PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS

ESCOLA POLITÉCNICA E DE ARTES

GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

ALGORITMO DO VIZINHO MAIS PRÓXIMO PARA PREDIÇÃO DE DOENÇAS

DANIEL PIRES TORRES

GOIÂNIA

2024

DANIEL PIRES TORRES

ALGORITMO DO VIZINHO MAIS PRÓXIMO PARA PREDIÇÃO DE DOENÇAS

Trabalho de conclusão de curso apresentado à Escola politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos necessários para a obtenção do título de Bacharel em Engenharia de Computação.
Orientadora:
Prof.ª Dr.ª Solange da Silva.

Banca examinadora:

Prof. Dr. Clarimar José Coelho

Prof. Dr. Nilson Cardoso Amaral

GOIÂNIA

2024

DEDICATÓRIA

Quero dedicar este trabalho de conclusão de curso a todas as pessoas que depositaram sua confiança em mim e me ofereceram apoio ao longo desta jornada acadêmica. Em especial, expresso minha gratidão aos meus familiares e amigos, cujo constante suporte emocional e incentivo foram fundamentais.

Também desejo dedicar este trabalho aos meus estimados professores e orientadores, cuja orientação e ensinamentos foram inestimáveis ao longo desses anos de estudo. Seus conhecimentos, experiências e conselhos foram indispensáveis para o êxito deste trabalho.

Por último, dedico este trabalho à minha própria trajetória de aprendizado e desenvolvimento pessoal. Agradeço por todas as lições e desafios que encontrei ao longo do percurso, pois foram eles que me impulsionaram até este ponto.

AGRADECIMENTOS

Gostaria de agradecer primeiramente a Deus, por sempre me conceder força e sabedoria para a conclusão do curso e deste trabalho.

Expresso meus sinceros agradecimentos a todas as pessoas que estiveram ao meu lado durante esta jornada acadêmica, apoiando-me e incentivando-me a superar desafios e alcançar meus objetivos

Agradeço à minha família, pelo amor incondicional, paciência e apoio constante ao longo desses anos de estudo. Vocês foram meu alicerce, sempre me encorajando a perseguir meus sonhos e acreditando em mim.

À minha orientadora, agradeço a orientação e dedicação durante todo o desenvolvimento deste trabalho de conclusão de curso. Seu conhecimento, paciência e incentivo foram fundamentais para o meu crescimento acadêmico. Sou grato pela oportunidade de aprender com você e pela confiança que depositou em mim.

Aos meus professores, agradeço pelos ensinamentos transmitidos ao longo de toda a graduação. Suas aulas, discussões e orientações foram essenciais para a minha formação acadêmica. Agradeço pela generosidade em compartilhar seu conhecimento e por despertarem em mim o interesse pelo aprendizado contínuo.

Aos meus colegas de curso, agradeço a parceria, apoio mútuo e momentos compartilhados ao longo dessa jornada. Juntos, enfrentamos desafios, superamos obstáculos e celebramos conquistas. Sou grato por cada amizade construída e por todo o apoio recebido.

Por fim, expresso minha gratidão a todos os demais familiares, amigos e pessoas que de alguma forma contribuíram para o meu sucesso acadêmico. Seu apoio, incentivo e palavras de encorajamento foram de grande importância para que eu chegasse até aqui.

RESUMO

O objetivo geral deste trabalho foi avaliar a eficiência do modelo do Algoritmo do Vizinho Mais Próximo ou *K-Nearest Neighbor* (KNN) para a predição de doenças, realizando testes do modelo para detecção precoce de diabetes, utilizando uma abordagem que combina revisão bibliográfica e pesquisa experimental. Durante o estudo, observou-se que o modelo KNN, aplicado a uma base de dados composta por variáveis categóricas e numéricas obtidas de pacientes do *Sylhet Diabetes Hospital em Bangladesh*, apresentou em seus resultados um desempenho com acurácia de até 72,4%, considerando o cenário onde houve um pré-processamento com escolhas de variáveis que desenvolvam resultados mais precisos, como a idade do paciente. As etapas de pré-processamento incluíram a normalização dos dados, tratamento de valores ausentes e balanceamento das classes, fatores cruciais para a obtenção de melhores resultados. Além disso, a escolha adequada do hiperparâmetro “K” revelou-se determinante para o equilíbrio entre precisão e estabilidade do modelo, sendo testada com diferentes valores para identificar o mais apropriado para o conjunto de dados utilizado, no caso da aplicação deste trabalho o valor de igual 3 apresentou os melhores resultados. Para avaliar a eficácia do modelo foram utilizadas métricas como acurácia, matriz de confusão e precisão. O estudo permitiu concluir que o modelo KNN é uma abordagem eficiente para a predição de doenças, desde que os dados sejam pré-processados adequadamente e o parâmetro K seja escolhido com critério. Apesar de sua simplicidade e versatilidade, o KNN enfrenta limitações, como sensibilidade à dimensionalidade dos dados e ao desbalanceamento de classes. Técnicas como normalização e seleção de características ajudam a mitigar esses problemas, mas o cálculo de distâncias em grandes volumes de dados ainda representa um desafio em contextos complexos, ressaltando a necessidade de avaliar cuidadosamente sua aplicação em problemas clínicos.

Palavras chaves: Inteligência Artificial. Predição de doenças. *Machine Learning*. Regressão Linear. KNN.

ABSTRACT

The general objective of this study was to evaluate the efficiency of the K-Nearest Neighbor (KNN) algorithm for disease prediction, conducting tests for early diabetes detection using an approach that combines a literature review and experimental research. During the study, it was observed that the KNN model, applied to a dataset composed of categorical and numerical variables obtained from patients at the Sylhet Diabetes Hospital in Bangladesh, achieved a performance accuracy of up to 72.4%. This result was based on scenarios where preprocessing included variable selection aimed at improving accuracy, such as the patient's age. The preprocessing stages involved data normalization, handling missing values, and class balancing, which were crucial for achieving better outcomes. Additionally, the appropriate choice of the "K" hyperparameter was critical for balancing model accuracy and stability, with different values tested to identify the most suitable one for the dataset used. In this study, a K value of 3 yielded the best results. Metrics such as accuracy, confusion matrix, and precision were used to evaluate the model's effectiveness. The study concluded that the KNN model is an efficient approach for disease prediction, provided the data is properly preprocessed and the K parameter is carefully selected. Despite its simplicity and versatility, KNN faces limitations, such as sensitivity to data dimensionality and class imbalance. Techniques like normalization and feature selection help mitigate these issues, but distance calculation in large datasets remains a challenge in complex contexts, highlighting the need for careful evaluation of its application in clinical problems.

Keywords: Artificial Intelligence. Disease Prediction. Machine Learning. Linear Regression. KNN.

LISTA DE ILUSTRAÇÕES

Figura 1 – Gráfico de Regressão Linear Simples. 16

Figura 2 – Gráfico de Regressão Linear Múltipla. 16

Figura 3 – Exemplo de aplicação KNN com valor de K entre 3 e 6. 18

Figura 4 – Ilustração da distância entre dois pontos. 18

Figura 5 – Valor de K = 1 resultando em classificação verde. 42

Figura 6 – Valor de k=2, resultando em classificação verde. 42

Figura 7 – Valor de k=3, resultando em classificação verde. 43

Figura 8 – Valor de k=4, resultando em empate. 43

Figura 9 – Valor de k=5, resultando em classificação vermelha. 44

Figura 10 – Importação das bibliotecas para desenvolvimento do modelo. 52

Figura 11 – Configurações de conexão com o banco de dados. 53

Figura 12 – Consulta SQL para importar os dados do banco de dados. 54

Figura 13 – Carregamento em um DataFrame e conversão das colunas para minúsculo. 55

Figura 14 – Alteração das colunas para boolean. 55

Figura 15 – Processamento dos dados. 56

Figura 16 – Pré-processamento e tratamento de valores nulos. 57

Figura 17 – Separação dos dados em classes e filtro de classes positivas. 58

Figura 18 – Separação das colunas e pipeline para o pré-processamento. 58

Figura 19 – Separação dos dados para treino e teste. 59

Figura 20 – Treinamento do modelo KNN. 59

Figura 21 – Previsão no conjunto de teste e avaliação do modelo. 60

Figura 22 – Avaliação da eficácia do modelo. 61

Figura 23 – Implementação do gráfico de dispersão. 62

Figura 24 – Gráfico de dispersão dos resultados. 63

LISTA DE SIGLAS

API	*Application Programming Interface*
FDA	*Food and Drug Administration*
GDPR	*General Data Protection Regulation*
IDE	*Integrated Development Environment*
IA	Inteligência Artificial
KNN	*K-Nearest Neighbors*
MAE	*Mean Absolute Error*
ML	*Machine Learning*
MSE	*Mean Squared Error*
OMS	Organização Mundial da Saúde
PCA	*Principal Component Analysis*
RMSE	*Root Mean Squared Error*
SLR	*Systematic Literature Review*
SQL	*Structured Query Language*
SVM	*Support Vector Machines*
TCC	Trabalho de Conclusão de Curso

LISTA DE QUADROS

Quadro 1 – Exemplo de matriz de confusão para classes positiva e negativa 46

Quadro 2 - Variáveis da base de dados “Previsão de risco de diabetes em estágio inicial” 51

Sumário

1 INTRODUÇÃO 12
2 REFERENCIAL TEÓRICO 14
2.1 Conceitos e definições 14
2.1.1 Machine Learning 15
2.1.2 Regressão Linear 16
2.1.3 Algoritmo *K-Nearest Neighbors* (KNN) 17
2.1.4 A linguagem Python e suas aplicações de IA 19
2.2 Trabalhos relacionados 21
2.2.1 Comparação e Seleção de Algoritmos *de Machine Learning* para Previsão de Diabetes: Um Estudo Quantitativo Exploratório Baseado em Análise de Dados Médicos 21
2.2.2 Predição de diabetes tipo 2 usando algoritmo de vizinho mais próximo K 22
2.2.3 A importância do papel dos médicos diante a utilização de IA para diagnósticos e questões de segurança 26
2.2.3.2 O papel da inteligência artificial na saúde: uma revisão estruturada da literatura 28
3 MÉTODO 31
4 Implementação do algoritmo do modelo KNN 34
4.1 Como o modelo KNN é desenvolvido e aplicado com a linguagem Python 34
4.1.1 Análise e Armazenamento dos Dados 35
4.1.2 Definição das Métricas de Distância 37
4.1.3 Definição do Valor de K 39
4.1.4 Classificação ou Regressão dos dados 39
4.1.5 Análise de resultados 44
4.1.6 Biblioteca Python S*klearn* para desenvolver o modelo KNN 48
4.2 Desenvolvendo o modelo KNN para predição de diabetes 50
4.2.1 Base de dados utilizadas 50
4.2.2 Desenvolvimento do modelo KNN utilizando Python 51
5 Análise dos resultados obtidos 62
5.1 Análise dos resultados do modelo KNN desenvolvido 64
5.1.4 Conclusão sobre a análise dos resultados 71
6 CONCLUSÃO 77
REFERENCIAS 79

1 INTRODUÇÃO

As doenças cardiometabólicas, como hipertensão, diabetes, dislipidemia e obesidade, são grandes causas de mortalidade global e estão ligadas a fatores de risco modificáveis, como hipertensão arterial, tabagismo, glicemia elevada, inatividade física e obesidade, que frequentemente se combinam, agravando os efeitos na saúde. Com o envelhecimento populacional, desigualdades no acesso à saúde e escassez de profissionais, é relevante o uso de ferramentas tecnológicas preditivas para mapear e reduzir riscos associados ao estilo de vida, promovendo mudanças preventivas e melhorias na saúde futura (Lopes et al., 2021).

A Inteligência Artificial (IA) representa um campo da ciência da computação dedicado ao desenvolvimento de sistemas e algoritmos capazes de executar tarefas que tradicionalmente demandam inteligência humana. Essas atividades englobam aprendizado, raciocínio e percepção (Spadini, 2023).

O aprendizado de máquina, ou *Machine Learning* (ML), é uma área da ciência da computação que combina técnicas matemáticas, estatísticas e algoritmos para identificar padrões e realizar previsões, sendo amplamente aplicado na medicina e no contexto de B*ig Data* (grandes dados), que exige métodos avançados para processar grandes volumes de dados variados. O desenvolvimento de algoritmos de ML envolve três etapas: pré-processamento, treinamento (supervisionado ou não) e avaliação, garantindo precisão e confiabilidade na classificação de novos dados. Para resultados eficazes, é essencial trabalhar com bases de dados validadas, evitando erros e permitindo ao modelo generalizar e prever com sucesso (Santos, 2024).

A Regressão Linear é uma técnica estatística usada para modelar a relação entre uma variável dependente e uma ou mais variáveis independentes, com a premissa de que essa relação seja linear. Ela é amplamente utilizada em várias áreas, como na economia, para entender a correlação entre variáveis como renda e consumo, ajudando a prever comportamentos de consumo com base nas variações de renda. Na saúde, é empregada para investigar a relação entre os resultados de tratamentos médicos e fatores de risco, permitindo identificar quais fatores influenciam os resultados de saúde. A técnica é particularmente útil para prever o

comportamento de uma variável dependente com base em variáveis independentes e para compreender melhor as relações entre essas variáveis (Erden, 2023).

O *K-Nearest Neighbors* (KNN) é um algoritmo de aprendizado de máquina utilizado para classificação e regressão. Trata-se de um método não paramétrico, métodos que não assumem uma relação fixa entre variáveis, mas emprega métricas de distância para identificar os “K” pontos de dados mais próximos no conjunto de treinamento em relação a cada ponto de dados do conjunto de teste. A previsão para o ponto de teste é obtida por meio de uma votação majoritária ou da média ponderada dos valores dos “K” vizinhos mais próximos (Erden, 2023).

É relevante estudar este tema, devido ao uso crescente da IA no campo da medicina, no caso da aplicação de Regressão Linear, que pode ser uma técnica que auxilie na prevenção de doenças, pelas análises estatísticas. Durante a condução de estudos clínicos para investigar fatores associados a doenças ou tratamentos, com o objetivo de melhorar o atendimento ao paciente e a prática clínica, a avaliação estatística dos dados é frequentemente essencial (Bzovsky et al., 2022).

Diante deste contexto, este projeto visa responder a seguinte questão de pesquisa: - **Como o uso do Algoritmo do Vizinho Mais Próximo ou *K-Nearest Neighbors* (KNN) pode ser eficiente na predição de doenças?**

O objetivo geral deste trabalho é fazer uma revisão bibliográfica e pesquisa experimental para analisar a eficiência que o modelo do Algoritmo do Vizinho Mais Próximo ou *K-Nearest Neighbors* (KNN) pode apresentar para predição de possíveis doenças.

Os objetivos específicos são:

- Estudar o funcionamento do modelo KNN;
- Realizar a implementação do modelo KNN utilizando a linguagem Python;
- Analisar os desafios e as limitações da implementação;
- Avaliar o impacto da aplicação nos dados fornecidos;
- Avaliar a eficácia do algoritmo.

Espera-se que os resultados deste estudo possam contribuir:

- Mostrando o potencial do modelo KNN para a prevenção de doenças;

- Orientando a adoção e a implementação eficaz do algoritmo em ambientes onde serão importantes a predição de doenças;
- Propondo recomendações do método nos processos de detecção de doenças.

Quantos aos aspectos metodológicos, a natureza desta pesquisa é um resumo de assunto. Quanto aos seus objetivos é uma pesquisa exploratória e descritiva. Em relação aos procedimentos técnicos, é uma pesquisa bibliográfica e experimental.

Esta monografia está estruturada da seguinte maneira: neste Capítulo é apresentado o contexto do trabalho, a questão de pesquisa, objetivo e resultados esperados. O Capítulo 2 traz o referencial teórico com conceitos, definições, trabalhos relacionados com o tema. No Capítulo 3 é descrito o método, mostrando como ele é aplicado no contexto da pesquisa. No Capítulo 4 é apresentado e contextualizado as técnicas aplicadas. No Capítulo 5 é apresentado a análise dos resultados obtidos. Finalmente, o Capítulo 6 traz as considerações finais do TCC e sugestões para trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo é composto por duas partes: uma de conceitos e definições e outra de trabalhos relacionados.

2.1 Conceitos e definições

O conceito de IA está fundamentado na criação de sistemas que imitam o comportamento humano através de máquinas, utilizando algoritmos complexos para aprender e tomar decisões com base nos dados fornecidos. Essa área pode ser subdividida em *Machine Learning* e *Deep Learning*. O *Machine Learning* envolve o aprendizado contínuo da máquina, enquanto o *Deep Learning* capacita a máquina a lidar com tarefas mais complexas, como reconhecimento de padrões em dados como fala e imagens. Esses sistemas têm a capacidade de analisar uma grande quantidade

de dados em pouco tempo, tornando-os eficientes em comparação com a intervenção humana (Damasceno e Vasconcelos, 2018).

A relação entre *Deep Learning* e *Machine Learning* é complementar, com o primeiro fornecendo dados para o segundo, substituindo assim a intervenção humana. Embora a IA possa apresentar margens de erro, suas capacidades de processamento rápido e análise de grandes volumes de dados em tempo mínimo tornam-na altamente eficaz. O verdadeiro objetivo da IA é executar tarefas de forma inteligente, não necessariamente imitando o comportamento humano, mas realizando ações de forma disciplinada e com um certo grau de autonomia, o que a torna uma ferramenta valiosa em diversas áreas (Damasceno e Vasconcelos, 2018).

2.1.1 Machine Learning

Segundo Costa (2024), *Machine Learning* (ML) é um ramo da IA que se concentra em criar sistemas capazes de aprender a partir de dados. Em vez de serem programados com regras específicas, esses sistemas são treinados usando dados e algoritmos, permitindo que melhorem seu desempenho ao longo do tempo. Isso significa que os computadores podem aprender com os dados sem a necessidade de programação explícita para cada tarefa. O ML tem uma variedade de aplicações, como previsão do tempo, detecção de fraudes, recomendação de produtos e diagnóstico médico, aproveitando a grande quantidade de dados disponíveis atualmente, que podem ser de diferentes tipos, como números, textos, imagens, vídeos e áudios.

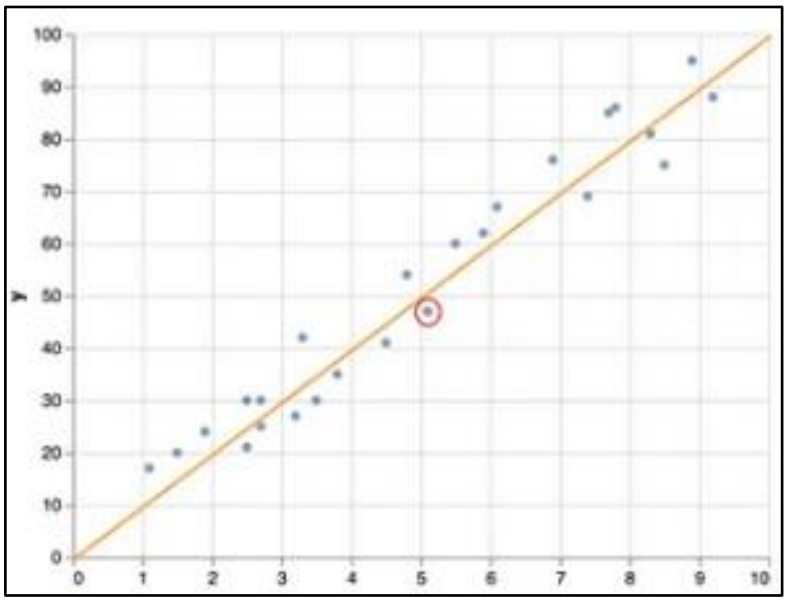
O funcionamento do ML envolve o treinamento de um modelo para detectar padrões em um conjunto de dados. Inicialmente, os dados são enviados a um algoritmo de ML, que treina o modelo para identificar esses padrões. Após o treinamento, o modelo pode ser testado, avaliado, otimizado e colocado em produção, dependendo da aplicação desejada. O resultado do modelo depende do tipo de algoritmo utilizado: algoritmos de classificação retornam à categoria prevista para uma amostra, algoritmos de regressão preveem valores numéricos e algoritmos de agrupamento agrupam amostras semelhantes.

2.1.2 Regressão Linear

Conforme diz Damasceno (2020), a regressão linear é um algoritmo supervisionado de *Machine Learning* utilizado para estimar o valor de uma variável com base em dados históricos, ou seja, ao analisar o passado, é possível "prever" o futuro. Existem dois tipos de regressão linear: a simples e a múltipla.

- **Regressão Linear Simples:** mostrada na Figura 1, ocorre quando utilizamos apenas uma variável independente (X) para fazer a predição.

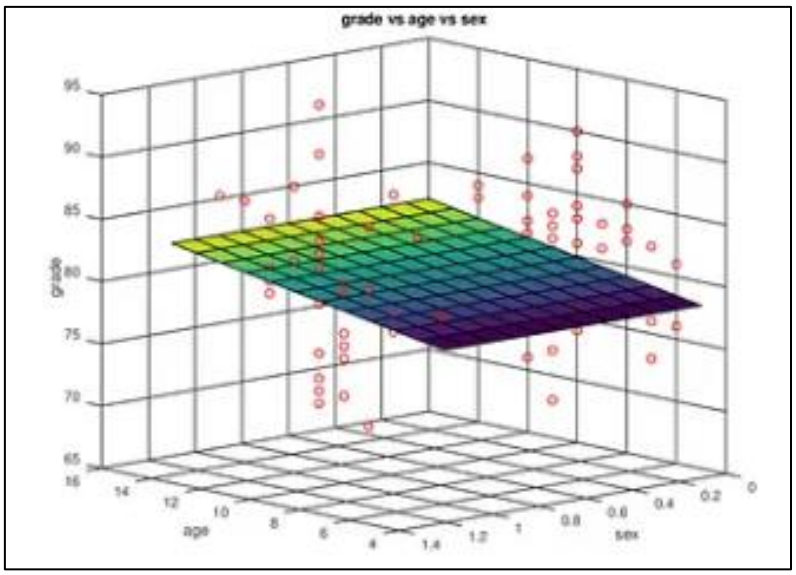
Figura 1 – Gráfico de Regressão Linear Simples



Fonte (Damasceno, 2020)

- **Regressão Linear Múltipla:** ilustrada na Figura 2, envolve o uso de várias variáveis independentes (X) para realizar a predição.

Figura 2 – Gráfico de Regressão Linear



Fonte (Damasceno, 2020)

A forma de representação gráfica dessas regressões varia conforme o tipo. Na regressão linear simples, a representação gráfica é uma linha reta em um plano bidimensional. Já na regressão linear múltipla, a representação ocorre em um espaço que pode ter várias dimensões (*n*D). Esse algoritmo pode ser aplicado em qualquer problema em que as variáveis de entrada e saída sejam valores contínuos. Alguns exemplos incluem:

- Previsão das vendas de um determinado produto;
- Avaliação do valor de um imóvel no setor imobiliário;
- Estimativa da expectativa de vida de um país;
- Cálculo da pressão sanguínea de um paciente (Damasceno, 2020).

2.1.3 Algoritmo *K-Nearest Neighbors* (KNN)

Segundo Matos (2023), o algoritmo KNN, ou "K-vizinhos mais próximos", é um algoritmo supervisionado de aprendizado de máquina que se destacou pela sua simplicidade e facilidade de implementação, especialmente com o suporte de bibliotecas especializadas em inteligência artificial.

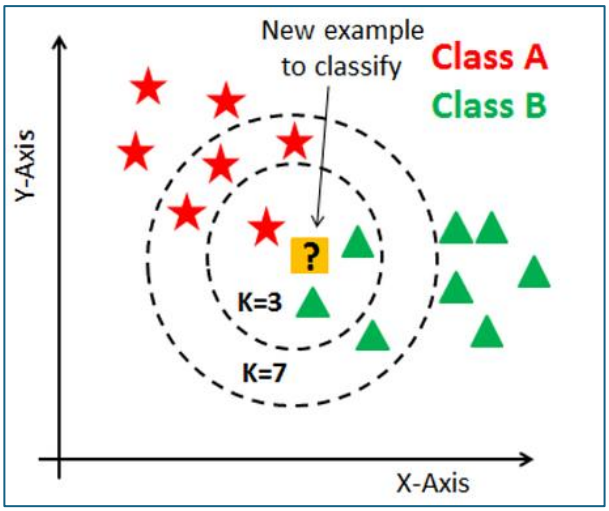
Como o nome indica, trata-se de um algoritmo de classificação ou regressão baseado na proximidade entre "vizinhos". Ao inserir um novo ponto em um conjunto de dados, calcula-se a distância entre esse ponto e os K pontos mais próximos, a fim de determinar a qual grupo o novo ponto pertence.

Esse conceito surgiu no artigo “*Discriminatory Analysis, Nonparametric Discrimination: Consistency Properties*.”, de 1951, de Evelyn Fix e Joseph Hodges. Embora o artigo não tenha introduzido a técnica como a conhecemos hoje, ele lançou as bases para seu desenvolvimento. Em 1967, o artigo “*Nearest Neighbor Pattern Classification*”, de Thomas Cover e Peter E. Hart, explorou o conceito em detalhes, abordando questões práticas, como a escolha do número de vizinhos (parâmetro "K") e a definição de métricas de distância para medir a proximidade entre os pontos.

O cerne do algoritmo reside na determinação do número de pontos (vizinhos) a ser utilizado para fazer a classificação ou regressão de uma nova entrada, o que torna o valor "K o hiperparâmetro mais relevante do algoritmo.

Esse valor pode ser abordado de duas maneiras: com um valor de "K" pequeno, o que permite uma interpretação mais detalhada dos padrões, mas tornando o algoritmo mais suscetível a ruídos e outliers; ou com um "K" maior, o que proporciona maior robustez e estabilidade na interpretação dos dados, embora possa acarretar perda de informações. A Figura 3 ilustra com precisão a influência do valor de K para o resultado das previsões do modelo.

Figura 3 – Exemplo de aplicação KNN com valor de K entre 3 e 6

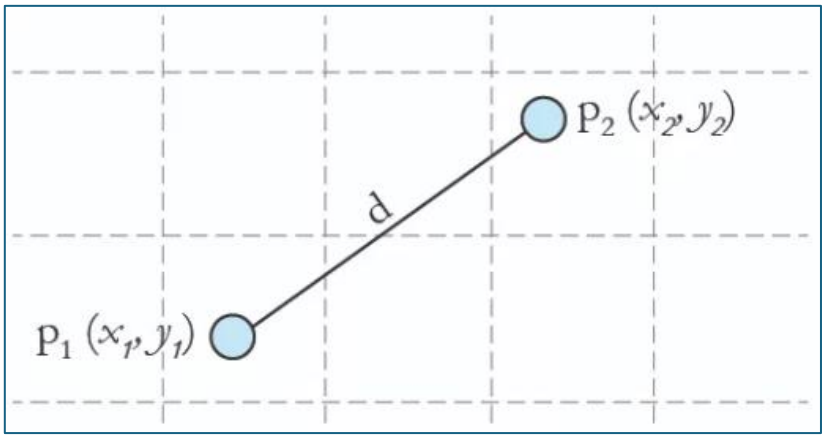


Fonte (Matos, 2023)

Para encontrar esses vizinhos, são aplicadas equações matemáticas para calcular a distância entre os pontos, sendo a mais comum a distância Euclidiana.

Em resumo, esse cálculo determina a menor distância "d" entre dois pontos dados, P1(x1, y1) e P2(x2, y2), usando suas coordenadas. Ilustrada na Figura 4.

Figura 4 – Ilustração da distância entre dois pontos



Fonte (Matos, 2023)

2.1.4 A linguagem Python e suas aplicações de IA

Segundo (Gomes, 2024) a IA tem causado transformações significativas em várias áreas, como análise de dados e desenvolvimento de aplicativos inteligentes. Para quem começa a trabalhar com essa tecnologia, uma das primeiras decisões que precisará tomar é escolher a linguagem de programação certa. Neste artigo, vamos explicar por que Python é a melhor escolha para trabalhar com IA.

O Que é Python?

Python é uma linguagem de programação de alto nível, interpretada, orientada a objetos e funcional. Criada por Guido Van Rossum e lançada em 1991, Python se destaca pela sua simplicidade e legibilidade, tornando-se uma excelente opção tanto para iniciantes quanto para profissionais experientes.

Por Que Python é Ideal para Inteligência Artificial?

Há algumas vantagens a serem consideradas ao escolher utilizar Python para trabalhar com IA, como por exemplo:

- **Simplicidade e Facilidade de Uso**

Python é conhecida por sua sintaxe clara e objetiva, o que facilita o aprendizado e a escrita de código. Isso permite que os desenvolvedores se concentrem mais na lógica do problema do que na complexidade da sintaxe.

- **Ampla Gama de Bibliotecas e *Frameworks***

Python conta com uma grande variedade de bibliotecas e frameworks voltados para IA e *Machine Learning*, como *TensorFlow*, *Keras*, *Scikit-learn* e *PyTorch*. Essas ferramentas tornam o desenvolvimento de modelos de IA mais rápido e eficiente. Além de que a maior parte da documentação dessas bibliotecas está em Python, o que faz com que o aprendizado seja facilitado mais ainda.

- **Comunidade Ativa e Suporte**

A comunidade de Python é uma das mais engajadas e colaborativas na área da programação. O que possibilita ter acesso a uma grande quantidade de recursos, tutoriais, fóruns e grupos de discussão para resolver dúvidas e compartilhar conhecimentos. Saber procurar respostas é uma habilidade essencial na programação, e a comunidade de Python facilita esse processo.

- **Integração com APIs**

Python é amplamente utilizada para integração com APIs de IA, como a API da OpenAI. Isso colabora com a criação de aplicativos que utilizam modelos avançados de linguagem, como o ChatGPT, para realizar tarefas como geração de texto, tradução e até mesmo transcrição de áudios.

Em resumo, Python é uma linguagem muito eficiente para trabalhar com Inteligência Artificial. Sua simplicidade, vasta oferta de bibliotecas, comunidade ativa e facilidade de integração com APIs tornam-na a escolha perfeita para o desenvolvimento de aplicações de IA. Se você está iniciando seus estudos em IA, Python é a ferramenta que permitirá transformar suas ideias em realidade de maneira rápida e eficiente.

2.2 Trabalhos relacionados

Essa seção aborda alguns estudos relacionados ao uso de Machine Learning e Regressão Linear para previsões e auxílio de diagnósticos, além da importância dos médicos durante esse processo.

2.2.1 Comparação e Seleção de Algoritmos *de Machine Learning* para Previsão de Diabetes: Um Estudo Quantitativo Exploratório Baseado em Análise de Dados Médicos

O objetivo do estudo de Santos (2024), foi empregar técnicas de *Machine Learning* para prever a incidência de diabetes em uma população específica de mulheres de herança Pima, utilizando uma base de dados de medidas diagnósticas. O estudo buscou avaliar a eficácia de diferentes algoritmos, incluindo SVM, Redes Neurais Artificiais, KNN, Árvores de Decisão e Floresta Aleatória, para desenvolver modelos preditivos capazes de melhorar a precisão e a eficiência no diagnóstico precoce de diabetes.

O estudo utilizou uma base de dados pública, processando-a para imputação de valores faltantes e normalização das variáveis. Além disso, a *Principal Component Analysis* ou Análise de Componentes Principais (PCA) foi implementada para a redução de dimensionalidade. Diversos algoritmos de *Machine Learning* foram aplicados, incluindo *Support Vector Machines* (SVM), Redes Neurais Artificiais, KNN, Árvores de Decisão e Floresta Aleatória, e os modelos foram avaliados com base em métricas como acurácia, precisão, *F1-Score* e matriz de confusão.

Os resultados evidenciaram que o algoritmo de Floresta Aleatória apresentou a melhor performance em termos de acurácia e precisão, destacando-se como o mais eficaz para a previsão de diabetes entre os algoritmos testados. A utilização do PCA contribuiu para otimizar o desempenho dos modelos, ao reduzir a complexidade dos dados e destacar as variáveis diagnósticas chave.

O estudo concluiu que a Floresta Aleatória é a abordagem mais viável para a previsão de diabetes nesta população específica, sugerindo sua aplicabilidade como

ferramenta clínica para diagnóstico. A pesquisa contribuiu para o campo de aplicações de inteligência artificial na saúde, oferecendo *insights* para a prevenção e tratamento precoce do diabetes.

2.2.2 Predição de diabetes tipo 2 usando algoritmo de vizinho mais próximo K

O trabalho de Muthu e Suriya (2023) apresenta um estudo detalhado sobre a utilização do algoritmo KNN para prever o risco de diabetes tipo 2, utilizando dados clínicos e de estilo de vida. A pesquisa visa a detecção precoce da diabetes tipo 2, permitindo que pacientes em alto risco possam receber cuidados preventivos antes do surgimento de complicações. O estudo combina técnicas de aprendizado de máquina, pré-processamento de dados e avaliação de modelos para fornecer uma visão robusta sobre o uso do KNN em contextos de saúde.

O principal objetivo do estudo é desenvolver um modelo preditivo capaz de classificar pacientes entre categorias de alto risco e baixo risco para o desenvolvimento de diabetes tipo 2. A motivação do estudo é a crescente prevalência da diabetes tipo 2 em todo o mundo, uma condição frequentemente associada a altos níveis de glicose no sangue devido à resistência à insulina ou deficiência na produção de insulina.

O KNN foi escolhido por ser um método não-paramétrico de fácil implementação e aplicável a dados médicos. Ele permite uma abordagem de classificação que se baseia nos "vizinhos" mais próximos do ponto de dados alvo, determinando a categoria desse novo ponto com base na maioria de seus vizinhos mais próximos.

Para realizar o estudo, os autores utilizaram dados da Pima *Indian Heritage Dataset*, um conjunto de dados publicamente disponível no Kaggle, uma plataforma de ciência de dados. A base de dados inclui informações detalhadas de saúde sobre pacientes com e sem diagnóstico de diabetes tipo 2. Esse conjunto de dados foi escolhido por ser um padrão em estudos de predição de diabetes e incluir variáveis clinicamente relevantes para a condição.

O conjunto de dados contém 768 amostras e 9 atributos principais, sendo eles:

- *Pregnancies* - Número de gestações.
- Glucose - Concentração de glicose plasmática duas horas após o teste de tolerância oral.
- *Blood Pressure* - Pressão arterial diastólica em mm/Hg.
- *Skin Thickness* - Espessura da pele do tríceps em milímetros.
- *Insulin* - Níveis de insulina sérica em mU/ml.
- BMI - Índice de Massa Corporal (IMC).
- *Diabetes Pedigree Function* - Grau de hereditariedade para diabetes, com base no histórico familiar.
- *Age* - Idade do paciente.
- *Outcome* - Variável binária que indica a presença (1) ou ausência (0) de diabetes.

O pré-processamento dos dados incluiu várias etapas essenciais para preparar a base de dados para o algoritmo KNN, garantindo que as variáveis fossem adequadamente tratadas para maximizar o desempenho do modelo. Essas etapas foram:

- Remoção de valores nulos e outliers para reduzir inconsistências e ruídos, pois dados ausentes e valores extremos poderiam comprometer a precisão do modelo.
- Utilização de normalização para que todos os atributos numéricos fossem ajustados à mesma escala, o que é especialmente importante no KNN, que é sensível à magnitude dos dados.
- Divisão do conjunto de dados em 80% para treino e 20% para teste, permitindo uma avaliação confiável do modelo em novos dados não vistos.

Para implementar o modelo, os autores experimentaram com diferentes valores de “K”. Este parâmetro é crítico para o desempenho do KNN: valores muito altos podem levar à subadequação (*underfitting*), enquanto valores muito baixos podem resultar em sobreajuste (*overfitting*). A escolha de “K” foi feita com base na acurácia do modelo, utilizando valores que variaram entre 5 e 40 para identificar o “K” ótimo em diferentes configurações.

O desenvolvimento do modelo foi realizado em Python com o uso de bibliotecas populares de aprendizado de máquina e análise de dados, incluindo NumPy, Pandas, *Scikit-Learn* e *Matplotlib* para visualização. Essas ferramentas facilitaram o pré-processamento dos dados, a implementação do KNN, a avaliação de desempenho e a visualização dos resultados.

Os autores avaliaram o modelo KNN em três conjuntos de dados diferentes, analisando como o desempenho do KNN variava conforme o tamanho e a complexidade dos dados:

- **Primeiro Conjunto:** Com 1.873 registros e 22 variáveis, o modelo atingiu sua melhor acurácia com K=40, mas obteve uma precisão relativamente baixa de 70%. A análise indicou que, com o aumento do número de atributos, o modelo encontrou dificuldades em identificar corretamente as classes.
- **Segundo Conjunto:** Continha 769 registros e 9 variáveis (um subconjunto mais restrito de dados). A acurácia máxima foi atingida com K=21, e o desempenho foi superior ao primeiro conjunto, sugerindo que o modelo KNN se adapta melhor a conjuntos de dados com menos variáveis e maior especificidade.
- **Terceiro Conjunto:** Com 692 registros e 9 variáveis, este conjunto foi ainda mais reduzido, permitindo o uso de *Principal Component Analysis* (PCA) para diminuir a dimensionalidade. Com K=5, o modelo mostrou resultados robustos e boa acurácia, aproximadamente 76%, sugerindo que o KNN tende a funcionar bem em conjuntos de dados menores e bem segmentados.

A eficácia do modelo foi avaliada usando matrizes de confusão, que apresentam os valores de verdadeiros positivos, falsos positivos, verdadeiros negativos e falsos negativos, além das métricas de precisão, *recall* e *F1-score*. Estas métricas ajudaram a quantificar a capacidade do KNN em identificar corretamente indivíduos com e sem diabetes.

Além disso, os autores aplicaram o método de avaliação *K-Fold Cross Validation,* ou validação cruzada, com 10 folds para mitigar problemas de sobreajuste e avaliar a estabilidade do modelo em diferentes partições do conjunto de dados.

Os resultados indicaram que a precisão do KNN é altamente dependente do tamanho e da complexidade do conjunto de dados. Para conjuntos de dados menores, o modelo conseguiu obter uma boa acurácia, especialmente quando o valor de “K” foi otimizado. Em bases de dados maiores, o modelo mostrou limitações, refletindo uma diminuição de precisão com o aumento da quantidade de variáveis. Isso sugere que, apesar de ser eficaz, o KNN tem restrições quando aplicado a conjuntos de dados complexos ou com alta dimensionalidade.

Os autores observaram que, em conjuntos de dados grandes, a precisão do modelo se beneficia de valores mais altos de “K”, mas em conjuntos menores, o uso de valores menores de “K” mostrou melhores resultados. Além disso, o pré-processamento adequado, especialmente a normalização, foi essencial para maximizar o desempenho do KNN, devido à sensibilidade do algoritmo a diferenças de escala entre variáveis.

O estudo concluiu que o KNN é uma abordagem viável para a predição de diabetes tipo 2, particularmente em conjuntos de dados que passaram por um rigoroso pré-processamento e possuem uma dimensionalidade controlada. O KNN se mostrou eficiente ao categorizar corretamente indivíduos em alto ou baixo risco para diabetes quando os dados eram bem balanceados e normalizados.

Contudo, o desempenho do modelo mostrou-se limitado em bases de dados com muitos atributos. Os autores sugerem que, para futuras pesquisas, é recomendável utilizar conjuntos de dados maiores e considerar algoritmos de classificação mais avançados, como *Random Forest* ou SVM, que podem oferecer melhor acurácia em cenários de alta dimensionalidade. Outra possibilidade é investigar métodos de otimização automática de hiperparâmetros e explorar técnicas de seleção de características para melhorar a eficácia do modelo. Esses aprimoramentos poderiam transformar o modelo em uma ferramenta prática para aplicação em ambientes clínicos, oferecendo uma abordagem mais confiável e robusta para a predição de diabetes.

2.2.3 A importância do papel dos médicos diante a utilização de IA para diagnósticos e questões de segurança

Essa subseção aborda especificamente estudos sobre a importância do papel do médico em situações em que a IA é usada para diagnósticos, e questões de segurança ao utilizar essa tecnologia.

2.2.3.1 Inteligência Artificial em Cuidados de Saúde: Aplicações e Problemas Atuais

De acordo com o estudo de Park et al (2020), apesar das vantagens proporcionadas pelas tecnologias de IA, como a capacidade de processar grandes volumes de dados de maneira eficiente, os dispositivos dependentes da computação em nuvem podem suscitar preocupações sérias de segurança em relação aos dados de saúde privados. Isso ocorre devido à transmissão e ao armazenamento dos dados em servidores remotos, aumentando o risco de violações de segurança e acesso não autorizado.

Para solucionar esses problemas, estão sendo conduzidas pesquisas técnicas e esforços para modificar leis e regulamentos. Em termos de pesquisa técnica, diversas tecnologias de criptografia e técnicas de desidentificação ou anonimização estão sendo desenvolvidas para proteger os dados de saúde. Por exemplo, o aprendizado federado e a criptografia homomórfica são técnicas que permitem o processamento de dados em sua forma criptografada, preservando simultaneamente a privacidade dos dados.

Em todo o mundo, muitos países estão estabelecendo sistemas institucionais e legais para lidar com os interesses divergentes entre o uso da informação de saúde e a proteção das informações pessoais. Nos Estados Unidos, a Lei de Portabilidade e Responsabilidade do Seguro de Saúde, promulgada em 1996, concedeu aos indivíduos direitos sobre seus dados de saúde e incentivou o desenvolvimento de sistemas como o Blue Button, que permite aos pacientes acessar e visualizar seus próprios registros médicos online. Além disso, a Lei de Tecnologia da Informação em Saúde para a Saúde Econômica e Clínica de 2009 estimulou o desenvolvimento de

registros eletrônicos de saúde para melhorar a interoperabilidade das informações médicas entre os hospitais. Os Centros de *Medicare* & *Medicaid* também lançaram serviços como *MyHealthEData* e *Blue Button* 2.0 para capacitar os pacientes a acessar e controlar seus próprios dados de saúde. Na Europa, o Regulamento Geral de Proteção de Dados ou *General Data Protection Regulation* (GDPR), adotado em 2016, fortaleceu os direitos individuais sobre informações pessoais e estabeleceu princípios rigorosos de proteção de dados. Na Coreia, alterações legislativas, como revisões na Lei de Bioética e Segurança, foram realizadas para promover a pesquisa em big data de saúde.

Apesar dos esforços em todo o mundo, as questões de privacidade relacionadas aos dados de saúde ainda não foram totalmente resolvidas. O equilíbrio entre a utilização desses dados para avanços médicos e a proteção da privacidade pessoal continua sendo um desafio complexo e em constante evolução. As políticas regulatórias referentes aos novos dispositivos médicos baseados em IA estão em constante evolução, com diversas iniciativas sendo implementadas ao redor do globo.

Nos Estados Unidos, a *Food and Drug Administration* (FDA) estabeleceu diretrizes para o "Software como Dispositivo Médico", reconhecendo a necessidade de uma abordagem distinta para a aprovação e regulação desses softwares. No Japão e na Coreia, esforços estão sendo concentrados na criação de regras abrangentes para governar o uso de IA em dispositivos médicos, com o objetivo de minimizar controvérsias e facilitar o desenvolvimento. Contudo, ainda não há uma padronização global para a revisão de segurança e eficácia. Questões relacionadas à segurança e responsabilidade também ganham destaque. Nos Estados Unidos, há um foco na equidade e na prevenção da discriminação na aplicação de IA promovendo avaliações embasadas em evidências.

A responsabilidade legal em casos de acidentes médicos associados à IA é uma preocupação, com sugestões de políticas visando esclarecer responsabilidades e estabelecer centros de monitoramento. A integração da IA com os sistemas de saúde já existentes demanda cuidadosas considerações. A implementação da IA deve ser feita de forma harmoniosa e ser constantemente monitorada para evitar possíveis problemas 32 inesperados. Além disso, é crucial desenvolver interfaces intuitivas para

garantir que a tecnologia seja facilmente adotada pela equipe médica, assegurando a continuidade das práticas médicas atuais.

O estudo conclui que a expectativa em torno das tecnologias de IA impulsionem inovações tanto nas tecnologias médicas existentes quanto nos cuidados de saúde futuros. As tecnologias de saúde baseadas em IA disponíveis atualmente têm demonstrado excelentes resultados na precisão do diagnóstico e na classificação das condições dos pacientes, além de prever o curso das doenças utilizando os dados médicos acumulados. Como resultado, espera-se que essas tecnologias contribuam para auxiliar a equipe médica na tomada de decisões de tratamento, potencialmente melhorando os resultados dos tratamentos.

No entanto, as tecnologias de saúde baseadas em IA enfrentam várias questões relacionadas à privacidade, confiabilidade, segurança e responsabilidade. Para que essas tecnologias de IA sejam mais amplamente adotadas na área da saúde, será necessário um aumento da conscientização pública sobre a IA, o estabelecimento de diretrizes padronizadas e melhorias sistemáticas no futuro, além dos avanços tecnológicos.

2.2.3.2 O papel da inteligência artificial na saúde: uma revisão estruturada da literatura

O estudo de Secinaro et al (2021), aborda o crescente papel da IA na saúde e analisa as principais áreas em que essa tecnologia pode auxiliar o setor. A partir de uma revisão sistemática e uma análise bibliométrica, o estudo visa mapear como a IA tem sido aplicada e identificar os principais desafios e oportunidades para o futuro. Para isso, os pesquisadores utilizaram 288 artigos revisados por pares da base de dados da Scopus, o que permitiu uma análise robusta do estado atual da IA na saúde.

A metodologia utilizada combina uma Revisão Sistemática da Literatura ou *Systematic Literature Review* (SLR) e técnicas de análise bibliométrica, auxiliadas pelo software Bibliometrix. O uso dessa abordagem possibilitou que os autores extraíssem e estudassem variáveis qualitativas e quantitativas, como redes de colaboração, palavras-chave, autores mais citados e periódicos de maior relevância. A partir da análise, foram identificadas cinco áreas principais em que a IA vem

desempenhando um papel significativo e que têm potencial para transformar práticas de saúde no futuro. O estudo destaca também as principais áreas de aplicação da IA na Saúde, que são elas:

- **Gestão de Serviços de Saúde:** A IA tem sido amplamente utilizada para otimizar a gestão de serviços de saúde, melhorando desde a administração de recursos até o suporte em decisões estratégicas. No contexto hospitalar, por exemplo, sistemas de IA podem organizar de maneira mais eficaz a distribuição de medicamentos e materiais médicos, utilizando algoritmos de previsão de demanda. Isso é especialmente importante em tempos de crise, como durante a pandemia de COVID-19, onde houve uma necessidade urgente de coordenação eficiente de suprimentos e informações. Além disso, a IA ajuda profissionais da saúde a receberem atualizações constantes de informações médicas de diversas fontes, como periódicos e protocolos clínicos, agilizando o processo de atualização profissional e permitindo uma resposta mais rápida a novas situações de saúde pública.
- **Medicina Preditiva:** A aplicação da IA na medicina preditiva é uma das áreas mais promissoras, pois permite a previsão do desenvolvimento de doenças e a personalização de tratamentos. Ao analisar dados extensivos de pacientes, algoritmos de IA conseguem identificar padrões que ajudam a prever complicações ou mudanças no estado de saúde de um paciente, possibilitando uma abordagem proativa e preventiva. Além disso, a IA auxilia na descoberta de novos medicamentos e na criação de tratamentos personalizados com base no perfil genético e histórico clínico de cada paciente, o que resulta em um tratamento mais eficaz e menos invasivo. A capacidade de prever resultados clínicos e ajustar os cuidados de acordo com o risco específico de cada paciente pode melhorar consideravelmente o atendimento oferecido.
- **Dados de Pacientes e Diagnóstico:** O grande volume de dados gerados no setor de saúde representa um desafio para profissionais e administradores. A IA é capaz de processar e analisar esses dados de forma mais rápida e eficiente, extraindo insights valiosos que podem ser utilizados no diagnóstico e no tratamento de doenças. Uma aplicação prática é no

processamento de imagens médicas, onde a IA pode ajudar a identificar lesões ou anomalias em exames, como ressonâncias magnéticas ou tomografias, com precisão e em alta velocidade. A análise de dados de pacientes também suporta o desenvolvimento de sistemas de monitoramento remoto, ampliando o uso da telemedicina, especialmente em locais onde o acesso a serviços de saúde é limitado. Durante a pandemia, a telemedicina se tornou uma ferramenta vital, e a IA continua a aprimorar esses serviços, fornecendo suporte para diagnósticos e tratamento à distância.

- **Suporte à Tomada de Decisão Clínica:** A IA se mostra uma aliada fundamental na tomada de decisões clínicas, auxiliando médicos e profissionais de saúde com informações e sugestões baseadas em evidências. Através de algoritmos que analisam grandes bases de dados clínicos, a IA pode gerar recomendações de diagnóstico e tratamento em casos complexos, apoiando médicos na escolha do melhor caminho para cada paciente. A capacidade de fornecer recomendações em tempo real reduz o tempo necessário para decisões críticas, minimizando o risco de erros e permitindo um atendimento mais ágil e seguro. Esse suporte à decisão pode ser especialmente útil em emergências médicas, onde cada segundo conta e decisões precisam ser tomadas com rapidez e precisão.
- **Análise Bibliométrica e Identificação de Tendências:** Na análise bibliométrica, os autores identificaram as principais revistas e autores de destaque na área de IA aplicada à saúde. As publicações mais frequentes sobre o tema foram encontradas em periódicos como *Journal of Medical Systems* e IEEE *Journal of Biomedical and Health Informatics*. Em termos de produtividade, os Estados Unidos, China e Reino Unido se destacam como os países com maior número de publicações, indicando um alto nível de investimento e pesquisa na área. Além disso, o estudo destaca que os termos mais comuns encontrados nos artigos analisados foram "inteligência artificial", "sistema de suporte à decisão clínica" e "análise preditiva", refletindo as principais áreas de foco dos pesquisadores.

Os autores concluíram que, embora a IA ofereça avanços promissores para a saúde, ainda existem desafios significativos a serem superados. Questões como a qualidade dos dados, a preparação dos profissionais de saúde para lidar com novas tecnologias e os desafios éticos relacionados ao uso de dados de pacientes precisam ser abordados para que a IA possa ser adotada de forma segura e eficaz. O artigo enfatiza a importância de uma gestão cuidadosa e ética dos dados para garantir a privacidade e a transparência, especialmente quando se trata de dados de saúde.

Além disso, o estudo sugere a necessidade de pesquisas interdisciplinares que combinem tecnologia, medicina e ciências sociais para explorar completamente o potencial e as limitações da IA. Isso inclui a análise das implicações éticas do uso de IA na tomada de decisões médicas e a investigação de possíveis riscos, como a dependência excessiva da tecnologia e a consequente "desqualificação" dos profissionais de saúde, que poderiam perder algumas habilidades clínicas devido à automação de processos. A aplicação da IA tem o potencial de transformar a prática médica, melhorando a precisão diagnóstica, otimizando o uso de recursos e personalizando tratamentos. Contudo, o desenvolvimento e a implementação de IA na saúde devem ser realizados com cuidado, respeitando as questões éticas e considerando a preparação dos profissionais para essa nova realidade.

3 MÉTODO

Este trabalho quanto a sua natureza é um resumo de assunto, com o objetivo de aprofundar o conhecimento sobre um tema específico. Através da investigação e análise crítica de informações relevantes, busca-se compreender as causas e as explicações subjacentes ao tema em questão (Wazlawick, 2014).

Quanto aos objetivos esta pesquisa é exploratória. Nessa modalidade, o pesquisador não possui necessariamente uma hipótese ou objetivo pré-definido, mas sim busca explorar e compreender um conjunto de fenômenos em busca de novas descobertas. A pesquisa exploratória pode ser vista como a etapa inicial de um processo investigativo mais amplo. Através da análise de dados e da identificação de anomalias ainda desconhecidas, o autor abre caminho para pesquisas futuras mais elaboradas (Wazlawick, 2014).

Quanto aos procedimentos técnicos esta pesquisa é bibliográfica e experimental.

Conforme Gil (2017), a pesquisa bibliográfica segue um conjunto de etapas essenciais para sua efetividade:

a) Escolha do Tema: Uso do modelo KNN para predição de doenças.

b) Levantamento Bibliográfico Preliminar: A etapa seguinte envolve um mergulho inicial no tema, buscando familiarização e aprofundamento. Através de pesquisas em bases de dados especializadas como Capes e Google Scholar, além de livros relevantes do acervo pessoal, o pesquisador obtém uma visão geral do tema e identifica pontos de pesquisa promissores.

c) Formulação do Problema: - Como o uso do Algoritmo do Vizinho Mais Próximo ou *K-Nearest Neighbors* (KNN) pode ser eficiente na predição de doenças?

d) Busca das Fontes: Foi buscado artigos científicos em plataformas tais como Capes e Google Scholar, livros e materiais relevantes compõem o arsenal de informações que sustentarão o estudo. Além de TCC nos repositórios das universidades.

e) Leitura do Material: Foi realizada leituras atentas e críticas do material selecionado para identificar os pontos-chave relacionados ao tema de pesquisa.

f) Fichamento: Para organizar as informações coletadas, a elaboração de fichas de citação se torna essencial. Cada ficha registra as citações importantes do material bibliográfico, permitindo um acesso rápido e organizado às informações durante a pesquisa.

g) Redação do Texto: A escrita da monografia de TCC II.

De acordo com Wazlawick (2014), a pesquisa experimental envolve a manipulação de um aspecto específico da realidade pelo pesquisador. Por exemplo, ao introduzir uma nova técnica em uma empresa de software, o pesquisador observa se há um aumento na produtividade. Essa abordagem requer o controle de uma ou mais variáveis experimentais (como o uso ou não da técnica) e a medição de variáveis observadas para verificar se há dependência entre elas, como no caso de avaliar se a técnica impacta significativamente a produtividade dos programadores.

Pesquisas experimentais têm como objetivo determinar o objeto de estudo, selecionar as variáveis que podem influenciá-lo e definir as formas de controle e

observação dos efeitos gerados por essas variáveis no objeto em questão (GIL, 2017). Assim, a presente pesquisa experimental segue as seguintes etapas:

a) **Formulação do problema**: Como o uso do Algoritmo do Vizinho Mais Próximo ou *K-Nearest Neighbors* (KNN) pode ser eficiente na predição de doenças?

b) **Definição do plano experimental**: Foi coletado a base de dados do estudo: ***Likelihood Prediction of Diabetes at Early Stage Using Data Mining Techniques*****.** A partir desses dados foi desenvolvido um modelo KNN para realizar a previsão dos casos de diabetes, utilizando a linguagem Python.

c) **Determinação do ambiente**: Para realizar a implementação do modelo, foi escolhida a *Integrated Development Environment* (IDE) *Virtual Studio Code*, na versão 1.95, por ser uma ferramenta robusta e amplamente utilizada no desenvolvimento de projetos Python, que foi usado a versão 3.11.4 da linguagem. A configuração do ambiente inclui a utilização das bibliotecas: numpy, pandas, scikit-learn e matplotlib. O computador utilizado possui uma configuração com um processador AMD Ryzen 5 3500U com Radeon Vega Mobile Gfx 2.10 GHz, 12 GB de memória RAM com Windows 11 versão 23H2 de 64 bits.

d) **Coleta de dados**: A partir da base de dados do estudo selecionado, foi realizado o pré-processamento dos dados, incluindo tratamento de valores ausentes e normalização. Depois, o conjunto de dados foi dividido em treinamento e teste. O modelo foi treinado com o conjunto de treinamento, ajustando o valor de k.

e) Análise e interpretação dos resultados: Após o treinamento do modelo, as métricas como precisão, revocação, *F1-Score* e acurácia foram utilizadas para avaliar a eficácia do modelo. As médias macro avg e *weighted* avg foram usadas para ajudar a entender o desempenho em cenários de classes desbalanceadas. As conclusões ajudam a determinar a efetividade do modelo e identificar melhorias.

f) Redação do relatório: A escrita da monografia de TCC II.

4 Implementação do algoritmo do modelo KNN

Este capítulo descreve como o modelo KNN é implementado utilizando a linguagem Python e apresenta o desenvolvimento do modelo realizado para este estudo.

4.1 Como o modelo KNN é desenvolvido e aplicado com a linguagem Python

O KNN é um modelo direto e intuitivo, ele opera analisando quais dados estão "mais próximos" uns dos outros. Diferente de outros modelos que fazem um treinamento de fato, antes de fazer as previsões, o KNN só "aprende" na hora da classificação ou previsão, olhando diretamente para o conjunto de dados guardado. Para isso ele realiza os seguintes procedimentos:

- **Analisar e armazenar os Dados**: O primeiro passo é armazenar os dados de treinamento. Isso significa que ele não ajusta ou processa esses dados antes. Ele apenas "memoriza" todas as características de cada amostra e os rótulos (a classe ou valor numérico) que ela possui.
- **Calcular a Distância para Previsão:** Para classificar ou prever uma nova amostra, o modelo calcula a distância entre essa amostra e todos os pontos do conjunto de treinamento, sendo a distância Euclidiana (que mede o "caminho reto" entre dois pontos) é a mais usada nesse tipo de modelo, porém existem outras opções. Esse cálculo mostra quais pontos do conjunto de dados estão mais próximos da amostra nova.
- **Encontrar os Vizinhos Mais Próximos:** Depois de calcular todas as distâncias, o KNN escolhe os “K” pontos mais próximos (vizinhos) da amostra nova. O número “K” é escolhido por quem está usando o modelo, e é bom que seja um número que balanceie precisão e estabilidade.
- **Fazer a Classificação ou Regressão:**

- Para classificação, o KNN vê as classes desses k vizinhos mais próximos e atribui à nova amostra a classe que aparece com mais frequência. Então, se a maioria dos vizinhos for da classe "A", a nova amostra também será da classe "A".

- Para regressão, o KNN calcula a média dos valores dos “K” vizinhos mais próximos e usa esse valor como a previsão para a nova amostra.

Como o KNN não passa por um treinamento prévio, ele é chamado de algoritmo de "aprendizado preguiçoso" (*lazy learning*), pois só faz o "trabalho" no momento da previsão. Isso faz dele um método bem fácil de entender e usar, mas ele pode ficar mais lento se o conjunto de dados for muito grande, pois precisa calcular a distância para cada ponto.

4.1.1 Análise e Armazenamento dos Dados

Quando se lida com grandes volumes de dados, a eficiência do KNN pode ser desafiadora, pois ele precisa calcular distâncias entre os pontos. Além disso, com tanta informação disponível, dados irrelevantes podem se acumular, aumentando o esforço computacional e dificultando o desempenho do modelo.

Para evitar esse problema, técnicas como pré-processamento e seleção de características são extremamente úteis. Também é importante considerar a dimensionalidade do espaço de características, pois isso pode afetar a precisão e o tempo de resposta do KNN.

Portanto, o ideal é encontrar um equilíbrio entre eficiência, capacidade de generalização e resistência ao ruído. Um exemplo seria um grande conjunto de dados de avaliações de produtos: a eficiência do KNN permitiria recomendações rápidas, enquanto a seleção de características eliminaria informações irrelevantes, resultando em sugestões mais precisas e relevantes para os usuários (Cinnecta, 2024).

Para armazenar os dados de forma eficiente em um algoritmo de KNN pode-se usar estruturas tais como:

- Listas ou *Arrays*: Simples e rápidas para acessar os dados, mas não são as mais eficazes em busca e distância.
- Matrizes NumPy: No Python, as matrizes da biblioteca NumPy permitem operações matemáticas rápidas e eficientes, o que é útil para calcular distâncias.

- Árvores KD (*K-Dimensional Tree*): Essa estrutura organiza os dados de forma hierárquica e é muito eficiente para consultas de proximidade, especialmente em grandes conjuntos de dados.
- DataFrame: Uma tabela bidimensional, semelhante a uma planilha ou a uma tabela SQL, onde os dados são organizados em linhas e colunas. Cada coluna pode ter um tipo de dado diferente (números, *strings*, booleanos, etc.), permitindo uma grande flexibilidade na manipulação dos dados.

As matrizes NumPy são uma boa opção para conjuntos de dados menores, mas para grandes volumes, as árvores KD ou estruturas semelhantes podem reduzir bastante o tempo de busca.

Tipos de Dados Mais Eficientes para Análise do KNN

Para que o KNN funcione bem, os dados devem ser numéricos e padronizados, como por exemplo:

- **Dados Numéricos:** O KNN funciona melhor com dados contínuos e numéricos, pois ele calcula distâncias para identificar vizinhos próximos. Dados categóricos podem ser usados, mas é necessário convertê-los para valores numéricos (por exemplo, usando *one-hot encoding* ou rótulos numéricos).
- **Escala Consistente:** Como o KNN depende de cálculos de distância, é importante que os dados estejam na mesma escala. Variáveis com escalas diferentes (por exemplo, uma variável que vai de 0 a 1000 e outra de 0 a 1) podem distorcer as distâncias. Padronizar os dados (normalizar entre 0 e 1 ou usar padronização *z-score*) ajuda a garantir que cada característica contribua igualmente para a análise.

Garantia de que os Dados Sejam Úteis para Previsão

Para que os dados armazenados sejam realmente úteis para o modelo KNN, é importante considerar alguns pontos como:

- **Qualidade dos Dados:** Dados ruidosos (valores atípicos ou inconsistentes) podem prejudicar a precisão do KNN, especialmente com valores de k baixos. Fazer uma limpeza nos dados, removendo ou ajustando valores atípicos e preenchendo dados ausentes, é essencial.
- **Relevância das Características:** Cada característica ou variável deve agregar valor à previsão. Variáveis irrelevantes ou redundantes podem acrescentar "ruído" e reduzir a precisão. Técnicas de seleção de características, como análise de correlação ou seleção baseada em modelos, ajudam a escolher apenas as variáveis mais relevantes.
- **Balanceamento de Classes (em Classificação):** Se o conjunto de dados estiver muito desbalanceado (muitas amostras de uma classe e poucas de outra), o KNN pode acabar favorecendo a classe majoritária, já que os vizinhos mais próximos provavelmente pertencerão a ela. Em casos de desequilíbrio, pode ser útil aplicar técnicas como *oversampling* (aumentar a quantidade de amostras da classe minoritária) ou *undersampling* (reduzir amostras da classe majoritária).

4.1.2 Definição das Métricas de Distância

Conforme a (IBM, s.d.) o objetivo do algoritmo *K-Nearest Neighbors* (KNN) é identificar os vizinhos mais próximos de um ponto de consulta para determinar sua classe. Para isso, é necessário:

A distância entre o ponto de consulta e os demais pontos precisa ser calculada para identificar quais estão mais próximos. Essas métricas de distância formam os chamados limites de decisão, que dividem o espaço de consulta em diferentes regiões. Geralmente, esses limites são visualizados através de diagramas de Voronoi, um tipo especial de decomposição de um dado espaço.

Existem várias métricas de distância, sendo as principais:

- **Distância Euclidiana (p=2):** Apresentada na fórmula (1.1), é a mais comum para o modelo KNN, usada para medir a distância em linha reta entre dois pontos em um espaço de valores reais.

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(yi - xi)^2} \quad (1.1)$$

x, *y* = dois pontos no espaço euclidiano n

yi, *xi* = vetores euclidianos, partindo da origem do espaço (ponto inicial)

n = espaço n

- **Distância de Manhattan (p=1):** Também conhecida como "Distância de Táxi" ou "Distância de Quarteirão", a Distância Manhattan, ilustrada na fórmula (1.2), mede a soma dos valores absolutos entre dois pontos. Costuma ser visualizada em uma grade urbana.

$$d(x, y) = \quad (\sum_{i=1}^{m}|xi - yi|) \quad (1.2)$$

- **Distância de Minkowski:** Apresentada na fórmula (1.3), uma generalização das distâncias Euclidiana e de Manhattan. O parâmetro "p" permite ajustar a fórmula para calcular diferentes métricas.

$$(\sum_{i=1}^{n}|xi - yi|)^{1/p} \quad (1.3)$$

- **Distância de Hamming:** Usada para comparar vetores booleanos ou *strings*, a Distância de Hamming, ilustrada na fórmula (1.4), mede a quantidade de posições em que os vetores diferem. É comum em análise de sobreposição.

$$Dh = (\sum_{i=1}^{k}|xi - yi|) \quad (1.4)$$

x = *y* ; *D* = 0; *x* ≠ y; *D* ≠ 0

4.1.3 Definição do Valor de K

Para determinar o melhor valor de “K”, o número de vizinhos mais próximos a serem considerados, é necessário testar diferentes valores e identificar aquele que oferece previsões mais precisas, com menos erros. Essa escolha exige equilíbrio:

Valores baixos de “K” tornam as previsões mais instáveis. Por exemplo, imagine que um ponto de consulta esteja cercado por dois pontos verdes e um triângulo vermelho. Se K=1 e o ponto mais próximo for um dos verdes, o algoritmo pode prever incorretamente que o ponto de consulta será verde. Valores baixos de “K” apresentam alta variância (o modelo se adapta excessivamente aos dados de treinamento), alta complexidade e baixo viés (o modelo se ajusta muito bem aos dados de treinamento).

Valores altos de “K” podem trazer muito ruído, que é qualquer variação ou informação irrelevante nos dados que pode distorcer ou atrapalhar a análise e o aprendizado de um modelo. Um k maior tende a aumentar a precisão das previsões, pois há mais dados para calcular a média ou moda. No entanto, um “K” excessivamente alto pode resultar em baixa variância, baixa complexidade e alto viés (o modelo se torna simples demais para se ajustar bem aos dados de treinamento).

Idealmente, busca-se um valor de “K” que equilibre alta variância e alto viés. É recomendado, ainda, escolher um número ímpar para “K”, evitando empates na classificação.

O valor ideal de k também depende do seu conjunto de dados. Para selecioná-lo, uma abordagem inicial é usar a raiz quadrada de N, onde N é a quantidade de dados no conjunto de treinamento. Métodos de validação cruzada também ajudam a definir o k mais adequado ao seu conjunto de dados. (Elastic, s.d)

4.1.4 Classificação ou Regressão dos dados

No KNN, a etapa de classificação ou regressão ocorre após o cálculo das distâncias entre a nova amostra de dados (o ponto que se deseja prever) e os pontos do conjunto de dados já armazenados. Dependendo se o problema é de classificação (definir em grupos) ou de regressão (prever um valor numérico), o modelo KNN utiliza

um procedimento específico para fazer a predição. Cada caso possui uma forma de trabalhar:

Classificação com KNN

Na classificação, o objetivo é saber a qual grupo ou classe a nova amostra pertence. Por exemplo, um caso em que há um conjunto de dados que divide frutas em "maçã" e "laranja" e deseja-se saber se uma nova fruta inserida no conjunto é uma maçã ou uma laranja, a partir de suas características.

O primeiro passo é selecionar os vizinhos mais próximos. Depois de calcular a distância entre a nova fruta e todas as frutas no conjunto de dados, o modelo KNN seleciona os “K” vizinhos mais próximos da nova amostra.

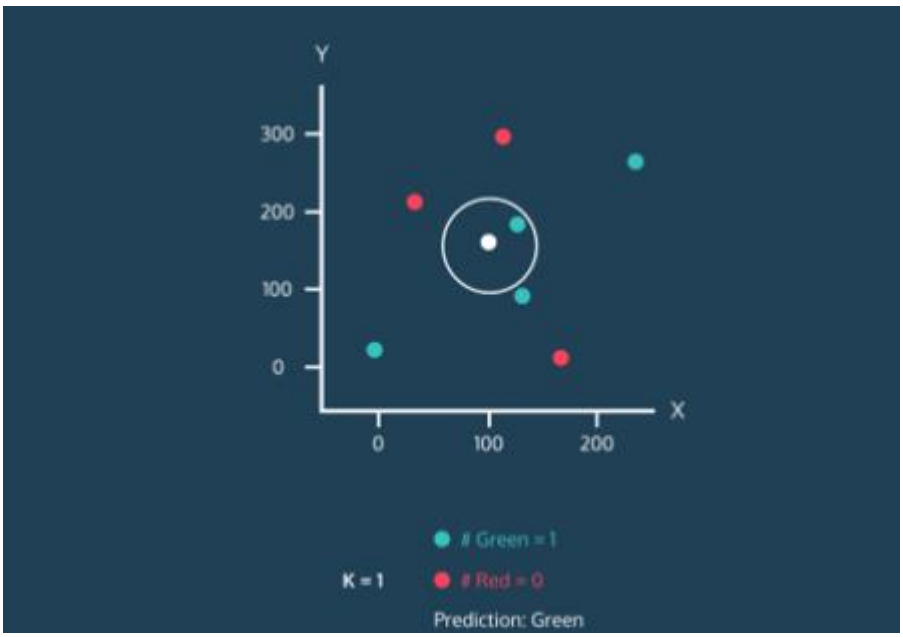
O segundo passo é verificar as classes dos vizinhos. Ele então verifica a classe (ou rótulo) de cada um desses “K” vizinhos. Suponha-se que K=5 e que, entre os cinco vizinhos mais próximos, três são "maçãs" e dois são "laranjas".

Por último, é atribuído a classe mais comum. Na classificação, o KNN funciona como uma "votação". Ele define para nova fruta inserida no conjunto, a classe que for mais comum entre os vizinhos. Neste caso, como a maior parte dos vizinhos são "maçãs", o modelo classificará a nova fruta como uma "maçã".

Esse processo de classificação faz do KNN um método simples, porém eficaz, para categorizar dados. É importante lembrar que o valor de k influencia diretamente nos resultados. Valores muito baixos podem fazer o modelo ser muito específico (suscetível a ruídos), enquanto valores muito altos podem fazer com que o modelo perca precisão.

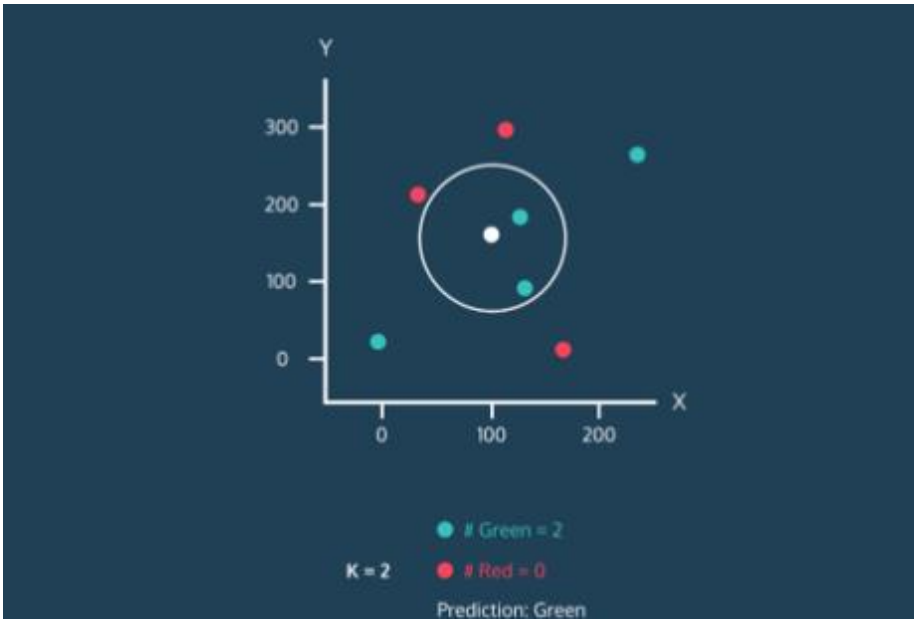
Um exemplo de como a classificação é feita, é ilustrado nas Figuras 5, 6, 7, 8 e 9 onde o novo dado inserido (bola branca), terá a sua classificação baseada no tipo das bolas que estão mais próximas, sendo vermelho ou verde. Pode-se notar também pelas figuras, que a distância e a quantidade de bolas (sendo o valor de K), influencia diretamente no resultado da classificação.

Figura 5 – Valor de K = 1 resultando em classificação verde



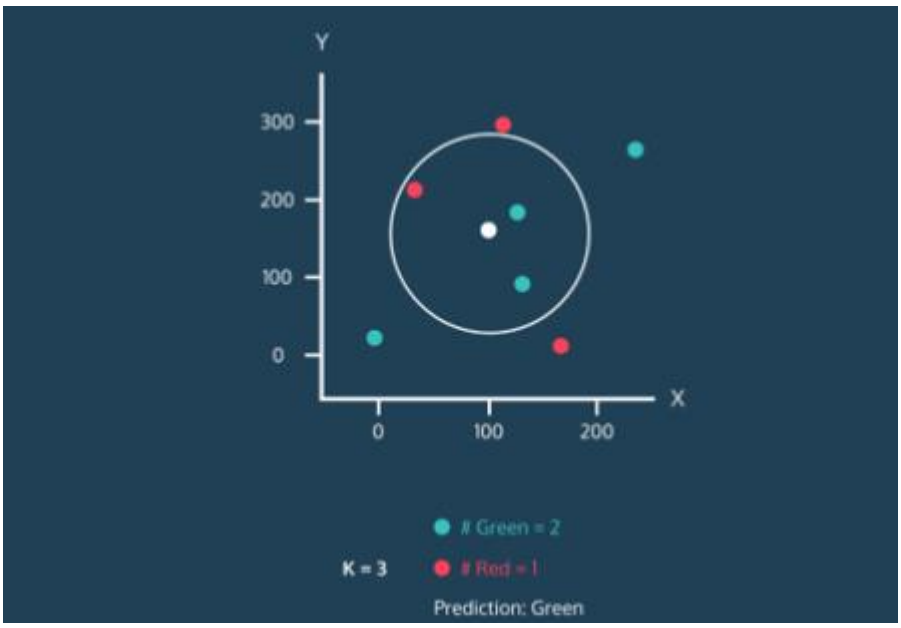
Fonte: (KIM, 2019)

Figura 6 – Valor de K = 2 resultando em classificação verde



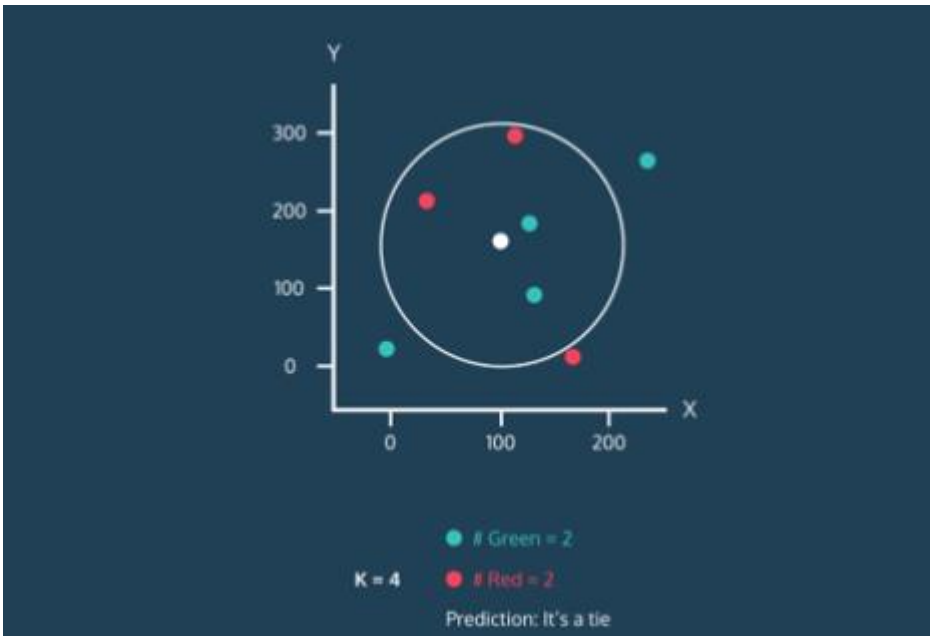
Fonte: (KIM, 2019)

Figura 7 – Valor de K = 3 resultando em classificação verde



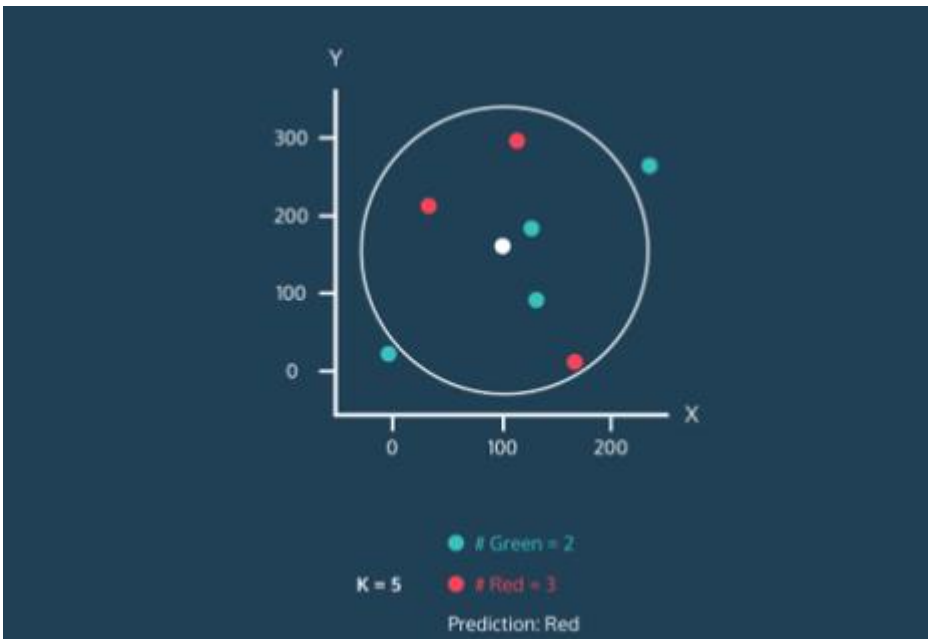
Fonte: (KIM, 2019)

Figura 8 – Valor de K = 4 resultando em empate



Fonte: (KIM, 2019)

Figura 9 – Valor de K = 5 resultando em classificação vermelha



Fonte: (KIM, 2019)

Regressão com KNN

Na regressão, em vez de classificar o novo dado inserido, o objetivo é estimar um valor numérico para ele, com base nos valores dos vizinhos mais próximos. Por exemplo, um caso em que há um conjunto de dados com informações sobre o preço de casas, incluindo fatores como área e localização. O objetivo é prever o preço de uma nova casa usando o KNN.

Primeiro seleciona-se os vizinhos mais próximos. Assim como na classificação, o modelo calcula a distância entre a nova casa e cada casa do conjunto de dados estabelecido, selecionando os “K” vizinhos mais próximos.

Em seguida, observa-se os valores numéricos dos vizinhos: No caso da regressão, cada vizinho próximo terá um valor associado (neste exemplo, o preço da casa). Supondo que K=5, o modelo considerará o preço das cinco casas mais próximas.

Por último, calcula-se a média (ou mediana): Para fazer a previsão, o modelo calcula a média dos preços das “K” casas vizinhas e usa esse valor como a previsão

para o preço da nova casa. Em alguns casos, pode ser mais eficiente utilizar a mediana em vez da média para reduzir o impacto de valores extremos.

Esse método possibilita o modelo KNN fazer uma estimativa baseada em proximidade, o que pode ser muito útil em casos em que as características semelhantes geralmente significam valores numéricos próximos.

4.1.5 Análise de resultados

Para avaliar a eficiência de um modelo, é importante utilizar métricas que indiquem o quão bem ele está fazendo as previsões desejadas. Cada métrica analisa o desempenho do modelo de uma forma diferente, dependendo se está resolvendo um problema de classificação (separar os dados em categorias) ou regressão (estimar valores numéricos).

Métricas para classificação

Quando o modelo KNN é utilizado para classificar as amostras aplicadas, ele precisa atribuir a elas uma categoria correta. Para isso há métricas que auxiliam a avaliar o desempenho do modelo.

-Acurácia: Mede a proporção de previsões corretas em relação ao total de previsões, como ilustrado na fórmula (1.5).

$$\textbf{Acurácia} = \frac{Número\ de\ previsões\ corretas}{Número\ total\ de\ amostras} \quad (1.5)$$

A acurácia é simples e direta, mostrando rapidamente a proporção de acertos. No entanto, ela pode ser enganosa se as classes estiverem desbalanceadas (se houver muito mais exemplos de uma classe do que de outras).

-Matriz de Confusão: Organiza as previsões do modelo em uma tabela que mostra os acertos e os erros para cada classe. A matriz tem quatro valores principais:

- **Verdadeiro Positivo (VP):** Previsões corretas para a classe positiva.

- **Verdadeiro Negativo (VN):** Previsões corretas para a classe negativa.
- **Falso Positivo (FP):** Previsões incorretas para a classe positiva (o modelo previu positivo, mas era negativo).
- **Falso Negativo (FN):** Previsões incorretas para a classe negativa (o modelo previu negativo, mas era positivo).

A matriz de confusão, exemplificada na Tabela 1, ajuda a entender em quais classes o modelo erra mais. Por exemplo, pode mostrar se o modelo confunde duas classes específicas.

Quadro 1 - Exemplo de matriz de confusão para classes positiva e negativa

	Previsto Positivo	**Previsto Negativo**
Real Positivo	VP	FN
Real Negativo	FP	VN

Fonte: (Autoria própria)

Precisão, Revocação e *F1-Score*:

Essas métricas são baseadas nos valores da matriz de confusão e são especialmente úteis para classes desbalanceadas.

-Precisão (ou *Precision*): Mede a proporção de previsões corretas para a classe positiva. Pode ser observada na fórmula (1.6)

$$\textbf{Precisão} = \frac{VP}{VP+FP} \quad (1.6)$$

Ajuda a entender se o modelo está cometendo muitos erros ao prever uma amostra como positiva.

-Revocação (ou *Recall*): Mede a capacidade do modelo de identificar todos os exemplos positivos corretamente, sua aplicação é ilustrada na fórmula (1.7).

Revocação = $\frac{VP}{VP+FN}$ (1.7)

Avalia a capacidade do modelo de não perder amostras positivas, o que é importante em cenários onde não perder nenhuma amostra positiva é crítico, como em diagnósticos médicos.

-F1-Score: Combina precisão e revocação em uma única métrica, sendo a média harmônica dessas duas. O *F1-Score*, apresentado na fórmula (1.8), é útil quando se procura um equilíbrio entre precisão e revocação.

F1-Score = 2 * $\frac{Precisão * Revocação}{Precisão + Revocação}$ (1.8)

É uma métrica balanceada que leva em conta tanto a precisão quanto a revocação, útil quando ambas são importantes.

Métricas para Regressão

Em problemas de regressão, o objetivo é prever um valor numérico. Essas métricas medem o quão próximo os valores previstos pelo modelo estão dos valores reais. As métricas mais usadas para esse tipo de previsão são:

-Erro Médio Absoluto ou *Mean Absolute Error* (MAE): Apresentado na fórmula (1.9), o MAE calcula a média dos erros absolutos entre os valores previstos e os valores reais.

MAE = $\frac{1}{n}\sum_{i=1}^{n}|yprevisto - yreal|$ (1.9)

O MAE é simples de entender, pois mede o erro médio em unidades da variável. É eficiente para obter uma medida direta da distância média entre previsões e valores reais.

-Erro Quadrático Médio ou *Mean Squared Error* (MSE): Ilustrado na fórmula (1.10), o MSE calcula a média dos erros ao quadrado. Ele penaliza erros grandes, o que é útil para identificar se o modelo está cometendo grandes erros em algumas previsões.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(yprevisto - yreal)^2 \quad (1.10)$$

-Raiz do Erro Quadrático Médio ou *Root Mean Square Error* (RMSE): Representado na fórmula (1.11), o RMSE é a raiz quadrada do MSE e traz o erro para a mesma unidade dos valores previstos.

$$RMSE = \sqrt{MSE} \quad (1.11)$$

O MSE destaca os erros maiores, sendo útil quando se procura evitar grandes erros de previsão. O RMSE facilita a interpretação dos resultados porque o erro é expresso na mesma unidade dos dados.

-Coeficiente de Determinação (R^2): O R^2, apresentado na fórmula (1.12) mede a proporção de variação nos valores reais que é explicada pelo modelo. Ele varia de 0 a 1, onde valores próximos a 1 indicam que o modelo explica bem a variação dos dados.

$$R^2 = 1 - \frac{\Sigma(yreal - yprevisto)^2}{\Sigma(yreal - \overline{yreal})^2} \quad (1.12)$$

Sendo $\overline{yreal}$ é a média dos valores reais.

O R^2 ajuda a entender o quanto o modelo está capturando das variações dos dados. Ele é útil para saber se o modelo está realmente explicando as mudanças na variável que estamos tentando prever.

4.1.6 Biblioteca Python S*klearn* para desenvolver o modelo KNN

O *Scikit-Learn* é uma biblioteca de ML de código aberto para Python que fornece uma ampla gama de algoritmos e ferramentas voltadas para análise de dados e modelagem preditiva. Desde algoritmos de classificação e regressão até *clustering* e pré-processamento de dados, o *Scikit-Learn* se tornou uma ferramenta indispensável para cientistas de dados e desenvolvedores que atuam no campo da inteligência artificial.

O que é o *Scikit-Learn*?

O *Scikit-Learn*, também conhecido como *sklearn*, é uma biblioteca Python de código aberto desenvolvida sobre bibliotecas fundamentais como *NumPy*, *SciPy* e *Matplotlib*. Ela oferece uma interface simples e eficiente para realizar tarefas comuns de aprendizado de máquina, incluindo classificação, regressão, *clustering* e redução de dimensionalidade.

Além de implementar algoritmos populares, o *Scikit-Learn* fornece ferramentas para o pré-processamento de dados, avaliação e seleção de modelos, entre outras funcionalidades. Com uma documentação extensa e uma comunidade ativa, ele facilita tanto o uso quanto a contribuição de novos recursos para a biblioteca.

Desde o seu lançamento em 2010, o *Scikit-Learn* se destacou no ecossistema de aprendizado de máquina em Python. Ele oferece APIs consistentes e padronizadas, o que permite que os usuários apliquem algoritmos de forma eficiente, utilizando um fluxo de trabalho baseado no padrão ajuste/predição.

O *Scikit-Learn* também se integra facilmente com outras bibliotecas populares do Python, como Matplotlib e *Plotly* para visualização, NumPy para vetorização de *arrays*, biblioteca Pandas para manipulação de *DataFrames* e SciPy para cálculos científicos. Dessa forma, você pode passar diretamente Arrays NumPy e DataFrames Pandas para os algoritmos implementados no Scikit-Learn.

Algoritmos de Aprendizado

O *Scikit-Learn* oferece uma vasta coleção de algoritmos tanto de aprendizado supervisionado quanto não supervisionado, abrangendo diversas áreas, como:

- **Classificação:** Determinar a qual categoria um objeto pertence.
- **Regressão:** Prever valores contínuos associados a um objeto.
- ***Clustering*:** Agrupar automaticamente objetos semelhantes, com modelos como o *K-Means*.
- **Redução de Dimensionalidade:** Reduzir o número de atributos nos dados para fins de sumarização, visualização ou seleção de características, utilizando métodos como a Análise de Componentes Principais (PCA).
- **Seleção de Modelos:** Comparar, validar e escolher os melhores parâmetros e modelos.
- **Pré-processamento:** Utilizado no desenvolvimento do modelo, o pré-processamento, visa extrair e normalizar características, incluindo a definição de atributos em dados de imagem e texto. os conjuntos de dados apresentam valores ausentes com frequência, porém o algoritmo KNN permite estipular esses valores em um processo chamado de imputação de dados ausentes.

Escrito em Python, o *Scikit-Learn* faz uso intensivo do NumPy para operações de álgebra linear e *Arrays*, e alguns de seus algoritmos principais são implementados em *Cython* para otimizar o desempenho.

Com essa robustez e integração, o *Scikit-Learn* apresenta ser uma excelente escolha para realizar análises e experimentos de aprendizado de máquina de forma prática e eficiente (Habbema, 2024).

4.2 Desenvolvendo o modelo KNN para predição de diabetes

Para analisar a eficiência do modelo KNN foi desenvolvida uma implementação, utilizando a linguagem python, para analisar a sua capacidade de definir um possível caso de doença a partir de uma base de dados relacionada a saúde dos pacientes.

4.2.1 Base de dados utilizadas

Para a análise do modelo deste presente trabalho foi utilizado a base de dados **“**Previsão de risco de diabetes em estágio inicial”**,** ela foi coletada por meio de questionários diretos de pacientes do *Sylhet Diabetes Hospital* em Sylhet, Bangladesh e aprovado por um médico. Base de dados desenvolvida no artigo: ***Likelihood Prediction of Diabetes at Early Stage Using Data Mining Techniques*.** By M. M. F. Islam, Rahatara Ferdousi, Sadikur Rahman, Humayra Yasmin Bushra. 2019. *Published in Computer Vision and Machine Intelligence in Medical Image Analysis.*

Características do conjunto de dados: Multivariado

Tarefas associadas: Classificação

Tipo de Recurso: Categórico, Inteiro

Instâncias: 520

Características: 16

O Quadro 2 traz Variáveis da base de dados **“**Previsão de risco de diabetes em estágio inicial”.

Quadro 2 – Variáveis da base de dados **“**Previsão de risco de diabetes em estágio inicial”.

Nome da Variável	Tipo
age	*Integer*
gender	*Categoriacal*
polyuria	*Binary*
polydipsia	*Binary*
sudden_weight_loss	*Binary*
weakness	*Binary*
polyphagia	*Binary*

genital_thrush	*Binary*
visual_blurring	*Binary*
itching	*Binary*
irritability	*Binary*
delayed_healing	*Binary*
partial_paresis	*Binary*
muscle_stiffness	*Binary*
alopecia	*Binary*
obesity	*Binary*
class	*Binary*

Fonte: (Autoria própria)

As variáveis "*age'*, vão de 20 a 65 anos. Variáveis "*gender*" é definido em "*Male*" ou "*Female*". As variáveis "*class*" definem a condição positiva ou negativa para diabetes. As demais variáveis são definidas em "*yes*" ou "*no*".

4.2.2 Desenvolvimento do modelo KNN utilizando Python

A Figura 10 apresenta a importação das bibliotecas utilizadas, sendo cada uma delas explicada a seguir:

Figura 10 – Importação das bibliotecas para desenvolvimento do modelo

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from sklearn.impute import SimpleImputer
from sqlalchemy import create_engine
import matplotlib.pyplot as plt
```

Fonte: (Autoria própria)

- **pandas (pd)**: Fornece suporte essencial para manipular e transformar dados em tabelas (*DataFrames*), o que facilita o pré-processamento.

- **scikit-learn** (sklearn): Biblioteca fundamental para machine learning, que disponibiliza métodos robustos para dividir dados, treinar e avaliar modelos, e realizar pré-processamento.
- **train_test_split:** Divide o conjunto de dados em conjuntos de treino e teste, essencial para avaliar o desempenho do modelo de forma imparcial.
- **KNeighborsClassifier:** Implementa o algoritmo KNN, que classifica dados novos com base na proximidade a exemplos conhecidos.
- **LabelEncoder** e **OneHotEncoder**: Converte dados categóricos para um formato numérico, os algoritmos de ML conseguem interpretar e processar essas informações.
- **ColumnTransformer**: Permite que diferentes pré-processamentos sejam aplicados simultaneamente a colunas específicas, facilitando o tratamento de colunas numéricas e categóricas no mesmo pipeline.
- **SimpleImputer**: Trata valores ausentes com métodos flexíveis de imputação, como preenchimento com o valor médio ou mais frequente.
- **accuracy_score**, **confusion_matrix**, **classification_report**: Fornecem métricas de desempenho do modelo, essenciais para análise e interpretação dos resultados.
- **SQLAlchemy**: Permite uma conexão direta com o banco de dados PostgreSQL para executar consultas e carregar dados no ambiente de análise.
- **Matplotlib**: Ferramenta de visualização que facilita a criação de gráficos para interpretação visual dos resultados.

A Figura 11 apresenta a configura a conexão com o banco de dados no PostgresSQL, que foi utilizado para armazenar os dados obtidos pelo artigo. Para essa configuração foram criadas as variáveis:

Figura 11 – Configurações de conexão com o banco de dados

```
# Configurações de conexão com o PostgreSQL
url_banco_dados =
'postgresql://postgres:senha@localhost:5432/diabetes_data'
engine = create_engine(url_banco_dados)
```

Fonte: (Autoria própria)

- **url_banco_dados**: Define o Localizador Uniforme de Recursos, ou *Uniform Resource Locator* (URL), que especifica o tipo de banco de dados (postgresql), usuário (postgres), senha do banco de dados, endereço (localhost:5432) e nome do banco de dados (diabetes_data).
- **engine**: Cria um objeto engine, que é uma interface para se conectar e consultar dados no banco. Ao estabelecer essa conexão, podemos realizar operações SQL diretamente a partir do Python.

A Figura 12 ilustra a definição da consulta SQL (*Structured Query Language*) para selecionar colunas de interesse, como idade, gênero, sintomas específicos e a variável *class*, que indica a condição de diabetes.

Figura 12 – Consulta SQL para importar os dados do banco de dados

```
# Consulta SQL para obter os dados do banco de dados
consulta = """
SELECT Age AS idade, Gender AS genero, Polyuria, Polydipsia,
sudden_weight_loss, weakness,
       Polyphagia, Genital_thrush, visual_blurring, Itching,
Irritability, delayed_healing,
       partial_paresis, muscle_stiffness, Alopecia, Obesity, class
FROM diabetes_data;
"""
```

Fonte: (Autoria própria)

Cada coluna representa características importantes no diagnóstico de diabetes, fornecendo dados demográficos (idade e gênero) e sintomas comuns.

Colunas Age e Gender são renomeadas para idade e gênero para padronizar em português, facilitando a compreensão do código.

A Figura 13 demonstra dois procedimentos para otimização da aplicação, sendo eles a utilização de:

Figura 13 – Carregamento em um DataFrame e conversão das colunas para minúsculo

```
# Carregar os dados do PostgreSQL em um DataFrame
dados_do_banco = pd.read_sql(consulta, engine)

# Converter todas as colunas para minúsculas
dados_do_banco.columns = dados_do_banco.columns.str.lower()
```

Fonte: (Autoria própria)

- **pd.read_sql**: Executa a consulta SQL e carrega o resultado em um DataFrame pandas, facilitando a manipulação dos dados.

- **dados_do_banco.columns.str.lower()**: Converte os nomes das colunas para letras minúsculas, uniformizando a nomenclatura para evitar possíveis erros com diferenciação entre maiúsculas e minúsculas no código.

A Figura 14 mostra a conversão de algumas colunas para o tipo booleano. Estas colunas representam sintomas presentes ou ausentes, com valores booleanos *True* ou *False*.

Figura 14 – Alteração das colunas para boolean

```
# Definir colunas booleanas
boolean_columns = ['polyuria', 'polydipsia', 'sudden_weight_loss',
'weakness',
                   'polyphagia', 'genital_thrush', 'visual_blurring',
'itching',
                   'irritability', 'delayed_healing', 'partial_paresis',
                   'muscle_stiffness', 'alopecia', 'obesity']
```

Fonte: (Autoria própria)

Por tanto, mapeá-las para 1 e 0 ajuda o modelo a processá-las numericamente, melhorando a compreensão de presença ou ausência de sintomas.

A Figura 15 apresenta a implementação da função de processamento de dados. A função realiza as atividades de:

Figura 15 – Processamento dos dados

```
# Função para preprocessar os dados
def preprocessar_dados(df):
    # Substituir valores booleanos 'true'/'false' para 1/0
    for col in boolean_columns:
        df[col] = df[col].map({True: 1, False: 0})

    # Codificar a coluna 'class' se for a variável alvo
    label_encoder = LabelEncoder()
    df['class'] = label_encoder.fit_transform(df['class'])  # Se 'class'
for categórica

    return df
```

Fonte: (Autoria própria)

- **Loop de Colunas Booleanas**: Converte (*True*/*False)* para (1/0) nas colunas de sintomas, criando um padrão numérico que facilita o aprendizado pelo modelo KNN.
- **Codificação da Variável *class***: Utiliza *LabelEncoder* para transformar a coluna *class* em valores 0 (ausência de diabetes) ou 1 (presença de diabetes), permitindo que o modelo entenda a variável alvo como uma classe binária.

A Figura 16 aborda a aplicação da função de pré-processamento ao *DataFrame*, convertendo todas as colunas booleanas e a variável alvo *class* para o formato adequado.

Figura 16 – Pré-processamento e tratamento de valores nulos

```
# Pré-processar os dados
dados_do_banco = preprocessar_dados(dados_do_banco)

# Verificar valores ausentes
print(dados_do_banco.isnull().sum())

# Se existirem colunas com todos os valores faltantes, podemos removê-las
dados_do_banco = dados_do_banco.dropna(axis=1, how='all')

# Se houver valores faltantes em outras colunas, preencher com a
estratégia desejada, neste caso, é usado a estratégia 'most_frequent'
para as colunas booleanas
imputer = SimpleImputer(strategy='most_frequent')
dados_do_banco[boolean_columns] =
imputer.fit_transform(dados_do_banco[boolean_columns])
```

Fonte: (Autoria própria)

É apresentado também na figura atividades como:

- **Identificação de Valores Nulos**: isnull().sum() exibe a quantidade de valores nulos em cada coluna, auxiliando na identificação de possíveis problemas nos dados.
- **Remoção de Colunas Vazias**: dropna(*axis*=1, *how='all'*) remove colunas com todos os valores nulos. Isso evita que colunas irrelevantes impactem o modelo.
- **Preenchimento de Valores Nulos**: SimpleImputer(strategy='most_frequent') preenche valores ausentes nas colunas booleanas com o valor mais frequente. A escolha dessa estratégia é baseada na hipótese de que o valor mais comum representa um padrão nos dados.

A Figura 17 apresenta o processo de separação dos dados com classe 0 e 1. “X” contém as variáveis independentes, ou seja, todas as características dos pacientes, exceto a variável *class*. “y” contém a variável alvo, *class*, que representa a presença (1) ou ausência (0) de diabetes, sendo o valor que se deseja prever. Depois é feito o filtro para selecionar as classes positivas para diabetes.

Figura 17 – Separação dos dados em classes e filtro de classes positivas

```
# Separar os dados com classe 0 e 1
X = dados_do_banco.drop(columns=['class'])
y = dados_do_banco['class']

# Filtrar os pacientes com classe positiva para diabetes
dados_positivos = dados_do_banco[dados_do_banco['class'] == 1]
```

Fonte: (Autoria própria)

A Figura 18 apresenta as atividades de separação das colunas e pipeline para o pré-processamento. Essas atividades são:

Figura 18 – Separação das colunas e pipeline para o pré-processamento

```
# Separar colunas e preparar o pré-processamento
colunas_numericas = ['idade']
coluna_categorica = ['genero']

# Pipeline para pré-processamento
preprocessor = ColumnTransformer(
    transformers=[
        ('num', SimpleImputer(strategy='mean'), colunas_numericas),
        ('cat', OneHotEncoder(), coluna_categorica)
    ]
)
```

Fonte: (Autoria própria)

- **colunas_numericas e coluna_categorica**: Definem, respectivamente, as colunas de variáveis numéricas (idade) e categóricas (gênero).
- **Pipeline de Pré-processamento *ColumnTransformer***: Aplica um **SimpleImputer()** à coluna idade para substituir valores ausentes pela média, mantendo a consistência dos dados numéricos. Utiliza **OneHotEncoder()** para codificar a coluna gênero (categórica) em variáveis binárias. Isso evita que o modelo interprete a diferença entre valores de gênero como uma hierarquia.

A Figura 19 ilustra a implementação da função **train_test_split()**, que é utilizada para dividir os dados em conjuntos de treino e teste, e embaralhar os dados aleatoriamente separando em duas partes, conforme especificado pelos parâmetros.

Figura 19 – Separar os dados para treino e teste

```
# Dividir os dados em conjunto de treino e teste
X_treino, X_teste, y_treino, y_teste = train_test_split(X, y,
test_size=0.3, random_state=42)
```

Fonte: (Autoria própria)

Essa divisão permite que o modelo seja treinado em uma parte dos dados e avaliado em outra, garantindo uma medição imparcial de desempenho. É utilizado os seguintes parâmetros:

- **X, y:** Representam as variáveis preditoras e a variável alvo, respectivamente.
- **test_size=0.3:** Define que 30% dos dados serão destinados ao conjunto de teste e 70% ao conjunto de treino. Isso proporciona uma quantidade suficiente de dados para avaliar o modelo e ainda manter uma amostra significativa para o treinamento.
- **random_state=42:** Define uma semente para o gerador de números aleatórios, garantindo que a divisão dos dados seja reproduzível em execuções subsequentes.

A Figura 20 aborda a implementação do treinamento do KNN sobre os dados. Foram utilizadas as funções:

Figura 20 – Treinamento do modelo KNN

```
# Treinar o modelo KNN
knn = KneighborsClassifier(n_neighbors=5)
knn.fit(preprocessor.fit_transform(X_treino), y_treino)
```

Fonte: (Autoria própria)

- **KneighborsClassifier()**: Esta classe implementa o algoritmo KNN, onde **n_neighbors=5**, alterar para K=3 pois a acurácia foi melhor, indica que o modelo irá considerar os 5 vizinhos mais próximos para classificar um novo ponto. Esse número é um valor comumente utilizado, mas pode ser ajustado para encontrar um valor ideal com base nos dados.

- **preprocessor.fit_transform(X_treino):** Aplica o pipeline de pré-processamento, ao conjunto de treino, transformando colunas numéricas e categóricas para o formato adequado.
- **knn.fit(...):** Ajusta o modelo KNN aos dados de treino, permitindo que ele “aprenda” com as características dos pacientes, estabelecendo a relação entre os sintomas e a presença ou ausência de diabetes. O KNN não “aprende” no sentido tradicional dos modelos de aprendizado de máquina, pois ele não cria uma função para prever os resultados. Em vez disso, ele simplesmente armazena os dados de treino para usá-los no momento da classificação. No KNN, a etapa de “treinamento” consiste em armazenar as amostras de treino para comparação com novas amostras durante a previsão.

A Figura 21 apresenta a previsão sobre os dados separados para teste. Para isso é utilizado:

Figura 21 – Previsão no conjunto de teste e avaliação do modelo

```
# Fazer previsões nos dados de teste
y_pred_knn = knn.predict(preprocessor.transform(X_teste))
```

Fonte: (Autoria própria)

- **preprocessor.transform(X_teste):** Aplica o pipeline de pré-processamento ao conjunto de teste, garantindo que os dados estejam no mesmo formato do conjunto de treino.
- **knn.predict():** Utiliza o modelo treinado para prever as classes (0 ou 1) no conjunto de teste.

Esse passo gera as previsões do modelo sobre se cada paciente no conjunto de teste possui ou não diabetes, com base nos vizinhos mais próximos no conjunto de treino.

A Figura 22 apresenta as métricas de avaliação da eficácia do modelo. É utilizado:

Figura 22 – Avaliação da eficácia do modelo

```
# Avaliar o modelo com acurácia
accuracy_knn = accuracy_score(y_teste, y_pred_knn)
print(f"Acurácia com KNN: {accuracy_knn}")

# Relatório de classificação
print("\nRelatório de Classificação:")
print(classification_report(y_teste, y_pred_knn))

# Matriz de confusão
print("\nMatriz de Confusão:")
print(confusion_matrix(y_teste, y_pred_knn))
```

Fonte: (Autoria própria)

- **accuracy_score(y_teste, y_pred_knn):** Calcula a acurácia do modelo, ou seja, a proporção de previsões corretas feitas pelo modelo em relação ao total de previsões. A acurácia é uma métrica útil, mas não é suficiente para avaliar totalmente o desempenho, especialmente em dados desbalanceados.
- **classification_report(y_teste, y_pred_knn):** Fornece um relatório detalhado com métricas como precisão, revocação (*recall*), F1-score e suporte para cada classe. Essas métricas permitem uma avaliação mais profunda do modelo:

  -Precisão: Proporção de verdadeiros positivos em relação ao total de previsões positivas.

  -Revocação: Proporção de verdadeiros positivos em relação ao total de elementos reais da classe positiva.

  -F1-score: Média harmônica entre precisão e revocação, útil para dados desbalanceados.
- **confusion_matrix(y_teste, y_pred_knn):** Gera a matriz de confusão, que mostra a quantidade de verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos. Isso permite visualizar erros específicos, como quando o modelo classifica erroneamente um caso de diabetes como ausência da doença (falso negativo).

As métricas ajudam a entender se o modelo é confiável para detecção de diabetes e quais tipos de erros ele comete com mais frequência.

A Figura 23 ilustra a aplicação do gráfico de dispersão. Este gráfico de dispersão (*scatter plot*) exibe a relação entre a idade dos pacientes (eixo X) e a classe prevista

(eixo Y) pelo modelo KNN, colorindo os pontos de acordo com a classe verdadeira (y_teste), para facilitar a comparação visual entre previsões corretas e incorretas.

Figura 23 – Implementação do gráfico de dispersão

```
# Gráfico de dispersão
plt.scatter(X_teste['idade'], y_pred_knn, c=y_teste, cmap='coolwarm',
label='Predições')
plt.xlabel('Idade')
plt.ylabel('Classe Prevista')
plt.title('Dispersão de Idade x Classe Prevista')
plt.colorbar(label='Classe Verdadeira')
plt.legend()
plt.show()
```

Fonte: (Autoria própria)

Para a apresentação do gráfico foi utilizado a função **plt.scatter()**, ela Cria os componentes do gráfico, onde:

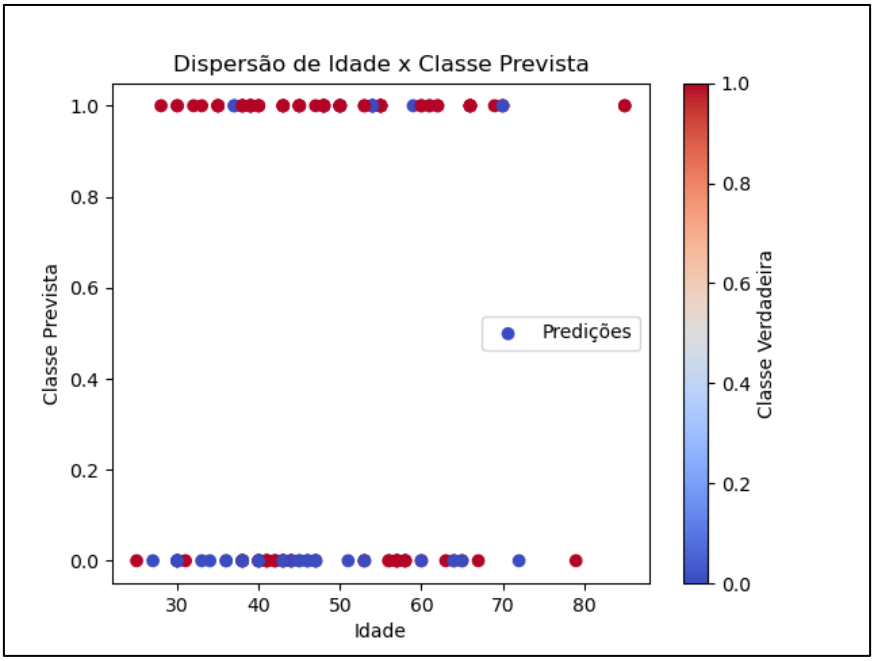
- **X_teste['idade']:** A idade dos pacientes no conjunto de teste, representada no eixo X.
- **y_pred_knn:** A classe prevista para cada paciente, representada no eixo Y.
- **c=y_teste:** Define a cor dos pontos com base na classe verdadeira, permitindo ver onde o modelo acertou ou errou.
- **cmap='coolwarm':** Define o esquema de cores para diferenciar as classes (0 e 1).
- **plt.xlabel, plt.ylabel, plt.title:** Adicionam rótulos e título ao gráfico, tornando-o mais interpretável.
- **plt.colorbar(label='Classe Verdadeira'**): Adiciona uma barra de cores para indicar o mapeamento das classes verdadeiras (0 e 1).
- **plt.legend():** Adiciona uma legenda para indicar que os pontos representam previsões.
- **plt.show():** Exibe o gráfico.

Este gráfico facilita a análise visual do desempenho do modelo, destacando como as previsões estão distribuídas em função da idade e permitindo verificar se há alguma tendência relacionada a esse fator. Isso pode ajudar a identificar possíveis limitações do modelo ou padrões que exigem ajustes.

5 Análise dos resultados obtidos

Após o desenvolvimento do modelo são apresentados os resultados como o gráfico de dispersão apresentado na Figura 24, e as métricas de avaliação da performance do modelo.

Figura 24 – Gráfico de dispersão dos resultados



Fonte: (Autoria própria)

O gráfico de dispersão apresenta a relação entre a idade dos pacientes (eixo X) e a classe prevista (eixo Y) pelo modelo KNN, com a coloração representando a classe verdadeira de cada paciente. As cores variam de azul a vermelho:

- **Azul** representa a classe verdadeira 0 (ausência de diabetes).
- **Vermelho** representa a classe verdadeira 1 (presença de diabetes).

Cada ponto no gráfico indica a previsão do modelo para um paciente específico, onde:

- A **posição no eixo Y** indica a **classe prevista** (0 ou 1) para aquele paciente.
- A cor representa a classe verdadeira, facilitando a visualização de acertos e erros do modelo.

Interpretação dos Resultados no gráfico de dispersão

Pontos Acertados:

Os pontos azuis na parte inferior (classe 0) e os vermelhos na parte superior (classe 1) representam previsões corretas, onde a classe prevista coincide com a classe verdadeira.

A concentração de pontos vermelhos na classe 1 (parte superior) e azuis na classe 0 (parte inferior) indica que o modelo conseguiu identificar corretamente a presença e a ausência de diabetes para vários pacientes.

Pontos Errados:

Os pontos **vermelhos na parte inferior** e **azuis na parte superior** representam previsões incorretas.

A presença desses pontos indica que o modelo cometeu alguns erros ao prever a classe para esses pacientes, classificando erroneamente alguns pacientes com diabetes como saudáveis (falsos negativos) e vice-versa (falsos positivos).

Distribuição de Erros em Relação à Idade:

Observando o eixo X (idade), não há uma relação clara entre a idade e os erros do modelo, pois os pontos estão distribuídos ao longo de todas as faixas etárias. Isso sugere que a idade pode não ser um fator isolado determinante para a classificação de diabetes neste modelo.

O modelo KNN apresentou um desempenho razoável, acertando várias previsões para ambos os grupos (com e sem diabetes), mas ainda cometeu alguns erros.

A distribuição de cores no gráfico, com predominância de pontos vermelhos na parte superior e azuis na parte inferior, mostra que o modelo consegue diferenciar entre as classes, embora com algumas falhas.

A partir desses resultados, o modelo poderia ser um auxílio inicial para prever diabetes com base em características básicas de saúde, mas ainda não é suficientemente confiável para uso clínico isolado. Em uma aplicação prática, seria ideal combinar o modelo com outros métodos de análise e supervisão médica para aumentar a precisão da classificação.

O gráfico fornece uma visão visual intuitiva do desempenho do modelo KNN e destaca áreas onde ele pode melhorar, especialmente em relação à classificação correta de pacientes com diabetes.

5.1 Análise dos resultados do modelo KNN desenvolvido

Foram feitos dois testes com valores diferentes para K, sendo o primeiro com valor de 5 e o segundo com o valor de 3. A seguir é apresentado as análises com os valores usados para K.

5.1.2 Análise dos resultados com K = 5

O primeiro teste do modelo foi utilizado o valor de K =5. A partir disso foram obtidos os seguintes resultados:

Acurácia: 0.7051 (70.51%):

A acurácia indica que o modelo KNN classificou corretamente aproximadamente 70.51% das instâncias no conjunto de teste.

Classe 0 (Não Diabéticos):

- **Precisão: 0.55**: Apenas 55% das previsões positivas para a classe 0 estavam corretas. O que indica que muitas das instâncias previstas como não diabéticas eram, na verdade, diabéticas.
- ***Recall*: 0.78**: O modelo conseguiu identificar 78% dos casos reais de classe 0. Isso é um resultado relativamente bom, mas significa que existe uma taxa significativa de falsos negativos.
- ***F1-Score*: 0.65**: Um F1-score de 0.65 indica que o modelo tem um desempenho equilibrado.

Classe 1 (Diabéticos)

- **Precisão: 0.85**: O modelo tem uma precisão de 85% para a classe 1, o que é um bom resultado. Isso indica que a maioria das instâncias previstas como diabéticas realmente eram diabéticas.
- ***Recall*: 0.67**: O recall de 67% indica que o modelo identificou 67% das instâncias reais de diabéticos. Isso significa que 33% dos casos reais de diabéticos não foram identificados (falsos negativos).
- ***F1-Score*: 0.75**: Um F1-score de 0.75 para a classe 1 mostra um bom equilíbrio entre precisão e recall, embora haja espaço para melhorias.

Média Macro e Ponderada (*Weighted*):

Macro Avg

Precisão: 0.70, ***Recall*: 0.72**, ***F1-Score*: 0.70**

A média macro considera todas as classes igualmente. Isso mostra um desempenho geral moderado do modelo em ambas as classes.

***Weighted* Avg**

Precisão: 0.75, ***Recall*: 0.71**, ***F1-Score*: 0.71**

A média ponderada leva em conta o número de instâncias em cada classe. Neste caso, o modelo é mais preciso na classe 1, refletindo seu desempenho.

Matriz de Confusão

- **Verdadeiros Positivos (VP)**: 68 casos foram corretamente classificados como diabéticos (Classe 1).
- **Verdadeiros Negativos (VN)**: 42 casos foram corretamente classificados como não diabéticos (Classe 0).
- **Falsos Positivos (FP)**: 12 casos foram incorretamente classificados como diabéticos, mas eram não diabéticos.

- **Falsos Negativos (FN)**: 34 casos foram incorretamente classificados como não diabéticos, mas eram diabéticos.

Cálculo de Métricas

Com base nos valores da matriz de confusão, pode-se calcular algumas métricas de desempenho:

Precisão (*Precision*):

- **Para a Classe 0 (Não Diabéticos):**

$$\textit{Precision}\ 0 = \frac{42}{42+12} \approx \mathbf{0.778}$$

- **Para a Classe 1 (Diabéticos):**

$$\textit{Precision}\ 1 = \frac{68}{68+34} \approx \mathbf{0.667}$$

Recall (Sensibilidade):

- **Para a Classe 0 (Não Diabéticos):**

$$\textit{Recall}\ 0 = \frac{42}{42+34} \approx 0.553$$

- **Para a Classe 1 (Diabéticos):**

$$\textit{Recall}\ 1 = \frac{68}{68+12} \approx 0.85$$

F1-Score:

- **Para a Classe 0:**

$$\textit{F1-Score}\ 0 = 2 * \frac{0.778 * 0.553}{0.778 + 0.553} \approx 0.646$$

- **Para a Classe 1:**

$$\textit{F1-Score}\ 1 = 2 * \frac{0.667 * 0.85}{0.667 + 0.85} \approx 0.75$$

Resumo dos resultados:

Classe 0 (Não Diabéticos):

- **Precisão**: 77.8%
- ***Recall***: 55.3%
- ***F1-Score***: 64.6%

Classe 1 (Diabéticos):

- **Precisão**: 66.7%
- ***Recall***: 85%
- ***F1-Score***: 75%

5.1.3 Análise dos resultados com K = 3

O segundo teste do modelo foi utilizado o valor de K =3, que apresentou melhores resultados que o teste anterior. A partir desse teste foram obtidos os seguintes resultados:

Acurácia: 0.7243 (72,43%)

A acurácia do modelo foi de 72,4%. Isso significa que, em média, o modelo classificou corretamente cerca de 72% dos casos no conjunto de teste. A acurácia é uma métrica útil, mas, por si só, pode ser insuficiente, especialmente em cenários onde as classes estão desbalanceadas ou onde um tipo de erro é mais crítico do que outro.

No contexto de diagnóstico de diabetes, pode ser mais importante identificar corretamente os casos positivos (classe 1) para garantir que pessoas com diabetes sejam corretamente diagnosticadas e tratadas. Por isso, outras métricas como precisão, *recall* e f1-score devem ser avaliadas, principalmente para a classe 1.

Classe 0 (Não Diabéticos)

- **Precisão:** A precisão indica a proporção de verdadeiros negativos (TN) em relação a todos os casos classificados como negativos (classe 0). Nesse caso, 58% das vezes em que o modelo previu que uma pessoa não tinha diabetes, ele estava correto.
- ***Recall*:** O recall para a classe 0 representa a capacidade do modelo de identificar corretamente os indivíduos que realmente não têm diabetes. No caso, o modelo conseguiu identificar corretamente 78% das pessoas sem diabetes no conjunto de teste.
- ***F1-Score*:** Esse valor é a média harmônica da precisão e do recall, proporcionando um equilíbrio entre esses dois indicadores. Um f1-score de 0.66 indica que o modelo tem um desempenho razoável para a classe 0, mas poderia ser aprimorado.

Classe 1 (Diabéticos)

- **Precisão:** O modelo obteve uma boa precisão para a classe 1, o que significa que, quando o modelo classificou alguém como diabético, ele estava correto em 86% das vezes. Essa métrica é importante para evitar diagnósticos incorretos em pessoas saudáveis.
- ***Recall*:** O recall para a classe 1 mostra a capacidade do modelo de identificar corretamente os casos de diabetes. Aqui, 70% das pessoas com diabetes foram corretamente identificadas pelo modelo. Esse valor poderia ser mais alto, considerando a importância de detectar casos positivos.
- ***F1-Score:*** O f1-score é relativamente alto para a classe 1, indicando que o modelo está equilibrado entre precisão e recall para identificar casos de diabetes.

Média Macro e Ponderada (*Weighted*):

Macro Avg:

- **Precisão:** 0.72
- ***Recall*:** 0.74
- ***F1-Score*:** 0.71

A média macro calcula a média simples entre as métricas de cada classe. Ela dá igual peso a ambas as classes, independentemente da frequência, mostrando um desempenho geral de 72-74%.

***Weighted* Avg:**

- **Precisão:** 0.76
- ***Recall*:** 0.72
- ***F1-Score*:** 0.73

A média ponderada considera o número de amostras em cada classe. Esse resultado reflete um desempenho ligeiramente melhor, influenciado pela quantidade maior de amostras da classe 1.

Matriz de Confusão

- **Verdadeiros Positivos (VP)**: O modelo identificou corretamente 71 pessoas com diabetes (Classe 1).
- **Verdadeiros Negativos (VN)**: O modelo previu corretamente que 42 pessoas não tinham diabetes (Classe 0).
- **Falsos Positivos (FP)**: Em 12 casos, o modelo previu incorretamente que pessoas não diabéticas tinham diabetes. Esse é um erro que pode gerar ansiedade ou procedimentos médicos desnecessários.
- **Falsos Negativos (FN)**: O modelo classificou erroneamente 31 pessoas com diabetes como não diabéticas. Esse tipo de erro é mais preocupante, pois pode levar à ausência de tratamento em indivíduos que precisam.

Cálculo de Métricas

Com base nos valores da matriz de confusão, pode-se calcular algumas métricas de desempenho:

Precisão (*Precision*):

- **Para a Classe 0 (Não Diabéticos):**

$$\textit{Precision}\ 0 = \frac{42}{42+12} \approx \mathbf{0.78}$$

- **Para a Classe 1 (Diabéticos):**

$$\textit{Precision}\ 1 = \frac{71}{71+31} \approx \mathbf{0.70}$$

Uma precisão de 0.78 para a classe 0 significa que, entre todas as previsões de "Não Diabético" feitas pelo modelo, 78% estavam corretas.

Para a classe 1, a precisão de 0.70 indica que, entre todas as previsões de "Diabético", 70% estavam corretas.

***Recall* (Sensibilidade)**:

- **Para a Classe 0 (Não Diabéticos):**

$$\textit{Recall}\ 0 = \frac{42}{42+31} \approx \mathbf{0.58}$$

- **Para a Classe 1 (Diabéticos):**

$$\textit{Recall}\ 1 = \frac{71}{71+12} \approx \mathbf{0.86}$$

O *recall* de 0.58 para a classe 0 significa que o modelo conseguiu identificar corretamente 58% das pessoas que realmente não têm diabetes.

Para a classe 1, o *recall* de 0.86 significa que o modelo identificou corretamente 86% dos casos de diabetes. Um *recall* alto é desejável, especialmente para a classe de interesse (diabetes).

F1-Score:

- **Para a Classe 0:**

$$\textit{F1-Score}\ 0 = 2 * \frac{0.78 * 0.58}{0.78 + 0.58} \approx \mathbf{0.66}$$

- **Para a Classe 1:**

$$\textit{F1-Score}\ 1 = 2 * \frac{0.70 * 0.86}{0.70 + 0.86} \approx \mathbf{0.77}$$

O *F1-Score* de 0.66 para a classe 0 e 0.77 para a classe 1 indica que o modelo está mais equilibrado para a classe 1. Isso mostra que ele consegue balancear a precisão e o recall melhor para a classe "Diabético".

Resumo dos resultados

Classe 0 (Não Diabéticos):

- **Precisão**: 78%
- ***Recall***: 58%

- ***F1-Score***: 66%

Houve 12 falsos positivos para a classe 0, ou seja, casos em que pessoas sem diabetes foram incorretamente classificadas como diabéticas. Esse tipo de erro pode levar a exames desnecessários.

Classe 1 (Diabéticos):

- **Precisão**: 70%
- ***Recall***: 86%
- ***F1-Score***: 77%

Houve 31 falsos negativos para a classe 1, onde pessoas com diabetes foram classificadas como não diabéticas. Esse tipo de erro deve ser analisado com muito cuidado, pois falsos negativos em diagnósticos clínicos podem acarretar risco à saúde, uma vez que pessoas com a condição da doença podem não receber o devido tratamento.

5.1.4 Conclusão sobre a análise dos resultados

O modelo com K = 5 apresenta uma alta precisão para a classe 1 (diabéticos), mas o recall para essa classe é relativamente baixo (67%). Isso significa que o modelo teve dificuldade em identificar todos os casos positivos, deixando uma parte considerável de pacientes com diabetes sem diagnóstico (falsos negativos).

No teste com K = 3, o modelo apresentou uma melhora geral na acurácia (72.43%) e uma melhoria no recall para a classe 1 (70%), indicando que conseguiu identificar mais casos de diabetes corretamente. Esse valor de *recall* é particularmente importante, pois ajuda a reduzir a quantidade de falsos negativos, que são críticos em um contexto clínico.

O valor de K = 3 obteve uma acurácia melhor do que K = 5 (72.43% contra 70.51%), o que demonstra que um valor menor de K foi mais eficiente para o conjunto de dados testado.

Como o foco principal é diagnosticar corretamente casos de diabetes (classe 1), o *recall* e o *F1-score* dessa classe são especialmente importantes. Com K = 3, o

modelo teve um desempenho superior na detecção de casos de diabetes, com um recall de 70% e *F1-score* de 77% para a classe 1, em comparação com recall de 67% e F1-score de 75% para K = 5.

Ambos os testes mostram um recall semelhante para a classe 0, mas o teste com K = 3 apresentou uma leve melhoria na precisão e no *F1-score*, indicando um desempenho mais equilibrado para ambas as classes.

Para essa aplicação, o teste com K = 3 apresentou melhores resultados gerais e se mostrou mais eficaz em identificar corretamente a classe 1 (diabéticos), o que é crucial para um modelo de diagnóstico clínico. Essa análise demonstra que testar diferentes valores de K permite encontrar o melhor balanço entre precisão e recall, garantindo um modelo mais robusto e confiável.

Em resumo, o valor de K = 3 foi mais adequado para esse conjunto de dados, resultando em uma acurácia mais alta e uma melhor capacidade de identificar corretamente os casos de diabetes (classe 1). Com os testes foi possível perceber a importância da escolha do valor de K e realizar testes com valores diferentes para garantir que o modelo KNN seja eficaz, especialmente em problemas críticos como o diagnóstico de doenças, onde o tipo de erro (falso positivo ou falso negativo) tem um impacto significativo.

5.2 Comparação dos resultados obtidos com os resultados do trabalho relacionado

Para realizar a análise dos resultados e a eficiência do modelo desenvolvido, vale comparar os resultados do estudo mencionado anteriormente “**Predição de diabetes tipo 2 usando algoritmo de vizinho mais próximo K**”. A comparação abrange desde a metodologia e pré-processamento até as métricas finais de desempenho, com foco nas implicações clínicas de cada abordagem.

Objetivo e Algoritmo

Tanto a implementação deste trabalho quanto a do estudo anterior usaram o KNN com o objetivo de prever o risco de diabetes. Ambos os estudos visam a detecção precoce da doença, uma vez que uma classificação precisa dos pacientes em grupos de alto e baixo risco pode melhorar significativamente o tratamento e as intervenções preventivas.

O estudo mencionado, explorou uma ampla gama de valores para “K”, variando de 5 a 40, especialmente para avaliar o comportamento do KNN em conjuntos de dados maiores e com mais variáveis.

A implementação do modelo deste trabalho testou especificamente os valores de K=5 e K=3, priorizando um valor menor de “K” devido à menor dimensionalidade do conjunto de dados, explorando como o modelo lida com uma base de dados mais restrita.

Conjunto de Dados e Atributos

Ambos os estudos utilizaram dados que incluíam atributos clínicos e de estilo de vida associados ao diabetes. Contudo, houve diferenças significativas no tamanho e no detalhamento dos dados.

O estudo utilizou um conjunto de dados mais complexo, com um número maior de variáveis e maior diversidade de registros. Por isso, valores mais altos de “K” foram aplicados para permitir uma melhor suavização e reduzir o risco de *overfitting* em um conjunto de dados de alta dimensionalidade.

A implementação deste trabalho com uma base de dados menor e com atributos mais específicos, justificando o uso de valores mais baixos de “K” para capturar uma maior precisão nos vizinhos mais próximos, o que poderia aumentar a especificidade na classificação.

Pré-Processamento dos Dados

Ambos os estudos passaram por etapas de pré-processamento, essenciais para maximizar o desempenho do modelo KNN, que é sensível à escala dos dados.

O estudo realizou limpeza de dados, normalização e técnicas adicionais de seleção de atributos, como redução de dimensionalidade com *Principal Component Analysis* (PCA) em alguns conjuntos, para lidar com a alta dimensionalidade e complexidade dos dados.

A implementação focou em transformar valores booleanos e categorizar variáveis, sem aplicar PCA devido ao menor número de atributos. O pré-processamento incluiu a transformação de variáveis categóricas e preenchimento de dados faltantes com a estratégia "***most_frequent***", adaptada ao menor tamanho do conjunto de dados.

Comparação dos Resultados

A seguir é analisado as métricas de desempenho para cada valor de K nas duas implementações, com foco nas diferenças entre a abordagem do estudo anterior e a implementação atual.

Melhores resultados do estudo

O estudo mencionado obteve os melhores resultados no Conjunto de Dados 2 (Diabetes Dataset, 769 amostras e 9 atributos), com o valor de K = 21. Nesse cenário:

A combinação de menos e um tamanho de amostra intermediário permitiu que o KNN aproveitasse as relações entre os dados sem ser sobrecarregado por ruídos ou alta dimensionalidade.

O modelo apresentou valores significativos de verdadeiros positivos (259) e verdadeiros negativos (92), indicando uma classificação eficaz dos pacientes.

Este conjunto de dados apresentou melhor equilíbrio entre simplicidade e riqueza de informações.

O valor K = 21 forneceu um bom ajuste ao modelo, evitando tanto subajustes (*underfitting*) quanto sobreajustes (*overfitting*).

Apesar disso, os resultados gerais do estudo indicam que o desempenho do KNN tende a ser melhor com conjuntos de dados menores e menos complexos, como o Conjunto de Dados 3, que alcançou seu pico de precisão com K = 5.

Melhores resultados do modelo desenvolvido

O modelo desenvolvido para este trabalho obteve seus melhores resultados com o valor de K = 3. Nesse cenário, várias métricas indicaram a eficácia do modelo KNN na classificação de pacientes em classes de risco para diabetes. Os detalhes incluem:

A combinação de um valor baixo para K (K = 3) e o conjunto de dados empregado permitiu que o modelo aproveitasse as relações mais diretas entre os dados, minimizando o impacto de ruídos ou alta complexidade.

O modelo apresentou resultados significativos para a Classe 1 (diabéticos), com precisão de 86% e um *F1-Score* de 77%, destacando sua eficiência em identificar corretamente pacientes com diabetes.

A matriz de confusão demonstrou 71 verdadeiros positivos e 42 verdadeiros negativos, indicando boa capacidade de classificação geral.

Este cenário apresentou um equilíbrio interessante entre simplicidade e robustez nas informações, onde K = 3 proporcionou um ajuste que foi evitado tanto subajustes (*underfitting*) quanto sobreajustes (*overfitting*).

Apesar dos bons resultados, foi identificado que a maior limitação do modelo foram os 31 falsos negativos, que representam pacientes com diabetes classificados incorretamente como não diabéticos. Esse aspecto ressalta a importância de melhorias nos processos de ajuste e pré-processamento de dados para reduzir erros críticos em aplicações clínicas.

Conclusão do desempenho KNN para predição de doenças

Com base nos melhores resultados obtidos nos dois estudos analisados, observa-se que o desempenho do algoritmo KNN na predição de diabetes tipo é fortemente influenciado pela complexidade do conjunto de dados e pela escolha do valor de K. Ambos os estudos apontam que conjuntos de dados menores e com atributos bem selecionados permitem que o modelo capture padrões mais relevantes, resultando em melhores métricas de desempenho.

Ambos os estudos reforçam que o KNN é uma abordagem viável para a predição de diabetes, desde que:

- **Os dados sejam pré-processados com qualidade** – A normalização e a remoção de outliers são fundamentais para maximizar a precisão do modelo.
- **O valor de K seja cuidadosamente ajustado** – Valores intermediários (como K = 21) equilibram sensibilidade e especificidade, enquanto valores menores (como K = 3) são mais adequados para padrões locais.
- **Conjuntos de dados compactos e informativos** tendem a gerar melhores resultados, reduzindo a interferência de ruídos e de alta dimensionalidade.

Por fim, embora o KNN mostre bom potencial, os resultados destacam a necessidade de explorar métodos complementares, como algoritmos mais robustos (*Random Forest* ou SVM), especialmente para reduzir falsos negativos em aplicações clínicas críticas. Essas melhorias podem tornar o modelo mais adequado para cenários reais, oferecendo maior confiabilidade no diagnóstico precoce de diabetes.

6 CONCLUSÃO

A questão de pesquisa que norteia este trabalho foi: **Como o uso do Algoritmo do Vizinho Mais Próximo ou *K-Nearest Neighbors* (KNN) pode ser eficiente na predição de doenças?**

O estudo permitiu concluir que o uso do modelo de KNN pode ser uma abordagem eficiente para a predição de doenças. A análise realizada ao longo do trabalho demonstrou que, com o pré-processamento adequado dos dados e a escolha criteriosa do parâmetro K, o modelo é capaz de identificar padrões relevantes, oferecendo predições confiáveis para o diagnóstico.

Além disso, o estudo reforçou que, embora o KNN apresente vantagens significativas, como simplicidade de implementação e versatilidade para classificação e regressão, ele também possui limitações. A sensibilidade do modelo à dimensionalidade dos dados e ao desbalanceamento de classes pode comprometer sua eficácia, exigindo cuidados específicos. Técnicas de normalização e seleção de características foram aplicadas para mitigar esses desafios, mas a dependência do cálculo de distâncias em grandes volumes de dados ainda se apresenta como uma barreira em contextos mais complexos. Esses fatores reforçam a importância de considerar cenários de aplicação cuidadosamente ao optar por esse algoritmo em problemas clínicos.

Além disso, conclui-se que a integração da inteligência artificial no campo da saúde vai além da aplicação de modelos matemáticos. A colaboração com profissionais médicos é indispensável para validar e contextualizar os resultados obtidos, assegurando que eles contribuam para uma prática clínica mais precisa e segura. A participação dos médicos é especialmente relevante na interpretação dos resultados, na definição de critérios diagnósticos complementares e na adaptação dos modelos às especificidades dos pacientes. Essa sinergia entre tecnologia e expertise médica pode melhorar não apenas a precisão dos diagnósticos, mas também a aceitação e a confiabilidade das soluções tecnológicas entre os profissionais de saúde.

A segurança dos dados também foi destacada como um aspecto fundamental na utilização de algoritmos de inteligência artificial em saúde. A coleta, armazenamento e processamento de dados sensíveis requerem protocolos rigorosos

para garantir a privacidade e a conformidade com regulamentações legais. O estudo abordou a importância de tecnologias como criptografia homomórfica e aprendizado federado, que permitem a análise de dados de maneira descentralizada e segura, preservando a confidencialidade das informações médicas. Essas medidas são essenciais para aumentar a confiança no uso de sistemas baseados em inteligência artificial, tanto por parte dos profissionais quanto dos pacientes.

Com base nos resultados obtidos, conclui-se que o modelo KNN tem potencial para se tornar uma ferramenta prática e eficiente no campo da saúde. Entretanto, a aplicação dessa tecnologia deve ser acompanhada por profissionais da saíde com alto rigor técnico, responsabilidade ética e esforços para integrar as soluções propostas aos sistemas de saúde e às práticas médicas já consolidadas. Assim, o avanço do uso da inteligência artificial em diagnósticos poderá contribuir para uma medicina mais precisa, proativa e personalizada.

Para a continuidade deste estudo, sugere-se os seguintes trabalhos futuros:

- Aplicar o modelo KNN a outras bases de dados relacionadas à saúde: Ao utilizar bases de dados com características diferentes ou informações mais amplas sobre outras doenças, será possível avaliar a capacidade de generalização e robustez do modelo, além de identificar adaptações necessárias para novos contextos.

- Testar a base de dados utilizada com outros modelos de Machine Learning: O uso de algoritmos alternativos, como redes neurais, Random Forest ou regressão logística, permitirá uma análise comparativa para avaliar o desempenho do KNN em relação a outras técnicas preditivas e identificar cenários nos quais o modelo proposto pode ser mais vantajoso.

- Explorar combinações de técnicas e otimização de hiperparâmetros: A integração de abordagens híbridas, associando o KNN a técnicas como PCA (Análise de Componentes Principais) ou modelos baseados em ensamble, pode ampliar sua capacidade preditiva e eficiência. Além disso, a automatização da escolha de K e outras variáveis poderia contribuir para aumentar a precisão e reduzir o tempo de configuração manual.

REFERENCIAS

ANTÓNIO, M.; PEREIRA, P. INSTITUTO POLITÉCNICO DE LISBOA. **Previsão de volumes pulmonares não mobilizáveis com base em parâmetros espirométricos**. [s.l: s.n.]. Disponível em: <https://repositorio.ipl.pt/bitstream/10400.21/16535/1/MarcoPereira_46797%20_MEB.pdf>. Acesso em: 3 set. 2024.

BENTO, P. et al. **Classificação Automática de Termogramas do Pé Diabético Usando Técnicas de Machine Learning**. [s.l: s.n.]. Disponível em: <https://repositorio.utad.pt/server/api/core/bitstreams/7a7a4866-b881-432f-8207-83053c163300/content>. Acesso em: 3 set. 2024.

BZOVSKY, S., Phillips, MR, Guymer, RH *et al.* **Guia do clínico para interpretar uma análise de regressão**. *Eye* **36** , 1715–1717 (2022). https://doi.org/10.1038/s41433-022-01949-z

Cinnecta**. Como otimizar o KNN?** Disponível em: <https://cinnecta.com/conteudos/knn-k-nearest-neighbors/#3>. Acesso em: 3 nov. 2024.

DAMACENO, L. **Regressão Linear?** Disponível em: <https://medium.com/@lauradamaceno/regress%C3%A3o-linear-6a7f247c3e29>.

Elastic. **O que é k vizinhos mais próximos (kNN)? | Um guia abrangente de k vizinhos mais próximos**. Disponível em: <https://www.elastic.co/pt/what-is/knn>.

GIL, Antônio Carlos. **Como Elaborar Projetos de Pesquisa**. 6. ed. São Paulo: Editora Atlas Ltda, 2017.

GOMES, A. M. **Inteligência Artificial com Python: Por que usar Python para trabalhar com IA?** Disponível em: <https://hub.asimov.academy/tutorial/inteligencia-artificial-com-python-por-que-usar-python-para-trabalhar-com-ia/#o-que-e-python%3f>. Acesso em: 6 nov. 2024.

HABBEMA, H. **Introdução ao Scikit-Learn**. Disponível em: <https://medium.com/@habbema/introdu%C3%A7%C3%A3o-ao-scikit-learn-f00b7201dbf7>.

IBM. **What Is the k-nearest Neighbors algorithm? | IBM**. Disponível em: <https://www.ibm.com/topics/knn>.

KAAN ERDEN. **Linear Regression, Logistic Regression and KNN: Fundamental Approaches in Machine Learning**. Disponível em: <https://medium.com/@kaanerdenn/linear-regression-logistic-regression-and-knn-fundamental-approaches-in-machine-learning-7d23647ce989>. Acesso em: 15 nov. 2024.

KIM, E. **KNN(K-Nearest Neighbors) 알고리즘 개념정리**. Disponível em: <https://blog.eunsukim.me/posts/understanding-KNN>. Acesso em: 4 nov. 2024.

LOPES, A. et al. **Aprendizado de Máquina Aplicado à Predição de Doenças Cardiometabólicas com Utilização de Indicadores Metabólicos e Comportamentais de Risco à Saúde**. Anais do XII Computer on the Beach **- COTB '21**, 29 abr. 2021.

MAFFAZIOLI, U. **Machine Learning | Algoritmo kNN**. Disponível em: <https://medium.com/@ulissesmaffa/machine-learning-algoritmo-knn-26eb7b702c37>.

PARK, C.-W. et al. **Artificial Intelligence in Health Care: Current Applications and Issues. Journal of Korean Medical Science**, v. 35, n. 42, 27 out. 2020.

RAUL SIDNEI WAZLAWICK. **Metodologia de pesquisa para ciência da computação**. 2° edição. Rio De Janeiro: Elsevier, 2014.

Santos, Vinicius (2024). **Comparison and Selection of Machine Learn-ing Algorithms for Diabetes Prediction: An Ex-ploratory Quantitative Study Based on Medi-cal Data Analysis**. Disponível em: https://www.researchgate.net/publication/379807553_Comparison_and_Selection_of_Machine_Learn-ing_Algorithms_for_Diabetes_Prediction_An_Ex-ploratory_Quantitative_Study_Based_on_Medi-cal_Data_Analysis

S SURIYA; J MUTHUKRISHNAN. **Type 2 Diabetes Prediction using K-Nearest Neighbor Algorithm**. Journal of Trends in Computer Science and Smart Technology, v. 5, n. 2, p. 190–205, 1 jun. 2023.

SPADINNI, Segovia Allan. **O que é Inteligência Artificial? Como funciona uma IA, quais os tipos e exemplos**. Disponível em: https://www.alura.com.br/artigos/inteligencia-artificial-ia?utm_term=&utm_campaign=%5BSearch%5D+%5BPerformance%5D+-+Dynamic+Search+Ads+-+Artigos+e+Conte%C3%BAdos&utm_source=adwords&utm_medium=ppc&hsa_acc=7964138385&hsa_cam=11384329873&hsa_grp=111087461203&hsa_ad=687448474447&hsa_src=g&hsa_tgt=dsa-2276348409503&hsa_kw=&hsa_mt=&hsa_net=adwords&hsaver=3&gad_source=1&gclid=Cj0KCQjwwMqvBhCtARIsAIXsZpYMIBIzo3R0uUIvqTFJSYd6C6Kz6deM9Po5hD9fN93QMkVfFeKR6PAaAucwEALw_wcB. Acesso em 19 mar. 2024

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS GABINETE DO REITOR

Av. Universitária, 1069 Caixa Postal 86 • CEP 74605• Setor Universitário -010
Goiânia reitoria@pucgoias.edu.br• Goiás • Brasil Fone: (62) 3946.1000 www.pucgoias.edu.br •

RESOLUÇÃO n° 038/2020 – CEPE

ANEXO I

APÊNDICE ao TCC

Termo de autorização de publicação de produção acadêmica

O estudante Daniel Pires Torres do Curso de Engenharia da Computação, matrícula 20201003300757, telefone: 66 999105051 e-mail danptorres@hotmail.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do Autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado ALGORITMO DO VIZINHO MAIS PRÓXIMO PARA PREDIÇÃO DE DOENÇAS, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto(PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 13 de DEZEMBRO de 2024.

Assinatura do autor_ gov.br Documento assinado digitalmente DANIEL PIRES TORRES Data: 13/12/2024 22:28:31-0300 Verifique em https://validar.iti.gov.br

Nome completo do autor: Daniel Pires Torres _____

Assinatura do professor–orientador: _ gov.br Documento assinado digitalmente SOLANGE DA SILVA Data: 16/12/2024 21:39:24-0300 Verifique em https://validar.iti.gov.br _____

Nome completo do professor-orientador: __Solange da Silva_