Centro Federal de Educação Tecnológica de Minas Gerais
Engenharia de Computação - Inteligência Artificial

Trabalho 2 de IA: Algoritmos de Aprendizagem

Pedro Augusto Gontijo Moura
Jader Oliveira Silva

Prof. Thiago Alves de Oliveira

Divinópolis/MG
Dezembro de 2025

Resumo

Este relatório apresenta a implementação e os resultados das quatro partes requeridas no Trabalho 2 da disciplina de Inteligência Artificial: (i) árvore de decisão manual; (ii) estudo comparativo de métodos supervisionados (KNN, SVM e Árvore de decisão) sobre conjunto de dados selecionado; (iii) implementação de um Algoritmo Genético aplicado a um problema de otimização; e (iv) implementação e comparação de um método de enxame e de um algoritmo do paradigma imune. Para cada parte são descritas a modelagem, os parâmetros utilizados, as métricas avaliadas e as principais conclusões. O repositório com o código e instruções de reprodução está referenciado nas referências.

Parte 1 — Árvore de Decisão Manual

Introdução

Foi desenvolvida uma árvore de decisão construída manualmente, cujo objetivo é recomendar um hobby ao usuário com base em suas respostas a uma sequência de perguntas binárias (sim/não). A proposta consiste em estruturar logicamente um processo de escolha, definindo um conjunto de questões que direcionam o usuário ao longo de diferentes caminhos até chegar a uma recomendação final.

A árvore foi planejada de forma hierárquica, iniciando com uma pergunta ampla — relacionada à preferência por atividades ao ar livre — e ramificando progressivamente para perguntas mais específicas, como interesse em atividades físicas, tecnologia, criatividade ou relaxamento. Cada resposta leva o usuário a um novo nó da árvore até que um nó folha seja alcançado, contendo uma lista de hobbies adequados ao perfil identificado. Toda a estrutura da árvore foi representada em formato JSON, permitindo fácil leitura, manutenção e expansão.

Árvore

A árvore foi construída manualmente e organizada de forma hierárquica a partir de dez perguntas, onde cada nó interno representa uma decisão binária e cada nó folha contém um conjunto final de hobbies recomendados.

A pergunta inicial escolhida foi: *"Você gosta de atividades ao ar livre?"*. Optou-se por essa pergunta porque ela gera uma divisão clara e bem definida entre dois grandes perfis de hobbies: os que dependem de ambientes externos (esportes, trilhas, exploração) e os que ocorrem predominantemente em ambientes internos (mídia, tecnologia, artes). Essa separação inicial reduz ambiguidades e permite que os ramos seguintes sigam caminhos temáticos mais organizados.

A partir dessa raiz, a árvore ramifica-se em decisões relacionadas a atividade física, competitividade, relaxamento, tecnologia, criatividade e outras características pessoais. Cada sequência de respostas leva a um nó folha com sugestões específicas, garantindo um fluxo simples, coerente e sem repetição de perguntas.

Diagrama

Esse diagrama permite visualizar claramente a lógica de navegação implementada no sistema, bem como a coerência entre as respostas fornecidas pelo usuário e as sugestões

finais de hobbies apresentadas. A estrutura também evidencia que todas as perguntas são distintas e que cada caminho da árvore termina em um conjunto de recomendações específico.

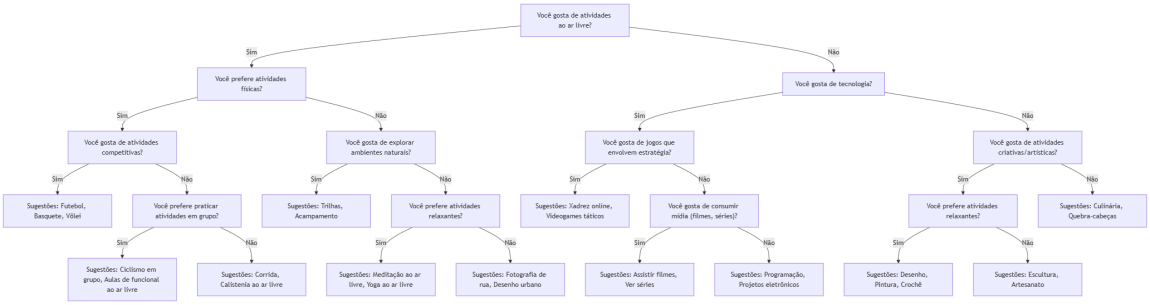

Figura 1: Diagrama completo da árvore de decisão utilizada para recomendação de hobbies.

Exemplos de Execução

Para ilustrar o funcionamento da árvore de decisão desenvolvida, foram realizadas execuções completas do programa, respondendo às perguntas de acordo com diferentes perfis de usuário. A seguir, são apresentados dois exemplos representativos.

Exemplo 1 — Usuário que prefere atividades ao ar livre

```
============================================================
SISTEMA DE RECOMENDAÇÃO DE HOBBIES
============================================================

Responda às perguntas a seguir para descobrir qual hobby
combina melhor com você!

Você gosta de atividades ao ar livre? (s/n): s
Você prefere atividades físicas? (s/n): s
Você gosta de atividades competitivas? (s/n): n
Você prefere praticar atividades em grupo? (s/n): s

============================================================
HOBBIES RECOMENDADOS PARA VOCÊ:
============================================================
1. Ciclismo em grupo
```

```
2. Aulas de funcional ao ar livre
============================================================
```

Neste caso, as respostas direcionam o usuário por um ramo da árvore associado a atividades ao ar livre, com foco em prática física em grupo, resultando em recomendações coerentes com o perfil informado.

Exemplo 2 — Usuário que prefere atividades internas e tecnologia

```
============================================================
SISTEMA DE RECOMENDAÇÃO DE HOBBIES
============================================================

Responda às perguntas a seguir para descobrir qual hobby
combina melhor com você!

Você gosta de atividades ao ar livre? (s/n): n
Você gosta de tecnologia? (s/n): s
Você gosta de jogos que envolvem estratégia? (s/n): n
Você gosta de consumir mídia (filmes, séries)? (s/n): s

============================================================
HOBBIES RECOMENDADOS PARA VOCÊ:
============================================================
1. Assistir filmes
2. Ver séries
============================================================
```

Esse segundo cenário mostra um usuário com preferência por ambientes internos, tecnologia e consumo de mídia, levando a recomendações compatíveis com esse perfil.

Parte 2 — Aprendizado Supervisionado (KNN, SVM, Árvore)

Introdução

A segunda parte deste trabalho tem como objetivo aplicar algoritmos de aprendizagem supervisionada para solucionar um problema de classificação utilizando um conjunto de dados reais.

Nesta etapa, foram implementados três modelos: *K-Nearest Neighbors* (KNN), *Support Vector Machine* (SVM) e uma *Árvore de Decisão*. Cada modelo foi treinado, validado e comparado quanto ao desempenho preditivo, explorando métricas como acurácia, precisão, recall e F1-score. Também foram adotadas práticas essenciais de pré-processamento, como normalização, seleção de atributos e divisão estratificada dos dados.

O propósito desta análise, além de apenas avaliar o desempenho individual de cada algoritmo, é também compreender como características como escalonamento, dimensionalidade e desbalanceamento entre classes influenciam o processo de aprendizagem. Dessa forma, busca-se oferecer uma visão comparativa e fundamentada sobre as vantagens, limitações e adequações de cada método diante do problema proposto.

Dataset

O conjunto de dados utilizado foi o *Diabetes Health Indicators Dataset* (Kaggle) [2], que reúne informações clínicas (ex.: glicemia, colesterol, IMC) e comportamentais relevantes para a identificação de indivíduos com diagnóstico positivo para diabetes. Esse tipo de aplicação é particularmente adequado para algoritmos de aprendizado supervisionado devido à presença de variáveis numéricas bem definidas e uma varável alvo, binária, claramente estabelecida (*diagnosed_ diabetes*).

Pré-processamento

Os passos de pré-processamento foram [1]:

- Seleção de 16 variáveis preditoras relevantes (idade, histórico familiar, pressão arterial, lipídios, glicemia, IMC, entre outras) e da variável alvo.
- Remoção de linhas com valores ausentes (`dropna`) e verificação de consistência.
- Normalização (StandardScaler) foi aplicada antes de KNN e SVM. Para evitar vazamento, o escalonamento foi sempre feito dentro das dobras de validação (pipeline).
- Divisão treino/teste estratificada (80/20) com semente fixa (`random_state=42`) para garantir reprodutibilidade.

Tipo de tarefa

O problema tratado nesta etapa é um problema de classificação binária, pois o objetivo do modelo é prever se um indivíduo é diagnosticado ou não com diabetes. A variável alvo *diagnosed_ diabetes* assume apenas dois valores (0 = não diagnosticado, 1 = diagnosticado), o que caracteriza a tarefa como classificação supervisionada de duas classes.

Resultados e Discussão

Nesta seção são apresentados os resultados obtidos pelos três algoritmos de classificação implementados: KNN, SVM e Árvore de Decisão. Todas as métricas, matrizes de confusão e análises foram geradas utilizando um subconjunto de 10.000 linhas do conjunto de dados original de diabetes, selecionadas aleatoriamente, porém de forma reprodutível pelo uso da semente fixa "$random\_state = 42$".

Os modelos foram avaliados por meio das métricas de acurácia, precisão (macro), recall (macro), F1-score (macro) e área ROC-AUC.

Resultados do KNN

O algoritmo KNN foi avaliado inicialmente por meio de uma busca do melhor valor de k usando validação cruzada estratificada. A seguir, a Figura 2 apresenta o gráfico de acurácia em função de k, que permitiu identificar o valor que maximiza o desempenho no conjunto de treino.

Curva de Acurácia para Diferentes Valores de K

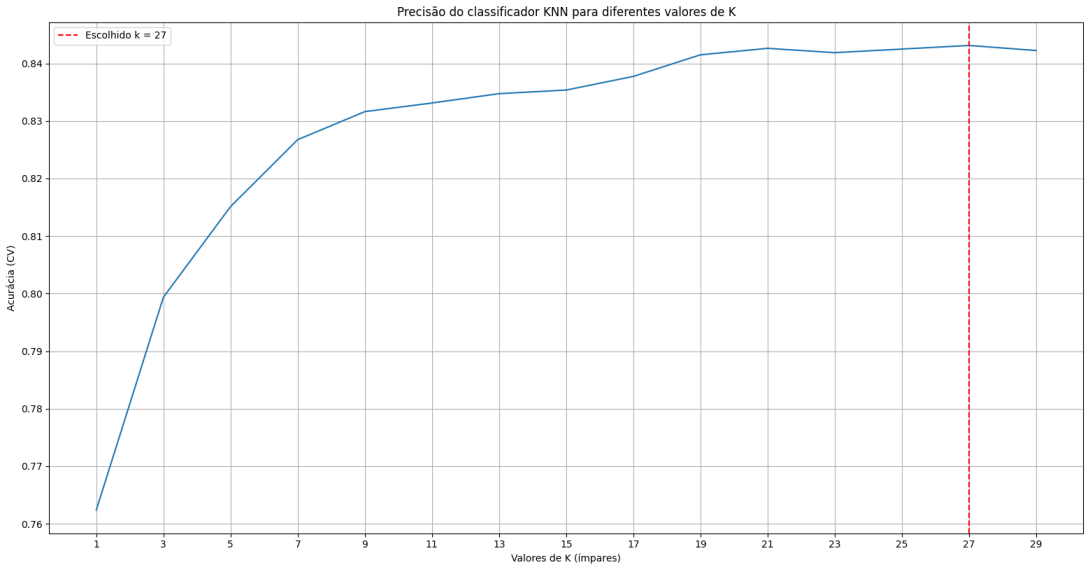

Figura 2: Acurácia do KNN em função do valor de k (Cross-Validation).

Após a seleção do melhor k, o modelo foi treinado e avaliado no conjunto de teste, gerando as seguintes métricas:

- **Acurácia:** 0.8370
- **Precisão (macro):** 0.8302
- **Recall (macro):** 0.8346
- **F1-Score (macro):** 0.8321
- **ROC-AUC:** 0.9144

A Figura 3 apresenta a matriz de confusão obtida para o classificador KNN no conjunto de teste. Essa representação permite analisar de forma detalhada a distribuição das predições corretas e incorretas entre as classes, evidenciando os verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos produzidos pelo modelo.

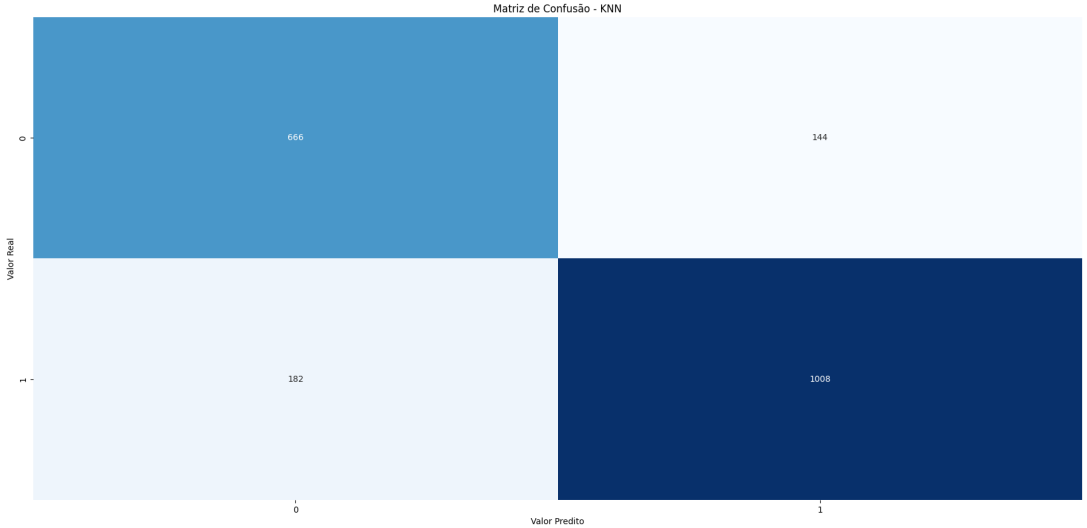

Figura 3: Matriz de confusão obtida para o KNN.

A Figura 4 mostra a curva ROC-AUC do modelo KNN, que relaciona a taxa de verdadeiros positivos com a taxa de falsos positivos para diferentes limiares de decisão. Essa curva é especialmente útil para avaliar a capacidade discriminativa do classificador de forma independente de um limiar fixo.

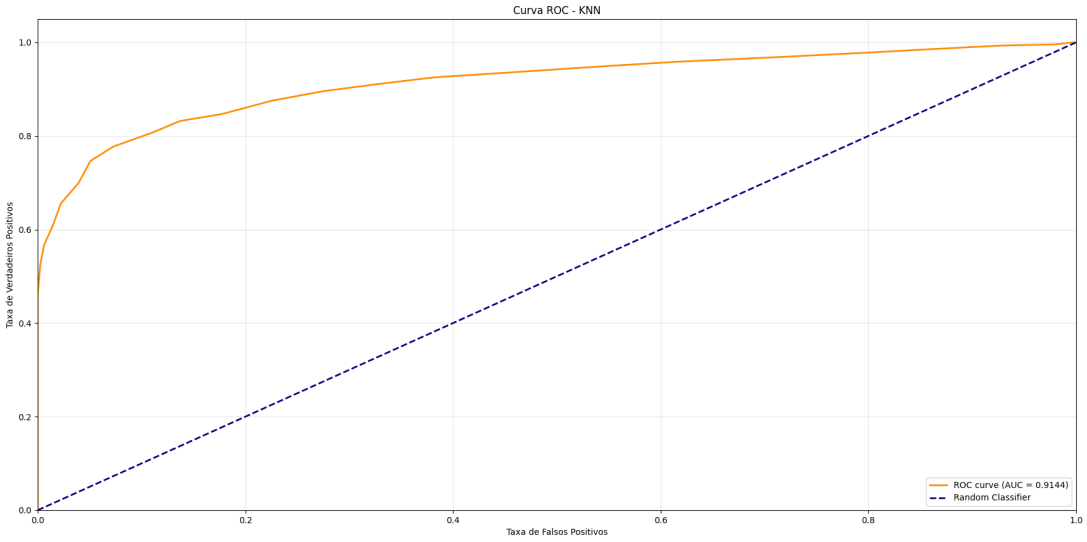

Figura 4: Curva ROC-AUC do KNN.

Resultados do SVM

O modelo SVM foi treinado com kernel linear e, antes do ajuste, os dados foram transformados por meio de uma Análise de Componentes Principais (PCA) configurada para manter 95% da variância total do conjunto original. Essa configuração mostrou-se vantajosa por duas razões: (i) proporcionou uma leve melhora na acurácia em comparação ao uso de todas as variáveis brutas, e (ii) reduziu a dimensionalidade sem causar aumento significativo no tempo de processamento, mantendo o treinamento eficiente. As métricas obtidas foram:

- **Acurácia:** 0.9515
- **Precisão (macro):** 0.9452
- **Recall (macro):** 0.9488
- **F1-Score (macro):** 0.9468
- **ROC-AUC:** 0.9182

A Figura 5 apresenta a matriz de confusão obtida para o classificador SVM após a aplicação do PCA. Essa visualização permite compreender como o modelo distribui seus acertos e erros entre as classes, complementando a análise quantitativa das métricas globais.

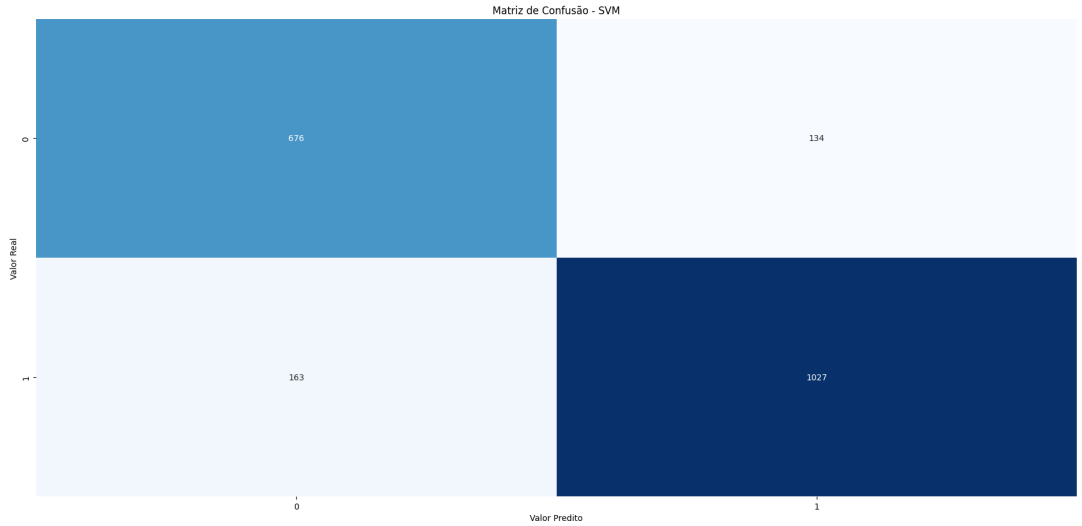

Figura 5: Matriz de confusão obtida para o SVM.

A Figura 6 ilustra a curva ROC-AUC do modelo SVM. Essa curva evidencia o desempenho do classificador ao variar o limiar de decisão, sendo uma medida importante da qualidade da separação entre as classes.

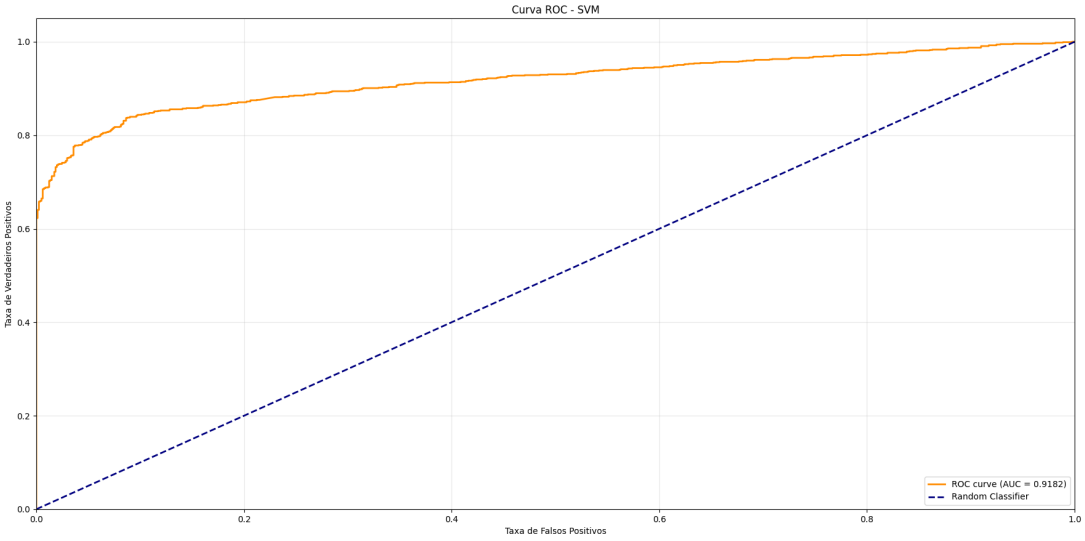

Figura 6: Curva ROC-AUC do SVM.

Resultados da Árvore de Decisão

A Árvore de Decisão apresentou o melhor desempenho geral entre os modelos avaliados, obtendo métricas superiores em todas as categorias:

- **Acurácia:** 0.9050
- **Precisão (macro):** 0.9021
- **Recall (macro):** 0.9166
- **F1-Score (macro):** 0.9038
- **ROC-AUC:** 0.9376

A Figura 7 apresenta a matriz de confusão da Árvore de Decisão. Essa representação permite observar com clareza o comportamento do modelo na classificação das duas classes, evidenciando sua capacidade de aprender regras decisórias eficazes a partir dos dados.

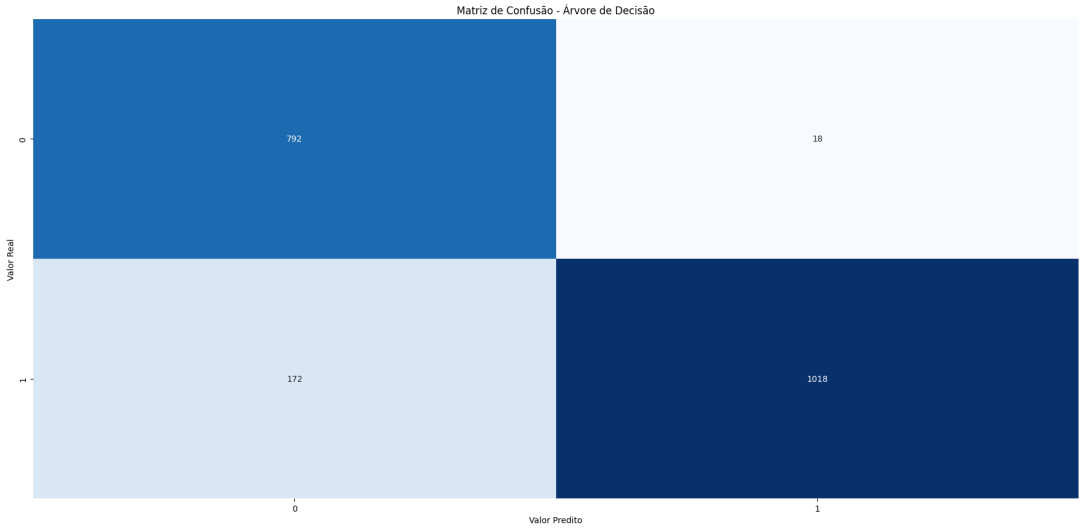

Figura 7: Matriz de confusão obtida para a Árvore de Decisão.

A Figura 8 apresenta a curva ROC-AUC associada à Árvore de Decisão. Essa curva permite avaliar a capacidade do modelo em distinguir corretamente entre indivíduos com e sem diagnóstico de diabetes para diferentes limiares de decisão.

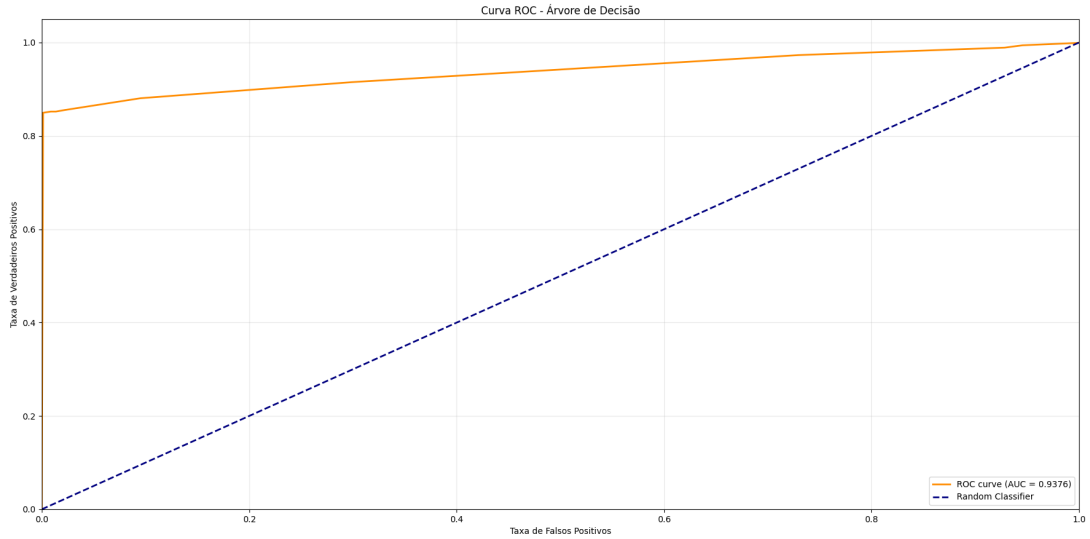

Figura 8: Curva ROC-AUC da Árvore de Decisão.

Comparação entre os Modelos

Após a análise individual de cada algoritmo, é possível comparar diretamente seus desempenhos. A Tabela 1 resume todas as métricas em um único formato padronizado.

Tabela 1: Comparação das métricas entre KNN, SVM e Árvore de Decisão (10.000 linhas).

Modelo	**Acurácia**	**Precisão**	**Recall**	**F1-Score**	**ROC-AUC**
KNN	0.8370	0.8302	0.8346	0.8321	0.9144
SVM	0.9515	0.9452	0.9488	0.9468	0.9182
Árvore de Decisão	**0.9050**	**0.9021**	**0.9166**	**0.9038**	**0.9376**

Observa-se que o modelo de Árvore de Decisão supera os demais algoritmos em todas as métricas avaliadas. O SVM apresenta desempenho intermediário, com acurácia ligeiramente superior ao KNN. O KNN, embora eficaz, foi o modelo com menor desempenho geral, possivelmente devido à sensibilidade ao escalonamento e à alta dimensionalidade do conjunto de atributos.

Conclusão

Entre os modelos avaliados, a Árvore de Decisão demonstrou a melhor capacidade preditiva, apresentando maior acurácia, melhores métricas macro e o maior valor de ROC-AUC. Seu bom desempenho sugere que o conjunto de dados possui divisões relativamente claras entre as classes, permitindo ao modelo aprender regras diretas e interpretáveis.

O SVM se destacou como uma alternativa robusta e estável, enquanto o KNN mostrou desempenho satisfatório, mas inferior aos demais, reforçando a importância de técnicas de ponderação e otimização para esse tipo de classificador.

Parte 3 — Algoritmo Genético

Introdução

Algoritmos Genéticos (AGs) são métodos de otimização inspirados no processo de seleção natural descrito por Darwin. Esses algoritmos trabalham com uma população de soluções candidatas, avaliadas por meio de uma função de *fitness*, e aplicam operadores como seleção, cruzamento e mutação para produzir novos indivíduos ao longo das gerações. [3]

Seu uso é especialmente adequado para problemas em que o espaço de busca é amplo, não linear ou apresenta múltiplos ótimos locais, dificultando a aplicação de métodos determinísticos tradicionais. A abordagem evolutiva permite explorar diferentes regiões do espaço de soluções enquanto mantém diversidade suficiente para evitar convergência prematura.

Nesta parte do trabalho, um Algoritmo Genético completo foi implementado manualmente, sem o uso de bibliotecas específicas de AG. São definidos a representação das soluções, o cálculo do *fitness*, os operadores genéticos utilizados e o critério de parada adotado. Por fim, são apresentados os resultados obtidos e uma análise do desempenho do método para o problema escolhido.

Problema escolhido

O problema selecionado para aplicação do Algoritmo Genético foi o *Feature Selection*, ou seleção de atributos. Assim, utilizar um AG para identificar subconjuntos de atributos potencialmente mais relevantes permite explorar reduções no custo computacional e, ao mesmo tempo, possibilita melhorias ou manutenção da acurácia dos modelos.

A seleção de atributos é um problema clássico para Algoritmos Genéticos, pois pode ser naturalmente representada por cromossomos binários (onde cada gene indica "usar" ou "não usar" uma feature), possui múltiplas soluções possíveis e apresenta um espaço de busca amplo e não linear — características adequadas para o processo evolutivo. Além disso, o AG permite otimizar o conjunto de atributos sem necessidade de gradientes, sendo robusto frente a relações complexas entre variáveis.

Modelagem

A seguir, descrevem-se as escolhas de modelagem adotadas para a implementação do Algoritmo Genético (AG) aplicado ao problema de *feature selection*.

Representação (codificação)

Cada indivíduo é codificado como um vetor binário de comprimento $m = 16$. Cada gene corresponde a um dos atributos considerados no experimento; valor 1 indica que o atributo é selecionado e 0 indica que ele é descartado. Essa codificação é direta e adequada ao problema, pois permite representar qualquer subconjunto de atributos de forma compacta.

Função de fitness

A função de fitness implementada no experimento é uma heurística simples que combina a soma dos genes (número de atributos selecionados) com uma penalização linear proporcional ao mesmo número:

$$\text{fitness}(\mathbf{x}) = \begin{cases} 0, & \text{se } \sum_i x_i = 0, \\ \sum_i x_i - 0{,}2 \cdot \sum_i x_i, & \text{caso contrário.} \end{cases}$$

Na implementação utilizada, isso equivale a fitness $= 0{,}8 \times$ (#features selecionadas). A escolha desta função foi intencionalmente simples para demonstrar e validar o funcionamento do AG; em aplicações práticas, a função de fitness normalmente incorpora uma medida de desempenho (por exemplo F1 por validação cruzada) e um termo de penalização pela cardinalidade do subconjunto.

Operador de seleção

A seleção de pais é realizada por uma estratégia probabilística baseada em aptidões: as aptidões de toda a população são normalizadas e uma amostra é extraída segundo essa distribuição (i.e., seleção proporcional à aptidão). Na prática isso favorece indivíduos com fitness maior, mantendo chance (menor) para os demais. Para evitar divisão por zero, usa-se um pequeno ε no denominador na implementação.

Cruzamento (crossover)

O cruzamento adotado é do tipo *one-point*: escolhe-se um ponto de corte aleatório $1 \leq p < m$ e os dois filhos são gerados trocando as caudas dos pais após esse ponto. O código disponibiliza um parâmetro de taxa de crossover, mas, na versão atual, a recombinação é aplicada de forma consistente ao gerar descendentes (i.e., a recombinação segue sempre a etapa de mutação). Caso se deseje respeitar estritamente a taxa de crossover, basta condicionar a aplicação do operador ao evento probabilístico `random() < crossover_rate`.

Mutação

A mutação é do tipo *bit-flip*: cada gene tem probabilidade p_m de ser invertido ($0 \leftrightarrow 1$). A implementação suporta uma mutação adaptativa com duas taxas diferentes: a taxa inicial (`mutation_rate`) e uma taxa aumentada (`mutation_rate_late`) que é ativada após a geração `mutation_switch_gen`. Essa estratégia visa aumentar a exploração caso o progresso se torne lento nas gerações posteriores.

Elitismo

Emprega-se elitismo simples: os melhores indivíduos da geração corrente (definidos pela fração `elitism_frac`) são copiados diretamente para a próxima geração sem alteração. No experimento reportado, `elitism_frac = 0.05` com `pop_size = 100`, o que preserva, na prática, pelo menos um indivíduo de elite por geração.

Critério de parada

O algoritmo é executado por um número fixo de gerações (`n_generations = 30`). Não foi implementado, na versão-base, um critério de parada adicional por estagnação; em vez disso, utiliza-se a mutação adaptativa para tentar escapar de platôs durante a execução.

Saída e monitoramento

A rotina de execução registra, por geração, o melhor fitness da população e mantém um histórico (`melhores_fitness`) para análise posterior. Uma figura com a evolução do melhor fitness ao longo das gerações é gerada e salva (arquivo: `evolucao_fitness.png`).

Parâmetros

Os parâmetros do AG usados na execução foram:

- Tamanho da população: pop_size = 100.
- Número máximo de gerações: n_generations = 30.
- Taxa de crossover: crossover_rate = 0,8 (parâmetro declarado; recombinação aplicada na geração).
- Taxa de mutação inicial: mutation_rate = 0,1.
- Taxa de mutação tardia: mutation_rate_late = 0,3 (ativa a partir de mutation_switch_gen = 20).
- Elitismo: fração elitism_frac = 0,05.
- Semente (reprodutibilidade): `seed` = 42.

Resultados

A seguir, apresenta-se os resultados obtidos com a execução do Algoritmo Genético aplicado ao problema de seleção de atributos. São analisados o comportamento da população ao longo das gerações, o valor do melhor fitness encontrado e a composição do indivíduo final. Também são discutidos aspectos observados no processo de convergência, permitindo compreender como os operadores evolutivos influenciaram o desempenho do algoritmo e a solução alcançada.

Evolução do Fitness

A Figura 9 apresenta a evolução do melhor fitness global ao longo das gerações. Observa-se que, inicialmente, o valor do fitness permanece em aproximadamente 12,0 até a 10ª geração, quando ocorre um salto para 12,8. A partir desse ponto, o algoritmo estabiliza, mantendo o mesmo valor até o final da execução.

Esse comportamento indica convergência rápida para uma solução estável. O elitismo contribuiu para preservar os melhores indivíduos ao longo das iterações, enquanto a mutação adaptativa — ativada nas gerações finais — não produziu melhora adicional, reforçando a ideia de que a população já havia se estabilizado.

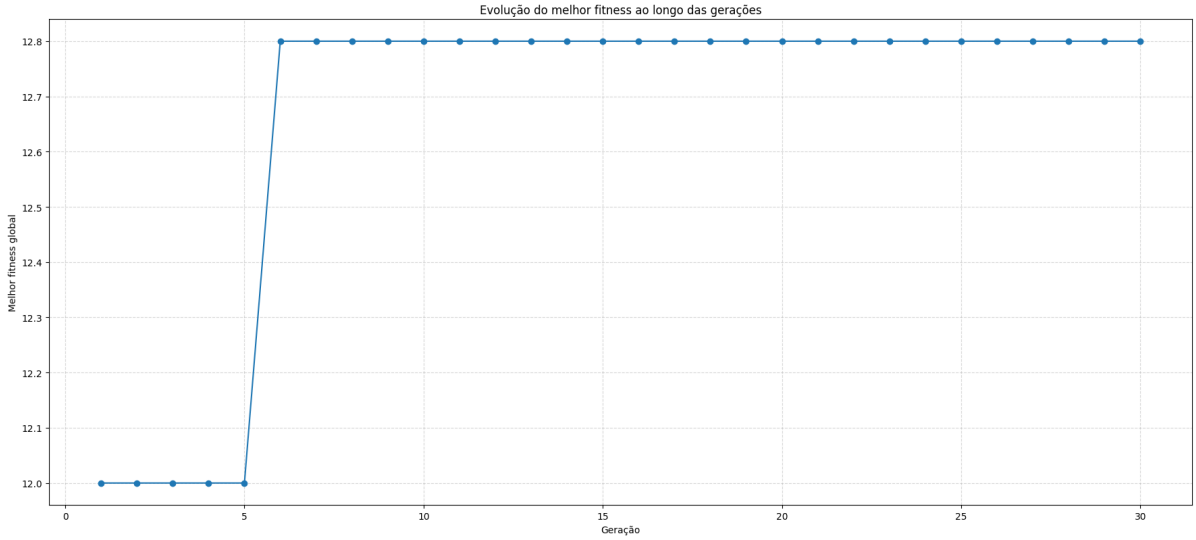

Figura 9: Evolução do melhor fitness global ao longo das gerações.

Melhor indivíduo encontrado

O melhor indivíduo encontrado ao final da execução foi:

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```

Esse vetor binário corresponde à seleção de todos os 16 atributos disponíveis. A lista completa é:

- Atributo 1, Atributo 2, Atributo 3, Atributo 4
- Atributo 5, Atributo 6, Atributo 7, Atributo 8
- Atributo 9, Atributo 10, Atributo 11, Atributo 12
- Atributo 13, Atributo 14, Atributo 15, Atributo 16

O valor de fitness correspondente foi:

$$\text{fitness} = 12{,}80$$

Interpretação dos resultados

A função de fitness utilizada combina o número de atributos selecionados com uma penalização linear. Como o ganho por atributo supera o valor da penalização, a solução

que maximiza o fitness é justamente aquela que seleciona todos os atributos. Assim, o resultado obtido é coerente com a formulação do problema.

A convergência precoce observada — com estabilização ainda na 10ª geração — reflete a simplicidade da função de avaliação, que favorece soluções densas e não impõe pressão seletiva suficiente para buscar subconjuntos menores.

Conclusão

O Algoritmo Genético convergiu rapidamente para a solução de maior fitness, mantendo estabilidade ao longo das gerações e ilustrando o funcionamento dos mecanismos de elitismo, cruzamento e mutação adaptativa, permitindo visualizar claramente o comportamento do AG e sua dinâmica de evolução em problemas de seleção de atributos.

Parte 4 — Enxame e Algoritmo Imune

Introdução

Nesta parte do trabalho, o problema de seleção de atributos foi abordado novamente, porém sob a ótica de meta-heurísticas bioinspiradas - sistema imune e comportamento de enxame - utilizando algoritmos já estabelecidos em literatura. A seleção de atributos é um problema clássico de otimização combinatória, cujo objetivo é identificar subconjuntos de variáveis relevantes capazes de manter ou melhorar o desempenho de um classificador, ao mesmo tempo em que se reduz a dimensionalidade do espaço de dados.

Tendo como base as implementações fornecidas pelo docente, foram implementados um algoritmo de enxame baseado em Particle Swarm Optimization (PSO binário) e um algoritmo imune baseado em Clonal Selection Algorithm (CLONALG). Com o objetivo de manter a estabilidade dos testes, ambos os métodos foram avaliados sob as mesmas condições experimentais, utilizando a mesma função fitness, o mesmo conjunto de dados e a mesma semente aleatória, conforme exigido no arquivo que guia este trabalho, de forma que fosse possível garantir a reprodutibilidade e comparação justa entre os métodos utilizados.

Problema

O problema tratado consiste na seleção de um subconjunto ótimo de atributos para um classificador supervisionado. Cada solução candidata é representada por um vetor binário, no qual o valor 1 indica a seleção do atributo correspondente, enquanto o valor 0 indica sua exclusão. O objetivo é minimizar uma função fitness multiobjetivo que combina o erro de classificação e a penalização pelo número de atributos selecionados.

A função fitness utilizada é definida por:

$$\text{fitness} = \alpha \cdot (1 - \text{acurácia}) + (1 - \alpha) \cdot \frac{|S|}{d}$$

onde a acurácia representa a média obtida por validação cruzada, $|S|$ é o número de atributos selecionados, d é o número total de atributos e α é um parâmetro de ponderação que controla o compromisso entre desempenho preditivo e redução dimensional. Neste trabalho, adotou-se $\alpha = 0{,}9$.

Parâmetros

Os principais parâmetros utilizados nos algoritmos foram definidos empiricamente e são apresentados a seguir.

CLONALG (Algoritmo Imune):

- Tamanho da população: 30
- Número de anticorpos selecionados: 5
- Fator de clonagem (β): 5
- Número de anticorpos aleatórios por geração: 2
- Número de gerações: 20
- Probabilidade inicial de ativação de atributos (p_{on}): 0,3

PSO Binário (Enxame):

- Número de partículas: 30
- Número de iterações: 20
- Peso de inércia (w): 0,7
- Coeficientes cognitivo e social ($c_1 = c_2$): 1,5

Todos os experimentos foram executados com a mesma semente aleatória, novamente com o objetivo de assegurar a reprodutibilidade dos resultados.

Resultados

Tabela 2: Resultados obtidos pelos algoritmos de enxame e imune no problema de seleção de atributos.

Algoritmo	**Fitness final**	**Atributos selecionados**	**Convergência**
CLONALG	0.037034	4	Rápida (Geração 4)
PSO Binário	0.045113	5	Moderada (Iteração 12)

O algoritmo imune CLONALG apresentou rápida convergência nas primeiras gerações, reduzindo o valor da função fitness de aproximadamente 0,053 para 0,037 até a 11ª geração, quando ocorreu estabilização do melhor valor. A solução final selecionou apenas 4 atributos, esse fator demonstra sua elevada capacidade preditiva e capacidade de redução de dimensões.

Esse comportamento indica convergência para um ótimo local de alta afinidade, o que é um comportamento esperado em algoritmos baseados em seleção clonal, em que a taxa de mutação diminui à medida que soluções de qualidade são encontradas.

O PSO binário, por sua vez, apresentou comportamento exploratório mais prolongado, com oscilações iniciais no valor do fitness antes de atingir a convergência. Em geral, o PSO tende a explorar regiões mais amplas do espaço de busca, podendo alcançar soluções com fitness semelhante ou inferior àquelas obtidas pelo CLONALG, ainda que, em alguns casos, com um número ligeiramente maior de atributos selecionados.

Comparação com AG

A comparação entre os métodos de enxame (PSO), imune (CLONALG) e o Algoritmo Genético (AG), apresentado na Parte 3 deste trabalho, mostra diferenças importantes decorrentes da formulação da função de fitness e dos mecanismos de busca empregados por cada abordagem.

No Algoritmo Genético, a função de fitness foi definida de forma mais simples, priorizando o número de atributos selecionados, com uma penalização linear baixa para desestimular soluções densas. Como consequência disso, o AG convergiu mais rapidamente para a solução que seleciona todos os atributos disponíveis, uma vez que essa configuração maximiza o valor da função objetivo. A convergência rápida observada nesse caso reflete a ausência de pressão seletiva maior, de modo que conjuntos menores fossem favorecidos, medida que certamente alteraria os resultados obtidos.

De forma oposta, tanto o CLONALG quanto o PSO foram aplicados ao mesmo problema de seleção de atributos utilizando uma função fitness multiobjetivo que combina o erro de classificação obtido por validação cruzada com um termo de penalização proporcional ao número de atributos selecionados. Essa formulação faz com que exista um compromisso explícito entre desempenho preditivo e redução dimensional, e isso gera soluções mais compactas e informativas.

Os resultados mostraram que o CLONALG obteve o menor valor de fitness final, selecionando apenas quatro atributos, e isso indica uma grande capacidade de compactação sem perda significativa de desempenho. O PSO, por outro lado, apresentou comportamento exploratório mais prolongado, convergindo para uma solução competitiva e com um subconjunto maior de atributos.

Dessa forma, enquanto o AG demonstrou corretamente o funcionamento dos operadores evolutivos em um cenário didático, os métodos de enxame e imune mostraram-se mais adequados para a resolução prática do problema de seleção de atributos quando combinados a uma função de avaliação que reflete simultaneamente desempenho e complexidade do modelo. Essa comparação reforça a importância da escolha da função fitness na aplicação de meta-heurísticas e evidencia como diferentes paradigmas de otimização respondem a distintas pressões seletivas.

Conclusões e Trabalhos Futuros

Este trabalho possibilitou a exploração prática e conceitual de diferentes paradigmas da Inteligência Artificial, abrangendo abordagens simbólicas, métodos de aprendizado supervisionado e técnicas de otimização baseadas em meta-heurísticas. Ao longo das quatro partes desenvolvidas, foi possível compreender como cada abordagem modela problemas, representa conhecimento e toma decisões, bem como suas vantagens, limitações e contextos de aplicação.

Na Parte 1, a construção de uma árvore de decisão manual contribuiu para o entendimento de como processos decisórios podem ser estruturados de forma lógica, hierárquica e interpretável. Essa etapa reforçou a importância da escolha adequada da pergunta inicial, da organização do espaço de decisões e da representação explícita do conhecimento, evidenciando o papel das abordagens simbólicas na transparência e explicabilidade de sistemas de IA.

Na Parte 2, a implementação de algoritmos de aprendizado supervisionado permitiu con-

solidar conceitos fundamentais relacionados ao fluxo completo de uma tarefa de classificação. Foram explorados aspectos como pré-processamento de dados, escalonamento, redução de dimensionalidade, validação cruzada e avaliação por múltiplas métricas. Essa etapa possibilitou compreender como diferentes modelos respondem às características dos dados e como escolhas metodológicas influenciam o desempenho e a robustez das soluções.

Na Parte 3, a implementação de um Algoritmo Genético do zero proporcionou uma compreensão aprofundada dos mecanismos evolutivos, incluindo codificação, função de fitness, seleção, cruzamento, mutação e elitismo. A aplicação do algoritmo ao problema de seleção de atributos evidenciou a flexibilidade das meta-heurísticas e destacou como a modelagem da função de fitness impacta diretamente o comportamento do processo evolutivo e a diversidade das soluções encontradas.

Na Parte 4, a aplicação de métodos bioinspirados baseados em enxame e sistema imune permitiu ampliar a compreensão sobre estratégias alternativas de otimização. Essa etapa reforçou o entendimento sobre conceitos como exploração e intensificação da busca, diversidade populacional e convergência, além de possibilitar uma comparação conceitual entre diferentes meta-heurísticas aplicadas ao mesmo problema de seleção de atributos.

De forma geral, o trabalho contribuiu para a consolidação de uma visão integrada da Inteligência Artificial, evidenciando como abordagens simbólicas, estatísticas e bioinspiradas podem ser utilizadas de maneira complementar. A principal contribuição está no aprendizado prático sobre a escolha adequada de modelos, métricas e funções objetivo, de acordo com as características e objetivos do problema abordado.

Como trabalhos futuros, diversas extensões podem ser exploradas. A árvore de decisão manual pode ser expandida para incluir respostas não binárias ou mecanismos automáticos de aprendizado da estrutura. No aprendizado supervisionado, podem ser avaliados outros classificadores, técnicas mais avançadas de ajuste de hiperparâmetros e estratégias de balanceamento de classes. No contexto das meta-heurísticas, estudos futuros podem investigar abordagens híbridas que combinem Algoritmo Genético, PSO e algoritmos imunes, bem como funções de fitness baseadas diretamente no desempenho de modelos de aprendizado. Essas extensões podem contribuir para aprofundar ainda mais a compreensão das técnicas estudadas e ampliar sua aplicabilidade a problemas reais.

Reprodutibilidade

A reprodutibilidade dos experimentos foi garantida por meio do uso consistente de uma semente fixa de aleatoriedade ($seed = 42$) em todas as etapas que envolvem processos não determinísticos. Essa prática assegura que a seleção de dados, a divisão entre conjuntos de treino e teste, bem como os comportamentos de algoritmos que dependem de inicialização aleatória, produzam sempre os mesmos resultados quando executados sob as mesmas condições.

Dessa forma, os experimentos, métricas, gráficos e conclusões apresentados neste trabalho podem ser reproduzidos de maneira fiel, desde que sejam mantidos o ambiente computacional e as versões das bibliotecas utilizadas.

Referências Bibliográficas

[1] MOURA, Pedro A.; SILVA, Jader O. **IA-Trabalho-2 — Repositório do projeto**. Disponível em: `https://github.com/PedroAugusto08/Trabalho-II-IA.`

git. Acesso em: nov. 2025.

[2] THALLA, Mohan Krishna. **Diabetes Health Indicators Dataset**. Kaggle, 2021. Disponível em: https://www.kaggle.com/datasets/mohankrishnathalla/diabetes-health-indicators-dataset. Acesso em: nov. 2025.

[3] T. A. de Oliveira, **Algoritmos Genéticos**, Sistema Integrado de Gestão de Atividades Acadêmicas. CEFET-MG, Campus Divinópolis. Acesso em nov. 2025.