UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

ULISSES VIANA TAVEIRA

APLICAÇÃO DE FLORESTAS ALEATÓRIAS E TÉCNICAS DE INTERPRETABILIDADE NA ANÁLISE DE DADOS BIOLÓGICOS E CLÍNICOS

Ouro Preto, MG
2024

ULISSES VIANA TAVEIRA

APLICAÇÃO DE FLORESTAS ALEATÓRIAS E TÉCNICAS DE INTERPRETABILIDADE NA ANÁLISE DE DADOS BIOLÓGICOS E CLÍNICOS

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Dr. Rafael Alves Bonfim de Queiroz

Ouro Preto, MG
2024

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E BIOLOGICAS
DEPARTAMENTO DE COMPUTACAO

FOLHA DE APROVAÇÃO

Ulisses Viana Taveira

Aplicação de Florestas Aleatórias e Técnicas de Interpretabilidade na Análise de Dados Biológicos e Clínicos

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 10 de Outubro de 2024.

Membros da banca

Rafael Alves Bonfim de Queiroz (Orientador) - Doutor - Universidade Federal de Ouro Preto
Danielle Emely de Souza Almeida (Examinadora) - Bacharel - PPG Biotecnologia - Universidade Federal de Ouro Preto
Cristiano Amaro da Matta (Examinador) - Bacharel - Universidade Federal de Ouro Preto

Rafael Alves Bonfim de Queiroz, Orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 10/10/2024.

Documento assinado eletronicamente por **Rafael Alves Bonfim de Queiroz**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 17/10/2024, às 02:51, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0789768** e o código CRC **5DCC8E11**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591692 - www.ufop.br

Dedico este trabalho à minha mãe Fátima e à minha avó Elzira, que sempre acreditaram em mim e me deram todo apoio.

Agradecimentos

Agradeço a minha família, em particular aos meus pais e a minha avó Elzira por todo o apoio que me deram ao longo dessa jornada. Agradeço a todos os amigos e colegas que tive o prazer de encontrar em Ouro Preto. Agradeço ao meu orientador Rafael Alves Bonfim de Queiroz pela orientação e paciência para a conclusão deste trabalho. Agradeço a Universidade Federal de Ouro Preto, ao seu Departamento de Computação e a seus professores pelo ensino público de qualidade.

Resumo

O aprendizado de máquina está presente em muitas áreas do conhecimento, como a Biologia e a Medicina. Apesar da ampla aplicação de modelos de aprendizado para previsões existe a questão de interpretar os modelos de modo que seja possível a um ser humano compreender seu funcionamento, uma vez que muitos modelos são "caixas pretas". A falta de conhecimento acerca do funcionamento da tomada de decisões de um modelo pode suscitar a replicação de erros, falta de aceitação social e problemas éticos e morais, que são fruto da sua aplicação em produtos para a sociedade. Dois interpretadores agnósticos ao modelo, LIME (*Local Interpretable Model-Agnostic Explanations*) e SHAP (*Shapley Additive Explanations*), oferecem interpretação a modelos de aprendizado. O objetivo central deste trabalho consiste em aplicar os métodos LIME e SHAP para interpretação de modelos de aprendizado gerados com o algoritmo Floresta Aleatória para casos de classificação em uma base de dados de botânica (Iris) e uma base de dados médica (Diabetes), e apresentar um panorama do funcionamento dos modelos, dos interpretadores e comparar os resultados de ambos os métodos de interpretabilidade.

Palavras-chave: Aprendizado de Máquina. Interpretabilidade. Floresta Aleatória. LIME. SHAP.

Abstract

Machine learning is present in many areas of knowledge, such as Biology and Medicine. Despite the wide application of learning models for predictions, there is the issue of interpreting the models so that it is possible for a human being to understand how they work since many models are "black boxes." The lack of knowledge about how a model's decision-making works can lead to the replication of errors, lack of social acceptance, and ethical and moral problems resulting from its application in products for society. Two model-agnostic interpreters, LIME (Local Interpretable Model-Agnostic Explanations) and SHAP (Shapley Additive Explanations), provide interpretation for learning models. The central objective of this work is to apply the LIME and SHAP methods to learning models generated with the Random Forest algorithm for classification cases in a botanical database (Iris) and a medical database (Diabetes) and present an overview of the functioning of the models and interpreters and compare the results of both interpretability methods.

Keywords: Machine Learning, Interpretability, Random Forest, LIME, SHAP.

Lista de Ilustrações

Lista de Abreviaturas e Siglas

LIME	*Local Interpretable Model-agnostic Explanation*
SHAP	*SHapley Additive exPlanations*
IA	*Inteligência Artificial*
ML	*Machine Learning*

Sumário

1 Introdução

Machine Learning (ML), ou seja, o Aprendizado de Máquina, é um subcampo da IA (Inteligência Artificial) que lida com algoritmos de computação que podem ser melhorados via dados de treinamento sem programação explícita. Eles ajudam a explorar e analisar conjuntos de dados complexos e ver sentido neles. Cada algoritmo é um conjunto finito de instruções passo a passo não ambíguas que um computador pode seguir para atingir determinado objetivo. Em um modelo de ML, o objetivo é estabelecer ou descobrir padrões que podem ser utilizados para fazer previsões ou categorizar informações. Diferentes algoritmos de ML utilizam diferentes técnicas para analisar dados, e são agrupados pelas técnicas de aprendizado para as quais são utilizados, geralmente divididos em três grupos: aprendizado supervisionado, aprendizado não supervisionado, e aprendizado por reforço. Neste trabalho, o método de ML estudado é chamado *Random Forest*, ou seja, a Floresta Aleatória, um algoritmo de aprendizado supervisionado desenvolvido por Breiman (2001) e apresentado no artigo *Random Forests*, baseado no trabalho anterior de Amit e Geman (1997). O algoritmo de Floresta Aleatória cria muitas árvores de decisão, utilizando um conjunto de dados aleatórios para cada árvore a partir do conjunto de dados originais, formando o que podemos enxergar como uma floresta, onde os resultados obtidos por cada árvore será utilizado no cálculo do resultado final, em uma espécie de votação.

Um problema que surge com o desenvolvimento e uso dos algoritmos de ML é o que se chama de interpretabilidade. A interpretabilidade é a capacidade humana de explicar como os algoritmos tomam as decisões que tomam, ou de como eles chegam aos resultados (MILLER, 2019). A possibilidade de explicar as razões e fundamentar as decisões desses algoritmos é muito importante em vários aspectos, que são científicos, sociais e morais. Será apresentado mais a frente um capítulo acerca da interpretabilidade e seus métodos. A interpretabilidade pode ser alcançada através da aplicação de métodos de interpretabilidae aos modelos de aprendizado. Existem vários desses métodos, que variam de acordo com a natureza dos algoritmos. Neste trabalho serão apresentados e utilizados dois métodos de interpretabilidade: o LIME (*Local Interpretable Model-Agnostic Explanations*) e o SHAP (*Shapley Additive Explanations*).

O LIME, ou seja, um interpretador local de explicações agnósticas de modelo foi proposto por Ribeiro, Singh e Guestrin (2016) no artigo *Why Should I Trust You? Explaining the Predictions of Any Classifier*, sendo um algoritmo para prover interpretabilidade para todo os tipos de modelos de aprendizado.

O SHAP foi proposto por Lundberg e Lee (2017) no artigo *A unified approach to interpreting model predictions*. Os valores SHAP são baseados nos valores *Shapley* da teoria dos jogos. Na teoria dos jogos, os valores de Shapley ajudam a determinar quanto cada jogador em um jogo colaborativo contribuiu para o pagamento total. Para um modelo de aprendizado, cada

atributo é considerado um “jogador”. O valor *Shapley* para um atributo representa a magnitude média da contribuição desse atributo em todas as combinações possíveis de atributos.

A aplicação de interpretadores a modelos de aprendizado tem um papel fundamental na compreensão dos resultados, mas também no funcionamento dos próprios modelos gerados (GARREAU; LUXBURG, 2020).

O ponto central deste trabalho é explorar a aplicação do LIME e do SHAP para a interpretação de modelos de Floresta Aleatória. Isso será feito com duas bases de dados distintas, que serão utilizadas para treinar os modelos de aprendizado, e posteriormente a interpretação de ambos os modelos utilizando os métodos LIME e SHAP.

1.1 Justificativa

Os algoritmos de ML têm se tornado cada vez mais importantes para a tomada de decisões em várias áreas (ZENG; USTUN; RUDIN, 2017), desde aplicações simples como sugerir um filme, até aplicações que podem ser classificadas como sensíveis - que podem colocar a vida das pessoas em risco por exemplo - como dirigir um carro ou fazer um diagnóstico médico. O grande problema com esses modelos de aprendizado é que em muitos casos, principalmente em tarefas mais sofisticadas, eles são o que se chama de “caixa preta”, no sentido de que o processo pelo qual tais modelos fazem previsões pode ser difícil para os humanos compreenderem.

A falta de conhecimento acerca do funcionamento da tomada de decisões de um modelo de aprendizado pode suscitar problemas éticos e morais, que são fruto da sua aplicação em produtos para a sociedade. Por motivos como esses a interpretabilidade se faz necessária (NARAYANAN et al., 2018), para verificarmos se erros, mas também preconceitos e discriminações, não estão sendo propagados pelo modelo.

Considerando esse cenário, torna-se muito importante a necessidade de explicar como os modelos fazem as previsões, quais parâmetros ou características estão sendo considerados e qual o peso dado a elas. Isso é importante para aumentar a confiança dos usuários no modelo e a aceitabilidade social das aplicações do modelo (LIPTON, 2018). É importante ressaltar que se a explicação das decisões do modelo puder ser garantida, então outras características importantes também poderão ser verificadas, como a imparcialidade e não discriminação na tomada de decisões, garantir a proteção de dados confidenciais e a fidelidade e robustez do modelo (ZHANG et al., 2019).

1.2 Objetivos

O objetivo geral deste trabalho consiste em aplicar os métodos de interpretabilidade LIME e SHAP a modelos de aprendizado gerados com o algoritmo Floresta Aleatória para casos de classificação utilizando duas bases de dados distintas: a base de dados Iris e a base de dados Diabetes. A partir das previsões feitas com os dois modelos é apresentada uma comparação entre os resultados de ambos os métodos de intrepretabilidade. Os objetivos específicos podem ser divididos em:

- Explorar os conceitos básicos de ML e interpretabilidade;
- Explorar os conceitos do algoritmo Floresta Aleatória para modelos de aprendizado;
- Treinar dois modelos de Floresta Aleatória com as bases de dados Iris e Diabetes;
- Aplicar os métodos LIME e SHAP aos modelos previamente treinados para interpretar as previsões locais;
- Comparar as interpretações locais obtidas com os dois métodos LIME e SHAP.

1.3 Organização do Trabalho

No Capítulo 2 é apresentada uma revisão bibliográfica acerca dos trabalhos correlatos, e uma fundamentação teórica que engloba vários tópicos relacionados ao tema de estudo: uma breve apresentação da área de ML, uma introdução a área de interpretabilidade, a apresentação dos tipos de métodos de interpretabilidade agnósticos aos modelos, tanto os globais quando os locais. No Capítulo 3 são apresentados o algoritmo Floresta Aleatória e os métodos LIME e SHAP. No Capítulo 4 são apresentados os casos de treinamento dos modelos de Floresta Aleatória para as bases de dados Iris e Diabetes, com uma explicação do conjunto de dados de treinamento e das previsões esperadas. Em seguida, é apresentada a aplicação dos métodos LIME e SHAP aos modelos. No Capítulo 5 é apresentada a conclusão do trabalho.

2 Revisão Bibliográfica

Este capítulo apresenta uma visão geral da área de ML e dos métodos de interpretabilidade de modelos de aprendizado, além de trabalhos relacionados, com a finalidade de fundamentar teoricamente os conceitos centrais do presente trabalho.

2.1 Trabalhos Relacionados

Os modelos de ML são adotados para fazer todo tipo de previsões, que vão da recomendação de filmes ao diagnóstico médico, apesar de muitos deles serem o que se chama de "caixa preta", ou seja, não se sabe muito bem o que acontece entre a entrada dos dados e a saída das previsões. Isso pode gerar uma desconfiança em relação ao modelo e aos *insights* apresentados por este. Entre os principais métodos de interpretação de modelos de aprendizado existem o LIME (RIBEIRO; SINGH; GUESTRIN, 2016), e o SHAP (LUNDBERG; LEE, 2017). Estes dois métodos em particular têm sido objeto de estudo conjuntamente devido a sua técnica similar baseada em perturbações de entrada, para fins de comparação e verificação da confiabilidade dos métodos (SLACK et al., 2020).

Os algoritmos de aprendizado que podem ser interpretados pelos métodos de interpretabilidade são variados, como Redes Neurais, Floresta Aleatória, Árvore de Decisão entre outros. E os modelos criados a partir desses algoritmos podem trabalhar com texto, imagem, ou dados tabulares, e a interpretabilidade fornecida por métodos como LIME e SHAP precisam lidar com cada um destes tipos de dados (MARDAOUI; GARREAU, 2021). A busca por fornecer mais transparência aos algoritmos de aprendizado tem gerado um grande interesse por pesquisa focada em explicar explicitamente decisões ou ações dos modelos para os seres humanos (MILLER, 2019).

A interpretabilidade de modelos de aprendizado leva em consideração a robustez, fidelidade e a interpretabilidade, que também pode ser chamada de interpretabilidade humana, que garante a facilidade com que um humano interpreta o resultado da explicação do método. Aspectos como a robustez e a fidelidade medem se instâncias de entrada semelhantes obtêm explicações semelhantes (PLUMB; MOLITOR; TALWALKAR, 2018; ALVAREZ-MELIS; JAAKKOLA, 2018). É necessário introduzir métricas para quantificar a robustez e demonstrar que os métodos atuais não têm um bom desempenho de acordo com essas métricas, e propor maneiras pelas quais a robustez pode ser aplicada em abordagens como LIME e SHAP. Muitos trabalhos defendem a importância da robustez. Ela geralmente é garantida pelo cálculo direto de quanto a saída de um método de explicação muda com sua entrada (MELIS; JAAKKOLA, 2018) ou mostrando a sensibilidade das explicações a ataques adversários (GHORBANI; ABID; ZOU, 2019).

Vários trabalhos avaliam a fidelidade como medida da qualidade de uma explicação. A fidelidade é avaliada diretamente (TAN et al., 2018) medindo as diferenças nas previsões do substituto e modelos explicados, bem como indiretamente (RIBEIRO; SINGH; GUESTRIN, 2018), medindo como bem, um ser humano pode prever a saída de um sistema de ML com e sem ser exposto a explicações. Novamente, esta é outra métrica desvinculada do impacto no mundo real de mostrar uma explicação para um determinada pessoa, pois se concentra em quão bem o modelo se aproxima a função aprendida pelo modelo de ML original.

A busca pela interpretabilidade de modelos, considerando sua robustez, fidelidade e a própria interpretabilidade humana, ainda não encontrou um consenso em relação aos métodos de mediar essas propriedades. Alguns *frameworks* foram desenvolvidos (MOHSENI; ZAREI; RAGAN, 2021) para tentar enfrentar o desafio da avaliação dos modelos, mas ainda não tiveram ampla aceitação. Fica claro que falta uma maneira sistemática e objetiva de comparar métodos (LIPTON, 2017), e existe um grau de liberdade para que cada pesquisa utilize suas métricas e objetivos para estabelecer comparação entre métodos. A importância dessa escolha de métodos é particularmente justificada quando se trata de tomada de decisões que têm impacto direto na vida das pessoas. O caminho mais interessante para estabelecer métricas está relacionado em como uma explicação pode ajudar o usuário final para melhor executar sua tarefa (JESUS et al., 2021). A interpretabilidade também pode ser avaliada medindo quão aproximada uma explicação de um método está associada a uma explicação produzida por um especialista humano.

2.2 Fundamentação Teórica

2.2.1 Aprendizado de Máquina

O ML é um subcampo da inteligência artificial que lida com algoritmos de computação que podem ser melhorados via dados de treinamento sem programação explícita. Podemos definir ML como um conjunto de métodos computacionais utilizados para fazer previsões baseados em um determinado conjunto de dados. Os algoritmos de aprendizado usam parâmetros baseados em dados de treinamento, geralmente um subconjunto de dados pertencentes a um conjunto maior. A medida que os dados de treinamento se expandem para representar o mundo de modo mais realista, o algoritmo calcula resultados mais precisos. Diferentes algoritmos analisam dados de maneiras diferentes. Geralmente, eles são agrupados pelas técnicas de aprendizado para as quais são usados: aprendizado supervisionado, aprendizado não supervisionado e aprendizado de reforço. Os algoritmos usados com mais frequência (SHINDE; SHAH, 2018) usam a regressão e a classificação para prever categorias de destino e valores.

Por exemplo, pode-se utilizar uma base de dados de empréstimos bancários para treinar um algoritmo com o intuito de, inserindo os dados bancários de uma determinada pessoa, determinar se ela deve ter acesso a um empréstimo ou não. Esse tipo de aprendizado, onde sabemos qual o resultado nos interessa, chama-se aprendizado supervisionado. Na sua construção são feitas

entradas de rótulos e exemplos atuais a respectiva saída desejada e isso permite ao algoritmo aprender as regras que mapeiam entradas e saídas. O objetivo do aprendizado supervisionado é aprender um modelo preditivo que mapeia características dos dados para uma saída; se a saída for categórica, a tarefa é chamada de classificação e, se for numérica, é chamada de regressão (MOLNAR, 2020).

As técnicas de aprendizado de máquinas, de maneira simplificada, se enquadram em uma dessas três técnicas:

- Aprendizado supervisionado: os algoritmos fazem previsões com base em um conjunto de exemplos rotulados fornecidos previamente. Essa técnica é útil quando se sabe como deve ser o resultado;
- Aprendizado não supervisionado: os rótulos não são fornecidos e, portanto, o algoritmo pode encontrar sua própria estrutura de processamento das entradas. Os pontos de dados não são rotulados, o algoritmo os rotula para que seja possível organizar os dados ou descrever a estrutura deles. Essa técnica é útil quando não se sabe como deve ser o resultado;
- Aprendizado por reforço: o algoritmo interage repetidamente com um ambiente dinâmico com um objetivo específico como, por exemplo, ganhar um jogo ou dirigir um carro. Após cada ação, o algoritmo recebe comentários que o ajudam a determinar se a escolha feita foi correta, neutra ou incorreta. É uma boa técnica a ser usada para sistemas automatizados que precisam tomar muitas decisões simples sem diretrizes humanas.

Um algoritmo de aprendizado aprende um modelo estimando parâmetros ou estruturas, e guiado por uma função que calcula pontuação ou minimiza perdas. No exemplo do empréstimo financeiro apresentado acima, primeiro é feita uma base de dados com o histórico de empréstimos anteriores, assim como outras informações que sejam relevantes para compreender o fenômeno. Depois o algoritmo é alimentado com esses dados para gerar um modelo de previsão de valor, esse modelo com novos dados pode ser integrado a um site de uma instituição financeira ou ser utilizado em um banco. O aprendizado de máquina é uma mudança de paradigma da programação normal, onde todas as instruções devem ser explicitamente dadas ao computador, para a programação indireta, que ocorre por meio do fornecimento de dados. Apesar da simplificação nas três técnicas de aprendizado apresentadas, de modo mais aprofundado elas podem ser subdivididas em muitas outras que podem ser encontradas na bibliografia da área (ver Figura 2.1).

2.2.2 Interpretabilidade de Modelos de Aprendizado de Máquina

Em aprendizado de máquina, a interpretabilidade diz respeito a facilidade que um ser humano tem de compreender porque o modelo tomou certas decisões ou fez as previsões que fez, isso implica que um modelo é mais interpretável do que outro se suas decisões são mais fáceis de serem compreendidas por nós humanos do que as decisões de outro modelo. Ou como Miller

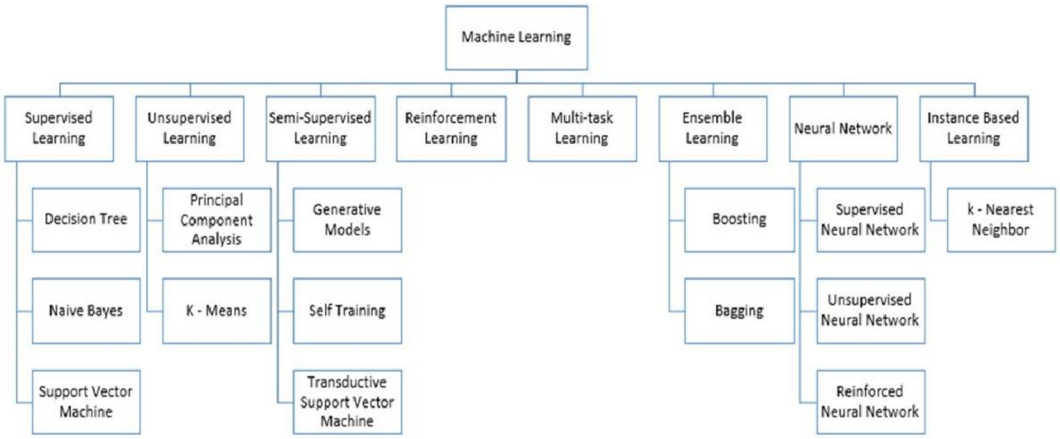

Figura 2.1 – Técnicas de aprendizado de máquina. Fonte: (MAHESH, 2020).

(2019) mencionou: interpretabilidade é o grau em que um ser humano pode entender a causa de uma decisão.

A interpretabilidade é crucial para a adoção de negócios, documentação do modelo, supervisão regulatória e aceitação e confiança humana. Em alguns casos, não seria necessário compreender o porquê do modelo fazer as previsões que fez, como nos ambientes de baixo risco, onde as decisões tomadas a partir dos resultados do modelo não teriam consequências graves, como em um sistema de recomendação de filmes. Mas em outros casos, é necessária a garantia da previsão que o modelo fez, como problemas que podem colocar a vida de pessoas em risco, como diagnósticos médicos, e para determinados problemas não é apenas importante entender o porquê das decisões do modelo por uma questão de segurança, mas faz parte do resultados compreender como se chegou até ele. Além disso, compreender a tomada de decisão pode ajudar a prever erros e melhorar o desempenho do modelo. Por fim, há a curiosidade humana, que nos leva querer compreender o porquê das coisas, e nesse caso não seria diferente.

2.2.2.1 Métodos Agnósticos a Modelo

A interpretabilidade de modelos em aprendizado de máquina é particularmente importante nos casos de "caixa preta", onde se conhece bem a entrada e a saída mas não há uma clareza em relação a como o modelo toma as decisões. Nesses casos, são utilizados os métodos agnósticos a modelo (não específicos para um determinado algoritmo) que utilizam os dados de entrada e saída para compreender internamente o algoritmo. Eles podem ser globais ou locais. Os métodos globais descrevem como os atributos afetam a previsão em média. Em contraste, os métodos locais visam explicar as previsões individuais.

2.2.2.1.1 Métodos Globais

Os métodos globais descrevem o comportamento médio de um modelo de aprendizado de máquina. Os métodos globais são frequentemente expressos como valores esperados com base

na distribuição dos dados. Por exemplo, o gráfico de dependência parcial, um gráfico de efeito de recurso, é a previsão esperada quando todos os outros atributos são marginalizados. Como os métodos de interpretação global descrevem o comportamento médio, eles são particularmente úteis quando o modelador deseja entender os mecanismos gerais nos dados ou depurar um modelo (MOLNAR, 2020). Abaixo temos algumas das principais técnicas de interpretação global:

- Gráfico de dependência parcial: mostra o efeito médio de um atributo sobre a previsão do modelo, mantendo os demais fixos;
- Gráficos de efeitos locais acumulados: capturam o impacto de um atributo nas previsões ao longo do espaço de valores, considerando variações locais;
- Interação de atributos (estatística H): quantifica o quanto dois atributos interagem na explicação do modelo além de seus efeitos individuais;
- Decomposição funcional: divide a previsão do modelo em contribuições atribuídas a cada atributo ou conjunto de atributos;
- Importância do atributo de permutação: mede a queda de desempenho do modelo ao embaralhar um atributo, indicando sua relevância;
- Modelos substitutos globais: criam um modelo simples e interpretável para aproximar o comportamento do modelo original complexo;
- Protótipos e críticas: selecionam exemplos representativos (protótipos) e casos atípicos (críticas) para explicar o comportamento dos dados e do modelo.

2.2.2.1.2 Métodos Locais

Os métodos de interpretação local explicam as previsões individuais. Abaixo temos alguns dos principais métodos locais:

- Curvas de expectativas condicionais individuais: mostram como a previsão do modelo varia para um único exemplo ao modificar apenas um atributo;
- Modelos substitutos locais (LIME): aproximam localmente o modelo complexo por um modelo simples e interpretável para explicar a previsão;
- Regras de escopo (âncoras): fornecem regras *if-then* que, quando satisfeitas, garantem previsões estáveis do modelo;
- Explicações contrafactuais: indicam quais mudanças mínimas nos atributos alterariam a decisão do modelo para outro resultado;

- Valores *Shapley*: distribuem a previsão de forma justa entre os atributos, considerando todas as possíveis combinações;
- *SHAP*: aplica valores de *Shapley* em modelos de machine learning, atribuindo importância consistente e aditiva a cada atributo.

3 Desenvolvimento

Neste capítulo, são apresentadas as definições do algoritmo de aprendizado Floresta Aleatória, e dos métodos de interpretação LIME e SHAP.

3.1 Floresta Aleatória

Floresta Aleatória é um algoritmo de aprendizado supervisionado para uso em classificação e regressão (BREIMAN, 2001). Esse algoritmo é composto por uma coleção de árvores de decisão, e cada árvore do conjunto é composta por uma amostra de dados extraída de um conjunto de treinamento com reposição, chamada de amostra *bootstrap*. Da amostra de treinamento, uma parte é reservada como dado para teste. Outra instância de aleatoriedade é injetada por meio de *feature bagging*, adicionando mais diversidade ao conjunto de dados e reduzindo a correlação entre as árvores de decisão. Dependendo do tipo de problema, a determinação da previsão irá variar. Para uma tarefa de regressão, será calculada a média das árvores de decisão individuais e, para uma tarefa de classificação, será obtida uma votação majoritária, ou seja, a variável categórica mais frequente produzirá a classe prevista. Finalmente, a amostra de teste é então usada para validação cruzada, finalizando essa previsão.

3.1.1 Aprendizagem em Conjunto

Floresta Aleatória é baseada no princípio de aprendizagem em conjunto, onde múltiplos modelos (neste caso, árvores de decisão) são treinados para resolver o mesmo problema (ver Figura 3.1). A ideia é que a combinação de vários decisores fracos (árvores individuais) pode levar a um modelo geral mais forte. Essa floresta é um grupo de árvores de decisão, no entanto existem algumas diferenças entre os dois. Uma árvore de decisão tende a criar regras, que ela usa para tomar decisões. Já floresta escolherá aleatoriamente os atributos e fará observações, construirá o conjunto de árvores de decisão e, em seguida, calculará a média dos resultados.

A teoria é que um grande número de árvores não correlacionadas criará previsões mais precisas do que uma árvore de decisão individual. Isso ocorre porque o volume de árvores trabalha em conjunto para proteger cada uma de erros individuais e *overfitting*. Um algoritmo de Floresta Aleatória pode ser usado em um cenário em que há uma variedade de dados de entrada e um conjunto complexo de circunstâncias. Do ponto de vista da implementação, os algoritmos de Floresta Aleatória possuem três hiperparâmetros principais, que precisam ser definidos antes do treinamento. Isso inclui o tamanho do nó, o número de árvores e o número de atributos amostrados. A partir daí, o classificador pode ser usado para resolver problemas de regressão ou classificação.

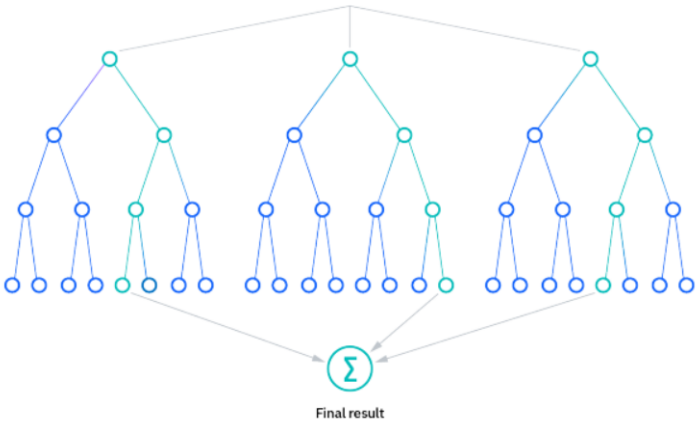

Figura 3.1 – Soma das árvores de decisão na Floresta Aleatória.

Fonte: *What is random forest?* Disponível em: ibm.com/topics/random-forest. Acesso em: 20/10/2024.

3.1.2 Árvores de Decisão

Cada modelo individual em uma Floresta Aleatória é uma árvore de decisão. Uma árvore de decisão é construída dividindo recursivamente os dados em subconjuntos com base em valores de atributos. O objetivo é criar subconjuntos homogêneos onde a variável alvo seja tão pura quanto possível. Para cada subconjunto, uma árvore de decisão é construída. Ao contrário das árvores de decisão tradicionais, a Floresta Aleatória introduz a aleatoriedade de duas maneiras: Ao dividir um nó, ele seleciona um subconjunto aleatório de atributos em vez de considerar todos eles. Isso ajuda a reduzir a correlação entre as árvores. Cada árvore pode crescer até sua profundidade total sem poda, o que ajuda a capturar padrões complexos (SPEISER et al., 2019).

Os critério de divisão para tarefas de classificação comumente incluem os dois critérios abaixo:

- Impureza de Gini: para decidir qual atributo será o nó raiz é preciso entender algo chamado índice de Gini. Para selecionar um atributo para dividir ainda mais, é preciso saber quão impura ou pura será essa divisão. Uma subdivisão pura significa que deve receber sim ou não. Basicamente, é preciso saber a impureza do conjunto de dados e considerar esse atributo como o nó raiz que fornece a impureza mais baixa, ou digamos, qual tem o índice de Gini mais baixo. Matematicamente, o índice de Gini pode ser escrito como:

$$\begin{aligned} Gini(D) =& 1 - \sum_{i=1}^{n} (p_i)^2 \\ =& 1 - [(p_+)^2 + (p_-)^2], \end{aligned} \tag{3.1}$$

  sendo que p_+ é a probabilidade de uma classe positiva e p_- é a probabilidade de uma classe negativa.

- Entropia:

$$Entropy(D) = -\sum_{i=1}^{n} p_i log_2(p_i), \tag{3.2}$$

nos quais p_i é a proporção da classe i no conjunto de dados D e n é o número de classes.

3.1.3 Agregação *Bootstrap*

O *bootstrapping* cria vários subconjuntos de dados de treinamento, onde cada subconjunto é gerado por amostragem aleatória dos dados com substituição, o que significa que alguns pontos de dados podem ser repetidos enquanto outros podem ser omitidos. Esses subconjuntos têm origem na base de dados original utilizada para treinar o modelo. A amostragem *bootstrap* é feita a partir de amostra aleatória de n instâncias do conjunto de dados de treinamento D com substituição para criar B conjuntos de dados de *bootstrap* D_b.

A construção das árvores é feita a partir do conjunto de dados de *bootstrap*, treinando uma árvore de decisão usando um subconjunto aleatório de atributos em cada divisão. O número de atributos considerados para divisão em cada nó é normalmente $m = \sqrt{p}$ para classificação, no qual p é o número total de atributos.

O *bagging*, também conhecido como agregação *bootstrap*, permite que as árvores de decisão individuais obtenham amostras aleatoriamente do conjunto de dados e substituam os dados, criando resultados muito diferentes em árvores individuais. Isso significa que, em vez de incluir todos os dados disponíveis, cada árvore recebe apenas alguns dos dados. Essas árvores individuais tomam decisões com base nos dados que possuem e preveem resultados com base apenas nesses pontos de dados (ALTMAN; KRZYWINSKI, 2017).

Isso significa que em cada Floresta Aleatória, existem árvores que são treinadas em dados diferentes e que usaram atributos diferentes para tomar decisões. Isso fornece um *buffer* para as árvores, protegendo-as de erros e previsões incorretas. O processo de *bagging* usa apenas cerca de dois terços dos dados, de modo que o terço restante pode ser usado como um conjunto de teste. Em geral, a Floresta Aleatória é poderosa porque combina os pontos fortes de muitas árvores de decisão enquanto minimiza seus pontos fracos, levando a um desempenho robusto em vários conjuntos de dados.

3.1.4 Combinando Previsões

Para tarefas de classificação, cada árvore na floresta gera um rótulo de classe, e a previsão final é feita com base na votação majoritária (a classe que obtém mais votos). Para tarefas de regressão, é calculada a média das previsões de todas as árvores para produzir um resultado final. Uma vez que B árvores são construídas, as previsões são feitas da seguinte forma:

- Para classificação: Cada árvore T_b fornece um rótulo de classe, e a previsão final é determinada por votação majoritária:

$$\hat{y} = model(T_1(x), T_2(x), ..., T_B(x)). \tag{3.3}$$

- Para regressão: a previsão final é a média das previsões de todas as árvores:

$$\hat{y} = \frac{1}{B} \sum_{b=1}^{B} T_b(x). \tag{3.4}$$

3.1.5 Reduzindo o *Overfitting*

Ao calcular a média das previsões de múltiplas árvores e usar a seleção aleatória de atributos, a Floresta Aleatória reduz o risco de *overfitting* (sobreajuste) que geralmente ocorre em árvores de decisão única. Isto se deve à alta variação das árvores individuais, que é mitigada pela abordagem de conjunto. A aleatoriedade introduzida tanto na seleção de dados quanto na seleção de atributos ajuda a melhorar a generalização para dados não vistos. A Floresta Aleatória também pode fornecer *insights* sobre a importância do atributo, que geralmente é calculada usando medidas como a diminuição da impureza de Gini ou a diminuição média da precisão.

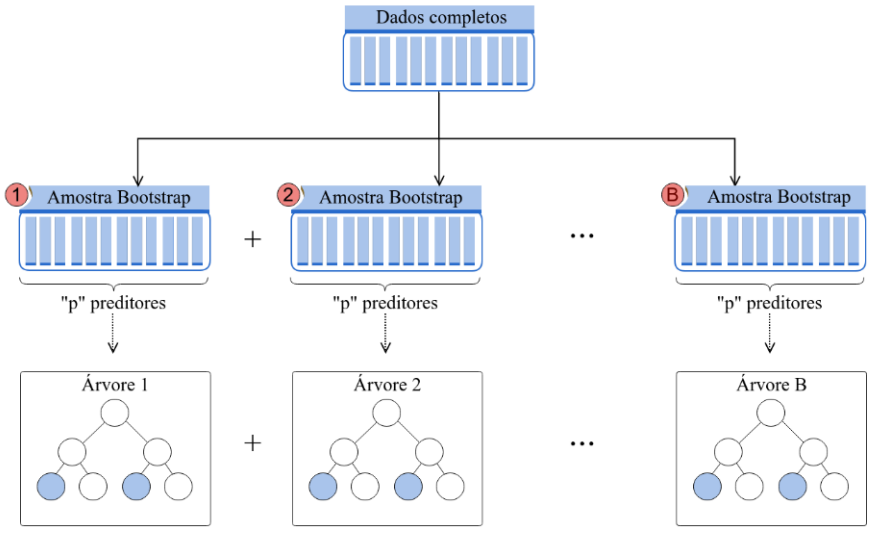

Figura 3.2 – Sequência de criação de um modelo de aprendizado com Floresta Aleatória.

Fonte: LEG aulas. Disponível em: cursos.leg.ufpr.br. Acesso em: 20/10/2024.

Em resumo, o algoritmo Floresta Aleatória consiste em múltiplas árvores de decisão construídas com diferentes subconjuntos de dados e atributos, com suas previsões combinadas para produzir um resultado final (ver Figura 3.2). Este método aproveita a força do aprendizado conjunto para aumentar a precisão preditiva e a robustez contra *overfitting*.

3.2 LIME

A descrição do LIME apresentada nesta seção são baseadas nas referências (MOLNAR, 2020; RIBEIRO; SINGH; GUESTRIN, 2016).

O LIME é uma ferramenta de aprendizado independente de modelo que ajuda a interpretar modelos de ML (RIBEIRO; SINGH; GUESTRIN, 2016). O termo independente de modelo significa que o LIME pode ser utilizado com qualquer modelo de aprendizado ao treinar seus dados e interpretar os resultados. LIME usa modelos inerentemente interpretáveis, como árvores de decisão, modelos lineares e modelos heurísticos baseados em regras para explicar resultados a usuários não técnicos com formas visuais. É possível usar o LIME para problemas de regressão e classificação para interpretar modelos de "caixa preta".

O LIME propõe uma implementação concreta de modelos substitutos locais, que são modelos interpretáveis usados para explicar previsões individuais de modelos de aprendizado de máquina de “caixa preta”. Modelos substitutos são treinados para aproximar as previsões do modelo de “caixa preta” subjacente. Em vez de treinar um modelo substituto global, o LIME se concentra no treinamento de modelos substitutos locais para explicar as previsões individuais.

A ideia é bastante intuitiva. Primeiro, esqueça os dados de treinamento e imagine que você tem apenas o modelo de “caixa preta” onde pode inserir pontos de dados e obter as previsões do modelo. Você pode sondar a caixa quantas vezes quiser. Seu objetivo é entender porquê o modelo de aprendizado de máquina fez uma determinada previsão. O LIME testa o que acontece com as previsões quando você fornece variações de seus dados no modelo de aprendizado de máquina. O LIME gera um novo conjunto de dados que consiste em amostras perturbadas e as previsões correspondentes do modelo de “caixa preta”. Nesse novo conjunto de dados, ele treina um modelo interpretável, que é ponderado pela proximidade das instâncias amostradas à instância de interesse. O modelo interpretável pode ser qualquer um, por exemplo, uma árvore de decisão. O modelo aprendido deve ser uma boa aproximação das previsões do modelo de aprendizado de máquina localmente, mas não precisa ser uma boa aproximação global. Esse tipo de precisão também é chamado de fidelidade local.

Matematicamente, modelos substitutos locais com restrição de interpretabilidade podem ser expressos da seguinte forma:

$$explanation(x) = argmin_{g \in G}[L(f, g, \pi_x) + \Omega(g)]. \tag{3.5}$$

Este modelo de explicação para a instância x é o modelo g (modelo de regressão linear) que minimiza a perda L (erro quadrático médio), que mede o quão perto a explicação está da previsão do modelo original f, enquanto a complexidade do modelo $\Omega(g)$ é mantida baixa (menos atributos). G é a família de explicações possíveis, por exemplo, todos os modelos de regressão linear possíveis. A medida de proximidade π_x define quão grande é a vizinhança em torno da instância x que consideramos para a explicação. Na prática, o LIME otimiza apenas a parte da perda. O

usuário tem que determinar a complexidade, por exemplo, selecionando o número máximo de atributos que o modelo de regressão linear pode usar.

Considere x' (representação interpretável) o vetor binário que é uma versão compreensível para humanos dos atributos reais usados pelo modelo original, z' a amostra perturbada que é uma fração de elementos diferentes de zero de x', $f(z)$ rótulo da classe, e $g(z')$ o modelo aprendido pelo LIME (modelo de explicação). A fim de garantir a interpretabilidade e a fidelidade local, a perda de reconhecimento da localidade é minimizada, mantendo o segundo termo baixo o suficiente para ser interpretável por humanos. Isso será referenciado como $\Omega(g)$. Ao otimizar para perda com reconhecimento de localidade, o LIME alcança fidelidade local.

Lembrando que g é o modelo a ser aprendido, z' é uma instância dos dados de treinamento e $f(z)$ é y. Para criar um conjunto de treinamento completo, realizamos amostragem aleatória uniforme de x'. Em outras palavras, criamos múltiplos z' a partir de uma única linha de x (exemplo de treinamento original). Estes são então ponderados por π_x para focar mais em z' que está mais próximo de x. Dado este conjunto de dados e rótulos, a Equação (3.5) é otimizada para aprender o modelo de explicação. Para resumir, não há dependência do tipo de modelo original para o LIME fornecer explicações (modelo agnóstico).

A explicação linear esparsa utiliza $g(z') = Cz'$ tornando o modelo de explicação linear, a perda com reconhecimento local. $\pi_x(z)$ é $\exp(-D(x,z)(2)/\sigma^2)$, a pesagem de proximidade para as amostras, e $D(x,z)$ é a função distância. A perda quadrada com reconhecimento local:

$$L(f,g,\pi_x) = \sum_{z,z' \in Z} \pi_x(z)(f(z) - g(z'))^2. \tag{3.6}$$

Resumidamente, o treinamento de modelos substitutos locais consiste:

- Selecione sua instância de interesse para a qual você deseja ter uma explicação de sua previsão de “caixa preta”;
- Perturbe seu conjunto de dados e obtenha as previsões da caixa preta para esses novos pontos;
- Pondere as novas amostras de acordo com sua proximidade com a instância de interesse;
- Treine um modelo ponderado e interpretável no conjunto de dados com as variações;
- Explique a previsão interpretando o modelo local.

O LIME trabalha basicamente com três tipos de dados: imagens, textos e dados tabulares.

3.2.1 LIME para dados tabulares

Dados tabulares são organizados em forma de tabela, em que cada linha representa uma instância e cada coluna corresponde a uma característica. No LIME, as amostras não são coletadas em torno da instância de interesse, mas sim em torno do centro de massa dos dados de treinamento, o que pode ser problemático. Por outro lado, isso aumenta a chance de que algumas previsões feitas sobre os pontos amostrados sejam diferentes da previsão para a instância analisada, permitindo ao LIME gerar pelo menos algum tipo de explicação. Definir uma vizinhança adequada em torno de um ponto é um desafio. Atualmente, o LIME utiliza um *kernel* de suavização exponencial para estabelecer essa vizinhança. Um *kernel* de suavização é uma função que recebe duas instâncias e retorna uma medida de proximidade entre elas. A largura do *kernel* define o tamanho da vizinhança: quanto menor a largura, mais próximo um ponto precisa estar para influenciar o modelo local; já uma largura maior permite que pontos mais distantes também tenham influência (MOLNAR, 2020).

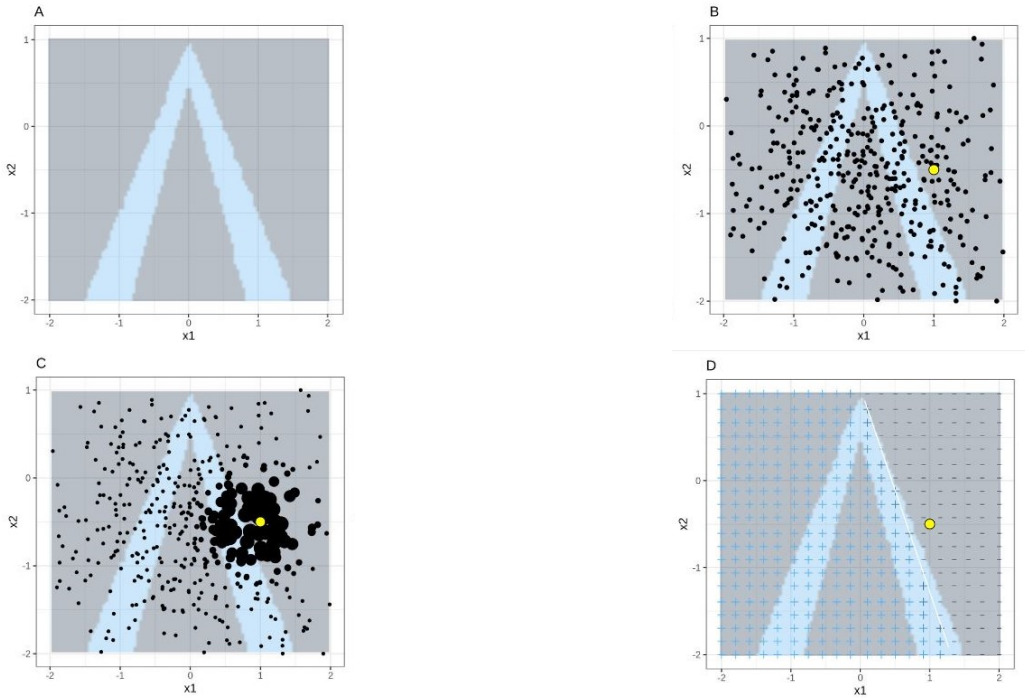

Figura 3.3 – LIME para dados tabulares. Fonte: (MOLNAR, 2020).

Figura 3.3 (A) são previsões aleatórias de floresta dadas as características $x1$ e $x2$. Classes previstas: 1 (escuro) ou 0 (claro). Figura 3.3 (B) indica instância de interesse (ponto grande) e dados amostrados de uma distribuição normal (pontos pequenos). Figura 3.3 (C) atribui maior peso aos pontos próximos à instância de interesse. Na Figura 3.3 (D), os sinais da grade mostram as classificações do modelo aprendido localmente a partir das amostras ponderadas. A linha branca marca o limite de decisão.

3.2.2 LIME para texto

O LIME aplicado a textos busca identificar quais palavras mais influenciam a decisão de um modelo. Para isso, parte-se de uma instância de interesse (um texto específico) e o método gera versões perturbadas desse texto ao remover aleatoriamente algumas palavras. Cada versão é representada em formato binário (figura 3.4), indicando a presença ou ausência de termos, e submetida ao modelo original para registrar as previsões. Em seguida, o LIME pondera os exemplos conforme sua proximidade com o texto original e treina um modelo simples e interpretável, como uma regressão linear, para aproximar o comportamento local do modelo complexo. O resultado é uma explicação que destaca as palavras mais relevantes para a previsão, permitindo entender como o modelo chegou à sua decisão.

	CONTENT	CLASS
267	PSY is a good guy	0
173	For Christmas Song visit my channel! ;)	1

Figura 3.4 – Divisão de classes LIME para texto. Fonte: (MOLNAR, 2020).

3.2.3 LIME para imagem

O LIME para imagens funciona de maneira distinta em relação ao aplicado a dados tabulares ou de texto. Alterar *pixels* individuais não seria muito útil, pois uma classe geralmente depende da contribuição de vários *pixels* ao mesmo tempo. Assim, modificações aleatórias em *pixels* isolados dificilmente alterariam as previsões do modelo. Por isso, as variações da imagem são geradas a partir da segmentação em *superpixels*, que podem ser ativados ou desativados. Esses *superpixels* correspondem a grupos de *pixels* vizinhos com cores semelhantes e podem ser "desligados" substituindo cada *pixel* por uma cor definida pelo usuário, como o cinza. Além disso, é possível estabelecer a probabilidade de um *superpixel* ser desativado em cada permutação.

Figura 3.5 – LIME para imagem. Fonte: (MOLNAR, 2020).

No exemplo da Figura 3.5 a imagem mostra pães em uma tigela, e o modelo prevê *"bagel"* com 77% e *"strawberry"* com 4%. O LIME explica essas previsões destacando regiões da imagem: verde aumenta a probabilidade do rótulo e vermelho diminui. Apesar da explicação ser coerente, a previsão de *"bagel"* está incorreta, pois os pães não têm buraco no meio.

3.3 SHAP

As descrições desta seção foram baseadas nas referências (LUNDBERG; LEE, 2017; MOLNAR, 2020).

O SHAP é um método para interpretar modelos de aprendizado (LUNDBERG; LEE, 2017). Os valores do SHAP são baseados nos valores *Shapley* da teoria dos jogos, que ajudam a determinar quanto cada jogador em um jogo colaborativo contribuiu para o pagamento total.

Para um modelo de ML, cada atributo é considerado um "jogador". O valor *Shapley* para um atributo representa a magnitude média da contribuição desse atributo em todas as combinações possíveis de atributos. Especificamente, os valores SHAP são calculados comparando as previsões de um modelo com e sem a presença de um atributo específico. Isso é feito iterativamente para cada atributo e para cada amostra no conjunto de dados. Ao atribuir a cada atributo um valor de importância para cada previsão, os valores SHAP fornecem uma explicação local e consistente de como o modelo se comporta. Eles revelam quais características têm maior impacto em uma previsão específica, seja positiva ou negativa. Isso é valioso para compreender o raciocínio por trás de modelos complexos de ML.

3.3.1 Aplicando os Valores de *Shapley*

A abordagem do SHAP consiste em explicar pequenas partes da complexidade do modelo de ML, explicando as previsões individuais, uma de cada vez. Ele explica as contribuições de cada atributo para um valor previsto individual. Os valores de *Shapley* são utilizados em jogos cooperativos para determinar a contribuição de cada jogador em um resultado coletivo. No contexto de modelos de aprendizado de máquina, consideramos cada atributo como um "jogador" que contribui para a "vitória" (ou previsão) do modelo. A definição matemática do valor de *Shapley* pode ser definido como v uma função que atribui um valor a uma coalizão de jogadores. Para um jogador i, o valor de *Shapley* é definido como:

$$\phi_i(v) = \sum_{S \subseteq Ni} \frac{|S|!(|N| - |S| - 1)!}{|N|!}(v(S \cup i) - v(S)), \tag{3.7}$$

nos quais

- N é o conjunto de todos os jogadores (atributos);
- S é uma coalizão de jogadores;
- $v(S)$ é o valor da coalizão S;
- $|S|$ e $|N|$ são os números de jogadores nas coalizões S e N, respectivamente.

Para um modelo que faz previsões $f(x)$ para uma instância x, podemos definir $v(S)$ como a previsão do modelo considerando apenas os atributos em S. A contribuição de cada atributo i para a previsão é dada por:

$$f(x) = \phi_0 + \sum_i \phi_i, \tag{3.8}$$

sendo que ϕ_0 é a previsão média de todas as instâncias.

O cálculo exato dos valores de *Shapley* pode ser computacionalmente intenso, especialmente para modelos com muitos atributos, pois envolve considerar todas as coalizões possíveis. Por isso, métodos de aproximação são frequentemente utilizados. Um deles é amostragem Monte Carlo que utiliza uma amostra de coalizões para estimar os valores de *Shapley*. Outros são baseados em algoritmos específicos que levam em conta otimização, como o algoritmo de *Shapley Tree*, que facilitam o cálculo. Os valores de *Shapley* possuem algumas propriedades importantes que os tornam adequados para interpretação, tais como:

- Eficácia: A soma das contribuições de todos os atributos é igual à previsão do modelo;
- Simetria: Se dois atributos têm a mesma contribuição para todas as coalizões, eles devem ter o mesmo valor de *Shapley*;
- *Null Player*: Se um atributo não contribui para nenhuma coalizão, seu valor de *Shapley* é zero.

3.3.2 SHAP: Local e Global

O SHAP pode apresentar explicações para predições locais ou globais.

O SHAP gera explicações locais simulando coalizões de variáveis: cada *feature* é considerada como um "jogador" que pode entrar ou não em uma coalizão, e sua contribuição é medida pelo impacto marginal na previsão quando ela é adicionada ao conjunto. Essa abordagem garante duas propriedades fundamentais: justiça, pois todas as variáveis são avaliadas em todos os contextos possíveis, e consistência, já que, se um modelo atribuir maior importância a uma variável em todas as circunstâncias, seu valor SHAP não diminuirá. Dessa forma, a explicação de uma previsão individual é representada como a soma aditiva das contribuições de cada característica em relação a uma previsão base. Isso permite interpretar como cada atributo específico empurrou o resultado para cima ou para baixo em comparação à média do modelo. Por exemplo, em um modelo de crédito, o SHAP pode mostrar que a renda elevada contribuiu positivamente para aumentar a probabilidade de aprovação, enquanto um histórico de inadimplência reduziu essa probabilidade. Assim, o método oferece uma decomposição transparente, intuitiva e fiel ao funcionamento interno do modelo.

A Figura 3.6 apresenta os *Explanation Force Plots* (gráficos de força explicativa) do SHAP. Os *Explanation Force Plots* são uma forma visual de representar como cada característica

de uma instância específica contribui para a previsão. Na primeira imagem, a cor rosa indica que essa instância tem alta probabilidade de ser classificada em uma determinada classe. Na segunda imagem a cor azul indica que essa outra instância tem baixa probalidade de ser classificada nessa classe.

Figura 3.6 – Valores SHAP para duas instâncias. Cada característica pode ser vista como uma força que empurra a previsão para cima ou para baixo. Fonte: (MOLNAR, 2020).

O SHAP gera explicações globais combinando valores de *Shapley*. Se executarmos o SHAP para cada instância, obteremos uma matriz de valores de *Shapley*. Essa matriz possui uma linha por instância de dados e uma coluna por característica. Podemos interpretar todo o modelo analisando os valores de *Shapley* nessa matriz. Isso permite observar, por exemplo, quais variáveis apresentam maior impacto no modelo em termos de magnitude e direção da contribuição. Além disso, é possível identificar relações mais complexas, como efeitos não lineares ou interações entre variáveis, quando se analisa a distribuição dos valores de SHAP em relação aos atributos. Dessa forma, as explicações globais do SHAP não derivam de um ajuste direto de um modelo explicativo simplificado, mas sim da soma consistente e fundamentada das explicações locais. Isso garante maior robustez e fidelidade.

Na Figura 3.7 é apresentado um gráfico de *Feature Importance* (gráfico de importância dos recursos) do SHAP. O *Feature Importance* é uma abordagem para quantificar a relevância de cada característica em um modelo preditivo, utilizando os valores de *Shapley*. A ideia central é simples: quanto maior o valor absoluto de *Shapley* de uma característica, maior sua contribuição para a previsão do modelo. Para obter uma importância global, calcula-se a média dos valores absolutos de *Shapley* de cada característica em todo o conjunto de dados. Em seguida, as características são ordenadas por essa média, permitindo identificar quais têm maior influência sobre o modelo. O SHAP se concentra nas atribuições individuais de cada recurso para cada previsão, refletindo a magnitude de sua contribuição. Isso torna a análise do SHAP mais informativa, pois indica não

apenas quais características afetam o modelo, mas também em que direção e intensidade cada uma contribui para a previsão.

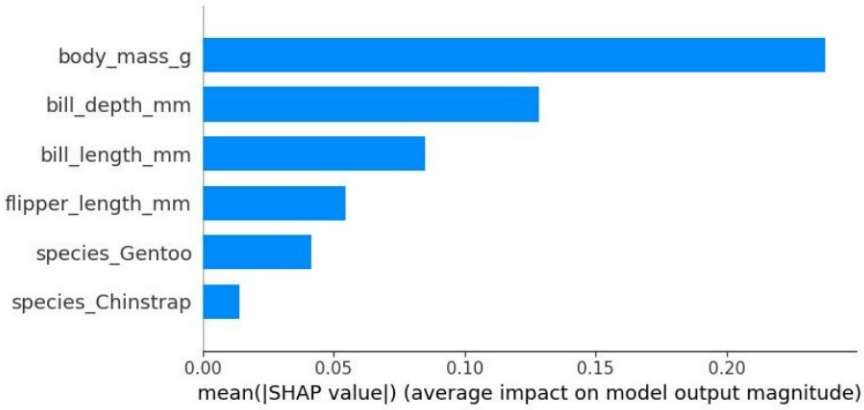

Figura 3.7 – SHAP *feature importance* medida como valores absolutos médios de *Shapley*. Fonte: (MOLNAR, 2020).

Embora o gráfico *Feature Importance* do SHAP seja útil para identificar rapidamente os recursos mais relevantes, ele não fornece informações detalhadas sobre como cada característica influencia as previsões individuais, sendo necessário complementar com gráficos como *Summary Plots* ou *Dependence Plots* para obter uma visão mais completa. O SHAP apresenta muitos recursos gráficos para explicações globais.

4 Resultados

Neste capítulo, é apresentado um estudo de interpretabilidade de Floresta Aleatória utilizando o LIME e o SHAP. Esse estudo apresenta dois modelos de Floresta Aleatória treinados com duas bases de dados distintas, e em cada um é comparada a interpretação local dos dois interpretadores LIME e SHAP para algumas instâncias das bases de dados. O primeiro modelo utiliza a base de dados Iris, que classifica a amostra como de uma das três espécies de flor a partir de suas características. O segundo modelo utiliza uma base de dados de classificação para Diabetes, onde o modelo classifica entre diabéticos e não diabéticos.

4.1 Base de dados Iris

Um exemplo simples de treinamento de modelo e de interpretação pode ser a explicação de previsões de um modelo treinado para classificar plantas do tipo íris através de algumas características de suas flores. Para isso pode ser utilizado o conjunto de dados Iris, que é bem conhecido na literatura, e é composto por quatro colunas de características com valores numéricos, e uma coluna que corresponde ao tipo da íris. O exemplo que será desenvolvido aqui fará uso deste conjunto de dados.

O conjunto de dados flor íris ou conjunto de dados Iris de Fisher é um conjunto de dados multivariados introduzido pelo estatístico e biólogo britânico Fisher (1936) em seu artigo *O uso de múltiplas medições em problemas taxonômicos*, com os dados para quantificar a variação morfológica das flores da íris de três espécies relacionadas. O conjunto de dados consiste em 50 amostras de cada uma das três espécies de íris (íris setosa, íris virginica e íris versicolor). Quatro variáveis foram medidas em cada amostra: o comprimento e a largura das sépalas e pétalas, em centímetros. Com base na combinação dessas quatro características, Fisher desenvolveu um modelo discriminante linear para distinguir as espécies umas das outras.

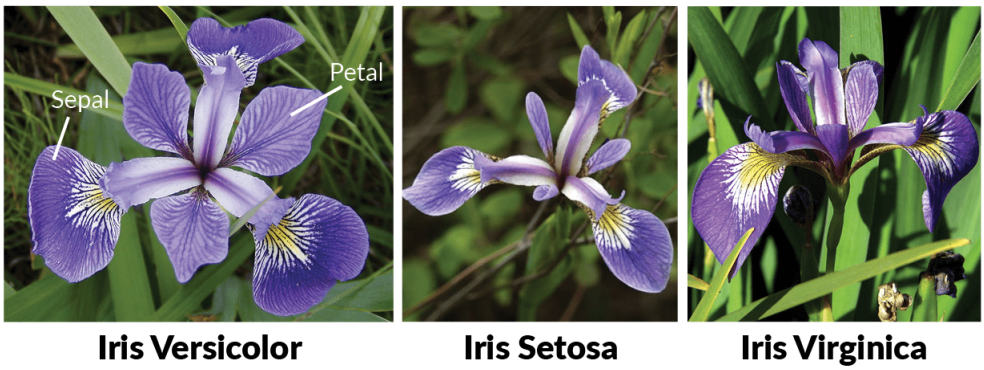

Figura 4.1 – As três espécies de íris no conjunto de dados Iris. Fonte: (FISHER, 1936).

Com esse conjunto de dados é possível treinar um modelo de Floresta Aleatória com uma parte dos dados, e utilizar outra parte dos dados para fazer os testes. Como dito anteriormente, o conjunto de dados contém 50 amostras de cada uma das três espécies, totalizando 150 amostras. Utilizando função *head* da biblioteca *Pandas* é possível ver como são representados os dados na tabela do conjunto de dados nas cinco primeiras linhas como apresenta a Figura 4.2.

	sepal.length	sepal.width	petal.length	petal.width	variety
0	5.1	3.5	1.4	0.2	Setosa
1	4.9	3.0	1.4	0.2	Setosa
2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa

Figura 4.2 – O conjunto de dados Iris contém 5 colunas e 150 linhas no total. Fonte: Autor.

Conhecendo o conjunto de dados já é possível iniciar a construção do modelo a ser treinado. Os atributos definidos como a variável X são: *sepal.length*, *sepal.width*, *petal.length*, *petal.width*. Os rótulos são as variáveis de destino que o modelo está tentando prever. Nesse caso, o rótulo é a coluna de espécies, o modelo está tentando prever a espécie da flor. O rótulo é a coluna *variety*, definida como Y. Os rótulos possíveis são apenas esses três: setosa, virginica e versicolor, sendo esse então um problema de classificação, porque tenta encontrar uma classe dentro de possibilidades limitadas existentes.

A função utilizada para construir o modelo de aprendizado de máquina foi o *RandomForestClassifier*, que é um classificador baseado em Floresta Aleatória. Também foram utilizadas funções *metrics*, que é usado para calcular a pontuação de precisão do modelo durante a previsão (quanto maior a pontuação de precisão, melhor o modelo faz previsões), e o *train_test_split* , o qual é usado para dividir o conjunto de dados em dois conjuntos. O primeiro conjunto é usado para treinar o modelo, enquanto o segundo conjunto é usado para testar o modelo.

O conjunto de dados foi dividido em duas partes, a primeira é utilizada para o treinamento, sendo *X_train* e *Y_ train* , e a segunda parte para teste, sendo *X_test* e *Y_test* . A taxa de divisão é de 0,30, isso implica que 70% do conjunto de dados será usado para treinar o modelo e 30% para testar o mesmo modelo. Para o treinamento do modelo usando Floresta Aleatória é inicializado a função *RandomForestClassifier* com 100 árvores na floresta. Depois de inicializar o modelo, ele é ajustado ao conjunto de dados. O modelo é ajustado usando o conjunto de dados de treinamento, o modelo usa esse conjunto de dados para reconhecer o padrão que aprimora a análise preditiva. Durante a fase de treinamento, o modelo ganha conhecimento por meio do aprendizado e o armazena. Ele usa esse conhecimento armazenado para fazer previsões. Durante a fase de treinamento, o modelo ajusta seus parâmetros, e em seguida, gera o modelo com os melhores parâmetros. O modelo ajustado será usado para fornecer a solução ótima.

Em seguida, é calculada a pontuação de precisão do modelo utilizando a função *metrics.accuracy_score*. A pontuação de precisão mostra o número de previsões corretas feitas pelo modelo, quanto maior a pontuação de precisão, melhor o modelo em fazer previsões. Esse cálculo é feito comparando o conjunto *Y_ test* com o resultado do modelo aplicado ao conjunto de teste *X_test*. A pontuação de precisão encontrada foi de 0,93 ou 93% de acerto. Com a função *classification_report* (relatório de classificação) é possível ver a previsão do modelo para cada classe com indica a Figura 4.3.

	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	12
Versicolor	0.88	0.88	0.88	16
Virginica	0.88	0.88	0.88	17
accuracy			0.91	45
macro avg	0.92	0.92	0.92	45
weighted avg	0.91	0.91	0.91	45

Figura 4.3 – Relatório de classificação do modelo para a base de dados Iris. Fonte: Autor.

Na Figura 4.3, é possível notar que a precisão das previsões do modelo são maiores (igual a 1 ou 100%) para a classe Setosa, e menor para as outras duas classes (igual a 0,88). Já a acurácia geral do modelo (proximidade entre o valor obtido experimentalmente e o valor verdadeiro) foi de 0,91. Com ambas as funções foram encontrados resultados de precisão bons. Com o treinamento feito e com o modelo fazendo previsões com uma precisão aceitável, agora é preciso explicar o porquê da previsão. É nesse ponto que o LIME e o SHAP são utilizados, eles conseguem demonstrar quais as variáveis, e seus respectivos pesos, tiveram importância para o modelo fazer as previsões.

4.1.1 Explicação com LIME

Com o LIME instalado basta criar o *explainer* utilizando a função do *LimeTabularExplainer* passando como parâmetros o conjunto de dados de treino. Em seguida, já é possível utilizar o *explainer* para explicar a previsão de uma determinada instância do conjunto de dados de teste. Como dito anteriormente, o LIME é um interpretador de previsões de modelos local, isso quer dizer que ele não explica as previsões do modelo como um todo, mas apenas uma determinada previsão em uma determinada linha da tabela. Porém, essa explicação é construida coletando informações do centro da massa de dados de treinamento, e não em torno da instância como o LIME faz para imagens por exemplo, porque estamos lidando com uma tabela.

A explicação que o LIME (na sua saída padrão utilizando a função *show_in_notebook*) apresenta é bem simples de ser interpretada (Figura 4.4). No canto esquerdo, ele apresenta um quadro chamado *Prediction probabilities*, com as três classes possíveis, e qual é a probabilidade da predição da amostra estar em um determinado grupo da classificação, nesse caso a probabilidade de ser setosa, versicolor ou virginica. Esse valor é apresentado em um intervalo de 0 a 1, sendo a

soma de todas as probabilidades igual a 1. No centro é apresentado um gráfico vertical com o intervalo de valores em que cada atributo (*sepal.length*, *sepal.width*, *petal.length*, *petal.width*) foi colocado pelo modelo para fazer a previsão, e seu respectivo peso para a classificação. Os valores a direita da linha são os valores que contribuiram para a classificação da amostra na classe, e à esquerda são apresentados os valores que contribuiram para a não classificação naquela classe. Essa informação do intervalo de valores que foram considerados pelo modelo para fazer a previsão é a informação mais interessante apresentada pelo LIME, porque a partir desses valores é possível ter uma ideia de quais os critérios o modelo utilizou para construir as árvores e fazer as previsões. Por fim, no lado direito é apresentada uma tabela com os valores de cada campo dos atributos, coloridos de acordo com o grupo do qual eles se classificam.

Aplicando o modelo de aprendizado para o conjunto de dados de teste na amostra oito (#8) o resultado previsto é setosa, e está correto, pois a classificação dessa amostra oito no conjunto de dados de teste é Setosa.

Aplicando o LIME para a amostra oito #8 é apresentado uma expicação de porquê o modelo fez a previsão de Setosa. Como pode ser visto na imagem abaixo (saída padrão do LIME para seus resultados) a previsão foi feita com 1 de certeza para setosa Figura 4.4).

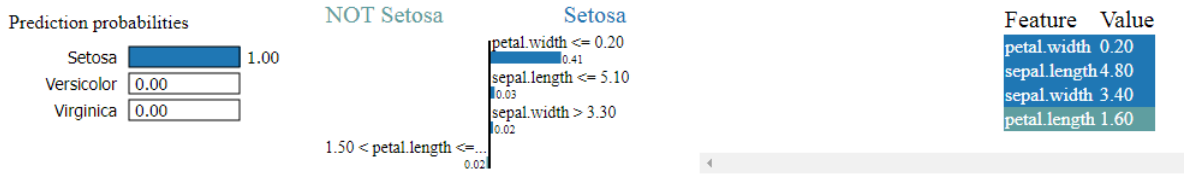

Figura 4.4 – LIME para amostra #8 do conjunto de dados Iris. Fonte: Autor.

O gráfico da Figura 4.5 apresenta o mesmo resultado do gráfico central da Figura 4.4, nele é mais fácil ver os valores dos intervalos e os pesos de cada atributo.

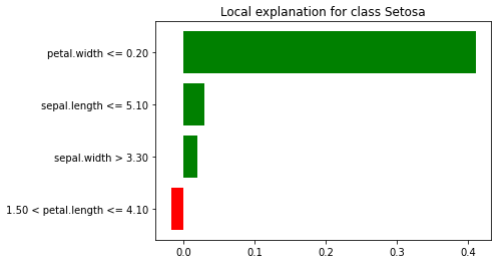

Figura 4.5 – LIME para amostra #8. Fonte: Autor.

A explicação da classificação pode ser feita através dos valores dos atributos. Nesse caso, o atributo *petal.width* com valor menor ou igual a 0,20 foi o mais relevante para a classificação, com peso 0,41. Em seguida, o atributo *sepal.length* com valor menor ou igual a 5,10 foi o segundo valor utilizado para a classificação, mas com uma relevância bem menor do que o primeiro, com peso de 0,03. O terceiro atributo utilizado foi *sepal.width*, com valor maior que 3,30. O peso de sua relevância foi ainda menor que a segunda, apenas 0,02. O último atributo na verdade pesou para a não classificação como setosa. O atributo *petal.length* com valor maior que 1,50 e menor ou igual a 4,10 teve um peso de 0,02 para a não classificação no grupo setosa, como esse valor é muito pequeno ele praticamente não teve relevância para a classificação.

Neste exemplo, em particular, o resultado foi explicado sem muita ambiguidade, isso quer dizer que o resultado previsto, setosa, foi explicado com uma porcentagem muito alta de probabilidade de predição. Mas, nem sempre isso ocorre. Na Figura 4.6, é apresentada a saída do LIME para a amostra #41 do conjunto de dados de teste.

Figura 4.6 – LIME para amostra #41 do conjunto de dados Iris. Fonte: Autor.

Neste caso, a previsão foi versicolor, porém a probabilidade de previsão foi de apenas 0,60 para versicolor, e apresentou 0,40 para virginica. Considerando os atributos, apenas o *sepal.length* pesou para a não classificação da instância como versicolor. O intervalo desse valor foi *sepal.length* maior que 6,30, e o seu valor atual é 6,80, o seu peso foi de apenas 0,03. No gráfico da Figura 4.7, é possível ver melhor os intervalos e a discrepância entre os atributos em verde, que pesaram para a classificação versicolor, e o atributo em vermelho que pesou para a classificação virginica:

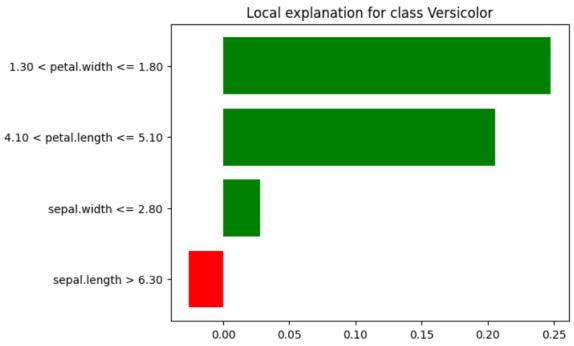

Figura 4.7 – LIME para amostra #41. Fonte: Autor.

Nesse caso, o LIME acabou mostrando que o modelo fez uma previsão bem dividida, e que os parâmetros que fizeram isso acontecer foi apenas um atributo com um valor não muito

distante do intervalo. A partir de casos como esse, é possível constatar que as previsões do modelo podem não ter uma justificativa muito forte.

4.1.2 Explicação com SHAP

Com o SHAP instalado basta criar o *explainer* utilizando a função *shap.TreeExplainer* passando como parâmetro o modelo treinado. Em seguida são gerados os valores de SHAP utilizando a função *explainer.shap_values* passando o conjunto X de teste, para ver a interpretação das previsões do conjunto de teste.

A interpretação da base de dados Iris feita pelo método SHAP segue exatamente a mesma divisão de treino/teste e o mesmo treinamento da Floresta Aleatória feita para o método LIME. A precisão e a acurácia encontrados anteriormente são os mesmos para o método SHAP, e a interpretação é aplicada às mesmas instâncias para fins de comparação. Antes de apresentar a interpretação local é interessante apresentar um recurso que o SHAP dispõe para a análise dos dados de maneira global.

O *summary plot* (ou gráfico de resumo) mostra a importância de cada atributo no modelo de maneira global. Os resultados mostram que *petal.length* e *petal.width* desempenham impactos importantes na determinação dos resultados, existindo um certo equilíbrio na importância que exercem em cada classe. A *Class* 0 é setosa, a *Class* 1 é versicolor e a *Class* 2 é virginica. No gráfico da Figura 4.8, temos no eixo Y as quatro características, e no eixo X os valores SHAP, que indicam o grau de mudança nas probabilidades logarítmicas.

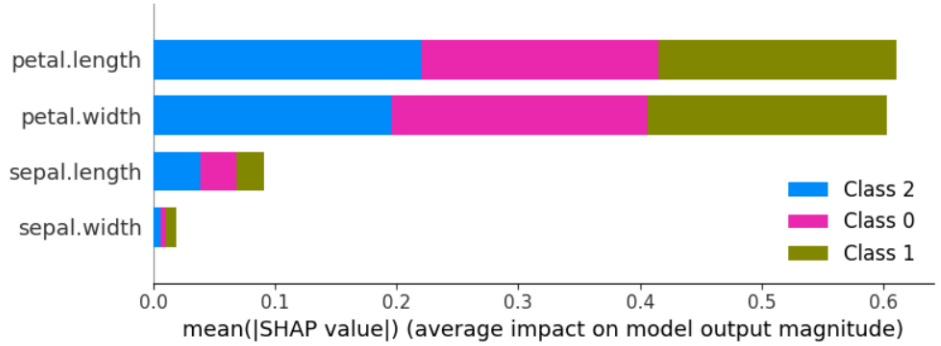

Figura 4.8 – SHAP global considerando o conjunto de dados Iris. Fonte: Autor.

Para a interpretação local das previsões, o SHAP dispõe da função *force plot* (ou gráfico de força), que apresenta qual a probabilidade de uma determinada amostra ser classificada e quais atributos contribuíram para que ela fosse classificada em uma das classes. Para tanto, é utilizado um gráfico de força e é fornecido o valor esperado, o valor SHAP e a amostra de teste. Como a base de dados tem três classes é preciso gerar três gráficos de força, um para cada classe, onde são apresentadas as contribuições dos atributos para a classificação da amostra nessa classe ou

não. As amostras a serem analisadas são a amostras $\#8$ e $\#41$ do conjunto de testes, para fins de comparação com os resultados apresentados pelo LIME.

O gráfico de força da Figura 4.9 apresenta quais atributos contribuíram para a classificação da amostra $\#8$, da base de dados de testes, na classe 0 que é setosa. A cor rosa representa valores de alta probabilidade que levam ao valor $f(x)$ igual a 1 de classificação, que representa que a amostra foi classificada como setosa com 100% de certeza. O atributo *petal.length* igual a 1,6 teve a maior contribuição para a classificação setosa, o atributo *petal.width* igual a 0,2 também teve uma importância para a classificação.

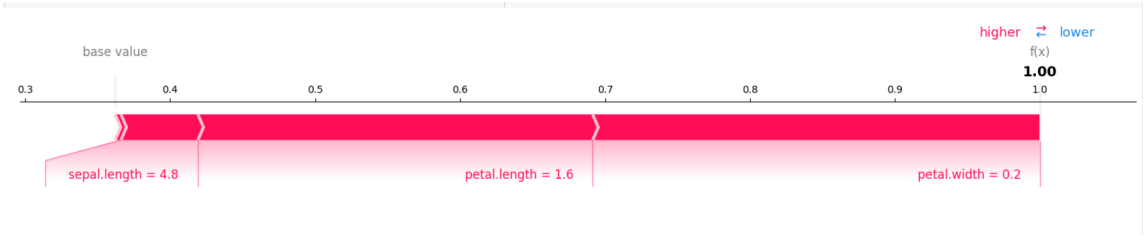

Figura 4.9 – SHAP para amostra $\#8$, classe setosa. Fonte: Autor.

Como o gráfico de força para a classe 0 teve valor 1, se olharmos para os gráficos de força da classe 1, que é versicolor, e da classe 2 que é virginica, veremos que os valores tiveram um peso negativo para a classificação, ou seja, os valores de *petal.length* igual a 1,6 e *petal.width* igual a 0,2 foram as principais características que tiveram peso negativo para a classificação em 1 ou 2, uma vez que já pesaram positivamente em 100% para a classificação na classe 0 setosa.

Os gráficos de força da Figura 4.10 apresentam respectivamente os valores que contribuíram para a classificação da mesma amostra $\#8$ para as classes 1 e 2. Todos os atributos em azul tiveram força negativa, de baixa probabilidade de classificação.

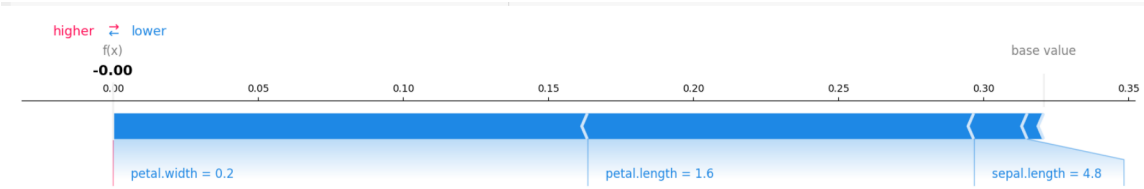

Figura 4.10 – SHAP para amostra $\#8$, classe versicolor. Fonte: Autor.

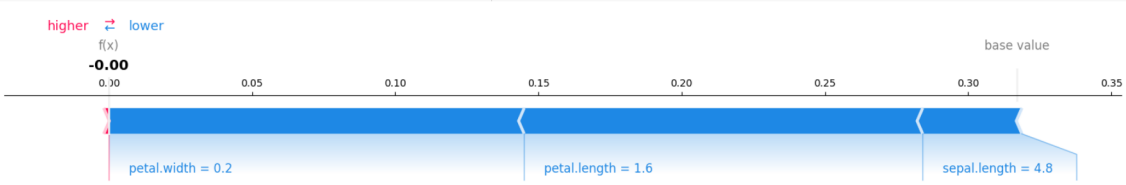

Figura 4.11 – SHAP para amostra #8, classe virginica. Fonte: Autor.

Outra maneira de olhar para esse resultado é utilizando a função *decision plot* (ou gráfico de decisão) cujo resultado pode ser visulizado na Figura 4.12, que nesse caso é o gráfico de decisão para a classe 0 ou setosa. Ele apresenta a saída igual a 1, ou seja, 100% de certeza de classificação na classe 0 setosa. Os atributos que mais contribuíram para a classificação foram *petal.length* e *petal.width*, assim como apresentado no gráfico de força.

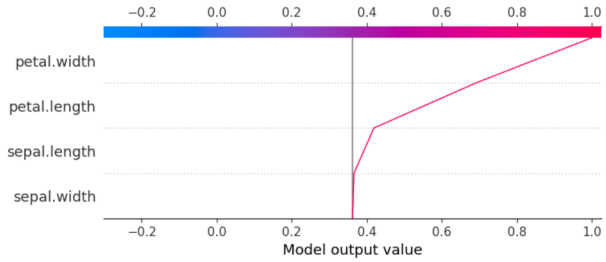

Figura 4.12 – Gráfica de decisão para amostra #8, classe setosa. Fonte: Autor.

Olhando para o outro exemplo da base de dados de teste temos a amostra #41. Na Figura 4.13, é apresentado o gráfico de força para a classificação dessa amostra na classe 1 ou versicolor, onde o atributo *petal.width* igual a 1,4 pesou de 0,30 para 0,60, levando ao $f(x)$ igual a 0,60 de probabilidade de classificação na classe 1 (versicolor). Os outros três atributos pesaram negativamente para essa classificação.

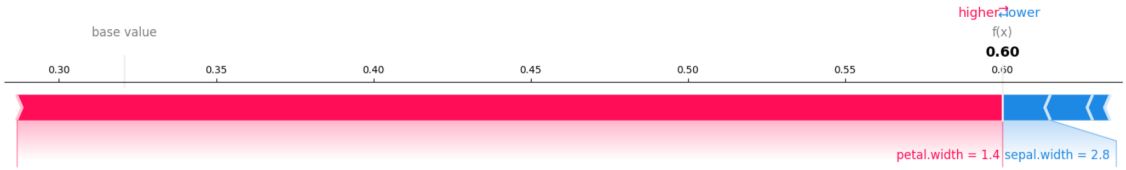

Figura 4.13 – SHAP para amostra #41, classe versicolor. Fonte: Autor.

Já o gráfico de força dessa amostra para a classificação na classe virginica apresenta as características *petal.length*, *sepal.length* e *sepal.width* contribuindo para uma $f(x)$ 0,40 de probabilidade de classificação na classe 2, e a característica *petal.width* contribuiu negativamente para essa classificação. Isso faz sentido uma vez que *petal.width* foi o atributo mais forte para a classificação na classe versicolor apresentado no gráfico anterior. Já o gráfico de força da classe

setosa dessa amostra apresentado mais abaixo tem todos os atributos com o peso negativo, e apresenta a probabilidade 0 de classificação.

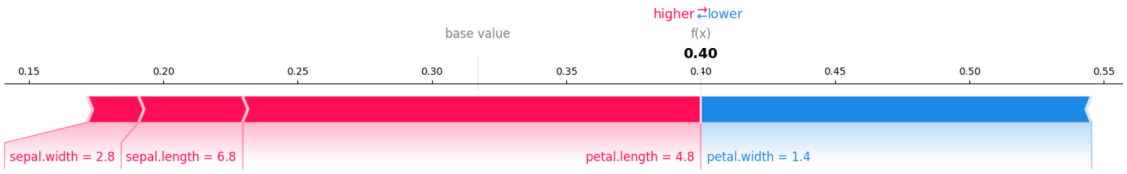

Figura 4.14 – SHAP para amostra #41, classe virginica. Fonte: Autor.

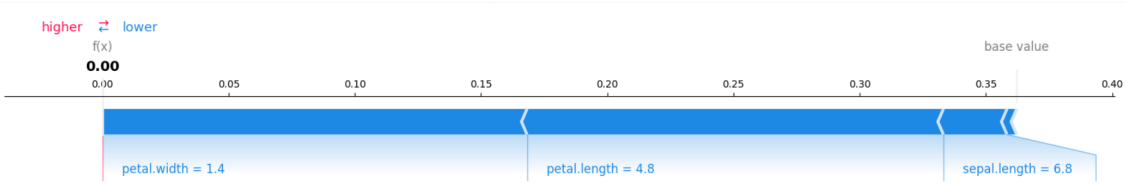

Figura 4.15 – SHAP para amostra #41, classe setosa. Fonte: Autor.

No gráfico de decisão na classe 1 (versicolor) da Figura 4.16 é possível ver a contribuição de *petal.width* para o valor 0,6 de probabilidade de classificação. O ponto de divisão está próximo de 0,3, com as características *sepal.width* e *sepal.length* pesando negativamente para a classificação.

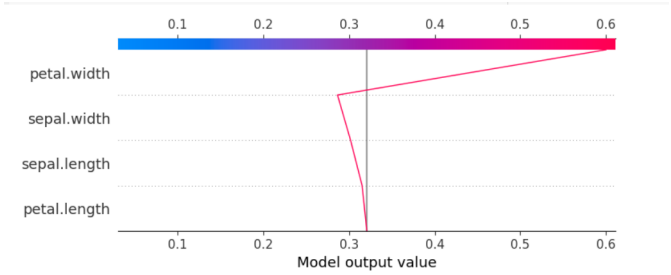

Figura 4.16 – Gráfico de decisão para amostra #41, classe versicolor. Fonte: Autor.

4.1.3 Conclusão da base de dados Iris

Os resultados de intepretabilidade local no modelo aplicado a base de dados Iris demonstrou robustez em comparação aos dois métodos aplicados LIME e SHAP. Nos exemplos apresentados é possível notar que para a amostra #8 do conjunto de testes ambos os interpretadores apresentaram a mesma saída para a explicação local, classificada como setosa com 100% de certeza. Tanto LIME quanto SHAP afirmaram que o atributo *petal.width*, no valor menor ou igual a 0,20, teve o maior impacto nessa decisão. Por outro lado, o LIME apresentou o atributo

petal.length com peso negativo para a classificação, e o SHAP apresentou o mesmo atributo com peso positivo para a classificação, o que demonstra uma divergência entre os dois métodos.

Para a amostra $\#41$ ambos, LIME e SHAP, apresentaram 60% de certeza de classificação na classe versicolor, e 40% de certeza para a classe virginica. Com o atributo *petal.width* no valor 1,40 como o atributo de maior peso para a classificação versicolor. Por outro lado, para a classificação em virginica, o LIME considerou o atributo *sepal.length* com o maior peso para essa escolha, e o SHAP considerou o *petal.length* com maior peso para essa escolha.

Pode se observar que as diferenças entre os resultados dos métodos não são muito relevantes, uma vez que ambos os modelos acertaram em relação a classe e qual o principal atributo que mais pesou para a classificação.

Outra observação pertinente é o fato de que o SHAP global apresentou os atributos *petal.width* e *petal.length* como tendo o maior peso para as decisões do modelo, o que de fato foi constatado, esses são os atributos que mais pesaram para as classificações.

Apesar do experimento ter utilizado apenas duas instâncias da base de dados para compreender os interpretadores, o ponto principal deste experimento foi compreender como criar e interpretar os resultados do LIME e do SHAP, não sendo um estudo amplo ou conclusivo acerca da base de dados Iris, mas apenas um exercício do uso do modelo de aprendizado e dos intrepretadores.

4.2 Base de dados Diabetes

Um exemplo de treinamento de modelo de aprendizado e interpretação na área médica pode ser construido sobre a base de dados Diabetes, geralmente refere-se ao *Pima Indians Diabetes Database* (SMITH et al., 1988).

Este conjunto de dados é originalmente do Instituto Nacional de Diabetes e Doenças Digestivas e Renais. O objetivo é prever, com base em medidas diagnósticas, se um paciente tem diabetes. Várias restrições foram impostas à seleção dessas instâncias em um banco de dados maior. Em particular, todos os pacientes aqui são mulheres com pelo menos 21 anos de idade, de ascendência indígena Pima. Obter um diagnóstico preciso a partir de um grande volume de dados é um problema que os profissionais da saúde enfrentam. As ferramentas baseadas em ML são um importante auxílio nesse sentido, pois conseguem processar esses dados automaticamente em questão de segundos, dando mais tempo para o médico se concentrar em outras tarefas.

Com esse conjunto de dados é possível treinar um modelo de Floresta Aleatória para prever a doença de diabetes. Esse conjunto de dados contém 768 pacientes classificados como 1 (diabéticos) ou 0 (não diabéticos). O modelo treinado pode prever, a partir de um conjunto das mesmas características, se um paciente é diabético ou não. Na Figura 4.17, são apresentados as cinco primeiras linhas do conjunto de dados.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Figura 4.17 – O conjunto de dados Diabetes contém 768 linhas e 9 colunas. Fonte: Autor.

Conhecendo o conjunto de dados apresentados na Figura 4.17, já é possível iniciar a construção do modelo a ser treinado. Os atributos utilizados são as colunas: *Pregnancies*, *Glucose*, *BloodPressure*, *SkinThickness*, *Insulin*, *BMI*, *DiabetesPedigreeFunction* e *Age*, esses oito atributos são definidos como a variável X. O rótulo é a variável de destino que o modelo está tentando prever, que está em *Outcome*, onde guarda o rótulo que será previsto 0 ou 1, definido como Y no modelo.

Dicionário dos dados:

- *Pregnancies*: Número de gestações;
- *Glucose*: Nível de glicose no sangue;
- *BloodPressure*: Medição da pressão arterial;
- *SkinThickness*: Espessura da pele;
- *Insulin*: Nível de insulina no sangue;
- *BMI*: Índice de massa corporal;
- *DiabetesPedigreeFunction*: Porcentagem de Diabetes;
- *Age*: Idade;
- *Outcome*: Resultado final (1 é Sim e 0 é Não).

Na base de dados o rótulo *Outcome* é distribuido em: 500 indivíduos classificados como 0 (não diabéticos), e 268 classificados como 1 (diabéticos). Na Figura 4.18, são representadas as proporções entre a divisão do rótulo.

O algoritmo de aprendizado utilizado para treinar o modelo foi o Floresta Aleatória para classificação, responsável por construir e treinar o modelo. Também foram utilizados as funções *metrics* e *train_test_split*. A taxa de divisão é de 0,30, isso implica que 70% do conjunto de dados será usado para treinar o modelo e 30% para testar o mesmo modelo. O conjunto de dados de treinamento ficou com 537 indivíduos. A precisão da Floresta Aleatória foi de 0,76 ou 76% de acerto. Com a função *classification_report* é possível ver a precisão do modelo para cada

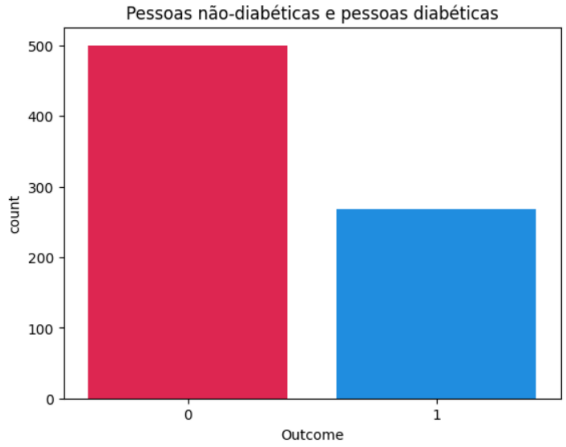

Figura 4.18 – Distribuição dos dois conjuntos de classificação, pessoas não diabéticas em rosa, e pessoas diabéticas em azul. Fonte: Autor.

classe, e ela é distinta. Para a classe 0 a precisão é de 0,84 e para a classe 1 a precisão é de 0,62. É possível que essa variação seja decorrente da diferença de amostras no conjunto, como pode ser visto na Figura 4.18, onde a classe 0 tem quase o dobro de amostras que a classe 1.

	precision	recall	f1-score	support
0	0.84	0.79	0.81	155
1	0.62	0.68	0.65	76
accuracy			0.76	231
macro avg	0.73	0.74	0.73	231
weighted avg	0.77	0.76	0.76	231

Figura 4.19 – Relatório de classificação do modelo para a base de dados Diabetes. Fonte: Autor.

A Floresta Aleatória disponibiliza uma função que informa de maneira global a importância de cada atributo para o processo de treinamento do modelo.A Figura 4.20 apresenta os atributos e a respectiva pontuação de importância de cada um deles para o modelo de aprendizado.

Glucose	0.258341
BMI	0.163634
Age	0.135946
DiabetesPedigreeFunction	0.125304
BloodPressure	0.092748
Pregnancies	0.075787
SkinThickness	0.075314
Insulin	0.072927

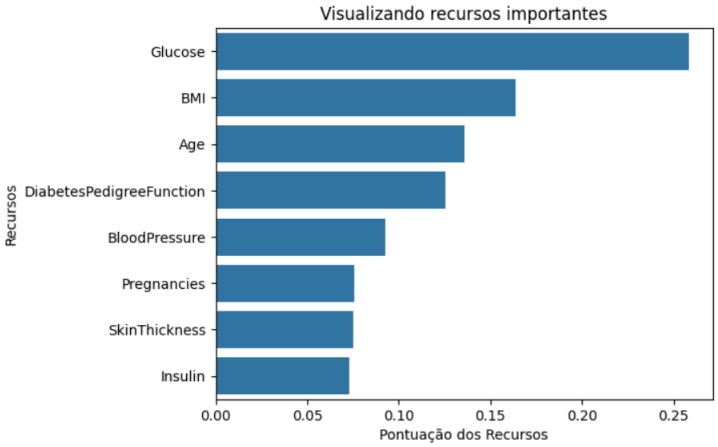

Figura 4.20 – Classificação dos atributos mais importantes no treinamento do modelo de Floresta Aleatória. Fonte: Autor.

O marcador de glicose (*Glucose*) apresenta a maior importância para o modelo, com a pontuação de 0,258341. Em seguida, o marcador de índice de massa corporal (*BMI*) apresenta pontuação 0,163634 e é o segundo atributo mais importante. É interessante considerar essa importância dos atributos gerado pela Floresta Aleatória para posteriormente comparar com os atributos mais importantes apresentados pelo LIME e SHAP.

4.2.1 Explicações LIME e SHAP

A seguir são apresentadas as interpretações do LIME e do SHAP respectivamente para cinco amostras do conjunto de dados de teste. A intenção é analisar e comparar o resultado dos dois interpretadores, para compreender melhor o funcionamento de ambos.

- **Interpretações do LIME e SHAP para a amostra** #1 (Figuras 4.21 e 4.22).

  A amostra #1 da base de dados de teste tem o valor 0, ou seja, não diabético. O LIME considerou o valor 0, não diabético, com 100% de certeza, e apresentou os atributos *Glucose* e *BMI* como mais relevantes para a classificação da amostra. O SHAP considerou o valor 0, não diabético com 100% de certeza também, e apresentou os atributos *Glucose* ,*Age* e *BMI* como os mais relevantes nessa ordem. Tanto para o LIME quanto para o SHAP a *Glucose* foi o atributo mais relevante para a classificação.

- **Interpretações do LIME e SHAP para a amostra** #2 (Figuras 4.23 e 4.24). A amostra #2 da base de dados de teste tem o valor 1, diabético. O LIME previu a classe 1 com 0,94 de certeza, e a classe 0 com 0,06 de certeza. Os atributos *Glucose* e *Age* tiveram maior relevância para a classificação. O SHAP também previu a classe 1 com 0,94 de

Figura 4.21 – Explicação LIME para amostra #1 do conjunto de dados Diabetes. Fonte: Autor.

Figura 4.22 – Explicação SHAP para amostra #1 do conjunto de dados Diabetes. Fonte: Autor.

certeza e também considerou os atributos *Glucose* e *Age* como os mais relevantes para a classificação.

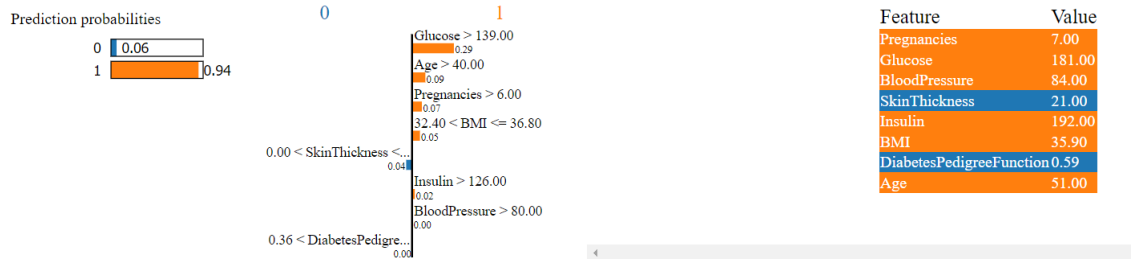

Figura 4.23 – Explicação LIME para amostra #2 do conjunto de dados Diabetes. Fonte: Autor.

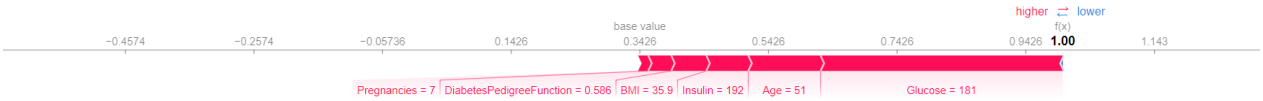

Figura 4.24 – Explicação SHAP para amostra #2 do conjunto de dados Diabetes. Fonte: Autor.

- **Interpretações do LIME e SHAP para a amostra** #3 (Figuras 4.25 e 4.26).

  A amostra #3 da base de dados de teste tem o valor 1, diabético. O LIME previu a classe 1 com 0,59 de certeza, e considerou os atributos *Glucose* e *DiabetesPedigreeFunction* como mais relevantes para a classificação. O SHAP também previu a classe 1 e considerou *Glucose* e *Pregnancies* como mais relevantes para a classificação. Ambos indicaram *BMI* como influencia negativa para a classificação na classe 1.

- **Interpretações do LIME e SHAP para a amostra** #4 (Figuras 4.27 e 4.28).

  A amostra #4 na base de dados de teste tem o valor 0, não diabético. Tanto LIME quanto SHAP previram a classificação na classe 0 com alta porcentagem. O LIME considerou a

Figura 4.25 – LIME para amostra #3 do conjunto de dados Diabetes. Fonte: Autor.

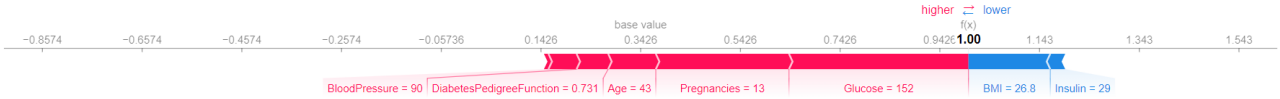

Figura 4.26 – SHAP para amostra #3 do conjunto de dados Diabete. Fonte: Autor.

Glucose e *Age* como os atributos mais relevantes para a classificação. O SHAP também considerou a *Glucose* e *Age* como os atributos mais relevantes para a classificação, porém também considerou o *BMI* como relevante para a classificação em 0. Já o LIME considerou o *BMI* com peso negativo para a classificação da amostra na classe 0.

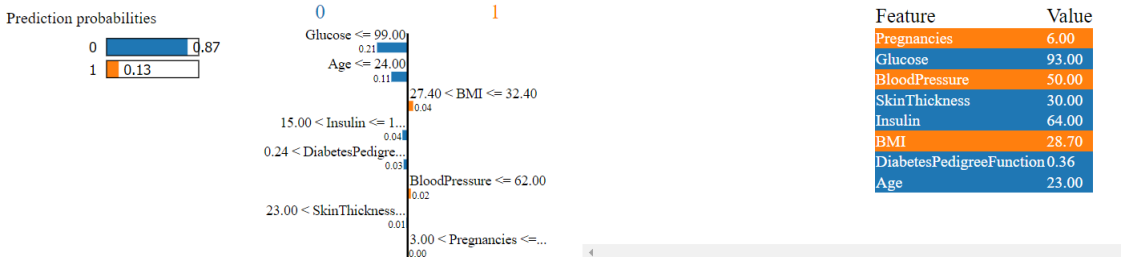

Figura 4.27 – LIME para amostra #4 do conjunto de dados Diabetes. Fonte: Autor.

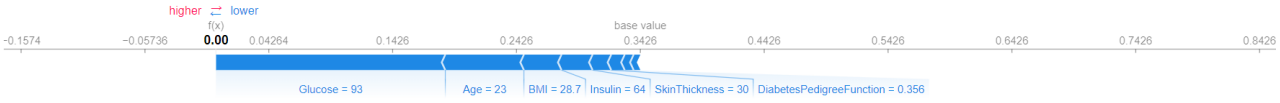

Figura 4.28 – SHAP para amostra #4 do conjunto de dados Diabetes. Fonte: Autor.

- **Interpretações do LIME e SHAP para a amostra** #5 (Figuras 4.29 e 4.30).

  A amostra #5 na base de dados de teste tem o valor 1, diabético. O LIME previu a classe 1 com 0,57 de certeza, e considerou os atributos *Age* e *BMI* como os mais relevantes para a classificação. O SHAP previu a classe 1 com 0,94 de certeza, e considerou os atributos *Age* e *Glucose* como os mais relevantes para a classificação.

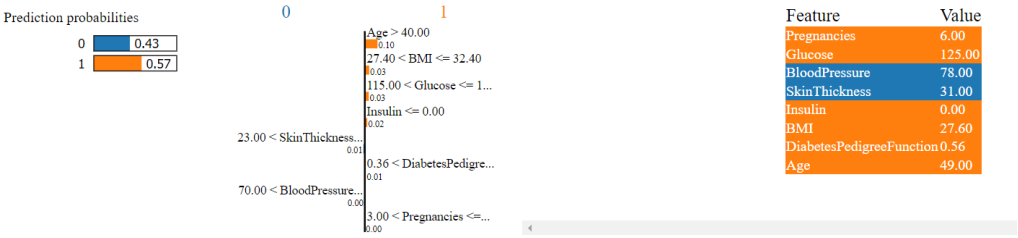

Figura 4.29 – LIME para amostra #5 do conjunto de dados Diabetes. Fonte: Autor.

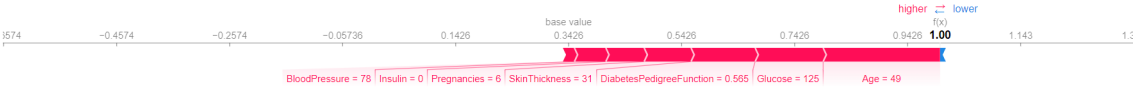

Figura 4.30 – SHAP para amostra #5 do conjunto de dados Diabetes. Fonte: Autor.

4.2.2 Conclusão da base de dados Diabetes

Os resultados de intepretabilidade local no modelo aplicado a base de dados Diabetes demonstrou robustez em comparação aos dois métodos aplicados LIME e SHAP.

Nos exemplos apresentados, é possível notar que as amostras analisadas apresentaram a mesma classificação para ambos os interpretadores com praticamente a mesma porcentagem de certeza. Não houve divergência em relação as classificações, apesar de ter ocorrido alguma diferença em relação ao grau de certeza. Um ponto muito relevante que demonstra uma grande homogeneidade na saída de ambos os interpretadores é o fato de em todas as amostras o mesmo atributo foi eleito como o mais relevante para a classificação. Na amostra #1 ambos os interpretadores elegeram a *Glucose* como o atributo mais relevante, e esse resultado se repetiu nas amostras 2,3 e 4. Na amostra #5 ambos elegeram o atributo *'Age'* como o mais relevante para a classificação.

As divergências passam a ocorrer quando se olha para os atributos da segunda importância para baixo. Na amostra #1 o LIME considerou *BMI* como o segundo atributo mais importante para a classificação, já o SHAP considerou *Age*, e colocou o *BMI* em terceiro lugar. Na amostra #4 houve uma divergência ainda maior, o LIME considerou o *BMI* como um atributo que pesou negativamente para a classificação obtida, enquanto o SHAP considerou *BMI* como um atributo que pesou positivamente para a classificação. Apesar dessas diferenças, no geral os dois interpretadores demonstraram uma grande semelhança em relação explicação da classificação das amostras.

Um ponto interessante a ser notado é o fato de que em quatro das amostras analisadas o atributo *'Glucose'* foi apresentado como o mais importante para a classificação, e apenas em uma amostra *Age* foi considerado o mais importante. Se olharmos para a Figura 4.20 que apresenta os atributos mais importantes para o treinamento da Floresta Aleatória podemos notar que *'Glucose'* é o atributo mais importante para o modelo de aprendizado também. Além disso o segundo e terceiro atributos mais importantes para o modelo foram *BMI* e *Age*, justamente os dois atributos

que mais apareceram nas explicações dos interpretadores. Apesar da pequena quantidade de amostras analisadas é possível notar uma convergência entre a explicação global apresentada pela Floresta Aleatória e as explicações locais apresentadas por LIME e SHAP.

5 Considerações Finais

Durante o desenvolvimento deste trabalho, foram explorados vários conceitos da área de aprendizado de máquina. Foi apresentado um panorama da área de aprendizado, além de uma introdução a interpretabilidade de modelos, particularmente para os métodos agnósticos. Essa apresentação deu base para uma compreensão das ferramentas principais deste trabalho, o algoritmo de Floresta Aleatória, utilizado para criar os modelos de aprendizado, e os métodos de interpretabilidade de modelos LIME e SHAP.

Com a base de dados Iris foi possível apresentar em pormenores o processo de contrução de um modelo e a aplicação dos métodos de interpretabilidade para as previsões do modelo, e com a base de dados Diabetes foi possível comparar melhor os resultados dos métodos LIME e SHAP.

De modo geral, este trabalho apresentou uma introdução ao tema de interpretabilidade de modelos com uma fundamentação teórica e com experimentos em duas áreas distintas do conhecimento, na botânica e na medicina, base de dados Iris e Diabetes, respectivamente. Com isso foi possível demonstrar a aplicabilidade dessas técnicas em diferentes áreas, além de oferecer um guia para o uso dessas tecnologias. Não houve a intenção de ser exaustivo, mas apenas adentrar na compreensão e nas suas aplicações.

Com o exemplo desenvolvido sobre a base de dados Iris foi possível compreender como os métodos LIME e SHAP apresentam interpretações para o modelo localmente, explicando a classificação do modelo com recursos gráficos acessíveis ao público leigo, mas que apresentam um certo grau de dificuldade para serem interpretados. Os resultados se demonstraram humanamente compreensíveis em relação a elucidar o modelo, principalmente quando se tem clareza acerca da base de dados.

Com o exemplo desenvolvido sobre a base de dados Diabetes foi possível concluir que os interpretadores LIME e SHAP apresentam robustez em relação as interpretações que criam para o modelo, apresentando homogeneidade em relação aos atributos que pesaram para a classificação das amostras. Também apresentaram fidelidade quando comparados a interpretação global do algoritmo Floresta Aleatória, considerando os atributos que ele apresentou como mais importantes para o treinamento como os atributos mais importantes para as interpretações locais também. Por fim, a intenção deste trabalho não foi apresentar resultados conclusivos acerca das bases de dados trabalhadas, mas apenas utilizar dessas bases de dados para apresentar a interpretação local com os métodos LIME e SHAP.

Referências

ALTMAN, N.; KRZYWINSKI, M. Ensemble methods: bagging and random forests. *Nature Methods*, Nature Publishing Group, v. 14, n. 10, p. 933–935, 2017.

ALVAREZ-MELIS, D.; JAAKKOLA, T. S. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*, 2018.

AMIT, Y.; GEMAN, D. Shape quantization and recognition with randomized trees. *Neural computation*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, v. 9, n. 7, p. 1545–1588, 1997.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, p. 5–32, 2001.

FISHER, R. A. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, Wiley Online Library, v. 7, n. 2, p. 179–188, 1936.

GARREAU, D.; LUXBURG, U. Explaining the explainer: A first theoretical analysis of LIME. In: PMLR. *International Conference on Artificial Intelligence and Statistics*. [S.l.], 2020. p. 1287–1296.

GHORBANI, A.; ABID, A.; ZOU, J. Interpretation of neural networks is fragile. In: *Proceedings of the AAAI conference on artificial intelligence*. [S.l.: s.n.], 2019. v. 33, n. 01, p. 3681–3688.

JESUS, S.; BELÉM, C.; BALAYAN, V.; BENTO, J.; SALEIRO, P.; BIZARRO, P.; GAMA, J. How can I choose an explainer? An application-grounded evaluation of post-hoc explanations. In: *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*. [S.l.: s.n.], 2021. p. 805–815.

LIPTON, Z. C. The doctor just won't accept that! *arXiv preprint arXiv:1711.08037*, 2017.

LIPTON, Z. C. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, ACM New York, NY, USA, v. 16, n. 3, p. 31–57, 2018.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, v. 30, 2017.

MAHESH, B. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, v. 9, n. 1, p. 381–386, 2020.

MARDAOUI, D.; GARREAU, D. An analysis of LIME for text data. In: PMLR. *International conference on artificial intelligence and statistics*. [S.l.], 2021. p. 3493–3501.

MELIS, D. A.; JAAKKOLA, T. Towards robust interpretability with self-explaining neural networks. *Advances in Neural Information Processing Systems*, v. 31, 2018.

MILLER, T. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, Elsevier, v. 267, p. 1–38, 2019.

MOHSENI, S.; ZAREI, N.; RAGAN, E. D. A multidisciplinary survey and framework for design and evaluation of explainable ai systems. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, ACM New York, NY, v. 11, n. 3-4, p. 1–45, 2021.

MOLNAR, C. *Interpretable machine learning*. [S.l.]: Lulu. com, 2020.

NARAYANAN, M.; CHEN, E.; HE, J.; KIM, B.; GERSHMAN, S.; DOSHI-VELEZ, F. How do humans understand explanations from machine learning systems? An evaluation of the human-interpretability of explanation. *arXiv preprint arXiv:1802.00682*, 2018.

PLUMB, G.; MOLITOR, D.; TALWALKAR, A. S. Model agnostic supervised local explanations. *Advances in Neural Information Processing Systems*, v. 31, 2018.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. "Why should I trust you?"Explaining the predictions of any classifier. p. 1135–1144, 2016.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. Anchors: High-precision model-agnostic explanations. In: *Proceedings of the AAAI Conference on Artificial intelligence*. [S.l.: s.n.], 2018. v. 32, n. 1.

SHINDE, P. P.; SHAH, S. A review of machine learning and deep learning applications. In: IEEE. *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. [S.l.], 2018. p. 1–6.

SLACK, D.; HILGARD, S.; JIA, E.; SINGH, S.; LAKKARAJU, H. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. [S.l.: s.n.], 2020. p. 180–186.

SMITH, J. W.; EVERHART, J. E.; DICKSON W. C., K. W. C.; JOHANNES, R. S. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In: *Proceedings of the Annual Symposium on Computer Application in Medical Care*. [S.l.: s.n.], 1988. p. 261–265.

SPEISER, J. L.; MILLER, M. E.; TOOZE, J.; IP, E. A comparison of random forest variable selection methods for classification prediction modeling. *Expert systems with applications*, Elsevier, v. 134, p. 93–101, 2019.

TAN, S.; CARUANA, R.; HOOKER, G.; LOU, Y. Distill-and-compare: Auditing black-box models using transparent model distillation. In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. [S.l.: s.n.], 2018. p. 303–310.

ZENG, J.; USTUN, B.; RUDIN, C. Interpretable classification models for recidivism prediction. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, Wiley Online Library, v. 180, n. 3, p. 689–722, 2017.

ZHANG, Y.; SONG, K.; SUN, Y.; TAN, S.; UDELL, M. "why should you trust my explanation?"understanding uncertainty in lime explanations. *arXiv preprint arXiv:1904.12991*, 2019.