Explicabilidade em Modelos de IA Aplicados à Seleção de Recursos Humanos

Reginaldo Gregório de Souza Neto - 61482

Dissertação apresentada à **Escola Superior de Tecnologia e Gestão** do **Instituto Politécnico de Bragança** para obtenção do Grau de **Mestre em Informática** no âmbito da **Dupla Diplomação** com a **Universidade Tecnológica Federal do Paraná - *Campus* Campo Mourão**.

Trabalho realizado sob a orientação de

Prof. João Paulo Ramos Teixeira

Prof. Juliano Henrique Foleis

Bragança

Junho de 2025

Explicabilidade em Modelos de IA Aplicados à Seleção de Recursos Humanos

Reginaldo Gregório de Souza Neto - 61482

Dissertação apresentada à **Escola Superior de Tecnologia e Gestão** do **Instituto Politécnico de Bragança** para obtenção do Grau de **Mestre em Informática** no âmbito da **Dupla Diplomação** com a **Universidade Tecnológica Federal do Paraná - *Campus* Campo Mourão**.

Trabalho realizado sob a orientação de

Prof. João Paulo Ramos Teixeira

Prof. Juliano Henrique Foleis

Bragança

Junho de 2025

Dedicatória

Dedico este trabalho aos meus pais, tios, primos e todos os familiares, próximos e distantes, que de alguma forma contribuíram para minha trajetória, oferecendo apoio, incentivo e ajuda durante minha formação acadêmica e pessoal. Em especial aos meus avós que, independentemente da situação ou adversidade, sempre foram sinônimo de acolhimento e carinho.

Dedico também àqueles que se mostraram solícitos quando mais precisei e que fizeram parte desta jornada, se tornando pessoas muito importantes para mim, compartilhando momentos memoráveis que levarei por toda a vida.

Agradecimentos

Sou grato à UTFPR e ao IPB pela oportunidade desta dupla diplomação. Agradeço aos orientadores João Paulo Ramos Teixeira e Juliano Henrique Foleis pelo apoio, paciência e ensinamentos durante o desenvolvimento deste trabalho. Sou também grato ao professor Rui Pedro Lopes, que, por meio do programa "Blended Intensive Program - Machine Learning for Data Science", me possibilitou uma viagem à Alemanha para uma semana de estudos intensivos em Inteligência Artificial, experiência que agregou muito na minha formação acadêmica.

À minha mãe, Lidiane Sousa Basso, ao meu pai, Rogério Gregório de Souza, às minhas avós, Francisca Ferreira e Ana Maria Cavalcante, e ao meu avô, Reginaldo Gregório de Souza, agradeço por todo o cuidado, ensinamentos e valores que me prepararam para a vida. Amo cada um de vocês.

Por fim, agradeço à Gabriela Paola Sereniski, por viver ao meu lado nesta jornada em Portugal, compartilhando momentos incríveis e muitas horas de estudo, inspirando meu desenvolvimento pessoal e académico. Agradeço também à sua mãe, Marciana Tomazini, que me acolheu com carinho e me fez sentir parte da família.

A todos vocês, minha sincera gratidão!

Resumo

Este trabalho tem como objetivo propor um sistema de apoio à decisão para o processo de recrutamento e seleção de candidatos ao cargo de consultor de vendas, por meio da predição automatizada de notas atribuídas a currículos. Foram empregados algoritmos de aprendizado de máquina supervisionado, incluindo KNN, Random Forest, SVM e redes neurais MLP, para prever a avaliação de um recrutador humano a partir de 14 atributos técnicos extraídos de currículos reais. Como diferencial, este estudo incorporou técnicas de explicabilidade como SHAP, LIME e TreeInterpreter, promovendo a explicabilidade na análise preditiva, possibilitando identificar a importância de cada variável tanto local quanto globalmente. A base de dados foi padronizada, normalizada e complementada com dados sintéticos a fim de mitigar desequilíbrios. A avaliação dos modelos foi conduzida com base em métricas como MAE, MSE, RMSE e R^2, além da análise de resíduos e da matriz de confusão. O melhor desempenho entre as redes neurais foi obtido com a MLP (23-12-6-1), treinada com 50% de dados sintéticos, que alcançou MAE de 0,23, MSE de 0,28, RMSE de 0,53 e $R^2 = 0{,}97$. No entanto, o modelo com maior desempenho geral foi a Random Forest com 1000 árvores, que atingiu MAE de 0,14, MSE de 0,16, RMSE de 0,40 e $R^2 = 0{,}98$.

Palavras-chave: Inteligência Artificial; Recrutamento e Seleção; Explicabilidade; Aprendizado Supervisionado; Regressão.

Abstract

This study proposes a decision support system for the recruitment and selection process of sales consultant candidates through automated score prediction based on resumes. Supervised machine learning algorithms, including KNN, Random Forest, SVM, and MLP neural networks, were employed to predict the evaluation of a human recruiter based on 14 technical attributes extracted from real resumes. As a distinctive feature, this study incorporated explainability techniques such as SHAP, LIME, and TreeInterpreter, promoting explainability in predictive analysis and enabling the identification of the importance of each variable both locally and globally. The dataset was standardized, normalized, and supplemented with synthetic data to mitigate imbalances. Model evaluation was conducted based on metrics such as MAE, MSE, RMSE, and R^2, in addition to residual analysis and the confusion matrix. Among the neural networks, the best performance was achieved by the MLP (23-12-6-1), trained with 50% synthetic data, which reached an MAE of 0,23, MSE of 0,28, RMSE of 0,53, and an $R^2 = 0{,}97$. However, the overall best-performing model was the Random Forest with 1000 trees, which achieved an MAE of 0,14, MSE of 0,16, RMSE of 0,40, and an $R^2 = 0{,}98$.

Keywords: Artificial Intelligence; Recruitment and Selection; Explainability; Supervised Learning; Regression.

Conteúdo

Lista de Tabelas

Lista de Figuras

Siglas

AM Aprendizado de Máquina. 1, 3, 4, 28, 30

AP *Average Precision*. 23

AUC área sob a curva ROC. 23

CV *Curriculum Vitae*. 3, 25, 27, 30, 31, 34, 68

DL *Deep Learning*. 35, 36

IA Inteligência Artificial. 1, 3, 4, 7

KNN *K-Nearest Neighbors*. x, xi, 4, 10–12, 20, 21, 30, 36, 41, 42, 47–49, 69

LIME *Local Interpretable Model-agnostic Explanations*. x, 5, 16, 17, 45, 59, 62–64, 69, 70

LLMs Modelos de Linguagem de Grande Escala. 21, 71

MAE Erro Absoluto Médio. 2, 19, 20, 34, 47, 51, 55, 57, 68

MLP *Multi-Layer Perceptron*. x, xii, 4, 7, 10, 19, 30, 36, 38, 41, 43–45, 47, 56–60, 62, 68–70

MSE Erro Quadrático Médio. 2, 19, 20, 34, 35, 47, 55, 57, 67, 68

PLN Processamento de Linguagem Natural. 20, 71

RAG *Retrieval-Augmented Generation*. 21, 22

RBF *Radial Basis Function*. 13, 14, 42, 53–55, 68

ReLU *Rectified Linear Unit*. 43

REN Reconhecimento de Entidades Nomeadas. 20

ReS Recrutamento e Seleção. 1, 3, 4, 25, 68

RF *Random Forest*. xi, 4, 14–16, 18, 36, 41, 43, 47, 50–53, 64, 67–69

RH Recursos Humanos. 1, 64, 69, 72

RMSE Raiz do Erro Quadrático Médio. 2, 34, 47, 51, 55, 57, 67, 68

RNA Rede Neural Artificial. xi, 7, 9, 10, 19

SHAP *SHapley Additive exPlanations*. xii, 5, 16–18, 45, 59, 60, 69, 70

SVM *Support Vector Machine*. xi, 4, 12, 13, 20, 21, 30, 36, 41, 42, 47, 54, 55, 69

TF-IDF T*erm Frequency-Inverse Document Frequency*. 21

Capítulo 1

Introdução

Os processos de Recrutamento e Seleção (ReS) passaram por transformações significativas nos últimos anos, impulsionados pelo aumento no volume de candidaturas e pela crescente complexidade na correspondência entre perfis profissionais e requisitos das vagas. Os métodos tradicionais, fortemente dependentes da atuação humana, enfrentam limitações de escalabilidade, com a falta de consistência e risco de vieses decisórios (Will et al., 2023). Nesse contexto, soluções tecnológicas, especialmente aquelas baseadas em Inteligência Artificial (IA), têm-se consolidado como alternativas promissoras para a modernização e aprimoramento dessas práticas (Li et al., 2020) (Rajesh et al., 2018).

Este trabalho tem como objetivo desenvolver e avaliar modelos de predição automatizada de notas atribuídas a currículos de candidatos ao cargo de consultor de vendas, utilizando algoritmos de Aprendizado de Máquina (AM) supervisionado. A proposta consiste na construção de um sistema de apoio à decisão no contexto da triagem curricular, capaz de auxiliar profissionais da área de Recursos Humanos (RH) na tarefa de selecionar candidatos com base em critérios técnicos previamente definidos. Espera-se, com isso, reduzir o tempo gasto na etapa inicial do recrutamento e promover maior padronização e objetividade na avaliação de perfis profissionais.

A pesquisa parte da hipótese de que, a partir da estruturação de dados curriculares em atributos técnicos, é possível treinar modelos capazes de replicar, com alta precisão, as

decisões de um avaliador humano. Para isso, foram testados diferentes algoritmos supervisionados, com enfoques distintos: modelos baseados em instâncias (KNN), em árvores de decisão (RF), em máquinas de vetores de suporte (SVM) e em redes neurais (MLP). Todos os modelos foram avaliados no contexto de regressão, prevendo notas inteiras entre 0 e 10 atribuídas aos candidatos.

A base de dados utilizada foi composta por currículos reais estruturados manualmente, contendo atributos como idade, escolaridade, tempo de experiência, formações complementares, idiomas e histórico de promoção. Após análise exploratória, os dados passaram por um processo de padronização, codificação e normalização. Para mitigar problemas de desequilíbrio entre classes, foram geradas amostras sintéticas com ruído controlado, o que permitiu o aumento da amostragem de instâncias durante o treinamento.

A avaliação dos modelos foi realizada por meio de métricas estatísticas amplamente utilizadas em regressão, como o Erro Absoluto Médio (MAE), Erro Quadrático Médio (MSE), Raiz do Erro Quadrático Médio (RMSE) e Coeficiente de Determinação (R^2). Além disso, os resíduos das predições foram analisados em gráficos de dispersão e histogramas, permitindo uma visão mais detalhada sobre os comportamentos sistemáticos de erro.

Um dos diferenciais deste trabalho é a aplicação de técnicas contemporâneas de explicabilidade, como SHAP, LIME e TreeInterpreter. Essas ferramentas permitiram analisar de forma interpretável como cada atributo contribuiu individualmente para cada predição, proporcionando maior transparência ao modelo e agregando valor às decisões geradas.

Os resultados demonstram que a utilização de modelos supervisionados bem treinados e interpretáveis pode ser uma ferramenta eficaz para apoiar o processo de recrutamento e seleção, especialmente nas etapas iniciais de filtragem curricular. O sistema proposto mostrou-se promissor tanto em termos de desempenho estatístico quanto de potencial aplicação prática.

1.1 Recrutamento e Seleção

A crescente complexidade dos processos de ReS tem conduzido à adoção de técnicas de IA com o objetivo de automatizar e aprimorar a tomada de decisão. Essas tecnologias possibilitam o processamento de grandes volumes de dados de candidatos, além da identificação de padrões que, muitas vezes, não são facilmente perceptíveis por meio da análise humana tradicional (M. Jatobá et al., 2023).

Apesar dos avanços, a incorporação da IA em processos de ReS apresenta desafios relacionados à transparência e equidade. Os sistemas automatizados devem não apenas alcançar elevados níveis de acurácia, mas também proporcionar mecanismos de explicabilidade, permitindo que profissionais de recursos humanos e os próprios candidatos compreendam os fundamentos das decisões algorítmicas sugeridas pelos sistemas de suporte à decisão. Muitos modelos de AM, especialmente aqueles com arquiteturas complexas, operam como "caixas-pretas", o que compromete a confiança e a aceitação dessas soluções em contextos decisórios sensíveis (Blumen & Cepellos, 2023).

Diante desses fatores, observa-se a necessidade de soluções que combinem capacidade preditiva com mecanismos de interpretação. Este trabalho propõe a aplicação de modelos supervisionados ao processo de triagem de currículos, incorporando técnicas de explicabilidade para tornar as decisões algorítmicas acessíveis à análise humana.

1.2 Algoritmos de Aprendizado Supervisionado

O aprendizado supervisionado é uma abordagem de AM na qual um modelo é treinado utilizando um conjunto de dados rotulado, ou seja, cada entrada é associada a uma saída desejada. O objetivo é que o modelo aprenda uma função que mapeie entradas para saídas, permitindo a previsão de resultados para novos dados não vistos. Essa metodologia é amplamente utilizada em tarefas de classificação e regressão, exatamente o contexto em que a classificação de *Curriculum Vitae* (CV) se enquadra.

O objetivo é fazer com que a máquina reconheça o padrão de avaliação e seja capaz

de replicá-lo igualitariamente entre todos os candidatos, buscando a agilidade no processo de triagem dos candidatos para uma possível entrevista de acordo com a qualificação dos mesmos para a vaga pretendida.

O aprendizado supervisionado baseia-se na minimização do risco empírico, onde o modelo busca minimizar a diferença entre as previsões e os valores reais nos dados de treinamento (IBM Cloud Education, 2024). Essa abordagem é fundamentada na teoria estatística do aprendizado, que fornece garantias sobre a capacidade de generalização do modelo para novos dados.

1.3 Objetivos

A presente dissertação baseia-se na proposta original do projeto (ver Apêndice A), cuja finalidade é a de explorar a aplicação da IA na automatização da classificação de currículos em processos de ReS. O foco deste trabalho é demonstrar a aplicação adequada de técnicas de AM no contexto de recrutamento, enfatizando a importância da explicabilidade para tornar compreensíveis os critérios utilizados pelos modelos na avaliação de candidatos.

1.3.1 Objetivo Geral

Desenvolver e avaliar modelos de aprendizado de máquina para a classificação automatizada de currículos em processos de recrutamento, com ênfase na explicabilidade dos modelos.

1.3.2 Objetivos Específicos

- Pré-processar e estruturar um conjunto de dados de currículos, de forma a adequá-lo ao treinamento de modelos de aprendizado de máquina.

- Comparar o desempenho de diferentes algoritmos de aprendizado supervisionado, sendo estes: MLP, KNN, SVM e *Random Forest* (RF).

- Aplicar e avaliar técnicas de explicabilidade como o SHAP, LIME e TreeInterpreter para os modelos que apresentarem melhor resultado na classificação.

Capítulo 2

Revisão da Literatura

Este capítulo apresenta os fundamentos teóricos dos modelos utilizados neste trabalho. Inicialmente, são descritos conceitos básicos relacionados ao reconhecimento de padrões, incluindo arquiteturas e técnicas de classificação associadas. Em seguida, é realizada uma revisão da literatura científica sobre estudos prévios que empregaram métodos de IA para classificação e avaliação automatizada de currículos na seleção de candidatos para vagas de emprego.

2.1 MultiLayer Perceptron (MLP)

O MLP é uma arquitetura de RNA do tipo *feedforward*, composta por múltiplas camadas densamente conectadas (Moriarty & Miikkulainen, 1998), se tratando de modelos computacionais inspirados na estrutura e no funcionamento do sistema nervoso biológico. Essas redes são compostas por unidades de processamento denominadas neurónios artificiais, organizadas em camadas interligadas. Cada neurónio aplica uma transformação não linear sobre uma combinação ponderada das entradas recebidas, o que permite à rede aprender representações complexas dos dados (Rumelhart, Hinton et al., 1986).

O neurónio artificial pode ser descrito por três componentes fundamentais:

- **Pesos Sinápticos**: os pesos associados às entradas controlam a influência de cada

variável sobre a saída do neurónio. Durante o treinamento da rede, esses valores são ajustados com o objetivo de minimizar o erro de predição.

- **Função de Soma (ou agregação)**: cada entrada é multiplicada pelo seu respetivo peso, e os produtos são somados. O resultado dessa operação é denominado potencial de ativação, sendo a base para o cálculo da saída do neurónio.

- **Função de Ativação**: após a agregação ponderada, o valor resultante é transformado por meio de uma função de ativação, que introduz não linearidade ao modelo. Essa etapa faz com que a rede consiga representar relações complexas entre os dados. A saída gerada pode então ser repassada para as camadas seguintes.

Matematicamente, a saída y do neurónio pode ser representada como:

$$y = f(u) \tag{2.1}$$

$$u = \sum_{i=1}^{n} w_i x_i + b \tag{2.2}$$

Em que w_i são os pesos associados às entradas x_i, b é o viés adicionado para permitir o ajuste da função de ativação no espaço de entrada, e f representa a função de ativação. Os parâmetros w_i e b são otimizados durante o treinamento por meio de algoritmos baseados no método de retro-propagação do erro (*backpropagation*) (McClelland et al., 1986; Rumelhart, McClelland et al., 1986), tais como o *Gradient Descent* ou o *Stochastic Gradient Descent* (Velazco & Lyra, 2002; Wu et al., 2021).

A Figura 2.1 ilustra a operação de um único neurónio artificial (também chamado de nó). Nela, cada entrada é multiplicada por um peso sináptico, os resultados são somados juntamente a um viés, e o valor agregado passa por uma função de ativação que define a saída do neurónio. Essa estrutura básica serve como unidade de processamento em todas as camadas da rede.

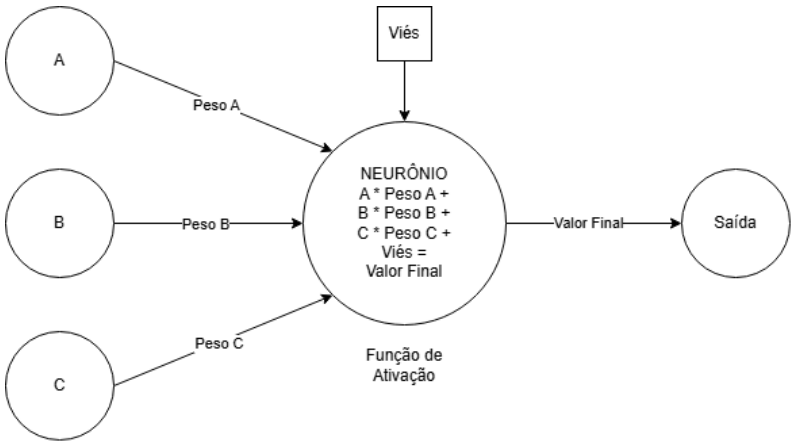

Figura 2.1: Representação de um neurónio artificial com entradas ponderadas, viés e função de ativação. Fonte: Autoria Própria.

A combinação de diversos neurónios interconectados origina a topologia das redes neuronais. A Figura 2.2 exemplifica uma RNA com múltiplas camadas ocultas, mostrando como os neurónios se organizam em camadas e como os sinais são propagados da entrada até a saída final.

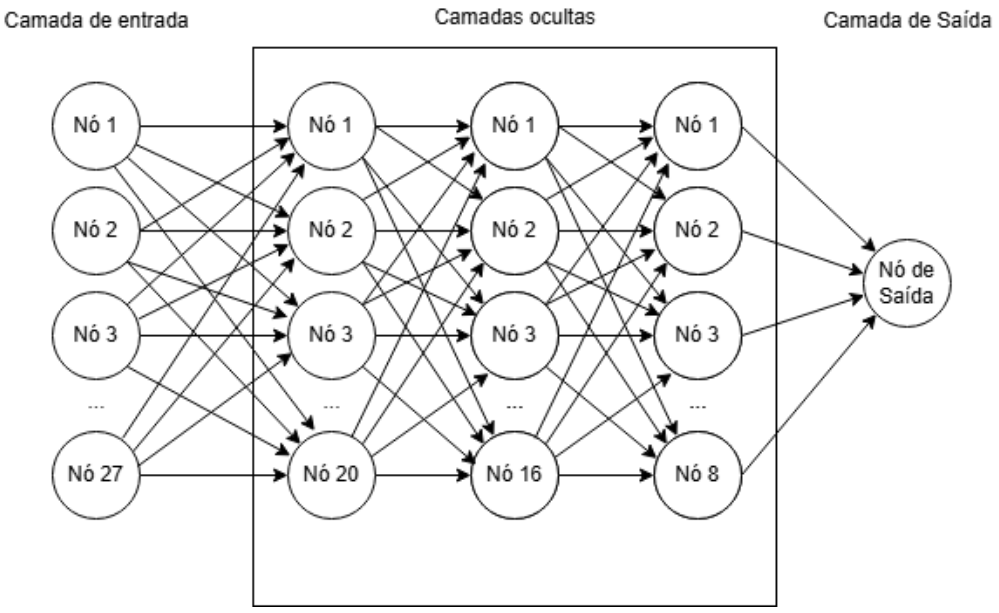

Figura 2.2: Exemplo de uma RNA com 3 camadas ocultas. Fonte: Autoria Própria.

As RNAs são compostas por múltiplos neurónios organizados em camadas. As principais camadas que compõem uma rede são:

- **Camada de Entrada**: responsável por receber os dados brutos. Cada neurónio dessa camada corresponde a uma característica do vetor de entrada, garantindo a estruturação adequada dos dados no início do processamento.
- **Camadas Ocultas**: localizadas entre as camadas de entrada e saída, essas camadas realizam o processamento intermediário da informação. Cada neurónio aplica uma transformação não linear às entradas recebidas da camada anterior. Em arquiteturas mais profundas, podem existir múltiplas camadas ocultas, o que amplia a capacidade de representação da rede.
- **Camada de Saída**: recebe as ativações da última camada oculta e gera a predição final. O número de neurónios e o tipo de função de ativação nesta camada dependem da natureza do problema.

A estrutura de uma rede neural pode variar conforme a aplicação, sendo possível ajustar parâmetros como o número de camadas, a quantidade de neurónios por camada, as funções de ativação e os critérios de treinamento. Essa flexibilidade permite a adaptação da arquitetura da rede a diferentes tipos de dados e objetivos, maximizando seu desempenho em tarefas específicas. No caso da figura 2.2 a arquitetura consiste em uma MLP que se trata de uma RNA com 3 camadas ocultas, podendo ser representada textualmente por MLP(27-20-16-8-1) onde cada número representa a quantidade de nós em cada camada da rede.

2.2 K-Nearest Neighbors (KNN)

O algoritmo KNN é uma técnica de aprendizado supervisionado do tipo não paramétrico, baseada em instâncias, cuja classificação é realizada por meio da votação maioritária dos k vizinhos mais próximos no espaço amostral. A similaridade entre as amostras é

geralmente determinada por uma métrica de distância, como a distância euclidiana. O KNN é amplamente utilizado em tarefas de classificação devido sua simplicidade e eficácia em conjuntos de dados com estrutura clara (Altman, 1992). No entanto, sua sensibilidade a ruídos e dimensionalidade requer cuidados específicos em etapas de pré-processamento.

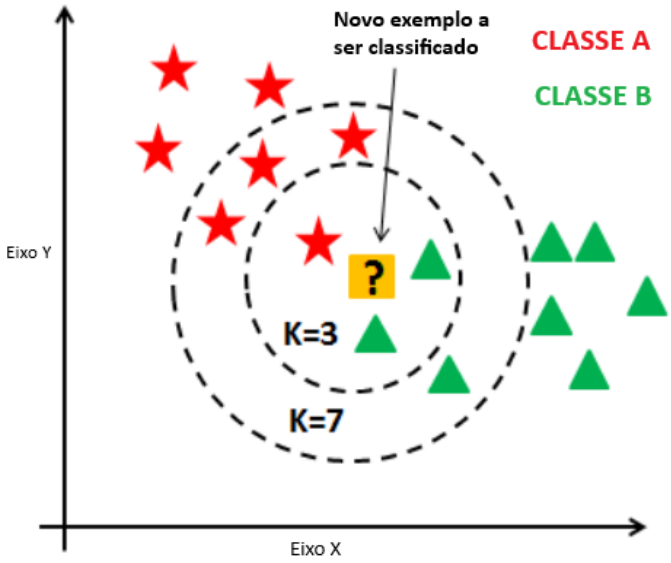

Figura 2.3: KNN influência do valor de k sobre a classificação. Fonte: Gokte (2020).

Um exemplo de ambiguidade introduzida por ruído pode ser observado na Figura 2.3, onde a predição do novo exemplo depende diretamente do valor escolhido para k. Quando $k = 3$, a maioria dos vizinhos mais próximos pertence à Classe B, levando à sua provável classificação nesta categoria. No entanto, com $k = 7$, a decisão muda para Classe A, demonstrando a sensibilidade do algoritmo ao valor de k e à distribuição local dos dados.

No contexto do presente trabalho, ao contrário de uma representação bidimensional como a da Figura 2.3, o espaço de entrada é composto por 23 atributos derivados de características extraídas dos currículos dos candidatos. Esse aumento de dimensionalidade torna o problema mais complexo, pois, em espaços de alta dimensão, as distâncias entre os pontos tendem a se tornar menos discriminativas — um fenómeno conhecido como *problema da dimensionalidade* (Beyer et al., 1999). Nesses casos, os pontos vizinhos mais

próximos podem não ser semanticamente relevantes, o que compromete a precisão do modelo se o pré-processamento e a seleção de atributos não forem realizados corretamente.

Além disso, a presença de atributos ruidosos ou não informativos pode distorcer o cálculo de distância e induzir erros na classificação. Por esse motivo, foram adotadas estratégias de padronização e geração de dados sintéticos descritas na secção 3.1, com o objetivo de mitigar os efeitos negativos da alta dimensionalidade e garantir um ambiente mais estável para a aplicação do KNN.

2.3 Support Vector Machine (SVM)

O algoritmo SVM é uma técnica de aprendizado supervisionado baseada na definição de um hiperplano ótimo que separa as classes de forma maximamente distante de seus pontos mais próximos, denominados vetores de suporte. Trata-se de um método robusto tanto para problemas de classificação linear quanto para situações em que os dados não são linearmente separáveis, sendo particularmente eficaz em contextos de alta dimensionalidade (Cortes & Vapnik, 1995).

Na essência, o objetivo da SVM é encontrar uma fronteira de decisão que maximize a margem entre as classes, reduzindo o risco de erro de generalização. Esse conceito pode ser visualizado na Figura 2.4, onde três classes são separadas por hiperplanos que delimitam as regiões de decisão. A capacidade do algoritmo de identificar a melhor fronteira torna-o menos propenso ao sobreajuste, especialmente quando comparado a classificadores baseados em instância como o KNN.

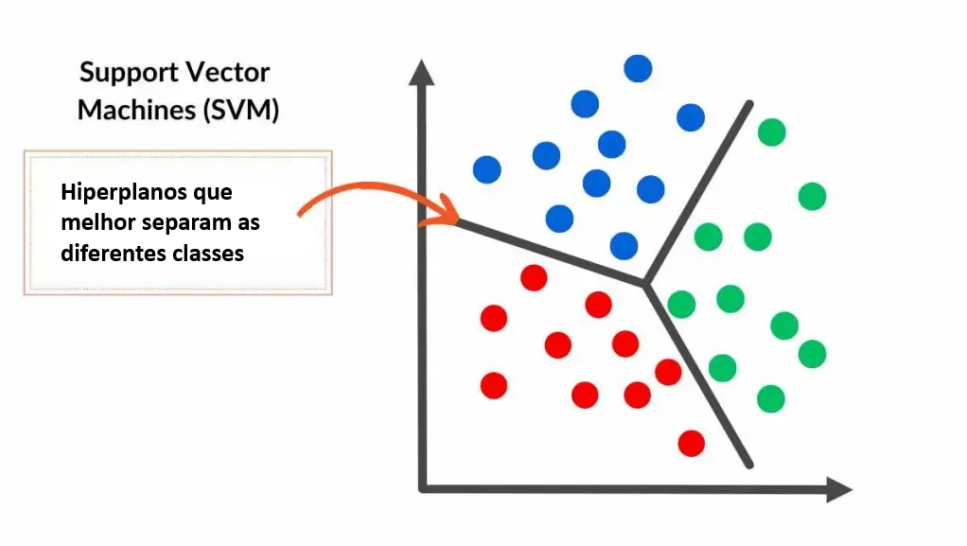

Figura 2.4: Separação entre classes por meio de hiperplanos — SVM. Fonte: Otten (2024).

Entre as principais vantagens do uso da SVM estão a sua capacidade de lidar com conjuntos de dados com um número elevado de atributos, a boa performance em bases com margens claras de separação entre as classes e a flexibilidade proporcionada pelas diferentes funções *kernel* disponíveis (Roy et al., 2024).

Entretanto, em muitos casos reais, os dados não são linearmente separáveis no espaço original de atributos. Para lidar com essa limitação, a SVM utiliza a técnica de *kernel trick*, que consiste em aplicar uma função de transformação não linear capaz de projetar os dados em um espaço de características de maior dimensão. Nesse novo espaço, torna-se possível encontrar uma separação linear que não seria viável no espaço original (Hofmann et al., 2008).

A Figura 2.5 ilustra esse processo, no qual os dados originalmente embaralhados em duas dimensões são transformados por meio de uma função *kernel* (como o *Radial Basis Function* (RBF) ou o polinomial) em uma superfície tridimensional, onde uma separação linear, e portanto, uma decisão mais precisa, se torna viável.

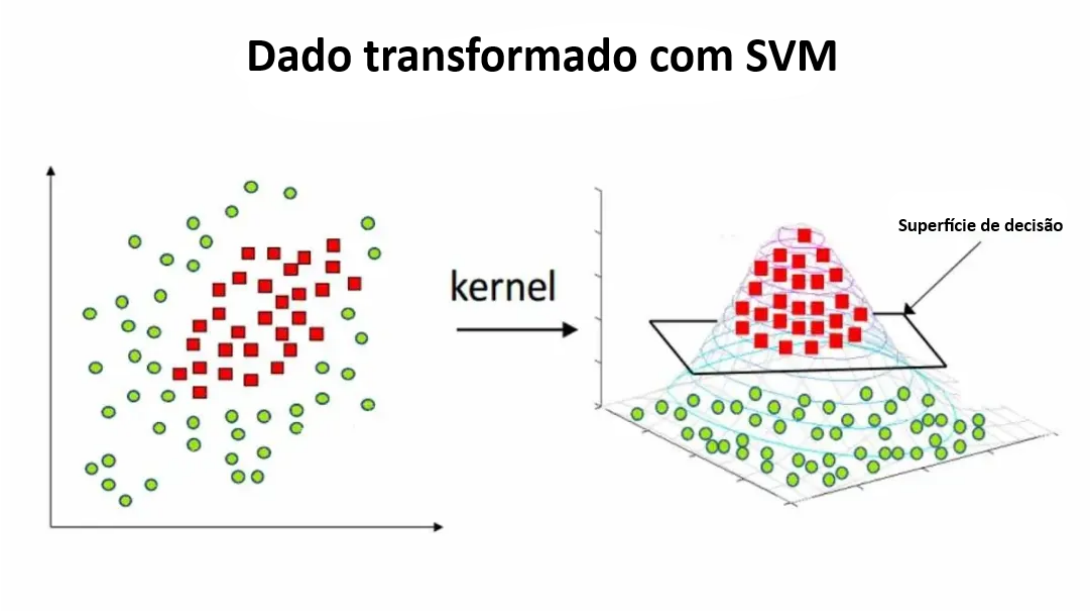

Figura 2.5: Transformação de dados via função *kernel*. Fonte: Otten (2024).

Por outro lado, sua aplicação pode exigir alto custo computacional, especialmente em bases muito grandes, e requer atenção na escolha do *kernel* e de seus hiperparâmetros. Entre os mais relevantes estão o parâmetro de regularização C, que controla o equilíbrio entre a complexidade do modelo e a penalização por erros de classificação no treinamento, e o coeficiente γ (*gamma*), que define o alcance da influência de cada ponto de dado no modelo, sendo particularmente importante na configuração do *kernel* RBF. Valores elevados de C e γ podem tornar o modelo excessivamente ajustado aos dados (*overfitting*), enquanto valores muito baixos podem resultar em subajuste (*underfitting*), comprometendo a capacidade de generalização do classificador (scikit-learn, 2010).

2.4 Random Forest (RF)

O algoritmo RF, ou Floresta Aleatória, é um modelo de aprendizado supervisionado amplamente utilizado em tarefas de classificação e regressão. Ele pertence à família dos métodos de *ensemble learning*, os quais combinam múltiplos modelos básicos com o objetivo de formar um modelo mais robusto e preciso. O princípio fundamental por trás

do *ensemble* é que uma coleção de modelos fracos, como árvores de decisão individuais, quando combinados de maneira apropriada, podem produzir resultados mais estáveis, precisos e generalizáveis do que um modelo isolado (Breiman, 2001).

A construção de um RF envolve a geração de diversas árvores de decisão, cada uma treinada a partir de uma amostra aleatória (com reposição) do conjunto de dados de entrada, em um processo conhecido como *bootstrap aggregating*, ou *bagging*. Ainda de acordo com Breiman (2001), para cada divisão dos nós durante o crescimento das árvores, um subconjunto aleatório de atributos é selecionado, reduzindo a correlação entre os modelos individuais. Essa aleatoriedade introduzida em dois níveis (dados e atributos) aumenta a diversidade entre as árvores e contribui diretamente para a capacidade do modelo de evitar o sobreajuste.

Em tarefas de regressão, a predição final do modelo RF é obtida pela média das saídas produzidas por todas as árvores do conjunto. Já em classificações, aplica-se a votação maioritária. Esse mecanismo de decisão coletiva permite uma redução significativa da variância sem um aumento substancial no viés, um dos principais diferenciais do RF em comparação com modelos simples de árvore (Biau & Scornet, 2016).

Um aspeto que confere ao RF uma vantagem adicional é sua capacidade de estimar automaticamente a importância dos atributos. Durante o treinamento, o algoritmo avalia o impacto de cada variável na redução da impureza em todos os pontos onde ela é utilizada para uma divisão. O resultado é um *ranking* que indica quais atributos mais contribuíram para as decisões do modelo. Para Biau e Scornet (2016) essa análise é particularmente útil em contextos que exigem certo grau de interpretabilidade, permitindo entender, por exemplo, quais características de um currículo mais influenciam sua avaliação final.

Outro ponto importante a ser considerado é a configuração dos hiperparâmetros do modelo, entre os quais se destacam: o número de árvores na floresta (`n_estimators`), a profundidade máxima das árvores (`max_depth`), o número mínimo de amostras por divisão e folha (`min_samples_split`, `min_samples_leaf`) e a quantidade de atributos considerados em cada divisão (`max_features`). A escolha adequada desses parâmetros pode impactar de forma significativa o desempenho do modelo, sendo comum a adoção

de estratégias como validação cruzada para sua otimização.

Apesar de o RF ser um modelo composto de múltiplos estimadores, o que tende a reduzir sua transparência em comparação a uma única árvore de decisão, existem métodos que possibilitam a extração de explicações locais a respeito de cada predição. Um exemplo é o uso da biblioteca `TreeInterpreter` (Saabas, 2017), que permite decompor a predição de uma instância em termos da contribuição individual de cada atributo, além de indicar o valor base da previsão (média populacional da resposta). Essa ferramenta será utilizada na análise de resultados do presente trabalho (ver Secção 5.2.3), possibilitando uma compreensão mais detalhada do comportamento do modelo e promovendo maior confiança nas decisões automatizadas.

2.5 Técnicas de Explicabilidade

Esta secção apresenta os conceitos teóricos das principais técnicas utilizadas para interpretar o funcionamento de modelos de aprendizado de máquina. São abordados métodos que permitem identificar quais variáveis contribuíram para cada predição e qual foi a influência de cada uma. As técnicas descritas incluem o `TreeInterpreter`, o LIME e o SHAP.

2.5.1 TreeInterpreter

O `TreeInterpreter` é uma técnica de explicabilidade desenvolvida especificamente para modelos de árvore de decisão e seus derivados, como o RF. Sua abordagem baseia-se na decomposição da predição final em contribuições individuais de cada variável de entrada, somadas a um valor base (valor médio da predição no conjunto de dados). Essa decomposição é possível devido à estrutura hierárquica das árvores, que permite rastrear o caminho percorrido por uma instância ao longo dos nós de decisão até a folha correspondente (Saabas, 2017).

Durante a travessia da árvore, cada divisão contribui com um valor incremental para a predição, dependendo do valor da variável analisada. O `TreeInterpreter` coleta essas

contribuições e as agrega por variável, fornecendo uma explicação local da predição de forma intuitiva e transparente. Essa técnica é particularmente útil para identificar quais atributos influenciaram positiva ou negativamente o resultado, e em que magnitude, sendo eficiente mesmo em conjuntos de modelos como florestas aleatórias.

2.5.2 LIME

O LIME (*Local Interpretable Model-agnostic Explanations*) é uma técnica de explicabilidade baseada na ideia de que modelos complexos podem ser localmente aproximados por modelos lineares simples. Para isso, o LIME gera uma vizinhança artificial em torno da instância de interesse, realizando pequenas perturbações nas suas variáveis de entrada, e em seguida observa como essas alterações impactam a saída do modelo (Ribeiro et al., 2016).

Com as instâncias perturbadas e suas respetivas predições, o LIME ajusta um modelo linear ponderado, onde os pesos refletem a proximidade de cada instância perturbada à instância original. O coeficiente de cada variável nesse modelo linear local indica sua importância relativa na decisão do modelo original naquela região específica do espaço de atributos. Esse processo permite obter explicações interpretáveis para modelos arbitrários, mesmo aqueles considerados "caixas-pretas", como redes neuronais profundas e modelos de *boosting*.

2.5.3 SHAP

O SHAP (*SHapley Additive exPlanations*) é um método de explicabilidade que se baseia na teoria dos jogos cooperativos, especificamente nos valores de Shapley. Cada valor SHAP representa a contribuição marginal de um atributo para a predição, considerando todas as combinações possíveis com as demais variáveis. Isso garante que as explicações respeitem propriedades desejáveis como a aditividade, a simetria e a consistência (S. M. Lundberg & Lee, 2017).

O cálculo dos valores SHAP envolve a avaliação do impacto de cada variável na predição do modelo, considerando todas as possíveis coalizões ou coligações de variáveis. Para uma instância específica, o valor SHAP de uma variável é a média das contribuições marginais dessa variável em todas as possíveis ordens de entrada no modelo. Matematicamente, para um modelo f e uma instância x, o valor SHAP ϕ_j para a variável j é dado por:

$$\phi_j = \sum_{S \subseteq N \setminus \{j\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} \left[f_{S \cup \{j\}}(x_{S \cup \{j\}}) - f_S(x_S) \right] \tag{2.3}$$

onde N é o conjunto de todas as variáveis, S é um subconjunto de N que não contém j, f_S é o modelo treinado com as variáveis em S, e x_S representa os valores de entrada correspondentes a S (Christoph Molnar, 2023).

Devido à complexidade computacional do cálculo exato dos valores SHAP, que cresce exponencialmente com o número de variáveis, métodos de aproximação são utilizados na prática. Para modelos baseados em árvores de decisão, como RF e `XGBoost`, algoritmos eficientes foram desenvolvidos para calcular os valores SHAP de forma exata em tempo polinomial.

Uma das principais vantagens do SHAP é sua capacidade de fornecer explicações locais e globais. As explicações locais detalham a contribuição de cada variável para a predição de uma instância específica, enquanto as explicações globais são obtidas ao agregar os valores SHAP de todas as instâncias, permitindo identificar as variáveis mais influentes no modelo como um todo (Christoph Molnar, 2023).

Além disso, o SHAPé capaz de lidar com interações entre variáveis e fornecer informação sobre como diferentes valores de uma variável afetam a predição. Por exemplo, o gráfico de dependência do SHAPmostra a relação entre os valores de uma variável e seus respetivos valores SHAP, permitindo identificar padrões não lineares e interações com outras variáveis.

2.6 Estudos de Caso em Triagem Automatizada de Currículos

A seguir, são apresentados quatro estudos que aplicam algoritmos de aprendizado supervisionado na triagem automatizada de currículos, detalhando suas metodologias, tratamento de dados e resultados obtidos.

2.6.1 Estudo 1: Inteligência artificial no recrutamento & seleção: inovação e seus impactos para a gestão de recursos humanos

O trabalho desenvolvido por M. N. Jatobá (2020) teve como objetivo aplicar RNAs do tipo MLP para a triagem automatizada de currículos de candidatos ao cargo de técnico de vendas. A pesquisa resultou na construção de uma base de dados com 800 currículos, a partir da leitura manual e anotação por um especialista da área de Recursos Humanos, que atribuiu a cada currículo uma nota de 0 a 10 com base em critérios técnicos.

Foram extraídos 14 atributos dos currículos, entre eles: sexo, idade, nível educacional, posse de veículo próprio, tempo e tipo de experiência em vendas, quantidade de cursos de formação, promoção no trabalho e conhecimento nos idiomas português, inglês e espanhol. Após codificação, esses atributos compuseram uma camada de entrada com 28 nós. A saída da RNA correspondia à nota atribuída pelo avaliador, permitindo o uso do modelo como uma regressão contínua.

A autora avaliou múltiplas configurações de arquitetura, número de neurónios ocultos (1, 2, 4, 6, 10 e 20), funções de ativação (*TanSig*, *LogSig*, *PureLin* e *Elliot*), e três algoritmos de treino distintos: *Resilient Backpropagation*, *Levenberg-Marquardt* e *Bayesian Regularization*. Cada combinação foi testada em 40 sessões de treino, sendo os resultados avaliados pelas métricas MAE e MSE, utilizando divisão da base em 70% para treino, 15% para validação e 15% para teste.

O melhor desempenho foi obtido com a arquitetura 28-2-1 (28 na entrada, 2 neurónios na camada oculta e 1 na saída), treinada com o algoritmo *Bayesian Regularization*, utilizando *TanSig* como função de ativação na camada oculta e *PureLin* na saída. Esse modelo atingiu um MAE de 0,292 e MSE de 0,375 no conjunto de teste. Além disso, segundo a autora, 75% das predições foram exatamente iguais à nota esperada, Dentro do subconjunto de currículos com notas entre 7 e 10 (total de 40 casos), 29 foram acertados com exatidão, e os demais ficaram com erro de no máximo uma unidade.

Contudo, foi possível constatar que nem todas as classes de notas estão representadas no conjunto de teste. Classes como 3 e 9, por exemplo, possuem frequência zero, o que indica uma limitação da aleatoriedade na separação da base. Isso pode ter implicações na avaliação geral do modelo, sobretudo em métricas como acurácia ou MAE, pois não há garantia de que o desempenho observado seja consistente em toda a escala de valores possíveis.

2.6.2 Estudo 2: Aplicação de KNN e SVM na Triagem Automatizada de Currículos

O estudo conduzido por Tayal et al. (2024) propôs um sistema automatizado para triagem de currículos, integrando técnicas de Processamento de Linguagem Natural (PLN) e algoritmos de aprendizado supervisionado, especificamente KNN e SVM. O objetivo principal foi aprimorar a eficiência e a precisão na seleção de candidatos, mitigando as limitações dos métodos tradicionais, frequentemente manuais e subjetivos.

O conjunto de dados utilizado consistiu em aproximadamente 5.000 currículos, coletados de diversas fontes, incluindo plataformas como o Kaggle (Goldbloom, 2010). Esses currículos foram submetidos a um processo de pré-processamento que incluiu a remoção de ruídos, normalização textual e *tokenização*. Técnicas de PLN, como o Reconhecimento de Entidades Nomeadas (REN) e a análise de partes do discurso (*POS tagging*), foram empregadas para extrair informações relevantes, tais como habilidades, experiências profissionais e formações académicas.

Após o pré-processamento, os dados textuais foram transformados em representações numéricas utilizando a técnica T*erm Frequency-Inverse Document Frequency* (TF-IDF) que é uma medida estatística que tem o intuito de indicar a importância de uma palavra de um documento em relação a uma coleção de documentos. Essa vetorização permitiu a aplicação dos algoritmos KNN e SVM para a classificação dos currículos em duas categorias: "selecionado" ou "rejeitado". O modelo KNN foi configurado com diferentes valores de k, enquanto o SVM utilizou o *kernel* linear, com ajustes nos parâmetros de regularização para otimizar o desempenho.

Os experimentos demonstraram que o modelo SVM alcançou uma acurácia de 94,76%, enquanto o KNN obteve 98,44%. Esses resultados indicam que ambos os modelos são eficazes na tarefa de triagem de currículos, com o KNN apresentando uma ligeira vantagem em termos de acurácia. No entanto, é importante considerar outros fatores, como o tempo de processamento e a escalabilidade, ao escolher o modelo mais adequado para aplicações práticas.

2.6.3 Estudo 3: Framework Multiagente com LLMs para Triagem de Currículos

O estudo conduzido por Lo et al. (2025) propôs uma abordagem inovadora para a triagem automatizada de currículos, alicerçada em um *framework* multiagente que integra Modelos de Linguagem de Grande Escala (LLMs) com módulos especializados para tarefas distintas dentro do *pipeline* de recrutamento. Essa arquitetura distribuída foi projetada para simular o julgamento humano com maior profundidade e contextualização, utilizando o paradigma de *Retrieval-Augmented Generation* (RAG) para enriquecer a tomada de decisão com informações externas relevantes.

O experimento foi conduzido sobre um conjunto de dados composto por currículos anonimizados oriundos de múltiplas plataformas de recrutamento online. Esses documentos passaram por um estágio preliminar de saneamento textual, no qual foram aplicadas técnicas de normalização linguística, remoção de ambiguidade e transformação semântica

utilizando *embeddings* contextuais.

O *framework* é composto por quatro agentes autónomos com funções específicas:

- **Extrator de Currículos:** Converte dados não estruturados em representações organizadas, identificando entidades como competências, instituições de ensino e cargos anteriores;
- **Avaliador:** Utiliza RAG para realizar inferência contextualizada com base em fontes externas, como por exemplo, *rankings* universitários ou perfis corporativos;
- **Resumidor:** Gera sumários executivos com base nas análises realizadas pelos agentes anteriores;
- **Formatador de Pontuação:** Organiza os resultados em um formato padronizado para integração com sistemas de recrutamento.

O sistema multiagente foi avaliado com base em métricas de acurácia, precisão contextual e tempo de resposta. Em relação a abordagens tradicionais baseadas apenas em classificadores supervisionados, o modelo proposto apresentou melhoria de até 18% na precisão da triagem. A inserção de contexto externo via RAG mostrou-se particularmente eficaz na análise de currículos com formações não convencionais ou experiências atípicas.

2.6.4 Estudo 4: ResumeNet – Avaliação Automática da Qualidade de Currículos

O trabalho de Luo et al. (2018) apresentou o *ResumeNet*, um *framework* baseado em redes neuronais profundas desenvolvido para realizar avaliação automática da qualidade de currículos. O modelo proposto visa inferir uma pontuação de qualidade para cada currículo a partir da análise conjunta de múltiplos fatores, como consistência semântica entre secções, número de experiências, nível educacional e diversidade de habilidades. O objetivo central foi reduzir a subjetividade associada à avaliação manual em processos seletivos.

Para viabilizar os experimentos, os autores construíram uma base de dados composta por 10.343 currículos, obtidos junto a uma empresa privada de gestão de currículos. Desses, apenas 122 instâncias foram rotuladas manualmente por especialistas da área de recursos humanos (33 positivas e 89 negativas), sendo os demais exemplos não rotulados. Para contornar a escassez de rótulos, o estudo empregou abordagens como *semi-supervised learning* com regularização por variedade de dados (*manifold regularization*) e funções de perda contrastiva, como *contrastive loss* e *triplet loss*, que possibilitam a exploração de relações entre pares e trios de instâncias rotuladas e não rotuladas.

A arquitetura do *ResumeNet* combina técnicas modernas de representação semântica, como o *Universal Sentence Encoder*, mecanismos de atenção para ponderar as secções do currículo e camadas totalmente conectadas para inferência da pontuação final. Além disso, foi introduzido um sub-módulo específico para mensurar a consistência entre as secções de habilidades e experiências profissionais, empregando similaridade de cosseno sobre *embeddings*.

Nos experimentos, os modelos foram avaliados com base em métricas como área sob a curva ROC (AUC), F1-score e *Average Precision* (AP). A versão com melhor desempenho utilizou a função de perda *triplet loss* e apresentou F1-score médio de 0,541, superando significativamente abordagens anteriores como o serviço *RezScore*, que obteve apenas 0,341. Os resultados evidenciam o potencial do *ResumeNet* em tarefas de triagem automática de currículos, especialmente em cenários com dados rotulados escassos.

Capítulo 3

Metodologia

Neste capítulo serão apresentados os materiais e métodos utilizados para o desenvolvimento do trabalho. Também são descritos os recursos computacionais utilizados e as estratégias empregadas, com ênfase na aplicação das técnicas de diminuição de viés no desenvolvimento dos modelos.

3.1 Base de Dados

A base de dados utilizada é a mesma que foi publicada no trabalho de Neto et al. (2025) e se encontra em formato *csv* com a dimensão 600 × 15. Onde cada linha da tabela representa um CV correspondente a um candidato à vaga de consultor de vendas. As primeiras 14 colunas contêm características extraídas dos candidatos, enquanto a 15ª coluna apresenta a nota final atribuída, variando de 0 a 10.

Na compreensão destes valores classificatórios, entende-se que: o CV com nota entre 0 e 6 não está adequado para continuar na fase de seleção, nomeadamente, insuficiente. Já o CV com notas 7, 8, 9 ou 10 são os mais adequados para continuar nas etapas de seleção, nomeadamente, suficiente, bom, muito bom e excelente, respetivamente.

A extração das características de cada CV, bem como a atribuição da respetiva nota, foram realizadas manualmente por um especialista na área de ReS. Os atributos das 600 instâncias foram organizados conforme descrito na Tabela 3.1.

Atributo	Qualificação	Tipo de Variável	Codificação
Idade	Sem informação 18 a 23 anos 24 a 29 anos 30 a 35 anos 36 a 41 anos 42 ou mais	Ordinal	0 1 2 3 4 5
Sexo	Masculino Feminino	Nominal	1 2
Nível Educacional	Menor que 2º grau 2º grau completo Graduação Pós-graduação/MBA Mestrado	Ordinal	1 2 3 4 5
Viatura própria	Não Possui / Possui	Nominal	1 / 2
Experiência no setor de vendas	Não informado Sem experiência Até 6 meses Entre 7 mêses e 1 ano De 1,5 a 2 anos 2,5 a 3 anos 3,5 a 5 anos 5,5 a 9 anos 10 ou mais anos	Ordinal	0 1 2 3 4 5 6 7 8
Experiência Tipo de função	Não se aplica Técnico Consultor Gestor	Nominal	0 1 2 3
Experiência em outros setores	Não informado Sem experiência Até 6 meses Entre 7 mêses e 1 ano De 1,5 a 2 anos 2,5 a 3 anos 3,5 a 5 anos 5,5 a 9 anos 10 ou mais anos	Ordinal	0 1 2 3 4 5 6 7 8
Cursos em vendas	Nenhuma formação De 1 a 3 cursos De 4 a 6 cursos 7 ou mais cursos	Ordinal	1 2 3 4
Cursos afins à área de vendas	Nenhuma formação De 1 a 3 cursos De 4 a 6 cursos 7 ou mais cursos	Ordinal	1 2 3 4
Promoção no trabalho	Não / Sim	Nominal	1 / 2
Conhecimentos em Português	Não / Sim	Nominal	1 / 2
Conhecimentos em Inglês	Não / Sim	Nominal	1 / 2
Conhecimentos em Espanhol	Não / Sim	Nominal	1 / 2
Experiência em venda de serviços	Não / Sim	Nominal	1 / 2

Tabela 3.1: Codificação dos atributos da base de dados original.

3.1.1 Identificação e correção de *Outliers*

Durante a etapa inicial de análise exploratória da base de dados, foi identificado um conjunto de inconsistências que comprometiam a integridade dos dados utilizados. Mais especificamente, observou-se que alguns valores presentes em determinadas colunas ultrapassavam os limites máximos definidos para os respetivos atributos, conforme especificado na Tabela 3.1. Essa incongruência acendeu um alerta quanto à possibilidade de erro humano na fase de construção da base, uma vez que os valores esperados para cada variável estavam previamente delimitados segundo critérios objetivos estabelecidos pela autora da base de dados original.

Diante dessa constatação, foi realizado contacto com a autora da base de dados a fim de verificar a origem dessas anomalias. A mesma confirmou que os erros se deviam à falhas manuais que ocorreram durante o processo de extração e/ou inserção dos dados, especialmente em atributos que demandavam interpretação subjetiva a partir dos conteúdos dos CVs. Após revisão minuciosa por parte da autora, os valores corretos foram recuperados e disponibilizados, permitindo a atualização da base de dados com informações fidedignas. Essa correção garantiu a validade das análises subsequentes e a confiabilidade dos modelos preditivos aplicados.

3.1.2 Padronização dos Dados

Ao examinar a base corrigida, foi identificado que os atributos numéricos apresentavam pontos de origem distintos. Enquanto alguns atributos eram codificados a partir do valor 0 (como, por exemplo, "Experiência no setor de vendas"), outros iniciavam em 1 (caso da quantidade de "Cursos em vendas"), conforme especificado na Tabela 3.1. Essa diferença de codificação pode introduzir viés nos modelos, especialmente naqueles baseados em distância (como o KNN ou SVM), ou que dependem de normalização para garantir convergência adequada, uma vez que tais algoritmos são sensíveis à distribuição dos valores e podem interpretar de forma distorcida a importância relativa de cada atributo (Soleimani et al., 2022).

Para mitigar esse problema e assegurar que todos os atributos fossem tratados de maneira equitativa pelos modelos, foi realizada uma padronização das variáveis. Em particular, os atributos originalmente iniciados em 1 foram ajustados para iniciar em 0, de modo a uniformizar a base e facilitar a aplicação de técnicas de normalização posteriormente, neste caso, o *Standard Scaling*, que será melhor descrito na secção 3.1.5. Essa uniformização não altera a informação semântica dos dados, mas contribui para a estabilidade numérica dos algoritmos de aprendizado, reduzindo vieses e acelerando o processo de otimização durante o treinamento (Ali & Faraj, 2014).

3.1.3 Codificação dos Dados

A base de dados utilizada neste estudo continha múltiplas variáveis categóricas nominais, ou seja, atributos que representam categorias sem relação ordinal entre si. Para que essas variáveis pudessem ser utilizadas em algoritmos de AM, foi necessário convertê-las em um formato numérico apropriado. O método adotado foi a codificação *One-Hot Encoding*, uma técnica amplamente reconhecida na literatura (Harris et al., 2020), que transforma cada categoria distinta em uma coluna binária separada, indicando a presença ou ausência daquela categoria em uma dada instância. Esse procedimento foi aplicado às seguintes variáveis nominais: `Genero`, `Carro`, `Exp_funcao`, `Promocao`, `Portugues`, `Ingles`, `Espanhol` e `Exp_vendas`.

Cada uma dessas colunas foi expandida em múltiplas colunas binárias, correspondentes a todas as categorias únicas observadas na base. Por exemplo: a variável `Ingles` foi dividida em colunas como `Ingles_0`, `Ingles_1`, dependendo da capacidade do candidato de falar a língua inglesa. Com isso, garantiu-se que a informação categórica fosse corretamente interpretada pelos algoritmos, sem introduzir ordens artificiais entre categorias, o que poderia ocorrer caso fossem utilizados outros métodos, como *Label Encoding*.

No que se refere à variável `Espanhol`, sua inclusão como atributo justifica-se pelo contexto geográfico do Brasil, país de origem de todos os candidatos. Considerando que seu território faz fronteira com diversas nações hispânicas na América do Sul, o domínio

da língua espanhola pode representar um diferencial competitivo para candidatos.

Quanto à variável alvo — ou saída — utilizada na modelagem, esta foi representada por um valor numérico inteiro no intervalo de 0 a 10, correspondente à nota atribuída ao currículo de cada candidato. Como os modelos utilizados são de natureza supervisionada, e a tarefa configurada é de regressão, não foi necessário aplicar codificações adicionais à variável de saída. Entretanto, para fins de estratificação durante a divisão dos dados e visualização das métricas, esta nota foi arredondada para o inteiro mais próximo, permitindo análises complementares baseadas em categorias discretas de desempenho.

3.1.4 Divisão dos Dados

Para simular um cenário realista de aplicação, 120 instâncias (20% da base) foram reservadas exclusivamente para a fase de teste final. Esse subconjunto permaneceu completamente isolado durante todas as etapas de treinamento, validação e ajuste de modelos, sendo utilizado apenas para a avaliação final dos algoritmos, de modo a representar a generalização do modelo frente a dados nunca vistos.

As 480 instâncias restantes constituíram o conjunto disponível para desenvolvimento (treinamento e validação). Sobre esse conjunto, foi adotada a técnica de validação cruzada estratificada com $k = 5$ (*5-Fold Stratified Cross-Validation*).

A figura 3.1 exibe o procedimento que consiste em particionar os dados em cinco subconjuntos aproximadamente iguais, preservando a distribuição da variável-alvo em cada partição. Em cada iteração, quatro subconjuntos são utilizados para o treinamento do modelo e o quinto para validação, alternando-se os papéis entre os subconjuntos até que todos tenham sido utilizados como validação. Essa abordagem reduz a variância das estimativas de desempenho e torna os resultados menos dependentes de uma divisão específica dos dados (Arlot & Celisse, 2010).

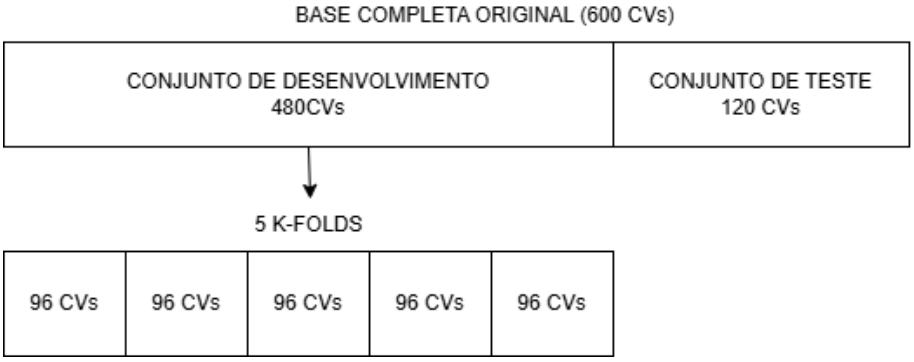

Figura 3.1: Divisão do conjunto de desenvolvimento. Fonte: Autoria Própria.

Dentro do conjunto de desenvolvimento, a validação cruzada serviu para selecionar a melhor arquitetura de modelo com base no desempenho médio entre todos os subconjuntos. Após a definição da arquitetura mais adequada, o modelo correspondente foi re-treinado com os 480 CVs originais e, somente então, submetido ao teste final com os 120 CVs previamente isolados. Essa metodologia visa simular uma situação de aplicação prática, em que novos dados são avaliados por um modelo previamente ajustado e validado.

Além da abordagem baseada exclusivamente nos dados reais, foram conduzidos dois experimentos adicionais com o objetivo de verificar o impacto da geração de dados sintéticos no desempenho dos modelos. Ambos seguiram a mesma estratégia de validação cruzada de 5 *folds*, diferenciando-se apenas pelo volume e composição do conjunto de dados (ver secção 3.1.6).

3.1.5 Normalização

A normalização dos dados é uma etapa de pré-processamento de conjuntos numéricos utilizados em modelos de AM, especialmente aqueles que são sensíveis à escala dos atributos, como é o caso da MLP, do KNN e também do SVM.

Neste trabalho, foi utilizada a técnica de padronização conhecida como *Standard Scaling*, que transforma cada atributo numérico da base de dados de forma que passe a apresentar média zero e desvio padrão igual a um. Matematicamente, a transformação é

dada por:

$$z = \frac{x - \mu}{\sigma} \tag{3.1}$$

onde:

- x representa o valor original da variável,
- μ é a média dos valores da variável na base de dados,
- σ é o desvio padrão dos valores da variável.

A aplicação dessa transformação garante que todas as variáveis estejam na mesma escala, evitando que atributos com maior magnitude exerçam influência desproporcional no processo de otimização dos algoritmos de aprendizado (Ali & Faraj, 2014). A padronização foi aplicada a todos os atributos numéricos da base de dados após o processo de limpeza e codificação, conforme descrito nas secções anteriores.

No presente caso, a escolha pelo *StandardScaler* se mostrou mais apropriada por diversos motivos. Primeiramente, os atributos numéricos da base estavam originalmente limitados a faixas pequenas — variando tipicamente de 0 a 8 — o que atenuava o risco de distorções causadas por valores extremos. Além disso, o objetivo principal era garantir que todas as variáveis contribuíssem de forma equilibrada para os modelos preditivos, e não necessariamente forçar um intervalo comum. Como os dados não apresentavam assimetrias marcantes, nem *outliers* severos, a padronização baseada na média e no desvio padrão foi suficiente para estabilizar o processo de treinamento.

3.1.6 Dados Sintéticos

No conjunto disponível para desenvolvimento (os 480 CVs separados anteriormente), foi aplicada a geração de dados sintéticos, com o objetivo de aumentar a diversidade e melhorar a representatividade do conjunto de treino. Essa estratégia é especialmente relevante em cenários com quantidade limitada de dados ou com desequilíbrio entre classes, como é o caso do presente estudo.

A técnica adotada consistiu na adição de ruído gaussiano aos dados existentes, gerando novas instâncias que preservam a estrutura estatística original dos dados, mas introduzem pequenas perturbações controladas. Essa abordagem permite simular variações realistas nos dados e melhora a capacidade dos modelos de generalizar para novos exemplos. Segundo Mariani et al. (2018), a introdução de ruído gaussiano é uma técnica eficaz na geração de dados sintéticos, especialmente em tarefas de aprendizado supervisionado, pois atua como uma forma de regularização, prevenindo o sobreajuste aos dados originais.

Duas abordagens distintas foram utilizadas para explorar o potencial dos dados sintéticos:

- **Aumento Proporcional com Preservação da Distribuição:** foram geradas instâncias sintéticas correspondentes a 50% do tamanho do conjunto de desenvolvimento. Os exemplos adicionados foram distribuídos proporcionalmente entre as classes, de modo a manter a distribuição original das notas atribuídas aos currículos. Deste modo totalizando 720 instâncias disponíveis na base de treino (480 reais + 240 sintéticas) como pode ser observado na figura 3.2. Esta estratégia visa enriquecer o conjunto de dados sem introduzir viés artificial nas distribuições.

- **Balanceamento de Classes:** teve como objetivo reduzir o desequilíbrio de classes na variável alvo. Para isso, foi identificado o número de instâncias na classe maioritária no conjunto de desenvolvimento (nota 7, com 82 exemplos), e, em seguida, todas as demais classes receberam instâncias sintéticas até igualarem essa quantidade. Ao final do processo, o conjunto de treino passou a conter 902 instâncias, sendo 480 reais e 422 sintéticas, como pode ser visto na figura 3.3. Esta estratégia busca fornecer ao modelo uma distribuição uniforme das classes durante o aprendizado, minimizando a tendência de superestimar classes mais frequentes.

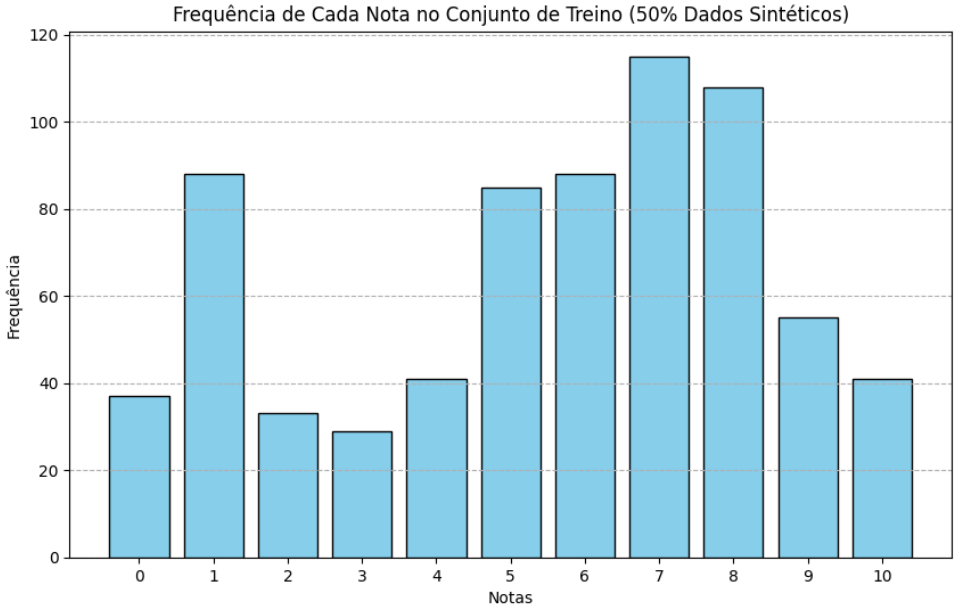

Figura 3.2: Distribuição das classes no conjunto de treino sintético proporcional.

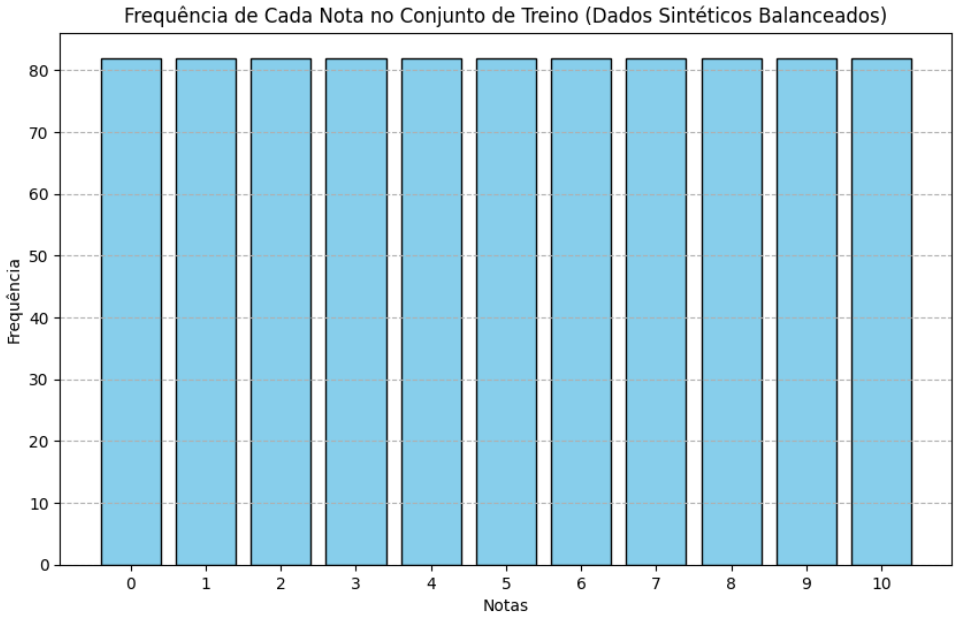

Figura 3.3: Distribuição das classes no conjunto de treino sintético balanceado.

Ambas as estratégias foram aplicadas exclusivamente sobre o conjunto de desenvolvimento, mantendo o conjunto de teste intocado para garantir uma avaliação imparcial do

desempenho dos modelos. A utilização de dados sintéticos, além de útil em domínios onde a obtenção de exemplos reais é custosa ou demorada (como é o caso da análise de CV) é considerada uma técnica promissora para ampliação de conjuntos de dados em tarefas supervisionadas. (Mariani et al., 2018).

3.2 Métricas de Desempenho

Para avaliar quantitativamente o desempenho dos modelos de regressão aplicados neste trabalho, foram utilizadas cinco métricas principais: MAE, MSE, RMSE, Coeficiente de Determinação (R^2) e a acurácia. Essas métricas oferecem perspetivas complementares sobre a qualidade das predições realizadas pelos modelos.

Antes de detalhar as métricas, é importante definir o conceito de **resíduo**. Em regressão, o resíduo é a diferença entre o valor real observado y_i e o valor previsto pelo modelo $\hat{y}_i$. Matematicamente, a Equação 3.2:

$$e_i = y_i - \hat{y}_i \tag{3.2}$$

Os resíduos representam o erro individual cometido para cada instância do conjunto de teste, sendo fundamentais para a análise de acurácia do modelo e também para a identificação de padrões sistemáticos de erro.

A primeira métrica considerada é o **Erro Absoluto Médio** (MAE, do inglês *Mean Absolute Error*). Ele é definido como a média das magnitudes dos resíduos, desconsiderando o sinal. Por ser uma métrica linear, todos os erros têm o mesmo peso na média, o que a torna mais robusta a *outliers* do que o MSE. O MAE é representado pela Equação 3.3:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.3}$$

Já o **Erro Quadrático Médio** (MSE, *Mean Squared Error*), apresentado na Equação 3.4, penaliza de forma mais intensa erros grandes, pois eleva os resíduos ao quadrado antes

de calcular a média. Por isso, é especialmente útil quando se deseja enfatizar predições com erros significativos.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.4}$$

Como o MSE possui unidade ao quadrado da variável de saída, é comum utilizar sua raiz quadrada, obtendo assim o **RMSE** (*Root Mean Squared Error*), que facilita a interpretação, vide Equação 3.5:

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.5}$$

Por fim, o **Coeficiente de Determinação** R^2 mede a proporção da variância da variável dependente que é explicada pelo modelo. Trata-se de uma métrica adimensional cujo valor varia entre $-\infty$ e 1, sendo que valores mais próximos de 1 indicam melhor ajuste. O cálculo do R^2 é realizado com base na Equação 3.6:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{3.6}$$

onde $\bar{y}$ é a média dos valores reais da variável dependente.

Um R^2 igual a 1 indica que o modelo explica perfeitamente a variância dos dados. Valores negativos sugerem que o modelo é pior do que uma simples média.

3.3 Deep Learning

O presente estudo optou por não empregar modelos baseados em *Deep Learning* (DL), considerando aspectos técnicos relacionados ao tipo de dados disponíveis, ao tamanho da amostra e à natureza da tarefa em questão. O conjunto de dados utilizado é composto por instâncias tabulares, com 14 atributos estruturados e apenas 600 exemplos. Apesar do uso de dados sintéticos para ampliação da base, as redes de DL geralmente exigem conjuntos de dados muito maiores para evitar o sobreajuste (*overfitting*) e garantir boa

capacidade de generalização (LeCun et al., 2015).

Embora redes neuronais profundas apresentem excelente desempenho em áreas como visão computacional, processamento de linguagem natural e reconhecimento de fala, isso ocorre principalmente devido à sua capacidade de aprender representações complexas a partir de dados não estruturados e com grande volume de informação. Nessas áreas, técnicas como a transferência de aprendizado (*Transfer Learning*) permitem reutilizar modelos pré-treinados em tarefas específicas, mesmo quando há pouca disponibilidade de dados rotulados (Tan et al., 2018).

Outro fator relevante diz respeito à interpretabilidade. Modelos de DL, que tendem a ser menos transparentes, dificultam a compreensão das decisões tomadas. Como um dos objetivos centrais deste estudo é promover a explicabilidade dos resultados, o uso de algoritmos supervisionados tradicionais, como KNN, SVM, RF e redes MLP, mostrou-se mais adequado devido ao seu desempenho satisfatório, baixo custo computacional e maior explicabilidade na tomada de decisão.

Capítulo 4

Desenvolvimento

4.1 Ferramentas e Ambiente de Desenvolvimento

A implementação da solução proposta foi realizada utilizando a linguagem de programação Python (Foundation, 2001), amplamente utilizada em projetos de ciência de dados e aprendizado de máquina devido à sua sintaxe acessível e ao suporte oferecido por um ecossistema maduro de bibliotecas voltadas para análise e modelagem de dados. Todos os modelos foram treinados e avaliados com base no mesmo conjunto de dados pré-processados, assegurando uma comparação justa entre os resultados obtidos.

Para o processamento e manipulação das informações tabulares, foram utilizadas as bibliotecas NumPy (NumPy, 2005) e Pandas (T. P. D. Team, 2008). O NumPy fornece estruturas de dados eficientes e operações vetoriais de alto desempenho (Harris et al., 2020). O Pandas, por sua vez, oferece funcionalidades robustas para leitura, transformação e análise de dados estruturados, facilitando etapas como agregação, filtragem e tratamento de valores ausentes (McKinney, 2010).

A análise exploratória dos dados e a criação de gráficos para visualização foram conduzidas por meio das bibliotecas Matplotlib (Hunter & Contributors, 2003) e Seaborn (M. Waskom & contributors, 2012). A Matplotlib permite a criação de gráficos detalhados e customizáveis, sendo amplamente empregada em artigos científicos e relatórios técnicos

(Hunter, 2007). Já a Seaborn oferece uma camada de abstração mais elevada, integrando-se ao Pandas para gerar gráficos estatísticos com maior facilidade e clareza visual (M. L. Waskom, 2021).

Para o treinamento e avaliação dos modelos preditivos, foi utilizada a biblioteca Scikit-learn (Scikit-learn, 2010), que disponibiliza uma coleção consolidada de algoritmos de aprendizado supervisionado e não supervisionado, além de ferramentas para validação cruzada, métricas de avaliação e rotinas de pré-processamento (Pedregosa et al., 2011).

Para a construção e o treinamento da MLP foi utilizado o TensorFlow (T. Team, 2015) e sua API, Keras (K. Team, 2015). O TensorFlow permite a definição de grafos computacionais e o uso de aceleração via GPU, enquanto o Keras simplifica o desenvolvimento de redes neuronais, facilitando o teste de diferentes arquiteturas e funções de ativação (Abadi et al., 2016; Chollet et al., 2015).

No que se refere à interpretabilidade dos modelos, foram utilizadas as bibliotecas SHAP (S. Lundberg & contributors, 2024), LIME (Ribeiro & contributors, 2016) e TreeInterpreter (Saabas, 2017).

O ambiente de desenvolvimento utilizado combinou recursos locais e computação em nuvem. Parte significativa dos experimentos foi conduzida no Google Colab (Google, 2017), que oferece acesso facilitado a recursos computacionais, além de permitir a execução de *notebooks* Jupyter (Jupyter, 2014) diretamente no navegador. Isso viabilizou o treinamento de modelos mais exigentes computacionalmente sem a necessidade de *hardware* local especializado.

Complementarmente, foi mantido um ambiente de desenvolvimento local com o editor Visual Studio Code (Microsoft, 2015) e uma estação de trabalho equipada com processador AMD Ryzen 9 5900X, placa-mãe AORUS B450 Elite e GPU NVIDIA GeForce GTX 1080. Esse ambiente foi utilizado principalmente para organização do código, controle de versionamento com Git e testes preliminares dos *scripts* em menor escala.

4.2 Tratamento dos Dados

A base original, composta por 600 currículos, foi dividida em dois subconjuntos principais: 480 instâncias (80%) foram destinadas ao desenvolvimento dos modelos, enquanto as 120 restantes (20%) foram reservadas exclusivamente para a etapa de teste final. Essa separação foi realizada com estratificação da variável-alvo, utilizando a função *train_test_split* da biblioteca *scikit-learn*, de modo a preservar a distribuição original das notas nos dois conjuntos e evitar viés amostral.

Sobre o subconjunto de desenvolvimento, foi aplicada validação cruzada estratificada com $k = 5$ folds. Esse procedimento permitiu avaliar o desempenho médio dos modelos e selecionar as arquiteturas mais promissoras, mantendo consistência na proporção das classes em cada partição de treino e validação.

A Figura 4.1 apresenta a distribuição das notas na base completa, enquanto a Figura 4.2 mostra a mesma distribuição no conjunto de teste isolado. Os gráficos demonstram que a estratégia de estratificação foi eficaz em manter a representatividade das diferentes faixas de notas em ambas as divisões.

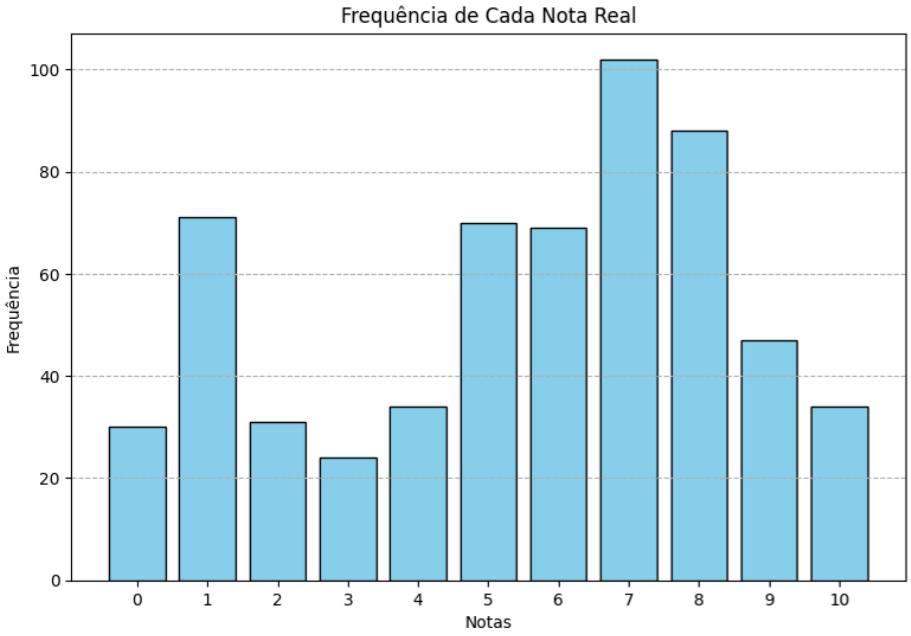

Figura 4.1: Distribuição das classes na base original completa.

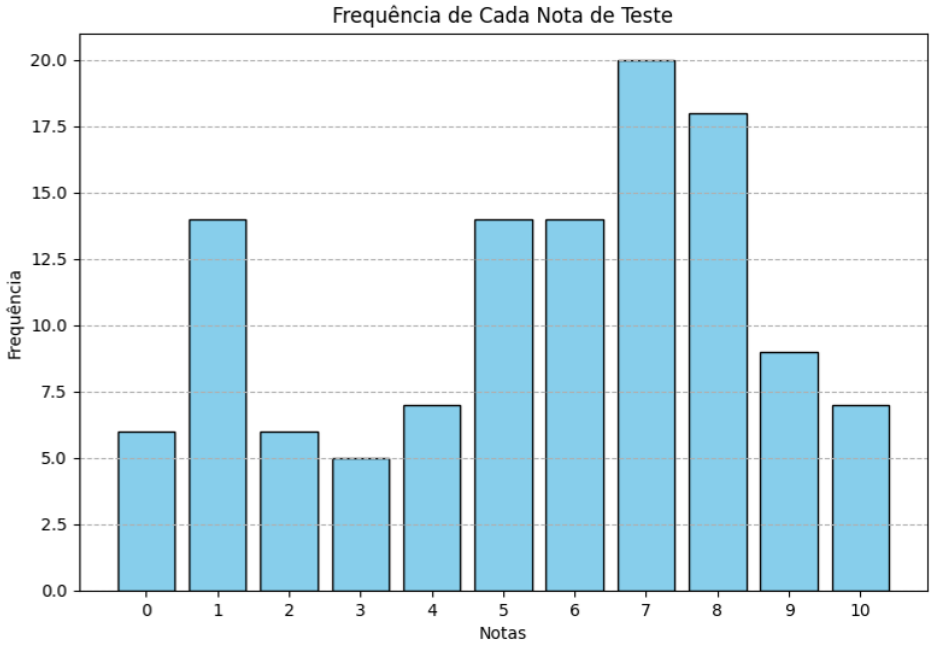

Figura 4.2: Distribuição das classes no conjunto de teste.

Na geração dos dados sintéticos, foi adicionado ruído gaussiano com média zero e desvio padrão de 0,1 às variáveis de entrada, após a normalização dos dados utilizando o *StandardScaler*, da biblioteca *scikit-learn*. Como o procedimento de normalização transforma os atributos em uma distribuição com média 0 e desvio padrão 1, a perturbação introduzida representa uma variação leve e proporcional, equivalente a 10% do desvio padrão. O ruído foi aplicado simultaneamente a todos os 14 atributos de cada instância, mantendo os valores da variável-alvo (nota atribuída ao currículo) inalterados. Essa estratégia permitiu a criação de novas amostras com pequenas variações em relação aos dados reais, preservando a coerência estatística e sem comprometer a consistência semântica dos rótulos.

A Tabela 4.1 apresenta um exemplo desse processo, comparando os valores padronizados de uma instância original com sua respetiva versão sintética. Nota-se que todas as variáveis foram suavemente alteradas, permanecendo próximas de seus valores iniciais, o que evidencia o controle na geração dos dados. Esse tipo de aumento da base de treino visa aprimorar a capacidade de generalização dos modelos, especialmente em cenários com

desbalanceamento ou quantidade limitada de exemplos.

Atributo	**Original Normalizada**	**Sintética**
Idade	0,12	0,19
Gênero	-0,45	-0,40
Educação	0,98	1,03
Possui Carro	-0,11	-0,08
Tempo em Vendas	1,36	1,25
Experiência na Função	-0,32	-0,29
Experiência em Outros	0,67	0,76
Treinamento	0,51	0,58
Curso na Área	-0,23	-0,19
Promoções	0,08	0,13
Português	0,03	0,07
Inglês	-0,61	-0,65
Espanhol	-0,58	-0,52
Experiência em Vendas	0,44	0,49
Nota	**9**	**9**

Tabela 4.1: Exemplo de aplicação de ruído gaussiano após padronização dos dados. Fonte: Autoria Própria.

4.3 Arquiteturas

Com o objetivo de avaliar o desempenho de diferentes abordagens de aprendizado supervisionado na tarefa de classificação de currículos, foram implementados quatro modelos distintos: KNN, SVM, RF e MLP. Todos os modelos foram desenvolvidos utilizando a biblioteca *scikit-learn*, com exceção do MLP, implementado por meio do *Keras* com *backend TensorFlow*. Cada algoritmo foi testado sob diferentes configurações internas com o intuito de explorar seu comportamento e identificar a melhor configuração para o problema

proposto.

4.3.1 K-Nearest Neighbors (KNN)

Para o algoritmo *K-Nearest Neighbors* (KNN), foi utilizada a implementação padrão do regressor KNN disponível na *scikit-learn*. A principal variável testada foi o número de vizinhos (`n_neighbors`), sendo avaliados os valores $K = 3$, $K = 5$ e $K = 7$. O objetivo dessa variação foi analisar o impacto do número de vizinhos mais próximos no desempenho do modelo, buscando um equilíbrio entre viés e variância. O modelo foi mantido com os parâmetros padrão, exceto pelo ajuste de K, e os dados foram previamente padronizados para assegurar a correta aplicação da métrica de distância euclidiana, que é sensível à escala das variáveis.

4.3.2 Support Vector Machine (SVM)

A SVM foi implementada por meio da classe *SVR* da *scikit-learn*, testando três variantes de *kernel*: linear, polinomial e radial RBF. Cada configuração foi ajustada manualmente com valores específicos para os hiperparâmetros C e ϵ (epsilon), de modo a controlar a regularização e a margem de tolerância ao erro, respetivamente. As três variações adotadas foram:

- **Kernel RBF:** $C = 10$, $\epsilon = 0{,}2$;
- **Kernel Linear:** $C = 1{,}0$, $\epsilon = 0{,}1$;
- **Kernel Polinomial (grau 3):** $C = 5{,}0$, $\epsilon = 0{,}2$, com $\gamma =$ scale e coeficiente coef0 $= 1$.

A escolha desses parâmetros se deu por conta de melhores resultados nos testes preliminares de desenvolvimento e buscou avaliar o comportamento da SVM sob diferentes formas de separação dos dados e complexidade da superfície de decisão, estimulando a capacidade de generalização do modelo.

4.3.3 Random Forest (RF)

O modelo RF foi desenvolvido a partir da implementação *RandomForestRegressor* da biblioteca *scikit-learn*. A principal variação aplicada refere-se ao número de estimadores (`n_estimators`), ou seja, o número de árvores na floresta. Foram testadas três configurações: 10, 100 e 1000 árvores. As demais configurações foram mantidas com seus valores padrão, exceto pelo uso de `random_state` com o valor 809 para garantir reprodutibilidade e `n_jobs` igual a -1, permitindo o uso de múltiplos núcleos do processador para acelerar o treinamento.

4.3.4 Multilayer Perceptron (MLP)

A MLP foi implementada com a biblioteca *Keras*, sendo utilizada em conjunto com o backend *TensorFlow*. Durante a fase de treinamento, foi adotada a função de ativação *Rectified Linear Unit* (ReLU) nas camadas ocultas. Essa função foi escolhida por sua simplicidade computacional, como pode ser notado na Equação 4.1, e por se adequar naturalmente ao contexto do problema, no qual as saídas previstas estão restritas ao intervalo de valores não negativos. Como a ReLU não produz resultados negativos, ela evita interpretações inconsistentes no processo de regressão.

$$f(x) = \max(0, x) \tag{4.1}$$

Adicionalmente, foi empregado o recurso de *Early Stopping*, técnica que monitoriza o desempenho do modelo no conjunto de validação e interrompe o treinamento quando os ganhos cessam. Essa técnica foi configurada para monitorizar a métrica de erro quadrático médio da validação (`val_loss`), com paciência de 10 épocas e tolerância mínima (`min_delta`) de 0,001, restaurando automaticamente os melhores pesos obtidos até o ponto de parada. O número máximo de épocas foi fixado em 500, mas observou-se que, na maioria dos casos, o treinamento foi interrompido antes da 180ª época, indicando uma boa convergência e estabilidade no processo de aprendizado da rede. Essa estratégia visa prevenir o sobreajuste (*overfitting*) e reduzir o tempo de treinamento, mantendo a

capacidade preditiva da rede.

Foram avaliadas três arquiteturas distintas, variando o número de camadas ocultas e a quantidade de neurónios por camada. Todas as configurações seguiram uma estrutura de formato cónico, ou seja, o número de neurónios decresce progressivamente à medida que se avança das camadas iniciais para as finais:

- **Modelo 1:** 23–20–12–6–1 (três camadas ocultas);
- **Modelo 2:** 23–12–6–1 (duas camadas ocultas);
- **Modelo 3:** 23–6–1 (uma camada oculta).

Essa escolha visa evitar o aumento desnecessário da complexidade do modelo e reduzir o risco de sobreajuste, além de manter a coerência com a dimensionalidade da entrada. De acordo com Srivastava et al. (2014), não é recomendável que uma camada oculta possua mais neurónios do que a camada anterior, especialmente a camada de entrada. Isso porque a adição de unidades sem justificativa baseada em complexidade dos dados pode levar à geração de representações redundantes ou superdimensionadas, comprometendo a eficiência do modelo.

A fim de garantir robustez estatística, foi realizada uma validação cruzada estratificada com 5 *folds* (StratifiedKFold (scikit-learn, 2011)) no conjunto de desenvolvimento. Assim como previamente exposto na Figura 3.1 foram utilizados 4 *folds* para treino e 1 *fold* para validação, com estratificação da variável-alvo. Para cada subdivisão, foram treinadas 50 instâncias independentes da MLP com inicializações aleatórias distintas, e o modelo com menor `val_loss` foi selecionado como representante daquele *fold*.

Ao fim da validação cruzada, selecionou-se a arquitetura que obteve o melhor desempenho médio entre os *folds*. Essa arquitetura foi então re-treinada utilizando a totalidade dos dados de treinamento (80% da base original), para que só então fosse avaliado de forma independente sobre o conjunto de teste final, composto por 120 currículos nunca utilizados em nenhuma etapa anterior do treinamento.

Apesar de seu desempenho elevado em tarefas com padrões não lineares, o MLP é considerado um modelo menos transparente, dificultando a interpretação direta de suas decisões. Por esse motivo, na Secção 5.2 serão apresentadas técnicas de explicabilidade, como LIME e SHAP, que permitem uma análise interpretativa do comportamento do modelo, elucidando quais atributos contribuíram mais significativamente para as predições realizadas.

Capítulo 5

Resultados e Discussão

Este capítulo apresenta os resultados obtidos com a aplicação dos modelos de aprendizado supervisionado propostos: KNN, RF, SVM e MLP. Cada modelo foi avaliado com base em métricas de desempenho, distribuição dos resíduos e interpretação da matriz de confusão. Os resultados visam comparar não apenas a acurácia dos modelos, mas também a qualidade das previsões e a explicabilidade nas decisões de cada arquitetura.

5.1 Modelos

5.1.1 KNN

Com base nos valores apresentados na Tabela 5.1, determinados usando o conjunto de teste, o modelo KNN com $K = 3$ e 50% de dados sintéticos obteve MAE de 0,38 e MSE de 0,52, resultando em um RMSE de 0,72. O coeficiente de determinação R^2 foi de 0,94, indicando que o modelo foi capaz de explicar 94% da variância das notas reais. Esses resultados destacam que, embora não tenha sido o melhor resultado absoluto dentre todos os tipos de arquiteturas experimentadas, a eficiência do KNN considerando sua simplicidade e ausência de aprendizado explícito de parâmetros é muito satisfatória.

Modelo	Base de Treino	MAE	MSE	RMSE	R^2
KNN Regressor ($K = 3$)	Original	0,45	0,55	0,74	0,93
KNN Regressor ($K = 5$)	Original	0,50	0,63	0,80	0,92
KNN Regressor ($K = 7$)	Original	0,48	0,57	0,75	0,93
KNN Regressor ($K = 3$)	**50% Sintético**	**0,38**	**0,52**	**0,72**	**0,94**
KNN Regressor ($K = 5$)	50% Sintético	0,42	0,54	0,74	0,94
KNN Regressor ($K = 7$)	50% Sintético	0,49	0,62	0,79	0,93
KNN Regressor ($K = 3$)	Sintético Balanceado	0,46	0,66	0,81	0,92
KNN Regressor ($K = 5$)	Sintético Balanceado	0,51	0,72	0,85	0,91
KNN Regressor ($K = 7$)	Sintético Balanceado	0,53	0,69	0,83	0,92

Tabela 5.1: Resultados dos modelos KNN.

O modelo KNN com $k = 3$, utilizando a base com 50% de dados sintéticos, apresentou o melhor desempenho entre as variantes testadas, sendo escolhido para análise detalhada. A Figura 5.1 apresenta a dispersão dos resíduos em relação aos valores preditos, para cada valor da classificação real dos currículos. Observa-se uma concentração significativa de pontos ao redor da linha de erro zero, indicando previsões exatas. O tamanho dos círculos representa a frequência relativa de cada combinação de predição e erro, sendo notável a distribuição equilibrada de resíduos positivos e negativos, sem padrões evidentes de viés.

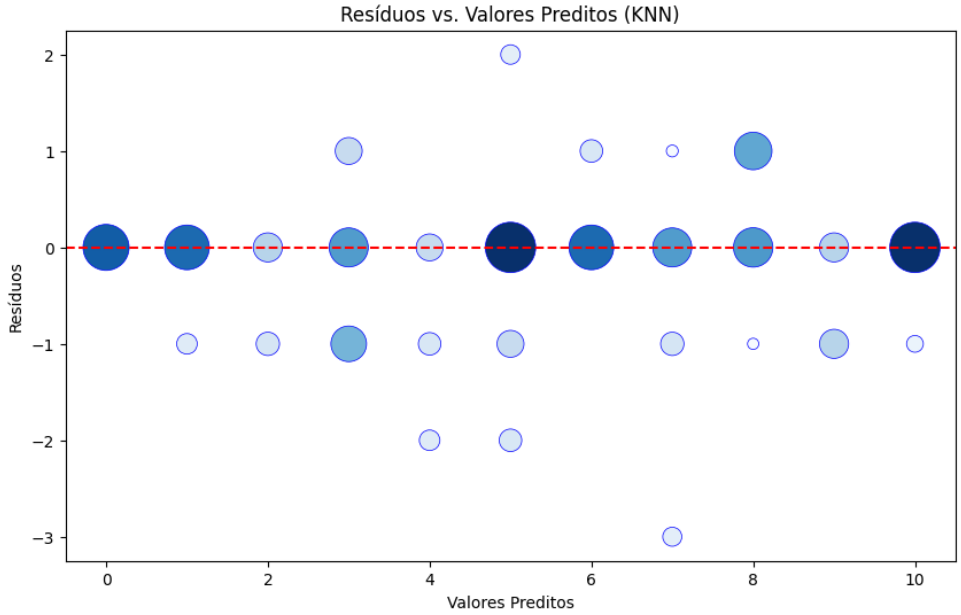

Figura 5.1: Gráfico de resíduos para o melhor modelo KNN.

O histograma dos resíduos, mostrado na Figura 5.2, reforça essa observação. A maioria dos resíduos está concentrada em torno do valor zero, com poucos desvios superiores a duas unidades. Essa distribuição estreita indica estabilidade nas predições, ainda que uma leve assimetria negativa sugira casos pontuais de subestimação por parte do modelo.

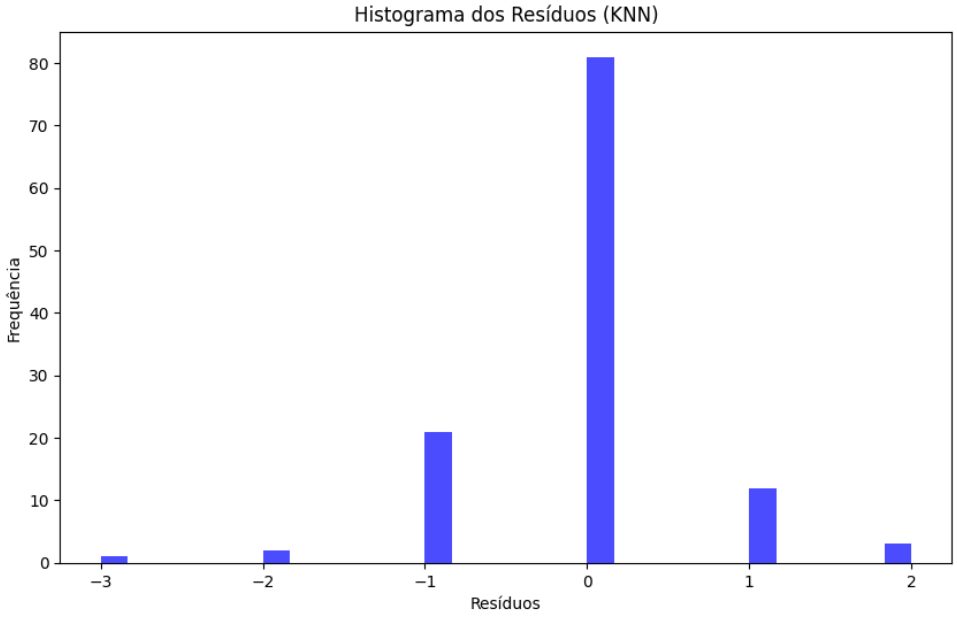

Figura 5.2: Histograma dos resíduos do melhor modelo KNN.

A matriz de confusão da Figura 5.3 mostra que os maiores acertos estão concentrados na diagonal principal, representando correspondência direta entre notas previstas e reais. A maior parte dos erros ocorre entre classes vizinhas, o que é esperado em problemas com variáveis de saída ordinais, como notas em uma escala de 0 a 10. A região central da matriz, que concentra as classes com maior volume de dados, é a que apresenta maior dispersão de erros. Ainda observando a matriz de confusão da Figura 5.3, pode-se constatar um total de 81 acertos precisos no conjunto de testes, equivalente a 67,50% de acurácia, avaliada em 11 classes.

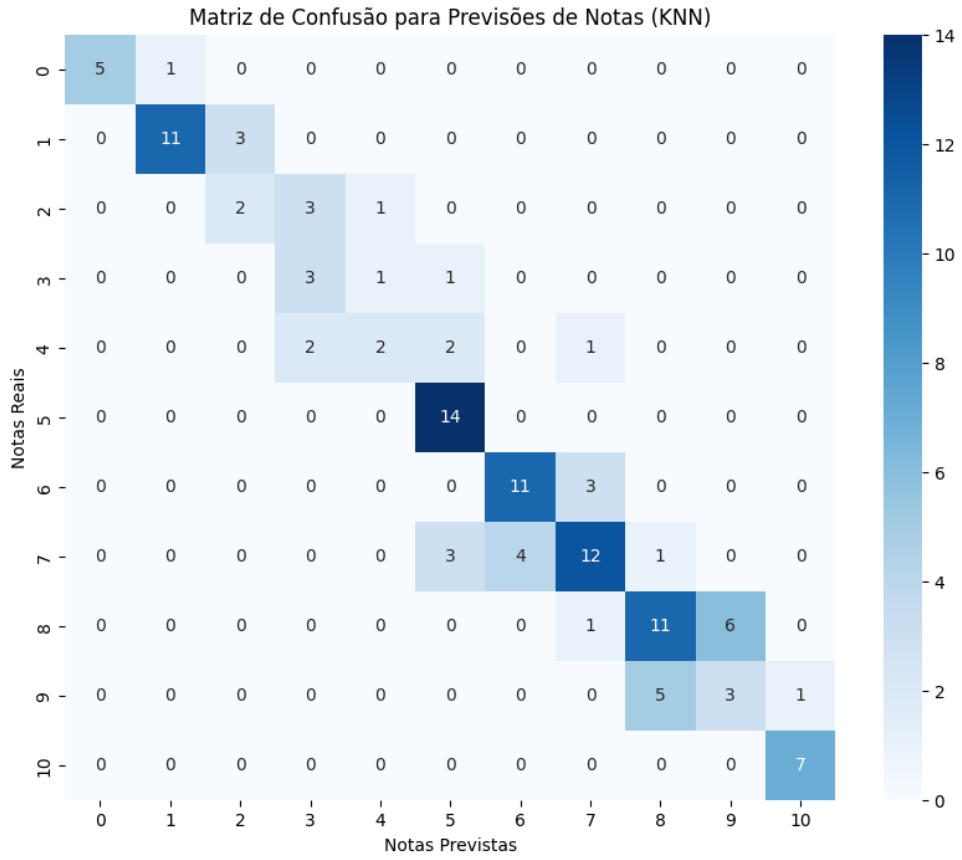

Figura 5.3: Matriz de confusão do melhor modelo KNN.

5.1.2 *Random Forest*

Modelo	Base de Treino	MAE	MSE	RMSE	R^2
RF (10 árvores)	Original	0,28	0,53	0,73	0,94
RF (100 árvores)	Original	0,23	0,43	0,66	0,95
RF (1000 árvores)	Original	0,22	0,40	0,63	0,95
RF (10 árvores)	50% Sintético	0,17	0,18	0,43	0,98
RF (100 árvores)	50% Sintético	0,16	0,17	0,42	0,98
RF (1000 árvores)	**50% Sintético**	**0,14**	**0,16**	**0,40**	**0,98**
RF (10 árvores)	Sintético Balanceado	0,18	0,20	0,45	0,98
RF (100 árvores)	Sintético Balanceado	0,19	0,21	0,46	0,98
RF (1000 árvores)	Sintético Balanceado	0,17	0,19	0,44	0,98

Tabela 5.2: Resultados dos modelos Random Forest.

Entre todas as configurações testadas, o modelo *Random Forest* (RF) com 1000 árvores treinado sobre a base com 50% de dados sintéticos foi o que apresentou o melhor desempenho. Como mostra a Tabela 5.2, ele alcançou um MAE de 0,14 e um RMSE de 0,40, com coeficiente de determinação $R^2 = 0{,}98$, indicando excelente ajuste aos dados.

A Figura 5.4 ilustra os resíduos em função dos valores preditos. Nota-se uma forte concentração de pontos sobre a linha de erro zero, com poucas dispersões significativas. A distribuição é densa e simétrica, sugerindo previsões consistentes e sem viés sistemático.

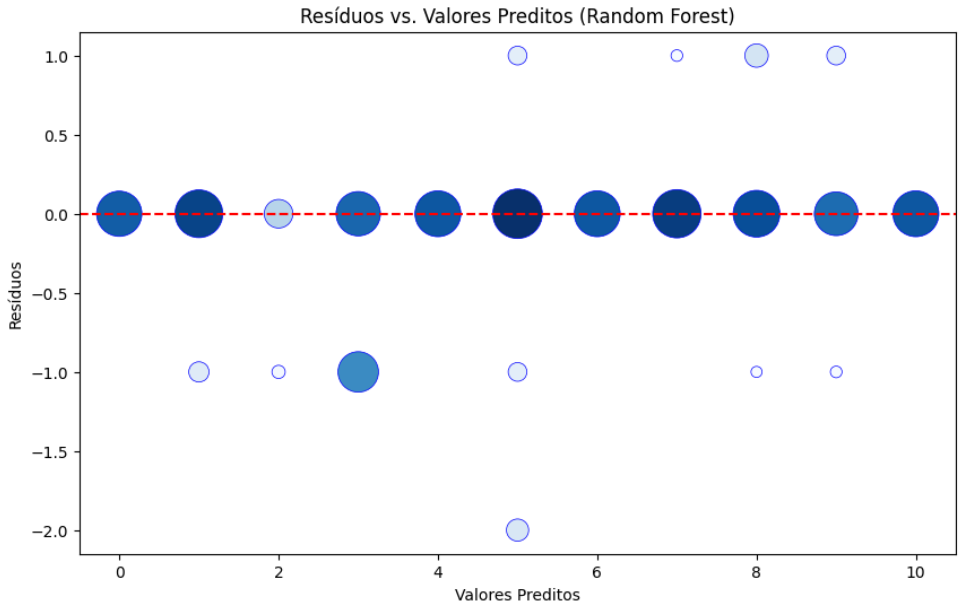

Figura 5.4: Gráfico de resíduos para o melhor modelo *Random Forest*.

Esse padrão é reforçado pelo histograma de resíduos apresentado na Figura 5.5. Exatamente 104 instâncias apresentaram erro nulo, enquanto os demais 15 resíduos se mantiveram entre -1 e +1, com apenas uma única instância de treino que obteve um erro de magnitude -2.

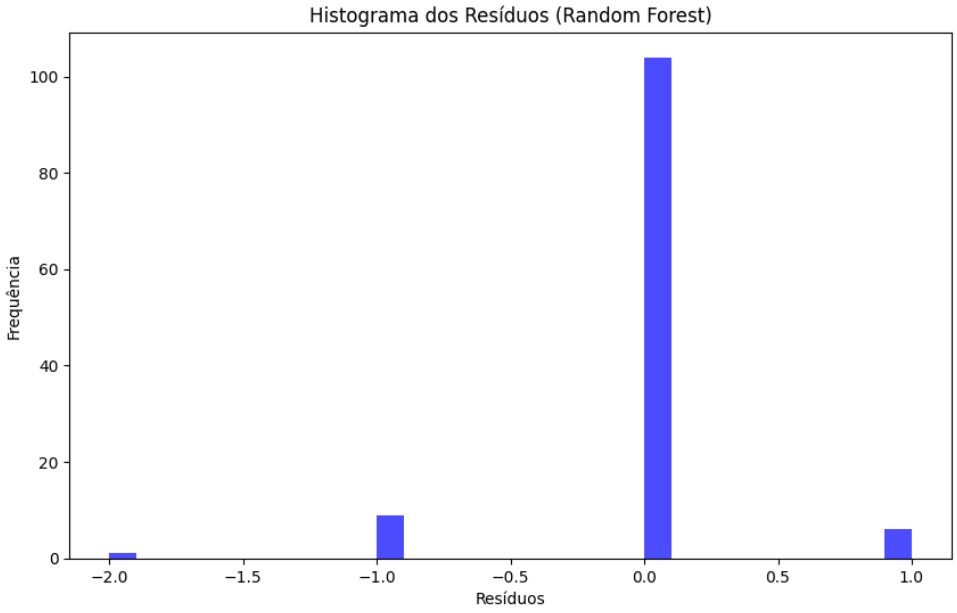

Figura 5.5: Histograma dos resíduos do melhor modelo *Random Forest*.

A matriz de confusão na Figura 5.6 evidencia a elevada precisão do modelo. Os maiores valores estão concentrados na diagonal principal, refletindo acertos exatos nas predições. Quando há erro, ele ocorre maioritariamente em classes vizinhas, o que é aceitável no contexto de notas ordinais. Esse padrão de erro suave indica boa estabilidade nas decisões do modelo, mesmo em casos de ambiguidade.

Os resultados obtidos reforçam o potencial do modelo RF como uma das abordagens mais eficazes para tarefas de regressão envolvendo variáveis discretas. Isso se deve ao fato de sua estrutura ser baseada em múltiplas árvores de decisão, que operam por meio de regras condicionais (*se-então*), similar ao processo lógico utilizado por profissionais de recrutamento, o que pode justificar seu destaque ao contexto da triagem de currículos.

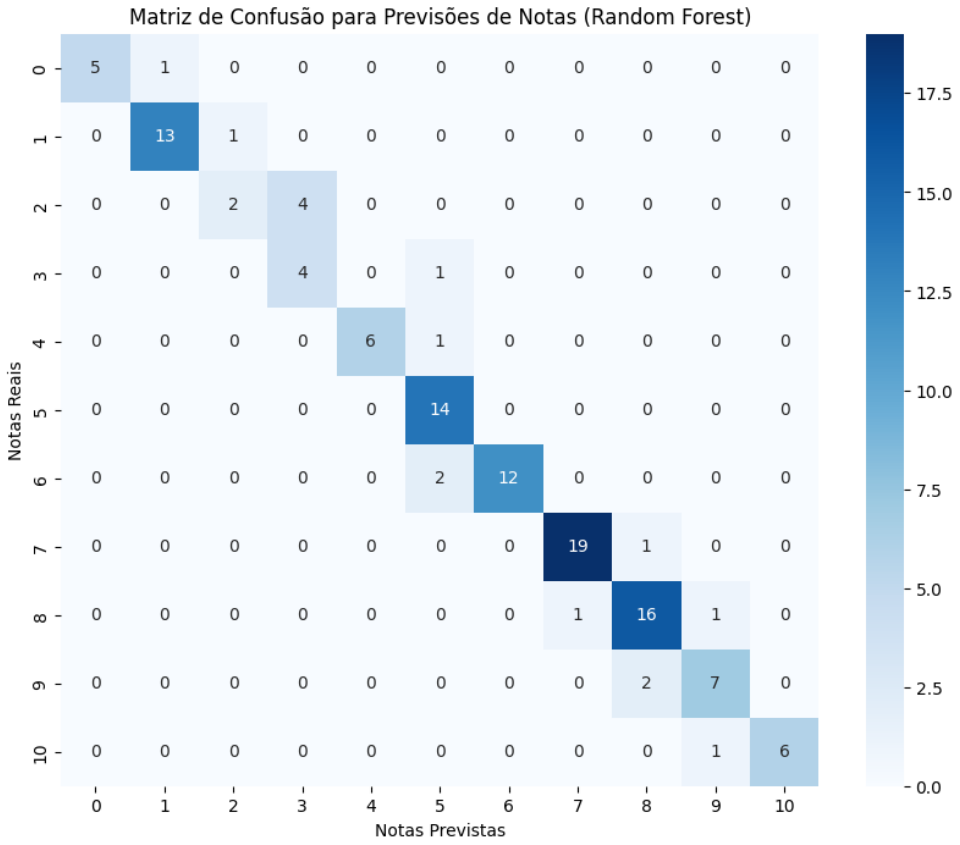

Figura 5.6: Matriz de confusão do melhor modelo *Random Forest*.

5.1.3 SVM

Modelo	**Base de Treino**	**MAE**	**MSE**	**RMSE**	R^2
SVM (*Kernel* RBF)	Original	0,31	0,47	0,69	0,94
SVM (*Kernel* linear)	Original	0,40	0,48	0,70	0,94
SVM (*Kernel* polinomial)	Original	0,28	0,50	0,71	0,94
SVM (*Kernel* RBF)	**50% Sintético**	**0,22**	**0,25**	**0,50**	**0,97**
SVM (*Kernel* linear)	50% Sintético	0,37	0,43	0,66	0,95
SVM (*Kernel* polinomial)	50% Sintético	0,24	0,29	0,54	0,97
SVM (*Kernel* RBF)	Sintético Balanceado	0,22	0,27	0,52	0,97
SVM (*Kernel* linear)	Sintético Balanceado	0,39	0,46	0,68	0,95
SVM (*Kernel* polinomial)	Sintético Balanceado	0,23	0,28	0,53	0,97

Tabela 5.3: Resultados dos modelos SVM.

Entre os modelos de SVM avaliados, a versão com *kernel* do tipo RBF apresentou o melhor equilíbrio entre os erros absolutos e quadráticos, atingindo um R^2 de 0,97 com o uso de dados sintéticos. A análise dos resíduos, ilustrada na Figura 5.7, mostra uma boa concentração de erros em torno de zero, sugerindo previsões estáveis. A maioria das instâncias se distribui de forma simétrica, com baixa incidência de desvios elevados, o que reforça a capacidade preditiva do modelo mesmo diante de padrões não lineares.

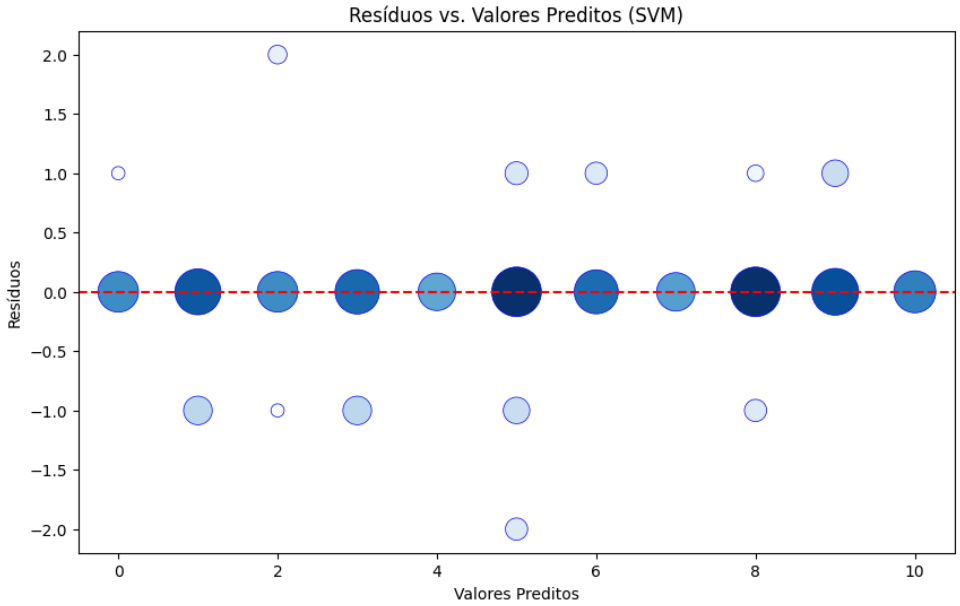

Figura 5.7: Gráfico de resíduos para o melhor modelo SVM.

O histograma da Figura 5.8 complementa essa análise, demonstrando que os resíduos mais comuns são próximos de zero, com distribuição aproximadamente simétrica. Pequenos erros de ± 1 são frequentes e aceitáveis dado o caráter discreto da variável resposta, enquanto desvios mais extremos ocorrem com baixa frequência.

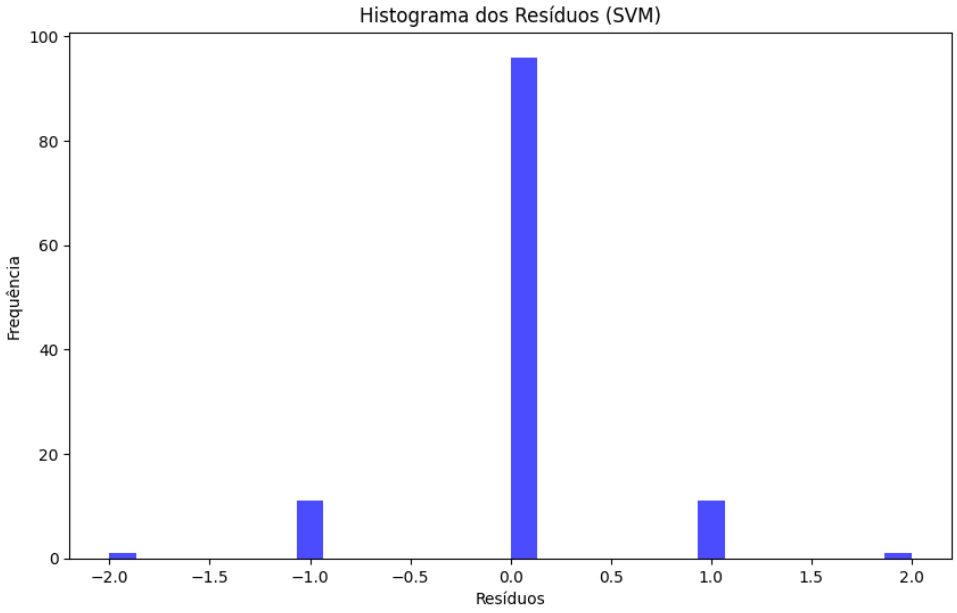

Figura 5.8: Histograma dos resíduos do melhor modelo SVM.

A Figura 5.9 exibe a matriz de confusão com os valores previstos arredondados para o inteiro mais próximo. Observa-se que a maioria das predições se alinha à diagonal principal, o que indica concordância entre as notas reais e previstas. Os erros mais recorrentes se concentram em classes vizinhas, especialmente entre as faixas de 4 a 8, o que é coerente com a natureza ordinal das notas.

Em termos de desempenho, os resultados obtidos com o *kernel* RBF — MAE de 0,22, MSE de 0,25 e RMSE de 0,50 — destacam a eficácia da modelagem com funções de *kernel* em problemas com dados não linearmente separáveis. Embora o *kernel* RBF tenha obtido os melhores resultados dentro das arquiteturas de SVM, os demais *kernels* (linear e polinomial) também apresentaram desempenho satisfatório em todos os cenários avaliados.

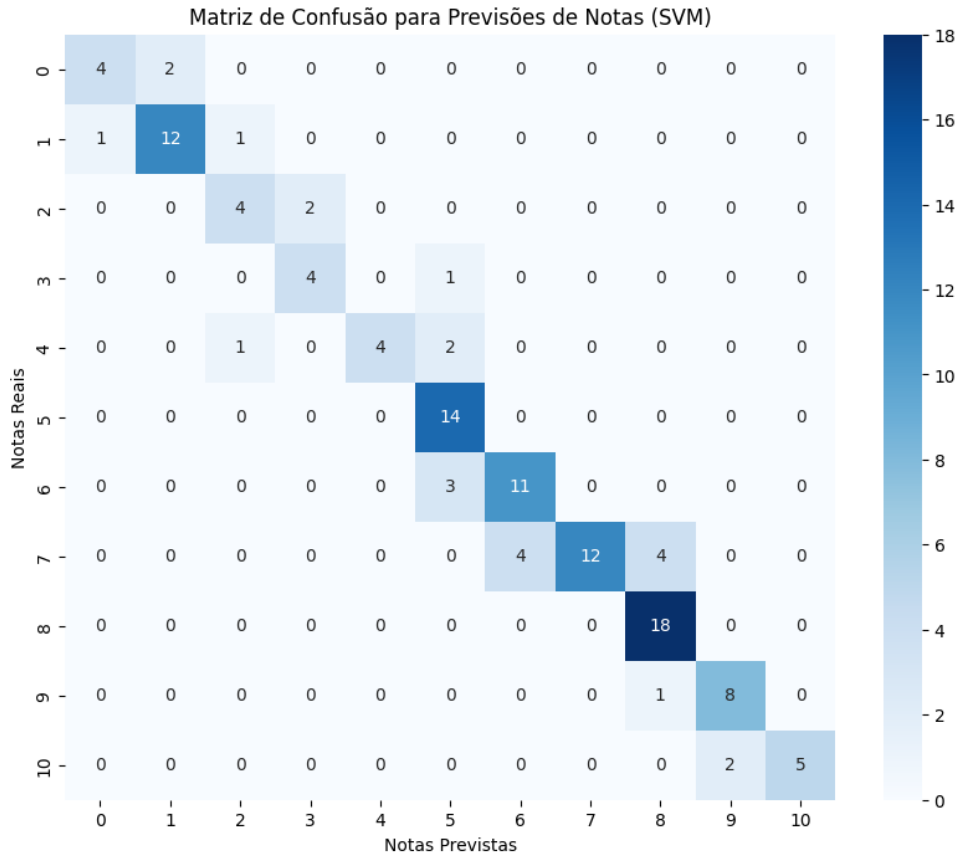

Figura 5.9: Matriz de confusão do melhor modelo SVM.

5.1.4 MLP

Modelo	Base de Treino	MAE	MSE	RMSE	R^2
MLP (23-6-1)	Original	0,62	0,91	0,95	0,89
MLP (23-12-6-1)	Original	0,32	0,53	0,73	0,94
MLP (23-20-12-6-1)	Original	0,41	0,61	0,78	0,93
MLP (23-6-1)	50% Sintético	0,42	0,47	0,68	0,94
MLP (23-12-6-1)	**50% Sintético**	**0,23**	**0,28**	**0,53**	**0,97**
MLP (23-20-12-6-1)	50% Sintético	0,28	0,32	0,56	0,96
MLP (23-6-1)	Sintético Balanceado	0,40	0,43	0,66	0,95
MLP (23-12-6-1)	Sintético Balanceado	0,28	0,34	0,58	0,96
MLP (23-20-12-6-1)	Sintético Balanceado	0,32	0,38	0,62	0,95

Tabela 5.4: Resultados dos modelos MLP.

Dentre as arquiteturas testadas, o modelo MLP (23-12-6-1), treinado com 50% de dados sintéticos, foi o que apresentou melhor desempenho geral. Com MAE de 0,23, MSE de 0,28 e RMSE de 0,53, o modelo demonstrou elevada capacidade de generalização e precisão nas previsões. O coeficiente de determinação de $R^2 = 0{,}97$ reforça sua adequação à tarefa de regressão aplicada aos dados de currículos.

A Figura 5.10 apresenta a dispersão dos resíduos em relação às previsões realizadas, evidenciando concentração ao redor do zero. A distribuição homogénea e com baixa variância indica que o modelo não apresentou viés sistemático nas predições. Já o histograma da Figura 5.11 reforça esse padrão, com pico acentuado em torno do erro nulo e queda rápida nas frequências para valores absolutos maiores, o que sugere estabilidade no comportamento preditivo.

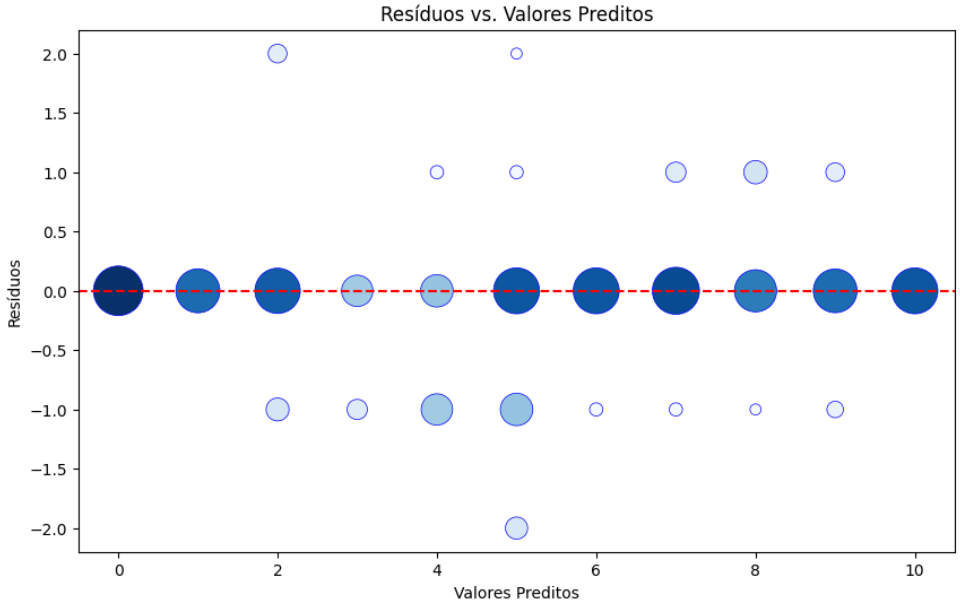

Figura 5.10: Gráfico de resíduos para o melhor modelo MLP.

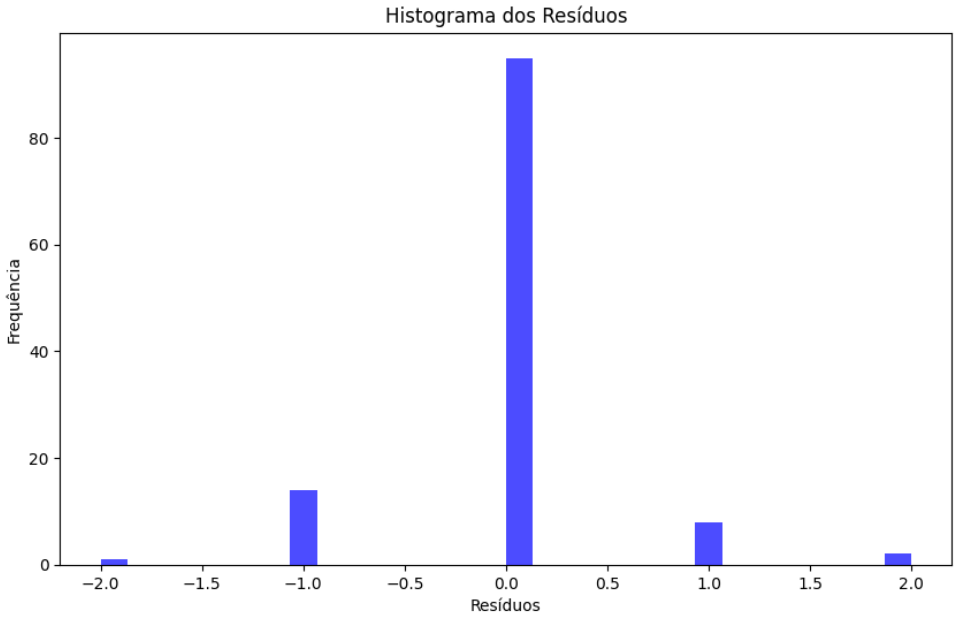

Figura 5.11: Histograma dos resíduos do melhor modelo MLP.

Na Figura 5.12, observa-se que a maioria das predições está localizada na diagonal da matriz de confusão, o que indica concordância entre os valores previstos e os valores reais. Os poucos erros observados ocorrem em sua maioria com diferença de apenas uma unidade, o que é aceitável para o tipo de variável alvo considerada neste estudo.

Foram testadas arquiteturas com diferentes profundidades, variando de uma a três camadas ocultas. A introdução de uma segunda camada resultou em melhorias expressivas, como observado no modelo 23-12-6-1. No entanto, a adição de uma terceira camada (modelo 23-20-12-6-1) não contribuiu para aumento de desempenho e, em alguns casos, resultou em leve degradação, o que pode ser atribuído à sobreparametrização. A ausência de ganhos com maior profundidade levou à decisão de não empregar técnicas adicionais de regularização como *dropout* ou *batch normalization*. A arquitetura 23-12-6-1, por conter mais de uma camada oculta com funções de ativação não lineares, pode ser considerada uma rede profunda.

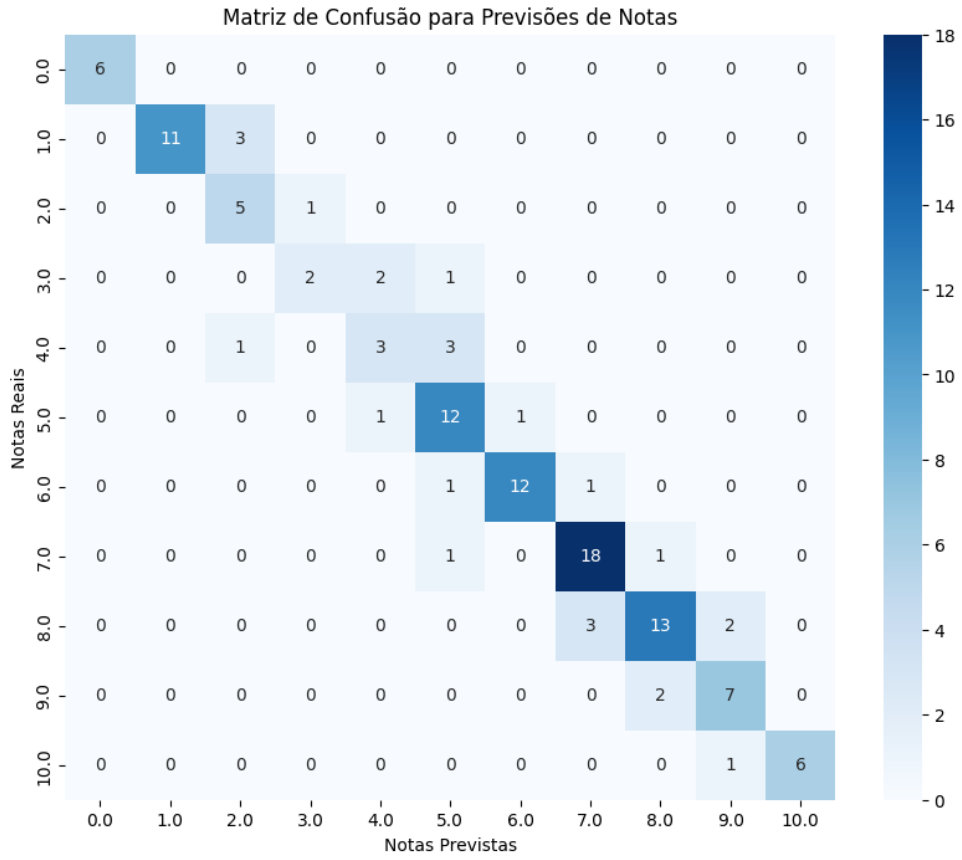

Figura 5.12: Matriz de confusão do melhor modelo MLP.

5.2 Explicabilidade

A utilização de modelos de aprendizado de máquina em contextos sensíveis, como a avaliação de currículos, demanda mais do que desempenho estatístico. É fundamental compreender o racional que fundamenta as decisões algorítmicas, especialmente quando estas impactam pessoas. Nesse sentido, métodos de explicabilidade pós-hoc (SHAP e LIME) e `TreeInterpreter` foram aplicados aos modelos desenvolvidos para interpretar, de forma local ou global, como cada variável de entrada influenciou a predição.

5.2.1 SHAP

A Figura 5.13 apresenta o *summary plot* gerado com o SHAP para o modelo MLP que obteve o melhor resultado. Cada ponto representa uma instância do conjunto de teste, com a posição no eixo horizontal indicando o valor SHAP (impacto na predição), e a cor refletindo o valor do atributo (do mais baixo em azul ao mais alto em vermelho). A ordenação vertical das variáveis reflete a ordem da sua importância média.

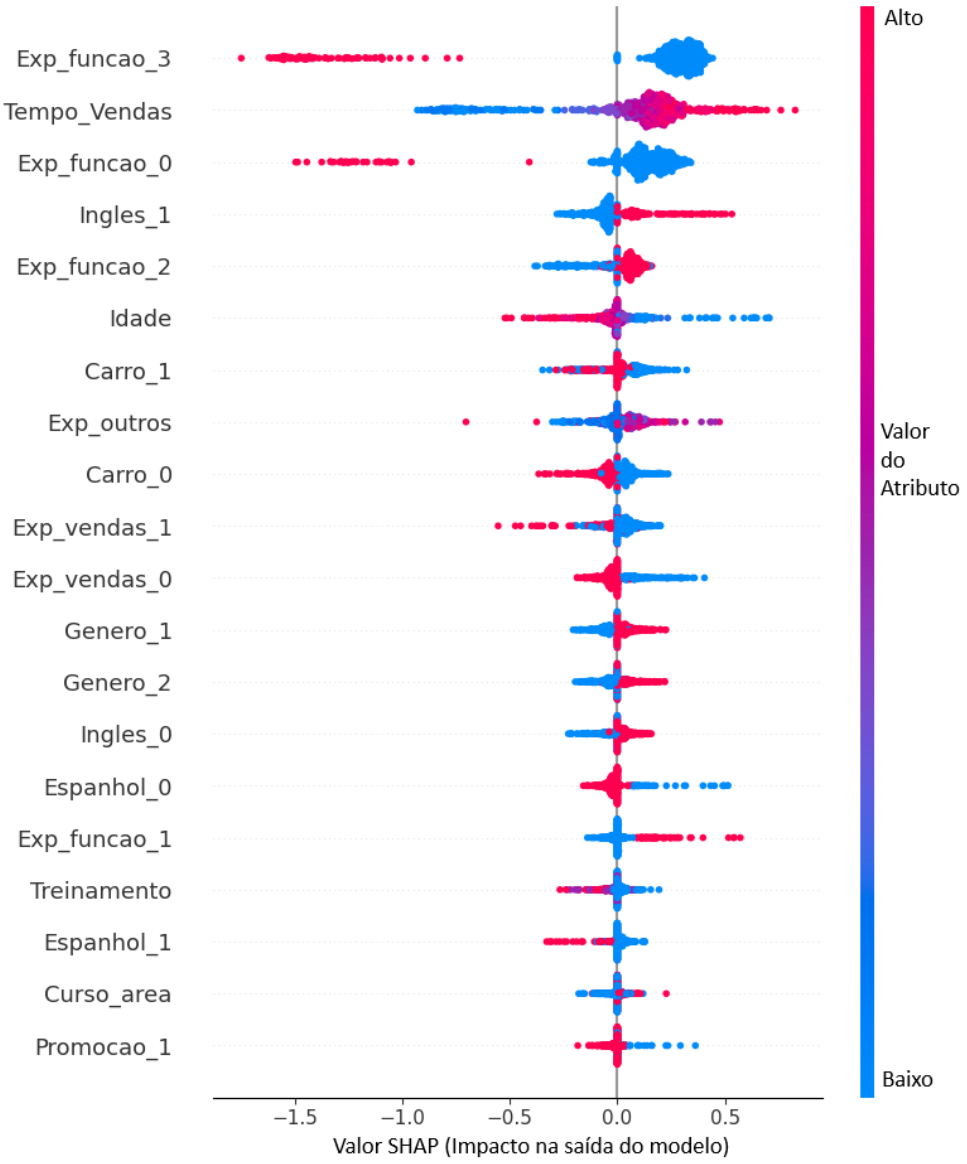

Figura 5.13: Gráfico de resumo do SHAP para o modelo MLP.

A interpretação do gráfico permite observar que variáveis como `Exp_funcao_3` (Experiência em vendas como gestor) e `Exp_funcao_0` (Não se aplica experiência em vendas) têm forte impacto negativo na nota prevista quando apresentam valores altos (vermelho à esquerda). Por consequência, quando seus valores são baixos, nota-se uma influência positiva relevante na avaliação do modelo. Isso sugere que nem toda experiência anterior é interpretada pelo modelo como compatível com o perfil buscado. Considerando que a vaga analisada é voltada ao cargo de consultor de vendas, é possível que a falta de experiências ou experiências excessivamente avançadas — como funções de liderança ou gestão — sejam interpretadas como um desalinhamento com a posição pretendida. Isso não implica, necessariamente, baixa qualificação do candidato, mas sim uma possível inadequação ao escopo da função analisada, o que reforça a importância de considerar o contexto do cargo ao interpretar os resultados gerados pelo modelo.

Por outro lado, variáveis como `Tempo_Vendas` (Tempo de experiência no setor de vendas) e `Ingles_1` (Conhecimentos em inglês) quando com valor elevado (vermelho à direita), aumentam a nota prevista. Demonstrando assim uma tendência positiva do modelo ao avaliar candidatos com experiências adequadas ao escopo da vaga e que possuem conhecimento da língua inglesa.

Outro aspeto relevante observado na análise diz respeito ao impacto do género dos candidatos nas predições do modelo. Os valores atribuídos às variáveis `Genero_1` (Masculino) e `Genero_2` (Feminino) apresentaram distribuições semelhantes de impacto positivo e negativo sobre a saída, sem predominância significativa de um dos géneros. Esse comportamento sugere que, no contexto da vaga analisada, o género dos candidatos não exerce influência expressiva nas decisões do modelo, indicando a imparcialidade quanto a esse atributo.

Por fim, a variável relacionada à `Idade` dos candidatos também apresentou um padrão interpretável. Observou-se que valores mais baixos para esse atributo (representados em azul à direita) influenciam positivamente a nota prevista pelo modelo, enquanto valores mais altos (vermelho à esquerda) estão associados a uma leve redução na avaliação. Esse

comportamento indica uma tendência do modelo em atribuir notas mais elevadas a candidatos mais jovens. Embora essa influência não seja determinante isoladamente, ela sugere que a variável idade pode estar correlacionada a outros fatores considerados relevantes para o perfil da vaga.

5.2.2 LIME

A seguir, apresenta-se o resultado da aplicação do LIME para a MLP que obteve melhores resultados, utilizando uma amostra real do conjunto de teste. O candidato recebeu nota real 8 e o modelo previu corretamente esse valor. Os dados originais (pré-normalização) da amostra são apresentados na Tabela 5.5.

Atributo	**Valor Original**
Idade	2
Educacao	3
Tempo_Vendas	6
Exp_outros	3
Treinamento	2
Curso_area	1
Genero_1	0
Genero_2	1
Carro_0	0
Carro_1	1
Exp_funcao_0	0
Exp_funcao_1	0
Exp_funcao_2	1
Exp_funcao_3	0
Promocao_1	1
Promocao_2	0
Portugues_1	1
Ingles_0	1
Ingles_1	0
Espanhol_0	1
Espanhol_1	0
Exp_vendas_0	0
Exp_vendas_1	1

Tabela 5.5: Valores originais da amostra explicada pelo LIME.

A Tabela 5.6 apresenta as regras extraídas pelo LIME para uma instância representativa com nota prevista e real igual a 8. Cabe destacar que os dados de entrada utilizados pelo modelo foram previamente normalizados. Assim, os intervalos apresentados na primeira coluna da tabela refletem condições estabelecidas sobre essas variáveis padronizadas, indicando os limites dentro dos quais o valor do atributo da instância analisada se encontra. Esse formato permite que o modelo linear local estimado pelo LIME quantifique a influência marginal de cada condição na predição final, conforme mostrado na segunda coluna.

Condição (normalizada)	**Contribuição para a predição**
$-1{,}54 <$ `Exp_funcao_2` $\leq 0{,}65$	$+3{,}25$
$0{,}04 <$ `Tempo_Vendas` $\leq 0{,}86$	$+1{,}32$
`Exp_funcao_1` $\leq -0{,}36$	$-0{,}54$
`Carro_0` $\leq -0{,}94$	$+0{,}30$
`Espanhol_0` $\leq 0{,}28$	$-0{,}23$
$-0{,}70 <$ `Treinamento` $\leq 0{,}94$	$+0{,}23$
$-0{,}01 <$ `Educacao` $\leq 0{,}09$	$+0{,}13$
`Exp_vendas_0` $\leq 0{,}35$	$+0{,}12$
`Ingles_1` $\leq -0{,}61$	$+0{,}11$
`Exp_funcao_3` $\leq -0{,}39$	$+0{,}09$
`Genero_1` $\leq -1{,}18$	$+0{,}08$
$-0{,}85 <$ `Genero_2` $\leq 1{,}18$	$+0{,}08$
`Curso_area` $\leq -0{,}71$	$-0{,}06$
`Espanhol_1` $\leq -0{,}28$	$-0{,}06$
`Exp_vendas_1` $> -0{,}38$	$-0{,}05$

Tabela 5.6: Regras extraídas via LIME para uma amostra com nota prevista e real igual a 8.

Ao analisar a tabela 5.6, que após a normalização dos dados, representa as contribuições de cada atributo para a nota escolhida, indica que a presença de `Exp_funcao_2` (Experiência como consultor de vendas) e valores intermediários de `Tempo_Vendas` (Tempo

de experiência no setor de vendas) foram os principais fatores responsáveis pela atribuição da nota elevada. Por outro lado, atributos como `Exp_funcao_1` (Experiência como técnico de vendas) e `Espanhol_0` (Ausência de conhecimentos em Espanhol) exerceram influência negativa na predição, sugerindo que determinadas experiências ou formações podem não estar alinhadas com os padrões que o modelo associou a candidatos com melhor desempenho.

Esse tipo de análise é útil para identificar o alinhamento entre o perfil de um candidato e os critérios implícitos aprendidos pelo modelo durante o treinamento. É importante destacar que uma contribuição negativa de determinada variável não implica que o atributo seja indesejável em termos absolutos, mas apenas que, na configuração específica daquela instância, ele reduziu a pontuação atribuída pelo sistema. Tal perspetiva reforça o papel do LIME como ferramenta de apoio à decisão, permitindo que profissionais de RH avaliem as decisões automatizadas com maior grau de compreensão e senso crítico.

5.2.3 TreeInterpreter

Para compreender quais atributos exercem maior influência nas predições do modelo RF, apresenta-se a seguir o resultado do `TreeInterpreter` aplicado a uma amostra do conjunto de teste. Como pode ser visto na equação 5.1, o modelo previu a nota $\hat{y} = 9{,}92$, valor muito próximo da nota real atribuída, que foi 10. Como o modelo aplica um arredondamento ao final do processo de inferência para produzir saídas inteiras, a predição final efetiva foi considerada como 10, caracterizando um acerto na avaliação.

$$\hat{y} = \text{bias} + \sum_{i=1}^{n} \text{contribuição}_i = 5{,}46 + 4{,}462 = 9{,}922 \tag{5.1}$$

A Tabela 5.7 ilustra como o `TreeInterpreter` permite decompor a predição de forma interpretável, indicando o papel de cada variável individual. Nota-se que os maiores impactos positivos vieram de `Tempo_Vendas` (Tempo de experiência no setor de vendas) e `Exp_funcao_2` (Experiência como consultor de vendas), que, em conjunto, representam experiência acumulada diretamente relacionada à área de vendas. A variável `Promocao_1`

(Já obteve promoção no trabalho) também exerceu influência relevante, sinalizando o reconhecimento prévio do candidato no ambiente corporativo. Outros atributos com contribuição positiva incluem `Treinamento` (Quantidade de cursos em vendas) e `Educacao` (Nível educacional), ambos refletindo o fato do candidato possuir uma qualificação adequada para a vaga.

Atributo	**Contribuição**
Tempo_Vendas	+1,864
Exp_funcao_2	+1,244
Promocao_1	+0,317
Treinamento	+0,331
Exp_vendas_1	+0,128
Educacao	+0,253
Curso_area	+0,067
Exp_outros	+0,053
Promocao_2	+0,040
Exp_funcao_3	+0,007
Genero_2	+0,013
Ingles_0	+0,011
Genero_1	+0,006
Ingles_1	+0,008
Espanhol_1	+0,001
Exp_vendas_0	+0,084
Idade	+0,054
Carro_1	+0,001
Carro_0	-0,001
Exp_funcao_0	-0,003
Exp_funcao_1	-0,016
Portugues_1	-0,000
Espanhol_0	+0,000
Soma das contribuições	**+4,462**
Bias (valor base)	**5,460**
Nota Prevista	**9,922**
Nota Real	**10,000**

Tabela 5.7: `TreeInterpreter` - Contribuições locais por atributo para uma amostra.

Embora originalmente concebido para análise local (instância a instância), também foi realizada a aplicação do `TreeInterpreter` de forma agregada sobre múltiplas amostras, a fim de inferir o comportamento global do modelo. Deste modo, a Figura 5.14 apresenta as contribuições médias das variáveis na predição final. As contribuições foram ordenadas pela magnitude (valores absolutos), sendo que barras à direita indicam impacto positivo e barras à esquerda, impacto negativo.

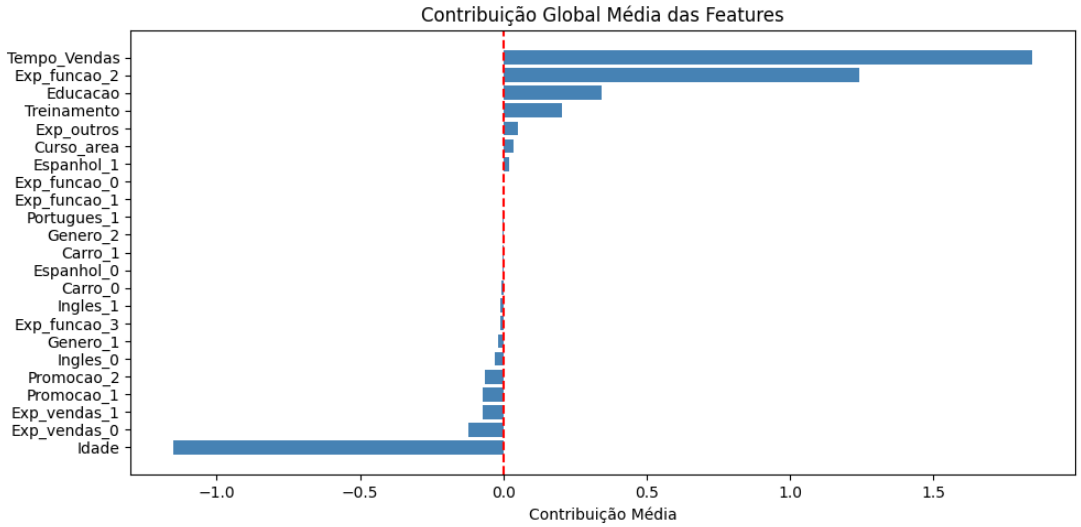

Figura 5.14: TreeInterpreter: Contribuições médias globais dos atributos na predição.

A análise global ilustrada na Figura 5.14, evidencia que o atributo `Tempo_Vendas` (Tempo de experiência no setor de vendas) apresenta a maior contribuição média positiva para as predições do modelo, seguido por `Exp_funcao_2` (Experiência como consultor de vendas), `Educacao` (Nível educacional) e `Treinamento` (Quantidade de cursos em vendas), todos diretamente relacionados à experiência e qualificação do candidato. Esses resultados são consistentes com a lógica esperada em processos de recrutamento, nos quais tempo de experiência e formação técnica tendem a ser valorizados.

Em contrapartida, a variável `Idade` destacou-se com a maior contribuição negativa média, sugerindo que o modelo, em média, associa faixas etárias mais elevadas a notas previstas ligeiramente menores. A Tabela 5.8 apresenta os valores médios das contribuições de cada atributo, possibilitando uma leitura detalhada da influência individual de

cada variável no comportamento global da RF.

Atributo	**Contribuição Média**
Tempo_Vendas	+1,8440
Exp_funcao_2	+1,2420
Educacao	+0,3417
Treinamento	+0,2049
Exp_outros	+0,0518
Curso_area	+0,0337
Espanhol_1	+0,0201
Exp_funcao_0	-0,0002
Exp_funcao_1	-0,0002
Portugues_1	-0,0012
Genero_2	-0,0014
Carro_1	-0,0021
Espanhol_0	-0,0047
Carro_0	-0,0086
Ingles_1	-0,0108
Exp_funcao_3	-0,0114
Genero_1	-0,0174
Ingles_0	-0,0280
Promocao_2	-0,0651
Promocao_1	-0,0707
Exp_vendas_1	-0,0715
Exp_vendas_0	-0,1201
Idade	-1,1492

Tabela 5.8: TreeInterpreter: Contribuições médias globais dos atributos na predição.

5.3 Discussão

Analisando a Tabela 5.9 que compara os melhores modelos de cada arquitetura, é notável que a utilização de dados sintéticos equivalentes à 50% da base de desenvolvimento foi a técnica que obteve melhores resultados em todas as ocasiões.

Dentre os modelos avaliados, o que apresentou melhor desempenho foi o **RF com 1000 árvores**, treinado com 50% de dados sintéticos adicionados à base real. Este modelo alcançou um *MAE* de **0,14**, *MSE* de **0,16**, *RMSE* de **0,40** e coeficiente de determinação R^2 de **0,98**, superando todos os demais algoritmos, totalizando 104 acertos de 120 instâncias (86,67% de acurácia) no conjunto de teste.

Modelo	Base de Treino	MAE	MSE	RMSE	R^2
RF (1000 árvores)	**50% Sintético**	**0,14**	**0,16**	**0,40**	**0,98**
SVM (*Kernel* RBF)	50% Sintético	0,22	0,25	0,50	0,97
MLP (23-12-6-1)	50% Sintético	0,23	0,28	0,53	0,97
KNN Regressor (K = 3)	50% Sintético	0,38	0,52	0,72	0,94

Tabela 5.9: Comparação entre os melhores modelos de cada arquitetura.

Os resultados indicam que o modelo apresenta excelente capacidade de replicar o comportamento do avaliador humano, com a maioria absoluta das predições divergindo no máximo em uma unidade em relação à nota real atribuída. Registra-se apenas um único caso com erro de magnitude 2, o que evidencia a consistência e a precisão do modelo na tarefa proposta.

No entanto, o modelo MLP (23-12-6-1), também treinado com 50% de dados sintéticos, destacou-se pelo seu bom desempenho (*MAE* de **0,23**, *MSE* de **0,28**, *RMSE* de **0,53** e coeficiente de determinação R^2 de **0,97**) com um total de 95 predições corretas dentre os 120 CVs de teste (79,17% de acurácia), e entre os currículos com notas entre 7 e 10 — grupo prioritário na triagem — 81,5% foram acertados exatamente, superando assim o melhor desempenho alcançado no primeiro estudo relacionado (2.6.1), que além de usar MLP também possui uma base de dados com características ligeiramente diferentes, porém com um propósito muito semelhante, sendo destinado à seleção de candidatos para a vaga de técnico de vendas (enquanto o estudo atual se destina à consultor de vendas, uma posição diferente e superior). Ao utilizar uma MLP (28-2-1) treinada com o algoritmo *Bayesian Regularization*, o estudo anterior obteve um MAE de 0,292 e um MSE de 0,375 no conjunto de teste, composto por 120 CVs. Quanto à exatidão, 90 das 120 predições (75%) coincidiram exatamente com a nota atribuída por um especialista em ReS, e entre os currículos prioritários na triagem (notas entre 7 e 10), 72,5% foram acertados exatamente, e os demais ficaram com erro de apenas uma unidade.

No que se refere ao segundo estudo descrito na Seção 2.6.2, observa-se uma divergência metodológica relevante em relação ao presente trabalho. O estudo anterior concentra-se na conversão do conteúdo textual dos currículos em representações numéricas por meio

da técnica TF-IDF, classificando-os posteriormente como “selecionado” ou “rejeitado”. Já a metodologia proposta neste trabalho adota uma abordagem mais orientada ao contexto real de avaliação, partindo da predição de uma nota contínua atribuída ao currículo por avaliadores humanos.

Para efeito de comparação com o cenário binário utilizado na literatura, estabeleceu-se um limiar entre as notas 6 e 7 como critério de decisão. Dessa forma, currículos com nota inferior a 7 foram considerados como “rejeitados”, e aqueles com nota igual ou superior a 7, como “selecionados”. Aplicando essa lógica aos 120 currículos do conjunto de teste, que permaneceram completamente fora do processo de treinamento, os resultados obtidos foram expressivos: o modelo KNN apresentou 10 classificações incorretas, resultando em uma acurácia de 91,66%; o modelo SVM errou apenas 4 instâncias (96,66% de acurácia); o MLP cometeu 2 erros (98,33% de acurácia); e a RF atingiu 100% de acerto no critério de separação entre currículos selecionados e rejeitados.

Esses resultados indicam que, mesmo considerando o cenário binário do estudo anterior, a abordagem adotada neste trabalho demonstrou desempenho superior, com precisão elevada na tomada de decisão classificatória, além de manter a granularidade da predição contínua como vantagem adicional.

Além disso, ao utilizar as técnicas de explicabilidade SHAP, LIME e `TreeInterpreter` no trabalho atual, foi possível decompor e analisar a contribuição individual de cada variável para as predições realizadas pelos modelos. Essas abordagens possibilitam duas formas complementares de interpretação: a explicação local, que esclarece a influência de cada atributo na classificação de um currículo específico, e a explicação global, que permite avaliar a importância média de cada variável ao longo de todas as predições. Dessa forma, essas técnicas promovem maior transparência ao processo decisório automatizado. Além de sua utilidade na validação técnica dos modelos, elas também viabilizam a aplicação prática em ambientes reais, ao possibilitar que profissionais de RH compreendam os fatores que motivaram cada recomendação gerada pelo sistema.

Um aspeto relevante observado nas análises de explicabilidade é que as variáveis consideradas mais importantes diferem entre os modelos utilizados, evidenciando como diferentes algoritmos podem interpretar de maneira distinta a mesma tarefa preditiva. No caso da rede neural MLP, a análise com SHAP revelou que as variáveis mais relevantes, em média, foram `Exp_funcao_3` (experiência como gestor de vendas), `Tempo_Vendas` (tempo de atuação no setor comercial) e `Exp_funcao_0` (ausência de experiência anterior). Por outro lado, na Random Forest, utilizando a técnica `TreeInterpreter`, destacaram-se com impacto positivo `Tempo_Vendas`, `Exp_funcao_2` (Experiência como consultor de vendas) e `Idade` (com impacto negativo), sendo esta última apenas a quinta mais importante segundo o SHAP aplicado à MLP.

É importante destacar que o LIME, ao ser aplicado individualmente a cada amostra, permite variações locais nas variáveis mais influentes: em determinadas instâncias por exemplo, o atributo `Idade` pode exercer maior influência que o `Tempo_Vendas`, e vice-versa. No entanto, o SHAP fornece uma visão global e agregada da importância das variáveis, permitindo observar tendências médias de comportamento do modelo.

Entre todos os métodos e modelos avaliados, a variável `Tempo_Vendas` demonstrou impacto consistente e recorrente, reforçando sua relevância como fator decisivo na seleção de candidatos. Tal resultado é coerente com critérios frequentemente adotados por avaliadores humanos no contexto de recrutamento. Essas informações evidenciam não apenas a capacidade adaptativa das diferentes arquiteturas, mas também a diversidade de perspetivas que cada modelo pode adotar na resolução do mesmo problema.

Outro ponto importante refere-se à origem das notas utilizadas como rótulos. Como o modelo busca emular o comportamento do avaliador humano original, quaisquer vieses presentes nas notas atribuídas manualmente poderão ser aprendidos e replicados pelo sistema. Assim, embora os resultados sejam estatisticamente promissores, é necessária cautela ao interpretar a neutralidade do modelo em termos éticos ou sociais.

Capítulo 6

Conclusões

O presente trabalho investigou a aplicação de modelos supervisionados de aprendizado de máquina no apoio ao processo de recrutamento e seleção de currículos, com ênfase na interpretabilidade das predições geradas. A proposta envolveu a predição automatizada das notas atribuídas por um avaliador humano, utilizando um conjunto de dados reais rotulado manualmente. A abordagem adotada compreendeu diversas etapas, incluindo o pré-processamento detalhado das variáveis, normalização, codificação categórica, geração de dados sintéticos e aplicação de técnicas de explicabilidade.

Os resultados reforçam que todos os modelos supervisionados testados foram capazes de realizar com competência a tarefa de predição, ainda que com níveis de precisão distintos. A análise revelou, por exemplo, que experiências excessivas em cargos superiores podem impactar negativamente as predições para uma vaga inferior, não por inadequação profissional, mas por desalinhamento com o escopo do cargo em questão. Essa distinção é fundamental para evitar julgamentos enviesados na interpretação dos resultados automatizados.

Como caminhos para trabalhos futuros, destaca-se a automatização da etapa de extração das variáveis dos currículos, viabilizando a ampliação da base por meio de técnicas de PLN e modelos generativos baseados em LLMs. Deste modo, seria possível realizar a integração entre a captação dos arquivos de CV, extração das características e classificação junto da explicabilidade a respeito da atribuição da nota para o candidato. Isso

permitiria a criação de bases para múltiplos cargos de forma escalável. Outro desenvolvimento promissor é a construção de uma plataforma interativa onde os currículos possam ser avaliados automaticamente, com interface explicativa para os candidatos e avaliadores humanos, possibilitando aprendizado contínuo do modelo com base em novas avaliações.

Conclui-se, portanto, que a aplicação de modelos supervisionados explicáveis ao contexto de triagem curricular representa uma contribuição técnica relevante e viável para o apoio à decisão em RH. Ao aliar desempenho preditivo elevado com mecanismos de interpretação, o sistema proposto torna-se não apenas funcional, mas também confiável e auditável — aspetos essenciais para sua adoção em ambientes sensíveis como o recrutamento de pessoas.

Bibliografia

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. Et al. (2016). TensorFlow: A system for large-scale machine learning. *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 265–283.

Ali, P. J. M. & Faraj, R. H. (2014). *Data Normalization and Standardization: A Technical Report* (rel. téc.). Machine Learning Lab, Koya University. https://www.researchgate.net/publication/340579135_Data_Normalization_and_Standardization_A_Technical_Report

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, *46*(3), 175–185.

Arlot, S. & Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statistics surveys*, *4*, 40–79.

Beyer, K., Goldstein, J., Ramakrishnan, R. & Shaft, U. (1999). When is "nearest neighbor"meaningful?, Em *International conference on database theory*, Springer.

Biau, G. & Scornet, E. (2016). A random forest guided tour. *Test*, *25*(2), 197–227.

Blumen, D. & Cepellos, V. M. (2023). Dimensões do uso de tecnologia e Inteligência Artificial (IA) em Recrutamento e Seleção (R&S): benefícios, tendências e resistências. *Cadernos EBAPE.BR*, *21*(2), e2022–0080. https://doi.org/10.1590/1679-395120220080

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32. https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf

Chollet, F. Et al. (2015). Keras [Acesso em: 2025-05-27].

Christoph Molnar. (2023). Interpretable Machine Learning [Acesso em: junho de 2025].

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273–297.

Foundation, P. S. (2001). Python Language Reference, version 3.x. https://www.python.org/

Gokte, S. A. (2020). Most Popular Distance Metrics Used in KNN and When to Use Them [Acesso em: 2025-05-27].

Goldbloom, A. (2010). Kaggle [Acesso em: 12 jun. 2025]. https://www.kaggle.com/

Google. (2017). Google Colaboratory [Acesso em: junho de 2025].

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J. Et al. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hofmann, T., Schölkopf, B. & Smola, A. J. (2008). Kernel methods in machine learning. *The Annals of Statistics*, *36*(3), 1171–1220.

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, *9*(3), 90–95.

Hunter, J. D. & Contributors. (2003). Matplotlib: Visualization with Python. https://matplotlib.org/

IBM Cloud Education. (2024). What is Supervised Learning? [Acesso em: 25 maio 2025].

Jatobá, M. N. (2020). *Inteligência artificial no recrutamento & seleção: inovação e seus impactos para a gestão de recursos humanos* (tese de mestrado) [Orientadores: Paula O. Fernandes, João Paulo Teixeira, Daniela Moscon]. APNOR e UNIFACS - Universidade Salvador. Orientadores: Paula O. Fernandes, João Paulo Teixeira, Daniela Moscon. http://hdl.handle.net/10198/21805

Jatobá, M., Ferreira, J., Fernandes, P. & Teixeira, J. (2023). Intelligent human resources for the adoption of artificial intelligence: a systematic literature review. *Journal of Organizational Change Management*, *36*(7), 1099–1124. https://doi.org/10.1108/JOCM-03-2022-0075

Jupyter, P. (2014). Jupyter Notebook [Acesso em: junho de 2025].

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

Li, D., Raymond, L. R. & Bergman, P. (2020). *Hiring as exploration* (rel. téc. w27736). National Bureau of Economic Research. https://doi.org/10.2139/ssrn.3630630

Lo, F. P. F., Qiu, J., Wang, Z., Yu, H., Chen, Y., Zhang, G. & Lo, B. (2025). AI Hiring with LLMs: A Context-Aware and Explainable Multi-Agent Framework for Resume Screening. *arXiv preprint arXiv:2504.02870*. https://arxiv.org/abs/2504.02870

Lundberg, S. M. & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems (NeurIPS)*, *30*, 4765–4774.

Lundberg, S. & contributors, S. (2024). SHAP (SHapley Additive exPlanations). https://shap.readthedocs.io/

Luo, X., Sun, F., Ni, X., Lin, Z., Hu, X. & Liu, S. (2018). ResumeNet: A Learning-based Framework for Automatic Resume Quality Assessment, Em *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, AAAI Press.

Mariani, G., Scheidegger, F., Istrate, R., Albarqouni, S. & Navab, N. (2018). BAGAN: Data Augmentation with Generative Adversarial Networks for Imbalanced Datasets, Em *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*. https://doi.org/10.1109/WACV.2018.00036

McClelland, J. L., Rumelhart, D. E. & the PDP Research Group. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition. Volume 1: Foundations*. MIT Press.

McKinney, W. (2010). Data Structures for Statistical Computing in Python (S. van der Walt & J. Millman, Eds.). Em S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference*. SciPy.

Microsoft. (2015). Visual Studio Code [Acesso em: junho de 2025].

Moriarty, D. E. & Miikkulainen, R. (1998). Hierarchical evolution of neural networks, Em *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360)*. IEEE.

Neto, R., Jatobá, M. N., Santana, M., Fernandes, P. O., Ferreira, J. J., Foleis, J. H. & Teixeira, J. P. (2025). Human Resources Optimization with MultiLayer Perceptron: An Automated Selection Tool [CENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies]. *Procedia Computer Science*, *256*, 238–245. https://doi.org/https://doi.org/10.1016/j.procs.2025.02.117

NumPy. (2005). NumPy: the fundamental package for scientific computing with Python. https://numpy.org/

Otten, N. V. (2024). Support Vector Machines (SVM) In Machine Learning Made Simple & How To Tutorial [Acesso em: 2025-05-27].

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Rajesh, D. S., Kandaswamy, M. U. & Rakesh, M. A. (2018). The impact of artificial intelligence in talent acquisition lifecycle of organizations. *International Journal of Engineering Development and Research*, *6*(2), 709–717.

Ribeiro, M. T. & contributors, L. (2016). LIME: Local Interpretable Model-agnostic Explanations. https://lime-ml.readthedocs.io/

Ribeiro, M. T., Singh, S. & Guestrin, C. (2016). Why Should I Trust You?: Explaining the Predictions of Any Classifier, Em *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM.

Roy, S., Saha, S., Ghosh, A. & Chakraborty, D. (2024). A distance-based kernel for classification via Support Vector Machines. *Frontiers in Artificial Intelligence*, *7*, 1287875. https://doi.org/10.3389/frai.2024.1287875

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://www.cs.toronto.edu/~hinton/absps/naturebp.pdf

Rumelhart, D. E., McClelland, J. L. & the PDP Research Group. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition. Volume 2: Psychological and biological models*. MIT Press.

Saabas, A. (2017). TreeInterpreter: Interpreting scikit-learn decision tree predictions. https://github.com/andosa/treeinterpreter

Scikit-learn. (2010). scikit-learn: Machine Learning in Python. https://scikit-learn.org/

scikit-learn. (2010). 1.4. Support Vector Machines [Acesso em: 4 jun. 2025].

scikit-learn. (2011). StratifiedKFold [Acesso em: 4 jun. 2025].

Soleimani, M., Intezari, A. & Pauleen, D. J. (2022). Mitigating cognitive biases in developing AI-assisted recruitment systems: A knowledge-sharing approach. *International Journal of Knowledge Management (IJKM)*, *18*(1), 1–18. https://doi.org/10.4018/IJKM.290022

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of machine learning research*, *15*(1), 1929–1958.

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C. & Liu, C. (2018). A Survey on Deep Transfer Learning. *CoRR*, *abs/1808.01974*arXiv 1808.01974. http://arxiv.org/abs/1808.01974

Tayal, D., Jain, R. & Arora, A. (2024). An Efficient Resume Ranking and Candidate Selection System Using NLP and Machine Learning Techniques, Em *Proceedings of the 5th International Conference on Advances in Computing, Communication and Control (ICAC3)*, IEEE. https://doi.org/10.1109/ICAC39262.2024.1234567

Team, K. (2015). Keras: the Python deep learning API. https://keras.io/

Team, T. (2015). TensorFlow: An end-to-end open source machine learning platform. https://www.tensorflow.org/

Team, T. P. D. (2008). pandas: powerful Python data analysis toolkit. https://pandas.pydata.org/

Velazco, M. I. & Lyra, C. (2002). Optimization with neural networks trained by evolutionary algorithms, Em *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02*. https://doi.org/10.1109/IJCNN.2002.1007742

Waskom, M. L. (2021). Seaborn: statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021

Waskom, M. & contributors. (2012). Seaborn: Statistical Data Visualization. https://seaborn.pydata.org/

Will, P., Krpan, D. & Lordan, G. (2023). People versus machines: introducing the HIRE framework. *Artificial Intelligence Review*, *56*(2), 1071–1100. https://doi.org/10.1007/s10462-022-10193-6

Wu, J. Et al. (2021). Direction Matters: On the Implicit Bias of Stochastic Gradient Descent with Moderate Learning Rate [Available at https://arxiv.org/abs/2011.02538], arXiv 2011.02538.

Apêndice A

Proposta Original do Projeto

Proposta de tema para
Dissertação/Estágio/Projeto - Trabalho de Conclusão de Curso

Orientador da Instituição onde se realiza o trabalho:

João Paulo Teixeira	joaopt@ipb.pt

Instituição do orientador:

Instituto Politécnico de Bragança	Escola Superior de Tecnologia e Gestão

Co-orientador da Instituição parceira:

Juliano Foleis	julianofoleiss@gmail.com

Instituição do co-orientador:

Universidade Tecnológica Federal do Paraná	Campus de Campo Mourão

Curso ou cursos da Instituição do orientador onde se propõe que o trabalho seja realizado:

Mestrado em Informática

Aluno: Reginaldo Gregório de Souza Neto (a61482@alunos.ipb.pt)

Título do trabalho:

Efficient and Intelligent Selection of Human Resources - Decision Support System

Palavras chave:

Inteligência Artificial, Sistemas de Classificação Inteligentes, Seleção de Recursos Humanos

Objetivos:

Elaborar um Sistema de apoio à tomada de decisão na seleção de Currículos para uma posição específica, utilizando Modelos de Inteligência Artificial, como redes neurais artificiais (MLP) e, possivelmente, redes convolucionais (Deep Learning - CNN).
Espera-se que o aluno conclua o trabalho com uma forte conhecimento teórico e prático sobre a utilização de modelos de IA, nomeadamente com um domínio abrangente em sistema de machine learning.

Descrição adicional:

O aluno será integrado num ambiente de investigação com investigadores experientes na área de recursos humanos e na área de inteligência artificial nos centros de investigação CeDRI e UNIAG, do IPB. Terá disponível um dataset com dados retirados de uma base de dados de CV de candidatos ao cargo de consultor de vendas. Pretende-se que o aluno explore e experimente uma variedade de modelos de machine learning para classificação dos CVs, entre eles redes neuronais MLP, redes convolucionais, a agregação de sistemas usando técnicas de Bootstrapping ou bagging, entre outras. Deverá aplicar técnicas de normalização, tratamento de outliers, feature selection e Análise de Componentes Principais (PCA).
Para a análise dos resultados deverá explorar métricas como a matriz de confusão, curvas ROC e Scatter Plots.
No final do trabalho espera-se que o aluno tenho um domínio abrangente sobre sistemas de IA.

Metodologia/Plano de trabalhos:

1. Estudo do estado da arte sobre machine learning (2 month)
2. Tomada de conhecimento dos objetivos do trabalho pelo contacto com o dataset disponível e dos objetivos do trabalho. (1 month)
3. Utilização de técnicas de normalização, tratameto ed outliers, PCA, e feature selection. (2 months)
4. Desenvolvimento de diversos modelos de machine learning para classificação dos CV. (3 months)
5. Aplicação de diversas métricas para avaliação dos resultados, como exatidão, precisão, matriz de confusão, curvas ROC e Scatter Plots. (1 month)
6. Escrita da dissertação. (1 month)

Recursos necessários:

Computador
Matlab ou Python
Dataset
(todos disponíveis para o trabalho)