Centro Federal de Educação Tecnológica de Minas Gerais
Professor: Tiago Alves — Disciplina: Inteligência Artificial
Aluno: Luan Gonçalves Santos — Matrícula: 20213004695
Aluno: Pedro Henrique Pires Dias — Matrícula: 20203011622

Divinópolis, 12 de dezembro de 2024.

Trabalho 03

Atividade Avançada: Árvores de Decisão, KNN e SVM em Python

1 Introdução

O aprendizado de máquina tem se destacado como uma área fundamental da Inteligência Artificial, permitindo a criação de modelos capazes de identificar padrões e tomar decisões com base em dados. Entre as técnicas de aprendizado supervisionado, destacam-se os algoritmos de **Árvore de Decisão**, **K-Nearest Neighbors (KNN)** e **Support Vector Machine (SVM)**, que possuem aplicações diversas em áreas como saúde, finanças e reconhecimento de padrões.

Este relatório tem como objetivo implementar e comparar o desempenho desses três algoritmos aplicados ao conjunto de dados **Predict Student Performance Dataset**, disponível no *Kaggle* [1]. Esse dataset contém informações sobre o desempenho de estudantes, permitindo a construção de modelos que possam prever seu rendimento acadêmico com base em fatores diversos.

Além disso, será desenvolvido um modelo de Árvore de Decisão baseado em perguntas para auxiliar na escolha de hobbies ou carreiras, aplicando os princípios dessa técnica de aprendizado de máquina.

A partir da análise dos resultados obtidos, será possível avaliar a acurácia de cada modelo, identificando suas vantagens e limitações, bem como discutir possíveis melhorias e aplicações futuras.

1.1 Definição do Problema

A previsão do desempenho acadêmico dos alunos é um desafio enfrentado por instituições de ensino em todo o mundo. Compreender os fatores que influenciam o rendimento

estudantil pode auxiliar educadores na tomada de decisões estratégicas para melhorar a aprendizagem e reduzir taxas de reprovação.

Neste estudo, utilizamos o conjunto de dados **Predict Student Performance Dataset**, disponível no repositório *Kaggle* [1]. O dataset contém variáveis relevantes, como *Socioeconomic Score* (pontuação socioeconômica), *Study Hours* (horas de estudo), *Sleep Hours* (horas de sono) e *Attendance (%)* (frequência escolar). Com base nessas informações, buscamos prever se um aluno será **aprovado** ou **reprovado**, considerando um limiar de nota igual a 60.

2 Descrição dos Algoritmos Implementados

Neste estudo, foram utilizados três algoritmos de aprendizado supervisionado para a previsão do desempenho acadêmico dos alunos: **Árvore de Decisão**, **K-Nearest Neighbors (KNN)** e **Support Vector Machine (SVM)**. Cada um desses métodos possui características específicas que influenciam sua precisão e aplicabilidade em diferentes tipos de problemas.

2.1 Árvores de Decisão

A **Árvore de Decisão** é um algoritmo de classificação que utiliza uma estrutura hierárquica de decisões para categorizar os dados. Esse modelo divide iterativamente o conjunto de dados em subconjuntos menores, baseando-se no atributo que melhor separa as classes, segundo uma métrica como *entropia* ou *índice de Gini*.

No presente trabalho, utilizamos a biblioteca *scikit-learn* para implementar a Árvore de Decisão, configurando o critério de divisão como *entropy*, o que permite medir a pureza dos nós gerados. O modelo foi treinado com 80% dos dados e testado nos 20% restantes.

As principais vantagens da Árvore de Decisão incluem: interpretabilidade, pois sua estrutura é semelhante a um conjunto de regras lógicas e também capacidade de lidar com dados categóricos e numéricos. Por outro lado, esse modelo pode sofrer de sobreajuste (*overfitting*) caso não seja podado corretamente.

2.2 K-Nearest Neighbors (KNN)

O algoritmo **K-Nearest Neighbors (KNN)** é um método baseado em instâncias que classifica um novo ponto considerando a maioria dos rótulos de seus vizinhos mais próximos. O número de vizinhos (K) é um hiperparâmetro crucial, pois afeta diretamente o desempenho do modelo.

No presente estudo, utilizamos K=*5*, um valor comumente adotado para evitar oscilações excessivas nos resultados, e a distância entre os pontos foi calculada usando a métrica Euclidiana. O algoritmo KNN apresenta algumas vantagens, como sua simplicidade de

implementação e interpretação, além de não assumir nenhuma distribuição prévia dos dados. No entanto, também possui limitações, sendo computacionalmente custoso para grandes volumes de dados e sensível tanto à escolha do valor de K quanto à escala das variáveis, o que pode impactar seu desempenho em diferentes conjuntos de dados.

2.3 Support Vector Machine (SVM)

O algoritmo **Support Vector Machine (SVM)** busca encontrar um hiperplano que melhor separa as classes, maximizando a margem entre os pontos mais próximos de cada classe, conhecidos como *vetores de suporte*. No presente estudo, utilizamos o **kernel linear**, pois o problema de classificação dos alunos se mostrou linearmente separável.

O algoritmo SVM apresenta como principais vantagens sua capacidade de generalização eficiente para conjuntos de dados de alta dimensionalidade e sua menor suscetibilidade ao *overfitting* em comparação com modelos mais complexos. No entanto, possui algumas limitações, como o alto custo computacional para grandes volumes de dados e a sensibilidade à escolha do kernel e dos hiperparâmetros, o que pode impactar seu desempenho dependendo do problema analisado.

3 Critérios de Avaliação

Para comparar o desempenho dos algoritmos implementados, utilizamos as seguintes métricas de avaliação:

- **Acurácia**: Mede a proporção de previsões corretas sobre o total de previsões realizadas. É expressa pela equação:

$$Acuracia = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

  onde TP representa os verdadeiros positivos, TN os verdadeiros negativos, FP os falsos positivos e FN os falsos negativos.

- **Relatório de Classificação**: Além da acurácia, utilizamos a função classification_report da biblioteca *scikit-learn* para obter informações detalhadas sobre as métricas de *precisão*, *recall* e *F1-score* de cada modelo.

- **Divisão Treino/Teste**: Os dados foram divididos em **80% para treinamento** e **20% para teste** para garantir uma boa generalização dos modelos.

Os resultados mostram que a **Árvore de Decisão** apresentou a maior acurácia (**99,64%**), seguida pelo **SVM** (**98,20%**) e pelo **KNN** (**97,12%**). Essas métricas serão analisadas em detalhes na seção de resultados.

4 Ferramentas Utilizadas

Para a implementação dos algoritmos apresentados neste relatório, foram utilizadas algumas ferramentas que facilitaram o desenvolvimento e a análise dos modelos. As principais tecnologias incluem **Python 3**, **Visual Studio Code** e **WSL Ubuntu**.

- **Python 3**: A linguagem de programação Python foi escolhida por recomendação do professor, possivelmente por sua ampla utilização na área de aprendizado de máquina, oferecendo diversas bibliotecas especializadas, como *scikit-learn*, *pandas* e *matplotlib*, que foram cruciais para a implementação e análise dos modelos. [3].
- **Visual Studio Code**: O Visual Studio Code é um editor de código-fonte desenvolvido pela Microsoft para Windows, Linux e macOS. Foi utilizado neste projeto devido à sua leveza, suporte a extensões e integração facilitada com ambientes de desenvolvimento baseados em Python. [2].
- **WSL (Windows Subsystem for Linux)**: O WSL permite a execução de um ambiente Linux dentro do Windows sem a necessidade de máquinas virtuais ou dual boot. Essa funcionalidade é essencial para o desenvolvimento em um ambiente GNU/Linux, garantindo compatibilidade com bibliotecas e ferramentas utilizadas no aprendizado de máquina. [4].

5 Instruções para execução

Tabela 1: Comandos úteis para compilar e executar o programa de computador

Comando	**Função**
python3 filename.py	Executa o programa após a realização da compilação

6 Resultados das Medições de Desempenho

Os modelos foram avaliados utilizando a métrica de **acurácia**, considerando uma divisão de dados em **80% para treinamento** e **20% para teste**. A Tabela 2 apresenta os resultados obtidos:

Tabela 2: Acurácia dos modelos avaliados.

Modelo	**Acurácia**
Árvore de Decisão	99,64%
KNN (K=5)	97,12%
SVM (Kernel Linear)	98,20%

Os resultados mostram que a **Árvore de Decisão** apresentou o melhor desempenho, atingindo uma acurácia de **99,64%**. O **SVM** obteve um resultado próximo, com **98,20%**, enquanto o **KNN** apresentou a menor acurácia, atingindo **97,12%**.

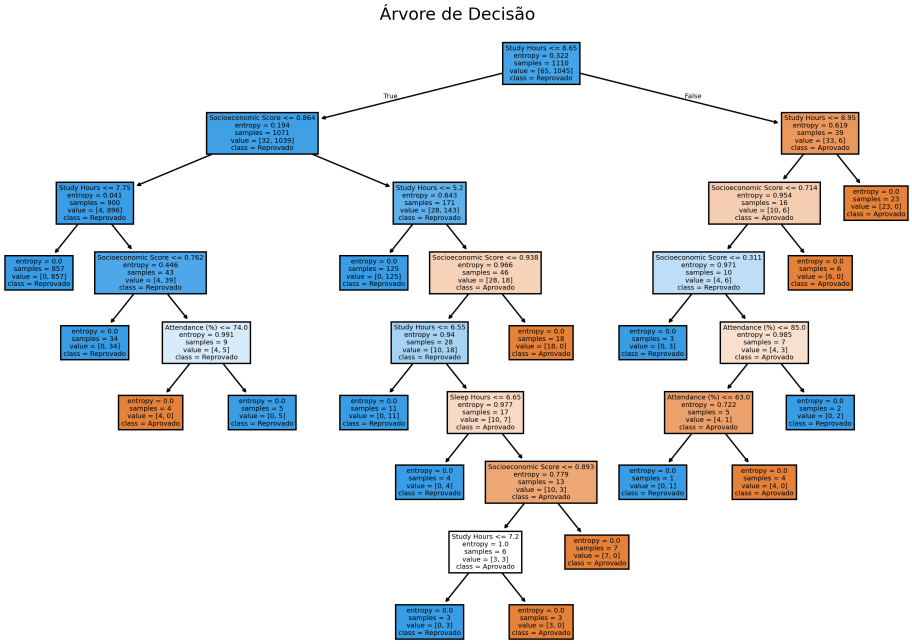

Figura 1: Árvore de decisão gerada pelo modelo.

Embora a Árvore de Decisão tenha apresentado uma alta taxa de acerto, isso pode indicar um possível sobreajuste (*overfitting*), pois o modelo pode estar memorizando padrões específicos do conjunto de treinamento, em vez de generalizar bem para novos dados. Para avaliar melhor a robustez dos modelos, métricas como *precision*, *recall* e *F1-score* devem ser exploradas.

O código desenvolvido para este trabalho está disponível publicamente no repositório GitHub e pode ser acessado através do seguinte link: https://github.com/peudias/decision-tree.

7 Conclusão e Trabalhos Futuros

Neste trabalho, avaliamos três algoritmos de aprendizado supervisionado para prever o desempenho acadêmico de estudantes com base em fatores socioeconômicos e hábitos

de estudo. Os resultados mostraram que a **Árvore de Decisão** teve a melhor acurácia, mas sua alta taxa de acerto pode indicar um sobreajuste. O **SVM** apresentou um bom equilíbrio entre simplicidade e desempenho, enquanto o **KNN** teve a menor acurácia, mas ainda assim um desempenho competitivo.

Apesar dos bons resultados, algumas limitações devem ser consideradas. A métrica de acurácia pode não ser suficiente para avaliar corretamente os modelos, principalmente se houver um desbalanceamento entre classes. Além disso, a base de dados utilizada contém apenas um conjunto limitado de variáveis, o que pode influenciar a capacidade preditiva dos modelos.

Para trabalhos futuros, este estudo pode ser aprimorado com a análise de métricas como *precision*, *recall* e *F1-score*, além da otimização dos modelos por meio de *pruning* na Árvore de Decisão e ajuste de hiperparâmetros no KNN e SVM. A inclusão de novas variáveis, como nível de estresse e ambiente de estudo, pode aumentar a robustez dos modelos, enquanto a experimentação com algoritmos mais avançados, como *Random Forest* e *Redes Neurais*, pode oferecer melhores resultados. Por fim, o desenvolvimento de um sistema preditivo interativo permitiria que alunos e educadores utilizassem os modelos para monitorar e melhorar o desempenho acadêmico.

Referências

[1] *Predict Student Performance Dataset*. Disponível em: `https://www.kaggle.com/datasets/stealthtechnologies/predict-student-performance-dataset`. Acesso em: fevereiro de 2025.

[2] Microsoft. Download for free of Visual Studio Code (Version 1.84). [S.l.: s.n., s.d.]. Disponível em: https://code.visualstudio.com/. Acesso em: 14 jan. 2024.

[3] PYTHON SOFTWARE FOUNDATION. Python Language Site: Documentation, 2020. Página de documentação. Disponível em: https://www.python.org/doc/. Acesso em: 14 de jan. de 2025.

[4] GRACIELLY, J. WSL 2 - A solução para rodar Linux dentro do Windows 10 - Root #08 [Vídeo]. YouTube, 2021. Disponível em:https://www.youtube.com/watch?v=hd6lxt5iVsg. Acesso em: 18 nov. 2023.