INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA DE INTELIGÊNCIA ARTIFICIAL




Explicabilidade automática e adaptativa para modelos de aprendizagem máquina

ANTÓNIO MORADO RAMOS
novembro de 2022

P.PORTO

INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA DE INTELIGÊNCIA ARTIFICIAL




Explicabilidade automática e adaptativa para modelos de aprendizagem máquina

ANTÓNIO MORADO RAMOS
Novembro de 2022

P.PORTO

Explicabilidade automática e adaptativa para modelos de aprendizagem máquina

António Morado Ramos

Aluno nº: 1970314

Dissertação para obtenção do Grau de

Mestre em Engenharia de Inteligência Artificial

Orientador: Doutora Zita Maria Almeida do Vale, Professora Coordenadora Principal do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Co-orientador: Doutor Tiago Manuel Campelos Ferreira Pinto, Professor Adjunto Convidado do Instituto Superior de Engenharia do Instituto Politécnico do Porto e Professor Auxiliar da Universidade de Trás-os-Montes e Alto Douro

Supervisor: Brígida Constança Correia Teixeira, Investigadora e Professora Adjunta Convidada do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Júri:

Presidente:

Doutor Luiz Felipe Rocha de Faria, Professor Coordenador do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Vogais:

Doutora Zita Maria Almeida do Vale, Professora Coordenadora Principal do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Doutor Gabriel José Lopes dos Santos, Investigador Auxiliar do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Porto, outubro 2022

Dedicatória

Dedico aos meus pais esta dissertação do Mestrado em Engenharia de Inteligência Artificial, que com trabalho e sacrifício me proporcionaram a oportunidade de aceder ao ensino universitário e assim poder abrir portas para novas oportunidades de realização pessoal que ambos não tiveram.

Resumo

O setor energético está a atravessar por um período de profundo processo de mudança estrutural devido à transição energética, na qual a digitalização é um dos pilares desta transição. No pilar da digitalização, é reconhecido o importante contributo da inteligência artificial, nomeadamente no campo da aprendizagem máquina. Contudo, a dificuldade em compreender como muitos dos modelos de aprendizagem máquina obtêm os resultados torna-se um grande desafio, principalmente em processos de tomada de decisão.

Muitas das previsões realizadas no setor energético, como por exemplo de consumo de eletricidade, consideram conjuntos de dados do tipo série temporal e utilizam modelos baseados em regressão. Contudo, verifica-se pouca aplicação dos métodos explicativos que contemplam estes dois temas. Este trabalho procura contribuir para melhor compreensão das previsões de consumo de eletricidade obtidas por modelos de aprendizagem máquina, com propostas de geração de explicações destas previsões, através de métodos explicativos.

Neste trabalho são exploradas as explicações visuais obtidas de dois métodos explicativos, o LIME e o SHAP, escolhidos para gerar explicações das previsões de consumo de eletricidade de dois modelos de aprendizagem máquina baseados em regressão. Estes métodos explicativos foram selecionados com base no estado da arte apresentado neste trabalho. Foi selecionado um conjunto de dados real, do tipo série temporal, com registos de consumos de três dispositivos existentes em cinco zonas de um edifício e que contribuem para o consumo de eletricidade deste: ar condicionado, tomadas e lâmpadas. Pretende-se avaliar como o uso destas explicações visuais possibilitam a compreensão de quais os atributos do conjunto de dados que os modelos de aprendizagem máquina atribuem maior importância no processo aprendizagem da previsão do consumo de eletricidade. Outro aspeto a ser avaliado é o tempo de processamento da geração das explicações.

Os resultados mostram que o SHAP é o mais robusto no sentido em que apresenta sempre os mesmos resultados em diferentes interações, ao contrário do LIME. Contudo, o primeiro apresenta maior degradação no tempo de processamento. Ambos identificam os atributos relativos aos dispositivos ar condicionado e tomadas como aqueles que os modelos de previsão consideram os mais importantes para a previsão do consumo, contudo com diferente ordem de importância em cada método explicativo.

Palavras-chave: *Explainable Artifical Inteligence*, *Machine Learning*, Sistemas de Energia Elétrica

Abstract

The energy sector is going through a period of profound structural change due to the energy transition, in which digitalization is one of the pillars of this transition. In the digitization pillar, the important contribution of artificial intelligence is recognized, namely in the field of machine learning. However, the difficulty in understanding how many of the machine learning models obtain the results becomes a great challenge, especially in decision-making processes.

Many of the forecasts made in the energy sector, such as electricity consumption, consider time series data sets and use regression-based models. However, there is little application of explanatory methods that contemplate these two subjects. This work seeks to contribute to a better understanding of electricity consumption forecasts obtained by machine learning models, with proposals for generating explanations of these forecasts, through explanatory methods.

In this work we explore the visual explanations obtained from two explanatory methods, LIME and SHAP, chosen to generate explanations of electricity consumption predictions from two regression-based machine learning models. These explanatory methods were selected based on the state of the art presented in this work. A real time series dataset was selected, with consumption records of three devices existing in five areas of a building and that contribute to its electricity consumption: air conditioning, sockets and lamps. It is intended to evaluate how the use of these visual explanations make it possible to understand which feature of the dataset that machine learning models identify as most important in the learning process of forecasting electricity consumption. Another aspect to be evaluated is the processing time of generating the explanations.

The results show that SHAP is the most robust in the sense that it always presents the same results in different interactions, unlike LIME. However, the first one presents greater degradation in the processing time. Both identify the features related to air conditioning devices and taken as those that the forecasting models consider the most important for forecasting consumption, however with different order of importance in each explanatory method.

Keywords: Explainable Artificial Intelligence, Machine Learning, Electric Power Systems

Agradecimentos

Em primeiro lugar, quero agradecer à Doutora Zita Vale e ao Doutor Tiago Pinto pela oportunidade em realizar este trabalho no contexto de um projeto desenvolvido pelo GECAD e pelas suas orientações.

Um especial agradecimento à professora Brígida por dedicar parte do seu tempo precioso na orientação e supervisão deste trabalho ao longo destes meses. A professora Brígida esteve sempre disponível para me ajudar com troca de ideias, expondo diferentes perspetivas e sempre com uma visão crítica e rigorosa. Foram, também, fonte grande inspiração para mim as suas palavras motivadoras, principalmente nos momentos difíceis deste percurso.

Agradeço ao professor Carlos Ramos pela iniciativa do Mestrado em Engenharia em Inteligência Artificial que me abriu novos horizontes na área da Inteligência Artificial assim como aos professores, sempre disponíveis para ajudar. Um agradecimento à entidade patronal Cleva pela flexibilidade em termos de esforço laboral mostrando uma atitude de compreensão e valorização pela minha necessidade de evolução profissional e reconhecimento da importância da formação contínua das pessoas.

Não menos importante foi a atitude do professor Nuno Bettencourt que me alertou para esta iniciativa do Mestrado em Engenharia em Inteligência Artificial o que me despertou o interesse nesta formação.

Por fim, um agradecimento àqueles que passaram por mim e disseram que este trabalho era perda de tempo. Foi um exercício de contraditório que me ajudou a acreditar que, efetivamente, eu deveria seguir em frente neste mestrado e que a constante aprendizagem é o melhor caminho para uma vida melhor.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Trechos de Código

Acrónimos e Símbolos

Lista de Acrónimos

ANN	*Artificial Neural Networks*
AI	*Artifical Inteligence*
BDS	*Background dataset*
CNN	*Convolutional Neural Networks*
CSV	*Comma-separated values*
DARPA	*Defense Advanced Research Projects Agency*
DL	*Deep Learning*
DNN	*Deep Neural Network*
FCT	Fundação para a Ciência e a Tecnologia
GA	*Genetic Algorithms*
GECAD	Grupo de Investigação em Engenharia e Computação Inteligente para a Inovação e o Desenvolvimento
HTML	*HyperText Markup Language*
HCI	*Human Computer Interaction*
ICE	*Individual Conditional Expectation*
ISEP	Instituto Superior de Engenharia do Porto
I&D	Investigação e Desenvolvimento
KNN	*K-Nearest Neighbors*
LIME	*Local Interpretable Model-Agnostic Explanation*
LSTM	*Long Short-Term Memory*
MEIA	Mestrado em Engenharia de Inteligência Artificial
ML	*Machine Learning*
MSE	*Mean Squared Error*

MLPR	*Multilayer Perceptron Regression*
PCA	*Principal Components Analysis*
PDP	*Partial Dependence Plot*
PRECISE	*Power and Energy Cyber-Physical Solutions with Explainable Semantic Learning*
REN	Redes Energéticas Nacionais
RF	*Random Forest*
RFR	*Random Forest Regression*
RMSE	*Root Mean Square Error*
R^2	*R-Squared*
SHAP	*Shapley Additive Explanations*
SVM	*Support Vector Machines*
SVR	*Support Vector Regression*
UE	União Europeia
W	*Watts*
XAI	*Explainable Artifical Inteligence*

1 Introdução

1.1 Motivação

O setor energético está a atravessar por um período de profundo processo de mudança estrutural devido à transição energética que apresenta quatro dimensões: descarbonização, eficiência energética, descentralização e digitalização (Asif, M. et al. 2022). A digitalização está a revolucionar o setor energético, melhorando a produtividade, a segurança, a acessibilidade e a sustentabilidade geral dos sistemas de energia, sendo importante a contribuição das tecnologias baseadas em *Artificial Intelligence* (AI) (Asif, M. et al. 2022). Em (Ahmad et al. 2021) os autores referem que a rede elétrica convencional não foi projetada para gerir a integração de fontes de energia renováveis e que alterações nas características destas fontes (por exemplo, eólica, solar, geotérmica, hidrogênio) criam desafios para atender às cargas variáveis na rede elétrica. Estes autores e (Machlev et al., 2022) referem ainda o contributo que a AI, nomeadamente no campo do *Machine Learning* (ML), pode trazer para o setor energético em temas como a cibersegurança dos sistemas de energia, redes de energia, consumo de energia (Ramos et al., 2022) e outros.

A Comissão Europeia adotou um conjunto de propostas legislativas, enquadradas no *European Grean Deal* (EGD), com o objetivo de tornar as políticas da União Europeia (UE), em matéria de clima, energia, transportes e fiscalidade aptas para alcançar uma redução das emissões líquidas de gases com efeito de estufa de, pelo menos, 55 % até 2030, em comparação com os níveis de 1990 (European Commission 2019a). O EGD pretende ser um instrumento para a promoção da justiça, da prosperidade e de uma economia eficiente, competitiva e mais sustentável para a Europa e que inclui, por exemplo, a transição para as energias limpas (European Commission 2019b). Esta transição será feita através da concretização de diversas metas que incluem estabelecer um mercado de energia da UE plenamente integrado, interligado e digitalizado, promover tecnologias inovadoras e infraestruturas modernas, impulsionar a eficiência energética e a conceção ecológica dos produtos, entre outras.

Recentemente, a invasão da Ucrânia provocou grandes alterações em diferentes áreas e tornou evidente as vulnerabilidades enérgicas na Europa. De acordo com (Osička & Černoch, 2022) a necessidade da Europa reduzir a vulnerabilidade energética tenderá a pressionar a execução dos principais objetivos políticos no setor energético provavelmente à custa de um maior desenvolvimento do mercado integrado de energia da UE.

O documento *The role of Artificial Intelligence in the European Green Deal* (Gailhofer et al. 2021), de iniciativa do Parlamento Europeu através do *Special Committee on Artificial Intelligence in a Digital Age* (European Parliament and AIDA et al. 2021), expõe a importância do contributo da AI para a concretização das metas definidas no EGD. Esta contribuição abrange áreas como a energia, a agricultura, a habitação e a mobilidade, nas quais a AI pode ser utilizada para reduzir o consumo de energia e dos recursos naturais, apoiar a descarbonização e promover a economia circular. Na área da energia, a AI poderá ser utilizada no apoio aos consumidores no sentido de tornar mais eficiente o consumo de energia. Tal inclui a análise de comportamentos anómalos dos sistemas de energia que possam estar a desperdiçar recursos, a monitorização e otimização do consumo de energia nos edifícios, para além de outras aplicações, como exposto no referido documento.

A utilização de sistemas baseados em AI em diferentes áreas tem sido acompanhada por iniciativas no sentido de tornar estes sistemas mais confiáveis. A Comissão Europeia apresentou um conjunto de princípios (European Commission et al. 2020) para o desenvolvimento de sistemas baseados em AI **confiáveis**. Estes princípios são baseados no trabalho realizado pela *High-Level Expert Group on Artificial Intelligence* que enumera sete requisitos (AI HLEG et al. 2019). Um destes requisitos é a **transparência**, na qual se inclui a **explicabilidade** (do inglês *explicability*) que é considerada crucial para manter a confiança dos utilizadores nestes sistemas. Nos Estados Unidos, a confiança nos sistemas baseados em AI é um dos pilares do programa *National AI Initiative* (NAII Office et al., 2020). Este identifica como características de um sistema confiável a precisão, a **explicabilidade** e a **interpretabilidade** (do inglês *interpretability*), a privacidade, a robustez, entre outras.

Verifica-se, portanto, um contexto no qual é reconhecida a importância do contributo da AI para um consumo adequado de energia (European Parliament and AIDA et al. 2021). Em simultâneo, instituições governamentais, como a Comissão Européia (AI HLEG et al. 2019) e a Casa Branca, nos Estados Unidos, (NAII Office et al., 2020), entendem que é necessário desenvolver soluções que permitam tornar os sistemas baseados em AI confiáveis para utilizadores humanos. Estudo como o de (Páez et al., 2019) e (Atzmueller et al., 2019) argumentam que a explicabilidade contribui para o aumento da confiança, por parte dos seres humanos, em sistemas baseados em AI. No estudo (Machlev et al., 2022), os autores referem que os especialistas em sistemas de energia têm dificuldade em confiar nas decisões e recomendações dos modelos de ML. Estes mesmos autores notam que, em resposta a este desafio, tem vindo a ser desenvolvidas técnicas de explicabilidade para melhor compreensão dos modelos de ML.

O Grupo de Investigação em Engenharia e Computação Inteligente para a Inovação e o Desenvolvimento (GECAD)[1] é uma unidade de Investigação e Desenvolvimento (IED) sediada no Instituto Superior de Engenharia do Instituto Politécnico do Porto (ISEP-IPP). Dentre as diferentes investigações promovidas pelo GECAD, uma das suas linhas de investigação contempla métodos explicativos no âmbito do projeto PRECISE[2] - *Power and Energy Cyber-Physical Solutions with Explainable Semantic Learning* (referência PTDC/EEI-EEE/6277/2020). O projeto PRECISE é financiado pela UE, no âmbito do programa de desenvolvimento H2020 e pela Fundação para a Ciência e a Tecnologia (FCT), e propõe modelos para soluções automáticas que permitem uma gestão de energia eficiente em tempo real, reduzindo a fatura energética e respeitando as necessidades e preferências dos consumidores.

O PRECISE parte do estado atual da arte da AI e avança, concebe e implementa modelos e métodos que operam autonomamente em tempo real, de forma permanente, com múltiplas fontes de dados, informação e conhecimento e garantindo que os consumidores, gestores e peritos confiam nas suas decisões e ações. Para tal, propõe modelos de ML capazes de aprender ao longo do tempo (Ramos et al., 2022) e métodos explicativos que forneçam aos utilizadores explicações compreensíveis. Pretende-se, assim, que ao fornecer aos utilizadores explicações adequadas relativamente ao comportamento destes modelos e à sua aprendizagem automática, estes utilizadores ganhem confiança nos modelos que operam forma automática. A Figura 1 ilustra os principais elementos que compõem o PRECISE.

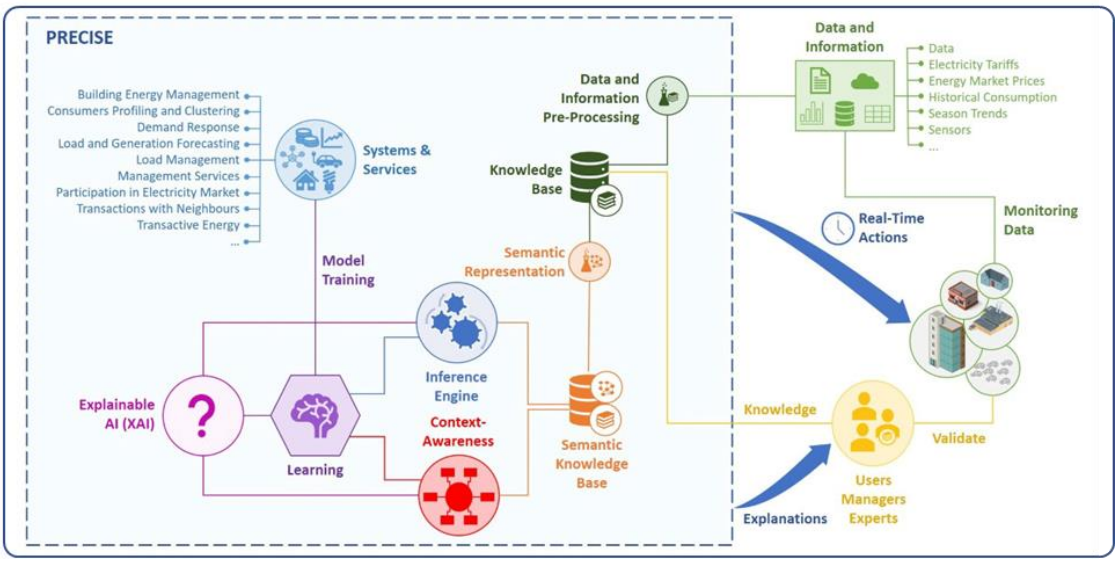


Figura 1 – PRECISE *Graphical Abstract*[3]

[1] GECAD *Website* – http://www.gecad.isep.ipp.pt/, último acesso em 15 de janeiro de 2022. [Online]
[2] PRECISE *Website* – https://www.gecad.isep.ipp.pt/precise/, último acesso em 11/10 de outubro de 2022. (Online)
[3] PRECISE *Graphical Abstract* – https://www.isep.ipp.pt/new/viewnew/6368, último acesso em 15 de janeiro de 2022. [Online]

Nesta representação gráfica dos componentes que compõem o projeto PRECISE, verifica-se que este é alimentado por dois canais, responsáveis por fornecer os dados, informações e conhecimento. O primeiro canal, *Data and Information Pre-Processing*, corresponde aos dados gerados por diferentes sistemas, físicos ou não, ligados ao consumo, produção e comercialização de energia nos quais se identificam: tarifas de eletricidade, preços de mercado de energia, histórico do consumo, sensores e outros. O segundo canal, *Users Managers Experts*, constituído por utilizadores, é responsável por alimentar o sistema com conhecimento e também suportar a validação do componente *Monitoring Data* o qual é alimentado, também, pelas respostas do sistema. As repostas dos modelos explicativos estão vocacionadas para o grupo do canal *Users Managers Experts*, ou seja, para os utilizadores do sistema.

No corpo do sistema, o componente *Learning*, apresenta uma posição central com ligações aos componentes *Inference Engine, Context-Awareness, Systems & Services e Explainable AI (XAI)*. O componente *Learning* é responsável pela geração dos modelos de ML. O componente *Explainable AI (XAI)*, responsável pelos métodos explicativos, além de estar diretamente ligado ao componente *Learning*, também se liga aos componentes *Inference Engine* e *Context-Awareness*. Assim, os métodos explicativos, responsáveis por gerar respostas para os utilizadores, não estão dependentes unicamente dos modelos de ML.

Uma das aplicações dos modelos de ML é a de realizar previsões de consumo de eletricidade para momentos futuros, como por exemplo, o consumo previsto para nos próximos 15 minutos. Para isto, estes modelos utilizam séries temporais com registos de consumos de energia ao longo do tempo. Estes registos podem ocorrer em diferentes intervalos, seja em minutos, por exemplo 5 minutos, em horas e outros. Como descrito no estado da arte, há poucos trabalhos de aplicação de *Explainable Artifical Inteligence* (XAI) em previsões baseadas em séries temporais, que utilizam modelos de ML aplicados em problemas de regressão, e em sistemas de energia. Portanto, é necessário o desenvolvimento de novas soluções que permitam ultrapassar esta limitação.

1.2 Objetivo

O problema a tratar nesta dissertação consiste no estudo de soluções que permitam obter explicações de previsões de consumos de eletricidade registados em conjuntos de dados do tipo série temporal, sendo que, estas previsões são realizadas por técnicas de ML. Este estudo tem como base o estado atual da arte dos métodos explicativos e pretende-se que seja uma contribuição para a integração de mecanismos de explicação em tarefas desenvolvidas no contexto do projeto PRECISE.

De forma a atingir este objetivo principal são definidos os seguintes objetivos específicos:

- Levantamento do estado da arte relacionado com XAI e modelos de previsão de consumo;
- Exploração métodos XAI existentes mais promissores para este problema;
- Experimentação com elaboração de casos de estudo aplicados a problemas reais;
- Conceção de uma proposta de solução para integração de XAI com métodos de ML.

Não é objetivo deste trabalho o estudo detalhado das técnicas de ML utilizadas em previsões de consumos de energia. As referências que possam existir limitam-se a um enquadramento, seja conceitual ou de implementação, para a compreensão do uso de mecanismos de explicação.

Este trabalho enquadra-se no âmbito da unidade curricular Projeto/Dissertação/Estágio (PROJIA), do Mestrado em Engenharia de Inteligência Artificial (MEIA) do Instituto Superior de Engenharia do Porto (ISEP).

1.3 Estrutura do Documento

O presente capítulo é uma introdução e contém a secção 1.1, Motivação, na qual se apresenta o contexto relativo aos mercados de energia considerando a AI e as motivações para o desenvolvimento de modelos explicativos. A secção 1.2, Objetivo, define o problema a tratar nesta dissertação.

O capítulo 2, Estado da Arte, é dedicado ao XAI e a sua relação com sistemas baseados em AI com destaque para o setor energético. No capítulo 3 é feita uma exposição dos dois métodos explicativos que serão analisados neste trabalho no contexto do projeto PRECISE. No capítulo 4, Metodologia para os Casos de Estudo, é exposta a metodologia utilizada neste estudo. O capítulo 5, Desenvolvimento e Implementação, apresenta a implementação da metodologia definida. O capítulo 6 é dedicado aos casos de estudo desenvolvidos para explicações de previsões de consumo de eletricidade considerando um conjunto de dados do tipo série temporal. No capítulo 7, Conclusões e Trabalho Futuro, são apresentadas as conclusões do estudo realizado assim como ideias e propostas de trabalho futuro a ser realizado no âmbito da tese de mestrado.

1.4 Terminologia

Neste documento, são utilizadas palavras e expressões em língua portuguesa as quais merecem destaque nesta fase introdutória.

- A expressão, aprendizagem máquina, refere-se ao conceito de ***Machine Learning*;**
- A expressão, **sistemas baseados em AI**, já referido, é utilizada em dois sentidos: expressar sistemas propriamente de AI, isto é, que contém apenas componentes ou

algoritmos de AI; outros sistemas que não são inteiramente de AI, mas que possuem um ou outro componente, ou mesmo algoritmo de AI;

- A expressão, **conjunto de dados**, será utilizada para expressar o conceito *dataset* que comumente refere uma coleção de exemplares de informação utilizada para o treino de um modelo de ML. Por exemplo, um ficheiro CSV com informação do consumo de eletricidade de um edifício em que cada registo deste é um exemplar da informação de consumo;
- Outra palavra frequentemente utilizada em ML é *feature*. Esta se refere a uma variável de entrada de um modelo de *M*. Mais uma vez considerando o exemplo de um CSV, uma *feature* é o nome da coluna do CSV. Neste documento será utilizada a palavra **atributo** para expressar o conceito de *feature*;
- A expressão, **importância do atributo**, será utilizada para referir *feature importance*;
- A palavra **explicabilidade,** no sentido de qualidade do que é explicável, será utilizada no lugar de *explicability*. A palavra **interpretabilidade**, no sentido de qualidade do que é interpretável, será utilizada no lugar de *interpretability*.

2 Estado da Arte

2.1 Introdução

Este capítulo é dedicado ao estado da arte atual do XAI. Destacam-se alguns dos seus conceitos e métodos explicativos e aborda-se, também, a importância das explicações em sistemas baseados em AI, a relação com ML, o problema dos modelos caixa-preta a interdisciplinaridade com outras áreas e uma breve discussão acerca da incorporação de explicações em sistemas de AI (secção 2.2).

É feita uma exposição da relação do XAI com a AI no campo do ML, tendo como foco o setor energético, assim os desafios e as tendências de XAI neste setor (secção 2.3). A avaliação da qualidade em XAI é outro tema apresentado (secção 2.4). Por fim, são apresentadas algumas considerações finais (secção 2.5) resumindo o conteúdo abordado ao longo das diferentes secções.

2.2 *Explainable Artifical Inteligence*

O artigo de (van Lent et al., 2004) é identificado por (Adadi and Berrada, 2018) e (Carvalho et al., 2019) como sendo aquele que, pela primeira vez, referiu o termo *Explainable Artificial Intelligence* (XAI). Este termo foi utilizado no referido artigo para descrever a capacidade de explicar o comportamento de um sistema baseado em AI utilizado de um jogo de simulação.

Em (Schoenborn et al., 2019), os autores definem XAI da seguinte forma: "*An explainable artificial intelligence enables an user to learn a transparent, relevant and justified information at the right time using an appropriate size*". De acordo com (Adadi and Berrada, 2018), não existe uma definição consensual para XAI. Segundo estes autores, XAI tende a se referir ao movimento, iniciativas e esforços feitos em resposta às preocupações de transparência e confiança na AI, mais do que a um conceito técnico formal.

Uma das entidades de grande relevância e que promoveu, desde 2015, o desenvolvimento de investigação em XAI (Adadi and Berrada, 2018), (Carvalho et al., 2019) é a *Defense Advanced Research Projects Agency* (DARPA)[4] e que em 2017 iniciou um programa de investigação dedicado ao tema XAI (Gunning et al. 2021). No entendimento da DARPA, o desenvolvimento de XAI é essencial para que os utilizadores entendam e confiem em sistemas baseados em AI, sendo que o interesse da DARPA é nos sistemas que utilizam ML. A DARPA não apresenta uma definição de XAI, mas enumera os objetivos pretendidos para o seu programa de pesquisa[5]: "*The Explainable AI (XAI) program aims to create a suite of machine learning techniques that: Produce more explainable models, while maintaining a high level of learning performance (prediction accuracy); Enable human users to understand, appropriately trust, and effectively manage the emerging generation of artificially intelligent partners*."

Em 2021, o programa de pesquisa da DARPA chegou ao fim. O documento (Gunning et al. 2021), destaca as principais pesquisas realizadas e conclusões. Algumas considerações finais descritas neste documento refletem o entendimento da DARPA quanto ao estado atual das pesquisas para a integração de mecanismos de explicação em sistemas baseados em AI: a) não há uma solução universal de XAI, para além de que diferentes utilizadores necessitam de diferentes tipos de explicações; b) a medição da eficácia das explicações é um dos maiores desafios; c) é necessária uma estreita colaboração em várias disciplinas, incluindo ciência da computação, ML, AI, fatores humanos e psicologia, entre outros, a fim de desenvolver técnicas de XAI que tenham eficácia.

2.2.1 Renovado Interesse

Uma pesquisa realizada no site *Web of Science*[6] permite ter uma noção do interesse dos pesquisadores em XAI. A pesquisa foi realizada em 8 de janeiro de 2022 e utilizou os seguintes termos de pesquisa: "*Explainable Artificial Intelligence*", "*XAI*", "*Explainable"* e "*Artificial Intelligence"*. Para a pesquisa destes termos, foram utilizados os operadores "OR" e "AND" da seguinte forma: *TOPIC: ("Explainable Artificial Intelligence" OR "XAI") OR TOPIC: ("Explainable" AND "Artificial Intelligence")*. Foi considerado o período temporal entre o ano de 1950 até o ano de 2022. Foi utilizado o campo de pesquisa *Topic*[7], disponível no formulário de pesquisa. Este campo permite que a pesquisa dos termos referidos seja realizada nas seguintes secções de um documento: título, resumo, palavras-chave. A Figura 2 ilustra a aplicação dos critérios de pesquisa.

[4] DARPA *Website* – https://www.darpa.mil/, último acesso dia 1 de dezembro de 2021, online.
[5] DARPA *Website Explainable Artificial Intelligence (XAI)* – http://www.darpa.mil/program/explainable-artificial-intelligence, último acesso dia 1 de dezembro de 2021, online.
[6] *Web of Science Website* – http://www.webofknowledge.com/, último acesso em 8 de janeiro de 2022. [Online]
[7] *Topic* – http://images.webofknowledge.com/WOKRS535R111/help/WOS/hs_topic.html, último acesso em 8 de janeiro de 2022. [Online]

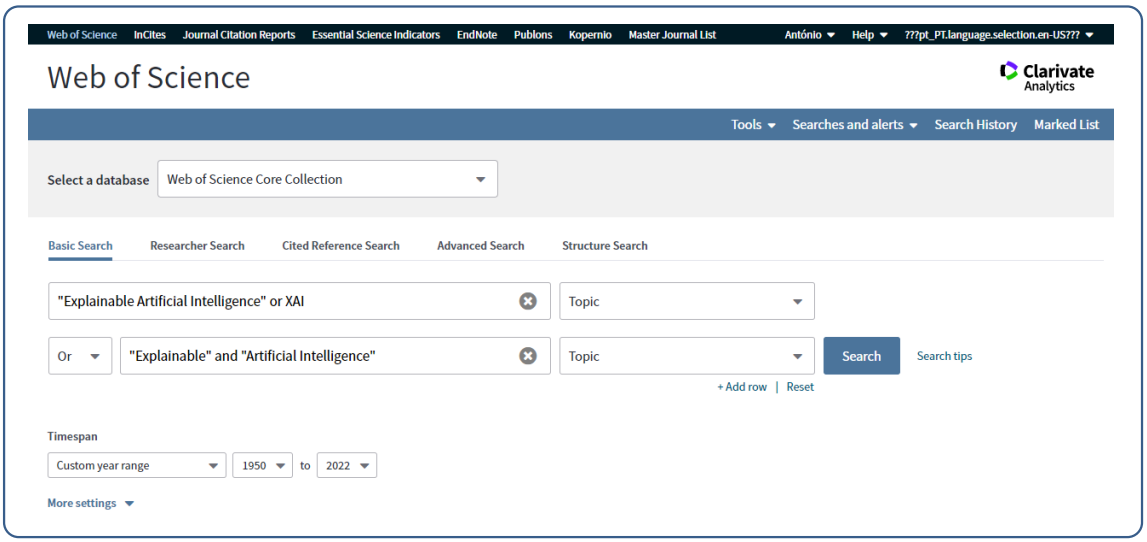


Figura 2 – Critérios de pesquisa das publicações em XAI

O resultado da pesquisa foi um total de 1259 registos. A seguir são apresentadas algumas análises que se consegue obter das ferramentas disponibilizadas pelo *Web of Science*. A Figura 3 apresenta um gráfico com a evolução da quantidade de publicações ao longo dos anos e a Figura 4 contém os valores relativos ao período entre 2016 e 2022.

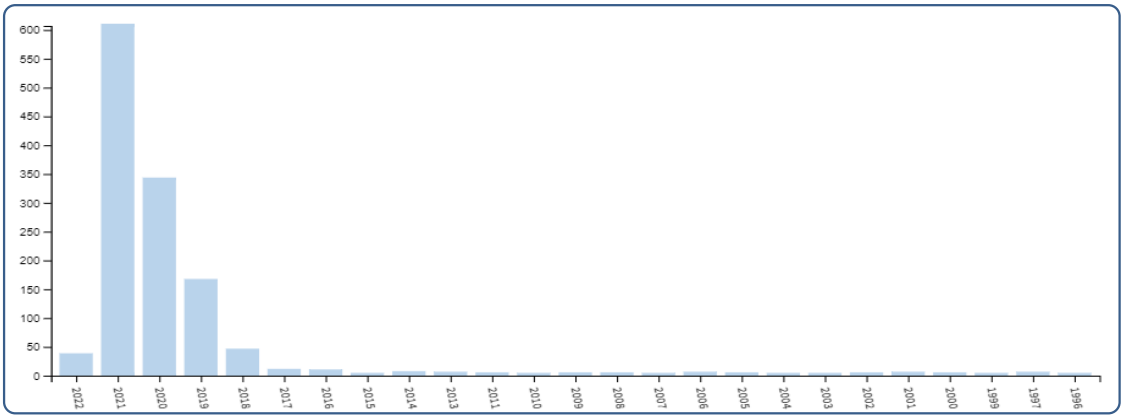
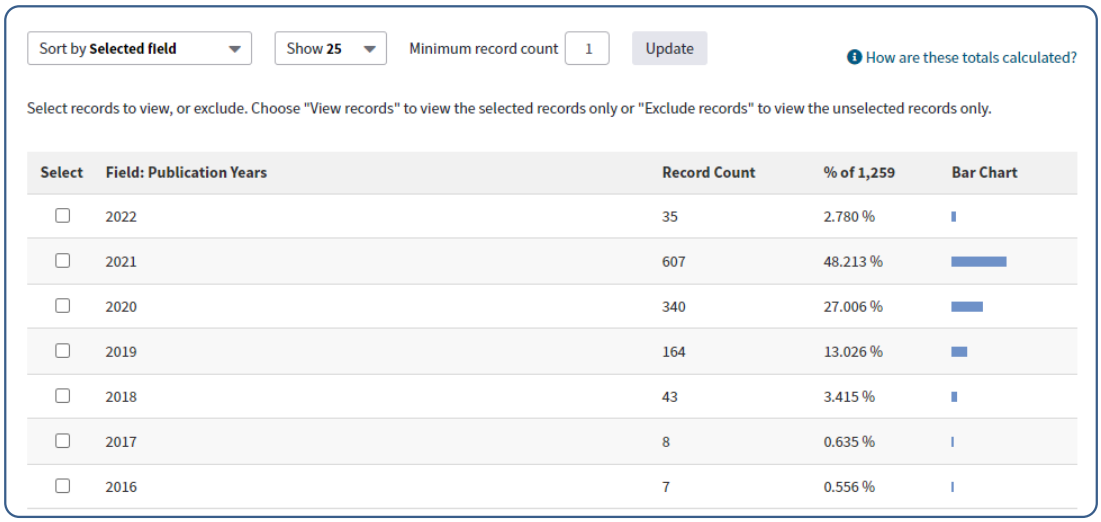


Figura 3 – Evolução da quantidade de publicações em XAI



Sort by **Selected field** | Show **25** | Minimum record count 1 | Update | How are these totals calculated?

Select records to view, or exclude. Choose "View records" to view the selected records only or "Exclude records" to view the unselected records only.

Select	Field: Publication Years	Record Count	% of 1,259	Bar Chart
☐	2022	35	2.780 %	
☐	2021	607	48.213 %	
☐	2020	340	27.006 %	
☐	2019	164	13.026 %	
☐	2018	43	3.415 %	
☐	2017	8	0.635 %	
☐	2016	7	0.556 %	

Figura 4 – Evolução da quantidade de publicações em XAI desde 2016

Verifica-se na Figura 3, que a partir de 2018 há um claro aumento de interesse da comunidade científica pelo tema XAI. Na Figura 4 nota-se que no início de 2022 já existiam quase tantos registos como em 2018 para todo o ano.

A seguir são apresentadas duas perspetivas da pesquisa realizada. A Figura 5 apresenta os tipos de documentos que abordam o tema XAI e a Figura 6 apresenta as áreas de pesquisa nas quais o tema XAI tem despertado interesse.

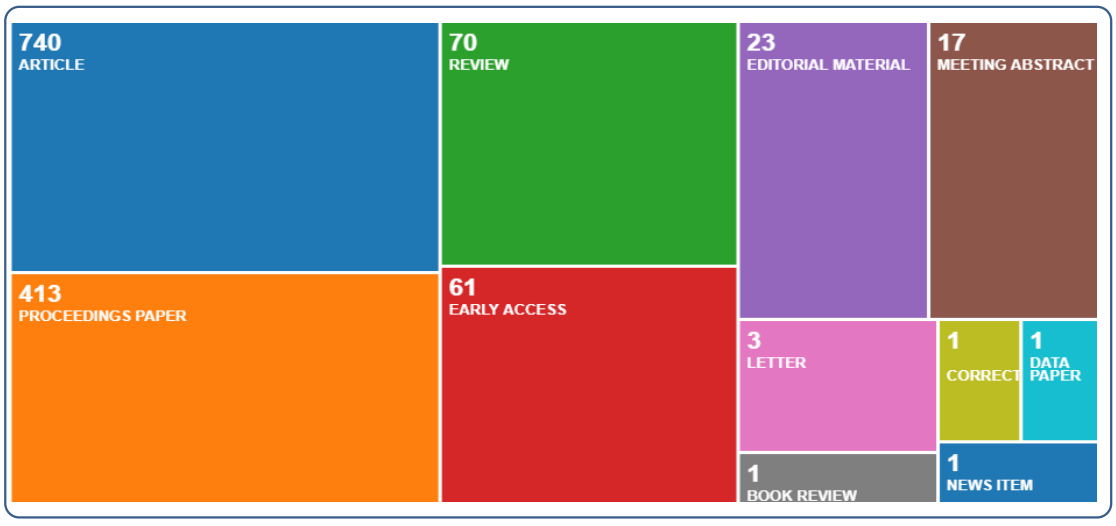


Figura 5 – Tipos de documentos que referem XAI

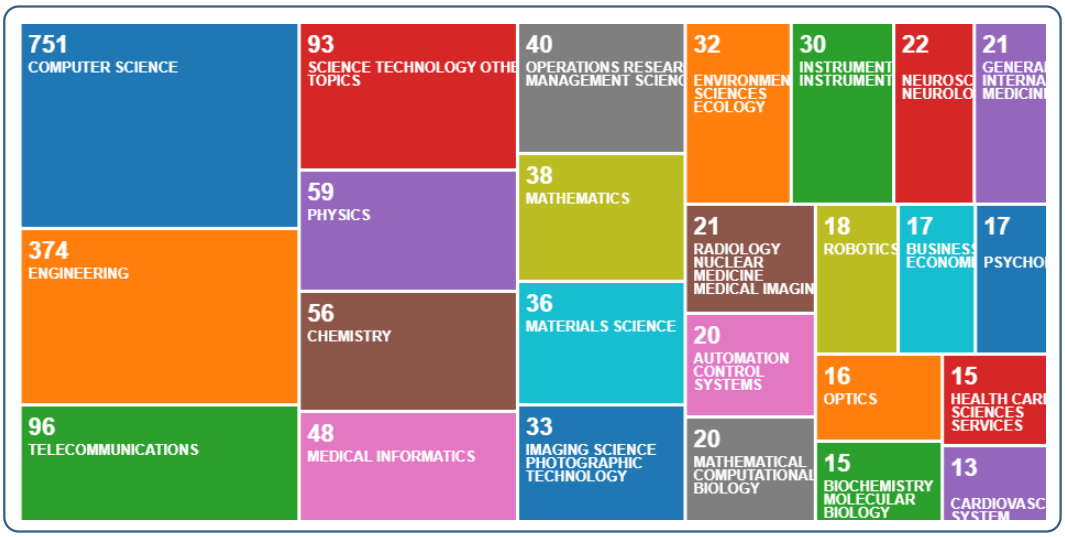


Figura 6 – Principais áreas de pesquisa com interesse em XAI

Na Figura 5, é notório o destaque da quantidade de documentos do tipo *article* publicados acerca da XAI. Na Figura 6, é interessante notar que esta pesquisa não identifica o tema da XAI no setor energético. Isto demonstra um grande potencial na sua utilização, pois ainda está muito pouco explorado.

2.2.2 Conceitos

Para (Ribeiro et al., 2016) o conceito de **confiança** em um sistema baseado em AI, e mais especificamente com um modelo de ML, decompõe-se em dois níveis: a) **confiança na previsão**, i.e., se o utilizador confia o suficiente em uma previsão individual para realizar ações baseadas nesta; b) **confiança no modelo**, i.e., se o utilizador confia que o modelo de ML apresenta comportamento aceitável. Nesta perspetiva do conceito de confiança, é possível que um modelo apresente um comportamento que confere confiança ao utilizador, para a generalidade das previsões, e no entanto, este mesmo modelo pode apresentar uma previsão específica a qual pode não ser confiável para o utilizador necessitando, assim, de melhor compreensão de como o modelo gerou determinada previsão.

Na secção 1.1 é referida a noção de **transparência** como um dos requisitos para tornar confiáveis aos utilizadores humanos os sistemas baseados em AI. De uma forma mais genérica, (Adadi and Berrada, 2018) entendem que transparência consiste em compreender a forma como é realizada a tomada de decisão. Com um âmbito mais restrito, em (Barredo Arrieta et al., 2020) este conceito refere-se à capacidade de compreensão do mecanismo de funcionamento de um modelo de ML.

Em (Barredo Arrieta et al., 2020) argumenta-se que **interpretabilidade** possui uma característica passiva enquanto **explicabilidade** possui uma característica ativa. Para estes autores, o primeiro expressa a transparência do modelo e o segundo refere-se a qualquer ação ou procedimento realizado pelo modelo com a intenção de esclarecer ou detalhar suas funções

internas. O estudo (Adadi and Berrada, 2018) apresenta estes termos como sendo fortemente relacionados de tal forma que "*interpretable systems are explainable if their operations can be understood by human.*".

Em (Linardatos et al., 2021) os autores referem que não existem definições matemáticas nem métricas para estes termos, o que permitiria clarificá-los de uma forma rigorosa. Segundo estes autores, uma definição que reúne maior consenso diz que interpretabilidade é "*the degree to which a human can understand the cause of a decision*". Para estes autores, o termo explicabilidade, por outro lado, está associado à lógica e à mecânica internas de um modelo de ML. Quanto mais explicável for o modelo, mais profundo será o entendimento que os humanos alcançam em termos dos procedimentos internos que ocorrem quer na fase de treino quer na fase de utilização do modelo treinado. Neste trabalho, os termos **interpretabilidade** e **explicabilidade** serão utilizados, muitas vezes, como sinónimos.

Outro conceito muito comum na literatura dedicada ao XAI é o de **modelo interpretável** (Molnar, C. 2022). Este é utilizado quando se pretende referir aos modelos reconhecidos por apresentarem maior grau de interpretabilidade, como a regressão linear, árvore de decisão, regras de decisão, em contraposição aos modelos classificados como caixa-preta (Molnar, C. 2022).

A seguir apresentam-se conceitos de XAI de cariz mais técnico, no sentido em que são atribuídos aos diferentes métodos de explicação existentes. A Figura 7, do trabalho de (Adadi and Berrada, 2018), apresenta alguns destes conceitos e que importa destacar, pois serão utilizados neste estudo.

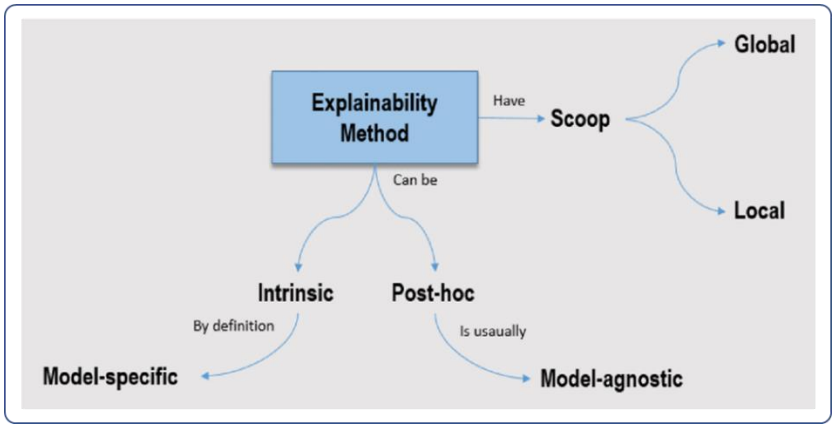


Figura 7 – *A pseudo ontology of XAI methods taxonomy* (Adadi and Berrada, 2018)

Os termos ***Local*** e ***Global***, para os quais neste estudo serão utilizados **interpretabilidade global** (do inglês *global interpretability*) e **interpretabilidade local** (do inglês *local interpretability*), respetivamente, referem-se à forma de classificar os métodos de explicação quanto à parte do processo de previsão de um modelo de ML no qual estes modelos de explicação serão aplicados.

A classificação interpretabilidade local refere-se à interpretação ao nível da previsão individual, também denominada de instância ou de observação, de um conjunto de dados e realizada pelo

modelo de ML. Está associado ao facto de a interpretabilidade ocorrer localmente e, portanto, relacionada às explicações das razões para uma decisão ou previsão específica do modelo de ML (Adadi and Berrada, 2018) e (Murdoch et al. 2019). O conceito de interpretabilidade local está relacionado com a ideia de confiança na previsão (Ribeiro et al., 2016). A classificação interpretabilidade global, também chamada de interpretação ao nível do conjunto de dados, indica que o método de explicação contempla todo o comportamento do modelo de ML, tendo como foco as relações entre os diferentes atributos do conjunto de dados, aprendidas pelo modelo de ML (Adadi and Berrada, 2018). O conceito de interpretabilidade global está relacionado com a ideia de confiança no modelo (Ribeiro et al., 2016).

O termo **modelo intrínseco** (do inglês *intrinsic model*) refere-se aos modelos de ML interpretáveis por si próprios (Adadi and Berrada, 2018) e (Barredo Arrieta et al., 2020). O termo de ***post-hoc*** é utilizado para os métodos explicativos que são aplicados ao modelo de ML, após o treino deste (Adadi and Berrada, 2018) e (Barredo Arrieta et al., 2020).

Os termos **modelo agnóstico** (do inglês *model-agnostic*) e **modelo específico** (do inglês *model-specific*) referem-se ao nível de generalização ou especialização de um método explicativo. Os métodos agnósticos não estão vinculados a um tipo específico de modelo de ML e são, portanto, independentes dos mesmos (Adadi and Berrada, 2018) e (Carvalho et al., 2019). Os métodos específicos são especializados, i.e., dedicados a um determinado modelo de ML e, portanto, não podem ser aplicados a qualquer modelo.

2.2.3 Explicações nos Sistemas de *Artifical Inteligence*

Incorporar, nos sistemas baseados em AI, a capacidade de explicação, isto é, extrair destes sistemas explicações acerca do seu funcionamento e dos seus resultados, é uma preocupação já presente no desenvolvimento destes sistemas desde os anos 70.

O sistema pericial MYCIN (Scott et al. 1977), utilizado na área da saúde para apoiar os médicos no tratamento dos pacientes, continha mecanismos de explicação. Nos anos 90, o sistema pericial SPARSE (Vale et al. 1999), utilizado em sistemas de energia, disponibilizava mecanismos de explicação para apoio aos operadores que interagiam com o sistema. Neste sistema, os mecanismos de explicações eram ainda utilizados como suporte ao componente de tutor que era utilizado em sessões de treinamento dos operadores.

As principais motivações para a integração de mecanismos de explicação nestes primeiros sistemas consistiam em: a) promover a credibilidade do sistema (Scott et al. 1977); b) permitir a formação dos utilizadores como se o sistema fosse um tutor (Scott et al. 1977) e (Vale et al. 1999); c) ao nível do desenvolvimento do software, seja para a manutenção ou para a evolução do software. Por exemplo, no apoio das equipas de desenvolvimento a verificar se o software executa as operações como o pretendido (Scott et al. 1977) e (Kass & Finin, 1988); d) justificar as recomendações (Kass & Finin, 1988) e (Vale et al. 1999);

Pesquisas com a utilização de ML têm sido conduzidas em diferentes áreas como em sistemas de energia (Ramos et al., 2022), (Ramos et al., 2020), (Donti and Kolter 2021) e (Pinto et al., 2016), na saúde (Kwekha-Rashid et al., 2021), recursos humanos (Garg et al. 2021), cibersegurança (Sarker 2021), dentre outras. Em ML, as arquiteturas do tipo *Deep Learning* (DL), como as *Deep Neural Network*s (DNN), têm sido em aplicações em áreas muito diversas, que incluem jogos, saúde, visão por computador e outras (Vilone & Longo, 2020).

O ML está a conquistar espaço nas mais variadas atividades da sociedade, o que vai de encontro às preocupações de instituições como a UE e o governo dos Estados Unidos no sentido de dotar os sistemas baseados em AI da capacidade de explicação no sentido de torná-los mais confiáveis.

2.2.4 *Explainable Artifical Inteligence* e *Machine Learning*

Dentre os objetivos do programa da DARPA, está a preocupação em tornar os modelos de ML explicáveis mantendo o alto nível de performance destes (secção 2.2). A Figura 8, obtida do documento (Gunning & Aha, 2019), ilustra a relação entre performance dos modelos de ML e explicabilidade.

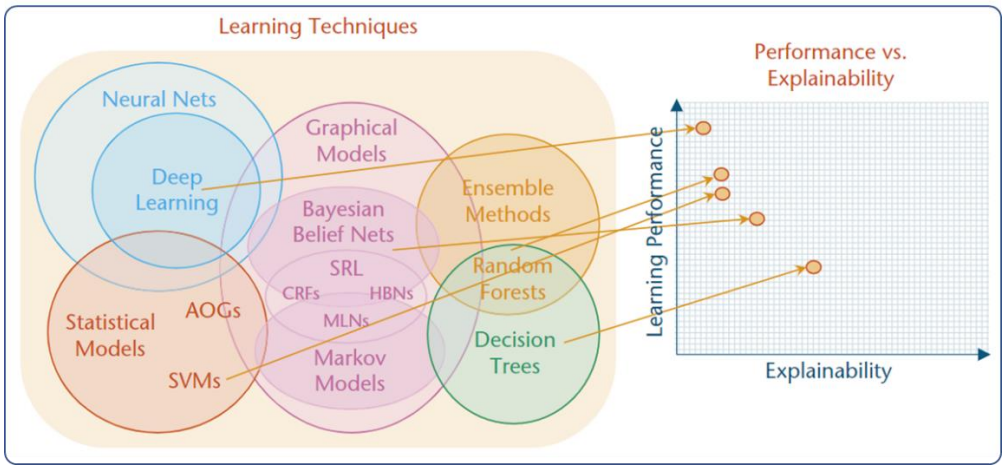


Figura 8 – Performance aprendizagem versus explicações

Como se pode verificar na Figura 8, quanto maior a performance de um modelo de ML, mais difícil é a capacidade de extrair explicações do modelo. Por exemplo, os modelos de DL são considerados os que apresentam maior performance, porém apresentam maior dificuldade em se obter explicações. No extremo oposto estão as árvores de decisão (do inglês *decision trees*).

Existem diversas técnicas para responder ao desafio de extrair explicações de modelos de ML. Em (Adadi and Berrada, 2018), os autores apresentam quatro tipos de técnicas: a) *Visualization*, b) *Knowledge Extraction*, c) *Influence Methods* e d) *Example-based Explanation*. No grupo *Visualization*, as técnicas mais populares são:

- ***Surrogate models*:** consiste em ter um modelo de ML simples interpretável, como por um exemplo um modelo linear, o qual é utilizado para explicar um modelo de ML complexo (Adadi and Berrada, 2018);
- ***Partial Dependence Plot (PDP):*** trata-se de uma representação gráfica que ilustra o efeito marginal que um ou dois atributos têm no comportamento global de um modelo de ML (Adadi and Berrada, 2018) e (Molnar, C. 2022);
- ***Individual Conditional Expectation (ICE):*** nesta representação gráfica, é possível visualizar o efeito no valor da previsão da instância do conjunto de dados quando há alterações dos atributos que influenciam a instância em análise, permitindo uma visão do comportamento da previsão local (Molnar, C. 2022).

No grupo *Knowledge Extraction*, que consiste em extrair de forma compreensível o conhecimento adquirido por um modelo de ML, nomeadamente redes neuronais, as principais técnicas são:

- ***Rule Extraction*:** abordagem que procura uma descrição simbólica e compreensível do conhecimento aprendido pela rede neuronal durante seu treino, extraindo regras que aproximam o processo de previsão e classificação utilizando a entrada e a saída da rede neuronal (Adadi and Berrada, 2018);
- ***Model Distillation:*** resumidamente, consiste em transferir o conhecimento adquirido por um modelo grande para um único modelo pequeno (Adadi and Berrada, 2018) e (Hinton et al., 2015).

O terceiro grupo, *Influence Methods*, que permite estimar a importância ou a relevância de um atributo do conjunto de dados, existem três métodos muito utilizados:

- ***Sensitivity analysis*:** refere-se a forma como o resultado de um modelo de ML, como por exemplo uma rede neuronal, é influenciado pelos dados de entrada e/ou perturbações de peso destes parâmetros. Permite verificar se o comportamento do modelo e os resultados permanecem estáveis quando os dados são intencionalmente perturbados ou outras mudanças são simuladas nos dados (Adadi and Berrada, 2018);
- ***Layer-wise relevance propagation:*** em uma visão de alto nível, consiste em executar a função de previsão em retropropagação começando na camada de saída de uma rede neuronal e até a sua camada de entrada (Adadi and Berrada, 2018) e (Bach et al., 2015);
- ***Feature Importance:*** consiste em quantificar a contribuição de cada atributo de entrada do modelo de ML para as suas previsões (Adadi and Berrada, 2018).

Finalmente, no quarto grupo, *Example-based Explanation*, que seleciona instâncias particulares do conjunto de dados para explicar o comportamento dos modelos de ML, há duas técnicas consideradas promissoras:

- ***Prototypes and criticisms:*** consiste na seleção das instâncias mais representativas do conjunto de dados (Adadi and Berrada, 2018);
- ***Counterfactuals explanations:*** o princípio desta técnica é explicar uma previsão local através de possíveis alterações que teriam de ocorrer no conjunto de dados para se obter um resultado esperado (Adadi and Berrada, 2018) e (Wachter et al. 2018).

Na revisão sistemática (Vilone & Longo, 2020), é apresentado o resultado da revisão de um conjunto de artigos que exploram o tema do XAI. Muitas das técnicas identificadas já haviam sido referidas no estudo de (Adadi and Berrada, 2018), como por exemplo importância do atributo (do inglês *Feature Importance*). Estes autores **destacam** o formato de **explicação visual** como a melhor forma de comunicação das explicações. Contudo, fazem notar a **importância de explicações textuais** e, inclusive, referem a combinação destes dois formatos de explicação em vários estudos analisados. Por fim, esta revisão sistemática refere que estes métodos explicativos são muito utilizados em problemas de classificação e que existem muitas aplicações de XAI vocacionadas para tipos de dados em formato para texto e imagens.

Em (Kuzlu et al., 2020), os autores destacam o facto de **existirem poucos estudos** de aplicação de XAI **em séries temporais**. No que refere aos problemas de regressão, o estudo de (Letzgus et al., 2021) destaca que há **pouca utilização de XAI** em problemas de **regressão** e que, quando utilizado, são de técnicas de XAI desenhadas para problemas de classificação aplicadas em regressão.

Em (Linardatos et al., 2021) são enumerados e descritos diversos modelos explicativos e a aplicação destes em modelos de ML do tipo caixa-preta. Nesta categoria de modelos, os autores identificam dois grupos. O primeiro grupo, com cerca de 16 métodos explicativos identificados, inclui métodos do tipo específico e dedicados a modelos de DL, sendo que os mais recentes são de 2018. O segundo grupo, também com cerca de 16 métodos explicativos, é constituído por modelos do tipo agnóstico e, portanto, aplicáveis a qualquer modelo de ML do tipo caixa-preta, sendo que os mais recentes são de 2019.

Dentre os diferentes métodos explicativos existentes, o *Local Interpretable Model-Agnostic Explanation* (LIME) (Ribeiro et al., 2016) e o *Shapley Additive Explanations* (SHAP) (Lundberg & Lee 2017) são considerados os mais populares e proeminentes (Linardatos et al., 2021), (Slack et al. 2020) e (Molnar, C. 2022). Estes caracterizam-se por serem do tipo modelo agnóstico e suportarem interpretabilidade local. O SHAP disponibiliza, ainda, mecanismos de interpretabilidade global (Molnar, C. 2022). Esses métodos estimam a contribuição dos atributos individuais de um conjunto de dados para uma previsão individual gerando perturbações nos valores dos atributos e observando o efeito dessas perturbações no resultado do modelo caixa-preta.

2.2.5 *Machine Learning* e o Problema dos Modelos Caixa-preta

Este estudo é dedicado ao XAI. Contudo, neste documento são referidas expressões como *Machine Learning* e modelo de *Machine Learning*. Não será feita uma exposição aprofundada acerca destes conceitos, mas importa um breve esclarecimento destes para compreender como estes se relacionam com o XAI.

O ML é um campo de estudo da AI (Russell et al., 1995). O trabalho de Arthur Samuel (Samuel et al., 1959) é considerado o primeiro estudo bem-sucedido em ML, de acordo com (Stuart Russel et al., 2009). Segundo (Géron et al., 2017), é de Arthur Samuel a primeira definição de ML: "*[Machine Learning is the] field of study that gives computers the ability to learn without being explicitly programmed".* Na década de 70, a tese de doutoramento de Patrick H. Winston (Winston 1970) pôs o ML como uma área de investigação de maior relevância na AI (Shavlik, Jude 1990).

Esta capacidade de aprender, referida na definição acima, é concretizada a partir do uso de algoritmos computacionais. Estes algoritmos podem receber conjuntos de dados – como por exemplo ficheiros em formato CSV com registos de consumos de eletricidade – e produzem resultados baseados nos padrões identificados no conjunto de dados, sendo que aos algoritmos denominam-se algoritmos de ML e os seus resultados denominam-se de modelo de ML (Helm et. al., 2020).

O reconhecimento de padrões em conjuntos de dados confere a estes modelos a capacidade de aprendizagem. Isto permite treinar uma máquina para que, de forma autónoma, possa realizar previsões e classificações de conjuntos de dados recebidos após o treino (Helm et. al., 2020). Por exemplo, dada uma imagem de um sinal de trânsito, a máquina pode classificar se esta imagem se refere a um sinal de trânsito de sentido proibido. É neste ponto que residem as preocupações enunciadas no capítulo 1. Há modelos que são de difícil compreensão o que cria muitos obstáculos para um ser humano compreender como estes calcularam os resultados, i.e., as previsões ou classificações. Isto dificulta, por parte dos seres humanos, a aceitação destes modelos em muitas áreas, incluindo em sistemas energéticos, nomeadamente em processos de tomada de decisão.

No documento (AI HLEG et al. 2019), de iniciativa da *High-Level Expert Group on Artificial Intelligence* (secção 1.1), os autores alertam para o tema da relação da explicabilidade com o que eles denominam de *black-box AI*. Este tema levanta a preocupação de que algumas técnicas de ML, embora muito bem-sucedidas do ponto de vista do resultado das previsões, são muito opacas em termos de compreensão de como estas realizam os cálculos. Para estes, a noção de *black-box AI* refere-se aos cenários nos quais não é possível fazer um rastreio das razões de determinados cálculos, ou mesmo decisões.

Em (Guidotti et al. 2019) e (Linardatos et al., 2021), os autores classificam os modelos de ML em caixa-branca (do inglês *white-box*) e caixa-preta (do inglês *black-box*) em função da dificuldade em se conseguir extrair, ou produzir, explicações que permitam a um ser humano compreender o funcionamento destes assim como os seus resultados. De acordo com estes autores, os modelos classificados como caixa-branca são aqueles dos quais é mais fácil extrair, ou produzir, explicações, enquanto os modelos do tipo caixa-preta são os mais difíceis. Muitas das técnicas de ML, nomeadamente de DL, são classificadas de caixa-preta devido às dificuldades acima referidas (Vilone & Longo, 2020). Os autores (Das & Rad, 2020) entendem que um modelo de DL pode ser considerado do tipo caixa-preta se o conjunto de parâmetros e a sua arquitetura estiverem ocultas do utilizador final.

Nas visões acima apresentadas quanto ao conceito de modelo caixa-preta, está a ideia de ser difícil a um ser humano compreender o funcionamento do modelo, devido aos obstáculos que dificultam a perceção do funcionamento destes e a extração de informação de como estes obtêm os resultados, ou decisões. Estes obstáculos podem ser a complexidade da arquitetura ou, ainda, a quantidade de parâmetros. Dentre os diferentes modelos reconhecidos como sendo do tipo caixa-preta, são referidos em (Ribeiro et al., 2016), (Lundberg & Lee 2017), (Adadi and Berrada, 2018), (Gunning & Aha, 2019) e (Tschora et al., 2022), o DNN e *Long Short-Term Memory* (LSTM) e, ainda, os modelos denominados de *ensemble* nos quais se incluem o *Random Forest* (RF), os chamados *Boosted Tree* como o *XGBoost*.

2.2.6 Interdisciplinaridade com outras áreas de conhecimento

Segundo (Adadi and Berrada, 2018) para um modelo de ML ser explicável este tem de ser compreendido pelo ser humano. Isto implica o desafio de comunicar ao ser humano processos de cálculo complexos, o que pode levar a incorporar no desenvolvimento de mecanismos de explicação os conhecimentos do domínio da *Human Computer Interaction* (HCI). Em (Miller, 2019), o autor destaca que os desenvolvimentos de mecanismos de explicação não devem estar centrados apenas em questões técnicas, ou de algoritmos, mas também devem considerar como os seres humanos avaliam as explicações o que conduz à procura do contributo das ciências cognitivas. Outro aspeto importante, referido pelo mesmo autor, é que as explicações não devem ser vistas apenas como a apresentação de uma relação de causa e efeito, sendo importante considerar que estas devem ser contextualizadas.

2.2.7 Será que todos os sistemas baseados em AI têm de ter mecanismos de explicação?

De acordo com o que foi referido, tudo indica que a adoção de mecanismos de explicação é benéfica para quem utiliza sistemas baseados em AI, pois, dentre muitos pontos, permite compreender o funcionamento destes assim como os seus resultados, e ainda potencia o aumento na confiança nestes sistemas. Contudo, há quem argumente que a adoção destes mecanismos tem de ser devidamente avaliada. Em (Adadi and Berrada, 2018) os autores expõem o argumento de Norvig, o diretor de investigação da Google, segundo o qual a explicação é importante nestes sistemas, mas nem sempre é necessária: "*In fact, requiring every AI system to explain every decision could result in less efficient systems, forced design choices, and a bias towards explainable, but less capable and versatile outcomes*" (Adadi and Berrada, 2018). Outro aspeto a considerar é o custo, pois o desenvolvimento de sistemas de AI explicáveis ainda é muito caro.

Ainda em (Adadi and Berrada, 2018), os autores apresentam dois critérios para a avaliação da integração de explicações em sistemas de AI: a) "*The degree of functional opacity caused by the complexity of AI algorithms: if it is low, no high level of interpretability is required.*"; b) "*The degree of resistance of the application domain to errors. If it has high resistance, unexpected*

errors are acceptable.". Também (Doshi-Velez & Kim, 2017) já argumentavam no sentido de que nem todos os sistemas baseados em AI necessitam de mecanismos de explicação. Mas recentemente, (Carvalho et al., 2019) reforçam esta posição.

Assim, desde os sistemas periciais, passando pelas soluções de AI para problemas de otimização e, mais recentemente, pelas atuais soluções desenvolvidas com ML, a adoção de mecanismos de explicação em sistemas baseados em AI tem como motivações:

- Permitir que os seres humanos possam compreender o funcionamento dos sistemas;
- Permitir que os seres humanos possam compreender os resultados dos sistemas para uma adequada tomada de decisão;
- Aumentar a confiança nestes sistemas;
- Apoio às equipes de desenvolvimento na manutenção e evolução de sistemas baseados em AI.

Apesar de ser reconhecida a importância da introdução de mecanismos de explicação em sistemas baseados em AI, a sua inclusão em cada sistema deverá ser ponderada e considerar a relação custo benefício. Ainda, há estudos que argumentam que o desenvolvimento destes mecanismos deve ter em consideração outras áreas como a HCI e as ciências cognitivas.

2.3 *Artifical Inteligence* e *Explainable Artifical Inteligence* em Sistemas de Energia

O estudo do XAI no contexto dos sistemas de energia é uma das vertentes deste trabalho. Considerando a inerente ligação de XAI com AI, importa ter uma melhor perceção da utilização de AI nestes sistemas e a consequente aplicação das técnicas de XAI.

2.3.1 *Artifical Inteligence* em Sistemas de Energia

O artigo (Santos et al. 1999), de finais dos anos 90, aborda a utilização de AI para apoio à tomada de decisão dos operadores de um Centro de Controlo de uma rede de energia elétrica. Este artigo apresenta o SPARSE, um sistema baseado em conhecimento, desenvolvido para os Centros de Controlo da Rede de Transporte Portuguesa, propriedade e operado pela Rede Portuguesa de Transmissão de Energia Elétrica, atualmente denominada Redes Energéticas Nacionais (REN).

Também importante no setor energético é o tema da previsão do preço da eletricidade. Em (Pinto et al., 2012) é realizada uma investigação para a previsão do preço da eletricidade considerando o modelo ANN com caracter dinâmico. Este dinamismo permite, a que o modelo considere os dados mais recentes e que o tempo de execução deste se adapte aos diferentes contextos de utilização. Neste outro trabalho (Pinto et al., 2016), os autores abordam a utilização de *Support Vector Machines* (SVM) no apoio à tomada de decisão no contexto do mercado da eletricidade.

No artigo (Gonzalez-Briones et al., 2019), é referida a importância da tarefa de previsão do consumo de energia como forma de as empresas adaptaram a sua capacidade de fornecimento. Dentre os modelos de ML identificados pelos autores estão o SVM, o KNN e o *Random Forest* (RF). Em (Verwiebe et al. 2021), os autores analisaram artigos publicadas entre 2015 e 2020 que abordam o tema da modelação de demanda de energia. Este estudo, destaca a ANN como o modelo de ML mais utilizado.

No estudo (Ramos et al., 2022), os autores exploram os modelos de ML, K-Nearest Neighbors (KNN) e Artificial Neural Networks (ANN) para a previsão de consumo de energia de forma a contribuir para uma adequada gestão do consumo de energia em edifícios. Em (Tschora et al., 2022), os autores utilizam séries temporais para avaliar o uso de modelos de ML (SVM, *Random Forest Regressor* (RFR), *Convolutional Neural Networks* (CNN) e DNN) na previsão do preço da eletricidade. O artigo (Gasparin et al., 2022) aborda a utilização de DL em previsões utilizando séries temporais. De acordo com os autores, o uso de DL para previsões tem despertado interesse em sistemas de energia devido a elevada performance quanto aos resultados que esta técnica tem apresentado em outras tarefas como a classificação de imagens. Os autores apresentam, ainda, um resumo das técnicas de ML utilizadas em diferentes pesquisas que utilizaram séries temporais. Dentre estas técnicas destacam-se a LSTM e a CNN.

2.3.2 *Explainable Artifical Inteligence* em Sistemas de Energia

O sistema pericial SPARSE (Vale et al. 1999) já disponibilizava mecanismos de explicação para apoio aos operadores que utilizavam o sistema (2.2.3). Com a possibilidade da utilização de técnicas ML em sistemas de energia, levanta-se o problema da extração, ou produção, de explicações, visto ser reconhecida a dificuldade em se obter explicações de alguns modelos, nomeadamente das arquiteturas DL, conhecidas como caixa-preta.

Como apresentado na Figura 6 (secção 2.2) o setor energético ainda não é uma área de investigação com grande destaque no que refere ao XAI. O estudo apresentado por (Puig & Carmona, 2019) consistiu em implementar um sistema para detetar perdas de energia chamadas de Perdas Não Técnicas (do inglês *Non-Technical Loss*). Foi utilizado o LIME como abordagem inicial. O SHAP foi utilizado para avaliar a capacidade dos algoritmos de aprendizagem detetarem os atributos relevantes em um conjunto de dados utilizado no estudo do problema do Perdas Não Técnicas.

Em (Kuzlu et al., 2020), os autores referem que existem poucos estudos com a utilização de XAI em sistemas de energia, como por exemplo, na previsão de geração de energia. Neste artigo, foram utilizados o SHAP e o LIME com o intuito de identificar a importância dos atributos nas previsões geradas pelo modelo de ML *Random Forest* em um caso de regressão.

O SHAP é também utilizado em estudos da aplicação de métodos de explicação visual para consumos de energia, em (Wastensteiner et al., 2021). Este último estudo, consistiu em aplicar o modelo SHAP e em desenvolver, e avaliar, mecanismos de explicação utilizando explicações visuais disponibilizadas pela biblioteca do SHAP.

Em (Machlev et al., 2022) os autores referem o recente interesse no XAI na área de previsão de produção de energia baseada em fontes renováveis, com destaque para os painéis fotovoltaicos. Segundo estes autores, outro tema de interesse para aplicar XAI é a gestão de energia em edifícios. Este estudo destaca ainda o facto de o SHAP e o LIME serem as abordagens mais utilizadas em diferentes estudos relacionados com sistemas de energia. O SHAP é também utilizado no estudo de (Tschora et al., 2022) para obter explicações de previsões do preço da eletricidade com modelos de ML.

O LIME e o SHAP são os mais populares e proeminentes métodos de XAI (secção 2.2.4). Interessante notar que, apesar de o LIME ser de 2016 e o SHAP ser de 2017 (secção 2.2.4), existem pesquisas no contexto dos sistemas de energia de 2022 (Machlev et al., 2022) e (Tschora et al., 2022) que utilizam estes dois métodos.

2.3.2.1 Desafios e Tendências

Em (Machlev et al., 2022) o estudo apresenta desafios e oportunidades para a utilização de XAI no setor energético. No que refere aos desafios e limitações, o estudo apresentado está em linha com o resultado do programa da DARPA (secção 2.2), que incluem a necessidade de equilíbrio da elevada performance nos resultados dos modelos de ML e os métodos explicativos; a dificuldade em encontrar métricas de medição da qualidade das explicações; a dificuldade em encontrar uma definição padrão do conceito de explicabilidade; dentre outros desafios.

Além destes pontos comuns com a DARPA, o estudo de (Machlev et al., 2022) destacam os seguintes desafios: a) o facto que muitas das técnicas atuais disponibilizam ferramentas de explicação vocacionadas para especialistas em AI ao invés de estarem vocacionadas para especialistas em sistemas de energia; b) questões de segurança como ataques maliciosos aos sistemas que incorporam métodos de XAI e que podem gerar explicações incorretas.

Quanto às tendências, os autores destacam: a utilização de XAI para a previsão da produção e consumo de energia baseada em fontes renováveis utilizando, por exemplo painéis fotovoltaicos; a utilização em modelos de classificação no contexto da segurança de redes de energia e cenários de falhas e a gestão de energia em edifícios.

2.4 Avaliação da Qualidade das Explicações em *Explainable Artifical Inteligence*

Nas considerações finais do programa de pesquisa da DARPA (secção 2.2), a medição da qualidade das explicações é considerada como um dos maiores desafios do XAI. (Doshi-Velez & Kim, 2017) referem que avaliações com a participação de utilizadores humanos é uma tarefa difícil pois consome muito tempo seja no desenho e na implementação das avaliações, o que torna o processo demorado e caro. Estes autores apresentam três categorias de avaliação dos modelos explicativos: *Application-grounded evaluation*, *Human-grounded evaluation* e

Functionality-grounded. As duas primeiras requerem a colaboração de utilizadores humanos enquanto a terceira é baseada em definições formais sem utilizadores humanos e utilizam métricas quantitativas.

(Carvalho et al., 2019) sugere dois tipos de indicadores para medição e comparação das explicações: qualitativos e quantitativos. O último é uma avaliação numérica que, segundo os autores, fornecem uma forma intuitiva de comparação de diferentes explicações. Em (Zhou et al. 2021) os autores referem a importância das avaliações dos métodos explicativos para quantificar a qualidade destes de forma a determinar se, e em que medida, a explicação oferecida vai de encontro ao objetivo definido. Outros aspetos referidos pelos autores é a possibilidade de comparar os métodos de explicação disponíveis e sugerir a melhor explicação para uma tarefa específica. Contudo, (Carvalho et al., 2019) e (Zhou et al. 2021) verificam que não existe um consenso de como avaliar a qualidade de uma explicação. De acordo com (Zhou et al. 2021), uma das razões deve-se ao facto de a explicação ser um conceito inerentemente subjetivo, para além de que a qualidade percebida de uma explicação é contextual e dependente das pessoas que utilizam o sistema bem como do tipo de informação que os utilizadores estão interessados.

Neste estudo (Amparore et al., 2021) os autores propõem um software desenvolvido em *python* denominado LEAF. O objetivo do LEAF é fornecer uma referência para a avaliação das explicações de forma padronizada e imparcial e orientar os pesquisadores no desenvolvimento de técnicas explicáveis.

2.5 Considerações Finais

A utilização de AI em sistemas de energia não é recente, sendo que as técnicas de ML têm vindo a ganhar cada vez mais espaço. A preocupação em dotar os sistemas de energia que utilizam AI da capacidade de explicações, também não é recente. Um exemplo, é a utilização de sistemas periciais, em finais dos anos 90, para apoio aos utilizadores nos centros de controlo de uma rede de energia. A possibilidade da utilização de modelos de ML tem renovado o interesse no desenvolvimento dos mecanismos de explicação, nomeadamente devido à utilização de modelos do tipo caixa-preta, por ser reconhecida a dificuldade de se conseguir extrair, ou produzir, explicações dos modelos deste tipo.

O desenvolvimento de métodos explicativos tem despertado o interesse da comunidade científica pela área de XAI, que teve na DARPA uma das principais instituições impulsionadoras da investigação nesta área. Contudo, a utilização dos mecanismos de explicação em sistemas baseados em AI é importante mas nem sempre necessário, sendo que a utilização destes mecanismos deve ser uma decisão ponderada considerando questões como a relação custo benefício. Verifica-se, ainda, que não há uma definição consensual acerca do conceito de XAI, inclusive quanto a alguns dos seus conceitos. Dentre os métodos explicativos existentes, o LIME e o SHAP são os mais relevantes e utilizados nestas investigações.

Ainda é pouco expressiva a investigação do uso de XAI na área de sistemas de energia, seja para consumo ou produção de energia o que evidencia um potencial de crescimento nesta área. As séries temporais são muito comuns no setor energético, seja para registo de consumos ou de produção de energia. Contudo, há pouca utilização de XAI com modelos de ML que utilizam séries temporais, assim como em tarefas de regressão.

Muitos dos desafios identificados no setor energético reforçam as conclusões do trabalho de pesquisa da DARPA. Quanto às tendências, destacam-se a previsão da geração e demanda de energia renovável utilizando, por exemplo painéis fotovoltaicos e a gestão de energia em edifícios.

Apesar do surgimento de métodos explicativos, não existe, ainda, um conjunto de métricas padrão para a avaliação da qualidade das explicações. Diferentes abordagens têm vindo a ser utilizadas pelos investigadores, porém ainda não há um consenso de quais as abordagens mais adequadas para cada caso. Uma das principais dificuldades é o facto de o próprio conceito de explicação ser um inerentemente muito subjetivo e, portanto, de difícil medição.

Dentro do que foi apresentado no estado da arte, no capítulo seguinte é apresentada uma descrição dos métodos explicativos LIME e SHAP com destaque para os principais conceitos e aspetos das bibliotecas que implementam estes métodos, as quais serão necessárias na aplicação deste trabalho no contexto do projeto PRECISE.

3 Métodos Explicativos LIME e SHAP

3.1 Introdução

Este capítulo apresenta as descrições conceituais do LIME (secção 3.3) e do SHAP (secção 3.5) assim como alguns aspetos das respetivas bibliotecas que implementam estes conceitos. Um aspeto a ter em atenção quanto aos conceitos aqui apresentados, é que não serão considerados todos os conceitos inerentes a estes métodos, mas apenas aqueles que serão necessários na aplicação deste trabalho no contexto do projeto PRECISE.

Serão apresentados os conceitos de regressão linear (secção 3.3) e de *shapley value* (secção 3.4) por estarem na base do LIME e do SHAP. No fim deste capítulo é apresentado um resumo (secção 3.6) do conteúdo exposto.

3.2 Modelos Interpretáveis – Regressão Linear

A denominação modelo interpretável, relacionada com o conceito de interpretabilidade (secção 2.2.2) refere-se aos modelos que apresentam maior grau de interpretabilidade em contraposição aos denominados modelos caixa-preta.

A regressão linear, cujo conceito foi introduzida por Sir Francis Galton (1885) (Ethington et al., 2002) e que constitui o chamado modelo de regressão linear, é um tipo de modelo interpretável (Molnar, C. 2022). A aplicação da regressão linear consiste em encontrar a reta, que se denomina de reta de regressão, que mais se aproxima dos pontos distribuídos em um plano bidimensional (Online Statistics Education: A Free Resource for Introductory Statistics, 2022). A Tabela 1 representa uma relação na qual para um dado valor de x corresponde um valor de y, sendo que para $x = 6$ é desconhecido o valor de y.

Tabela 1 – Regressão linear: Relação entre valores do eixo X e do eixo Y[8].

x	y
1	1
2	2
3	1.3
4	3.75
5	2.25
6	?

A reta de regressão tem uma representação matemática que permite prever outros valores de y para um dado valor de x, como por exemplo para $x = 6$. Na Figura 9 é possível observar a distribuição dos pontos em um plano de coordenadas X e Y (a) e a reta de regressão resultante deste caso (b).

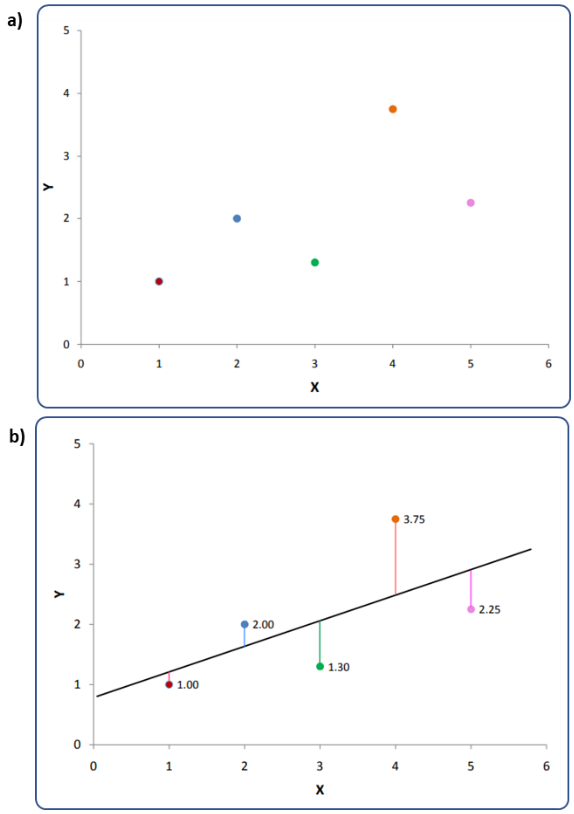


Figura 9 – Regressão Linear (a) distribuição dos pontos; (b) reta de regressão[9]

Em (b), as linhas verticais com as cores vermelho, azul, verde, laranja e rosa representam o erro do ponto da reta de regressão em relação aos pontos reais. É esta a reta cuja representação formal melhor consegue prever o valor de y para um dado valor de x.

[8] (Online Statistics Education: A Free Resource for Introductory Statistics, 2022)
[9] (Online Statistics Education: A Free Resource for Introductory Statistics, 2022)

Portanto, a regressão linear permite prever o valor de uma variável y, denominada de variável dependente, a partir de uma ou mais variáveis independentes $x_1, x_2, \ldots, x_{n,}$ e é expressa pela seguinte fórmula:

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon \quad (1)$$

, onde (Montgomery & Peck, 2013),

- y é a variável dependente, ou seja, o que se pretende prever;
- $x_1, \ldots, x_p$ são as variáveis independentes;
- β_0 , denominado de intercepto (do inglês *intercept*). Define a altura da reta e corresponde ao ponto do eixo Y no qual a reta de regressão encontra este eixo.
- $\beta_1, \ldots, \beta_p$ são denominados de pesos ou de coeficientes. Representam o peso de uma variável x_i (denominada de variável independente) no valor previsto;
- ε, é denominado de erro, isto é, a diferença entre o valor previsto e o valor real.

Dentre os diferentes parâmetros, será dado destaque aos coeficientes. Estes representam o peso de uma variável independente x_i no valor previsto. Isto permite interpretar o quanto uma variável independente influencia o valor previsto e faz com que este modelo seja reconhecido na literatura como um modelo interpretável (Molnar, C. 2022).

Transpondo este conceito para o ML, e supondo um conjunto de dados em formato CSV em que cada coluna é um atributo e as linhas são os possíveis valores. Um destes atributos será aquele que se pretende prever, i.e., a variável dependente y e os restantes são as variáveis independentes $x_1, \ldots, x_p$. A utilização de um modelo de ML, baseado em regressão linear, permite calcular a reta de regressão para prever os valores da variável independente do conjunto de dados. Os coeficientes, ou pesos, ajudam a interpretar o quanto cada atributo do conjunto de dados contribuiu para o valor previsto. A reta de regressão é comumente chamada de fronteira de decisão (Géron et al., 2017).

3.3 Local Interpretable Model-Agnostic Explanation

O Local Interpretable Model-Agnostic Explanation foi proposto em (Ribeiro et al., 2016). Os autores do artigo afirmam que: "*LIME, an algorithm that can explain the predictions of any classifier, by approximating it locally with an interpretable model*". Portanto, o LIME é um método que se propõe explicar previsões e que se caracteriza por ser agnóstico, i.e., para qualquer classificador e com interpretabilidade local.

O conceito de interpretabilidade local refere-se à interpretação de uma previsão individual, ou instância de um conjunto de dados, realizada pelo modelo de ML (secção 2.2.2). Esta instância é o conceito de instância de interesse referido na secção 2.2.4. Identificada a instância de interesse, o processo de execução do LIME pode ser assim explicado (Molnar, C. 2022):

- Geração de observações aleatórias entorno da instância de interesse. Este processo é chamado de perturbação, ou seja, introdução de variações no conjunto de dados utilizado pelo modelo caixa-preta. Portanto, é criado um novo conjunto de dados considerando o ponto de interesse e mais todos os novos pontos gerados;
- Treino de um modelo interpretável com o conjunto de dados do ponto anterior;
- Geração de previsões, relativa às observações aleatórias, entorno da instância de interesse. Estas previsões são ponderadas em função da distância destas à instância de interesse;
- Os pontos mais próximos da instância de interesse apresentam maior peso. O modelo que mais se aproxima da instância de interesse e que melhor se ajusta aos pontos previstos gerados, será considerado como o melhor modelo possível.

Como exemplo, considera-se um conjunto de dados em formato tabular com os atributos temperatura, humidade e pressão e para o qual um dado modelo de ML realizou previsões. A Figura 10 ilustra o resultado de um modelo de ML.

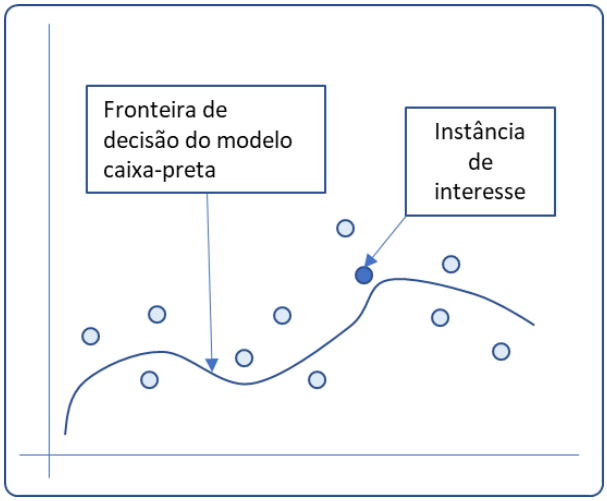


Figura 10 – Previsão de modelo caixa-preta

Neste exemplo, o modelo gerou uma fronteira de decisão não linear que melhor se ajusta ao conjunto de pontos do conjunto de dados. Em destaque a instância de interesse que se pretende explicar.

O conjunto de dados tabular utilizado pelo modelo de ML é representado na Figura 11. A Figura 12 representa o novo conjunto de dados criado a partir da geração de valores aleatórios em torno da instância de interesse.

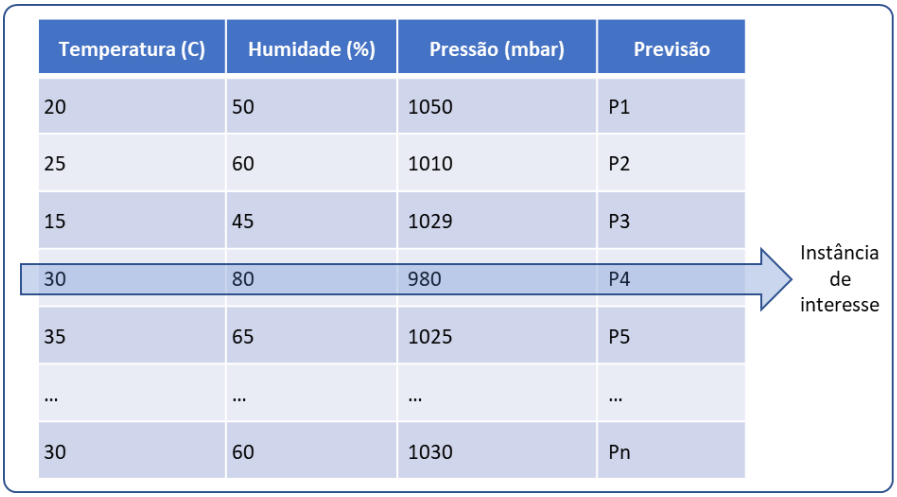


Figura 11 – Conjunto de dados e instância de interesse

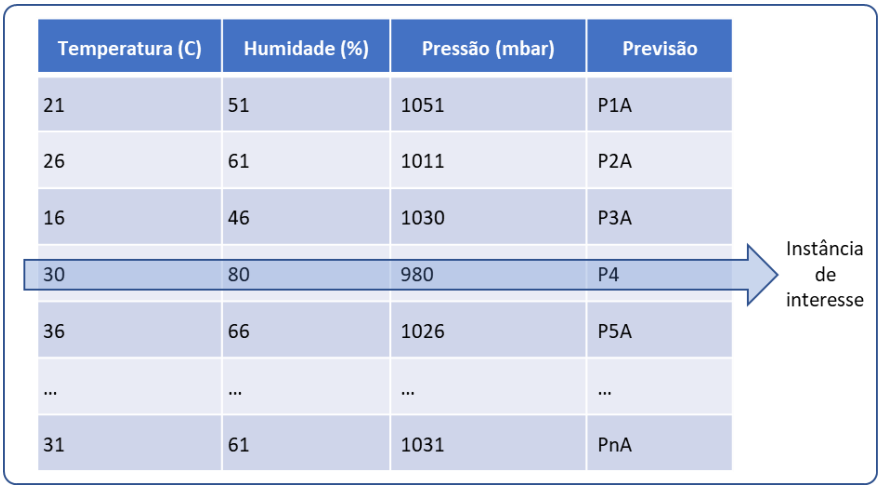


Figura 12 – Observações aleatórias e instância de interesse

A instância de interesse identificada apresenta os seguintes valores: temperatura = 30, humidade 80, pressão = 980 e previsão P4. A Figura 11 representa o conjunto de dados utilizado pelo modelo. A Figura 12 apresenta os novos valores obtidos, adicionando uma unidade aos valores originais do conjunto de dados da Figura 11, em torno da instância de interesse e respetivas previsões. A partir destes novos pontos do conjunto de dados da Figura 12, são realizadas novas previsões ilustradas na Figura 13, seguinte, e representadas por quadrados.

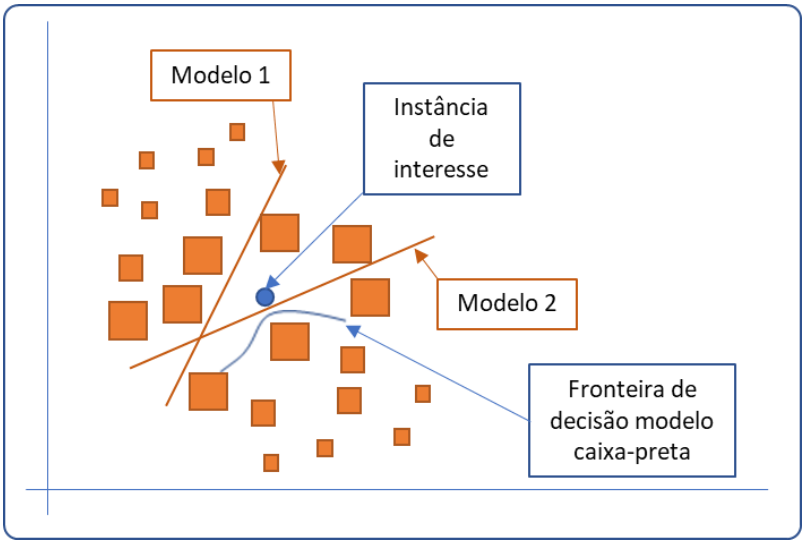


Figura 13 – Cálculo do melhor modelo interpretável

Os quadrados mais próximos da instância de interesse possuem um tamanho maior, representando o maior peso, ou seja, a ponderação realizada pelo modelo interpretável. As retas, modelo 1 e modelo 2, representam os modelos gerados pelo LIME. O modelo 2 é o que está mais próximo da instância de interesse e que apresenta o menor erro em relação aos novos pontos gerados. Portanto, este será o modelo que resultará dos cálculos do LIME. Como exposto na secção 3.2, relativa à regressão linear, a reta do modelo 2 corresponde à representação formal de uma reta de regressão do tipo:

$$y = \beta_0 + \beta_1 temperatura + \beta_2 humidade + \beta_3 pressão \quad (2)$$

Os atributos temperatura, humidade e pressão são as variáveis independentes e os coeficientes, β_1, β_2 e β_3, associados a cada variável independente, são os pesos destas. O LIME utiliza estes coeficientes, para interpretar o quanto estes atributos contribuem para o valor da previsão da instância de interesse. Quanto maior o seu valor absoluto, maior é o seu contributo. Se o valor do coeficiente for negativo, o LIME considera que se trata de uma contribuição negativa. Se o valor do coeficiente for positivo, então é considerado que a contribuição é positiva. Portanto, o LIME recorre a um modelo interpretável para gerar explicações de uma previsão local.

Existem dois fatores que podem influenciar o modelo gerado pelo LIME:

- A quantidade de observações geradas aleatórias;
- O peso dos pontos das previsões das observações geradas aleatoriamente em relação à instância de interesse.

Na biblioteca que implementa o LIME (secção 3.3.1), é possível indicar um valor para a quantidade de observações aleatórias, para além do definido por padrão. Uma das consequências é que um maior ou menor número de observações aleatórias poderá fazer com que o LIME gere modelos diferentes, os quais podem estar mais ou menos corretos para a compreensão da previsão da instância de interesse. Para além disso, o peso dos pontos

aleatórios é outro parâmetro que poderá fazer com que o LIME produza diferentes modelos. O artigo (Ribeiro et al., 2016) não define a forma de cálculo deste peso. A geração do modelo interpretável do LIME que melhor se aproxima da instância de interesse é matematicamente expresso por:

$$\mathcal{E}(x) = \underset{g \in G}{\operatorname{argmin}} \mathcal{L}\,(f, g, \Pi x) + \Omega(g) \tag{3}$$

, onde,

- $\mathcal{E}(x)$, é o modelo explicativo da instância de interesse x e que minimiza a função de perda $\mathcal{L}$.
- x, é a instância de interesse;
- G, é o conjunto dos potenciais modelos interpretáveis, tais como modelos lineares, árvores de decisão e outros;
- g, é uma possível modelo interpretável do conjunto G.
- $\mathcal{L}$, é a função de otimização que mede o quão próximo a explicação g, i.e., o modelo interpretável, está da instância de interesse x. Recebe os seguintes parâmetros:
    - f, função do modelo de ML utilizado na regressão ou na classificação. Este é utilizado para gerar as previsões das observações geradas aleatoriamente.
    - g, é o modelo interpretável dentre os possíveis G modelos;
    - Πx, é uma medida de proximidade entre os pontos. **Define os pesos** dos pontos na vizinhança do ponto de interesse. Também denominado de *kernel*.
      É este o parâmetro que irá ponderar a proximidade das novas observações aleatoriamente geradas na vizinhança da instância de interesse. Lembrando da imagem anterior, é este o parâmetro que irá definir o tamanho, ou peso, dos retângulos.
- $\Omega(g)$, mede a complexidade do modelo g, que os autores exemplificam da seguinte forma: "*For example, for decision trees Ω(g) may be the depth of the tree, while for linear models, Ω(g) may be the number of non-zero weights"* (Ribeiro et al., 2016). Portanto, este parâmetro atua como penalizador.

Na palavra dos autores: "*In order to ensure both interpretability and local fidelity, we must minimize L(f,g,πx) while having Ω(g) below enough to be interpretable by humans" (Ribeiro et al., 2016). Esta formulação visa encontrar o modelo que possa ser interpretável por seres humanos, e para tal, este modelo tem de apresentar o menor valor de* $\Omega(g)$ *e que minimiza o erro na previsão L(f,g,πx)*. Os autores referem o conceito *local fidelity*. Este conceito é a uma medida que permite avaliar o quanto o modelo interpretável se aproxima da previsão original, ou instância de interesse, do modelo caixa-preta (Molnar, C. 2022) e (Kuzlu et al., 2020).

Notar que a função modelo caixa-preta é parte integrante da equação e que corresponde ao parâmetro f da função de otimização $\mathcal{L}\,(f, g, \Pi x)$. No LIME, esta função é necessária não por que o LIME precise das previsões do modelo, mas sim, para gerar as previsões das observações

geradas aleatoriamente (Molnar, C. 2022). Lembrando da Figura 13, os retângulos são as previsões geradas pela função f.

3.3.1 Implementações

Na literatura é possível encontrar as seguintes implementações do LIME:

- **Versão na linguagem *python* (marcotcr/lime)**[10]: esta versão, disponível no *github*, foi desenvolvida por Marco Ribeiro, um dos autores do artigo (Ribeiro et al., 2016) no qual é apresentado o LIME. De acordo com a documentação, esta biblioteca tem suporte para os modelos de ML implementados na biblioteca *scikit-learn*;
- **Versão na linguagem R (cran.r)**[11]: uma outra versão, de Mara Averick, também baseada no artigo (Ribeiro et al., 2016).

Neste estudo será utilizada a primeira implementação, desenvolvida pelos autores do LIME. A Tabela 2 contém o endereço da documentação oficial da biblioteca do LIME utilizada neste estudo.

Tabela 2 – Documentação da biblioteca do LIME

Versão	Endereço
LIME 0.1	https://lime-ml.readthedocs.io/en/latest/ [12]

Será utilizada a palavra **explicador** para referir uma implementação concreta de um recurso da biblioteca utilizada para gerar explicações. Esta implementação do LIME suporta conjuntos de dados com os seguintes formatos: texto, imagem e tabular. A Figura 14 apresenta os tipos de explicadores disponibilizados pela biblioteca do LIME. Cada explicador é dedicado a um formato de conjunto de dados, e são agnósticos quanto ao modelo de ML, tal como referido na documentação desta implementação.

[10] marcotcr/lime – https://github.com/marcotcr/lime, último acesso em 15 de setembro de 2022. [Online]
[11] cran.r – https://cran.r-project.org/web/packages/lime/readme/README.html, último acesso em 15 de setembro de 2022. [Online]
[12] LIME *latest Website*– Último acesso em 15 de janeiro de 2022. [Online]

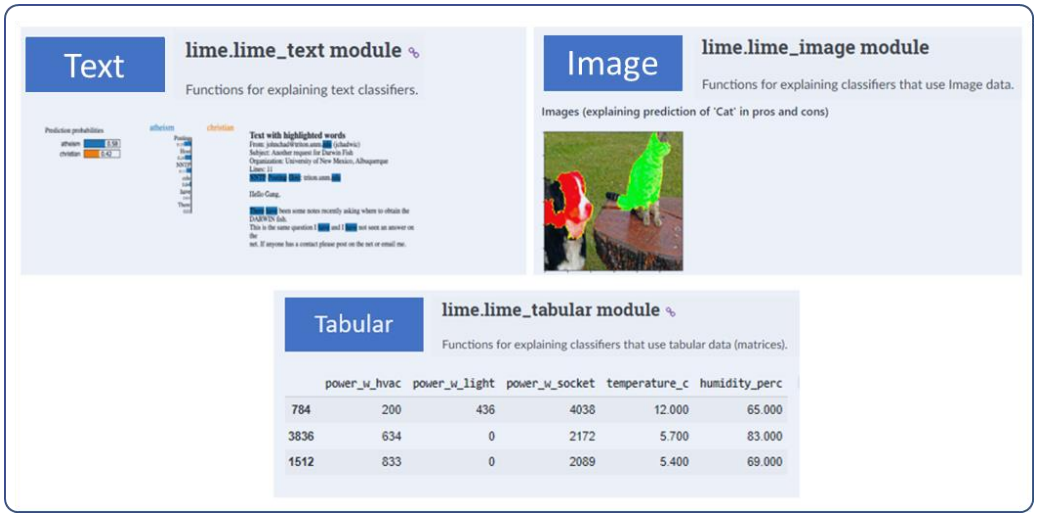


Figura 14 – Tipos de explicadores do LIME

3.3.1.1 Explicador *lime.lime_tabular.LimeTabularExplainer*

Dentre os explicadores apresentados, o *LimeTabularExplainer* [13] está vocacionado para conjuntos de dados no formato tabular, que será utilizado na aplicação deste trabalho no contexto do projeto PRECISE. A Figura 15 apresenta a definição deste explicador na biblioteca LIME.

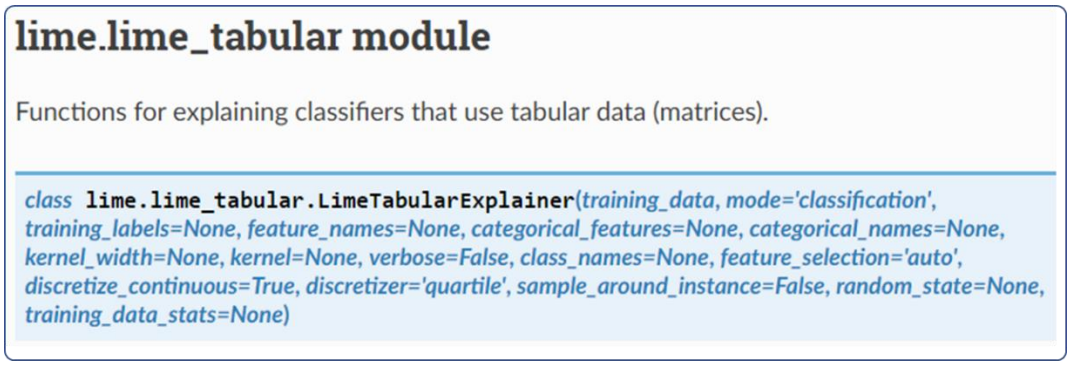


Figura 15 – Explicador *LimeTabularExplainer*

De acordo a figura acima, dentre os parâmetros deste explicador, destacam-se os seguintes:

- ***training_data*:** conjunto de dados de treino utilizado no treino do modelo de ML. Este parâmetro recebe o conjunto de dados com o qual o LIME irá gerar observações aleatórias em torno da instância de interesse, por via da introdução de perturbações nos valores originais presentes no conjunto de dados. Este conjunto de dados é o mesmo que foi utilizado para o treino do modelo de ML;
- ***mode*:** indica se é um caso de regressão ou classificação;

[13] *LimeTabularExplainer* – https://lime-ml.readthedocs.io/en/latest/lime.html#module-lime.lime_tabular, último acesso em 15 de janeiro de 2022. [Online]

- ***kernel_width*:** largura do *kernel*. Por padrão, o seu valor é determinado por: raíz quadrada(número de colunas) * 0,75. Contudo, o artigo (Ribeiro et al., 2016) não define a forma de cálculo deste peso. Este parâmetro irá determinar a largura, ou se quisermos os pesos, dos pontos gerados aleatoriamente em torno da instância de interesse (Molnar, C. 2022). Recordando a Figura 13, este parâmetro determina o peso dos retângulos mais próximos da instância de interesse.

Dentre as funções disponibilizadas por este explicador, a função *explain_instance()*[14] é utilizada para o cálculo do LIME. A Figura 16 apresenta a função do explicador do LIME e que é utilizada para gerar explicações para uma dada instância de interesse.

```
explain_instance(data_row, predict_fn, labels=(1, ), top_labels=None, num_features=10,
num_samples=5000, distance_metric='euclidean', model_regressor=None, sampling_method='gaussian')
```

Figura 16 – Função para a geração das explicações do LIME

De acordo a figura acima, dentre os parâmetros desta função, destacam-se os seguintes:

- ***data_row*:** valores dos atributos da instância de interesse da qual se pretende obter explicações. Recordando a equação (3), corresponde ao parâmetro x de $\mathcal{E}(x)$;
- ***predict_fn*:** função de previsão do modelo de ML. Na equação (3), corresponde ao parâmetro f da função de otimização $\mathcal{L}(f, g, \Pi x)$. Esta função irá gerar as previsões das observações geradas aleatoriamente em torno da instância de interesse;
- ***num_samples*:** indica o total de observações aleatórias que serão geradas em torno da instância de interesse. Este é um dos parâmetros, referido na explicação do conceito do LIME, que influencia a definição do modelo interpretável;
- ***model_regressor*:** indica o tipo do modelo interpretável que deverá ser utilizado pelo LIME para calcular o modelo que melhor se ajusta aos valores previstos para as observações aleatórias e que esteja o mais próximo possível da previsão original calculada pelo modelo de ML para a instância de interesse. Por padrão, é utilizado o *Ridge*[15] da biblioteca *scikit-learn*. Na equação (3), esta parâmetro indica o modelo g da função de otimização $\mathcal{L}(f, g, \Pi x)$.

3.3.1.2 Ferramentas de explicação

Dentre os formatos para explicação disponibilizados na biblioteca LIME destacam-se: a) formato html, utilizado na explicação visual; b) lista de valores, que permite extrair os valores dos

[14] *explain_instance()* – https://lime-ml.readthedocs.io/en/latest/lime.html#lime.lime_tabular.LimeTabularExplainer.explain_instance, último acesso em 15 de janeiro de 2022. [Online]
[15] *Ridge* – https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html, último acesso em 15 de janeiro de 2022. [Online]

coeficientes calculados. A Figura 17 é um exemplo de uma explicação visual em **formato HTML** de um caso de regressão obtido da documentação oficial.

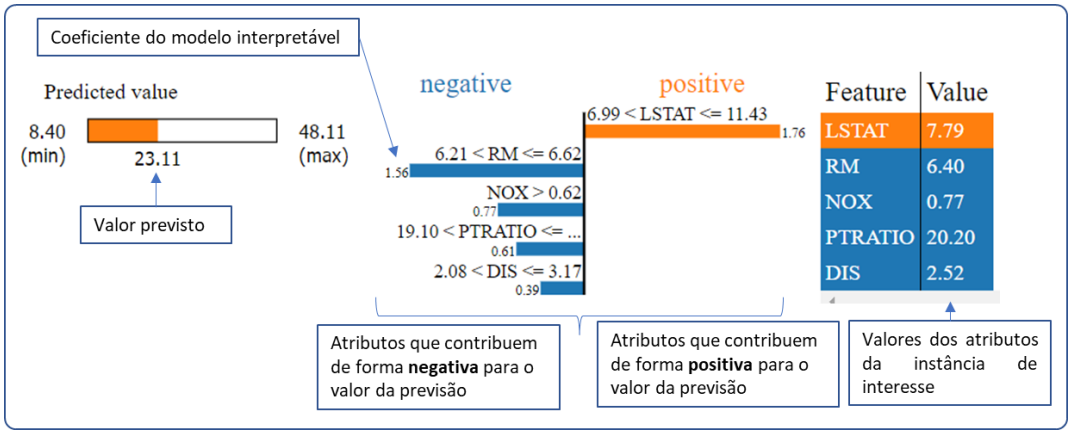


Figura 17 – Explicação visual formato HTML[16]

Este gráfico apresenta a seguinte informação:

- O valor **23.11**, à esquerda, é o valor previsto pelo modelo de ML;
- A tabela, à direita, contém os valores reais dos atributos da instância de interesse;
- O gráfico do meio apresenta a importância de cada atributo da instância de interesse que se pretende explicar.

Os atributos do gráfico do meio são ordenados em ordem decrescente de importância. A coluna ***negative*** corresponde aos atributos que contribuem de forma negativa para o valor previsto. A coluna ***positive*** corresponde aos atributos que contribuem de forma positiva para o valor previsto. Os valores apresentados junto das barras horizontais correspondem aos coeficientes do modelo interpretável. Por exemplo, o valor 1.56, destacado na figura acima, corresponde ao coeficiente do atributo RM, que neste caso este é o segundo atributo com maior importância para a previsão do valor da instância de interesse.

Considerando o que foi descrito quanto ao conceito de regressão linear (secção 3.2), este exemplo ilustra como o LIME recorre a este conceito para gerar explicações da instância de interesse. Os coeficientes (ou pesos) associados a cada atributo da instância de interesse são utilizados para determinar a importância de cada atributo para a previsão do modelo caixa-preta para uma dada instância de interesse. Isto permite gerar explicações visuais que apresentam a ordem de importância dos atributos da instância de interesse. Os coeficientes com valor positivos são classificados como tendo contribuição positiva na previsão. No sentido contrário, i.e., na contribuição negativa, estão os coeficientes com o valor negativo. Portanto, o formato HTML permite identificar quais os atributos que, segundo o LIME, um modelo de ML identificou como os mais relevantes para uma dada previsão.

[16] Formato HTML – https://marcotcr.github.io/lime/tutorials/Using%2Blime%2Bfor%2Bregression.html, último acesso em 15 de janeiro de 2022. [Online]

O **formato lista de valores** permite extrair os valores calculados pelo LIME. A Figura 18, obtida da documentação oficial do LIME, e relativa ao mesmo caso da Figura 17, ilustra a forma de se obter os valores calculados.

```
exp.as_list()
```

```
[('6.99 < LSTAT <= 11.43', 1.7571320048618118),
 ('6.21 < RM <= 6.62', -1.5638211582388033),
 ('NOX > 0.62', -0.77384372989110417),
 ('19.10 < PTRATIO <= 20.20', -0.60756112694664299),
 ('2.08 < DIS <= 3.17', -0.39085870918058263)]
```

Figura 18 – Explicação formato lista de valores[17]

Como se pode verificar, este formato apresenta exatamente os mesmos atributos do formato HTML e, inclusive, ordenados em ordem decrescente de importância. Muitos estudos combinam explicações visuais e textuais (secção 2.2.4). Esta funcionalidade do LIME permite elaborar explicações textuais que podem complementar as explicações visuais disponibilizadas pelo LIME ou ainda integrar explicações visuais de outro software.

3.3.2 Vantagens

As principais vantagens do LIME, referidas na literatura dedicada ao tema, são:

- É dos poucos métodos que suporta conjuntos de dados nos formatos tabular, texto e imagens (Molnar, C. 2022);
- Há, pelo menos, duas implementações, cada uma em uma linguagem *python* e outra em linguagem R (Molnar, C. 2022).

3.3.3 Desvantangens

As principais desvantagens do LIME, referidas na literatura dedicada ao tema, são:

- Perda de alguns atributos importantes do modelo de interpretação com a alteração dos parâmetros destes (Linardatos et al., 2021), (Garreau and & Luxburg, 2020);
- Diferentes interpretações podem ser obtidas para a mesma previsão do modelo de ML (Molnar, C. 2022).

[17] Formato lista de valores – https://marcotcr.github.io/lime/tutorials/Using%2Blime%2Bfor%2Bregression.html, último acesso em 15 de janeiro de 2022. [Online]

3.4 Shapley Value

Shapley Value (Shapley, Lloyd S. et al., 1953) é um conceito matemático, que tem origem na Teoria dos Jogos. Dado um jogo cooperativo, é atribuída uma distribuição de pontos entre os jogadores e proporcional à contribuição destes no resultado do jogo. Este conceito pode ser utilizado na explicação de previsões assumindo que: a) cada atributo do conjunto de dados é um jogador; b) a previsão de um modelo de ML é o resultado de um jogo. Desta forma, é possível calcular o *shapley value* de cada atributo de um conjunto de dados (Molnar, C. 2022) e assim determinar a contribuição de cada atributo (ou importância do atributo) para o valor previsto pelo modelo caixa-preta. O cálculo do *shapley value* é expresso por:

$$\phi_j(val) = \sum_{S \subseteq \{1,\dots,p\} \setminus \{j\}} \frac{|S|!\,(p - |S| - 1)!}{p!} (val(S \cup \{j\}) - val(S)) \quad (4)$$

, onde (Rozemberczki et al., 2022),

- ϕ_j é o *shapley value* do jogador j (ou atributo j da instância de interesse) para um certo valor val previsto;
- j é o jogador (ou atributo da instância de interesse) para o qual se pretende calcular o *shapley value*;
- S é o subconjunto com combinações, também denominados de **alianças**, de jogadores (ou dos atributos do conjunto de dados utilizado no modelo de ML) e que não inclui o jogador j;
- p é o total de jogadores (ou de atributos do conjunto de dados);
- $val(S)$ é o valor total de todos os jogadores (ou dos atributos do conjunto) de aliança S que não possui o jogador j (ou atributo j);
- $val(S \cup \{j\})$ é o valor dos jogadores (ou dos atributos) da aliança S com o jogador j (ou atributo j);
- $(val(S \cup \{j\}) - val(S))$ é a contribuição marginal do jogador j (ou atributo j).

Para ilustrar o cálculo do *shapley value*, será considerado um conjunto com três jogadores A, B e C e o cálculo da contribuição do jogador A. A Tabela 3 apresenta o valor atribuído a cada jogador e as combinações entre os três jogadores.

Tabela 3 – Tabela de jogadores

Jogadores	Valor
A	10
B	20
C	25
A,B	40
A,C	30
B,C	50
A,B,C	90

Uma vez que o objetivo é o cálculo da contribuição do jogador A, serão identificadas as alianças possíveis nas quais o jogador A não está presente: $S = \emptyset$, $S = \{B\}$, $S = \{C\}$ e $S = \{B, C\}$. Calculando a contribuição marginal para cada conjunto:

Para $S = \emptyset$: $(|0|! (3 - |0| - 1)!) * (val(A) - val(\emptyset)) = 2 * 10 = 20$;

Para $S = \{B\}$: $(|1|! (3 - |1| - 1)!) * (val(A, B) - val(B)) = 1 * 20 = 20$;

Para $S = \{C\}$: $(|1|! (3 - |1| - 1)!) * (val(A, C) - val(C)) = 1 * 5 = 5$;

Para $S = \{\mathrm{B}, C\}$: $(|2|! (3 - |2| - 1)!) * (val(A, B, C) - val(B, C)) = 2 * 40 = 80$

Aplicando a fórmula: $\phi_j(val) = \left(\frac{1}{6} * (20 + 20 + 5 + 80)\right) = 20{,}83$

Este processo é executado para os restantes jogadores considerando as possíveis alianças. O *shapley value* é a média da contribuição marginal de um jogador para todas as alianças possíveis (Molnar, C. 2022). No caso do jogador A, a sua contribuição média é 20.83. Transpondo este conceito para um conjunto de dados utilizado na aprendizagem de um modelo de ML, este método permite determinar a contribuição de cada atributo, do conjunto de dados, para uma previsão individual de um modelo caixa-preta.

Por fim, importa referir quatro propriedades do *shapley value*: eficiência, simetria, neutralidade e aditividade (Molnar, C. 2022). Não será realizado um estudo exaustivo destas propriedades, contudo importa referi-las para compreender alguns conceitos do SHAP. De seguida, apresentam-se mais detalhes sobre as propriedades.

Na **eficiência**, a soma das contribuições é igual à diferença da previsão de uma observação, ou instância de interesse, com a previsão média, e é expressa por:

$$\sum_{j=1}^{p} \phi_j = \hat{f}(x) - E_X(\hat{f}(X)) \quad (5)$$

, onde,

- ϕ_j é o *shapley value* do atributo;
- $\hat{f}(x)$, é a previsão do modelo de ML para uma instância de interesse x do conjunto de dados;
- $E_X(\hat{f}(X))$ é o valor médio esperado para o modelo de ML considerando todo um conjunto de dados X.

A equação (5) pode ser assim expressa:

$$\hat{f}(x) = \sum_{j=1}^{p} \phi_j + E_X(\hat{f}(X)) \quad (6)$$

Portanto, esta propriedade permite recuperar o valor da previsão de uma instância de interesse x do conjunto de dados a partir da soma do *shapley value* de cada atributo mais o valor médio esperado das previsões do modelo f, considerndo todo conjunto de dados X.

Na **simetria**, se dois atributos contribuem igualmente em todas as alianças, então o *shapley value* destes tem de ser igual.

Na **neutralidade**, se um atributo não contribui na previsão de qualquer aliança, então o *shapley value* é zero.

Na **aditividade**, para jogos com contribuição combinada, é possível somar cada *shapley value* calculado.

3.5 Shapley Additive Explanations

Tendo como base o conceito de *shaplay value*, em (Lundberg & Lee 2017) é proposto o SHAP que na palavra dos autores: "*We propose SHAP values as a unified measure of feature importance"*. O *shapley value* de cada atributo de um conjunto de dados é utilizado para medir o quanto cada uma destes contribui para a previsão de um modelo de ML (Molnar, C. 2022) e (Vilone & Longo, 2020). Em (Lundberg & Lee 2017), os autores referem que o foco é a interpretabilidade local, tal como o LIME. Contudo, a implementação do SHAP (secção 3.5.1) disponibiliza métodos de interpretabilidade global baseados na agregação do *shapley value* de cada atributo das instâncias de interesse.

O *shapley value* já permite calcular a contribuição de um atributo para o valor previsto por um modelo de ML (secção 3.4). Contudo, os autores do SHAP argumentam que este apresenta melhores resultados em termos de explicações pois é a unificação do cálculo da contribuição de um atributo que resulta da combinação de métodos de explicação, nos quais se inclui o LIME, com a consolidada fundamentação matemática do *shapley value*. A formulação matemática para o modelo explicativo do SHAP é dada por:

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j \tag{7}$$

, onde,

- g é o modelo explicativo;
- $z' \in \{0,1\}^M$ é um vetor binário de alianças e M é o tamanho máximo da aliança.
  O valor 0 da aliança indica que um dado atributo não está presente na aliança, enquanto o valor 1 indica que o atributo está presente (Molnar, C. 2022). Portanto, é o mesmo conceito do *shapley value* (secção 3.4), sendo que no SHAP o atributo não é retirado da aliança e a sua ausência é representada pelo valor 0 (Molnar, C. 2022).
- $\phi_j \in \mathbb{R}$ é designado pelos autores de *feature attribution* do atributo *j* do conjunto de dados, o que na realidade é o *shapley value* de cada atributo que está no vetor z' (Molnar, C. 2022).

Desta forma, o conceito de *shapley value* foi adaptado como sendo um modelo de regressão linear retornando, assim, ao tema dos modelos interpretáveis, já abordado no LIME, como forma de interpretar previsões realizadas por um modelo de ML (Molnar, C. 2022).

Dentre as propostas de cálculo do *shapley value*, apresentadas pelos autores do SHAP, inclui-se o *Kernel SHAP*, classificado como sendo do tipo modelo agnóstico. Outras abordagens são o *DeepSHAP* e o *TreeSHAP*, classificados como sendo do tipo modelo específico, mas que estão fora do âmbito deste estudo. O *Kernel SHAP* resulta da combinação do *shapley value* com o LIME. Nesta proposta, os autores adaptaram a formulação do LIME seguinte forma:

$$\Omega(g) = 0, \tag{8}$$

$$\pi_{x'} = \frac{(M-1)}{(M\ choose\ |z'|)|z'|(M - |z'|)} \tag{9}$$

$$\mathcal{L}(f, g, \Pi x) = \sum_{z' \epsilon Z} [f(h_x(z')) - g(z')]^2 \pi_{x'}(z') \tag{10}$$

Os autores redefiniram as funções do *kernel* (9) e de otimização (10) do LIME. Mais uma vez, o modelo de ML faz parte do processo de cálculo, aqui representada por f. Ao redefinir o *kernel* os autores redefinem o cálculo da distância entre o valor previsto pelo modelo de ML e os novos valores gerados aleatoriamente. Na expressão $[f\left(h_x\left(z'\right)\right) - g(z')]^2$ a função $g(z')$ é a definida em (7), i.e., o modelo explicativo.

Acerca do vetor binário $z' \in \{0,1\}^M$ importa referir o seguinte aspeto. Considerando um conjunto de dados em formato CSV, com cinco atributos, o qual é treinado em um modelo de ML. Uma vez que o modelo foi treinado com cinco atributos, estes não podem ser retirados tal como é feito no *shapley value* (secção 3.4), pois o modelo faz parte da formulação do SHAP como se pode verificar em $[f\left(h_x\left(z'\right)\right) - g(z')]^2$. Assim, a ausência do atributo é representada pelo valor 0. Esta é a forma encontrada pelos autores do SHAP para adaptar a lógica do cálculo marginal de um atributo do *shapley value* em modelos de ML. Uma questão que se põe é a seguinte: Se o modelo de ML é parte integrante da definição formal do *Kernel SHAP*, como o vetor binário de alianças $z' \in \{0,1\}^M$ será processado, uma vez que o modelo não é treinado com este vetor, mas sim com os valores reais do conjunto de dados? A resposta está nesta expressão: $f\left(h_x\left(z'\right)\right)$ (Molnar, C. 2022).

A função $h_x(z')$ converte o vetor binário de alianças para os valores reais dos atributos para que possam ser processados pelo modelo de ML f (Molnar, C. 2022). O vetor de alianças é binário em que o valor 1 indica a presença de um atributo e o valor 0 indica a ausência de um atributo. No caso do valor 1, a função $h_x(z')$ converte o valor 1 para o valor original do atributo. No caso do valor zero, este será convertido para um valor real aleatório o qual é obtido a partir de um conjunto de dados que regra geral é o conjunto de dados, ou parte deste, utilizado no processo de treino do modelo de ML. Este conjunto de dados é tipicamente denominado de ***background dataset*** (BDS) (secção 3.5.1).

A documentação disponível não é clara acerca da existência de uma fórmula, ou de um método, de como determinar o BDS. Este tema já foi abordado em um grupo de discussão (*Choosing the background set*)[18] que existe disponível no *github* e mantido pelos autores do SHAP.

De acordo com (Molnar, C. 2022), o processo de cálculo do SHAP pode ser resumidamente expresso nestas etapas:

- Exemplares de alianças $z'_k \in \{0,1\}^M$, $k \in \{1, \dots, K\}$ (1 = atributo presente na aliança, 0 = atributo ausente da aliança);
- Obter a previsão para cada z'_k realizando, primeiramente, a conversão de z'_k para os valores originais e aplicar o modelo de ML f: $f\left(h_x\left(z'\right)\right)$;
- Calcular o peso para cada z'_k utilizando o *Kernel*;
- Determinar o modelo de regressão linear que melhor se ajusta às previsões;
- Retornar os valores ϕ_k , i.e., o *shapley value* – fórmula (7) – que são os coeficientes do modelo linear.

[18] *Choosing the background set* – https://github.com/slundberg/shap/issues/391, último acesso em 15 de janeiro de 2022. [Online]

Para ilustrar o funcionamento do *Kernel SHAP*, será considerado o conjunto de dados, em formato tabular, utilizado na secção 3.3 dedicada ao LIME e apresentado na Figura 19. Este conjunto de dados será o *background dataset*.

Temperatura (C)	Humidade (%)	Pressão (mbar)
20	50	1050
25	60	1010
15	45	1029
30	80	980
35	65	1025
...	...	...
30	60	1030

Figura 19 – Conjunto de dados e instância de interesse

O SHAP irá gerar várias alianças a partir do BDS. Estas correspondem ao vetor binário $z' \in \{0,1\}^M$, em que $M = 3$. Considerando uma aliança na qual os atributos humidade e pressão estão ausentes, uma possível ocorrência do vetor binário é a seguinte: (1, 0, 0). A Figura 20 ilustra a conversão dos valores do vetor binário.

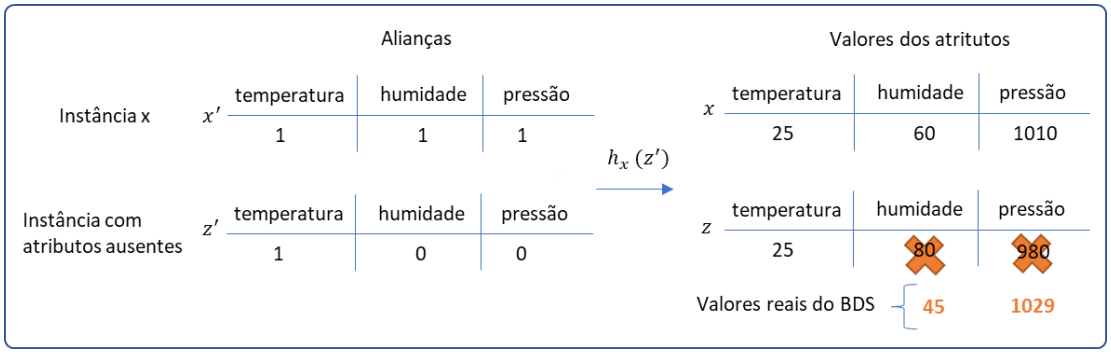


Figura 20 – Mapeamento das alianças para os valores reais[19]

Nas tabelas à esquerda estão representados dois vetores binários. O vetor x' representa a existência de todos os atributos da instância de interesse. O vetor z' é um dos possíveis vetores de alianças que será utilizado para o cálculo do *shapley value* o qual é convertido pela função $h_x(z')$. Para o atributo temperatura, a função $h_x(z')$ irá converter o valor 1 para o valor real do atributo, ou seja 25 (Figura 19). Para os atributos humidade e pressão, a função $h_x(z')$ irá converter o valor 0 para um dos possíveis valores do BDS, que neste caso foram 45 e 1029

[19] Mapeamento das alianças para os valores reais – Inspirado no livro (Molnar, C. 2022) em https://christophm.github.io/interpretable-ml-book/shap.html, último acesso em 15 de janeiro de 2022. [Online]

(Figura 19), respetivamente. Este processo repete-se para todas as alianças geradas pelo SHAP. Após a conversão, este resultado será utilizado pelo modelo de ML, representado na expressão $f\left(h_x\left(z'\right)\right)$, para realizar novas previsões a partir do resultado da função $h_x(z')$. Assim, e tal como no LIME, o processo calcula o modelo interpretável que melhor se ajusta às previsões e que mais se aproxima da previsão individual original do modelo de ML.

Uma vez que o SHAP tem como base ao conceito de *shapley value*, as propriedades do segundo estão presentes no primeiro. Contudo, no artigo do SHAP os autores apresentam as propriedades do SHAP de uma forma diferente do apresentado no conceito de *shapley value*, identificando apenas três: acurácia local, ausência e consistência**.** Destas três propriedades, será dado destaque à **acurácia local**.

A propriedade **acurácia local** do SHAP corresponde à propriedade eficiência (secção 3.4) do *shapley value* (Molnar, C. 2022). Os autores do SHAP explicam que, ao substituir na equação (7) o vetor z' por uma aliança x'contendo apenas o valor 1 em todos os atributos, surge esta equação:

$$f(x) = g(x') = \phi_0 + \sum_{j=1}^{M} \phi_j x_j' \qquad (11)$$

Uma vez que $x^{'}$ contém apenas o valor 1, e se ϕ_0 for definido como sendo $E_X(\hat{f}(X))$, tal como descrevem os autores do SHAP, i.e., o valor médio esperado descrito na secção do *shapley value* na explicação da propriedade de eficiência (secção 3.4), então a equação (11) pode ser assim escrita:

$$\hat{f}(x) = E_X(\hat{f}(X)) + \sum_{j=1}^{M} \phi_j \qquad (12)$$

A equação (12) é a definição formal da propriedade eficiência (equação 6 na secção 3.4) do *shapley value* (Molnar, C. 2022). Tal como no *shapley value*, esta propriedade do SHAP permite recuperar o valor da previsão de uma instância de interesse x a partir da soma do *shapley value* de cada atributo da instância de interesse mais o valor médio esperado (Molnar, C. 2022). Sendo que, no SHAP, os autores denominaram a expressão $E_X(\hat{f}(X))$ de **valor base** (do inglês *base value*) (Lundberg & Lee 2017). Desta forma, é possível informar o quanto um atributo se desvia da média, i.e., do valor base (Molnar, C. 2022). Este desvio indica o quanto o atributo contribui para pressionar (os autores utilizam a palavra *push*), o valor da previsão para cima ou para baixo em relação ao valor base (Lundberg & Lee 2017). A secção 3.5.1 ilustra este conceito a partir da biblioteca que implementa o SHAP.

3.5.1 Implementações

Tal como o LIME, existem, pelo menos, duas implementações do SHAP:

- **Versão na linguagem *python* (slundberg/shap)**[20]: esta versão disponível no *github* e desenvolvida por Lundberg, um dos autores do artigo (Lundberg & Lee 2017). De acordo com a documentação, esta biblioteca tem suporte para os modelos de ML implementados na biblioteca *scikit-learn*;
- **Versão na linguagem R (cran.r)**[21]: uma outra versão desenvolvida por Camilla Lingjærde, Martin Jullum & Nikolai Sellereite. É uma implementação do *Kernel SHAP* baseada no artigo (Lundberg & Lee 2017).

Neste estudo será utilizada a primeira versão que foi desenvolvida pelos autores do SHAP. A Tabela 4 apresenta a localização da documentação da biblioteca do SHAP e evidencia alguma dispersão desta documentação.

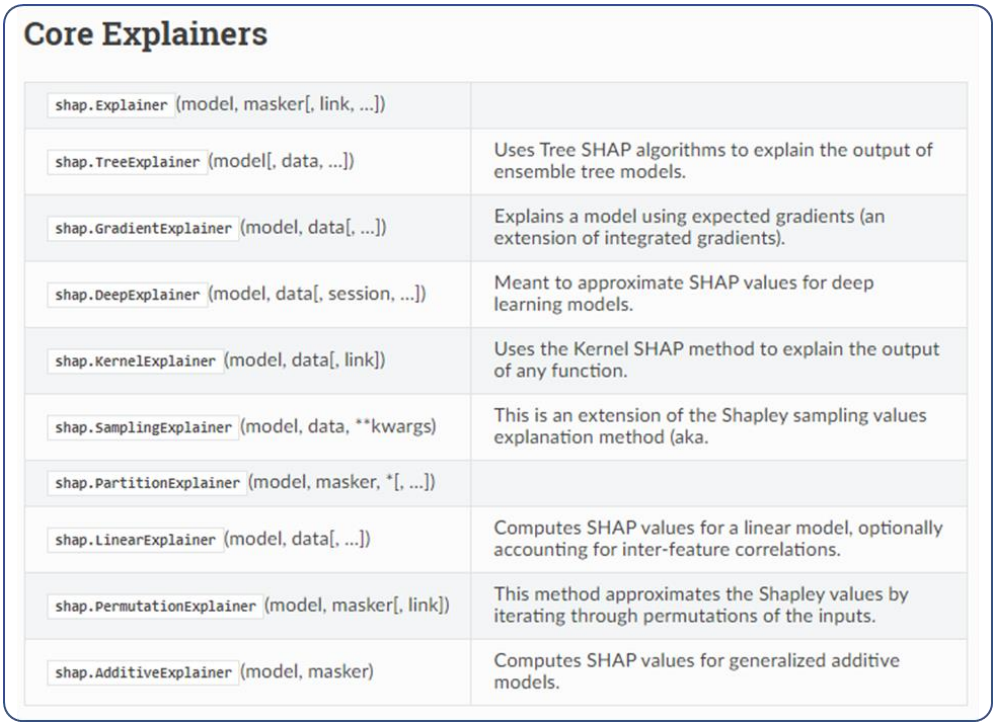
Tabela 4 – Documentação da biblioteca do SHAP[22]

Versão	Endereço
SHAP 0.41.0	https://shap.readthedocs.io/en/latest/api.html
SHAP 0.41.0	https://shap.readthedocs.io/en/stable/api.html
SHAP 0.41.0	https://shap-lrjball.readthedocs.io/en/latest/index.html
SHAP 0.41.0	https://shap-lrjball.readthedocs.io/en/stable/
SHAP 0.41.0	https://shap-lrjball.readthedocs.io/en/docs_update/index.html

Apesar desta dispersão, todos endereços referem a autoria de Scott Lundberg, um dos autores do artigo que deu origem ao SHAP. Ao longo deste documento poderá ser necessário referir mais do que um endereço para a função da biblioteca SHAP utilizada.

Tal como referido na implementação do LIME (secção 3.3.1), será utilizada a palavra **explicador** para referir uma implementação concreta de um recurso da biblioteca do SHAP utilizada para gerar explicações. A Figura 21 apresenta uma lista com os tipos de explicadores disponíveis na biblioteca do SHAP.

[20] slundberg/shap – https://github.com/slundberg/shap, último acesso em 15 de janeiro de 2022. [Online]
[21] cran.r – https://cran.r-project.org/web/packages/shapr/vignettes/understanding_shapr.html, último acesso em 15 de janeiro de 2022. [Online]
[22] Documentação biblioteca SHAP – Último acesso em 15 de janeiro de 2022. [Online]



Core Explainers

`shap.Explainer` (model, masker[, link, ...])	
`shap.TreeExplainer` (model[, data, ...])	Uses Tree SHAP algorithms to explain the output of ensemble tree models.
`shap.GradientExplainer` (model, data[, ...])	Explains a model using expected gradients (an extension of integrated gradients).
`shap.DeepExplainer` (model, data[, session, ...])	Meant to approximate SHAP values for deep learning models.
`shap.KernelExplainer` (model, data[, link])	Uses the Kernel SHAP method to explain the output of any function.
`shap.SamplingExplainer` (model, data, **kwargs)	This is an extension of the Shapley sampling values explanation method (aka.
`shap.PartitionExplainer` (model, masker, *[, ...])	
`shap.LinearExplainer` (model, data[, ...])	Computes SHAP values for a linear model, optionally accounting for inter-feature correlations.
`shap.PermutationExplainer` (model, masker[, link])	This method approximates the Shapley values by iterating through permutations of the inputs.
`shap.AdditiveExplainer` (model, masker)	Computes SHAP values for generalized additive models.

Figura 21 – Tipos de explicadores do SHAP[23]

A biblioteca do SHAP contém diferentes explicadores sendo que alguns são do tipo modelo específico, como por exemplo o *shap.DeepExplainer* vocacionado para modelos de DL. Desta lista, o *shap.KernelExplainer* é a implementação do conceito de *Kernel SHAP*.

3.5.1.1 Explicador *shap.KernelExplainer*

O explicador *shap.KernelExplainer*[24] é a implementação do conceito *Kernel SHAP* apresentado no artigo (Lundberg & Lee 2017), o qual é caracterizado pelos autores como sendo do tipo modelo agnóstico e será utilizado neste trabalho no contexto do projeto PRECISE. A Figura 22 apresenta a definição deste explicador da biblioteca do SHAP.

shap.KernelExplainer

class **shap.KernelExplainer**(*model, data, link=<shap.utils._legacy.IdentityLink object>, **kwargs*)

Figura 22 – Explicador *shap.KernelExplainer*

[23] Tipos de explicadores do SHAP – https://shap-lrjball.readthedocs.io/en/latest/api.html#core-explainers, último acesso em 15 de janeiro de 2022. [Online]
[24] *shap.KernelExplainer* – https://shap-lrjball.readthedocs.io/en/latest/generated/shap.KernelExplainer.html#shap-kernelexplainer, último acesso em 15 de janeiro de 2022. [Online]

Considerando a Figura 22, dentre os parâmetros deste explicador, destacam-se os seguintes:

- ***model*:** função de previsão do modelo de ML. Corresponde ao parâmetro $f\left(h_x\ (z')\right)$ da expressão $[f\left(h_x\ (z')\right) - g(z')]^2\ \pi_{x'}(z')$ da equação (10) da função de otimização;
- ***data*:** conjunto de dados utilizado no processo de cálculo da contribuição marginal de cada atributo. Este conjunto de dados é o *background dataset*.

A seguir, na Figura 23, é apresentada a função *shap_values()*[25], utilizada no o cálculo do *shapley value*, e que está disponível no explicador acima apresentado.

```
shap_values(X, **kwargs)
```

Figura 23 – Função para o cálculo do *shapley value*

A partir da Figura 23 acima, é feita uma breve descrição do parâmetro ***X***:

- ***X*:** recebe uma ou mais instâncias de interesse para as quais serão calculados o *shapley value*. Ao contrário do LIME, que recebe os valores dos atributos de apenas uma única instância de interesse, a função do SHAP permite receber valores dos atributos de várias instâncias de interesse.

3.5.1.2 Ferramentas de explicação

A biblioteca do SHAP contém um conjunto maior de ferramentas de explicação visual quando comparada com a biblioteca do LIME. Não serão descritas todas ferramentas de explicação visual desta biblioteca. Serão apresentados dois gráficos para interpretabilidade local: a) o gráfico de força (do inglês *force plot*); b) gráfico em cascata (do inglês *waterfall plot*). Para a interpretabilidade global será apresentado o gráfico sumário (do inglês *summary plot*). A escolha recaiu sobre estes gráficos visto serem os mais utilizados na literatura que refere a utilização do SHAP.

O gráfico de força[26] permite identificar a contribuição de cada atributo de uma instância de interesse para o valor previsto pelo modelo de ML. A Figura 24 é um exemplo obtido da documentação oficial.

[25] *shap_values()* – https://shap-lrjball.readthedocs.io/en/latest/generated/shap.KernelExplainer.html#shap.KernelExplainer.shap_values, último acesso em 15 de janeiro de 2022.
[26] gráfico de força – https://shap-lrjball.readthedocs.io/en/latest/generated/shap.force_plot.html#shap-force-plot, último acesso em 15 de janeiro de 2022. [Online]

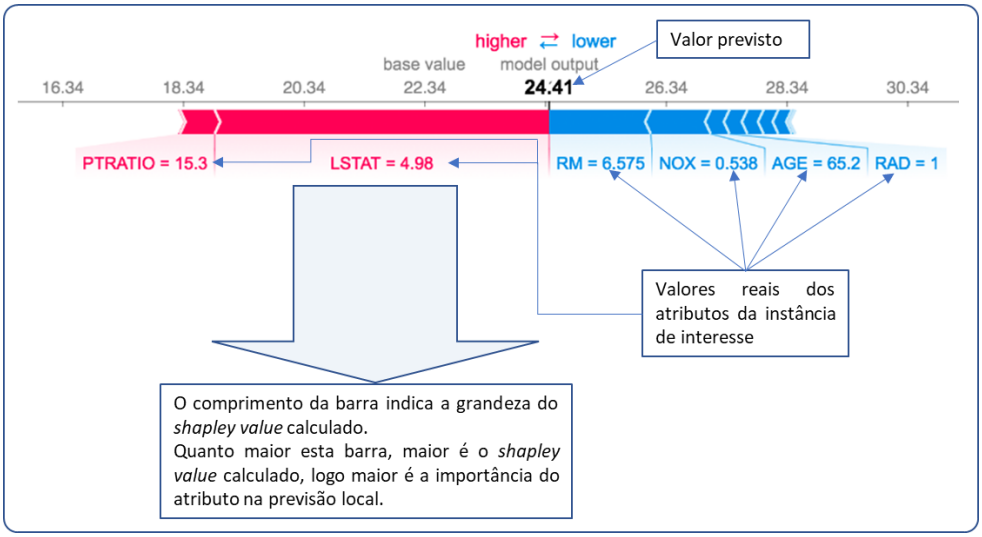


Figura 24 – Exemplo de um gráfico de força[27]

O gráfico de força fornece a seguinte informação:

- **Valor previsto *(Output value)*:** Corresponde ao valor **24.41**. Este é o valor previsto pelo modelo de ML para a instância de interesse em análise;
- **Cores vermelho e azul:** A cor vermelha indica o(s) atributo(s) que pressionam o aumento (*higher*) do valor da previsão a partir do valor base (*base value*). A cor azul indica o(s) atributo(s) que pressionam na diminuição (*lower*) do valor da previsão em relação ao valor base (*base value*). O comprimento das barras indica o quanto um atributo contribui para a previsão, i.e., a importância do atributo. Neste exemplo, o atributo LSTAT é aquele que, para o SHAP, o modelo de ML identificou como sendo o mais relevante, sendo que a sua contribuição é no sentido de aumentar o valor da previsão a partir do valor base. Este comprimento corresponde ao *shapley value* calculado para cada atributo;
- **Valor base (*Base value*):** é o conceito de valor base descrito na propriedade acurácia local exposto na secção 3.5 dedicada ao conceito de SHAP.

O gráfico em cascata[28], da Figura 25, também é utilizado para interpretabilidade local. Ao contrário do gráfico de força, o gráfico cascata permite visualizar o valor real do *shapley value* de cada atributo, ao invés de apresentar uma barra que indica a grandeza do *shapley value*.

[27] Exemplo de um gráfico de força – https://github.com/slundberg/shap#tree-ensemble-example-xgboostlightgbmcatboostscikit-learnpyspark-models, último acesso em 15 de janeiro de 2022. [Online]
[28] gráfico cascata – https://shap-lrjball.readthedocs.io/en/latest/generated/shap.waterfall_plot.html, , último acesso em 15 de janeiro de 2022. [Online]

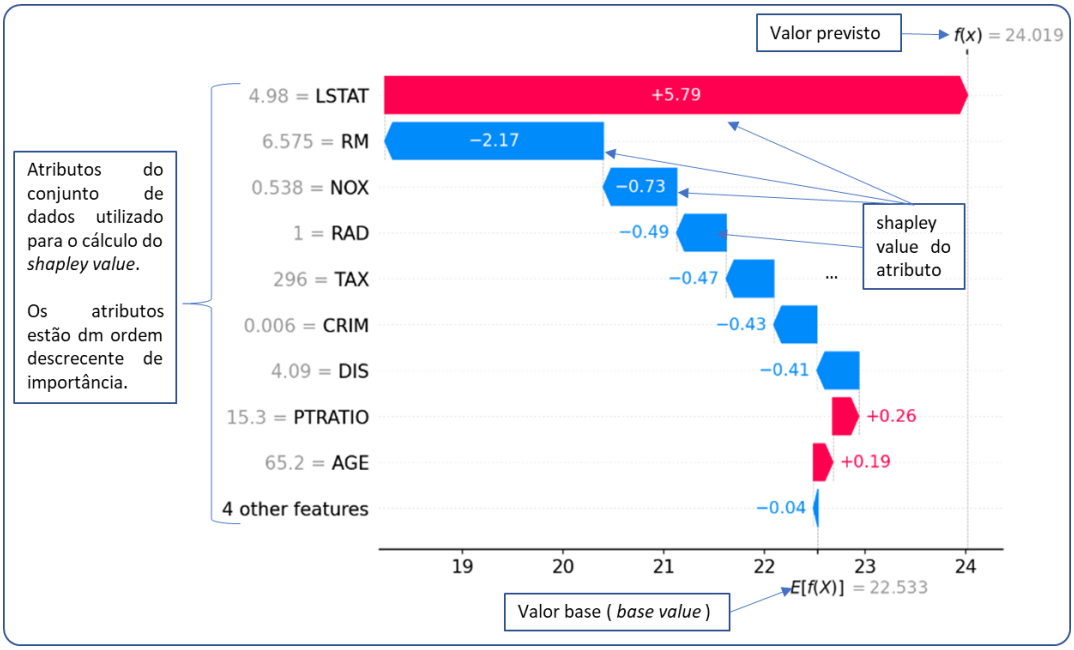


Figura 25 – Exemplo de um gráfico em cascata[29]

Neste gráfico é possível visualizar o conceito da propriedade acurácia local referido na explicação do conceito do SHAP. Ou seja, valor base, 22.533, mais a soma do *shapley value* de cada atributo, -0.04 + 0.19 + … + 5.79, terá de ser igual ao valor previsto 24.019, i.e., o valor previsto pelo modelo de ML[30].

O gráfico sumário[31] é a solução disponibilizada na biblioteca do SHAP para interpretação global e que se baseia na agregação dos valores do *shapley value* de cada atributo (Molnar, C. 2022). De acordo com a documentação da biblioteca do SHAP (slundberg/shap)[32], este gráfico utiliza o *shapley value* para exibir a distribuição dos impactos de cada atributo na previsão do modelo de ML, ou seja, apresenta o impacto global de cada atributo no conjunto de dados para o qual foram realizados os cálculos do *shapley value*. A biblioteca SHAP permite gerar este gráfico em dois formatos: *bar* (Figura 26), e *dot* (Figura 27).

[29] Exemplo de um gráfico cascata – https://github.com/slundberg/shap#tree-ensemble-example-xgboostlightgbmcatboostscikit-learnpyspark-models

[30] Neste exemplo, a referida soma apresenta uma pequena diferença que se justifica pelo arredondamento. Os valores do *shapley value*, do valor base e da previsão, apresentados no gráfico, possuem diferentes números de casas decimais de arredondamento diferente.

[31] gráfico sumário – https://shap-lrjball.readthedocs.io/en/latest/generated/shap.summary_plot.html#shap-summary-plot, último acesso em 15 de janeiro de 2022. [Online]

[32] slundberg/shap – https://github.com/slundberg/shap, último acesso em 15 de janeiro de 2022. [Online]

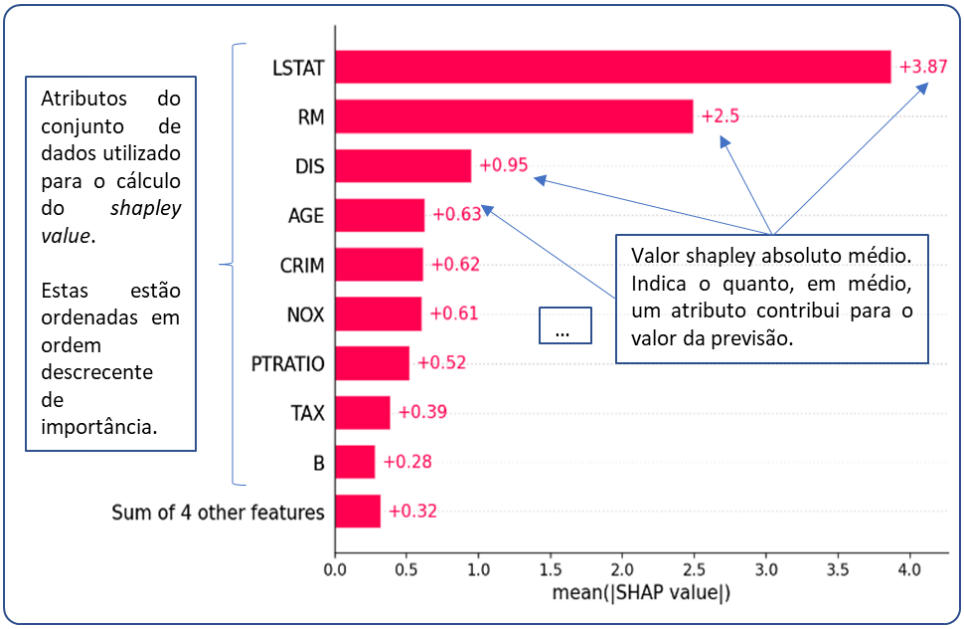


Figura 26 – Exemplo de um gráfico sumário tipo *bar*[33]

No tipo *bar*, os atributos estão ordenados em ordem decrescente de importância. Desta forma é possível identificar a importância de cada atributo no comportamento global do modelo. No exemplo acima, o atributo LSTAT é o que apresenta, em média, maior contribuição, considerando o *shapley value* calculado para cada atributo para um conjunto de dados. A sua contribuição para o comportamento do modelo é, em média, $\pm 3{,}87$.

[33] Exemplo de um gráfico sumário tipo *bar* – https://github.com/slundberg/shap#tree-ensemble-example-xgboostlightgbmcatboostscikit-learnpyspark-models, último acesso em 15 de janeiro de 2022. [Online]

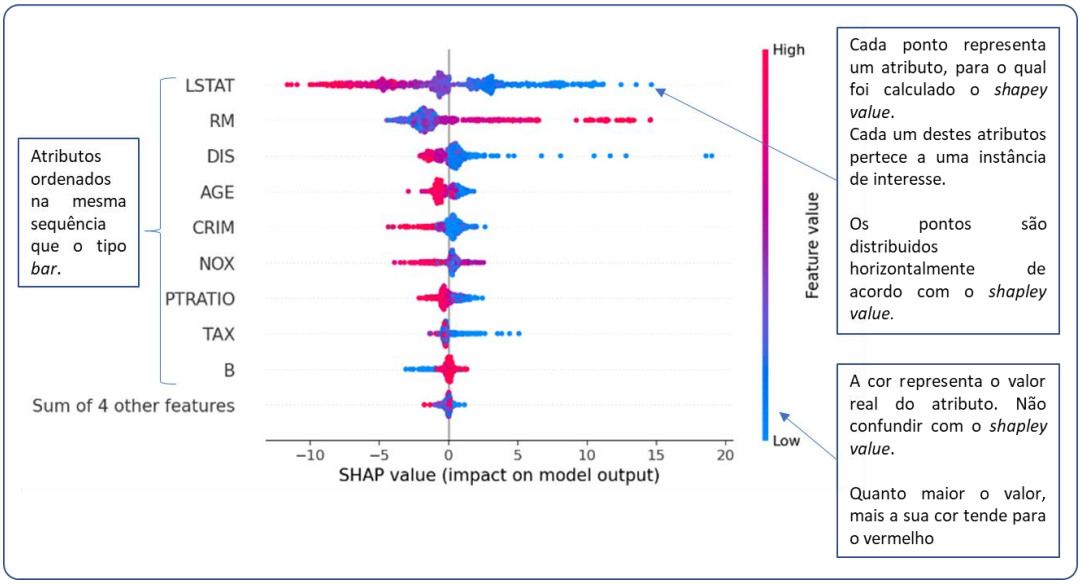


Figura 27 – Gráfico sumário do tipo *dot*[34]

O tipo *dot*, também denominado de *beeswarm*, apresenta as seguintes informações:

- **Importância do atributo:** Os atributos estão listados em ordem descendente de importância, tal como no tipo *bar*;
- **Impacto:** Corresponde ao eixo horizontal do gráfico. Indica o impacto do valor observado, i.e., do valor real do atributo. Os pontos à direita do valor zero do eixo horizontal apresentam impacto positivo, enquanto os pontos à esquerda deste valor apresentam impacto negativo nas previsões realizadas pelo modelo;
- **Valor original:** A cor vermelha indica que o valor observado é um valor alto. A cor azul, indica que o valor observado é um valor baixo.

Cada linha horizontal contém todas as ocorrências do respetivo atributo sendo que estas são distribuídas ao longo do eixo horizontal em função do seu *shapley value*. Por exemplo, no atributo LSTAT, o *shapley value* do ponto azul mais à direita é de aproximadamente 15. Uma das limitações deste gráfico é não ser possível identificar claramente qual o valor real do atributo, pois esta identificação é feita pela variação gradual da cor entre o azul e o vermelho.

3.5.1.3 Obter os valores calculados

A função *shap_values(X, **kwargs)* (Figura 23) utilizada para o cálculo do *shapley* value retorna uma estrutura de dados com o *shapley value* de cada atributo da instância de interesse, ou das instâncias de interesse. O valor base também é disponibilizado pelo explicador *shap.KernelExplainer*. Desta forma, é possível utilizar estes valores para gerar explicações textuais para complementar explicações visuais.

[34] Gráfico sumário tipo *dot* – https://github.com/slundberg/shap#tree-ensemble-example-xgboostlightgbmcatboostscikit-learnpyspark-models, último acesso em 15 de janeiro de 2022. [Online]

3.5.2 Vantagens

As principais vantagens do SHAP, referidas na literatura dedicada ao tema, são:

- Como o SHAP calcula o *shapley value*, todas as vantagens deste conceito se aplicam, além da base teórica sólida assente na teoria dos jogos (Molnar, C. 2022). A previsão é bem distribuída entre os valores dos atributos (Molnar, C. 2022) e (Puig & Carmona, 2019);
- Gera sempre as mesmas explicações em diferentes interações para o mesmo conjunto de dados (Molnar, C. 2022) e (Puig & Carmona, 2019);
- Há, pelo menos, duas implementações cada uma em uma linguagem *python* e outra em linguagem R (Molnar, C. 2022).

3.5.3 Desvantagens

As principais desvantagens do SHAP, referidas na literatura dedicada ao tema, são:

- O *Kernel SHAP* é lento. Isso torna seu uso impraticável quando se pretende calcular o *shapley value* para muitas instâncias (Molnar, C. 2022);
- O *Kernel SHAP* ignora a dependência entre os atributos de um conjunto de dados (Molnar, C. 2022);

3.6 Considerações Finais

O SHAP e o LIME utilizam modelos interpretáveis para gerar explicações locais de previsões realizadas por modelos de ML do tipo caixa-preta. O facto de o SHAP ter como base o conceito matemático da Teoria dos Jogos, o *shapley value*, faz com que este seja considerado mais robusto que o LIME. Por exemplo, ao contrário do LIME que gera novas explicações em diferentes interações para o mesmo conjunto de dados, isto não acontece no SHAP. Um aspeto final é o facto de estes métodos serem do tipo *post-hoc* (secção 2.2.2) uma vez que são executados após o treino do modelo de ML.

Por fim, numa primeira impressão, e sem avaliação com um grupo de utilizadores de diferentes perfis, a análise destas ferramentas de explicação visual sugere que a interpretação destas requer algum conhecimento especializado, nomeadamente de conceitos matemáticos. Desta forma, pode não ser imediata a utilização destas ferramentas por utilizadores sem este conhecimento especializado. No que refere à utilização destas bibliotecas em problemas de regressão, a documentação destas não é clara o suficiente para perceber em detalhe como interpretar os gráficos gerados.

O capítulo seguinte será dedicado à metodologia concebida para a elaboração dos casos de estudo utilizados para avaliar a utilização dos métodos LIME e SHAP em problemas de regressão utilizando um conjunto de dados real do tipo série temporal.

4 Metodologia para os Casos de Estudos

4.1 Introdução

Este capítulo apresenta a metodologia utilizada para a implementação de casos de estudo para análise de uma proposta de integração de modelos de ML com os métodos explicativos XAI (secção 4.2). É feita uma descrição do processo de identificação e recolha dos dados necessários para este trabalho (secção 4.3). Apresenta-se um cenário de previsão baseado em séries temporais (secção 4.4) que foi necessário implementar para este estudo, sendo que não é objetivo a análise do uso de modelos de ML para este tipo de previsões. Há uma breve descrição dos recursos das bibliotecas do LIME e do SHAP (secção 4.5) a serem utilizados. Os modelos de ML selecionados e respetivas métricas são também descritas neste capítulo, além de questões técnicas como as bibliotecas utilizadas no desenvolvimento e algumas características técnicas da máquina (secção 4.6 e secção 4.7). Há, ainda, uma breve referência aos casos de estudo (secção 4.8) e considerações da crescente preocupação acerca de aspetos éticos, de proteção de dados e análise de segurança (secção 4.9). Por fim, é feito um resumo (secção 4.10) dos pontos abordados neste capítulo.

4.2 Conceção da Integração de Modelos ML com Métodos XAI

A Figura 28 ilustra uma proposta de modelo de integração de tarefas de ML do PRECISE com os métodos explicativos XAI. Nesta proposta não estão identificadas todas as tarefas existentes no PRECISE para geração de previsões com base em modelos de ML.

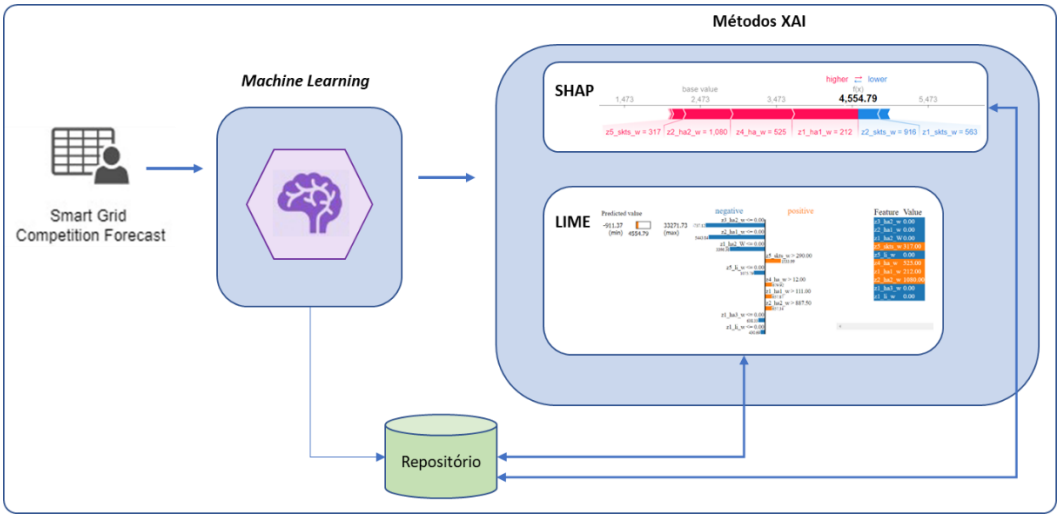


Figura 28 – Conceção da integração de modelos ML com métodos XAI

O componente *Machine Learning*, do PRECISE (secção 1.1), recebe o conjunto de dados, aqui representado por *Smart Grid Competion Forecast* e realiza as operações de ML implementadas. No componente Métodos XAI são executados os métodos explicativos. Os modelos e os dados de treino e teste existentes no repositório são carregados para o componente Métodos XAI e são geradas as explicações que, por sua vez, são persistidas no repositório. Desta forma, os resultados do LIME e do SHAP podem ser disponibilizados, imediatamente, a um utilizador ou em um momento futuro.

4.3 Identificação e Recolha de Dados

A identificação e recolha de dados considerou dois critérios. O primeiro teve por base o estudo de (Wastensteiner et al. 2021), onde é referido a existência de trabalhos de análise de consumo de eletricidade que utilizaram conjuntos de dados com séries temporais contendo registos com intervalos que iam desde os 15 minutos até aos 30 minutos. Portanto, entendeu-se que seria preciso encontrar um conjunto de dados com uma frequência de registos semelhante. Segundo, também referido no mesmo artigo, é o facto de muitos modelos de ML requererem uma grande quantidade de dados. O conjunto de dados selecionado pelos autores continha registos de consumos de moradias em um período de 76 semanas (julho de 2009 – dezembro de 2010) com intervalos de 30 minutos, portanto, um total de 2280 registos. Desta forma, esta foi a referência utilizada para a procura do conjunto de dados, em função da quantidade de registos.

Foram consideradas duas opções para selecionar o conjunto de dados: a) O mesmo conjunto de dados do artigo aqui referido. De acordo com o estudo, este é acessível a partir do site da *Irish Social Science Data Archive* [35] ; b) O conjunto de dados utilizado no evento

[35] *Irish Social Science Data Archive* – https://www.ucd.ie/issda/datasetsintheissda/commissionforenergyregulationcer/, último acesso em 15 de janeiro de 2022. [Online]

Smartgridcompetitions[36], promovido pelo GECAD, e acessível no site do evento. Uma vez que é mais rápido e mais fácil o acesso a este último ao conjunto de dados, decidiu-se pela segunda opção.

Dentre os diferentes conjuntos de dados disponíveis no site do evento *smartgridcompetition*s, foi selecionado o ficheiro excel "data1.xlsx - *Full Year of historical data (1st data set) (53 907 KB)*". Este contém um ano completo de dados históricos reais do ano de 2019 cujos registos possuem intervalos de 5 minutos contendo dados acerca de: a) Consumo total de um edifício; b) consumo de eletricidade por área construída; c) temperatura externa, informações meteorológicas; d) geração fotovoltaica. O facto de serem dados reais e conter informação do consumo de eletricidade tornam este ficheiro excel a fonte de dados ideal para este trabalho.

O ficheiro data1.xlsx contém onze folhas com dados históricos: *building_energy*, *building_sensor*, *zone#1_energy*, zone#1_sensor, *zone#2_energy*, *zone#2_sensor*, *zone#3_energy*, *zone#3_sensor*, *zone#4_energy*, *zone#5_energy* e *weather_data*. A folhas *zone#1_energy*, *zone#2_energy*, *zone#3_energy*, *zone#4_energy* e *zone#5_energy* contêm os dados da potência, da voltagem e da amperagem de cinco zonas do edifício. Nestas folhas estão registados os valores **dos consumos individuais de eletricidade, em watts (W), de três dispositivos**: **ar condicionado, lâmpadas e tomadas**, existentes **em cada zona**. A folha *building_energy* contém os totais **do consumo em watts** e da produção de energia do edifício. Ou seja, considerando o consumo de eletricidade, a folha *building_energy* contém o somatório do consumo de eletricidade de cada dispositivo em cada zona do edifício e registado nas folhas *zone#1_energy*, *zone#2_energy*, *zone#3_energy*, *zone#4_energy* e *zone#5_energy*. A folha *building_sensor* contém registos das temperaturas do edifício e a folha *weather_data* contém registos da temperatura, da humidade e da radiação. A folha zone#1_sensor contém a temperatura obtida da zona 1 do edifício e a intensidade de seis lâmpadas desta zona. As folhas *zone#2_sensor* e *zone#3_sensor* contêm a temperatura, a humidade e a intensidade das lâmpadas das zonas 2 e 3, respetivamente. Uma vez que o **objetivo deste estudo** é gerar **explicações de previsões de consumos de eletricidade**, optou-se por utilizar as seguintes folhas: *building_energy*, *zone#1_energy*, *zone#2_energy*, *zone#3_energy*, *zone#4_energy* e *zone#5_energy*. A partir destas folhas, será criado um conjunto de dados, em formato CSV, cujos atributos são obtidos a partir das colunas das referidas folhas.

4.4 Método de Previsão e Conjunto de Treino e Teste

Não é objetivo deste trabalho o estudo detalhado das técnicas de ML utilizadas em previsões de consumos de energia. Contudo, foi necessário definir um cenário de previsão baseado em séries temporais para a aplicação do LIME e do SHAP.

[36] *Smartgridcompetition*s – http://www.gecad.isep.ipp.pt/smartgridcompetitions/data/, último acesso em 15 de janeiro de 2022. [Online]

Importa uma breve introdução ao conceito de previsão (em inglês há a distinção entre forecast e predict): "*Forecasting, in its simplest form, deals with the prediction of a given quantity of interesting the future, given its available historical data*" (Stefani et al., 2022). Portanto, neste contexto, o exercício de previsão implica a existência de dados históricos. Para este trabalho, os dados de histórico são o conjunto de dados de uma série temporal de consumos de energia identificado na secção 4.2.

Uma série temporal pode ser definida como sendo uma sequência de histórico de medições de uma variável observável em igual intervalo de tempo (Bontempi et al., 2013). Ou ainda, uma série temporal pode ser considerada como uma sequência ordenada de observações em que cada observação x_i é registada em um instante de tempo t (Stefani et al., 2022).

O horizonte temporal de previsão é um aspeto importante a ter em atenção. Existem dois tipos: a) *one-step-ahead*; b) *multi-step-ahead*. O primeiro refere-se à previsão de um único instante de tempo t futuro, enquanto o segundo refere-se à previsão de uma sequência de instantes de tempo t futuros (Stefani et al., 2022). A Figura 29 ilustra o tipo de previsão *one-step-ahead forecast.*

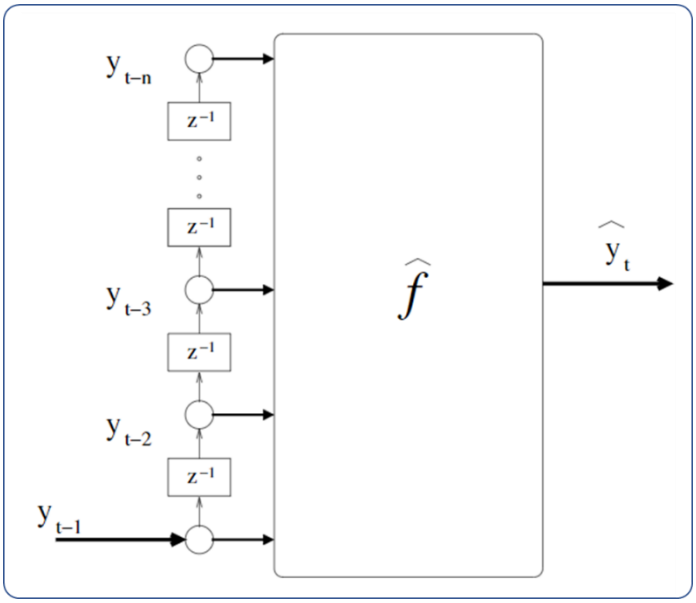


Figura 29 – *One-step-ahead forecast* (Bontempi et al., 2013)

Para um conjunto de dados histórico, $y_{t-n}, \ldots, y_{t-3}, y_{t-2}, y_{t-1}$ a função de previsão $\hat{f}$ realiza a previsão para o instante tempo t futuro. Transpondo este conceito para um modelo de ML, este é inicialmente treinado com os dados do histórico e a seguir é realizada a previsão.

O LIME e o SHAP são do tipo *post-hoc* (capítulo 3), ou seja, são métodos executados após o treino de um modelo de ML, sem que este tenha de, necessariamente, realizar as previsões. Uma vez que o modelo a ser escolhido tem de ser treinado, importa definir a metodologia a ser adotada.

O caso da previsão *one-step-ahead* pode ser tratado como um problema de aprendizagem supervisionada. Neste tipo de aprendizagem, os dados utilizados para o treinamento do modelo de ML incluem as soluções, ou os valores da variável dependente da qual se pretende que o modelo aprenda a prever (Géron et al., 2017). Uma forma de organizar um conjunto de dados de uma série temporal para aprendizagem supervisionada é a técnica da janela deslizante (do inglês *sliding window*) (Park et al., 2021). Nesta técnica, o conjunto de dados da série temporal é reestruturado de forma que os registos relativos ao instante $t - n$ conheçam o valor da variável dependente do registo do instante t seguinte. Ou seja, há um deslocamento nos registos relativos à variável dependente, que se pretende prever. A Figura 30 ilustra um exemplo deste processo com o deslocamento para cima de um instante no tempo:

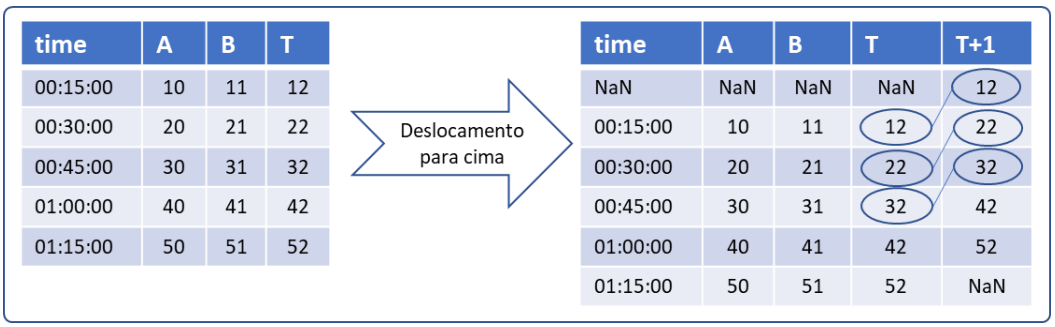


Figura 30 – Série temporal. Aprendizagem Supervisionada

Do lado esquerdo há uma série temporal em que os atributos A e B são as variáveis independentes e o atributo T é a variável dependente, ou seja, aquela que se pretende que um modelo de ML aprenda a prever. Do lado direito está o resultado do deslocamento para cima dos registos do atributo T. Este deslocamento deu origem à coluna T+1. Desta forma o registo dos atributos A e B do instante *time* = 00:15:00 conhecem o valor 22, do atributo T no instante seguinte *time* = 00:30:00. Por sua vez, o registo dos atributos A e B do instante 00:30:00 conhecem o valor 32, do atributo T no instante seguinte *time* = 00:45:00 e assim sucessivamente. Notar que o último registo, *time* = 01:15:00, não conhece nenhum valor de consumo do instante seguinte, pois este não existe. Desta forma este não entra no processo de treino. O mesmo acontece com o primeiro registo em que o atributo *time* tem o valor *NaN*.

Nesta metodologia pretende-se utilizar o LIME e o SHAP em uma sequência de previsões, ou seja, é um caso de *multi-step-ahead* utilizando sempre o mesmo modelo de ML. Sendo assim, será utilizada a estratégia recursiva (Bontempi et al., 2013). Esta consiste em fazer uma previsão *one-step-ahead*, para um instante t e a seguir adicionar ao histórico os dados do instante t previsto para prever o instante $t + 1$. Este processo é repetido até que o número desejado de etapas tenha sido executado.

Será criado um conjunto de dados, a partir do ficheiro excel "data1.xlsx - *Full Year of historical data (1st data set) (53 907 KB)*", para ser utilizado nas tarefas de treino e explicação. Este será dividido em dois conjuntos: treino e teste (secção 4.2). O conjunto de treino será utilizado para treinar os modelos. O conjunto de dados de teste será utilizado para a explicação, aplicando

multi-step-ahead e, em simultâneo, para a avaliação da performance do modelo com base nas métricas definidas (secção 4.6).

De forma a validar os resultados dos modelos e facilitar o cálculo para as métricas definidas, serão realizadas previsões dos modelos antes da geração das explicações. Com base nestas previsões, serão realizados os cálculos para as métricas definidas. Estes valores serão guardados para análise. A Figura 31 ilustra este processo de divisão do conjunto de dados em treino e teste, o treino do modelo, a explicação da instância de interesse e o *multi-step-ahead*.

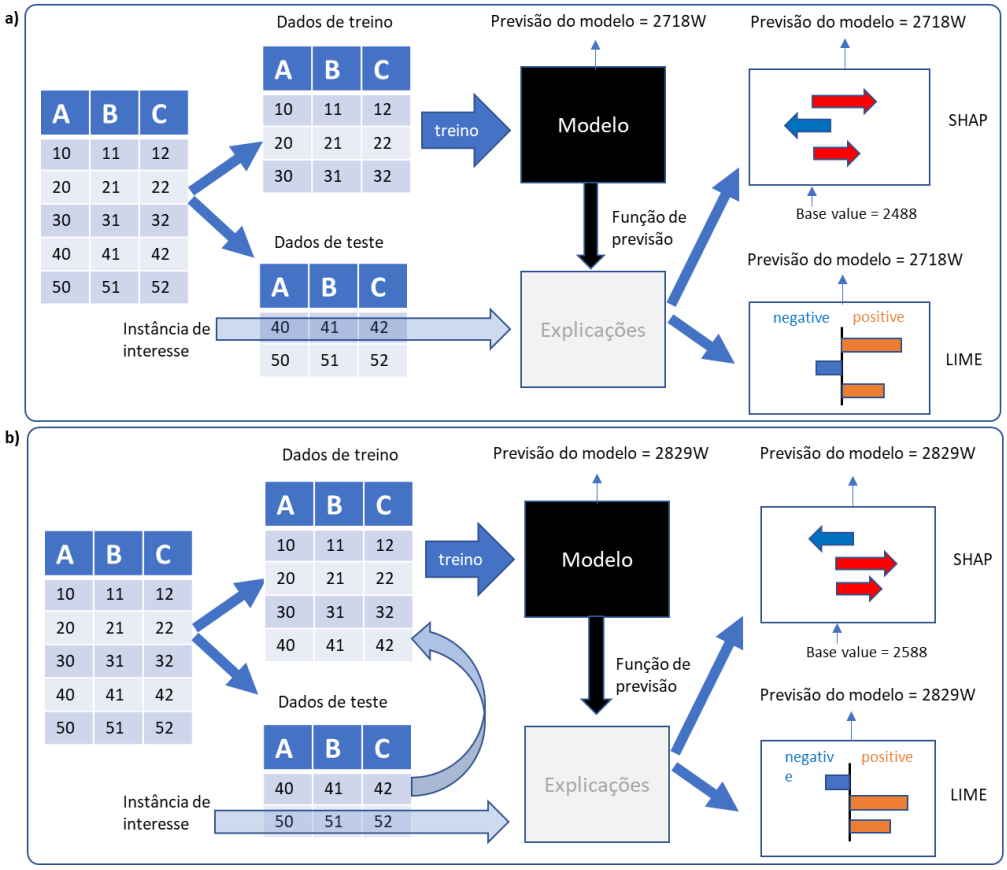


Figura 31 – (a) LIME e SHAP com *multi-step-ahead,* (b) LIME e SHAP com *multi-step-ahead*

Em (a), o modelo é treinado com o conjunto de dados de treino. O primeiro registo do conjunto de dados de teste é escolhido para instância de interesse. Os métodos explicativos são instanciados com a função de previsão do modelo e a instância de interesse para a geração das explicações visuais. Em (b), é representado o processo de *multi-step-ahead* no qual o registo do conjunto de dados de teste, utilizado no passo anterior, abastece o conjunto de dados de treino. Este novo conjunto irá alimentar o modelo para o seu treino e a instância de interesse seguinte é utilizada para gerar explicações. O LIME e o SHAP geram diferentes explicações para cada instância de interesse, às quais correspondem diferentes valores previstos pelo modelo. Este processo repete-se para todos os registos do conjunto de dados de teste.

4.5 Implementações do LIME e do SHAP

Para o LIME será utilizada a implementação *lime.lime_tabular.LimeTabularExplainer* para gerar as explicações. Este é o explicador cuja implementação está vocacionada para conjuntos de dados no formato tabular. Uma vez que será utilizado um conjunto de dados em formato CSV com registos de séries temporais de consumo de energia, este é o explicador adequado. Será utilizado o formato HTML para gerar explicações visuais. No caso do SHAP, será utilizado o explicador *shap.KernelExplainer* por ser do tipo modelo agnóstico e serão utilizados o gráfico de força e o gráfico sumário para gerar explicações visuais.

A escolha por explicadores do tipo agnóstico prende-se com o interesse em contemplar o maior número possível de modelos de ML utilizados no âmbito do projeto PRECISE. Pretende-se, ainda, recorrer às ferramentas de explicação visual disponíveis nas respetivas bibliotecas para interpretabilidade local e global. Os gráficos gerados pelas bibliotecas do LIME e do SHAP para cada interpretabilidade local serão armazenados em disco. O mesmo será feito para o gráfico de interpretabilidade global do SHAP.

4.6 Modelos de *Machine Learning* e métricas

Para a seleção dos modelos de ML foram considerados aqueles destacados na secção 2.2.5, dedicado aos modelos classificados como caixa-preta, e os modelos estudados nos artigos apresentados na secção 2.3, dedicada ao XAI em Sistemas de Energia. Assim, foram selecionados dois modelos para o estudo: a) RFR; b) ANN. Uma vez que o LIME e o SHAP têm suporte para as implementações do *scikit-learn* (secções 3.3 e 3.5, respetivamente), foram selecionadas as seguintes implementações desta biblioteca:

- ***sklearn.ensemble.RandomForestRegressor***[37]: implementação para o modelo RFR;
- ***sklearn.neural_network.MLPRegressor***[38]: implementação para o modelo ANN. Esta é uma implementação do *Multilayer Perceptron* que é um tipo de modelo ANN (Kaur et al., 2022). A implementação da biblioteca *scikit-learn* é também denominada de *Multilayer Perceptron Regressor* (MLPR) pelo facto de estar dedicado à problemas de regressão. Nos casos de estudo será utilizado o acrónimo MLPR para referir o modelo ANN.

Para a avaliação da performance dos modelos de ML utilizados em tarefas de regressão, foram definidas duas métricas para os modelos de regressão: a) *Root Mean Square Error* (RMSE); b)

[37] *sklearn.ensemble.RandomForestRegressor* – https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html, último acesso em 15 de janeiro de 2022. [Online]
[38] *sklearn.neural_network.MLPRegressor* – https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html?highlight=mlpr#sklearn.neural_network.MLPRegressor, último acesso em 15 de janeiro de 2022. [Online]

R-Squared (R^2) (Tomar et al., 2022) e (Park et al., 2021). Pretende-se um valor baixo para o RMSE e um valor alto para o R^2.

O RMSE é definido da seguinte forma:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (13)$$

Esta métrica expressa o erro médio do modelo em relação aos valores originais, sendo que penaliza as diferenças grandes entres o valor real e o valor previsto. Referir que o RMSE é a raiz quadrada do *Mean Squared Error* (MSE). Esta última é outra métrica que pode ser utilizada para avaliar modelos aplicados em cenários de regressão. Contudo, esta métrica não foi utilizada neste estudo.

O R^2 é definido pela seguinte fórmula:

$$R^2 = \sqrt{1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \quad (14)$$

Também conhecida como coeficiente de determinação, esta métrica representa o percentual da variância dos dados. Os resultados variam de 0 a 1 e, geralmente, também são expressos em termos percentuais, ou seja, variando entre 0% e 100%.

4.7 Ambiente de Execução e Ferramentas

O software foi desenvolvido e executado em uma máquina com sistema operativo Windows 10 Pro; processador: 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz 2.80 GHz; RAM: 32.0 GB: CPU: 64bits; Core(s): 8. Para a implementação dos casos de estudo foram utilizadas ferramentas de desenvolvimento e bibliotecas cujas designações e versões são apresentadas na Tabela 5.

Tabela 5 – Ferramentas utilizadas

Ferramenta	Versão	Ano
Linguagem *Python*[39]	3.7.12	2021
Biblioteca *scikit-learn*[40]	1.0.2	2021
Biblioteca *Pandas*[41]	1.1.5	2020
Biblioteca *NumPy*[42]	1.19.5	2020
Biblioteca *Matplotlib*[43]	3.5.3	2021
Pycharm IDE[44]	2022.2.1 (*Community Edition*)	2022
google colab[45]		

O *pycharm* foi utilizado como ambiente de desenvolvimento do software para os casos de uso. O *google colab* foi utilizado como laboratório para validação prévia de funcionalidades, como por exemplo a execução dos modelos de ML e mesmo dos métodos explicativos, além de permitir uma rápida leitura dos ficheiros do tipo CSV e excel.

4.8 Casos de estudo

Foram elaborados casos de estudo para avaliar a aplicação, com dados reais, das ferramentas de geração de explicações disponibilizadas pelas bibliotecas do LIME e do SHAP para as previsões dos modelos de ML (secção 4.6). Para além de avaliar a aplicação destas ferramentas, pretende-se realizar um estudo de possíveis interpretações que se podem fazer acerca do comportamento dos modelos referidos. Estes casos de estudo são detalhados no capítulo 6.

[39] *Python* – https://www.python.org/downloads/release/python-3712/, último acesso em 15 de janeiro de 2022. [Online]
[40] *scikit-learn* – https://scikit-learn.org/1.0/auto_examples/release_highlights/plot_release_highlights_1_0_0.html, último acesso em 15 de janeiro de 2022. [Online]
[41] *Pandas* – https://pandas.pydata.org/pandas-docs/version/1.1/index.html, último acesso em 15 de janeiro de 2022. [Online]
[42] *NumPy* – https://numpy.org/devdocs/release/1.19.5-notes.html, último acesso em 15 de janeiro de 2022. [Online]
[43] *Matplotlib* – https://matplotlib.org/3.5.3/users/installing/index.html, último acesso em 15 de janeiro de 2022. [Online]
[44] *Pycharm IDE* – https://www.jetbrains.com/help/pycharm/2022.2/quick-start-guide.html, último acesso em 15 de janeiro de 2022. [Online]
[45] google colab – https://colab.research.google.com/, último acesso em 15 de janeiro de 2022. [Online]

4.9 Aspetos Éticos, Proteção de dados e Análise de Segurança

A segurança é um dos desafios atuais para o uso de XAI (secção 2.3.2.1). A noção de segurança exposta refere-se à proteção dos sistemas baseados em AI contra ataques maliciosos que possam comprometer o funcionamento dos modelos de ML e também os métodos explicativos. Para além desta noção de segurança, importa ter em atenção mais dois aspetos: os aspetos éticos e a proteção de dados.

Não é objetivo deste estudo explorar a integração nos mecanismos de explicação dos métodos que respondam a estes três aspetos. Contudo, são apresentadas preocupações que devem ser consideradas no desenvolvimento de sistemas baseados em AI e na utilização de métodos explicativos.

4.9.1 Aspetos Éticos

As decisões baseadas nos resultados dos métodos de ML poderão ser tendenciosas ou simplesmente erradas. Isto levanta questões éticas seja no caso de mecanismos automáticos de decisão, seja nas recomendações que o sistema poderá sugerir aos utilizadores humanos.

No contexto deste trabalho, os resultados dos modelos de ML serão utilizados no estudo dos modelos explicativos. Os dados são a base dos modelos de ML e uma das fontes dos resultados inesperados destes. Existem diversos fatores que podem contribuir para resultados errados, como o uso de dados incompletos ou indevidamente alterados. A alteração dos dados pode ser feita de forma não intencional, ou mesmo intencional, como por exemplo por um ciberataque à rede, instalações ou equipamento elétricos. Tal como os processos de tomada de decisão, também os modelos explicativos podem ser afetados de forma negativa em função dos resultados obtidos dos modelos de ML.

O desenvolvimento dos modelos explicativos também deve ter em consideração aspetos éticos. Em (Lakkaraju & Bastani, 2020), os autores abordam o risco de os modelos explicativos manipularem a confiança dos utilizadores. Em (Slack et al. 2020) os autores estudam os modelos LIME e SHAP e explicam em que medida os seus resultados podem não ser confiáveis.

Um caso da importância deste tema ocorreu com a biblioteca *scikit-learn*. Esta é muito utilizada no desenvolvimento de soluções que utilizam ML. Um dos recursos disponibilizado nesta biblioteca é um conjunto de dados chamado *Boston dataset*[46]. Este contém registos de preços de imóveis da cidade de Boston, nos Estados Unidos. De acordo com fonte do site, foi identificado um problema ético neste conjunto de dados pois tende a considerar a segregação racial com um impacto positivo nos preços de imóveis.

[46] *Boston dataset* – https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_boston.html, último acesso em 15 de janeiro de 2022, online.

4.9.2 Proteção de Dados

Os dados utilizados nos modelos de ML podem ser obtidos de dispositivos como sensores ou dos recentes contadores inteligentes que, como referido em (Neto et al. 2019), "*correspondendo a sistema eletrónico que mede o consumo de energia, fornecendo mais informações do que um contador convencional, e que está preparado para transmitir e receber dados através de comunicações eletrónicas*". Ainda em (Neto et al. 2019), com a utilização dos referidos contadores, "*são recolhidas e transmitidas informações relativas ao exato consumo real de energia elétrica em relação a cada consumidor final e respetivo período real de utilização com uma intensa frequência (de 15 em 15 minutos – cf. artigo 7.º da Portaria n.º 231/2013, de 22 de julho, e a alínea f) do n.º 1 do respetivo Anexo I), com o objetivo declarado de promover a produção, a distribuição e o fornecimento racional e eficiente de energia elétrica*".

Desta forma, os dados utilizados nos modelos de ML implicam o tratamento de dados pessoais sempre que os consumidores finais sejam pessoas singulares e, porquanto, envolvem a recolha, conservação, comunicação e análise de informação relativa a indivíduos identificados ou identificáveis.

Algumas das consequências (Neto et al. 2019) no impacto sobre a vida privada dos titulares dos dados são:

- Possibilidade de deduzir um conjunto alargado de informação relativa aos mesmos (ex. quando se encontram em casa, se se encontra apenas uma pessoa ou mais, que eletrodomésticos estão a ser utilizados, quando estão ausentes);
- A criação e análise de perfis com base nas atividades que desenvolvem em casa;
- Risco de utilização indevida da informação, de negócio com a venda da informação e ainda de pretensão da sua utilização para os fins de investigação criminal.

A fonte de dados escolhida para este estudo é pública e foi obtida do site do evento *Smartgridcompetition*s promovida pelo GECAD (secção 4.3). Além de ser pública, estes dados estão descaracterizados o que não permite, por exemplo, identificar pessoas nem localizações geográficas.

4.9.3 Análise de Segurança

Por fim, o desenvolvimento de um sistema baseado na utilização de ML deve ter em conta a análise de segurança de forma que este seja robusto e confiável. O conceito de robustez aqui utilizado é baseado no estudo (Xiong et al., 2022): "*The robustness of an* ML *system can be defined as its resilience to malicious attacks to protect itself from the compromise of the system's integrity, availability, and confidentiality*".

Em (Xiong et al., 2022), os autores analisaram vários estudos o que permitiu identificar diferentes tipos de ataques aos sistemas que utilizam ML. Os autores exploram princípios e

boas práticas no sentido de apoiar o desenvolvimento de sistemas baseados em ML. Alguns dos tipos de ataques identificados podem ser assim classificados:

- Ataques nas diferentes tarefas de ML;

O fluxo de trabalho das tarefas de ML é modelado como uma sequência de tarefas denominada *pipeline*. Esta sequência de tarefas consiste em várias fases, incluindo coleta de dados, pré-processamento de dados, extração de atributos (do inglês *feature extraction*), treinamento e teste de modelo, previsão e, opcionalmente, novo treino do modelo (Xiong et al., 2022). Possíveis ataques neste pipeline são: a) *Stealthy Channel attack*: ataque que pode ocorrer na fase de coleta de dados; b) Polymorphic/Metamorphic: ataque na fase de extração de características; c) *Gradient Descent*: ataque que pode ocorrer na fase de aprendizagem do modelo; dentre outros.

- Acesso não autorizado a dados sensíveis e confidenciais

Os dados e informações relacionados a um sistema de ML, incluindo dados de treino, algoritmos e arquitetura de aprendizagem, hiper-parâmetros, função objetivo e parâmetros de modelo treinado (pesos), são considerados sensíveis ou confidenciais. Caso não seja considerado um adequado nível de autorização de acesso a estes dados e informações, pode ser possível um ataque que comprometa todo o sistema.

4.10 Considerações Finais

A identificação do conjunto de dados para este estudo teve como referência a pesquisa realizada no estado da arte. No sentido de avaliar a utilização do LIME e do SHAP em mais do que um modelo, foram escolhidos os modelos RFR e ANN tendo por base o estado da arte. No que refere às questões éticas, proteção de dados e análise de segurança, o conjunto de dados utilizado é público e foi descaracterizado, ou seja, não contém qualquer forma de identificação nomes das pessoas, endereços, valores como longitude ou latitude e outros que de alguma forma pudessem identificar pessoas singulares.

O SHAP disponibiliza explicadores do tipo modelo específico. Contudo, neste estudo pretende-se avaliar a utilização de explicadores do tipo modelo agnóstico de forma a permitir a integração em diferentes modelos de ML utilizados no contexto do PRECISE.

O capítulo seguinte tem forte cariz técnico. São apresentados extratos do software desenvolvido para os casos de estudo com destaque para os explicadores e as ferramentas disponibilizadas pelas bibliotecas do LIME e do SHAP.

5 Desenvolvimento e Implementação

5.1 Introdução

Neste capítulo são apresentados os principais desenvolvimentos realizados para a geração de explicações das previsões de consumos de energia em séries temporais utilizando os métodos explicativos LIME e SHAP. A primeira parte (secção 5.2) é dedicada à preparação do conjunto de dados com identificação dos atributos para as previsões e explicações e uma breve análise exploratória do ficheiro excel selecionado para este estudo. É descrito os processos de criação do conjunto de dados a partir do excel e de criação dos conjuntos de treino e teste para os modelos de ML. A seguir (secção 5.4) há uma breve descrição do desenvolvimento realizado para utilizar os modelos de ML. No que refere aos métodos explicativos (secção 5.4), são apresentadas as implementações desenvolvidas para a utilização das respetivas bibliotecas e ferramentas de explicação.

5.2 Preparação do Conjunto de Dados

5.2.1 Identificação dos Atributos para Explicação

A partir do ficheiro excel "data1.xlsx - Full Year of historical data (1st data set) (53 907 KB)", selecionado (secção 4.3) para ser a fonte de dados utilizada neste estudo, são obtidos os registos para a geração de um ficheiro CSV que irá conter os atributos que serão processados nas previsões dos modelos de ML assim como nos métodos explicativos LIME e SHAP. Estes atributos encontram-se nas seguintes folhas do excel: *building_energy, zone#1_energy*, *zone#2_energy*, *zone#3_energy*, *zone#4_energy* e *zone#5_energy* (secção 4.3). As tabelas Tabela 6, Tabela 7, Tabela 8, Tabela 9, Tabela 10 e Tabela 11 descrevem as colunas das referidas folhas e a que atributos irão corresponder no ficheiro CSV.

Tabela 6 – Atributo de consumo da folha *building_energy*

Coluna excel	Atributo CSV	Descrição
consumption (w)	consumption_w	consumo (W)

O atributo *consumption_w* é aquele do qual ser pretende gerar previsões. Este será denominado de variável, ou atributo, dependente. O seu valor é a soma dos consumos de cada dispositivo de cada zona do edifício.

Tabela 7 – Atributos de consumo da folha *zone#1_energy*

Coluna excel	Atributo CSV	Descrição
zone 1 - HVAC#1 power (W)	z1_ha1_w	zona 1 ar condicionado 1 consumo (W)
zone 1 - HVAC#2 power (W)	z1_ha2_w	zona 1 ar condicionado 2 consumo (W)
zone 1 - HVAC#3 power (W)	z1_ha3_w	zona 1 ar condicionado 3 consumo (W)
zone 1 - light power (W)	z1_li_w	zona 1 lâmpadas consumo (W)
zone 1 - sockets power (W)	z1_skts_w	zona 1 tomadas consumo (W)

Na folha da zona 1 existem três dispositivos do tipo ar condicionado, e ainda lâmpadas e tomadas. Não é possível identificar o total de lâmpadas e tomadas. A partir destes dados foram identificados cinco atributos para o ficheiro CSV.

Tabela 8 – Atributos de consumo da folha *zone#2_energy*

Coluna excel	Atributo CSV	Descrição
zone 2 - HVAC#1 power (W)	z2_ha1_w	zona 2 ar condicionado 1 consumo (W)
zone 2 - HVAC#2 power (W)	z2_ha2_w	zona 2 ar condicionado 2 consumo (W)
zone 2 - light power (W)	z2_li_w	zona 2 lâmpadas consumo (W)
zone 2 - sockets power (W)	z2_skts_w	zona 2 tomadas consumo (W)

Na folha da zona 2 existem dois dispositivos do tipo ar condicionado, e ainda lâmpadas e tomadas. Não é possível identificar o total de lâmpadas e tomadas. Nesta folha foram identificados quatro atributos para o ficheiro CSV.

Tabela 9 – Atributos de consumo da folha *zone#3_energy*

Coluna excel	Atributo CSV	Descrição
zone 3 - HVAC#1 power (W)	z3_ha1_w	zona 3 ar condicionado 1 consumo (W)
zone 3 - HVAC#2 power (W)	z3_ha2_w	zona 3 ar condicionado 2 consumo (W)
zone 3 - HVAC#3 power (W)	z3_ha3_w	zona 3 ar condicionado 3 consumo (W)
zone 3 - light power (W)	z3_li_w	zona 3 lâmpadas consumo (W)
zone 3 - sockets power (W)	z3_skts_w	zona 3 tomadas consumo (W)

Na folha A zona 3 apresenta um perfil de dispositivos semelhante à zona 1 (Tabela 7). Existem três dispositivos do tipo ar condicionado, e ainda lâmpadas e tomadas. Não é possível identificar o total de lâmpadas e tomadas. Foram identificados cinco atributos para o ficheiro CSV.

Tabela 10 – Atributos de consumo da folha *zone#4_energy*

Coluna excel	Atributo CSV	Descrição
zone 4 - HVAC power (W)	z4_ha_w	zona 4 ar condicionado consumo (W)
zone 4 - light power (W)	z4_li_w	zona 4 lâmpadas consumo (W)
zone 4 - sockets power (W)	z4_skts_w	zona 4 tomadas consumo (W)

Tabela 11 – Atributos de consumo da folha *zone#5_energy*

Coluna excel	Atributo CSV	Descrição
zone 5 - HVAC power (W)	z5_ha_w	zona 5 ar condicionado consumo (W)
zone 5 - light power (W)	z5_li_w	zona 5 lâmpadas consumo (W)
zone 5 - sockets power (W)	z5_skts_w	zona 5 tomadas consumo (W)

As folhas das zonas 4 (Tabela 10) e 5 (Tabela 11) apresentam igual perfil de dispositivos, com ar condicionado, lâmpadas e tomadas. Foram identificados três atributos em cada uma das folhas.

Os atributos obtidos das folhas, *zone#1_energy, zone#2_energy, zone#3_energy, zone#4_energy e zone#5_energy*, descritos nas tabelas acima serão denominados de variáveis, ou atributos, independentes. Portanto, existe **uma variável dependente**, o atributo *consumption_w* e **vinte variáveis independentes**. Todas estas folhas apresentam duas colunas com os valores da data e da hora do registo do consumo: coluna ***date*** e coluna ***time***. Estas duas colunas serão utilizadas para a união das folhas referidas no processo de criação de um conjunto de dados em formato CSV. A coluna *date* tem o formato ano-mês-dia e a coluna *time* tem o formato hora:minuto:segundo:

5.2.2 Análise Exploratória do Conjunto de Dados

Foi realizada uma observação visual dos dados das folhas *building_energy*, *zone#1_energy*, *zone#2_energy*, *zone#3_energy*, *zone#4_energy* e *zone#5_energy*. Estas folhas apresentam um cabeçalho que identifica cada dispositivo. A Figura 32 ilustra esta situação.

		zone 2 - HVAC#1			zone 2 - HVAC#2			zone 2 - light			zone 2 - sockets		
date	time	power (W)	voltage (V	amperage	power (W)	voltage (V	amperage	power (W)	voltage (V	amperage	power (W)	voltage (V	amperage (A)
01/01/2019	00:05:00	0	227	0	881	226	3,8	0	235	0	866	220	4,3

Figura 32 – Folha *zone#2_energy* do excel do evento *smartgridcompetition*

Há uma primeira linha que identifica cada um dos dispositivos de uma determinada zona do edifício. A linha a seguir a este cabeçalho é a que contém os registos do consumo de energia que correspondem à coluna *power (W)*.

A seguir, cada uma destas folhas foi convertida para um dataframe da biblioteca Pandas (pandas.DataFrame()) com os nomes ds_building_energy, ds_zone_1_energy, ds_zone_2_energy, ds_zone_3_energy, ds_zone_4_energy e ds_zone_5_energy, respetivamente. A conversão destas folhas obriga a ignorar este cabeçalho e a extrair apenas os registos das colunas power (W) de cada dispositivo. O Trecho de Código 1 seguinte ilustra a conversão da folha zone#2_energy.

```
def convert_zone_2_energy(self, xsls):
    ds_zone_2_energy = pd.read_excel(xsls, 'zone#2_energy',
                        skiprows=[0], usecols=[0, 1, 2, 5, 8, 11])
    new_columns_zone_2 = {'power (W)': "z2_ha1_w",
                          'power (W).1': "z2_ha2_w",
                          'power (W).2': "z2_li_w",
                          'power (W).3': "z2_skts_w"}
    ds_zone_2_energy.rename(columns=new_columns_zone_2, inplace=True)

    return ds_zone_2_energy
```

Trecho de Código 1 – Função para extrair informação da folha 2 do excel

A função *convert_zone_2_energy()* recebe o ficheiro excel e retorna um *dataframe* com os dados da folha *zone#2_energy*. O parâmetro *skiprows* permite ignorar o cabeçalho da folha e o parâmetro *usecols* permite indicar quais as colunas a considerar. Uma vez obtido o *dataframe*, as colunas do excel são renomeadas de acordo como definido na Tabela 8. A Figura 33 apresenta o resultado final desta função.

	date	time	z1_ha1_w	z1_ha2_W	z1_ha3_w	z1_li_w	z1_skts_w
0	2019-01-01	00:05:00	8	0	0	0	467
1	2019-01-01	00:10:00	4	0	0	0	469
2	2019-01-01	00:15:00	8	0	0	0	466
3	2019-01-01	00:20:00	8	0	0	0	463
4	2019-01-01	00:25:00	7	0	0	0	476

Figura 33 – *Dataframe* da folha *zone#2_energy*

No *dataframe* que resulta da função, as colunas*, date* e *time*, mantém-se, pois serão utilizadas para a união das diferentes folhas e as colunas da respetiva folha foram convertidas para os atributos definidos.

A lógica da função *convert_zone_2_energy()* foi implementado para as restantes folhas. A folha *building_energy* não apresenta cabeçalho, sendo assim foi necessário extrair, apenas, as colunas *date*, *time e consumption (w)*.

Antes de avançar para a união dos registos das folhas, foi realizada uma breve análise acerca da quantidade de registos, da ocorrência de valores nulos e se os registos da data e hora estavam corretos em todas as folhas. Esta última validação é importante para **garantir que o conjunto de dados apresenta o registo de todos os intervalos de tempo e que não há sobreposição destes para assim utilizar a série temporal**. O Trecho de Código 2 apresenta a função utilizada para validar os registos da data e da hora:

```
def validate_date_time(self, ds):
    ds_val_dt = pd.DataFrame()
    ds_val_dt["dt_ds"] = ds["date"] + " " + ds["time"]
    ds_val_dt["dt_ds"] = pd.to_datetime(ds_val_dt["dt_ds"])
    ds_val_dt["dt_gen"] = pd.date_range(start='2019-1-1 00:05:00',
                                        end='2020-01-01 00:00:00',
                                        freq='5min')
    ds_val_dt['output'] = ds_val_dt.apply(lambda row:
    row['dt_ds'].date() == row['dt_gen'].date()
    and row['dt_ds'].time() == row['dt_gen'].time(), axis=1)

    return ds_val_dt["output"].any()
```

Trecho de Código 2 – Função para validar as folhas do excel

A função *validate_date_time()* cria um *dataframe* que contém a data e hora do *dataframe* de cada folha do excel (recebido por parâmetro) e outra coluna com data e hora gerados programaticamente. Estes registos gerados apresentam um intervalo de cinco minutos, com início do dia 2019-01-01 no instante 00h05m e fim no dia 2020-01-01 no instante 00h00m e, portanto, equivalente ao esperado nas folhas do excel. O intervalo de cinco minutos justifica-se pelo que foi descrito em 4.2 acerca da escolha do conjunto de dados. Os instantes de início e de fim foram verificados em cada *dataframe* de cada folha do excel. A na Figura 34 apresenta um quadro resumo desta análise.

	ds_building_energy	ds_zone_1_energy	ds_zone_2_energy	ds_zone_3_energy	ds_zone_4_energy	ds_zone_5_energy
0	105120	105120	105120	105120	105120	105120
1	False	False	False	False	False	False
2	2019-01-01 00:05:00	2019-01-01 00:05:00	2019-01-01 00:05:00	2019-01-01 00:05:00	2019-01-01 00:05:00	2019-01-01 00:05:00
3	2020-01-01 00:00:00	2020-01-01 00:00:00	2020-01-01 00:00:00	2020-01-01 00:00:00	2020-01-01 00:00:00	2020-01-01 00:00:00
4	True	True	True	True	True	True

Figura 34 – Análise dos registos das folhas de excel do evento *smartgridcompetition*

O primeiro registo indica o total de registos de cada folha. Como se pode verificar, todas as folhas apresentam a mesma quantidade de registos. O segundo registo indica que não há valor nulo em nenhuma das folhas. O terceiro e o quarto registos apresentam a data e a hora do primeiro e do último registo de cada folha, respetivamente e, também aqui, não há diferenças. O último registo confirma que os registos da data e hora em cada folha apresentam intervalos de cinco minutos e que têm início no dia 2019-01-01, no instante 00h05m, e fim no dia 2020-01-01 no instante 00h00m.

5.2.3 Criar Conjunto de Dados para o Estudo

A partir dos objetos *dataframe ds_zone_1_energy, ds_zone_2_energy, ds_zone_3_energy, ds_zone_4_energy e ds_zone_5_energy*, foi realizada a união dos conjuntos de dados. A função do Trecho de Código 3 foi implementada para este efeito.

```
def _merge_ds_by_date_and_time(self, ds1, ds2):
    ds_merged = pd.merge(ds1, ds2, on=[DATE, TIME])
    return ds_merged
```

Trecho de Código 3 – Função para unir conjunto de dados

Esta função é chamada para cada par de *dataframe*, recebidos como parâmetros. A união é realizada pela função *merge()* com os atributos DATE e TIME. O resultado é um *dataframe* contendo os dados do par de *dataframe* recebidos.

O resultado final de todo o processo de união é um ficheiro em formato CSV com o nome gecad_competion_2019_A.csv. Este ficheiro será a base para a implementação dos casos de estudo. A Figura 35 apresenta os dados do primeiro registo do conjunto de dados criado.

	0
date	2019-01-01 00:00:00
time	00:05:00
consumption_w	2985
z1_ha1_w	8
z1_ha2_W	0
z1_ha3_w	0
z1_li_w	0
z1_skts_w	467

	0
date	2019-01-01 00:00:00
time	00:05:00
z2_ha1_w	0
z2_ha2_w	881
z2_li_w	0
z2_skts_w	866
z3_ha1_w	0

	0
date	2019-01-01 00:00:00
time	00:05:00
z3_ha2_w	0
z3_ha3_w	0
z3_li_w	0
z3_skts_w	351
z4_ha_w	9

	0
date	2019-01-01 00:00:00
time	00:05:00
z4_li_w	0
z4_skts_w	171
z5_ha_w	0
z5_li_w	0
z5_skts_w	232

Figura 35 – Primeiro registo do conjunto de dados do estudo

Nesta figura é possível identificar os atributos *date* e *time*, para a data e hora, respetivamente, a variável dependente, *consumption_w*, assim como e as vinte variáveis independentes. Este conjunto de dados apresenta registos com intervalos de 5 minutos. Verifica-se, um total 105120 registos, com intervalos de 5 minutos, e com um período desde 2019-01-01 até 2020-01-01. Para os primeiros casos de estudo optou-se por um conjunto de dados menor. Verificou-se que para o mês de janeiro, e considerando intervalos de 15 minutos, obtém-se um conjunto de

dados com 2976 registos. Este valor está acima do total de registos utilizado no estudo de (Wastensteiner et al. 2021) (secção 4.2). Desta forma, foi criado o ficheiro gecad_competion_2019_B.csv, a partir do ficheiro gecad_competion_2019_A.csv, contendo apenas os registos do mês de janeiro contendo intervalos de 15 minutos e com os mesmos atributos do ficheiro gecad_competion_2019_A.csv.

O motivo de utilizar este segundo conjunto de dados é por ser possível um conjunto com um total de registos próximo do artigo utilizado como referência e também para diminuir o tempo de processamento. Por fim, o conjunto de dados gecad_competion_2019_B.csv apresenta as seguintes características:

- Contém todos os atributos do conjunto de dados gecad_competion_2019_A.csv;
- Contém registos de consumo com intervalos de 15 minutos;
- Contém 2976 registos.

5.2.4 Separação em Conjunto de Dados de Treino e de Teste

O primeiro passo consiste em preparar o conjunto de dados gecad_competion_2019_B.csv para a previsão *multi-step-ahead* (secção 4.4). Para isto, foi criado o atributo *consumption_target* que contém os valores do atributo *consumption_w* deslocados para cima. Desta forma, o atributo *consumption_target* será a variável dependente para a qual se pretende realizar previsões e assim tratar o problema da previsão sem séries temporais como uma aprendizagem supervisionada. A Figura 36 ilustra este processo.

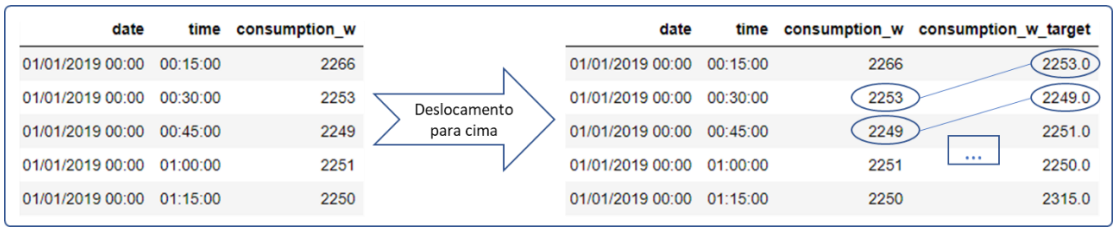


Figura 36 – Conjunto de dados e treino: série temporal como aprendizagem supervisionada

Cada registo passa a ter como variável dependente o valor do consumo do registo seguinte, i.e., dos 15 minutos seguintes. O último registo fica com o valor *NaN*, devido ao deslocamento, logo, este não será considerado para treino e explicações.

O conjunto de dados de teste será formado pelos últimos 10 registos, ou seja, os registos entre o intervalo 31/01/2019 21h30m e 31/01/2019 23h45m, como ilustrado na figura seguinte.

date	time	z1_ha1_w	z1_ha2_W	z1_ha3_w
31/01/2019 00:00	21:30:00	620	0	0
31/01/2019 00:00	21:45:00	239	0	0
31/01/2019 00:00	22:00:00	188	0	0
31/01/2019 00:00	22:15:00	672	0	0
31/01/2019 00:00	22:30:00	137	0	0
31/01/2019 00:00	22:45:00	212	0	0
31/01/2019 00:00	23:00:00	653	0	0
31/01/2019 00:00	23:15:00	44	0	0
31/01/2019 00:00	23:30:00	393	0	0
31/01/2019 00:00	23:45:00	568	0	0

Figura 37 – Excerto do conjunto de dados de teste

A seguir as funções *prepare_gecad_data()* do Trecho de Código 4 e *split_train_test()* do Trecho de Código 5, realizam as operações de deslocamento e a divisão do conjunto de dados em treino e teste.

```
def prepare_gecad_data(self):
    data = self.gecad_competition_2019(self._csv_name)
    self.ds_gc2019 = data.frame
    ds_temp = self.ds_gc2019.copy()
    ds_temp['consumption_target'] = ds_temp['consumption_w'].shift(-1)

    ds_temp = ds_temp.drop(ds_temp.tail(1).index)
    ds_temp = ds_temp.drop(columns='consumption_w', axis=1)

    train, test = self.split_train_test(ds_temp, self._split_index)

    return train, test
```

Trecho de Código 4 – Preparação do conjunto de dados para treino e teste

```
def split_train_test(self, ds, split_index):
    values_to_split = ds.values
    train = values_to_split[:split_index]
    test = values_to_split[split_index:len(values_to_split)]

    return train, test
```

Trecho de Código 5 – Criação do conjunto de dados de treino e de teste

Na função *prepare_gecad_data()* o ficheiro gecad_competion_2019_B.csv é carregado para memória em um *dataframe*. Este é mantido em memória em todo o processo. A partir da cópia do *dataframe* referido, é realizado o deslocamento para cima dos registos do atributo *consumption_w* os quais dão origem ao atributo *consumption_target*. A seguir são eliminados o último registo do conjunto de dados e a coluna do atributo *consumption_w*. No primeiro caso o registo é eliminado pois o valor do atributo *consumption_target* deste registo é *NaN*. No

segundo caso, o atributo *consumption_w* não entra no processo de aprendizagem. Por fim, a função *split_train_test()* o conjunto de dados, já com o deslocamento, é dividido em dois conjuntos: treino e teste. Esta função retorna duas estruturas do tipo *numpy array*.

5.3 Modelos de *Machine Learning*

5.3.1 Identificação dos Parametrização dos Modelos de *Machine Learning*

Não é objetivo de estudo avaliar os modelos de ML. Contudo, pretendeu-se conseguir uma configuração dos parâmetros destes que resultasse em modelos com elevados valores para as métricas definidas, pois esta é uma das motivações para a investigação de mecanismos explicativos que permitam perceber como estes modelos conseguem estes elevados valores.

A primeira aproximação para a definição dos parâmetros foi utilizar a implementação *GridSearchCV*[47] do *scikit-learn*. As funções *doGridSearch_MLPR()*, no Trecho de Código 6, e *doGridSearch_RFR(), no* Trecho de Código 7 foram implementadas para os modelos MLPR e RFR, repetivamente.

```
def doGridSearch_MLPR(self, model, x_train, y_train):
  param_grid = {
      'hidden_layer_sizes': [(100,100), (120,80), (100,50)],
      'max_iter': [50, 100, 2000],'activation': ['tanh', 'relu'],
      'solver': ['sgd', 'adam', 'lbfgs'],'alpha': [0.0001, 0.05],
      'learning_rate': ['constant','adaptive'],
  }
  grid_search = GridSearchCV(estimator = model, param_grid = param_grid,
                             cv = 3, n_jobs = -1, verbose = 2)
  grid_search.fit(x_train, y_train)
  return grid_search.best_params_ , grid_search.best_estimator_
```

Trecho de Código 6 – Definição dos parâmetros modelo MLPR

```
def doGridSearch_RF(self, model, x_train, y_train):
  param_grid = {
      'max_depth': [1, 2, 3, 4],
      'n_estimators': [100, 150, 200, 250, 300],
  }
  grid_search = GridSearchCV(estimator = model, param_grid = param_grid,
                             cv = 3, n_jobs = -1, verbose = 2)
  grid_search.fit(x_train, y_train)
  return grid_search.best_params_ , grid_search.best_estimator_
```

Trecho de Código 7 – Definição dos parâmetros modelo RFR

[47] *GridSearchCV* – https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html, último acesso em 15 de setembro de 2022. [Online]

Ambas as funções recebem os dados do conjunto de treino e retornam a melhor configuração obtida após o treino. Após este resultado, foi realizado ajuste manual tendo como base os resultados do *gridshearch* e realizadas duas previsões, para um único *step* (secção 4.5), sendo uma previsão para o resultado do *gridshearch* e outro com a configuração manual. A melhor performance foi obtida com os parâmetros ajustados manualmente e que são enumerados a seguir:

Parâmetros para o modelo MLPR:

- ***hidden_layer_sizes***: 2 camadas com 100 neurônios cada;
- ***solver***: lbfgs;
- ***max_iterv:*** 2000;
- ***shuffle***: True;
- ***random_state:*** 20.

Parâmetros para o modelo RFR:

- ***random_state:*** 42;
- ***n_estimators:*** 200.

A partir destes parâmetros foi implementada a instanciação dos modelos MLPR e RFR. Os trechos de código Trecho de Código 8, Trecho de Código 9 e Trecho de Código 10 apresentam o código implementado. No caso do MLPR foi utilizada a implementação *Pipeline*[48] da biblioteca *scikit-learn* de forma a adicionar um mecanismo de estandardização dos valores dos registos.

```
rf = RandomForestRegressor(random_state=42, n_estimators=200)
```

Trecho de Código 8 – Instanciar o modelo RFR

```
mlpr = MLPRegressor(hidden_layer_sizes=(100, 100), solver='lbfgs',
max_iter=2000, shuffle=True, random_state=20);
```

Trecho de Código 9 – Instanciar o modelo MLPR

```
def _make_pipeline(self, model):
  steps = list()
  steps.append(('standardize', StandardScaler()))

  steps.append(('model', model))
  pipeline = Pipeline(steps=steps)
  return pipeline
```

Trecho de Código 10 – Função para o *pipeline*

Os trechos de código Trecho de Código 9 e Trecho de Código 10 ilustram a instanciação dos modelos RFR e MLPR respetivamente. No Trecho de Código 10 a função *_make_pipeline()*

[48] *Pipeline* – https://scikit-learn.org/stable/modules/compose.html#, último acesso em 15 de setembro de 2022. [Online]

recebe a instância do modelo MLPR e cria um objeto *Pipeline* para a execução deste modelo com o mecanismo de estandardização dos valores utilizando o *StandardScaler*[49]. Esta função foi implementada no sentido de avaliar a performance do modelo MLPR em dois cenários: a) com o processo de estandardização; b) sem este processo. Uma vez verificado que este apresenta melhor performance no cenário a), então foi assumida esta implementação.

5.3.2 Avaliação do Modelo de *Machine Learning*

Durante a execução do *multi-step-ahead* (secção 4.4) será realizada a avaliação de cada previsão do registo do conjunto de dados de teste utilizando a métrica RMSE. No fim é calculado o RMSE e o R^2 (secção 4.6) com todos os valores reais e os previstos. O Trecho de Código 11 contém a função utilizada para o cálculo do RMSE e o Trecho de Código 12 contém a função para o cálculo do R^2.

```
def evaluate_rmse(self, actual, predicted):
    mse = mean_squared_error(actual, predicted)
    rmse = sqrt(round(mse, 2))
    rmse = round(rmse, 2)

    return rmse
```

Trecho de Código 11 – Função de cálculo do RMSE

```
def evaluate_r2(self, actual, predicted):
    r2 = None
    if len(actual) >= 2:
        r2 = r2_score(actual, predicted)
    else:
        print("Não é possível calcular R`2")
        return r2

    r2 = round(r2, 2)
    return r2
```

Trecho de Código 12 – Função de cálculo do R^2

A função *evaluate_rmse()* recebe o valor real e o valor previsto. Primeiro é realizado o cálculo do RMSE, que tem por base o cálculo do MSE. A partir do resultado do segundo é aplicada a raiz quadrada para obter o RMSE (secção 4.6). A função *evaluate_r2()* recebe uma lista com os valores atuais e outra lista com os valores previstos para cada valor da lista de valores atuais. A razão pela qual é feita a validação do tamanho da lista de nome *atual*, deve-se ao facto de que o cálculo do R^2 precisa de pelos menos dois registos, decorrente da sua definição formal (secção 4.6).

[49] *StandardScaler* – https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html, último acesso em 15 de setembro de 2022. [Online]

5.4 Métodos Explicativos

5.4.1 LIME

Os trechos de código Trecho de Código 13 e Trecho de Código 14 ilustram a instanciação e a execução, respetivamente, do explicador do LIME para o formato tabular. A seguir é feita uma descrição de como os respetivos parâmetros são abastecidos.

```
training_data = X_train
explainer = lime_tabular.LimeTabularExplainer(training_data=training_data,
                                    mode="regression",
                                    feature_names=column_names,
                                    verbose=True)
```

Trecho de Código 13 – Instanciar o explicador do LIME

A seguir é feita uma descrição dos parâmetros de instanciação do explicador do LIME. Para os restantes parâmetros foram utilizados os valores definidos como padrão e assumidos por omissão.

- ***training_data***: todo o conjunto de dados de treino de cada interação do *multi-step-ahead*. Este parâmetro recebe os dados de treino, i.e., os mesmos dados que foram utilizados no treino do modelo ML;
- ***mode***: indica que se trata de uma regressão;
- ***feature_names***: nomes dos atributos independentes utilizados no treino do modelo de ML. Esta lista de atributos será utilizada para gerar os nomes dos atributos nos gráficos do LIME;
- ***verbose***: Permite exibir, em log ou na consola, os resultados da execução do LIME.

Uma vez instanciado o explicador (variável *explainer*) é possível utilizar a função de execução dos cálculos do LIME. O Trecho de Código 14 seguinte apresenta a chamada da função que executa o explicar LIME:

```
lime_explications =
explainer.explain_instance(data_row=instance_to_explain,
                           predict_fn=model.predict,
                           num_samples=10000)
```

Trecho de Código 14 – Função de execução dos cálculos do LIME

A seguir é feita uma descrição dos parâmetros da função de execução das explicações do explicador do LIME:

- ***data_row***: valores para explicar a instância de interesse;
- ***predict_fn***: recebe a função do método de ML utilizado;
- ***num_samples***: total de exemplares que o LIME irá gerar aleatoriamente à volta do valor previsto de forma a aplicar um modelo de regressão para realizar a sua própria previsão.

É nesta aleatoriedade que reside um dos problemas do LIME, pois não há garantias de que, a cada nova iteração, seja gerada a mesma explicação para os pontos aleatórios gerados.

A função *save_lime_explications_as_html()* apresentada no Trecho de Código 15 é utilizada para gravar, em disco, a explicação visual em formato HTML gerada pelo LIME para interpretabilidade local.

```
def save_lime_explications_as_html(self, path_name=LIME_PLOTS_FOLDER_DFT,
file_name=LIME_PLOTS_FILE_NAME_DFT):
    lime_explanations = self.precise_lime_explications
    hasPath = os.path.isdir(path_name)
    if path_name != None and hasPath == False:
        try:
            os.makedirs(path_name)
            hasPath = True
        except:
            print("Erro ao tentar criar a estrutura ", path_name)

    file_name_html = file_name + ".html"
    full_file_path_name = path_name + "/" + file_name_html
    if hasPath:
        print(f"Gravar lime plot {file_name_html} in path = {path_name}")
        lime_explanations.save_to_file(full_file_path_name)
    else:
        print(f"Erro gravar o ficheiro {file_name_html} em {path_name}")
```

Trecho de Código 15 – Função para gravar gráfico LIME

Esta função recebe uma localização e um nome para o ficheiro, sendo que para cada um destes parâmetros assume um valor por padrão. A variável *lime_explanations*, é a estrutura de dados retornada pela função *explain_instance()* do LIME e que contém os resultados dos cálculos. Esta disponibiliza a função *save_to_file()*[50] para a gravação dos resultados.

5.4.2 SHAP

Os trechos de código Trecho de Código 16 e Trecho de Código 17 ilustram a instanciação e a execução, respetivamente, do explicador do SHAP que implementa o conceito de *Kernel SHAP* que é do tipo modelo agnóstico.

```
backgroud_dataset = X_train
explainer = shap.KernelExplainer(model.predict, backgroud_dataset[:100])
```

Trecho de Código 16 – Instanciar o explicador do SHAP

[50] *save_to_file()* – https://lime-ml.readthedocs.io/en/latest/lime.html#lime.explanation.Explanation.save_to_file

A seguir é feita uma descrição dos parâmetros de instanciação do explicador do SHAP.

- ***model***: a função de previsão do método de ML utilizado.
- ***data***: recebe o *background dataset*.

A documentação disponível não é clara acerca da existência de uma fórmula, ou de um método, de como determinar o total de registos do *background dataset* (secção 3.5). Neste estudo foi definido como sendo os primeiros 100 registos do conjunto de dados de treino por apresentar rapidez no processamento nas primeiras experiências exploratórias do SHAP. Para os restantes parâmetros foram utilizados os valores definidos como padrão e assumidos por omissão. O código seguinte apresenta a chamada da função que calcula o *shapley value*. Esta recebe a variável *instance_to_explain* que contém os valores para explicar a instância de interesse, ou seja, os valores de um registo do conjunto de dados de teste.

```
shapley_values = explainer.shap_values(instance_to_explain)
```

Trecho de Código 17 – Função de execução dos cálculos do SHAP

A função *save_force_plot_as_html()* apresentada no Trecho de Código 18 e a função *shap.force_plot()* do Trecho de Código 19 são utilizadas para gravar em disco o gráfico de força do SHAP utilizado para interpretabilidade local.

```
def save_force_plot_as_html(self, local_instace_pos,
        path_name=SHAP_PLOTS_FOLDER_DFT,
        file_name="force_plot"):
  pe = self.precise_explanations
  hasPath = os.path.isdir(path_name)
  if path_name != None and hasPath == False:
    try:
        os.makedirs(path_name)
        hasPath = True
    except:
        print("Erro ao criar a estrutura ", path_name)

 file_name_html = file_name + ".html"
 if hasPath:
    f = shap.force_plot(base_value=pe.get_base_value(),
      shap_values=pe.get_local_shapley_value(local_instace_pos),
     features=pe.get_local_instance_2_explain_as_numpy(local_instace_pos),
      feature_names=pe.get_instances_to_explain().columns.to_list(),
      show=False)  # para gerar HTML

    print(f"Gravar force plot {file_name_html} em = {path_name}")
    full_file_path_name = path_name + "/" + file_name_html
    shap.save_html(full_file_path_name, f)
 else:
    print(f"Erro ao gravar o ficheiro {file_name_html} em {path_name}")
```

Trecho de Código 18 – Função para gravar o gráfico de força

A função *save_force_plot_as_html()* recebe uma localização e um nome para o ficheiro, sendo que para cada um destes parâmetros assume um valor por padrão. A variável *pe* referencia uma estrutura de dados criada para armazenar os resultados dos cálculos do SHAP. Ao contrário do LIME, este explicador do SHAP não retorna uma estrutura própria com toda a informação. A função *force_plot()*, da biblioteca do SHAP, retorna uma referência para um objeto que é utilizado para a gravação do gráfico, como se pode verificar no Trecho de Código 19.

```
f = shap.force_plot(base_value=pe.get_base_value(),
      shap_values=pe.get_local_shapley_value(local_instace_pos),
      features=pe.get_local_instance_2_explain_as_numpy(local_instace_pos),
      feature_names=pe.get_instances_to_explain().columns.to_list(),
      show=False)
```

Trecho de Código 19 – Objeto referência para gravar o gráfico de força

A função *save_html()* grava o referido gráfico em formato HTML. Não foi encontrada na biblioteca do SHAP ou por outra via, uma forma imediata de gravar o gráfico de força *force* em outro formato que não seja HTML. De referir, ainda, uma dificuldade na implementação desta função. Não está explícita na documentação da biblioteca do SHAP a definição da função *save_html()*. Esta foi identificada a partir de exemplos de fóruns (*Saving SHAP plots*)[51] dedicados ao tema disponíveis no *github* e mantido pelos autores do SHAP.

A função *save_summary_plot()* apresentada no Trecho de Código 20 é utilizada para gravar em disco o gráfico de sumário do SHAP utilizado para interpretabilidade global.

[51] *Saving SHAP plots* – https://github.com/slundberg/shap/issues/153, último acesso em 15 de janeiro de 2022. [Online]

```
def save_summary_plot(self, plot_type=None,
                      path_name=SHAP_PLOTS_FOLDER_DFT,
                      file_name="summary_plot",format="jpg"):
    pe = self.precise_explanations
    hasPath = os.path.isdir(path_name)
    if path_name != None and hasPath == False:
        try:
            os.makedirs(path_name)
            hasPath = True
        except:
            print("Erro criar a estrutura ", path_name)
    if format == "pdf":
        full_path_name = path_name + "/" + file_name + ".pdf"
    elif format == "jpg":
        full_path_name = path_name + "/" + file_name + ".jpg"
    if hasPath:
        plt.ioff()
        if plot_type == None:
            shap.summary_plot(pe.get_shapley_values(),
                              pe.get_instances_to_explain(),
                              show=False)
        else:
            shap.summary_plot(pe.get_shapley_values(),
                              pe.get_instances_to_explain(),
                              plot_type=plot_type, show=False)
        plt.savefig(full_path_name, format=format, dpi=150,
                    bbox_inches='tight')
        plt.close()
    else:
        print(f"Erro gravar o ficheiro {file_name} em {path_name}")
```

Trecho de Código 20 – Função para gravar o gráfico sumário

A função save_summary_plot*()* recebe uma localização e um nome para o ficheiro, sendo que para cada um destes parâmetros assume um valor por padrão. Esta função recorre à função *summary_plot()* da biblioteca do SHAP para gerar o gráfico sumário. Notar que foi preciso recorrer à função *savefig()*[52] da biblioteca Matploblib, permite gravar este gráfico em disco. Portanto, a implementação original da biblioteca SHAP, para este gráfico, não suporta a gravação em disco, como acontece na implementação do gráfico de força.

5.5 Considerações Finais

Apesar de o software desenvolvido ser dedicado os casos de estudo deste trabalho, este pode evoluir para uma generalização que permita integração em um processo de ML para a geração de explicações.

As funções disponíveis nas bibliotecas do LIME e do SHAP podem ser integradas em mecanismos para persistência destes, como por exemplo, gravar em disco. Em alguns casos as funções já apresentam suporte para este efeito, como é o caso das funções

[52] *savefig()* – https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.savefig.html, último acesso em 15 de janeiro de 2022. [Online]

lime_explainer.*save_to_file()* e *shap.save_html()* do LIME e do SHAP, respetivamente. Desta forma, os gráficos podem ser reaproveitados para integração em outras aplicações para efeito de análise de histórico das explicações. O facto de ser possível extrair os valores calculados pelo LIME e pelo SHAP também permite que estes sejam gravados para reutilização futura.

O capítulo seguinte é dedicado aos casos de estudo. Este apresenta quatro casos sendo os dois primeiros dedicados às explicações geradas pelo LIME e pelo SHAP e os dois últimos dedicados à performance de execução. Estes casos de estudos utilizam um conjunto de dados reais.

6 Casos de Estudo

6.1 Introdução

As secções deste capítulo apresentam os casos de estudo elaborados neste trabalho. O objetivo principal é a aplicação prática dos métodos explicativos LIME e SHAP para gerar explicações de previsões de consumos de energia elétrica dos modelos RFR e ANN, identificado por MLPR (secção 4.6), e utilizando séries temporais.

Tabela 12 – Resumo dos casos de estudo

Caso de estudo	Descrição
1	Para este caso de estudo pretende-se gerar explicações locais e globais de previsões de consumos de eletricidade recorrendo aos métodos LIME e SHAP e ainda analisar possíveis interpretações dos modelos de ML. O conjunto de dados utilizado contém 2976 registos com intervalos de 15 minutos do mês de janeiro e um total de 20 atributos para explicação.
2	Pretende-se avaliar as alterações verificadas nos resultados do LIME e do SHAP quando são retirados os atributos que apresentam registos com o valor zero. Este caso de estudo difere do primeiro pelo facto de utilizar apenas 8 atributos para explicação.
3	Um aspeto importante é o tempo de processamento necessário para gerar explicações. Este caso de estudo avalia o tempo de processamento para o conjunto de dados utilizado nos casos de estudo 1 e 2. Será avaliado o tempo de processamento em função do total de atributos do conjunto de dados e do modelo de ML.

4	Este último caso avalia o tempo de processamento do SHAP em função da variação do total de registos do *background dataset.* São considerados três cenários: a) 500 registos; b) 1000 registos; c) 1500 registos. Os resultados destes três cenários serão comparados com o caso de estudo 3 que utiliza apenas 100 registos para o *background dataset.*

6.2 Caso de Estudo 1

6.2.1 Problema

A investigação da utilização de XAI na área de sistemas de energia ainda é pouco expressiva (secção 2.2.1). Inclusive, há pouca aplicação de XAI em conjuntos de dados no formato de séries temporais (secção 2.2.4). Neste sentido, este caso de uso tem como objetivo:

1. Gerar explicações locais e globais considerando explicadores do tipo modelo agnóstico;
2. Utilizar o explicador *shap.KernelExplainer* da biblioteca SHAP;
3. Utilizar o explicador *lime.lime_tabular.LimeTabularExplainer*, da biblioteca LIME;
4. Analisar as possíveis interpretações dos resultados apresentados pela ferramenta de explicação em formato HTML do LIME para interpertabilidade local;
5. Analisar as possíveis interpretações dos resultados apresentados pelas seguintes ferramentas do SHAP: a) gráfico de força para interpertabilidade local; b) gráfico sumário para interpretabilidade global.

Foi utilizado o conjunto de dados gecad_competion_2019_B.csv o qual foi dividido em dados de treino e teste. Este último conjunto é composto por dez registos com início nas 21h30m e fim nas 23h45m. Para cada um destes registos será gerada uma explicação do valor de consumo previsto.

6.2.2 Execução

Neste caso de estudo serão utilizados os vinte atributos identificados (secção 5.2.4). Como referido no capítulo 3, dedicado às descrições conceituais do LIME e do SHAP, a geração de explicações não implica que sejam realizadas previsões com base no modelo de ML. Contudo, para efeito de estudo, optou-se por realizar as previsões dos modelos de forma a comparar com os resultados das bibliotecas do LIME e do SHAP e obter os valores para as métricas definidas (secção 4.6).

6.2.3 Resultados e Discussão

A Tabela 13 apresenta os resultados da performance dos modelos MLPR e RFR. A Tabela 14 contém os valores previstos por cada um destes modelos para os dez registos do conjunto de dados de teste.

Tabela 13 – Métricas dos modelos MLPR e RFR

Modelo	RMSE (W)	R^2
MLPR	128.2	0.96
RFR	206.68	0.85

Tabela 14 – Previsões dos consumos dos modelos MLPR e RFR

Time	Real (W)	MLPR (W)	RFR (W)
21:30:00	4129	4151.92	4381.34
21:45:00	2758	2592.83	3055.61
22:00:00	2686	2802.23	2959.7
22:15:00	3332	3214.66	3253.81
22:30:00	3279	3250.8	3167.06
22:45:00	4142	4269.65	3947.28
23:00:00	4576	4554.79	4607.69
23:15:00	3516	3571.92	3697.6
23:30:00	3273	3197.09	3576.24
23:45:00	3377	3664.83	3244.6

Como se pode verificar na Tabela 13, ambos os modelos apresentam bons valores. Por exemplo, para a métrica R^2 o valor do MLPR é 0.96 e o valor do RFR é 0.85. Portanto, o MLPR é o modelo que apresenta melhor performance. Na Tabela 14 pode-se verificar que os valores previstos pelos modelos estão muito próximos dos valores reais em alguns pontos, como por exemplo no instante 23h00m (Real = 4576W; MLPR = 4554.79W; RFR = 4607.69W). A Figura 38 apresenta um gráfico que permite uma comparação visual entre os valores previstos pelos dois modelos e os valores reais do conjunto de dados de teste.

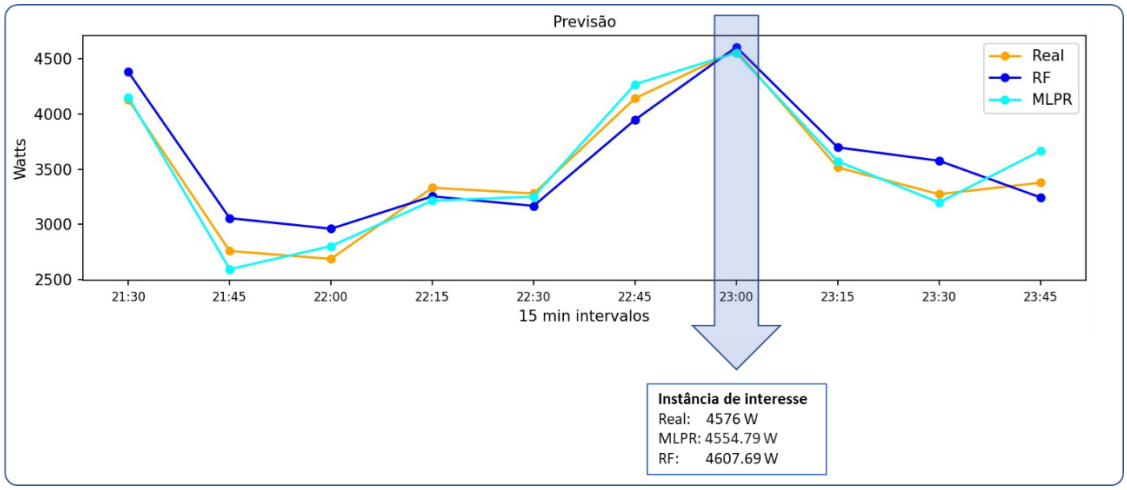


Figura 38 – Gráfico das previsões de consumos dos modelos MLPR e RFR

Neste gráfico é notório o quanto o valor previsto pelos dois modelos, para a instância 23h00m, está próximo do valor real. Outro aspeto, é a proximidade dos valores previstos pelo modelo MLPR em relação aos valores reais, o que está de acordo com os elevados valores das métricas.

Para esta análise será considerado, como instância de interesse, o registo das 23h00m, cujas previsões apresentam um valor muito próximo do valor real. O método de previsão consiste em prever o consumo utilizando *multi-step-ahead* (secção 4.4) com um horizonte de um instante para o futuro em que cada instante corresponde ao consumo em intervalos de 15 minutos. Assim, a previsão do consumo das 23h00m foi realizada com os valores dos atributos dos 15 minutos anteriores, ou seja, das 22h45m. Portanto, são os valores do instante de tempo 22h45m que são utilizados pelo LIME e pelo SHAP para gerar explicações para a instância de interesse das 23h00m. A Figura 39 apresenta os valores do instante 22h45m utilizados para gerar as explicações da previsão do instante seguinte:

	2970
date_time	2019-01-31 22:45:00
z1_ha1_w	212
z1_ha2_W	0
z1_ha3_w	0
z1_li_w	0
z1_skts_w	563
z2_ha1_w	0
z2_ha2_w	1080
z2_li_w	0
z2_skts_w	916
z3_ha1_w	0

	2970
date_time	2019-01-31 22:45:00
z3_ha2_w	0
z3_ha3_w	0
z3_li_w	0
z3_skts_w	366
z4_ha_w	525
z4_li_w	0
z4_skts_w	163
z5_ha_w	0
z5_li_w	0
z5_skts_w	317

Figura 39 – Detalhe dos valores para explicar a instância de interesse

6.2.3.1 Interpretabilidade Local

Na Figura 40 e na Figura 41, apresentam-se o gráfico formato HTML do LIME e o gráfico de força do SHAP para a interpretação local da previsão dos modelos MLPR e RFR, respetivamente, na instância de interesse 23h00m.

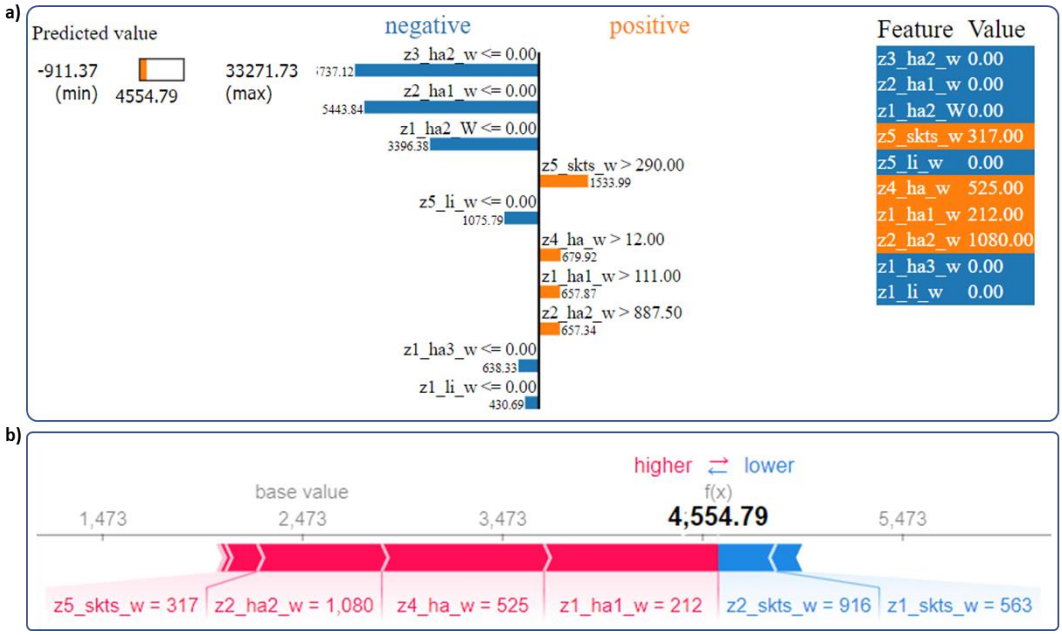


Figura 40 – Interpretação local da previsão do MLPR para as 23h00m com (a) LIME e (b) SHAP

Tanto o LIME como o SHAP apresentam a previsão de valor **4554.79W**, igual ao valor previsto pelo modelo de MLPR e apresentado na Tabela 14. Com destaque para os três primeiros atributos classificados como os mais importantes para a previsão do modelo, o LIME identificou os atributos z3_ha2_w, z2_ha1_w e z1_ha2_w. Por sua vez, o SHAP identificou os atributos z1_ha1_w, z4_ha_w e z2_ha2_w. Os atributos mais importantes identificados pelo LIME contribuem de forma negativa, enquanto os atributos identificados pelo SHAP contribuem positivamente. Portanto, verifica-se uma significativa diferença nos resultados dos dois métodos.

No que refere aos atributos que contribuem positivamente, para o LIME, o atributo z5_skts_w (tomadas da zona 5) é aquele que mais contribui, enquanto para o SHAP este mesmo atributo é o que menos contribui. Para o SHAP, o atributo z1_ha1_w é o que mais contribui positivamente. Ainda quanto à contribuição positiva, o dispositivo ar condicionado é predominante, tanto no LIME como no SHAP, e identificado pelos atributos z1_ha1_w, z4_ha_w, z2_ha2_w com diferente ordem de importância entre o LIME e o SHAP. Portanto, verifica-se que LIME e o SHAP identificaram os mesmos atributos, porém com diferente ordem de importância e há predominância dos atributos relativos ao dispositivo ar condicionado.

Quanto à contribuição negativa, o LIME identificou seis atributos enquanto o SHAP apenas dois. O LIME identificou os atributos z5_li_w e z1_li_w, os quais não foram identificados pelo SHAP.

No caso do SHAP, estão presentes apenas os atributos z2_skts_w e z1_skts_1, que por sua vez não foram identificados pelo LIME na contribuição negativa. Portanto, os métodos explicativos apresentam uma significativa diferença na interpretação local no que refere à contribuição negativa.

Por fim, de acordo com o LIME, há seis atributos, que apesar de terem o valor zero, estão entre os dez atributos que mais contribuem para o valor previsto pelo modelo, porém com contribuição negativa. O SHAP não identificou atributos com o valor zero. No LIME verifica-se que tanto o gráfico central como a tabela à direita apresentam apenas 10 atributos, apesar de ter sido considerado um total de 20. Isto deve-se pelo facto de ser o valor padrão definido pelo LIME.

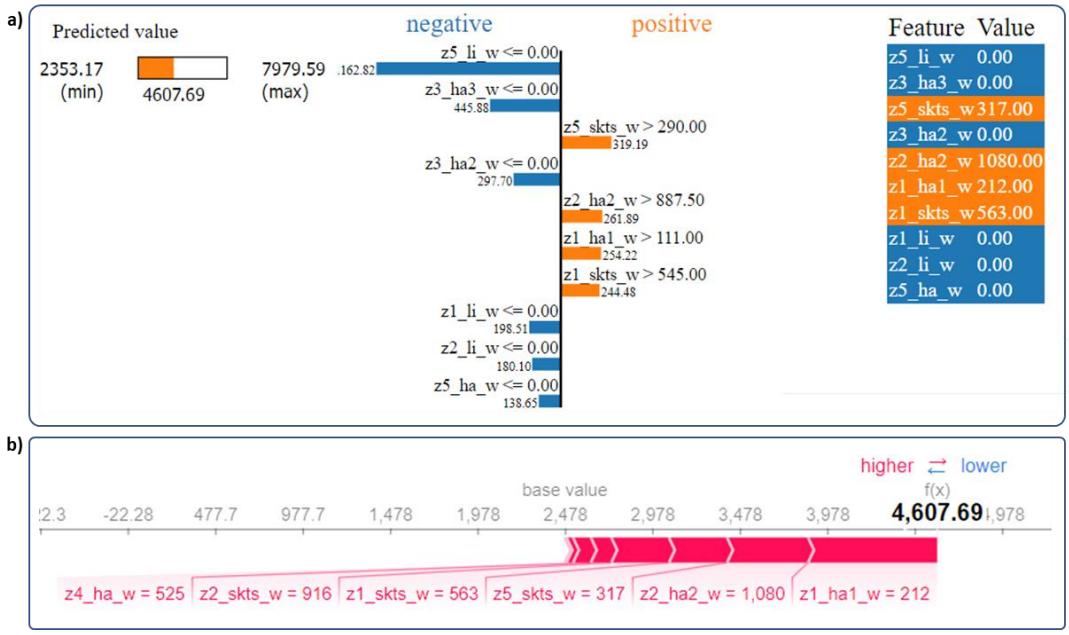


Figura 41 – Interpretação local da previsão do RFR para as 23h00m com (a) LIME e (b) SHAP

No caso do RFR, tanto o LIME como o SHAP apresentam a previsão de valor **4607.69W**, igual ao valor previsto pelo modelo e apresentado na Tabela 14. Considerando apenas os três atributos mais importantes, o LIME identificou os atributos z5_li_w, z3_ha3_w e z5_skts_w. Por sua vez, o SHAP identificou os atributos z1_ha1_w, z2_ha2_w e z5_skts_w. Para o LIME, os dois atributos mais importantes contribuem de forma negativa e o terceiro apresenta uma contribuição positiva, enquanto todo os atributos identificados pelo SHAP contribuem positivamente.

Quanto aos atributos que contribuem positivamente, o SHAP identificou seis atributos e o LIME apenas quatro. Os atributos identificados pelo LIME, z5_skts_w, z2_ha2_w, z1_ha1_w e z1_skts_w, estão presentes no conjunto de atributos identificados pelo SHAP, porém em diferente ordem de importância. O SHAP identificou ainda os atributos z4_ha_w e z2_skts_w. Verifica-se, portanto, uma aproximação entre o LIME e o SHAP na identificação dos atributos quanto à contribuição positiva.

Relativamente à contribuição negativa, o LIME identificou seis atributos z5_li_w, z3_ha3_w, z3_ha2_w, z1_li_w, z2_li_w e z5_ha_w. Para o SHAP, não existem atributos que contribuam negativamente para o valor da previsão. É, portanto, significativa diferença na interpretação local entre o LIME e o SHAP para a contribuição negativa.

Por fim, e por comparação com a Figura 40, o LIME identificou seis atributos com o valor zero, enquanto o SHAP não identificou atributos com o valor zero seja para contribuição positiva ou negativa.

Uma das desvantagens do LIME está no facto de que, a cada execução, este pode apresentar diferentes resultados para uma mesma previsão do modelo de ML (secção 3.3.3). A Figura 42, a Figura 43 e a Figura 44 apresentam os resultados de três execuções do LIME para a instância de interesse 23h00m do modelo MLPR.

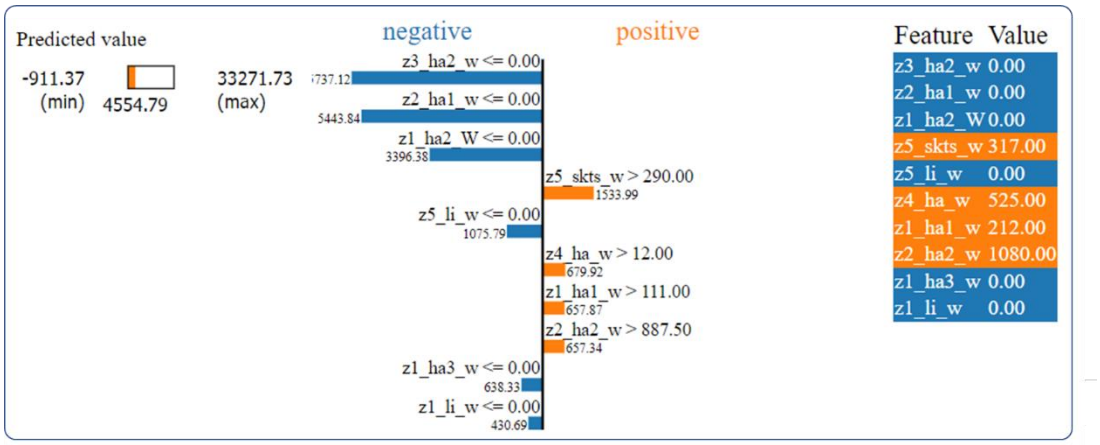


Figura 42 – Interpretação local do LIME para modelo MLPR. 1ª Execução

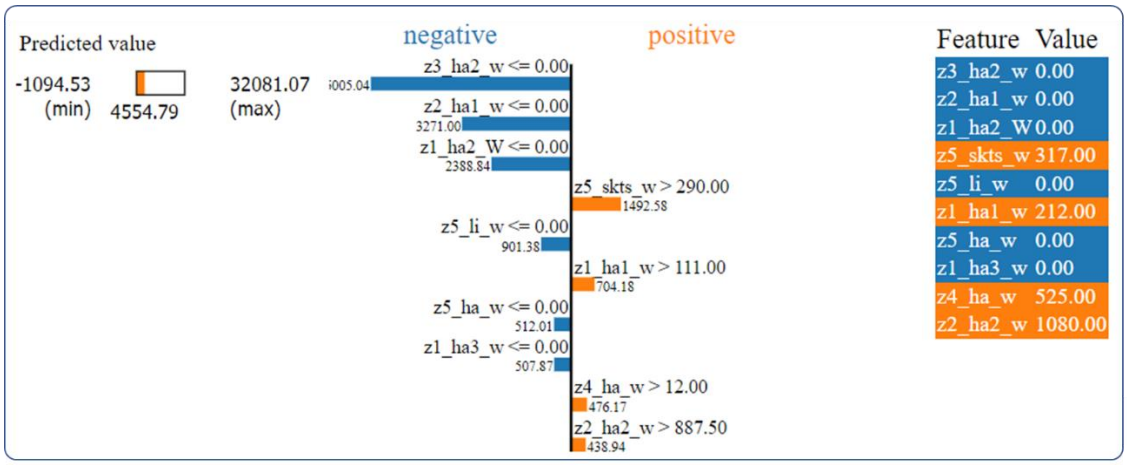


Figura 43 – Interpretação local do LIME para modelo MLPR. 2ª Execução

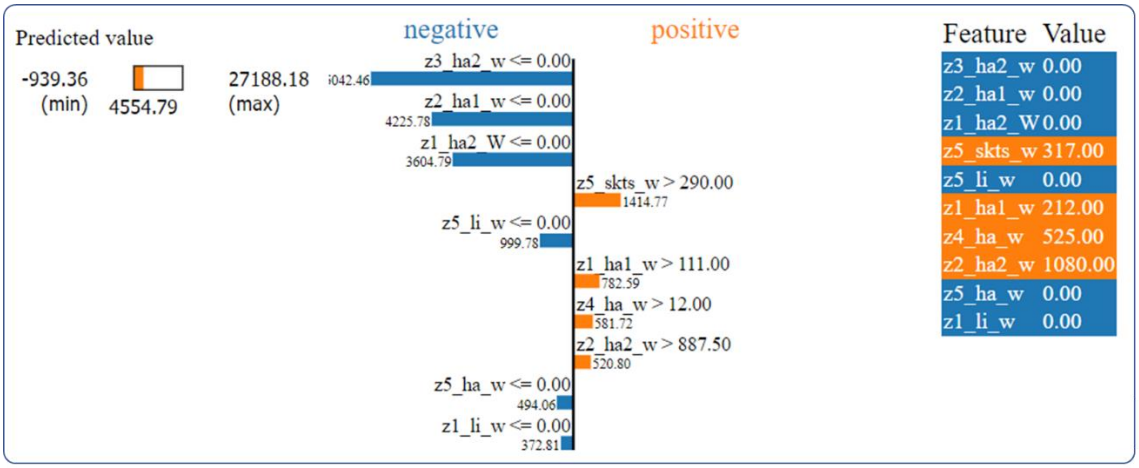


Figura 44 – Interpretação local do LIME para modelo MLPR. 3ª Execução

Por observação das figuras anteriores, verifica-se que não há alteração na ordem de importância entre os cinco primeiros atributos, ao contrário dos restantes atributos, apesar de haver diferenças nos valores dos coeficientes. Por exemplo, para o atributo z2_ha1_w os valores foram 5443.84, 3271.0 e 4225.78, para a primeira, segunda e terceira execuções, respetivamente. Nota-se que os atributos cuja ordem de importância variou são aqueles que apresentam valores dos coeficientes muito próximos como é o caso dos atributos z4_ha_w e z1_ha1_w.

6.2.3.2 Interpretabilidade Global

A interpretabilidade global do SHAP se baseia na agregação dos valores do *shapley value* de cada atributo das instâncias de interesse (secção 3.5). Ora, neste estudo, as instâncias de interesse são cada um dos dez registos do conjunto de dados de teste e que correspondem aos instantes entre as 21h30m e as 23h45m. O gráfico sumário dos modelos MLPR e RFR é apresentado na Figura 45 e na Figura 48, respetivamente.

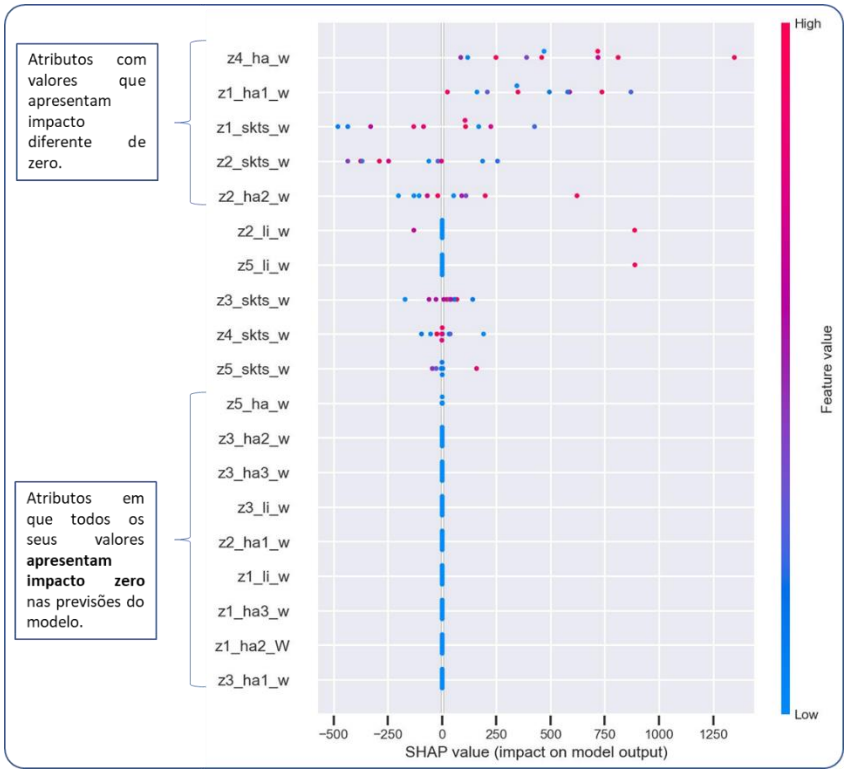


Figura 45 – Interpretação global do SHAP para o modelo MLPR

O primeiro aspeto que é realçado neste gráfico refere-se aos nove atributos em que todos os valores apresentam um impacto zero na previsão: z5_ha_w, z3_ha2_w, z3_ha3_w, z3_li_w, z2_ha1_w, z1_li_w, z1_ha3_w, z1_ha2_w e z3_ha1_w. Na listagem da ordem de importância do atributo, estes surgem no fim da lista o que indica que, para o SHAP, estes são os atributos que menos contribuem para a comportamento global da previsão do modelo de ML.

No caso dos cinco primeiros atributos que mais contribuem a previsão, há uma quantidade residual de valores com impacto zero: z4_ha_w, z1_ha1_w, z1_skts_w e z2_ha2_w. Estes cinco atributos referem-se aos dispositivos ar condicionado e tomadas. No caso do atributo z4_ha_w todos os valores apresentam impacto positivo, pois estão à direita do valor zero do eixo horizontal. Os quatro valores do atributo z4_ha_2, mais à direita, com maior *shapley value* e de cor vermelha, indicam que o valor alto do consumo tem forte impacto na previsão do modelo. No caso do atributo z2_ha2_w, a maior parte dos seus valores reais está próxima do valor de impacto zero. Isto decorre do facto de apresentarem baixo valor para o *shapley value*, e indica pouco impacto na previsão.

A seguir, a Figura 46 apresenta os valores reais dos nove atributos do conjunto de dados de teste identificados como os que apresentam impacto nulo. A Figura 47 apresenta os valores reais dos cinco primeiros atributos, i.e., aqueles com maior importância, e que apresentam valores com impacto diferente de nulo.

	date_time	z5_ha_w	z3_ha2_w	z3_ha3_w	z3_li_w	z2_ha1_w	z1_li_w	z1_ha3_w	z1_ha2_W	z3_ha1_w
2965	2019-01-31 21:30:00	0	0	0	0	0	0	0	0	0
2966	2019-01-31 21:45:00	0	0	0	0	0	0	0	0	0
2967	2019-01-31 22:00:00	0	0	0	0	0	0	0	0	0
2968	2019-01-31 22:15:00	0	0	0	0	0	0	0	0	0
2969	2019-01-31 22:30:00	0	0	0	0	0	0	0	0	0
2970	2019-01-31 22:45:00	0	0	0	0	0	0	0	0	0
2971	2019-01-31 23:00:00	0	0	0	0	0	0	0	0	0
2972	2019-01-31 23:15:00	0	0	0	0	0	0	0	0	0
2973	2019-01-31 23:30:00	0	0	0	0	0	0	0	0	0
2974	2019-01-31 23:45:00	0	0	0	0	0	0	0	0	0

Figura 46 – Valores dos atributos com impacto nulo

	date_time	z4_ha_w	z1_ha1_w	z1_skts_w	z2_skts_w	z2_ha2_w
2965	2019-01-31 21:30:00	318	620	554	910	852
2966	2019-01-31 21:45:00	119	239	550	911	172
2967	2019-01-31 22:00:00	100	188	563	901	170
2968	2019-01-31 22:15:00	274	672	552	895	174
2969	2019-01-31 22:30:00	380	137	561	897	548
2970	2019-01-31 22:45:00	525	212	563	916	1080
2971	2019-01-31 23:00:00	520	653	562	901	1140
2972	2019-01-31 23:15:00	522	44	566	900	702
2973	2019-01-31 23:30:00	513	393	559	904	169
2974	2019-01-31 23:45:00	427	568	562	920	169

Figura 47 – Valores dos atributos com impacto diferente de nulo

Como se pode verificar na Figura 46, todos os atributos apresentam o valor zero em todos os instantes de tempo. De acordo com o SHAP, estes apresentam uma contribuição nula para a aprendizagem do modelo MLPR. Por outro lado, os atributos apresentam valores diferentes de zero (Figura 47) apresentam contribuição diferente de nulo. A seguir é apresentado o gráfico sumário obtido para o conjunto de dados de teste utilizado neste caso de estudo para o modelo RFR e a apresentado na Figura 48 seguinte:

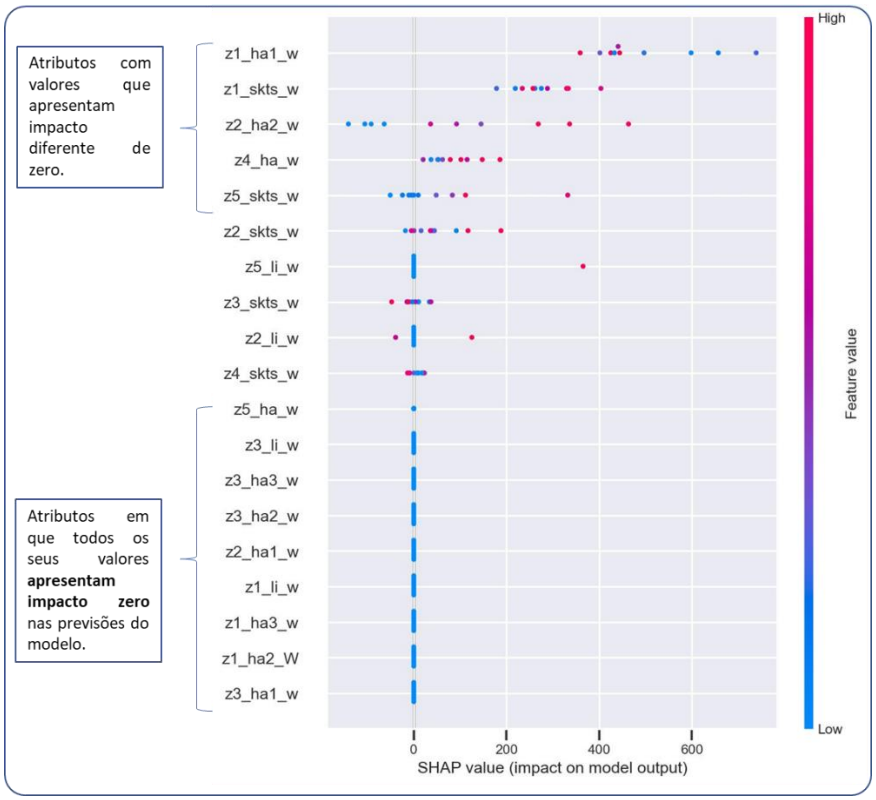


Figura 48 – Interpretação global do SHAP para modelo RFR. Gráfico sumário

Por comparação com a Figura 45, do modelo MLPR, também no RFR verificam-se os mesmos nove atributos que apresentam impacto zero no comportamento global do modelo. Verifica-se uma pequena alteração na ordem de importância, pois os atributos z3_li_w e z3_ha2_w trocam de posição. No caso dos cinco primeiros atributos que mais contribuem a previsão, há uma quantidade residual de valores com impacto zero: z1_ha1_w, z1_skts_w, z2_ha2_w, z4_ha_w e z5_skts.

6.2.3.3 Discussão

Na análise da interpretabilidade local verifica-se que o LIME identificou como atributos de maior importância aqueles com o valor zero, seja para o modelo MLPR ou para o RFR. Inclusive, na análise que considerou três execuções do LIME estes atributos mantém-se entre os mais importantes. No caso do SHAP, estes atributos simplesmente não surgem no gráfico de força. Portanto, para o SHAP, estes não contribuem para pressionar o valor da previsão local seja para um maior ou menor valor em relação ao valor base. Interessante notar que no gráfico sumário, para interpretabilidade global, os atributos com o valor zero são classificados como tendo contribuição nula para o comportamento global do modelo.

Uma hipótese que surge desta constatação é a de que os modelos podem não estar a prever consumo de energia, mas sim, não consumo. Talvez, devido ao facto de existirem muitos mais atributos com o valor zero, em comparação com atributos com valor diferente de zero, o que

provoca um desbalanceamento do conjunto de dados. Portanto, apesar de estes modelos apresentarem elevados valores das métricas de qualidade, MLPR com $R^2 = 0.96$ e RF com $R^2 = 0.85$, esta hipótese sugere que estes modelos não aprenderam a reconhecer os padrões de consumo, mas sim os padrões de não consumo. Esta análise é uma conclusão importante, pois permite identificar e avaliar se o modelo está realmente a agir como o esperado, ou seja, realmente a prever o que se pretende ao invés de estar enviesado por outros acontecimentos.

6.3 Caso de Estudo 2

6.3.1 Problema

No caso de estudo 1 foi possível identificar no gráfico sumário do SHAP um conjunto de atributos com impacto zero na previsão global dos modelos de ML. Nos gráficos de interpretabilidade local do LIME, para a instância de interesse das 23h00m do dia 31/01/2019, verifica-se que três dos atributos que apresentam o valor zero são classificados como tendo uma importância maior do que os atributos com valor diferente de zero. Outro aspeto é o facto de os modelos MLPR e RFR apresentarem elevados valores para as métricas de qualidade o que sugere um bom grau de aprendizagem dos modelos. Assim, com este caso de estudo, pretende-se analisar os resultados do LIME e do SHAP sem considerar os atributos identificados com contribuição nula e identificados no caso de estudo 1. Os objetivos deste caso de estudo são:

1. Avaliar as explicações geradas pelos métodos explicativos SHAP e LIME considerando um conjunto de dados contendo apenas os atributos que, de acordo com o SHAP, não apresentam impacto nulo nas previsões dos modelos MLPR e RFR;
2. Comparar com os resultados do caso de estudo 1.

Neste caso de estudo pretende-se utilizar os mesmos parâmetros de instanciação dos modelos de ML e dos explicadores do caso de estudo 1. Será criado um conjunto de dados novo, semelhante ao gecad_competion_2019_B.csv, porém sem os referidos atributos e que será utilizado para o treino dos modelos de ML e nos explicadores.

6.3.2 Execução

Para identificar os atributos que apresentam valor zero, foi implementada a função *calc_percentagem_zeros() (*Trecho de Código 21*)* para calcular a percentagem de registos com o valor zero em cada um dos vinte atributos do conjunto de dados gecad_competion_2019_B.csv.

```
def calc_percentagem_zeros(column_name, ds_gecad_2019_B):
    count = (ds_gecad_2019_B[column_name] == 0).sum()
    total_linhas = len(ds_gecad_2019_B)
    percentagem_zeros = (count / total_linhas) * 100
    round(percentagem_zeros, 2)
```

Trecho de Código 21 – Função de cálculo da percentagem de valor zero por atributo

Esta função recebe um *dataframe* com o conjunto de dados e o nome do atributo para o qual será realizado o cálculo. O retorno é um valor percentual de registos com o valor zero para o respetivo atributo. Este valor é guardado em um *dataframe*. Para facilitar a apresentação do resultado neste documento, este *dataframe* foi dividido em quatro, cada um contendo cinco atributos como se pode ver na Figura 49.

a)

	z1_ha1_w_perc	z1_ha2_W_perc	z1_ha3_w_perc	z1_li_w_perc	z1_skts_w_perc
0	0.0	99.29	91.5	72.35	0.0

b)

	z2_ha1_w_perc	z2_ha2_w_perc	z2_li_w_perc	z2_skts_w_perc	z3_ha1_w_perc
0	99.73	0.0	80.48	0.0	100.0

c)

	z3_ha2_w_perc	z3_ha3_w_perc	z3_li_w_perc	z3_skts_w_perc	z4_ha_w_perc
0	97.41	65.15	94.15	0.0	0.0

d)

	z4_li_w_perc	z4_skts_w_perc	z5_ha_w_perc	z5_li_w_perc	z5_skts_w_perc
0	81.01	0.0	86.02	74.19	0.0

Figura 49 – Atributos com o valor zero

Existem oito atributos (destacados no retângulo laranja) que não apresentam registos com o valor zero. Os restantes atributos apresentam, no mínimo, cerca de 65% de registos com o valor zero. Sendo assim, serão considerados apenas os oito atributos sem valor zero: z1_ha_w, z1_skts_w, z2_h2_w, z2_skts_w, z3_skts_w, z4_ha_w, z4_skts_w e z5_skts_w.

Um novo conjunto de dados foi criado, a partir do gecad_competion_2019_B.csv, contendo apenas os oito atributos acima identificados. Uma vez que dos restantes atributos, alguns apresentavam registos com valor diferente de zero, foi refeito o cálculo do valor do atributo *consumption_w* considerando apenas estes oito atributos.

6.3.3 Resultados e Discussão

A Tabela 15 apresenta os resultados da performance dos modelos MLPR e RFR nos dois casos de estudo. A Tabela 16 contém os valores previstos por cada um destes modelos para os dez registos do conjunto de dados de teste para o caso de estudo 2.

Tabela 15 – Métricas dos modelos MLPR e RFR

Caso de estudo	Conjunto de dados	Modelo	RMSE (W)	R^2
1	20 atributos	MLPR	128.2	0.96
2	8 atributos	MLPR	152.93	0.92
1	20 atributos	RFR	206.68	0.85
2	8 atributos	RFR	186.44	0.81

Tabela 16 – Previsões dos consumos modelos MLPR e RFR

Time	Real (W)	MLPR (W)	RFR (W)
21:30:00	4019	4038.5	4008.07
21:45:00	2758	3050.05	3065.14
22:00:00	2686	2721.82	2948.24
22:15:00	3332	3269.49	3356.7
22:30:00	3279	3477.2	3295.71
22:45:00	4142	3961.46	4013.98
23:00:00	4576	4718.44	4342.06
23:15:00	3516	3679.81	3754.19
23:30:00	3273	3331.89	3507.38
23:45:00	3377	3520.36	3403.42

O modelo MLPR apresenta diminuição na performance no caso de estudo 2. O R^2 baixou (-0,04) e o RMSE aumentou (+24,73). O RFR apresenta, no caso 2, uma diminuição no valor do R^2(-0,04), o que sugere perda de performance, contudo há uma melhoria no RMSE pois o seu valor diminuiu (-20,02). Em todo o caso, o MLPR continua a ser o modelo que apresenta melhor performance. Na Tabela 16 pode-se verificar que os valores previstos pelos modelos não estão tão próximos dos valores reais em alguns pontos, como por exemplo no instante 23h00m (Real = 4576W; MLPR = 4718.44W; RFR = 4342.06W). A Figura 50 apresenta um gráfico que permite uma comparação visual entre os valores previstos pelos dois modelos e os valores reais do conjunto de dados de teste.

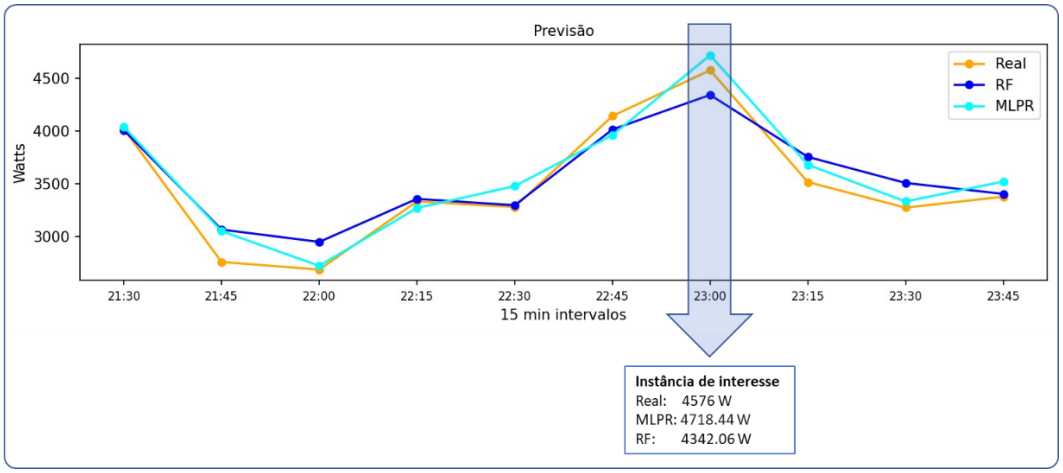


Figura 50 – Gráfico das previsões de consumos dos modelos MLPR e RFR

Por comparação com o caso de estudo 1 (Figura 38), é notório o quanto o valor previsto pelos dois modelos, para a instância 23h00m, deixou de estar tão próximo do valor real. A Figura 51 apresenta os valores do instante 22h45m que foram utilizados para gerar as explicações da previsão do instante seguinte, i.e., 23h00m.

	2970
date_time	2019-01-31 22:45:00
z1_ha1_w	212
z1_skts_w	563
z2_ha2_w	1080
z2_skts_w	916
z3_skts_w	366
z4_ha_w	525
z4_skts_w	163
z5_skts_w	317

Figura 51 – Detalhe dos valores para explicar a instância de interesse

Na figura acima identificam-se os oito atributos definidos para este caso de uso. Não há atributos com o valo zero e, por comparação com a Figura 39, do caso de estudo anterior, os valores dos atributos são exatamente iguais para este registo do conjunto de dados de teste.

6.3.3.1 Interpretabilidade Local

Esta secção contempla, primeiramente, a avaliação das explicações geradas pelos métodos explicativos SHAP e LIME para os modelos MLPR e RFR neste segundo caso de estudo. A seguir, é realizada uma comparação dos resultados dos métodos explicativos obtidos no primeiro e no segundo casos de estudo.

Na Figura 52 e na Figura 53, apresentam-se os resultados dos métodos explicativos para as previsões dos modelos MLPR e RFR, respetivamente. Ambas figuras contêm o gráfico formato HTML do LIME e o gráfico de força do SHAP para a interpretação local da previsão da instância de interesse 23h00m.

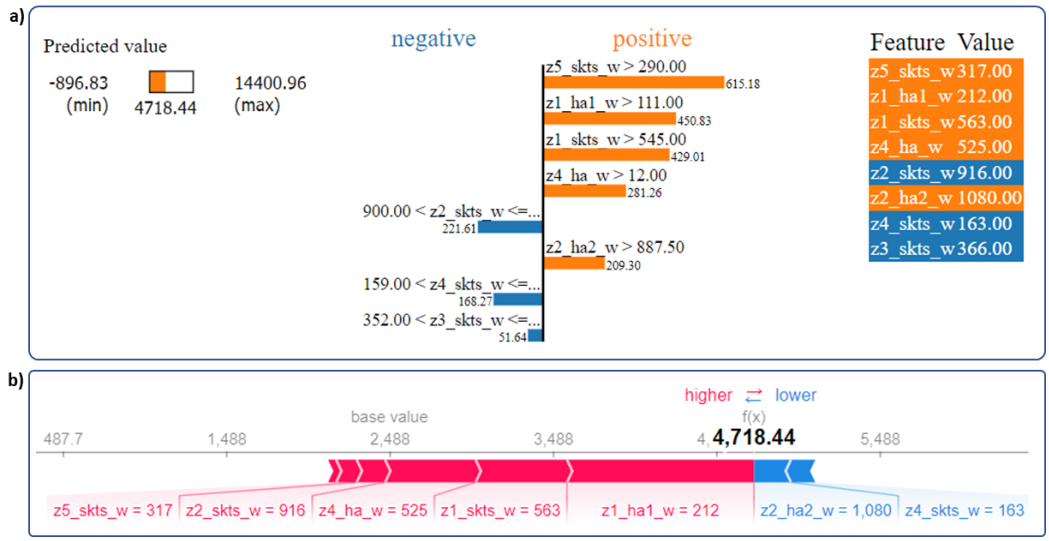


Figura 52 – Interpretação local da previsão do MLPR para as 23h00m com (a) LIME e (b) SHAP

Tanto o LIME como o SHAP apresentam a previsão de valor **4718.44W**, igual ao valor previsto pelo modelo de ML (Tabela 16). Considerando os três primeiros atributos classificados como os mais importantes para a previsão do modelo, o LIME identificou os atributos z5_skts, z1_ha1_w e z1_skts_w. Por sua vez, o SHAP identificou os atributos z1_ha1_w, z1_skts_w e z4_ha_w. Para o LIME e o SHAP, estes atributos contribuem de forma positiva para o valor da previsão do modelo. Os atributos z1_ha1_w e z1_skts_w surgem em ambos os métodos. Portanto, nota-se uma semelhança nos resultados os dois métodos.

Considerando a contribuição positiva, o LIME e o SHAP identificaram cinco atributos. Para o LIME, os atributos são: z5_skts_w, z1_ha1_w, z1_skts_w, z4_ha_w e z2_ha2_w. O SHAP identificou os mesmos cinco atributos, com exceção do atributo z2_ha2_w. O atributo z2_skts_w foi identificado pelo SHAP, mas não pelo LIME. Apesar de existirem muitos atributos em comum, a ordem de importância é diferente entre os dois métodos. Por exemplo, o LIME considera o atributo z5_skts_w como o mais importante deste grupo, enquanto o SHAP considera este mesmo atributo como o menos importante. Portanto, apesar de ambos os métodos terem identificado a mesma quantidade de atributos com contribuição positiva, e de apenas um destes ser diferente, os métodos atribuem uma diferente ordem de importância aos atributos.

Na análise da contribuição negativa, verifica-se que o LIME identificou três atributos: z2_skts_w, z4_skts_w e z3_skts_w. O SHAP identificou apenas dois: z2_ha2_w e z4_skts_w. Ambos os métodos apresentam apenas o atributo z4_skts_w em comum. O atributo z2_ha2_w, identificado pelo SHAP, é considerado pelo LIME como apresentando contribuição positiva.

Portanto, no que refere à contribuição negativa, há mais aspetos que distanciam os resultados dos dois métodos do que o contrário.

Por fim, apesar de este caso de estudo ter considerado oito atributos, o SHAP não os apresenta todos no gráfico de força, ao contrário do LIME. Este é um ponto que requer melhor análise para perceber este comportamento do SHAP.

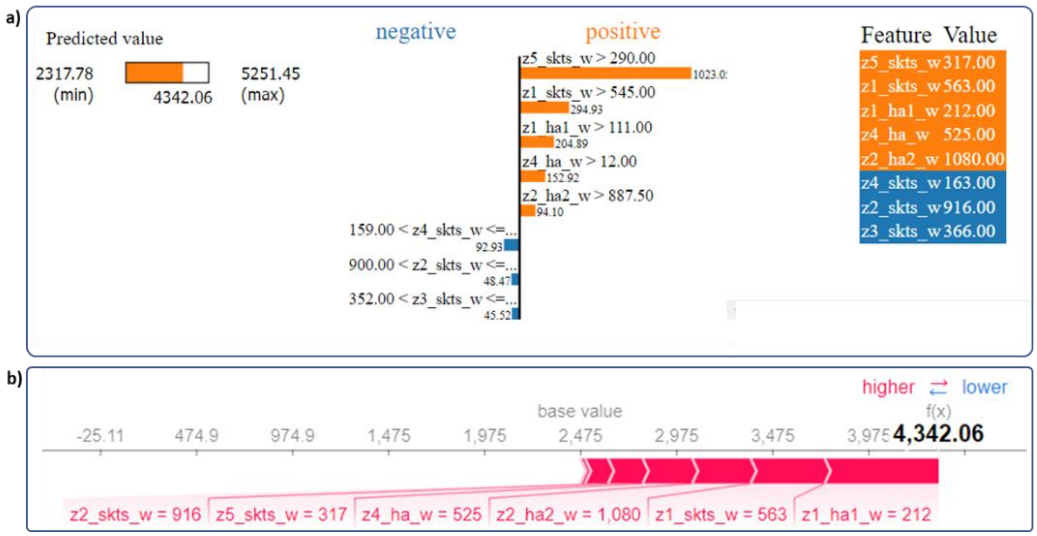


Figura 53 – Interpretação local da previsão do RFR para as 23h00m com (a) LIME e (b) SHAP

Tanto o LIME como o SHAP apresentam a previsão de valor **4342.06W** igual ao valor previsto pelo modelo de ML (Tabela 16). Observando os três primeiros atributos mais importantes, o LIME identificou: z5_skts_w, z1_skts_w e z1_ha1_w. O SHAP identificou os atributos z1_ha1_w, z1_skts_w e z2_ha2_w. Há dois atributos comuns: z1_skts_w e z1_ha1_w. O atributo z1_ha1_w, identificado pelo SHAP como o mais importante, é o terceiro mais importante para o LIME. Portanto, apesar de existirem atributos comuns, os métodos atribuem diferente ordem de importância a estes.

Quanto à contribuição positiva, o LIME identificou cinco atributos e o SHAP seis. Os cinco atributos do LIME, z5_skts, z1_skts_w, z1_ha1_w, z4_ha_w e z2_ha2_w, estão todos no conjunto dos seis atributos identificados pelo SHAP. Este último identificou, ainda, o atributo z2_skts_w. Apesar destas semelhanças entre os dois métodos, os atributos estão ordenados de forma diferente. Por exemplo, o atributo z5_skts_w é o mais importante para o LIME, enquanto para o SHAP este mesmo atributo é dos menos importantes.

No que refere à contribuição negativa, não há atributos identificados pelo SHAP. O LIME identificou os atributos z4_skts_w, z2_skts_w e z3_skts_w. É notória a diferença no resultado da interpretação local entre os dois métodos relativamente à contribuição negativa.

Apresenta-se, a seguir, uma análise comparativa dos resultados do LIME e do SHAP para os casos de estudo 1 e 2 considerando os modelos MLPR e RFR. A partir da Figura 54 e da Figura

55 seguintes, é apresentada uma comparação do resultado do LIME e do SHAP, respetivamente, para o modelo MLPR.

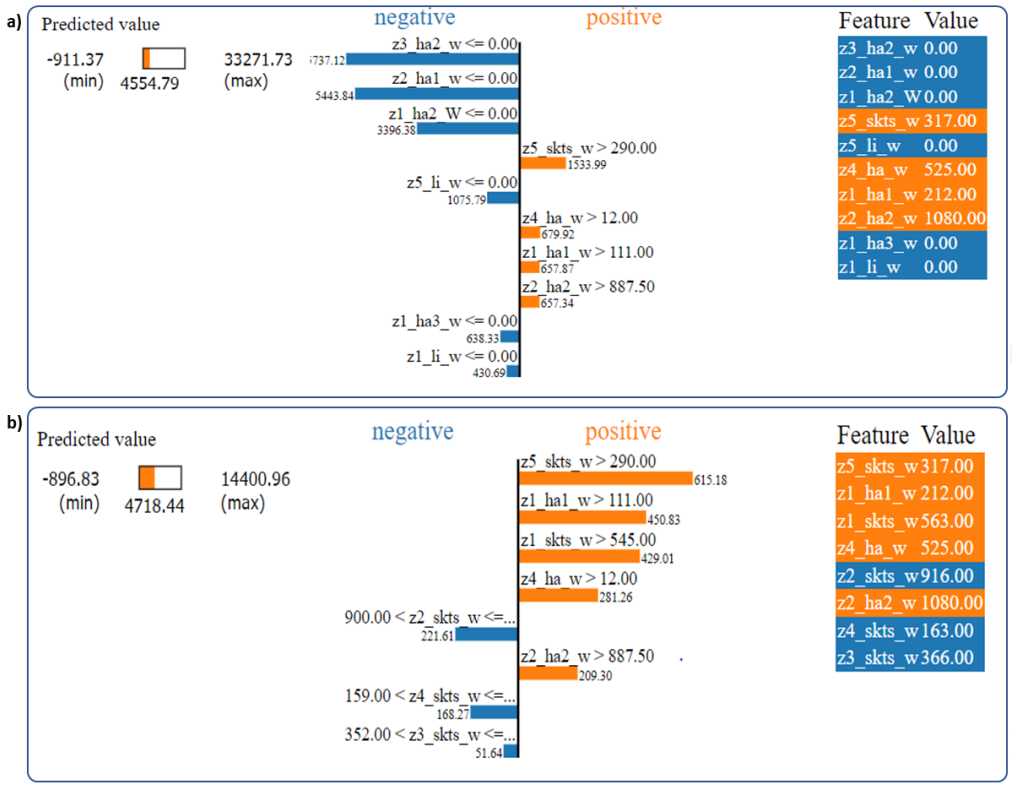


Figura 54 – Interpretação local LIME para o modelo MLPR. Comparação do (a) caso de estudo 1 com (b) caso de estudo 2

Quanto à contribuição positiva, o LIME identificou quatro atributos no caso de estudo 1 e cinco atributos no caso de estudo 2. Os atributos do primeiro caso de estudo, z5_skts, z4_ha_w, z1_ha1_w e z2_ha2_w estão presentes no segundo, sendo que este último apresenta, ainda, o atributo z1_skts_w. Verifica-se alteração na ordem de importância entre os atributos z1_ha1_w e z4_ha_w. Contudo, o atributo z5_skts_w é o mais importante deste grupo, em ambos os casos de estudo. No que refere à contribuição negativa, é notória a diferença. Os atributos z2_skts_w, z4_skts_w e z3_skts_w, do segundo caso de estudo, não estão presentes no primeiro.

Portanto, verificam-se semelhanças na interpretação local do LIME nos dois casos de estudo relativamente à contribuição positiva. A maior diferença reside na contribuição negativa, em consequência do efeito dos atributos com o valor zero.

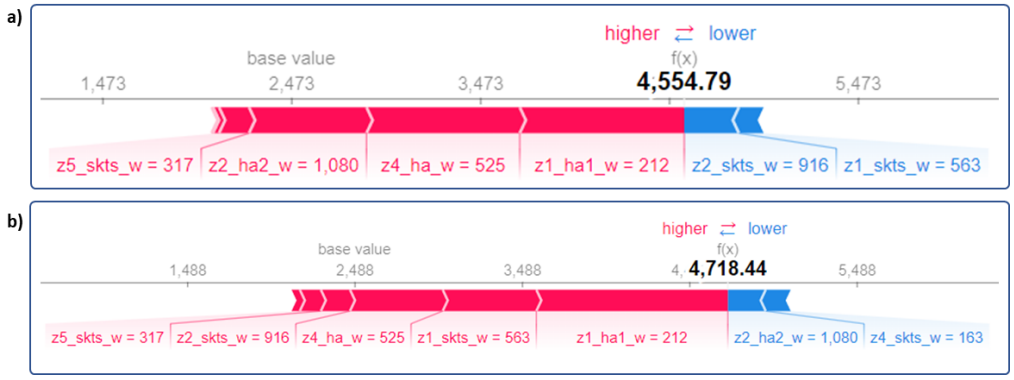


Figura 55 – Interpretação local SHAP para modelo MLPR. Comparação do (a) caso de estudo 1 com (b) caso de estudo 2

Na análise da contribuição positiva, o SHAP identificou quatro atributos no caso de estudo 1 (z1_ha1_w, z4_ha_w, z2_ha2_w, z5_skts_w), e cinco atributos (z1_ha1_w, z1_skts_w z4_ha_w, z2_skts_w e z5_skts_w), caso de estudo 2. Os atributos z1_ha1_w, z4_ha_w e z5_skts_w do primeiro caso estão presentes no segundo caso e com igual ordem de importância. Verifica-se que para a contribuição negativa, o SHAP identificou apenas dois atributos em ambos os casos de estudo. Contudo, há diferença nos atributos entre o primeiro caso de estudo (z2_skts_w e z1_skts_w) e segundo (z2_ha2_w e z4_skts_w).

Portanto, com a retirada dos atributos com valor zero no segundo caso de estudo, o SHAP identificou mais atributos no total (sete no caso de estudo 2 e seis no caso de estudo 1) e mais atributos contribuindo positivamente (seis no caso de estudo 2 e quatro no caso de estudo 1). A partir da Figura 56 e da Figura 57 seguintes, é apresentada uma comparação do resultado do LIME e do SHAP, respetivamente, nos dois casos de estudo do modelo RFR.

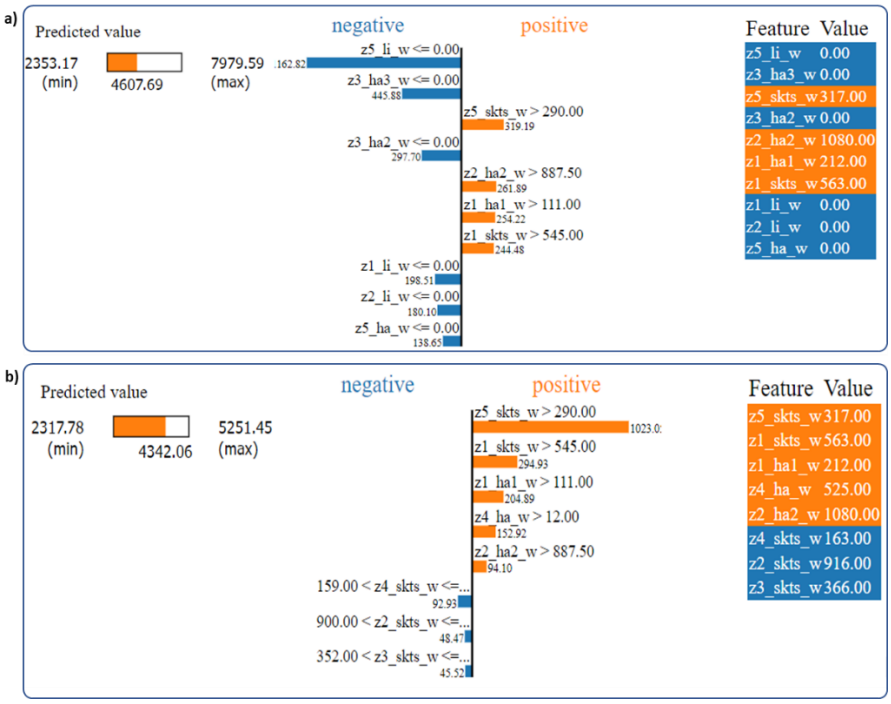


Figura 56 – Interpretação local LIME para o modelo RFR. Comparação do (a) caso de estudo 1 com (b) caso de estudo 2

Em relação à contribuição positiva, o LIME identificou quatro atributos no primeiro caso de estudo e cinco no segundo. Os quatro atributos do primeiro caso (z5_skts_w, z2_ha2_w, z1_ha1_w e z1_skts_1) estão presentes no segundo caso, sendo que este último apresenta ainda o atributo z4_ha_w. Apesar de o atributo z5_skts_w ser o mais importante nos dois casos de estudo, considerando apenas a contribuição positiva, os restantes atributos deste grupo apresentam uma diferente ordem de importância. No que refere à contribuição negativa, é notória a diferença. Os atributos z4_skts_w , z2_skts_w e z3_skts_w, do segundo caso de estudo, não estão presentes no primeiro.

Portanto, verificam-se semelhanças na interpretação local do LIME nos dois casos de estudo relativamente à contribuição positiva. A maior diferença reside na contribuição negativa, em consequência do efeito dos atributos com o valor zero.

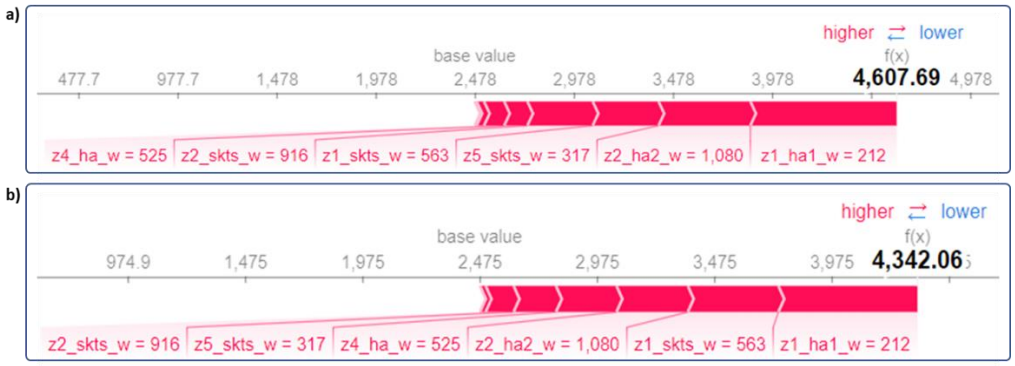


Figura 57 – Interpretação local SHAP para modelo RFR. Comparação do (a) caso de estudo 1 com (b) caso de estudo 2

O SHAP identificou seis atributos com contributo positivo em ambos os casos (z1_ha1_w, z1_skts_w, z2_ha2_w, z4_ha_w, z5_skts_w e z2_skts_w). Os atributos identificados no caso de estudo 2 são os mesmos do caso de estudo 1, porém com diferente ordem de importância. Apenas o atributo z1_ha1_w mantém-se como o atributo mais relevante em ambos os casos. O SHAP não identificou atributos com contribuição negativa, apesar de o conjunto de dados conter oito atributos e este gráfico apresentar apenas seis.

Portanto, verifica-se que a retirada dos atributos com o valor zero não alterou a quantidade de atributos identificados nem os tipos de atributos, contudo verifica-se alteração na ordem de importância destes.

6.3.3.2 Interpretabilidade Global

Nesta secção será apresentado, apenas, o resultado da comparação dos resultados dos métodos explicativos, para interpretabilidade global dos modelos MLPR e RFR, obtidos nos casos de estudo 1 e 2. Esta comparação incidiu apenas sobre os oito atributos do caso de estudo 2: z1_ha1_w, z1_skts_w, z2_ha2_w, z2_stks_w, z3_skts_w, z4_ha_w, z4_skts_w e z5_skts_w. A Figura 58 e a Figura 59 apresentam o gráfico sumário dos casos de estudo 1 e 2, respetivamente, para o modelo MLPR.

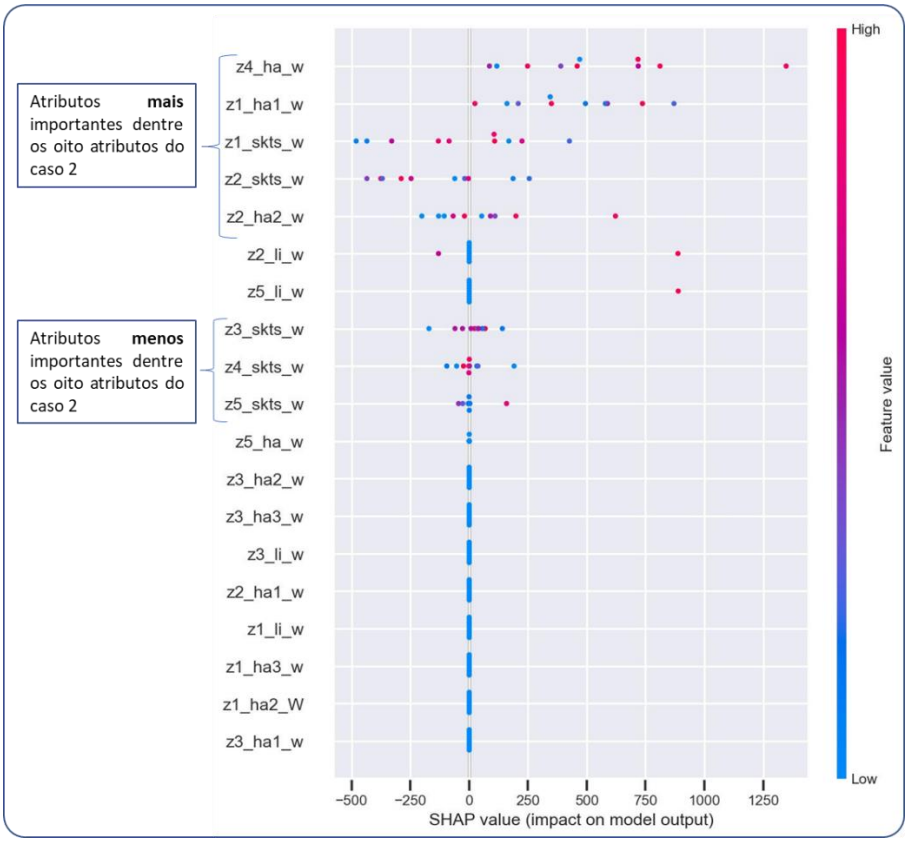


Figura 58 – Interpretação global SHAP do caso de estudo 1 para o modelo MLPR

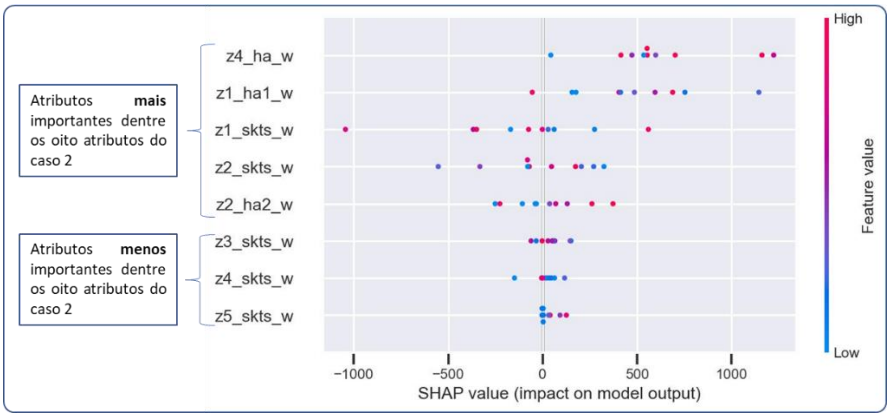


Figura 59 – Interpretação global SHAP do caso de estudo 2 para o modelo MLPR

A retirada dos atributos que apresentavam o valor zero não alterou a ordem de importância dos atributos que não continham o valor zero. Contudo, nota-se uma diferença na distribuição das ocorrências de cada atributo. Por exemplo, no caso do atributo z4_ha_w todos os seus valores apresentam impacto positivo em ambos os casos de uso. Contudo, no caso 1, existe apenas uma ocorrência com impacto positivo cujo *shapley value* superior a 1000. No caso 2 passou a existir duas ocorrências com o *shapley value* maior que 1000.

A Figura 60 e a Figura 61 apresentam o gráfico sumário dos casos de estudo 1 e 2, respetivamente, do modelo RFR.

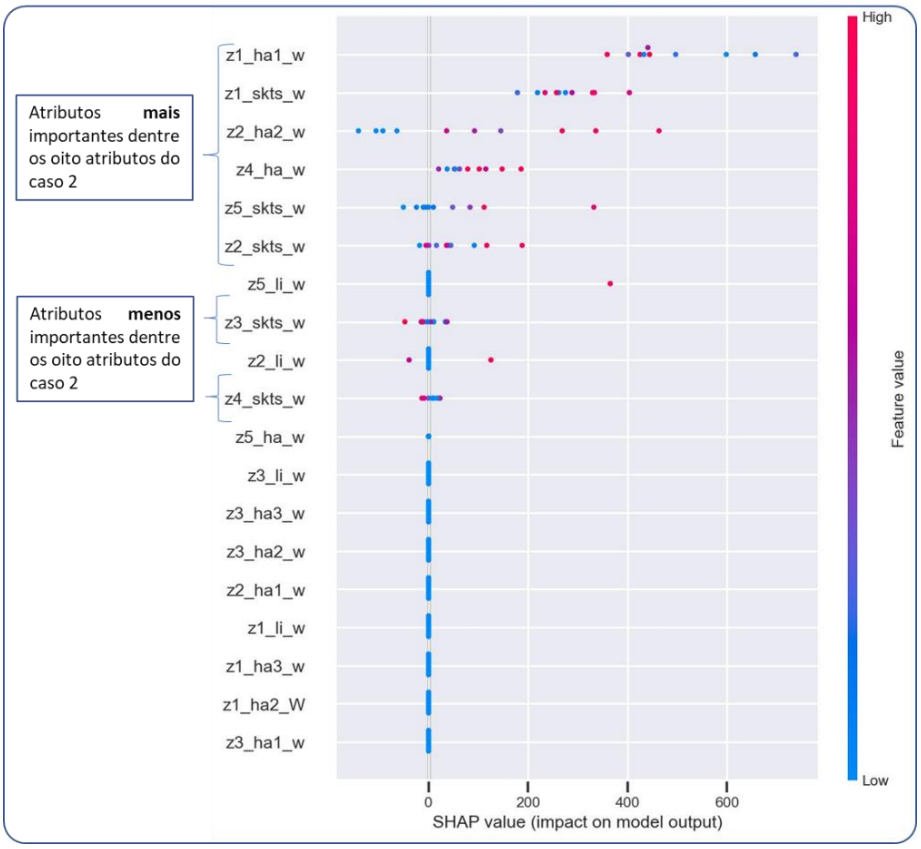


Figura 60 – Interpretação global SHAP do caso de estudo 1 para o modelo RFR

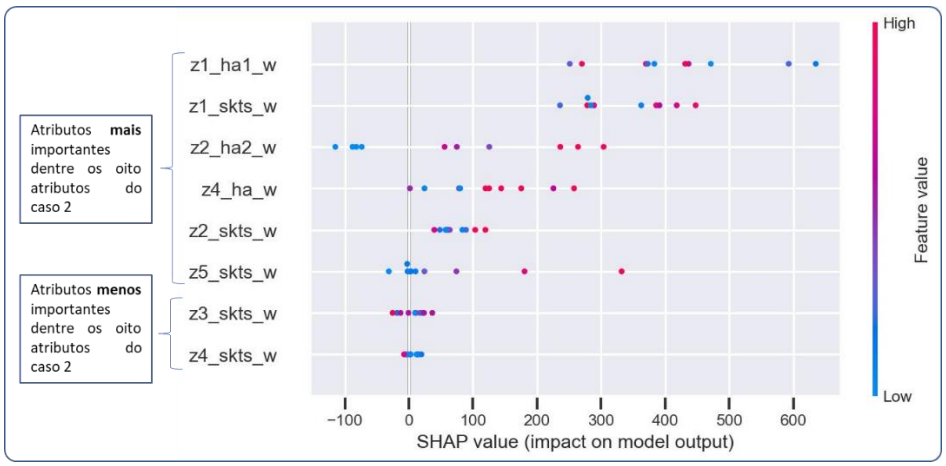


Figura 61 – Interpretação global SHAP do caso de estudo 2 para o modelo RFR

Verifica-se alteração na ordem de importância dos atributos que não contêm o valor zero. Há uma troca de importância entre os atributos z2_skts_w e z5_skts_w. Nota-se, ainda, uma diferença na distribuição das ocorrências de cada atributo. Por exemplo, no caso do atributo z1_ha_w todos os seus valores apresentam impacto positivo em ambos os casos de uso.

Contudo, no caso 1, existem uma ocorrências com impacto positivo cujo *shapley value* superior a 6000. No caso 2 passou a existir apenas uma ocorrência com o *shapley value* maior que 6000.

6.3.3.3 Discussão

No caso da interpretabilidade local, considerando a perspetiva da contribuição positiva, verifica-se que o LIME identificou, em ambos os modelos, mais atributos com contribuição positiva no caso de estudo 2, sendo que, os atributos do primeiro caso de estudo surgem no segundo caso (Figura 54 e Figura 56). Outro aspeto no LIME é a diferença na ordem de importância dos atributos entre os dois casos de uso. No caso do SHAP, apenas no modelo MLPR foram identificados mais atributos no caso de estudo 2 (Figura 55). Quanto à ordem de importância identificada pelo SHAP, há uma significativa diferença para o modelo RFR (Figura 57). Ainda no contexto da interpretabilidade local, mas considerando a contribuição negativa, o LIME é o método que apresenta maiores diferenças na comparação entre os dois casos de estudo (Figura 54 e Figura 56), o que se justifica pelo facto de os atributos com valor zero não existirem no caso de estudo 2.

Do ponto de vista da interpretabilidade global, verifica-se que quanto maior a percentagem de registos com valor zero, menor é o seu contributo para o comportamento global do modelo. Contudo, a retirada destes atributos não implicou uma grande alteração na interpretação do SHAP acerca do comportamento global dos modelos MLPR (Figura 58 e Figura 59) e RFR (Figura 60 e Figura 61).

Fica em aberto a certeza de que estes modelos do segundo caso de estudo são mais confiáveis que os do primeiro. Apesar de ao nível da interpretabilidade global não existir grandes diferenças, o mesmo não ocorre ao nível da interpretabilidade local, nomeadamente pela diferença na ordem de importância dos atributos. Verifica-se que os modelos do segundo caso de estudo apresentam métricas com menor valor que as apresentadas no caso de estudo 1 (Tabela 15). Contudo, este facto não é suficiente para assumir que os modelos do segundo caso de estudo são menos fiáveis. Aqui está uma situação em que é preciso um especialista, conhecedor da dinâmica de consumo do edifício, para avaliar se determinados atributos utilizados no modelo de ML são relevantes para as previsões e também analisar os resultados apresentados pelo LIME e pelo SHAP.

6.4 Caso de Estudo 3

6.4.1 Problema

Os casos de estudo anteriores consideraram conjuntos de dados do tipo série temporal com 2976 registos e que apresentam uma frequência de 15 minutos. É importante realizar uma análise do tempo de processamento consumido nestes casos de estudo de forma a permitir ter

uma perceção do tempo que poderá ser necessário para cenários com uma série temporal com diferentes frequências e, consequentemente, com um maior número de registos. Assim, o objetivo deste caso de estudo é analisar o tempo médio de processamento das bibliotecas do LIME e do SHAP nos casos de estudo 1 e 2.

6.4.2 Execução

Durante o processamento dos casos de estudo 1 e 2 foi realizada a recolha do tempo de execução dos explicadores para cada uma das instâncias de interesse do conjunto de dados de teste. Os valores obtidos são a média do tempo de processamento do explicador. Importa relembrar a configuração de instanciação dos explicadores do LIME e do SHAP:

LIME

- ***training_data*:** todo o conjunto de dados de treino de cada interação do *multi-step-ahead*;
- ***num_samples:*** 10000

SHAP

- ***background dataset***: os 100 primeiros registos do conjunto de dados de treino de cada interação da janela acumulativa

6.4.3 Resultados e Discussão

A Tabela 17 apresenta o resumo do tempo médio, em segundos, do processamento das bibliotecas do LIME e do SHAP para os casos de estudo 1 e 2 considerando os dois modelos de ML.

Tabela 17 – Tempo médio de processamento do LIME e do SHAP

Caso de estudo	Conjunto de dados	Modelo	LIME (tempo(s))	SHAP (tempo(s))
1	20 atributos	MLPR	5.2975	0.4736
2	8 atributos	MLPR	3.8274	0.1421
1	20 atributos	RFR	4.5185	0.7364
2	8 atributos	RFR	3.6115	0.3060

Verifica-se que o SHAP é, em média, mais rápido que o LIME. Apesar de este apresentar melhor performance, os resultados demonstram uma degradação significativa na performance do SHAP com modelo RFR seja com 20 ou com 8 atributos. No cenário com 20 atributos, tempo médio no modelo MLPR foi de 0.4736s e no RFR foi de 0.7364s, ou seja, um aumento de cerca de 55,4%. No cenário com 8 atributos, para modelo MLPR o tempo médio foi de 0.1421s e no RF foi de 0.3060s, ou seja, um aumento de cerca de 115%.

O LIME apresenta melhor performance com o modelo RFR. No cenário com 20 atributos, o tempo médio no modelo MLPR foi de 5,2975s e no RFR foi de 4,5185s, ou seja, uma diminuição de cerca de 14,7%. No cenário com 8 atributos, o tempo médio no modelo MLPR foi de 3,8274s e no RF foi de 3,6115s, ou seja, uma diminuição de cerca de 5,6%.

6.4.3.1 Discussão

Na secção 3.5.3 é referido que uma das desvantagens do *Kernel SHAP* é o facto de este ser lento. A fonte desta afirmação (Molnar, C. 2022) não indica qual é o fator de comparação, i.e., se o *Kernel SHAP* é, por exemplo, mais lento que o LIME. Neste estudo verificou-se que este método explicativo apresenta melhor performance que o LIME considerando a parametrização acima indicada. Contudo, verifica-se uma significativa degradação de performance no *Kernel SHAP* quando é feita a comparação entre os modelos MLPR e o RFR. Uma outra opção para o processamento do *shapley value* com modelos do gênero *Random Forest* é a utilização do *TreeSHAP* . Este se caracteriza por ser do tipo modelo específico e dedicado para modelos de ML do tipo árvore de decisão tendo sido proposto por (Lundberg et al., 2018). O estudo do *TreeSHAP* está fora deste âmbito.

6.5 Caso de Estudo 4

6.5.1 Problema

Em (Antwarg et al., 2021), os autores utilizam o SHAP para tarefas de deteção de anomalias, isto é, identificar, em um conjunto de dados, padrões que não estão em conformidade com o esperado (Prasad et al., 2009). Neste estudo, os autores verificaram que a escolha do *background dataset* influenciou de forma errada os cálculos do SHAP para identificação dos atributos relevantes. Portanto, pode haver interesse em utilizar diferentes quantidades de registos para o *background dataset* nos processos de geração de explicações. Neste sentido, será avaliado o tempo de processamento do SHAP para *background dataset* com diferentes tamanhos, i.e., total de registos. Assim será possível realizar uma comparação com o tempo de processamento obtido no caso de estudo 3 (secção 6.4). Nesta última análise são contemplados dois cenários para os modelos MLPR e RFR: a) oito atributos; b) vinte atributos.

6.5.2 Execução

Nos casos de uso anteriores, o *background dataset* era constituído pelos primeiros 100 registos do conjunto de dados de treino do modelo de ML. Para este caso de estudo foram considerados três cenários: a) 500 registos; b) 1000 registo; c) 1500 registos.

6.5.3 Resultados e Dicussão

As tabelas Tabela 18, Tabela 19, Tabela 20 apresentam uma comparação do tempo de processamento do SHAP consoante a variação do total de registo do *background dataset* em relação ao total de 100 registos do caso de estudo 3. Na Tabela 18, o total de registos é 500 (5 vezes superior ao total de registos do caso de estudo 3). Na Tabela 19, o total de registo é 1000 (10 vezes superior ao total de registos do caso de estudo 3). Na Tabela 20, o total de registo é 1500 (15 vezes superior ao total de registos do caso de estudo 3).

Tabela 18 – *Background dataset* com 500 registos

Cenário	Modelo	Caso 3 (tempo(s))	Caso 4 (tempo(s))	Lentidão
20 atributos	MLPR	0.4736	4.7044	≅ 10 vezes
8 atributos	MLPR	0.1421	0.6468	≅ 4,5 vezes
20 atributos	RFR	0.7364	12.1112	≅ 16 vezes
8 atributos	RFR	0.3060	1.4172	≅ 4,6 vezes

Para um aumento de 5 vezes no total de registos do *background dataset* verifica-se um aumento no tempo de processamento de no mínimo 4,5 vezes (8 atributos para MKPR) quando comparado com o caso de estudo 3. O maior aumento registado é de 16 vezes (20 atributos para RFR).

Tabela 19 – *Background dataset* com 1000 registos

Cenário	Modelo	Caso 3 (tempo(s))	Caso 4 (tempo (s))	Lentidão
20 atributos	MLPR	0.4736	9.4407	≅ 20 vezes
8 atributos	MLPR	0.1421	1.2471	≅ 8,7 vezes
20 atributos	RFR	0.7364	25.8072	≅ 35 vezes
8 atributos	RFR	0.3060	3.0490	≅ 10 vezes

Na Tabela 19, no cenário com 1000 registos, verifica-se que o processamento pode ser até 35 vezes mais lento (20 atributos RFR) do que com 100 registos, do caso de estudo 3. Verifica-se que esta lentidão é quase o dobro do valor mais lento verificado na Tabela 18, para 500 registos (16 vezes).

Tabela 20 – *Background dataset* com 1500 registos

Cenário	Modelo	Caso 3 (tempo(s))	Caso 4 (tempo(s))	Lentidão
20 atributos	MLPR	0.4736	15.2837	≅ 32 vezes
8 atributos	MLPR	0.1421	1.8277	≅ 12 vezes
20 atributos	RFR	0.7364	38.5966	≅ 52 vezes
8 atributos	RFR	0.3060	4.5378	≅ 14 vezes

Este último cenário é, sem dúvida, o mais lento. No caso do tempo de processamento mais lento (20 atributos RFR) com um tempo de processamento de cerca de 38.59s, o que é 52 vezes mais lento que o caso de estudo 3. Este aumento é mais do triplo que o valor verificado no cenário com 500 registos Tabela 18 (16 vezes).

Na Figura 62, dois gráficos resumem a evolução do tempo de processamento do SHAP em função da variação do total de registo do *background dataset*. Desta forma é possível ter uma perspetiva visual do aumento no tempo de processamento.

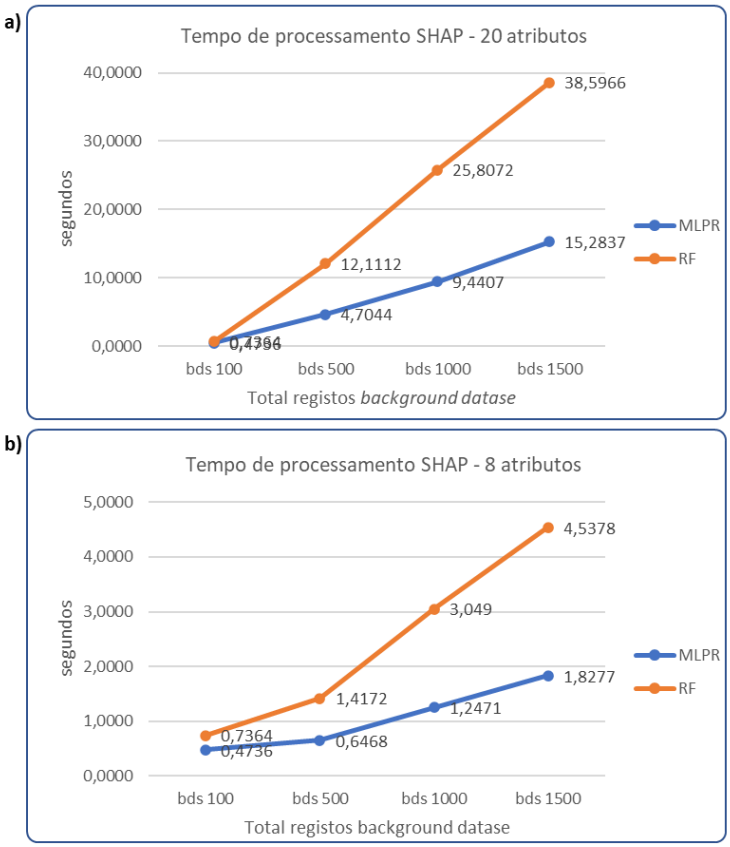


Figura 62 – Tempo de processamento do SHAP. Comparação do (a) 20 atributos com (b) 8 atributos

Como se pode verificar, o SHAP é mais lento no RFR seja no caso (a) 20 atributos ou no caso (b) 8 atributos. No caso (a) 20 atributos, o aumento no tempo de processamento apresenta uma tendência linear tanto para o MLPR como par ao RFR. No caso (b) 8 atributos, o tempo de processamento tende a aumentar, mas não chega próximo dos valores do caso (a) 20 atributos, o que confirma que para além do total de registo, a quantidade de atributos é outro fator determinante para o tempo de processamento.

6.5.3.1 Discussão

Este caso de estudo demonstra que, para além do número de registos do *background dataset*, também o total de atributos influencia o tempo de processamento do SHAP. Este caso de estudo evidencia a tendência verificada no caso de estudo 3 que indicava que o SHAP é muito mais lento com o RFR do que com o MLPR.

6.6 Considerações Finais

O gráfico formato HTML do LIME e o gráfico de força do SHAP permitem identificar os atributos que mais contribuem para uma previsão em um cenário de regressão utilizando séries temporais. Verifica-se uma significativa diferença de comportamento do LIME e do SHAP quando existem atributos das instâncias de interesse com o valor zero. Enquanto o primeiro pode atribuir elevada importância a alguns atributos com o valor zero, o mesmo não acontece com o SHAP.

Uma hipótese para explicar este comportamento do SHAP é a propriedade neutralidade (secção 3.4) do *shapley value*. De acordo com esta, se um atributo não contribui na previsão de qualquer das alianças, então o *shapley value* é zero. Isto poderá justificar o facto de o SHAP não reconhecer o impacto diferente de nulo, no comportamento global do modelo, dos atributos com o valor zero, tal como se verifica no gráfico sumário (secção 6.3.3.2).

No caso de estudo 4 nota-se o quanto o aumento do número de registos do *background dataset* afeta negativamente o tempo de processamento do SHAP. Isto não invalida que possam ser utilizados mais registos para uma melhor identificação dos padrões reconhecidos pelos modelos de ML. Contudo, é preciso ter em consideração o contexto de utilização. Se for um software de utilização em tempo real, isto é, com resposta imediata para um utilizador, o tempo de processamento poderá ter impactos negativos na utilização do software. No âmbito do projeto PRECISE, está previsto a utilização de soluções para a análise, em tempo real, das previsões de consumo de energia em edifícios para frequências de 5, 10 ou 15 minutos. Importa, portanto, ter em atenção o tempo de resposta dos métodos explicativos.

O capítulo seguinte é dedicado às conclusões deste estudo e perspetivas de trabalho futuro. Serão apresentados os principais contributos para a integração de métodos explicativos no contexto do projeto PRECISE.

7 Conclusões e Trabalho Futuro

O trabalho realizado neste estudo iniciou com a apresentação dos desafios no setor energético, nomeadamente devido à transição energética. É identificada a importância da AI neste setor, a necessidade de sistemas baseados em AI terem suporte para explicações. assim como a iniciativa do GECAD em implementar mecanismos de explicação em sistemas de energia através do projeto PRECISE.

Foi realizada uma apresentação do estado da arte de XAI com incidência no setor energético e nas técnicas de ML. Como resultado deste estado da arte, identificou-se que a investigação de XAI no setor energético ainda tem pouca expressão e com potencial de crescimento devido ao aumento do uso de AI nomeadamente com base nas técnicas de ML. Os métodos de XAI ainda têm pouca utilização em modelos de ML baseados em regressão e que utilizam conjuntos de dados do tipo série temporal, o qual é muito utilizado em previsões de consumos de eletricidade. Este estudo apresentou os conceitos que estão na base do LIME e do SHAP, dois dos métodos explicativos mais utilizados, tal como identificado no estado da arte, o LIME e o SHAP. Foi realizada uma breve descrição das bibliotecas que implementam estes métodos e suas ferramentas de explicação visual.

Foram implementados quatro casos de estudo baseados em dados reais e utilizando modelos de ML baseados em regressão para avaliação das ferramentas de explicação visual disponibilizadas pelas bibliotecas do LIME e do SHAP. Neste capítulo apresentam-se as conclusões dos casos de estudo, contribuições e perspetivas de trabalho futuro.

7.1 Conclusões e contribuições

Nos casos de estudo 1 e 2 foram aplicados os métodos explicativos LIME e SHAP para gerar explicações de previsões de consumos de energia. As previsões e explicações tiveram por base séries temporais e consideraram dois modelos de ML: *Random Forest Regressor* e *Artificial*

Neural Network. As explicações foram obtidas com base em modelos do tipo agnóstico e contemplaram dois níveis de interpretabilidade: local e global.

No caso de estudo 1, o conjunto de dados utilizado apresentava registos com valor zero para os atributos escolhidos no estudo de geração de explicações. Os métodos, LIME e SHAP, apresentaram distintos comportamentos relativamente aos atributos com valor zero. Ao nível da interpretabilidade local, o LIME destaca elevada importância a alguns destes atributos. Inclusive, alguns destes foram classificados como mais relevantes que os atributos com valor diferente de zero. No caso do SHAP, este método simplesmente não identifica os atributos com valor zero como apresentando qualquer contribuição para a previsão. Na interpretabilidade global do SHAP é notório o quanto para este método os atributos com o valor zero não apresentam qualquer contributo para previsão. No gráfico sumário do SHAP estes atributos são classificados como apresentando impacto zero no comportamento global do modelo. Quanto aos atributos com valor diferente de zero, o SHAP identificou os mesmos atributos identificados pelo LIME, sendo que consoante o modelo de ML existem duas diferenças: a) a ordem de importância dos atributos varia; b) o SHAP identificou mais atributos que o LIME.

Uma hipótese para explicar este comportamento do SHAP é a propriedade neutralidade (secção 3.4) do *shapley value*. De acordo com esta, se um atributo não contribui na previsão de qualquer das alianças, então o *shapley value* é zero. Isto poderá justificar o facto de o SHAP não reconhecer nos atributos com o valor zero que estes têm impacto na previsão da instância de interesse e mesmo no comportamento global do modelo, como se verifica no gráfico sumário (secção 6.3.3.2).

Há duas grandes diferenças a destacar no comportamento do LIME e do SHAP: a) o LIME pode apresentar diferentes resultados, nomeadamente entre atributos cujos valores dos coeficientes estão muito próximos; b) a forma como lidam com os atributos com o valor zero. A primeira tende a que se escolha o SHAP por ser mais estável em diferentes interações, desde que se mantenham os parâmetros de instanciação dos explicadores. A segunda pode pôr em causa a escolha do SHAP. O conjunto de dados utilizado contém dados reais de consumos de energia de diferentes dispositivos em diferentes momentos no tempo. Portanto, é inevitável que ocorram registos com o valor zero devido a diferentes fatores como: a) falha técnica no registo da informação; b) os dispositivos não apresentam consumo por estarem desligados; e outros.

Apesar de o LIME identificar alguns atributos com o valor zero como os mais relevantes, há outros com o valor zero que estão entre os menos relevantes, inclusive abaixo dos atributos com valor diferente de zero. Isto pode indicar um desbalanceamento do conjunto de dados relativamente a alguns atributos, situação sugerida pelo SHAP apenas na interpetabilidade global. Portanto, considerando a inevitabilidade de existirem atributos com valor zero, talvez a melhor abordagem seja a utilização dos dois métodos em conjunto enquanto complementares na análise dos atributos mais relevantes.

No caso de estudo 2, verifica-se na interpretabilidade global que a retirada dos atributos com valor zero não provocou grandes alterações na ordem de importância dos restantes atributos

no comportamento global em ambos os modelos. Ao nível da interpretabilidade local, a maior parte dos atributos identificados no LIME e no SHAP no caso 1 está presente no caso 2, em ambos os métodos explicativos. A maior diferença reside na ordem de importância dos atributos que se altera no caso 2. Os valores das métricas definidas para os modelos de ML são mais baixos no caso 2. No que refere aos atributos que contribuem de forma positiva para as previsões, a grande diferença reside na ordem de importância destes. Contudo, sem um especialista que conheça a dinâmica do edifício é difícil perceber qual dos dois modelos é o que melhor identifica os dispositivos que mais influenciam no consumo de energia.

Nos casos de estudo 3 e 4. o SHAP apresenta uma elevada degradação na performance para o modelo RFR, quando comparado com o MLPR. O caso de estudo 3 sugere que o SHAP, para a implementação do *Kernel SHAP*, é mais rápido que o LIME. Contudo, à medida que aumenta o total de registos do *background dataset* evidencia-se a maior lentidão do SHAP em relação ao LIME. O LIME foi executado nos casos de estudo 1, 2 e 3 sempre com a mesma quantidade de registos do conjunto de dados de teste e que era superior a 2000 registos. Nos testes realizados no SHAP, número máximo de registos foi de 1500.

Os métodos explicativos, LIME e SHAP, procuram estimar a contribuição de cada atributo para o valor previsto em uma instância de interesse. Ambos recorrem a modelos interpretáveis sendo que o SHAP incorpora um conceito matemático baseado na teoria dos jogos, o *shapley value*. Estes métodos **permitem perceber os atributos que os modelos de ML identificam como mais relevantes, contudo não se compreende por que o modelo identificou determinados padrões como mais importantes**. Não se consegue afirmar que estes métodos permitem aumentar a confiança dos utilizadores nos modelos de ML. Contudo, podem ser uma importante ferramenta para "lançar uma luz" nos valores previstos pelos modelos através dos atributos identificados por estes como os mais relevantes. Desta forma, podem contribuir no sentido de proporcionarem o desenvolvimento de modelos de ML cujas previsões estejam mais próximas da realidade.

Neste trabalho foi avaliada a possibilidade de estes métodos serem utilizados para gerar explicações de previsões de consumo de energia em séries temporais e de estes serem integrados em tarefas do projeto PRECISE. Do ponto de vista técnico, não há impedimentos. Contudo, a análise das ferramentas de explicação visual das bibliotecas do LIME e do SHAP sugere que a interpretação destas requer algum conhecimento especializado, nomeadamente de conceitos matemáticos. Desta forma, pode não ser imediata a utilização destas ferramentas por utilizadores sem este conhecimento especializado seja dos conceitos matemáticos inerentes aos métodos explicativos, seja de ML. Por fim, como principais contributos deste trabalho para o projeto PRECISE destacam-se:

- Analise do estado da arte;
- Estudo das ferramentas para produzir explicações;
- Aplicação das ferramentas a casos reais;
- Integração com as tarefas do projeto PRECISE;
- Estudo de possíveis interpretações a fazer dos modelos;

- Integração com mais do que um tipo de modelo.

De salientar que, deste trabalho resultou o artigo científico *Explainable artificial intelligence (XAI) techniques for energy consumption* [53] apresentado no evento *The 9th International Conference on Energy and Environment Research* (ICEER)[54] em 2022. Neste artigo é explorada a utilização do LIME e do SHAP para gerar explicações de previsões de consumos de eletricidade geradas pelos modelos KNN e ANN utilizando séries temporais. As explicações contemplavam interpretabilidade local e global. Está também em preparação um outro artigo, que será submetido em revista internacional de alto fator de impacto, como *Applied Energy*, que contém o desenvolvimento, resultados e conclusões finais deste trabalho.

7.2 Trabalho Futuro

Este estudo foi aplicado na explicação de consumos de eletricidade. O estado da arte refere a utilização de AI em previsões de produção de energia, nomeadamente a partir de fontes renováveis através de painéis fotovoltaicos. O conjunto de dados do *smart grid competion* também contém registos de produção de energia de um edifício e, portanto, pode ser a base para a aplicação dos métodos LIME e SHAP em cenários de previsão de produção de energia.

Do ponto de vista mais técnico, no que refere aos métodos LIME e SHAP, seria interessante poder explorar a execução destes com outros conjuntos de parâmetros que não foram explorados neste estudo. Por exemplo, instanciar o LIME com diferentes quantidades de exemplares ou outros tipos de modelos interpretáveis (secção 3.3.1). No caso do SHAP, seria interessante explorar outros gráficos disponibilizados pela biblioteca e que não foram alvo de estudo neste trabalho. Por exemplo, foi referido o gráfico em cascata, contudo este não foi contemplado nos casos de estudo.

Apesar de haver grande interesse em utilizar métodos do tipo modelo agnóstico, para contemplar o maior número possível de modelos de ML, a exploração de métodos do tipo modelo específico poderia trazer benefícios no caso do SHAP. Verificou-se uma significativa degradação na performance do SHAP, nomeadamente no modelo RFR, utilizando o explicador *KernelExplainer*, que implementa o conceito de *Kernel SHAP* o qual é do tipo modelo agnóstico. Os autores do SHAP desenvolveram o *TreeExplainer*, que uma implementação otimizada para modelos do tipo *Decision Tree*. Pode ser interessante explorar esta alternativa para se obter resultados com menor tempo de processamento. Outras técnicas de XAI (secção 2.2.4) poderiam ser exploradas em cenários de consumo como de produção.

No estado da arte foi identificado o uso de LSTM (secção 2.3.1) em séries temporais. Teoricamente, o *Kernel SHAP*, na sua implementação *KernelExplainer*, suporta a geração de

[53] *Explainable artificial intelligence (XAI) techniques for energy consumption* – https://www.gecad.isep.ipp.pt/precise/dissemination/, último acesso 10 de outubro de 2022
[54] ICEER – http://www.iceer.net/, último acesso 10 de outubro de 2022

explicações para este tipo de modelo de ML. Contudo, o âmbito deste estudo não permitiu explorar o SHAP com este modelo de ML.

Por fim, a possibilidade de persistir os resultados dos cálculos do LIME e do SHAP, sejam resultados visuais (os gráfios) ou textuais (os valores calculados), permite a incorporação destes em um software desenvolvido no contexto do PRECISE. Inclusive, os valores calculados podem ser utilizados para gerar explicações textuais que complementem as explicações visuais.

Referências

(Adadi & Berrada, 2018) Adadi, A., & Berrada, M. (2018). Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). IEEE Access, 6, 52138–52160. https://doi.org/10.1109/ACCESS.2018.2870052

(Afridi et al., 2021) Afridi, Y. S., Ahmad, K., & Hassan, L. (2021). Artificial intelligence based prognostic maintenance of renewable energy systems: A review of techniques, challenges, and future research directions. International Journal of Energy Research. https://doi.org/10.1002/er.7100

(Ahmad et al. 2021) Ahmad, T., Zhang, D., Huang, C., Zhang, H., Dai, N., Song, Y., & Chen, H. (2021). Artificial intelligence in sustainable energy industry: Status Quo, challenges and opportunities. *Journal of Cleaner Production*, *289*, 125834. https://doi.org/10.1016/j.jclepro.2021.125834

(AI HLEG et al. 2019) High-Level Expert Group on Artificial Intelligence (AI HLEG). 2019. "The Ethics Guidelines for Trustworthy Artificial Intelligence," no. Accessed: December 1, 2021. [Online]. https://ec.europa.eu/futurium/en/ai-alliance-consultation/guidelines.1.html.

(Amparore et al., 2021) Amparore, E., Perotti, A., & Bajardi, P. (2021). To trust or not to trust an explanation: using LEAF to evaluate local linear XAI methods. *PeerJ Computer Science*, *7*, e479. https://doi.org/10.7717/peerj-cs.479

(Antwarg et al., 2021) Antwarg, L., Miller, R. M., Shapira, B., & Rokach, L. (2021). Explaining anomalies detected by autoencoders using Shapley Additive Explanations. *Expert Systems with Applications*, *186*, 115736. https://doi.org/10.1016/j.eswa.2021.115736

(Asif, M. et al. 2022) Asif, M. (2022). Handbook of Energy Transitions (1st ed.). CRC Press. https://doi.org/10.1201/9781003315353

(Atzmueller et al., 2019) Atzmueller, M. (2019). *Towards Socio-Technical Design of Explicative Systems: Transparent, Interpretable and Explainable Analytics and its Perspectives in Social Interaction Contexts*.

(Bach et al., 2015) Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K. R., & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS ONE, 10(7). https://doi.org/10.1371/journal.pone.0130140

(Barredo Arrieta et al., 2020) Barredo Arrieta, A., Díaz-Rodríguez, N., del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion, 58, 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

(Bontempi et al., 2013) Bontempi, G., ben Taieb, S., & le Borgne, Y. A. (2013). Machine learning strategies for time series forecasting. *Lecture Notes in Business Information Processing*, *138 LNBIP*, 62–77. https://doi.org/10.1007/978-3-642-36318-4_3

(Carvalho et al., 2019) Carvalho, D. v., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. Electronics (Switzerland), 8(8), 1–34. https://doi.org/10.3390/electronics8080832

(Das & Rad, 2020) Das, A., & Rad, P. (2020). Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey. http://arxiv.org/abs/2006.11371

(Donti & Kolter, 2021) Donti, P. L., & Kolter, J. Z. (2021). Machine Learning for Sustainable Energy Systems. Annual Review of Environment and Resources, 46, 719–747. https://doi.org/10.1146/annurev-environ-020220-061831

(Doshi-Velez & Kim, 2017) Doshi-Velez, F., & Kim, B. (2017). Towards A Rigorous Science of Interpretable Machine Learning. http://arxiv.org/abs/1702.08608

(Ethington et al., 2002) Ethington, C.A., Thomas, S.L., Pike, G.R. (2002). Back to the Basics: Regression as It Should Be. In: Smart, J.C., Tierney, W.G. (eds) Higher Education: Handbook of Theory and Research. Higher Education: Handbook of Theory and Research, vol 17. Springer, Dordrecht. https://doi.org/10.1007/978-94-010-0245-5_6

(European Commission et al. 2019a) "A European Green Deal | European Commission," no. Accessed: December 1, 2021. [Online]. https://ec.europa.eu/info/strategy/priorities-2019-2024/european-green-deal_en#highlights

(European Commission et al. 2019b) "Energy and the Green Deal," no. Accessed: December 1, 2021. [Online]. https://ec.europa.eu/info/strategy/priorities-2019-2024/european-green-deal/energy-and-green-deal_en

(European Commission et al. 2020) "Assessment List for Trustworthy AI (ALTAI)," no. Accessed: December 1, 2021. [Online]. https://digital-strategy.ec.europa.eu/en/news/artificial-intelligence-commission-welcomes-opportunities-offered-final-assessment-list-trustworthy

(European Parliament and AIDA et al. 2021) "Special Committee on Artificial Intelligence in a Digital Age (AIDA) | European Parliament," no. Accessed: December 1, 2021. [Online]. https://www.europarl.europa.eu/committees/en/aida/about.

(Gailhofer et al. 2021) Gailhofer, Peter, Anke Herold, Jan Peter Schemmel, Cara-Sophie Scherf, Cristina Urrutia, Andreas R. Köhler, and Sibylle Braungardt. 2021. "The Role of Artificial Intelligence in the European Green Deal," no. May: 70. https://www.europarl.europa.eu/RegData/etudes/STUD/2021/662906/IPOL_STU(2021)662906_EN.pdf

(Garg et al., 2021) Garg, S., Sinha, S., Kar, A. K., & Mani, M. (2021). A review of machine learning applications in human resource management. International Journal of Productivity and Performance Management, ahead-of-p(ahead-of-print). https://doi.org/10.1108/IJPPM-08-2020-0427

(Garreau & von Luxburg, 2020) Garreau, D., & von Luxburg, U. (2020). Explaining the Explainer: A First Theoretical Analysis of LIME. In S. Chiappa & R. Calandra (Eds.), Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (Vol. 108, pp. 1287–1296). PMLR. https://proceedings.mlr.press/v108/garreau20a.html

(Gasparin et al., 2019) Gasparin, A., Lukovic, S., & Alippi, C. (2019). Deep Learning for Time Series Forecasting: The Electric Load Case. http://arxiv.org/abs/1907.09207

(Géron et al., 2017) Géron, A. (2017). Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. Sebastopol, CA: O'Reilly Media. ISBN: 978-1491962299

(Gonzalez-Briones et al., 2019) Gonzalez-Briones, A., Hernandez, G., Corchado, J. M., Omatu, S., & Mohamad, M. S. (2019). Machine Learning Models for Electricity Consumption Forecasting: A Review. 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS), 1–6. https://doi.org/10.1109/CAIS.2019.8769508

(Guidotti et al., 2019) Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2019). A Survey of Methods for Explaining Black Box Models. ACM Computing Surveys, 51(5), 1–42. https://doi.org/10.1145/3236009

(Gunning & Aha, 2019) Gunning, D., & Aha, D. (2019). DARPA's Explainable Artificial Intelligence (XAI) Program. AI Magazine, 40(2), 44–58. https://doi.org/10.1609/aimag.v40i2.2850

(Gunning et al., 2021) Gunning, D., Vorm, E., Wang, J. Y., & Turek, M. (2021). DARPA 's explainable AI (XAI) program: A retrospective . Applied AI Letters, 1–12. https://doi.org/10.1002/ail2.61

(Helm et al., 2020) Helm, J. M., Swiergosz, A. M., Haeberle, H. S., Karnuta, J. M., Schaffer, J. L., Krebs, V. E., Spitzer, A. I., & Ramkumar, P. N. (2020). Machine Learning and Artificial Intelligence: Definitions, Applications, and Future Directions. Current Reviews in Musculoskeletal Medicine, 13(1), 69–76. https://doi.org/10.1007/s12178-020-09600-8

(Hinton et al., 2015) Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network. http://arxiv.org/abs/1503.02531

(Kass & Finin, 1988) Kass, Robert, and Tim Finin. 1988. "The Need for User Models in Generating Expert System Explanations." International Journal of Expert Systems 1 (4): 345–75. http://repository.upenn.edu/cisreports/585

(Kaur et al., 2022) Kaur, J., Goyal, A., Handa, P., & Goel, N. (2022). Solar power forecasting using ordinary least square based regression algorithms. 2022 IEEE Delhi Section Conference, DELCON 2022. https://doi.org/10.1109/DELCON54057.2022.9753619

(Kuzlu et al., 2020) Kuzlu, M., Cali, U., Sharma, V., & Güler, Ö. (2020). Gaining Insight Into Solar Photovoltaic Power Generation Forecasting Utilizing Explainable Artificial Intelligence Tools. *IEEE Access*, *8*, 187814–187823. https://doi.org/10.1109/ACCESS.2020.3031477

(Kwekha-Rashid et al., 2021) Kwekha-Rashid, Ameer Sardar, Heamn N Abduljabbar, and Bilal Alhayani. 2021. "Coronavirus Disease (COVID-19) Cases Analysis Using Machine-Learning Applications." Applied Nanoscience. https://doi.org/10.1007/s13204-021-01868-7

(Lakkaraju & Bastani, 2020) Lakkaraju, H., & Bastani, O. (2020). "how do i fool you?": Manipulating user trust via misleading black box explanations. AIES 2020 - Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, 79–85. https://doi.org/10.1145/3375627.3375833

(Letzgus et al., 2021)	Letzgus, S., Wagner, P., Lederer, J., Samek, W., Müller, K.-R., & Montavon, G. (2021). Toward Explainable AI for Regression Models. http://arxiv.org/abs/2112.11407
(Linardatos et al., 2021)	Linardatos, P., Papastefanopoulos, V., & Kotsiantis, S. (2021). Explainable ai: A review of machine learning interpretability methods. Entropy, 23(1), 1–45. https://doi.org/10.3390/e23010018
(Lundberg & Lee, 2017)	Lundberg, S., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. http://arxiv.org/abs/1705.07874
(Lundberg et al., 2018)	Lundberg, S. M., Erion, G. G., & Lee, S.-I. (2018). Consistent Individualized Feature Attribution for Tree Ensembles. http://arxiv.org/abs/1802.03888
(Lyu & Liu, 2021)	Lyu, W., & Liu, J. (2021). Artificial Intelligence and emerging digital technologies in the energy sector. Applied Energy, 303(August), 117615. https://doi.org/10.1016/j.apenergy.2021.117615
(Machlev et al., 2022)	Machlev, R., Heistrene, L., Perl, M., Levy, K. Y., Belikov, J., Mannor, S., & Levron, Y. (2022). Explainable Artificial Intelligence (XAI) techniques for energy and power systems: Review, challenges and opportunities. Energy and AI, 9, 100169. https://doi.org/10.1016/j.egyai.2022.100169
(Miller, 2019)	Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. Artificial Intelligence, 267, 1–38. https://doi.org/10.1016/j.artint.2018.07.007
(Molnar, C. 2022)	Molnar, C. (2022). Interpretable Machine Learning: A Guide for Making Black Box Models Explainable (2nd ed.). https://christophm.github.io/interpretable-ml-book/, último acesso em 15 de setembro de 2022. [Online]
(Murdoch et al., 2019)	Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., & Yu, B. (2019). Definitions, methods, and applications in interpretable machine learning. Proceedings of the National Academy of Sciences of the United States of America, 116(44), 22071–22080. https://doi.org/10.1073/pnas.1900654116
(NAII Office et al., 2020)	National Artificial Intelligence Initiative Office. 2020. "ADVANCING TRUSTWORTHY AI," no. https://www.ai.gov/strategic-pillars/advancing-trustworthy-ai/, ultimo acesso em 15 de Janeiro de 2022. [Online].
(Neto et al. 2019)	Neto, Tavares, e Comissão Nacional de Proteção de Dados. 2019. "Processo 1 Processo 2," 100. https://www.cnpd.pt/umbraco/surface/cnpdDecision/download/121843, último acesso em 15 de janeiro de 2022. [Online]
(Online Statistics Education: A Free Resource for Introductory Statistics, 2022)	Online Statistics Education: A Multimedia Course of Study (http://onlinestatbook.com/). Project Leader: David M. Lane, Rice University. https://onlinestatbook.com/2/index.html, último acesso em 15 de setembro de 2022. [Online]

(Osička & Černoch, 2022) Osička, J., & Černoch, F. (2022). European energy politics after Ukraine: The road ahead. Energy Research & Social Science, 91, 102757. https://doi.org/10.1016/j.erss.2022.102757

(Páez et al., 2019) Páez, A. (2019). The Pragmatic Turn in Explainable Artificial Intelligence (XAI). Minds and Machines, 29(3), 441–459. https://doi.org/10.1007/s11023-019-09502-w

(Park et al., 2021) Park, S., Jung, S., Jung, S., Rho, S., & Hwang, E. (2021). Sliding window-based LightGBM model for electric load forecasting using anomaly repair. Journal of Supercomputing, 77(11), 12857–12878. https://doi.org/10.1007/s11227-021-03787-4

(Pinto et al., 2016) Pinto, T., Sousa, T. M., Praça, I., Vale, Z., & Morais, H. (2016). Support Vector Machines for decision support in electricity markets' strategic bidding. Neurocomputing, 172, 438–445. https://doi.org/10.1016/j.neucom.2015.03.102

(Pinto et al., 2012) Pinto, T., Sousa, T. M., & Vale, Z. (2012). Dynamic artificial neural network for electricity market prices forecast. INES 2012 - IEEE 16th International Conference on Intelligent Engineering Systems, Proceedings, 311–316. https://doi.org/10.1109/INES.2012.6249850

(Prasad et al., 2009) Prasad, N. R., Almanza-Garcia, S., & Lu, T. T. (2009). Anomaly detection. *Computers, Materials and Continua*, *14*(1), 1–22. https://doi.org/10.1145/1541880.1541882

(Puig & Carmona, 2019) Puig, Bernat Coma, and Josep Carmona. 2019. "Bridging the Gap between Energy Consumption and Distribution through Non-Technical Loss Detection." Energies 12 (9). https://doi.org/10.3390/en12091748

(Ramos et al., 2022) Ramos, D., Faria, P., Gomes, L., & Vale, Z. (2022). A Contextual Reinforcement Learning Approach for Electricity Consumption Forecasting in Buildings. IEEE Access, 10, 61366–61374. https://doi.org/10.1109/ACCESS.2022.3180754

(Ramos et al., 2020) Ramos, D., Faria, P., Vale, Z., Mourinho, J., & Correia, R. (2020). Industrial facility electricity consumption forecast using artificial neural networks and incremental learning. Energies, 13(18). https://doi.org/10.3390/en13184774

(Ribeiro et al., 2016) Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should i trust you?" Explaining the predictions of any classifier. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-August-2016*, 1135–1144. https://doi.org/10.1145/2939672.2939778

(Rozemberczki et al., 2022) Rozemberczki, B., Watson, L., Bayer, P., Yang, H.-T., Kiss, O., Nilsson, S., & Sarkar, R. (2022). The Shapley Value in Machine Learning. http://arxiv.org/abs/2202.05594

(Russell et al., 1995) Russell, S. J., & Norvig, P. (1995). Artificial intelligence: a modern approach. Englewood Cliffs, N.J., Prentice Hall.

(Santos et al. 1999) Santos, Jorge, Luíz Faria, Carlos Ramos, Zita A Vale, and Albino Marques. 1999. "Verification of Knowledge Based-Systems for Power System Control Centres." In IEA/AIE. https://doi.org/10.1007/978-3-540-48765-4_35

(Samuel et al., 1959)	Samuel, Arthur L. 1959. “Some Studies in Machine Learning Using the Game of Checkers.” IBM J. Res. Dev. 3: 210–29. https://doi.org/10.1147/rd.33.0210
(Sarker 2021)	Sarker, Iqbal H. 2021. “Machine Learning: Algorithms, Real-World Applications and Research Directions.” SN Computer Science 2 (3): 1–21. https://doi.org/10.1007/s42979-021-00592-x
(Schoenborn et al., 2019)	Schoenborn, Jakob Michael and Klaus-Dieter Althoff. “Recent Trends in XAI: A Broad Overview on current Approaches, Methodologies and Interactions.” ICCBR Workshops (2019).
(Scott et al. 1977)	Scott, A Carlisle, Will Iam, J Clancey, Randall Davis, Edward H Shortliffe, Cd C-, Fl, and T C -Li. 1977. “Explanation Capabilities of Knowledge-Based Production Systems.” American Jnl of Computational Linguistics., no. Microfiche 62
(Seltman, 2013)	Seltman, H. J. (2013). *Experimental Design and Analysis*. https://www.stat.cmu.edu/~hseltman/309/Book/Book.pdf, último acesso em 15 de setembro de 2022. [Online].
(Shapley, Lloyd S. et al., 1953)	Shapley, L. S.. "17. A Value for n-Person Games". *Contributions to the Theory of Games (AM-28), Volume II*, edited by Harold William Kuhn and Albert William Tucker, Princeton: Princeton University Press, 2016, pp. 307-318. https://doi.org/10.1515/9781400881970-018
(Shavlik, Jude 1990)	Shavlik, Jude, Thomas Dietterich, ed. 1990. Readings in Machine Learning. Morgan Kaufmann (June 15, 1990). https://www.amazon.com/Readings-Machine-Learning-Morgan-Kaufmann/dp/1558601430
(Slack et al. 2020)	Slack, Dylan, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. 2020. “Fooling LIME and SHAP.” In Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, 180–86. New York, NY, USA: ACM. https://doi.org/10.1145/3375627.3375830.
(Stefani et al., 2022)	Stefani, J. de, Ann, S. I., Bontempi, P. G., & Group, M. L. (2022). *Towards multivariate multi-step-ahead time series forecasting*
(Stuart Russel et al.,2009)	Stuart Russel, Peter Norving. 2009. Artificial Intelligence: A Modern Approach. Pearson; 3rd edition (December 1, 2009)
(Tomar et al., 2022)	Tomar, D., Tomar, P., Bhardwaj, A., & Sinha, G. R. (2022). Deep Learning Neural Network Prediction System Enhanced with Best Window Size in Sliding Window Algorithm for Predicting Domestic Power Consumption in a Residential Building. *Computational Intelligence and Neuroscience*, *2022*. https://doi.org/10.1155/2022/7216959
(Tschora et al., 2022)	Tschora, L., Pierre, E., Plantevit, M., & Robardet, C. (2022). Electricity price forecasting on the day-ahead market using machine learning. *Applied Energy*, *313*(March), 118752. https://doi.org/10.1016/j.apenergy.2022.118752
(Vale et al. 1999)	Vale, Zita A, Carlos Ramos, Nuno Malheiro, Jorge Santos, and Albino Marques. 1999. “Enabling Client-Server Explanation Facilities in a Real-Time Expert-System.” In IEA/AIE. https://doi.org/10.1007/978-3-540-48765-4_37

(van Lent et al., 2004)	Lent, Michael Van, William Fisher, and Michael Mancuso. 2004. “An Explainable Artificial Intelligence System for Small-Unit Tactical Behavior.” Proceedings of the National Conference on Artificial Intelligence, 900–907
(Verwiebe et al. 2021)	Verwiebe, Paul Anton, Stephan Seim, Simon Burges, Lennart Schulz, and Joachi Müller-Kirchenbauer. 2021. “Modeling Energy Demand—A Systematic Literature Review.” Energies (19961073) 14 (23): 7859. https://doi.org/10.3390/en14237859
(Vilone & Longo, 2020)	Vilone, G., & Longo, L. (2020). Explainable Artificial Intelligence: a Systematic Review. http://arxiv.org/abs/2006.00093
(Wastensteiner et al. 2021)	Wastensteiner, Jacqueline, Tobias M Weiss, Felix Haag, and Konstantin Hopf. 2021. “Explainable AI for Tailored Electricity Consumption Feedback – An Experimental Evaluation of Visualizations.” In , 1–19. Bamberg: Otto-Friedrich-Universität. https://doi.org/10.20378/irb-49912
(Wachter et al. 2018)	Wachter, S., Mittelstadt, B., & Russell, C. (2018). Counterfactual explanations without opening the black box: automated decisions and the GDPR. Harvard Journal of Law and Technology, 31(2), 841–887. https://arxiv.org/abs/1711.00399
(Winston 1970)	Winston, Patrick Henry. 1970. “Learning Structural Descriptions From Examples.” In . https://dl.acm.org/doi/book/10.5555/889456#issue-downloads,
(Xiong et al., 2022)	Xiong, P., Buffett, S., Iqbal, S., Lamontagne, P., Mamun, M., & Molyneaux, H. (2022). Towards a robust and trustworthy machine learning system development: An engineering perspective. *Journal of Information Security and Applications*, *65*. https://doi.org/10.1016/j.jisa.2022.103121
(Zafar and Khan 2019)	Zafar, Muhammad Rehman, and Naimul Mefraz Khan. 2019. “DLIME: A Deterministic Local Interpretable Model-Agnostic Explanations Approach for Computer-Aided Diagnosis Systems.” Machine Learning and Knowledge Extraction 3 (3): 525–41. https://doi.org/10.3390/make3030027
(Zhou et al. 2021)	Zhou, Jianlong, Amir H. Gandomi, Fang Chen, and Andreas Holzinger. 2021. “Evaluating the Quality of Machine Learning Explanations: A Survey on Methods and Metrics.” Electronics (Switzerland) 10 (5): 1–19. https://doi.org/10.3390/electronics10050593