação','RMSE Validação','MAPE Validação',
'MAE Treino', 'MSE Treino','RMSE Treino','MAPE Treino','Tempo Processo Treino'])
metricas_df['Simulacao']=['1','2']
metricas_df.iloc[0,1]=metricas[0]
metricas_df.iloc[0,2]=metricas[1]
metricas_df.iloc[0,3]=metricas[2]
metricas_df.iloc[0,4]=metricas[3]
metricas_df.iloc[0,5]=metricas[4]
metricas_df.iloc[0,6]=metricas[5]
metricas_df.iloc[0,7]=metricas[6]
metricas_df.iloc[0,8]=metricas[7]
metricas_df.iloc[0,9]=metricas[8]

#Guardar modelo com biblioteca pickle
Modelo_MARS_todas_features= 'Modelos Finais\Modelo_MARS_features.sav'
pickle.dump(model, open(Modelo_MARS_todas_features, 'wb'))

#Guardar resultados das métricas e relevância de features em excel
with pd.ExcelWriter('resultados_MARS_features_final.xlsx', engine='openpyxl', mode='w') as writer:
                        metricas_df.to_excel(writer,sheet_name='metricas')
                        importancias1.to_excel(writer,sheet_name='sumario fold1')
                        importancias2.to_excel(writer,sheet_name='sumario fold2')
                        importancias3.to_excel(writer,sheet_name='sumario fold3')

#Cálculo do tempo gasto na simulação
total_elapsed_time = time.process_time() - t
print('elapsed time='+str(total_elapsed_time))
```