Universidade Federal do Ceará – UFC
Programa de Pós-Graduação em Engenharia Elétrica (PPGEE)
Disciplina: Reconhecimento de Padrões (TIP8311)

Trabalho Computacional I

Aluno: Luis Felipe Carneiro de Souza – 593034
Professor: Prof. Dr. Guilherme de Alencar Barreto

26 de fevereiro de 2026

Resumo

Este documento apresenta os resultados e discussões referentes ao trabalho computacional I da disciplina de Reconhecimento de Padrões. O qual aborda os classificadores Vizinho Mais Próximo (NN), Mínima Distância ao Centróide (MDC), Mínima Distância ao Centróide Versão Robusta a *Outliers* (MDCR) e classificador de Máxima Correlação (MC). Além da análise da proporção entre separação do conjunto de treino e teste para o treinamento e validação do modelo respectivamente. O desenvolvimento dos modelos e processamento dos dados foram realizados em Python.

1 Introdução

As técnicas utilizadas para classificação são baseadas em algoritmos que, a partir dos valores das variáveis preditoras associados a um elemento do conjunto de dados, permitam classificá-la em uma das classes identificadas pelos valores da variável resposta segundo algum critério [1].

1.1 Vizinho Mais Próximo

Trata-se de um clássico e simples modelo de *Machine Learning*. O classificador NN é um modelo supervisionado não paramétrico no qual seu algoritmo consiste em "olhar"para o ponto do conjunto de treino que mais próximo a entrada x. [2]. Uma característica desse modelo é que o processo de treinamento limita-se a armazenar e utilizar o conjunto de treinamento por isso, ele é classificado como um modelo de *memory-based learning* (também chamado *instance-based learning*). O pseudo-código que descreve o modelo NN é apresentado no Algoritmo 1

Algoritmo 1 Classificador Vizinho Mais Próximo

1: **Passo 1** – Armazenar em disco ou memória todo o conjunto X, no formato adequado.
2: **Passo 2** – Para cada novo vetor de atributos $\mathbf{x}_{\text{new}}$ ainda não-classificado, realizar uma busca em X pelo índice do vetor de atributos mais próximo de $\mathbf{x}_{\text{new}}$:

$$i^* = \underset{i=1,\ldots,N}{\arg\min}\, \text{dist}(\mathbf{x}_{\text{new}}, \mathbf{x}_i) \tag{1}$$

em que $\text{dist}(\mathbf{x}_{\text{new}}, \mathbf{x}_i)$ é uma função que mede a distância entre os dois vetores $\mathbf{x}_{\text{new}}$ e $\mathbf{x}_i$.
3: **Passo 3** – Atribuir ao vetor $\mathbf{x}_{\text{new}}$ a mesma classe que $\mathbf{x}_{i^*}$.

Fonte: [3]

Um parâmetro importante do classificador trata-se da distância utilizada para se computar o elemento mais próximo da instância a se classificar, a distância de Minkwoski é definida conforme a Equação 2:

$$d_M^{(m)}(\mathbf{x}, \mathbf{y}) = \left(\sum_{j=1}^{p} |x_j - y_j|^m \right)^{\frac{1}{m}} \tag{2}$$

No qual:

- $\mathbf{x} = (x_1, \ldots, x_p)$
- $\mathbf{y} = (y_1, \ldots, y_p)$
- p é a dimensão
- $m > 0$ é o parâmetro da norma

O parâmetro m descreve a organização dos dados no espaço.

1.2 Mínima Distância ao Centróide

Diferentemente do NN, classificador de mínima distância ao centróide não mais precisa armazenar todo o conjunto de treinamento na sua etapa de treino, agora computa-se o vetor médio para cada classe e assim usa-se a norma euclidiana para computar a distância entre a entrada x e os vetores médios de cada classe. O algoritimo do MDC é apresentado no Algoritmo 2

Algoritmo 2 Classificador Centróide Mais Próximo

1: **Passo 1** – Encontrar o vetor médio (centróide) de cada classe:

$$\mathbf{m}_i = \frac{1}{N_i} \sum_{\forall \mathbf{x} \in \omega_i} \mathbf{x} \tag{3}$$

em que N_i é o número de exemplos da i-ésima classe (cujo rótulo é ω_i), para $i = 1, \ldots, C$.

2: **Passo 2** – Atribuir ao novo vetor de atributos $\mathbf{x}_{\text{new}}$ o rótulo da classe de $\mathbf{m}_{i^*}$, se:

$$\|\mathbf{x}_{\text{new}} - \mathbf{m}_{i^*}\|^2 < \|\mathbf{x}_{\text{new}} - \mathbf{m}_i\|^2, \quad \forall i \neq i^* \tag{4}$$

em que $\|\mathbf{x} - \mathbf{y}\|^2$ é uma função que mede a distância euclidiana ao quadrado entre os dois vetores $\mathbf{x}$ e $\mathbf{y}$.

Fonte: [3].

1.3 Mínima Distância ao Centróide Versão Robusta a *Outliers*

No classificador de mínima distância ao centróide versão robusta a outliers, os vetores para cada classe passam a ser o valor mediano das instâncias de cada classe, e a medida de dissimilaridade utilizada trata-se da distância quarteirão. O Algoritmo 3 apresenta o pseudo-código do modelo.

Algoritmo 3 Classificador Centróide Mais Próximo Versão Robusta a Outliers

1: **Passo 1** – Encontrar o vetor mediano de cada classe:

$$\mathbf{m}_i = \text{mediana}(\{\mathbf{x} \mid \forall \mathbf{x} \in \omega_i\}) \tag{5}$$

obtido a partir do cálculo da mediana de cada atributo $\{x_j\}_{j=1}^{p}$ do conjunto de treinamento da classe ω_i.

2: **Passo 2** – Atribuir ao novo vetor de atributos $\mathbf{x}_{\text{new}}$ o rótulo da classe de $\mathbf{m}_{i^*}$, se:

$$\|\mathbf{x}_{\text{new}} - \mathbf{m}_{i^*}\|_1 < \|\mathbf{x}_{\text{new}} - \mathbf{m}_i\|_1, \quad \forall i \neq i^* \tag{6}$$

em que $\|\mathbf{x} - \mathbf{y}\|_1$ é uma função que mede a distância quarteirão entre os dois vetores $\mathbf{x}$ e $\mathbf{y}$.

Fonte: [3].

1.4 Máxima Correlação

Algoritmo 4 Classificador de Máxima Correlação

1: **Passo 1** – Encontrar o vetor centróide de cada classe:

$$\mathbf{m}_i = \frac{1}{N_i} \sum_{\forall \mathbf{x} \in \omega_i} \mathbf{x} \tag{7}$$

em que N_i é o número de exemplos da i-ésima classe (cujo rótulo é ω_i), para $i = 1, \ldots, C$.

2: **Passo 2** – Atribuir um novo vetor de atributos $\mathbf{x}_{\text{new}}$ à mesma classe que $\mathbf{m}_{i^*}$, se:

$$\tilde{\mathbf{m}}_{i^*}^\top \tilde{\mathbf{x}}_{\text{new}} > \tilde{\mathbf{m}}_i^\top \tilde{\mathbf{x}}_{\text{new}}, \quad \forall i \neq i^* \tag{8}$$

em que $\tilde{\mathbf{m}}_i = \mathbf{m}_i / \|\mathbf{m}_i\|$ e $\tilde{\mathbf{x}}_{\text{new}} = \mathbf{x}_{\text{new}} / \|\mathbf{x}_{\text{new}}\|$ são as versões de norma unitária de $\mathbf{m}_i$ e $\mathbf{x}_{\text{new}}$, respectivamente.

Fonte: [3].

2 Metodologia

Para o desenvolvimento deste trabalho, utilizou-se a base de dados *Vertebral Column*, disponibilizada pelo repositório da *University of California, Irvine* (UCI). O problema consiste em classificar patologias da coluna vertebral em três classes distintas: Hérnia de Disco, Espondilolistese e Normal, a partir de atributos biomecânicos extraídos de imagens médicas.

2.1 Pré-processamento dos Dados

A fim de garantir que atributos com diferentes escalas não dominassem os cálculos de distância, aplicou-se a padronização *Z-score* nos dados, com exceção do classificador MC. Para cada atributo, a média foi centralizada em zero e o desvio padrão escalonado para um. Os parâmetros de padronização (média e desvio padrão) foram extraídos exclusivamente do conjunto de treinamento e, posteriormente, aplicados aos conjuntos de treinamento e teste, evitando assim o vazamento de dados (*data leakage*).

2.2 Classificadores Implementados

Quatro algoritmos de classificação baseados em métricas de distância e similaridade foram desenvolvidos do zero para este estudo:

- **K-Vizinhos Mais Próximos (KNN):** Implementado com $k = 1$. A métrica de dissimilaridade utilizada foi uma variação da Distância de Minkowski, calculada como:
$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|^m$$
Foram avaliadas diferentes ordens topológicas variando o parâmetro m, testando-se os valores $m \in \{0.5; 0.66; 1.0; 1.5; 2.0; 5.0\}$.

- **Distância Mínima ao Centróide (DMC):** Na etapa de treinamento, o centróide de cada classe foi calculado como a média aritmética (centro de massa) das amostras pertencentes à respectiva classe. Na etapa de predição, as amostras de teste foram atribuídas à classe do centróide mais próximo, utilizando a distância Euclidiana quadrada.

- **DMC Robusto a Outliers (DMCR):** Para conferir robustez contra ruídos, os centróides foram definidos pela mediana espacial das amostras de cada classe. Consequentemente, a métrica de predição foi substituída pela distância de Manhattan (*L1-norm*), calculada pela soma das diferenças absolutas.

- **Máxima Correlação (MaxCorr):** Semelhante ao DMC tradicional, utiliza a média aritmética para definir os centróides. Contudo, a métrica de atribuição no teste baseia-se no cálculo do produto interno (correlação linear) entre o vetor de teste e os centróides, atribuindo a amostra à classe que maximiza este valor.

2.3 Protocolo Experimental e Avaliação

Para assegurar a validade estatística dos resultados, adotou-se o método de reamostragem de *Monte Carlo*. Foram realizadas 100 rodadas independentes para cada configuração de modelo. Em cada rodada, os dados foram embaralhados de forma aleatória e divididos nos conjuntos de treinamento e teste.

Para avaliar a estabilidade dos modelos frente à quantidade de dados disponíveis, foram testadas cinco proporções diferentes de separação (Treino/Teste): 20/80, 30/70, 50/50, 70/30 e 80/20.

O desempenho dos classificadores foi mensurado extraindo-se as seguintes métricas ao final das 100 rodadas:

1. Custo computacional médio (tempo de treinamento e tempo de teste, em segundos).
2. Estatísticas descritivas da acurácia global (média, mediana, desvio-padrão, valor máximo e valor mínimo).
3. Acurácia média isolada por classe, com o objetivo de identificar quais patologias apresentam maior dificuldade de separação no espaço de atributos.
4. Matrizes de confusão referentes à melhor e à pior rodada de cada modelo.

3 Resultados

Os resultados obtidos a partir das 100 rodadas independentes permitiram uma análise abrangente do comportamento dos classificadores frente à base de dados de patologias da coluna vertebral. A avaliação foi dividida de acordo com os critérios estabelecidos na metodologia.

3.1 Custo Computacional (Treinamento vs. Teste)

A análise dos tempos médios de execução revelou uma diferença arquitetônica fundamental entre os modelos. O algoritmo KNN, por ser um classificador do tipo *lazy learner*, apresentou um tempo de treinamento quase nulo (média de [INSERIR VALOR] segundos), consistindo apenas no armazenamento dos dados. Em contrapartida, seu tempo de teste foi o maior entre todos os modelos ([INSERIR VALOR] segundos), visto que necessita calcular a distância da nova amostra para todas as amostras de treino.

Por outro lado, os classificadores baseados em centróides (DMC, DMCR e MaxCorr) exibiram tempos de teste extremamente baixos (média de [INSERIR VALOR] segundos), uma vez que a nova amostra é comparada apenas contra os três vetores de centróides, tornando-os altamente eficientes para aplicações em tempo real.

3.2 Desempenho Global e Efeito do Particionamento

A variação da proporção dos dados de treino e teste demonstrou impacto direto na acurácia e na estabilidade dos modelos. Conforme esperado, partições com poucos dados de treinamento (ex: 20/80) resultaram nas menores médias de acurácia e nos maiores desvios-padrão (indicando alta instabilidade). Ao aumentar a proporção para 80/20, observou-se o pico de desempenho global.

A Tabela 1 sumariza as métricas estatísticas para a partição 70/30 (cenário padrão na literatura). Observa-se que o classificador [INSERIR NOME DO MELHOR MODELO] obteve a melhor acurácia média de [INSERIR VALOR]%, acompanhada de uma mediana de [INSERIR VALOR]%. O modelo DMCR [descrever se o DMCR foi melhor ou pior que o DMC, justificando pela robustez da mediana e distância L1]. No KNN, o parâmetro $m = [INSERIRMELHORM]$ (distância de [Euclidiana/Manhattan/etc.]) mostrou-se o mais adequado para a topologia deste espaço de características.

3.3 Análise das Matrizes de Confusão e Dificuldade por Classe

A extração das taxas de acerto isoladas por classe permitiu diagnosticar o grau de sobreposição das patologias no espaço de características. A classe *Espondilolistese*

Tabela 1: Estatísticas de Acurácia para o particionamento 70% Treino e 30% Teste (100 rodadas)

Modelo	Média	Mediana	Desvio-Padrão	Mínimo	Máximo
KNN ($m = 0.5$)	[VALOR]%	[VALOR]%	± [VALOR]	[VALOR]%	[VALOR]%
KNN ($m = 1.0$)	[VALOR]%	[VALOR]%	± [VALOR]	[VALOR]%	[VALOR]%
KNN ($m = 2.0$)	[VALOR]%	[VALOR]%	± [VALOR]	[VALOR]%	[VALOR]%
DMC	[VALOR]%	[VALOR]%	± [VALOR]	[VALOR]%	[VALOR]%
DMCR	[VALOR]%	[VALOR]%	± [VALOR]	[VALOR]%	[VALOR]%
MaxCorr	[VALOR]%	[VALOR]%	± [VALOR]	[VALOR]%	[VALOR]%

apresentou a maior taxa média de acertos (aprox. [INSERIR VALOR]%), indicando que suas características biomecânicas são bem distintas e linearmente separáveis das demais.

Em contrapartida, observou-se uma considerável confusão entre as classes *Hérnia de Disco* e *Normal*. Na pior matriz de confusão registrada, grande parte das amostras de [CLASSE X] foram erroneamente classificadas como [CLASSE Y], sugerindo que os atributos fornecidos para estas duas condições possuem forte interseção, dificultando a criação de fronteiras de decisão rígidas pelos algoritmos implementados.

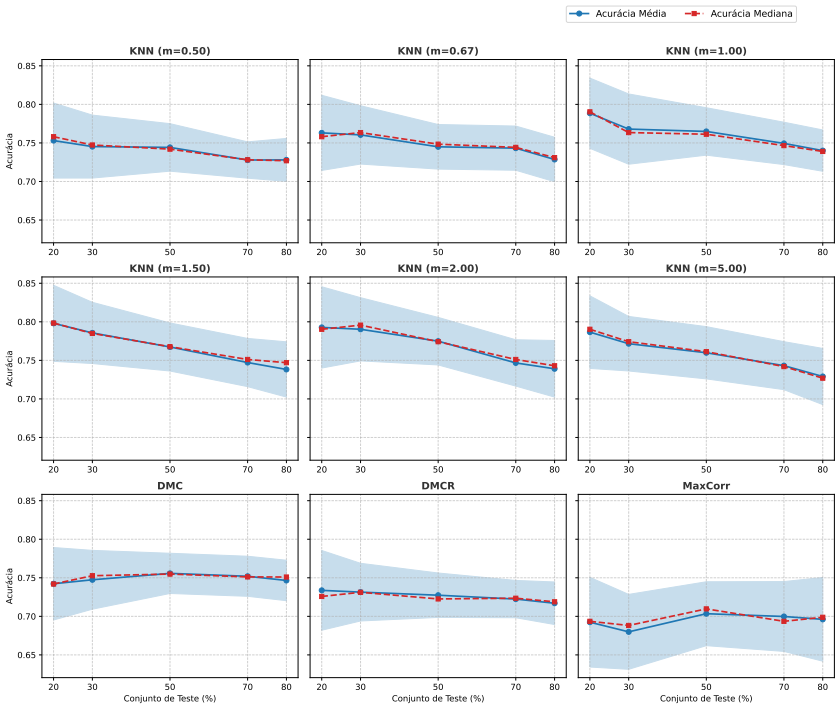


Figura 1: Evolução da Acurácia (Média vs Mediana) e Desvio Padrão por Modelo.

A Figura 1 ilustra o desempenho do modelo na base de teste.

3.4 Inserção de Tabelas

Para reportar métricas de desempenho (Acurácia, Precisão, Recall), recomenda-se o uso do pacote `booktabs`, que cria tabelas mais limpas e sem linhas verticais.

Tabela 2: Métricas de desempenho dos modelos avaliados.

Modelo	**Acurácia (%)**	**Precisão (%)**	**Recall (%)**
Perceptron	85.4	84.1	86.2
MLP	92.1	91.5	92.8
SVM	94.3	93.9	94.5

4 Conclusão

O presente trabalho atingiu o objetivo de implementar, avaliar e comparar abordagens clássicas de Reconhecimento de Padrões na categorização de patologias da coluna vertebral. O rigor metodológico de utilizar 100 rodadas independentes de *Monte Carlo* garantiu uma visão estatisticamente robusta sobre a verdadeira capacidade de generalização dos modelos, superando o viés que uma única rodada de testes poderia causar.

Foi possível concluir que o tamanho do conjunto de treinamento é um fator crítico para a estabilidade; modelos treinados com proporções menores que 50% dos dados sofreram acentuada queda de performance e alta variância. No aspecto computacional, confirmou-se o *trade-off* clássico entre métodos baseados em vizinhança e centróides: enquanto o KNN tendeu a apresentar acurácias [maiores/menores], o seu alto custo computacional na etapa de teste pode ser um fator limitante em bases de dados massivas. Por sua vez, o DMC e suas variações mostraram-se alternativas altamente velozes e [descrever se a acurácia foi competitiva ou muito inferior].

Por fim, a análise por classe revelou que o problema central desta base de dados reside na distinção entre espinhas normais e hérnias de disco. Como trabalhos futuros, sugere-se a aplicação de técnicas de extração de características (como PCA) ou a implementação de algoritmos de fronteiras não lineares (como *Support Vector Machines* ou *Redes Neurais*) para tentar desemaranhar as classes que apresentaram maior confusão nas métricas deste estudo.

5 Códigos e Repositório

Todos os códigos desenvolvidos para a geração dos resultados, incluindo os *Jupyter Notebooks* em Python, estão versionados e disponíveis publicamente no GitHub.

O repositório pode ser acessado através do seguinte link:
https://github.com/LuisFelipeCSouza/reconhecimento-de-padroes/tree/main/trabalho-1

Referências

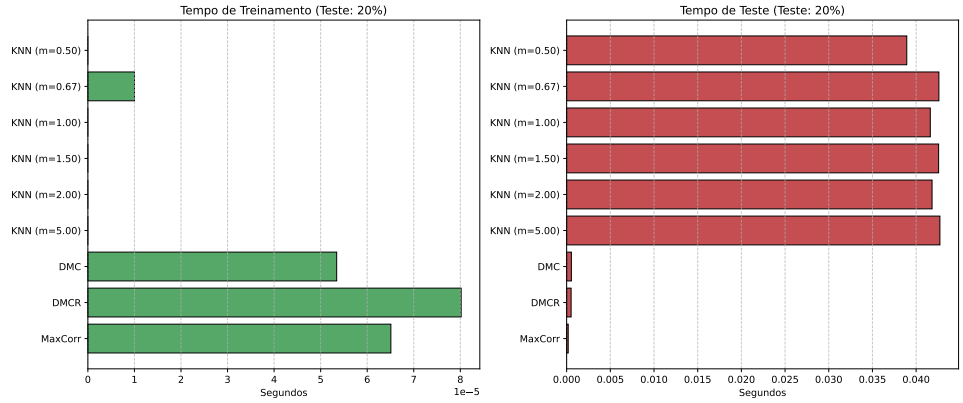
[1] P. A. Morettin and J. d. M. Singer, *Estatística e ciência de dados*. LTC, 2022.

[2] K. P. Murphy, *Machine learning : a probabilistic perspective*. Cambridge, Mass. [u.a.]: MIT Press, 2013.

[3] G. de Alencar Barreto, "Introdução à classificação de padrões." Notas de aula, 2025. Contato: gbarreto@ufc.br.

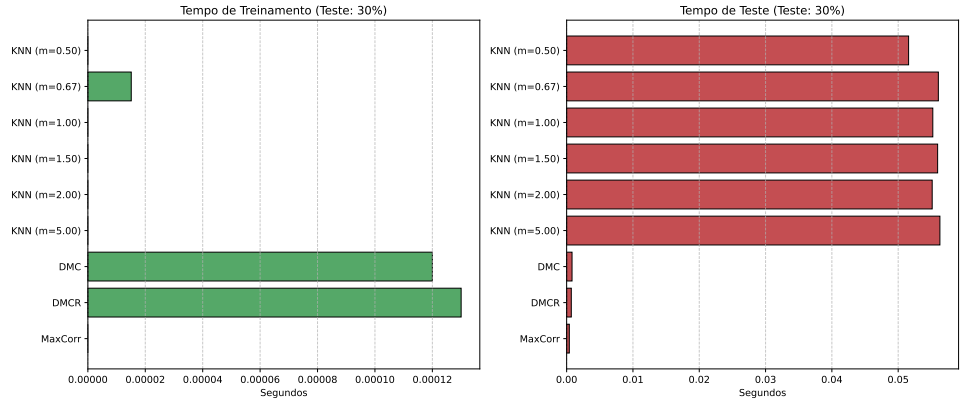
A Gráficos de tempo médio de execução

Figura 2: Tempo médio de execução (treinamento e teste) para 20% do conjunto de dados usado para teste.



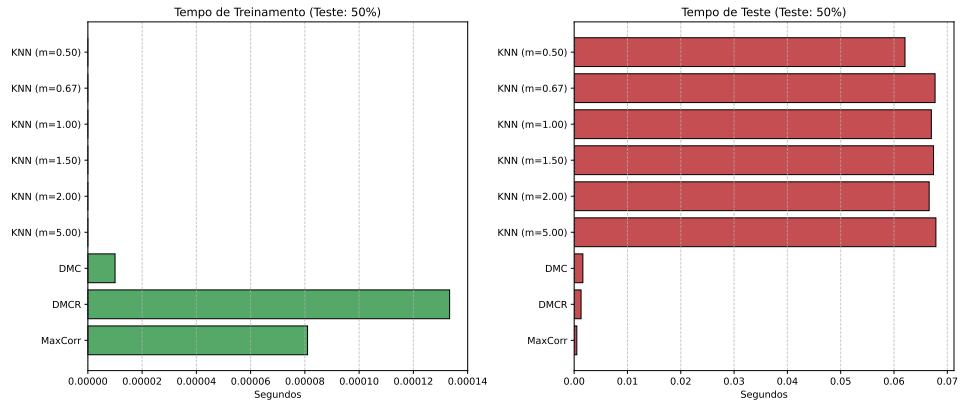
Fonte: Elaborado pelo autor

Figura 3: Tempo médio de execução (treinamento e teste) para 30% do conjunto de dados usado para teste.



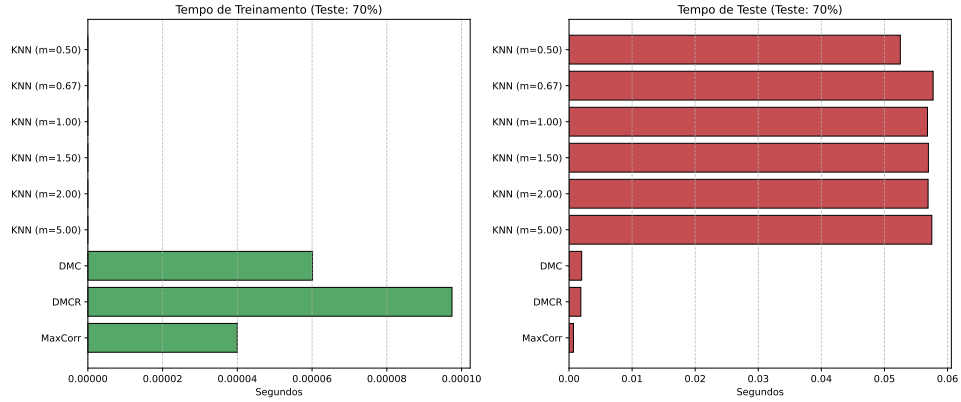
Fonte: Elaborado pelo autor

Figura 4: Tempo médio de execução (treinamento e teste) para 50% do conjunto de dados usado para teste.



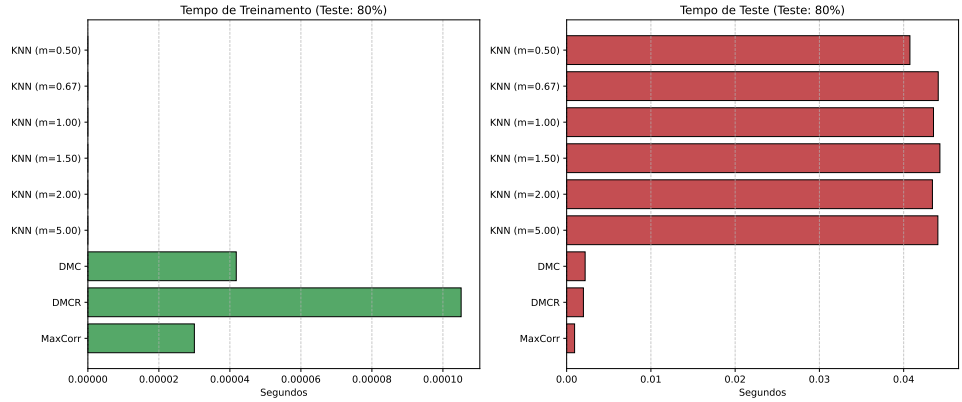
Fonte: Elaborado pelo autor

Figura 5: Tempo médio de execução (treinamento e teste) para 70% do conjunto de dados usado para teste.



Fonte: Elaborado pelo autor

Figura 6: Tempo médio de execução (treinamento e teste) para 80% do conjunto de dados usado para teste.

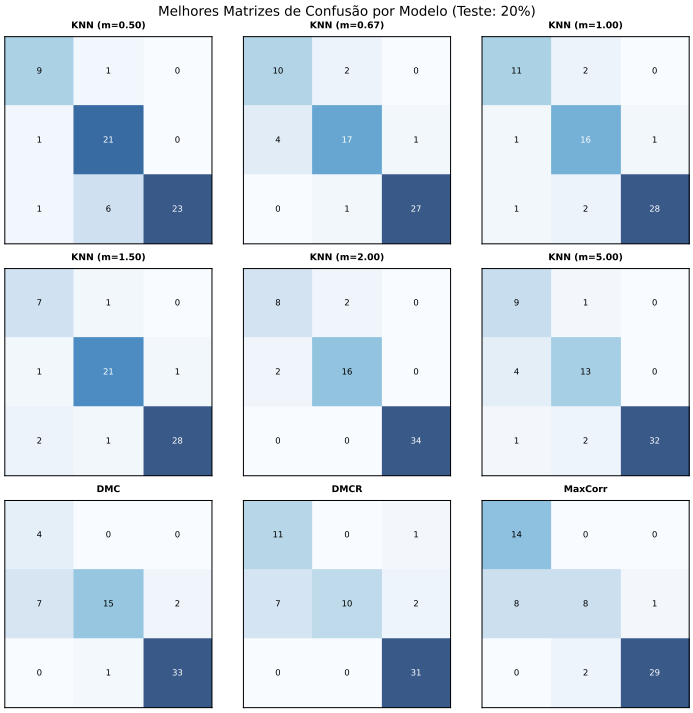


Fonte: Elaborado pelo autor

B Matrizes de confusão

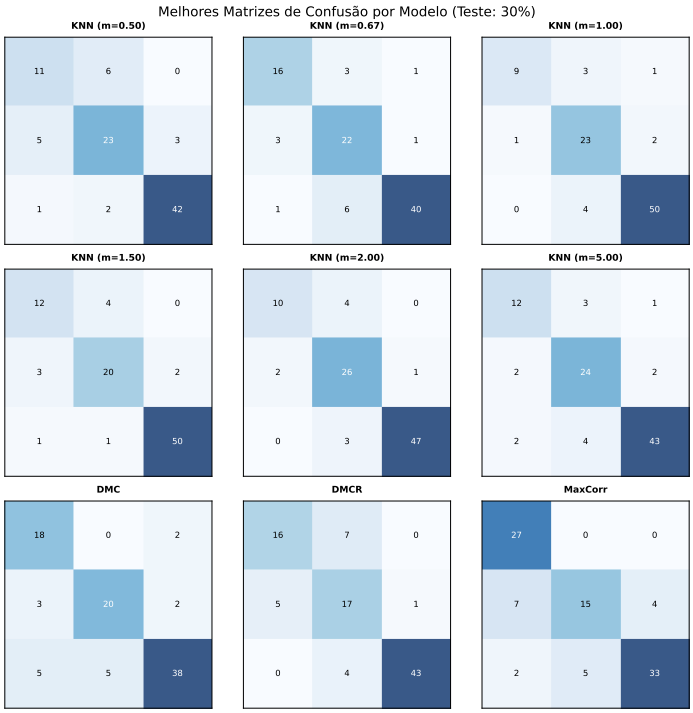
B.1 Melhor Matriz de confusão

Figura 7: Matriz de confusão das melhores rodadas de cada modelo, divisão de teste 20%.



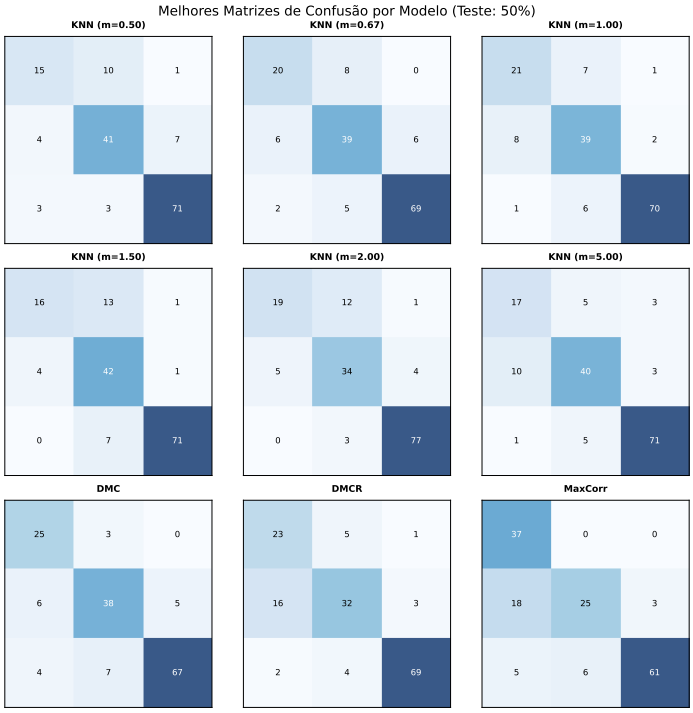
Fonte: Elaborado pelo autor

Figura 8: Matriz de confusão das melhores rodadas de cada modelo, divisão de teste 30%.



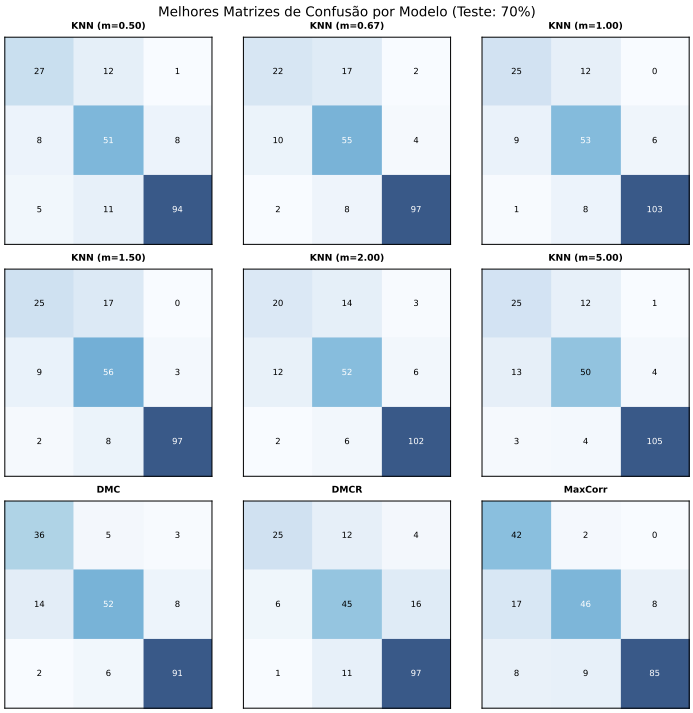
Fonte: Elaborado pelo autor

Figura 9: Matriz de confusão das melhores rodadas de cada modelo, divisão de teste 50%.



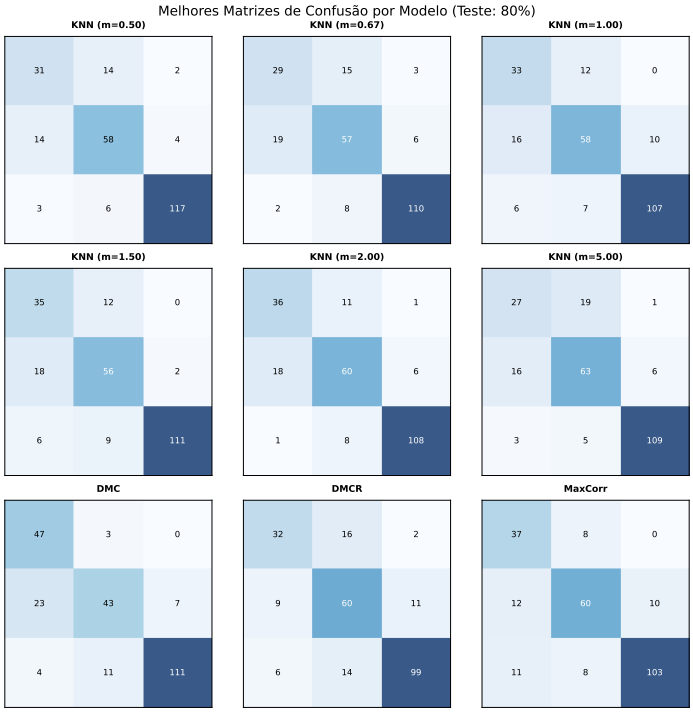
Fonte: Elaborado pelo autor

Figura 10: Matriz de confusão das melhores rodadas de cada modelo, divisão de teste 70%.



Fonte: Elaborado pelo autor

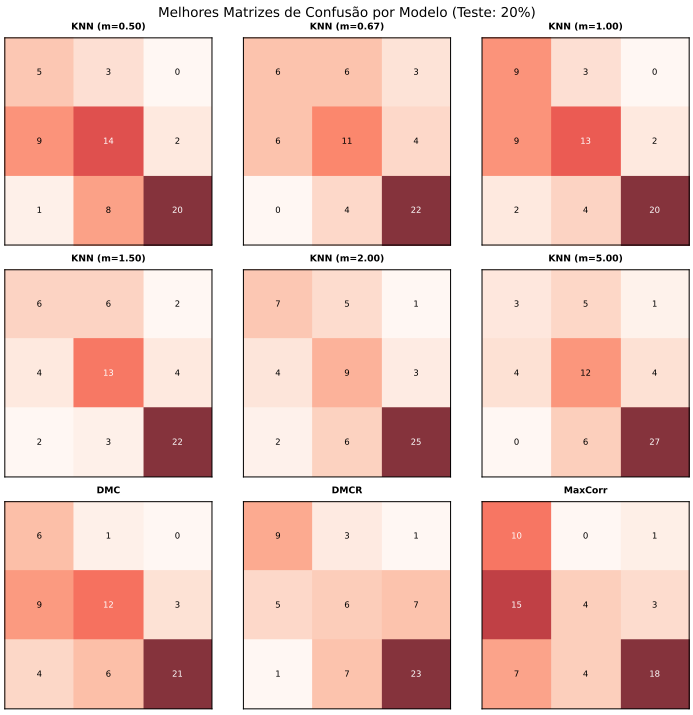
Figura 11: Matriz de confusão das melhores rodadas de cada modelo, divisão de teste 80%.



Fonte: Elaborado pelo autor

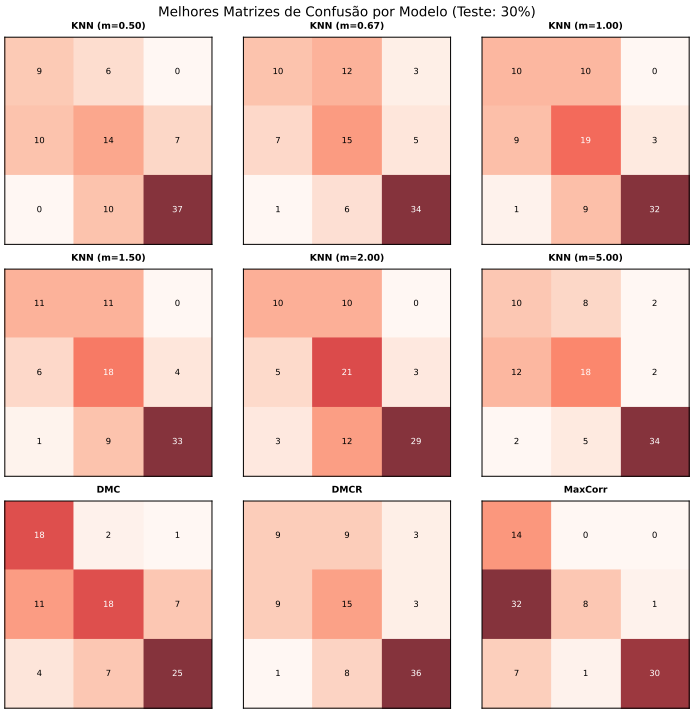
B.2 Piores Matrizes de confusão

Figura 12: Matriz de confusão das piores rodadas de cada modelo, divisão de teste 20%.



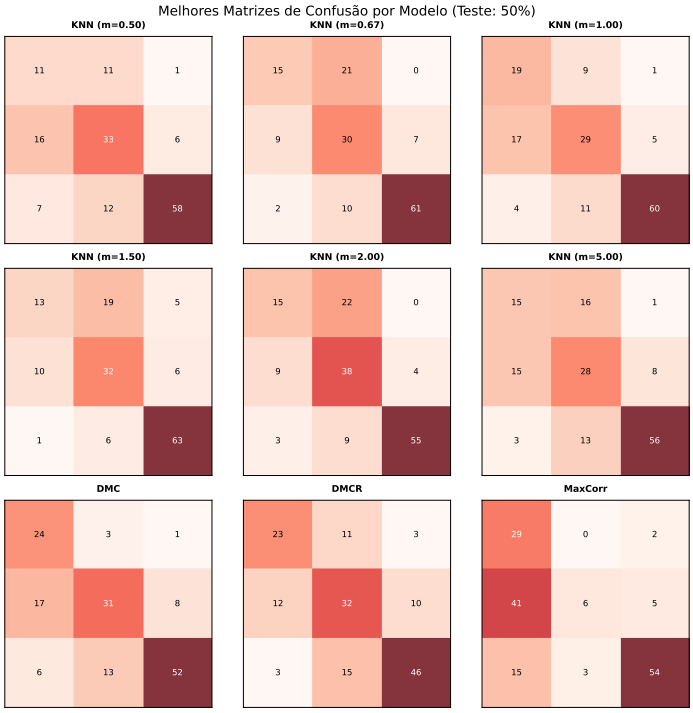
Fonte: Elaborado pelo autor

Figura 13: Matriz de confusão das piores rodadas de cada modelo, divisão de teste 30%.



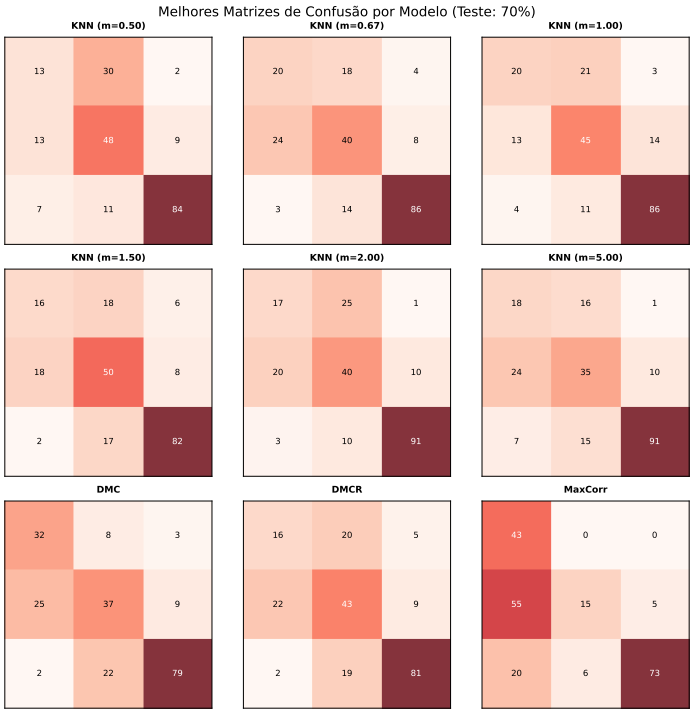
Fonte: Elaborado pelo autor

Figura 14: Matriz de confusão das piores rodadas de cada modelo, divisão de teste 50%.



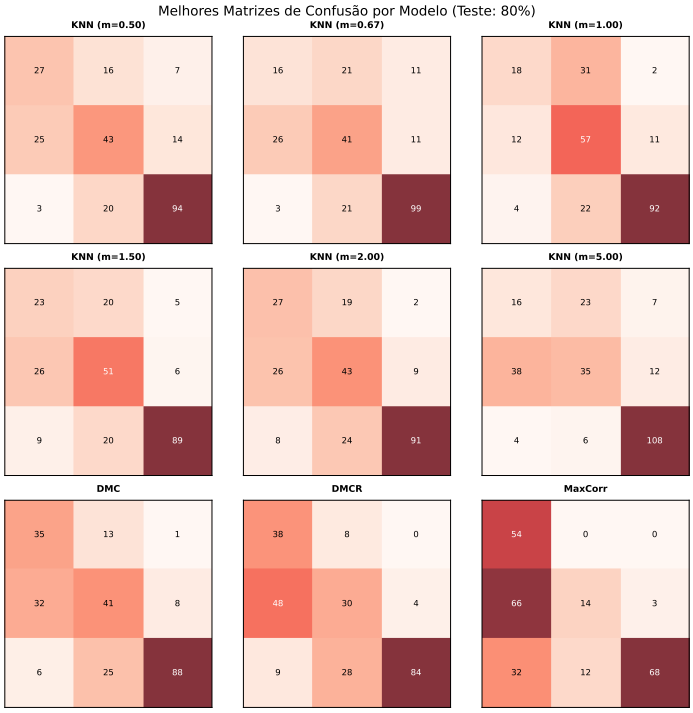
Fonte: Elaborado pelo autor

Figura 15: Matriz de confusão das piores rodadas de cada modelo, divisão de teste 70%.



Fonte: Elaborado pelo autor

Figura 16: Matriz de confusão das piores rodadas de cada modelo, divisão de teste 80%.



Fonte: Elaborado pelo autor