UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

ANA CLÁUDIA SEVERINO DE LIMA SANTOS
FRANCISCO DE ASSIS DE SOUZA RODRIGUES
ROBERTO DE MEDEIROS FARIAS FILHO
RONDINELLY DUARTE DE OLIVEIRA JÚNIOR

MULTIPLE FEATURES DATASET: UMA ANÁLISE DE CLASSIFICAÇÃO

RECIFE
2024

ANA CLÁUDIA SEVERINO DE LIMA SANTOS
FRANCISCO DE ASSIS DE SOUZA RODRIGUES
ROBERTO DE MEDEIROS FARIAS FILHO
RONDINELLY DUARTE DE OLIVEIRA JÚNIOR

MULTIPLE FEATURES DATASET: UMA ANÁLISE DE CLASSIFICAÇÃO

Projeto apresentado à disciplina IN1102 - Aprendizagem de Máquina, do Programa de Pós-graduação em Ciências da Computação, da Universidade Federal de Pernambuco, como requisito parcial para obtenção da nota final.

Docente: Prof. Dr. Francisco de Assis Tenorio de Carvalho

Recife-PE
2024

Resumo

Este estudo explora métodos de agrupamento e classificação de dados multidimensionais, combinando o algoritmo de agrupamento fuzzy baseado em kernel KFCM-K-W.1 com técnicas de *ensemble learning* e métodos de avaliação da partição fuzzy, como MPC e ARI. Para as bases `mfeat-fac` e `mfeat-zer`, os valores do MPC indicaram que a solução de agrupamento foi média, o ARI deu suporte a essa interpretaçã. Para a base `mfeat-fou` o resultado do MPC foi baixo, enquanto do ARI foi alto, sugerindo que a distribuição dos cluster pode não ser uniforme. Além disso, foi investigado o desempenho de quatro classificadores: bayesiano Gaussiano, bayesiano baseado em K-NN, bayesiano baseado na Janela de Parzen e Regressão Logística, utilizando a regra do voto majoritário para combinar suas predições. O desempenho da classificação para esses classificadores foi por meio de métricas como acurácia, precisão, revocação e f1-score. Para comparar o desempenho dos classificadores, foi aplicado o teste de Friedman e o teste *post-hoc* de Nemenyi, os quais indicaram que a regressão logística teve um melhor desempenho em relação aos demais, a um nível $\alpha = 0.05$.

Palavras-chave: classificadores bayesiano, *ensemble learning*, kFCM-K-W.1, teste de Friedman, teste de Nemenyi.

Abstract

This study explores methods of clustering and classifying multidimensional data, combining the kernel-based fuzzy clustering algorithm KFCM-K-W.1 with ensemble learning techniques and evaluation methods of fuzzy partition, such as MPC and ARI. For the `mfeat-fac` and `mfeat-zer` datasets, MPC values indicated that the clustering solution was average, supported by ARI interpretation. For the `mfeat-fou` dataset, MPC result was low while ARI was high, suggesting that the cluster distribution may not be uniform. Additionally, the performance of four classifiers was investigated: Gaussian Bayesian, K-NN based Bayesian, Parzen Window based Bayesian, and Logistic Regression, using majority voting rule to combine their predictions. Classification performance for these classifiers was assessed using metrics such as accuracy, precision, recall, and f1-mesure. To compare the classifiers' performance, Friedman test and Nemenyi post-hoc test were applied, indicating that logistic regression outperformed the others at a significance level of $\alpha = 0.05$.

Keywords: bayesian classifiers, ensemble learning, Friedman test, KFCM-K-W.1, Nemenyi test.

Lista de tabelas

Sumário

1 Introdução

Clustering é uma área de métodos de aprendizado não supervisionados que agrupa elementos de um conjunto de dados em conjuntos distintos, chamados de clusters, de modo que os objetos dentro de um mesmo cluster sejam mais parecidos entre si do que com objetos pertencentes a clusters diferentes, seguindo critérios específicos (HUANG et al., 2005). Vários estudos foram dirigidos em diferentes áreas do conhecimento envolvendo técnicas de agrupamento, como em genética (LOPEZ et al., 2018), engenharia (ZHANG; HAJJAR; SUN, 2020), educação (VANKAYALAPATI et al., 2021) e estatística (CARVALHO; NETO; SILVA, 2021).

As técnicas de agrupamento podem ser divididas em duas categorias: métodos hierárquicos e de particionamento. Os métodos hierárquicos fornecem uma saída representada por uma sequência aninhada de partições dos dados; sua saída é uma estrutura hierárquica de grupos conhecida como dendrograma (CHAVENT, 1998). Os métodos de particionamento visam obter uma única partição dos dados em um número fixo de clusters, geralmente com base na otimização de uma função objetivo adequada (CARVALHO et al. (2006), CARVALHO; LECHEVALLIER (2009)). Os métodos de particionamento foram desenvolvidos principalmente de duas maneiras diferentes: hard e fuzzy. No agrupamento hard, os clusters não se sobrepõem: qualquer ponto de dados pertence a um e apenas um cluster. O algoritmo de particionamento hard mais amplamente utilizado é o k-means. Já no clustering fuzzy, um ponto de dados pode pertencer a todos os clusters com uma certa medida de pertinência fuzzy. Essas associações fuzzy permitem que o algoritmo lide com clusters que se sobrepõem e distinga entre pontos no centro e nas margens de um cluster. O método mais conhecido de agrupamento difuso é o fuzzy c-means (FCM) (EZUGWU et al., 2022).

Os algoritmos de agrupamento que produzem uma divisão exclusiva, como o k-means (HUANG et al., 2005), são amplamente empregados em aplicações do mundo real para agrupar conjuntos de dados volumosos devido à sua eficácia e capacidade de lidar com variáveis numéricas e categóricas. Os métodos de particionamento visam criar uma única divisão dos dados de entrada em um número fixo de clusters. Geralmente, esses métodos buscam uma divisão que otimize localmente uma função objetivo apropriada. Para melhorar a qualidade do agrupamento, o algoritmo é executado várias vezes e a configuração que proporciona o melhor resultado de acordo com a função objetivo é selecionada como a divisão final em clusters.

Tanto o k-means quanto o FCM utilizam a distância euclidiana para calcular a diferença entre os pontos de dados e os representantes do cluster. Entretanto, SIMÕES; CARVALHO (2023) afirmam que eles não são eficazes em conjuntos de dados nos quais os clusters não são hiperesféricos e/ou linearmente separáveis. Para superar essa limitação,

várias abordagens foram propostas, incluindo métodos de agrupamento baseados em kernel. O algoritmo kernel fuzzy c-means (KFCM) (ZHANG; CHEN, 2004) emprega o conceito de "kernel trick", permitindo o cálculo de distâncias euclidianas no espaço de características usando kernels no espaço original.

Desde os primeiros avanços do algoritmo KFCM, foram propostos vários algoritmos de agrupamento fuzzy baseados em kernel. Para a primeira parte deste estudo, foi utilizado o algoritmo c-means fuzzy de kernel gaussiano com kernelização da métrica e cálculo automatizado de parâmetros de largura (KFCM-K-W.1), a fim de classificar os exemplos de três bases de dados de diferentes dimensões dos dados "Multiple features". Além disso, foram utilizadas as técnicas de validação das partições fuzzy como Coeficiente de Partição Modificado (MPC) (DAVE, 1996) e Índice de Rand Ajustado (ARI) (HUBERT; ARABIE, 1985).

A segunda parte deste estudo consiste em avaliar o desempenho de 4 classificadores (bayesiano Gaussiano, bayesiano baseado em K-vizinhos Mais Próximos, bayesiano baseado na Janela de Parzen e Regressão Logística), usando a regra do voto majoritário para os três conjunto de dados "Multiple features". Nessa etapa, foi utilizado o algoritmo k-means para realizar o agrupamento dos dados não supervisionados, seguindo os conceitos e metodologia do artigo de KITTLER et al. (1998). A avaliação da performance dos algoritmos foi realizada com base na metodologia do estudo de DEMŠAR (2006) para múltiplos classificadores. As métricas utilizadas, apropriadas para abordagens de partição cluster, podem ser encontradas no trabalho de MANNING; RAGHAVAN; SCHÜTZE (2008). As técnicas de avaliação envolvem o teste de Friedman (FRIEDMAN, 1937) e teste *post-hoc* de Nemenyi (NEMENYI, 1963).

2 Objetivos

Objetivo Geral

Este estudo tem como objetivo principal implementar o algoritmo KFCM-K-W.1, e comparar o desempenho de algoritmos de agrupamento e classificadores em três conjuntos de dados “Multiple featrures”, explorando abordagens tradicionais e técnicas avançadas de agrupamento e classificação.

Objetivos Específicos

1. Avaliar o desempenho do algoritmo KFCM-K-W.1, um algoritmo de agrupamento baseado em kernel fuzzy, na classificação de exemplos em três conjuntos de dados distintos.

2. Utilizar técnicas de validação das partições fuzzy, como o Coeficiente de Partição Modificado (MPC) e o Índice de Rand Ajustado (ARI), para avaliar a qualidade das partições produzidas pelo algoritmo KFCM-K-W.1.

3. Investigar o desempenho de quatro classificadores (bayesiano Gaussiano, bayesiano baseado em K-vizinhos Mais Próximos, bayesiano baseado na Janela de Parzen e Regressão Logística) na classificação dos conjuntos de dados "Multiple features".

4. Utilizar a regra do voto majoritário para combinar as predições dos classificadores e avaliar a eficácia dessa abordagem de ensemble.

5. Aplicar o teste de Friedman e o teste *post-hoc* de Nemenyi para avaliação do desempenho dos classificadores, seguindo a metodologia de DEMŠAR (2006).

3 Metodologia

Nesta seção, foram descritas as fundamentações teóricas por trás dos algoritmos e das técnicas utilizadas neste trabalho. As definições foram obtidas, principalmente, por meio dos trabalhos de SIMÕES; CARVALHO (2023), KITTLER et al. (1998), DEMŠAR (2006) e SHESKIN (2003). Para a implementação dos algoritmos, foram utilizadas algumas técnicas abordadas em SYAMSUL336 (2024), DITSKIH (2024b), DITSKIH (2024a), DAVISEK20 (2024), ARPIT512512 (2024). As distribuições de atividades pela equipe podem ser encontradas neste link: <https://github.com/rdmff/IN1102>, em que é possível encontrar Atas de reuniões, Script de algoritmos e andamento de todo desenvolvimento do projeto, do início ao fim.

3.1 Kernel gaussiano fuzzy c-means com cálculo de parâmetros de largura (KFCM-K-W.1)

O algoritmo KFCM-K-W, a partir de uma solução inicial, em três etapas, fornece iterativamente uma matriz $\mathbf{U} = (u_{ki})_{1\leq k\leq n}^{1\leq i\leq c}$ de graus de pertinência, parâmetros de largura para as variáveis e uma matriz de protótipos $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_c) = (g_{ij})_{1\leq i\leq c}^{1\leq j\leq p}$ dos clusters fuzzy pela minimização de uma função objetivo adequada, denotada como $J_{\text{KFCM-K-W}}$, que fornece a heterogeneidade total da partição fuzzy calculada como a soma da heterogeneidade em cada cluster fuzzy. Em SIMÕES; CARVALHO a função $J_{\text{KFCM-K-W}}$ é dada pela Equação 3.1.

$$J_{\text{KFCM-K-W}} = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m \|\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)\|^2 \text{ com } u_{ki} \geq 0 \text{ e } \sum_{i=1}^{c} u_{ki} = 1 \qquad (3.1)$$

em que $m > 1$ é o hiperparâmetro do fuzzificador que determina o nível de fuzificação do cluster.

O algoritmo itera até a minimização de $J_{\text{KFCM-K-W}}$, ajustando U, G e os parâmetros de largura automaticamente para encontrar a melhor partição fuzzy dos dados.

O cálculo de $\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)\|^2$ envolve o uso do truque do kernel de distância, já que o mapeamento não linear não é explicitamente conhecido. Para este estudo, foi abordada a variante do algoritmo KFCM-K-W, dada pela função de kernel Gaussiano denominada KFCM-K-W1.

A variante KFCM-K-W.1, introduzida por CARVALHO; SANTANA; FERREIRA (2018), é baseada em uma função de kernel Gaussiano com um vetor de parâmetros de largura global $\mathbf{s} = (s_1^2, \ldots, s_p^2)$ (Equação 3.2).

$$K^{(s)}(\mathbf{x}_l, \mathbf{x}_k) = \exp\left\{-\frac{1}{2}\sum_{j=1}^{p}\frac{(x_{l_j} - x_{k_j})^2}{s_j^2}\right\} \tag{3.2}$$

Note que cada variável tem seu próprio parâmetro de largura s_j^2 $(1 \leq j \leq p)$. Nesse caso, os autores definem a distância $\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)\|^2$ é dada pela Equação 3.3.

$$\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)\|^2 = K^{(s)}(\mathbf{x}_k, \mathbf{x}_k) - 2K^{(s)}(\mathbf{x}_k, \mathbf{g}_i) + K^{(s)}(\mathbf{g}_i, \mathbf{g}_i) \tag{3.3}$$

Como $K^{(s)}(\mathbf{x}_k, \mathbf{x}_k) = 1$, $\forall k$ e $K^{(s)}(\mathbf{g}_i, \mathbf{g}_i) = 1$, $\forall i$, temos que:

$$\|\Phi(\mathbf{x}_k) - \Phi(\mathbf{g}_i)\|^2 = 2 - 2K^{(s)}(\mathbf{x}_k, \mathbf{g}_i)$$

Com isso, a função objetivo torna-sea Equação 3.4.

$$J_{\text{KFCM-K-W.1}} = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m(2 - 2(K^{(s)}(\mathbf{x}_k, \mathbf{g}_i))) \tag{3.4}$$

sujeita a $u_{ki} \geq 0$ e $\sum_{i=1}^{c} u_{ki} = 1$.

3.1.1 Otimização da Função $J_{\text{KFCM-K-W.1}}$

3.1.1.1 Cálculo dos Parâmetros de Largura

Essa etapa fornece a solução ótima para o cálculo dos parâmetros de largura. Aqui, a matriz de protótipos $\mathbf{G}$ e a matriz $\mathbf{U}$ de graus de pertinência são mantidas fixas. É utilizado o método dos multiplicadores de Lagrange com a restrição $\prod_{j=1}^{p}\frac{1}{s_j^2} = 1$ (parâmetros de largura globais).

Em seguida, calcula-se as derivadas parciais das funções Lagrangianas em relação a $\frac{1}{s_j^2}$. Ao definir essas derivadas parciais igual a zero e aplicar alguma álgebra, a Equação 3.5 é a solução ótima para $\frac{1}{s_j^2}$ para a variante KFCM-K-W.1.

$$\frac{1}{s_j^2} = \frac{(\prod_{h=1}^{p}[\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m(K^{(s)}(\mathbf{x}_k, \mathbf{g}_i))(x_{kh} - g_{ih})^2])^{\frac{1}{p}}}{\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ki})^m(K^{(s)}(\mathbf{x}_k, \mathbf{g}_i))(x_{kj} - g_{ij})^2} \tag{3.5}$$

3.1.1.2 Cálculo dos Protótipos dos Clusters Fuzzy

Esta etapa fornece a solução ótima para o cálculo dos representantes dos clusters fuzzy. Aqui, o vetor global de parâmetros de largura $\mathbf{s}$ e a matriz $\mathbf{U}$ de graus de pertinência são mantidos fixos. A função objetivo $J_{\text{KFCM-K-W.1}}$ é otimizada em relação aos protótipos.

A partir de $\frac{\partial J_{\text{KFCM-K-W.1}}}{\partial \mathbf{g}_i} = 0$, a Equação 3.6 é a solução ótima para os protótipos dos clusters $\mathbf{g}_i$ para as variantes KFCM-K-W.1.

$$\mathbf{g}_i = \frac{\sum_{k=1}^{n}(u_{ki})^m(K^{(s)}(\mathbf{x}_k, \mathbf{g}_i))\mathbf{x}_k}{\sum_{k=1}^{n}(u_{ki})^m(K^{(s)}(\mathbf{x}_k, \mathbf{g}_i))}, \text{em que } (1 \leq i \leq x). \tag{3.6}$$

3.1.1.3 Cálculo dos Graus de Pertinência

Esta etapa fornece a solução ótima para o cálculo da matriz **U** dos graus de pertinência. Aqui, a matriz de protótipos **G** e o vetor global de parâmetros de largura **s** são mantidos fixos. Utiliza-se o método dos multiplicadores de Lagrange com a restrição $\sum_{i=1}^{c} u_{ki} = 1$, $u_{ki} \geq 0$ para calcular os graus de pertinência ótimos.

Em seguida, calculam-se as derivadas parciais das funções Lagrangianas em relação a u_{ki}. Ao definir essas derivadas parciais igual a zero, a Equação 3.7 é a solução ótima para o grau de pertinência u_{ki} para a variante KFCM-K-W.1.

$$u_{ki} = \left[\sum_{h=1}^{c} \left(\frac{2 - 2(K^{(s)}(\mathbf{x}_k, \mathbf{g}_i))}{2 - 2(K^{(s)}(\mathbf{x}_k, \mathbf{g}_h))} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{3.7}$$

3.2 Medidas de Validação de Partição Fuzzy

3.2.1 Coeficiente de Partição Modificado (MPC)

Considere $\{x_1, x_2, \ldots, x_N\}$ um conjunto de N pontos de dados representados por vetores de características n-dimensionais, ou seja, $\mathbf{x}_k \in \mathbf{R}^n$, $1 \leq k \leq N$. Representamos este conjunto como uma matriz **X** de dimensão $n \times N$. Cada coluna desta matriz representa um ponto de dados, e as linhas de **X** correspondem às n características. Por hipótese, **X** pode ser decomposta em $c \in (1, N)$ clusters difusos desconhecidos formando uma partição difusa de c. Uma partição fuzzy de c pode ser apropriadamente representada como uma matriz de dimensão $c \times N$ denotada por $\mathbf{U} = [u_{ik}]$, e referida como matriz de partição. Também assumimos que esta matriz é dada como saída do algoritmo usado para decompor **X**. Seu elemento genérico u_{ik} é o grau de pertinência de $\mathbf{x}_k$ ao cluster fuzzy i. As colunas de **U**, ou seja, os vetores

$$u_k = (u_{1k}, u_{2k}, \ldots, u_{ck}) \tag{3.8}$$

para $1 \leq k \leq N$, pertencem ao simplex unitário S_c

$$S_c = \left\{ (g_1, \ldots, g_c) : g_i \geq 0, \sum_{i=1}^{c} g_i = 1 \right\} \tag{3.9}$$

Este conjunto convexo é a contraparte geométrica da partição fuzzy. Os membros completos ou protótipos do cluster são representados pelos vetores de base canônicos de $\mathbf{R}^c$, e consequentemente localizam-se nos vértices ou pontos extremos de S_c. Chamamos

$$\mathbf{c} = \left(\frac{1}{c}, \frac{1}{c}, \ldots, \frac{1}{c} \right) \tag{3.10}$$

o centro da partição difusa de c ou, equivalentemente, de S_c; ele corresponde ao ponto mais difuso. Agora suponha que temos uma coleção de partições fuzzy de c de **X**, por exemplo,

$c = 2, 3, \ldots$, e o objetivo é avaliar quão bem cada partição se ajusta aos dados; em outras palavras, queremos saber como selecionar a melhor estimativa entre os candidatos que particionam $\mathbf{X}$. DAVE (1996) apresenta uma medida escalar ou índice de validade de cluster, chamado coeficiente de partição (VPC), dado por

$$VPC = \frac{1}{N} \sum_{k=1}^{N} \sum_{i=1}^{c} u_{ik}^2 \tag{3.11}$$

para avaliar as partições fuzzy competidoras de $\mathbf{X}$. Pode-se mostrar que $VPC \in \left[\frac{1}{c}, 1\right]$. O extremo inferior é atingido se $\mathbf{U} = \left[\frac{1}{c}\right]$, enquanto o valor $VPC = 1$ é mantido se a partição for rígida, ou seja, quando todos os pontos de dados são mapeados nos vértices do simplex unitário S_c. O VPC mede quão longe $\mathbf{U}$ está de ser uma matriz de partição nítida. Quanto maiores os valores de VPC, mais rígido é $\mathbf{U}$. Apesar de sua construção conceitual atrativa e simplicidade, o índice VPC tem a desvantagem de tender a aumentar monotonamente com o número de clusters c.

DAVE (1996) propôs uma modificação da Equação 3.16, por meio de uma transformação linear, para eliminar essa dependência de VPC em c. Ele se refere à nova medida assim obtida como coeficiente de partição modificado (MPC), que é expresso como

$$MPC = \left(\frac{c}{c-1} \times VPC\right) - \frac{1}{c-1} \tag{3.12}$$

A faixa de MPC é o intervalo unitário $[0, 1]$, onde $MPC = 0$ corresponde ao máximo de difusão e $MPC = 1$ a uma partição rígida. Essa característica torna o MPC adicionalmente mais atraente do que o VPC para comparar diferentes soluções de cluster para a matriz de dados $\mathbf{X}$, já que não depende do número de clusters como faz a faixa de VPC na Equação 3.11. Em geral, o número ótimo de clusters é encontrado ao resolver $max_c MPC$.

3.2.2 Índice de Rand Ajustado (ARI)

O Índice Rand Ajustado (ARI) é uma medida da similaridade entre dois agrupamentos de dados. Ele é uma correção do Índice Rand, que é uma medida básica de similaridade entre dois agrupamentos, mas tem a desvantagem de ser sensível ao acaso. O ARI leva em consideração o fato de que algum acordo entre dois agrupamentos pode ocorrer por acaso, e ajusta o Índice Rand para levar em conta essa possibilidade. É calculado da seguinte forma:

- 1. Seja N o número de amostras no conjunto de dados.
- 2. Sejam $C1$ e $C2$ dois agrupamentos diferentes do conjunto de dados.
- 3. Seja a o número de pares de amostras que estão no mesmo agrupamento tanto em $C1$ quanto em $C2$.

- 4. Seja b o número de pares de amostras que estão em agrupamentos diferentes em $C1$ e $C2$.
- 5. Calcule o Índice Rand como $RI = \frac{(a+b)}{N2}$, em que $\binom{N}{2}$ é o número de pares possíveis de amostras.
- 6. Calcule o valor esperado E do Índice Rand para agrupamentos aleatórios, dado por: $E = \frac{(\sum \binom{n_i}{2})(\sum \binom{n_j}{2}))}{N2}$, em que n_i é o número de amostras no agrupamento i e n_j é o número de amostras no agrupamento j.
- 7. Calcule o Índice Rand Ajustado como $ARI = \frac{(RI-E)}{(\max(RI)-E)}$, em que $\max(RI) = 1$.

Quanto maior o valor de ARI, mais próximos estão os dois agrupamentos. Ele varia de -1 a 1, onde 1 indica concordância perfeita entre os dois agrupamentos, 0 indica uma concordância aleatória e -1 indica que os dois agrupamentos são completamente diferentes. O ARI é amplamente utilizado em aprendizado de máquina, mineração de dados e reconhecimento de padrões, especialmente para a avaliação de algoritmos de agrupamento (HUBERT; ARABIE, 1985).

3.3 *Framework* Teórico

Considere um problema de reconhecimento de padrões em que o padrão Z deve ser atribuído a uma das m classes possíveis $(\omega_1, \ldots, \omega_m)$. Vamos supor que temos R classificadores, cada um representando o padrão dado por um vetor de medição distinto. Denotamos o vetor de medição usado pelo i-ésimo classificador por $\mathbf{x}_i$. No espaço de medição, cada classe ω_k é modelada pela função de densidade de probabilidade $p(\mathbf{x}_i|\omega_k)$ e sua probabilidade a priori de ocorrência é denotada por $P(\omega_k)$. Consideraremos os modelos como mutuamente exclusivos, o que significa que apenas um modelo pode ser associado a cada padrão.

Agora, de acordo com a teoria Bayesiana, dadas as medições $\mathbf{x}_i$, $i = 1, \ldots, R$, o padrão Z deve ser atribuído à classe ω_k desde que a probabilidade a posteriori dessa interpretação seja máxima, ou seja, atribua Z a ω_j se a Equação 3.13 for satisfeita.

$$P(\omega_j|\mathbf{x}_1, \ldots, \mathbf{x}_R) = \max_k P(\omega_k|\mathbf{x}_1, \ldots, \mathbf{x}_R) \tag{3.13}$$

A regra de decisão Bayesiana (3.13) afirma que, para utilizar todas as informações disponíveis corretamente para chegar a uma decisão, é essencial calcular as probabilidades das várias hipóteses considerando todas as medições simultaneamente. Isso é, claro, uma afirmação correta do problema de classificação, mas pode não ser uma proposição praticável. O cálculo das funções de probabilidade a posteriori dependeria do conhecimento das estatísticas de medição de ordem superior descritas em termos de funções de densidade de probabilidade conjunta $p(\mathbf{x}_1, \ldots, \mathbf{x}_k|\omega_k)$, o que seria difícil de inferir. Portanto, vamos

tentar simplificar a regra acima e expressá-la em termos de computações de suporte à decisão realizadas pelos classificadores individuais, cada um explorando apenas as informações transmitidas pelo vetor $\mathbf{x}_i$. Veremos que isso não só tornará a regra (3.13) computacionalmente gerenciável, mas também levará a regras de combinação comumente usadas na prática. Além disso, essa abordagem fornecerá um escopo para o desenvolvimento de uma variedade de estratégias eficientes de combinação de classificadores.

Vamos começar com a regra (3.13) e considerar como ela pode ser expressa sob certas suposições. Vamos reescrever a probabilidade a posteriori $P(\omega_k|\mathbf{x}_1, \ldots, \mathbf{x}_R)$ usando o teorema de Bayes. Temos a Equação 3.14.

$$P(\omega_k|\mathbf{x}_1, \ldots, \mathbf{x}_R) = \frac{p(\mathbf{x}_1, \ldots, \mathbf{x}_R|\omega_k P(\omega_k)}{p(\mathbf{x}_1, \ldots, \mathbf{x}_R} \tag{3.14}$$

onde $p(\mathbf{x}_1, \ldots, \mathbf{x}_R)$ é a densidade de probabilidade conjunta de medição incondicional. Esta última pode ser expressa em termos das distribuições condicionais de medição como a Equação 3.15.

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_R) = \sum_{j=1}^{m} p(\mathbf{x}_1, \ldots, \mathbf{x}_R|\omega_j)P(\omega_j) \tag{3.15}$$

Portanto, a seguir, podemos nos concentrar apenas nos termos do numerador de (3.14).

3.3.1 Regra do Produto

Como já mencionado, $p(\mathbf{x}_1, \ldots, \mathbf{x}_R|\omega_k)$ representa a distribuição de probabilidade conjunta das medidas extraídas pelos classificadores. Vamos supor que as representações usadas sejam condicionalmente estatisticamente independentes. O uso de diferentes representações pode ser uma causa provável dessa independência em casos especiais. Investigaremos as consequências dessa suposição e escreveremos a Equação 3.16.

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_R|\omega_k) = \prod_{i=1}^{R} p(\mathbf{x}_i|\omega_k) \tag{3.16}$$

onde $p(\mathbf{x}_i|\omega_k)$ é o modelo do processo de medição da i-ésima representação. Substituindo a partir da (Equação 3.16) e da (Equação 3.15) na (Equação 3.14), encontramos a Equação 3.17.

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_R|\omega_k) = \frac{P(\omega_k)\prod_{i=1}^{R} p(\mathbf{x}_i|\omega_k)}{\sum_{j}^{m} P(\omega_j)\prod_{i=1}^{R} p(\mathbf{x}_i|\omega_j)} \tag{3.17}$$

e, utilizando a (Equação 3.17) na (Equação **??** Eq. 1), obtemos a regra de decisão: atribua Z a ω_j se a Equação 3.18 for satisfeita.

$$P(\omega_j)\prod_{i=1}^{R} p(\mathbf{x}_i|\omega_j) = \max_{k=1}^{m} P(\omega_k)\prod_{i=1}^{R} p(\mathbf{x}_i|\omega_k) \tag{3.18}$$

ou, em termos das probabilidades a posteriori geradas pelos respectivos classificadores: atribua Z a ω_j se a Equação 3.19 for satisfeita.

$$P^{-(R-1)}(\omega_j)\prod_{i=1}^{R}P(\omega_j|\mathbf{x}_i) = \max_{k=1}^{m} P^{-(R-1)}(\omega_k)\prod_{i=1}^{R}p(\omega_k|\mathbf{x}_i) \tag{3.19}$$

A regra de decisão (Equação 3.19) quantifica a probabilidade de uma hipótese combinando as probabilidades a posteriori geradas pelos classificadores individuais por meio de uma regra de produto. É efetivamente uma regra rigorosa de fusão das saídas dos classificadores, pois basta que um único motor de reconhecimento iniba uma interpretação específica emitindo uma probabilidade próxima de zero para ela. Como veremos a seguir, isso tem uma implicação bastante indesejável na combinação de regras de decisão, pois todos os classificadores, no pior caso, terão que fornecer suas respectivas opiniões para que a identidade de uma classe hipotetizada seja aceita ou rejeitada.

3.3.2 Regra da Soma

Vamos considerar a regra de decisão (Equação 3.19) em mais detalhes. Em algumas aplicações, pode ser apropriado assumir que as probabilidades a posteriori calculadas pelos respectivos classificadores não se desviarão dramaticamente das probabilidades a priori. Esta é uma suposição bastante forte, mas pode ser prontamente satisfeita quando a informação discriminatória observacional disponível é altamente ambígua devido a altos níveis de ruído. Nessa situação, podemos assumir que as probabilidades a posteriori podem ser expressas conforme a Equação 3.20.

$$P(\omega_k|\mathbf{x}_i = P(\omega_k)(1+\delta_{ki}) \tag{3.20}$$

onde δ_{ki} satisfaz $\delta_{ki} << 1$. Substituindo a Equação 3.20 para as probabilidades a posteriori na Equação 3.19, obtemos a Equação 3.21.

$$P^{-(R-1)}(\omega_k)\prod_{i=1}^{R}P(\omega_k|\mathbf{x}_i) = P(\omega_k)\prod_{i=1}^{R}(1+\delta_{ki}) \tag{3.21}$$

Se expandirmos o produto e negligenciarmos quaisquer termos de segunda ordem e superiores, podemos aproximar o lado direito da Equação 3.21 como a Equação 3.22.

$$P(\omega_k)\prod_{i=1}^{R}(1+\delta_{ki}) = P(\omega_k) + P(\omega_k)\sum_{i=1}^{R}\delta_{ki} \tag{3.22}$$

Substituindo a Equação 3.22 e a Equação 3.20 na Equação 3.19, obtemos uma regra de decisão de soma: atribua Z a ω_j se a Equação 3.23 for satisfeita.

$$(1-R)P(\omega_j) + \sum_{i=1}^{R}P(\omega_j|\mathbf{x}_i) = \max_{k=1}^{m}\left[(1-R)P(\omega_k) + \sum_{i=1}^{R}P(\omega_k|\mathbf{x}_i)\right]. \tag{3.23}$$

3.3.3 Estratégias de Combinação de Classificadores

As regras de decisão (Equação 3.19) e (Equação 3.23) constituem os esquemas básicos para a combinação de classificadores. Interessantemente, muitas estratégias de combinação de classificadores comumente usadas podem ser desenvolvidas a partir dessas regras, observando a Equação 3.24.

$$\prod_{i=1}^{R} P(\omega_k|\mathbf{x}_i) \leq \min_{i=1}^{R} P(\omega_k|\mathbf{x}_i) \leq \frac{1}{R}\sum_{i=1}^{R} P(\omega_k|\mathbf{x}_i) \leq \max_{i=1}^{m} P(\omega_j|\mathbf{x}_i) \tag{3.24}$$

A relação (Equação 3.24) sugere que as regras de combinação de produto e soma podem ser aproximadas pelos limites superior ou inferior acima mencionados, conforme apropriado. Além disso, o endurecimento das probabilidades a posteriori $P(\omega_j|\mathbf{x}_i)$ para produzir funções de valor binário Δ_{ki} conforme a Equação 3.25.

$$\Delta_{ki} = \begin{cases} 1 & \text{if } P(\omega_k|\mathbf{x}_i) = \max_{j=1}^{m} P(\omega_j|\mathbf{x}_i) \\ 0 & \text{otherwise} \end{cases} \tag{3.25}$$

resulta na combinação dos resultados das decisões em vez da combinação das probabilidades a posteriori. Essas aproximações levam à seguinte regra na Subseção 3.3.4.

3.3.4 Regra do Voto Majoritário

Partindo da (Equação 3.23) sob a suposição de priores iguais e ao endurecer as probabilidades de acordo com (Equação 3.25), encontramos: atribua Z a ω_j se a Equação 3.26 for satisfeita.

$$\sum_{i=1}^{R} \Delta_{ji} = \max_{k=1}^{m} \sum_{i=1}^{R} \Delta_{ki} \tag{3.26}$$

Note que para cada classe ω_k, a soma do lado direito da (Equação 3.26) simplesmente conta os votos recebidos para essa hipótese dos classificadores individuais. A classe que recebe o maior número de votos é então selecionada como a decisão de consenso (maioria).

A probabilidade a posteriori é então calculada (veja a Equação 3.27).

$$P(\omega_i|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_i)P(\omega_i)}{\sum_k p(\mathbf{x}|\omega_k)P(\omega_k)} \tag{3.27}$$

3.4 Classificadores

3.4.1 Bayesiano Gaussiano

Aqui as classes no espaço de características são assumidas como possuindo uma distribuição normal (veja a Equação 3.28).

$$p(\mathbf{x}|\omega_i) = (2\pi)^{-\frac{d}{2}}|\Sigma_i|^{-\frac{1}{2}} \exp -(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i) \tag{3.28}$$

em que μ_i é o vetor médio e Σ_i é a matriz de covariância da classe i. Eles são estimados na fase de treinamento a partir do conjunto de dados de treinamento. d é o número de dimensões no espaço de características.

3.4.2 Bayesiano Baseado na Janela de Parzen

Considere uma variável aleatória X com densidade p. Então, a densidade $p(x)$ pode ser definida como o limite, quando h tende a zero, da expressão $\frac{1}{2h}P(x-h<X<x+h)$.

Estimando $P(x-h<X<x+h)$ para um h fixo e pequeno, obtemos a proporção de observações da amostra pertencentes ao intervalo $(x-h, x+h)$.

O estimador de $p(x)$, para um h fixo e pequeno, é dado por $\hat{p}(x) = \frac{1}{2nh}\#\{X_i \in (x-h, x+h)\}$.

A função janela $w(x)$, como definida na Equação 3.29, é uma função que tem valor de $\frac{1}{2}$ quando $|x|<1$ e zero caso contrário.

$$w(x) = \begin{cases} \frac{1}{2} & \text{se } |x|<1 \\ 0 & \text{caso contrário} \end{cases} \tag{3.29}$$

Substituindo a função janela na Equação 3.30, obtemos o estimador de p.

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}w\left(\frac{x-x_i}{h}\right) \tag{3.30}$$

O estimador é construído colocando-se um retângulo de largura $2h$ e altura $(2nh)^{-1}$ em cada observação e em seguida somando-se.

É importante notar que $\hat{p}(x)$ não é uma função contínua e tem derivada nula em todos os pontos exceto nos pontos de "salto"$x \pm h$, além de depender fortemente de h.

3.4.3 Bayesiano Baseado em k-Vizinhos Mais Próximos

Considerando uma amostra de n exemplos $\mathbf{x}_1, \ldots, \mathbf{x}_n$ identicamente distribuídos de acordo com $p(x)$ e obtidos independentemente. Temos c classes a priori: $\omega_1, \ldots, \omega_c$. Na amostra de n exemplos, estão presentes n_j exemplos da classe ω_j $(j = 1, \ldots, c)$, onde $n = \sum_{j=1}^{c} n_j$.

Suponha que em k exemplos (com $k << n$) existem k_j exemplos da classe ω_j, onde $k = \sum_{j=1}^{c} k_j$. Estimamos a densidade condicional como $\hat{p}(\mathbf{x}|\omega_j) = \frac{(\frac{k_j}{n_j})}{V}$ e a probabilidade a priori de ω_j como $\hat{P}(\omega_j) = \frac{n_j}{n}$. A regra de decisão é atribuir $\mathbf{x}$ à classe ω_j se $\hat{P}(\omega_j|\mathbf{x}) \geq \hat{P}(\omega_l|\mathbf{x}) \forall l \neq j$, ou seja, se $\frac{\frac{k_j}{n_j}}{V}\frac{n_j}{n} \geq \frac{\frac{k_l}{n_l}}{V}\frac{n_l}{n} \forall l \neq j$, ou ainda, se $k_j \geq k_l \forall l \neq j$.

A função de classificação é definida como $f : \mathbf{R}^p \to \{1, \ldots, c\} : y = f(\mathbf{x}) \in \{1, \ldots, c\}$. Para um exemplo e descrito por $\mathbf{x}$ e seus k vizinhos mais próximos, a função de classificação $\hat{f}(\mathbf{x}) = \arg\max_{\{} y \in \{1, \ldots, c\} \sum_{i=1}^{k} \delta(y, y_i)$, onde $\delta(y, y_i)$ é definida conforme a Equação 3.31.

$$\delta(y, y_i) = \begin{cases} 1 & \text{se } y = y_i \\ 0 & \text{se } y \neq y_i \end{cases} \tag{3.31}$$

A escolha da métrica e determinação da vizinhança podem ser feitas utilizando a distância Euclidiana $d(\mathbf{x}, \mathbf{z}) = \sqrt{\sum_{l=1}^{p}(x_l - z_l)^2}$. Quanto ao valor de k, geralmente é escolhido como um número pequeno e ímpar, como $k = 1, 3, 5, \ldots$, podendo também ser limitado por $\sqrt{n}$ ou estimado por validação cruzada.

É possível associar pesos à contribuição de cada vizinho. Na ponderação local, os pesos α_i são inversamente proporcionais à distância entre o exemplo $\mathbf{x}$ e seus vizinhos mais próximos, como definido na Equação 3.33:

$$\delta(y, y_i) = \begin{cases} 1 & \text{se } y = y_i \\ 0 & \text{se } y \neq y_i \end{cases} \tag{3.32}$$

$$\alpha_i = \frac{1}{d(\mathbf{x}, \mathbf{x}_i)} \tag{3.33}$$

em que $\delta(y, y_i)$ é definido conforme a Equação 3.32.

Já na ponderação global, os pesos α_i são definidos da mesma maneira, considerando todos os exemplos na amostra:

$$\delta(y, y_i) = \begin{cases} 1 & \text{se } y = y_i \\ 0 & \text{se } y \neq y_i \end{cases} \tag{3.34}$$

$$\alpha_i = \frac{1}{d(\mathbf{x}, \mathbf{x}_i)} \tag{3.35}$$

de forma que a soma dos pesos seja feita para todos os exemplos na amostra, conforme a Equação 3.34.

3.4.4 Regressão Logística

A regressão logística, baseada no princípio da máxima verossimilhança, é uma abordagem fundamental na modelagem de dados binários, onde estamos interessados na probabilidade de um evento ocorrer ou não. Inicialmente, consideramos um conjunto de itens ou exemplos representados por $E = \{e_1, \ldots, e_i, \ldots, e_m\}$. Cada exemplo consiste em um par $(\tilde{\mathbf{x}}_i, y_i)$, onde $y_i \in \{0, 1\}$ indica a classe do exemplo e $\tilde{\mathbf{x}}_i = (x_{i1}, \ldots, x_{in})^T \in \mathbf{R}^n$ representa os atributos. Introduzimos $\mathbf{x}_i = (1, x_{i1}, \ldots, x_{in})^T \in \mathbf{R}^{n+1}$ para facilitar o cálculo.

O modelo de regressão logística estima a probabilidade condicional de y dado $\mathbf{x}$ usando uma função logística $h_\theta(\mathbf{x}^T\theta)$, onde θ é um vetor de parâmetros. A função logística é

definida como $h_\theta(\mathbf{x}^T\theta) = \frac{\exp(\mathbf{x}^T\theta)}{1+\exp(\mathbf{x}^T\theta)}$. A partir do modelo, podemos calcular a probabilidade de y ser 1 ou 0 para um dado $\mathbf{x}$.

A verossimilhança dos dados dado θ é obtida multiplicando as probabilidades condicionais de cada exemplo. A log-verossimilhança é a função que maximizamos durante o treinamento do modelo. Para isso, utilizamos a função de custo, que é a versão negativa da log-verossimilhança. Nosso objetivo é minimizar essa função de custo ajustando os parâmetros θ.

O algoritmo de gradiente descendente é frequentemente empregado para encontrar os valores ótimos de θ. Este algoritmo atualiza iterativamente os parâmetros, movendo-se na direção oposta do gradiente da função de custo. Além disso, para problemas de classificação com mais de duas classes, uma abordagem "um contra todos" é comumente adotada, onde treinamos vários classificadores de regressão logística, um para cada classe, e selecionamos a classe com a maior probabilidade prevista para um dado exemplo. Essa abordagem é útil para problemas de classificação multiclasse.

3.5 Métricas

Quando se deseja orientar a tomada de decisão por meio da modelagem, é essencial empregar métricas para avaliar a eficácia dos modelos de classificação, ou seja, sua capacidade de se ajustar aos dados. Essas métricas fornecem valores numéricos que expressam o grau de precisão e erro nas previsões feitas pelo modelo.

A princípio, é necessário compreender o conceito de uma matriz de confusão, também conhecida como matriz de erro. Essa matriz organiza as frequências de categorias que representam tanto os acertos (quando as predições coincidem com os valores reais) quanto os erros (quando os valores preditos são diferentes dos valores reais) feitos pelo modelo.

Tabela 1 – Matriz de confusão

Matriz de Confusão		**Predito**	
		Sim	**Não**
Real	**Sim**	Verdadeiro Positivo (VP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (VN)

Fonte: Adaptação de LUQUE et al.(2019).

A Tabela 1 ailustra a configuração de uma matriz de confusão para um cenário com duas classes, que é um caso específico de uma matriz de confusão para n classes, conforme descrito no exemplo teórico do livro de CONGALTON; GREEN (2019). Nas linhas, estão representadas as classificações reais, enquanto nas colunas, estão representadas as classificações previstas pelo modelo. Nesse contexto, os seguintes termos são aplicáveis:

- Verdadeiro Positivo (VP): ocorre quando o modelo classifica corretamente uma instância como pertencente à classe Positivo; ou seja, o modelo acerta ao prever que uma instância pertence à classe Positivo quando realmente pertence.
- Falso Positivo (FP): ocorre quando o modelo erroneamente prevê que uma instância pertence à classe Positivo, mas na verdade ela pertence à classe Negativo.
- Verdadeiro Negativo (VN): ocorre quando o modelo classifica corretamente uma instância como pertencente à classe Negativo; ou seja, o modelo acerta ao prever que uma instância pertence à classe Negativo quando realmente pertence.
- Falso Negativo (FN): ocorre quando o modelo erroneamente prevê que uma instância pertence à classe Negativo, mas na verdade ela pertence à classe Positivo.

3.5.1 Acurácia

A medida de acurácia (chamemos de AC) reflete o quão bem os resultados de um teste se aproximam de um ponto de referência, indicando o quanto o modelo foi capaz de classificar os dados corretamente. Considerando a matriz de confusão da Tabela 1, essa métrica pode ser obtida pela Equação 3.36.

$$AC = \frac{VP + VN}{VP + VN + FP + FN} \tag{3.36}$$

3.5.2 Precisão

De acordo com MONICO et al. (2009), a distinção entre acurácia e precisão surge da existência de erros sistemáticos, conhecidos como tendência ou viés. Na matriz de confusão representada na Figura 1, a precisão (chamemos de P) é definida como a proporção de predições corretas feitas pelo modelo em relação ao total de predições positivas. O resultado dessa métrica pode ser obtido pela Equação 3.37.

$$P = \frac{VP}{VP + FP} \tag{3.37}$$

3.5.3 Recall/Revocação

A métrica recall, também chamada de revocação, sensibilidade, ou taxa de verdadeiro positivo (chamemos de R), representa a porcentagem de classificações corretas relacionadas às quantidades reais, ou seja, o quanto dos valores reais o modelo identificou corretamente como positivos. Considerando a matriz de confusão da Tabela 1, o cálculo dessa métrica pode ser obtido pela fórmula da Equação 3.38.

$$R = \frac{VP}{VP + FN} \tag{3.38}$$

3.5.4 F1-score

O F1-score é uma medida que combina precisão e recall em uma única métrica, fornecendo uma visão equilibrada do desempenho do modelo. Ele é calculado como a média harmônica entre essas métricas, permitindo avaliar ambas as medidas simultaneamente. Um baixo valor de F1-score indica que tanto a precisão quanto a métrica recall podem estar baixos. Considerando a Tabela 1, o cálculo pode ser encontrado pela Equação 3.39.

$$F1\text{-}score = \frac{2 \times P \times R}{P + R} \tag{3.39}$$

3.6 Testes para Comparar Múltiplos Classificadores

No cenário de pesquisa de aprendizado de máquina, geralmente, diferentes algoritmos são testados em vários conjuntos de dados para avaliar seu desempenho em diferentes contextos. No entanto, simplesmente observar as métricas de desempenho em cada conjunto de dados pode não ser suficiente, pois pode haver variações aleatórias nos resultados devido às características específicas dos conjuntos de dados GARCIA; HERRERA.

Comparar múltiplos classificadores é uma prática comum em aprendizado de máquina e análise estatística, especialmente em problemas de classificação. Existem várias métricas (ver Seção 3.5) que descrevem o desempenho de um classificador. DEMŠAR (2006),(1940) apresenta uma série de testes estatísticos que possibilitam a comparação entre diferentes classificadores, e a escolha do teste está relacionada às caracteristicas dos dados, número de classificadores a serem comparados. Este estudo aborda o teste de Friedman, o teste de Nemenyi e o teste de Bonferroni-Dunn para comparar os 6 algoritmos implementados.

3.6.1 Teste de Friedman

O teste de Friedman é uma técnica estatística não pramétrica usada para determinar se há diferenças estatísticamente significativas entre múltiplos grupos de dados realcionados. Ele é uma extensão do teste ANOVA para dados pareados e não paramétricos (FRIEDMAN, 1937).

Ele permite determinar se há diferenças significativas no desempenho médio dos algoritmos ao longo de múltiplos conjuntos de dados. Ao rejeitar a hipótese nula de que não há diferenças entre os algoritmos, o teste de Friedman indica que pelo menos um dos algoritmos tem um desempenho estatisticamente diferente dos outros. Ele classifica os algoritmos para cada conjunto de dados separadamente, o algoritmo com melhor desempenho recebe a classificação 1, o segundo melhor recebe a classificação 2, e assim por diante, a Tabela 9 exemplifica isso. Em caso de empate são atribuídas classificações médias.

Considere uma hipótese nula (H_0) e uma hipótese alternativa (H_1) com base no problema em estudo (comparação de classificadores). Ambas as hipóteses devem ser declarações concisas, duas versões da verdade. Essas versões são mutuamente exclusivas e exaustivas (abrangem todas as possibilidades).

$$\begin{cases} H_0 = \text{não há diferença entre classificadores.} \\ H_1 = \text{há diferença entre classificadores.} \end{cases}$$

Seja r_i^j o rank da j-ésimo algoritmo na i-ésima linha de N. O teste de Friedman compara a média dos ranks dos algoritmos, $R_j = \frac{1}{N}\sum_i r_i^j$. Sob a hipótese nula, que afirma que todos os algoritmos são equivalentes e, então, seus ranks R_j devem ser iguais. A estatística de Friedman é distribuída de acordo com χ_F^2 com $k1$ graus de liberdade, quando N e k são suficientemente grandes. Segundo SHESKIN (2003), a estatística de Friedman é expressão por meio da Equação 3.40.

$$\chi^2_{\text{calculado}} = \frac{12}{Nk(k+1)}\left[\sum_j^k (\sum R_j)^2\right] - 3N(k+1), \tag{3.40}$$

em que $\chi^2_{\text{calculado}}$ é a estatística do teste; N é o número de observações (linhas); k é o número de tratamentos (algoritmos); e R_j é média dos postos dos tratamentos (algoritmos) no j-ésimo grupo de observações.

Como a estatística de Friedman segue uma distribuição qui-quadrado com $k-1$ graus de liberdade, o valor $\chi^2_{\text{calculado}}$ será comparado com os valores críticos da distribuição qui-quadrado ($\chi^2_{\text{crítico}}$) para determinar se há diferenças estatisticamente significativas entre os tratamentos. Se $\chi^2_{\text{calculado}}$ for maior que o valor crítico correspondente com um nível de significância escolhido, geralmente $\alpha = 0.05$ ou $\alpha = 0.01$, então rejeita-se a hipótese nula de que não há diferenças significativas entre os tratamentos.

3.6.2 Teste de Nemenyi

O teste de Nemenyi (NEMENYI, 1963) é um teste estatístico utilizado como um método *post-hoc* após a rejeição da hipótese nula em um teste de Friedman. Ele é usado para determinar quais pares de tratamentos ou grupos diferem entre si, quando há uma diferença global significativa entre os grupos.

O desempenho de dois classificadores é significativamente diferente se as classificações médias correspondentes diferirem em pelo menos a diferença crítica, onde os valores críticos q_α são baseados na estatística de faixa Studentizada dividida por $\sqrt{2}$, como mostra a Tabela 2a.

A estatística do teste de Nemenyi é calculada usando a Equação 3.41.

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{3.41}$$

Tabela 2 – Valores críticos para testes *post-hoc* após teste de Friedman

Número de Classificadores	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$q_{0.05}$	1.960	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164
$q_{0.10}$	1.645	2.052	2.291	2.459	2.589	2.693	2.780	2.855	2.920

(a) Valores críticos para o teste de Nemenyi bicaudal.

Fonte: Tabela 5 de (DEMŠAR, 2006).

em que
CD é a diferença crítica entre pares de tratamentos;
q_α é o valor crítico associado a um nível de significância α;
k é o número de tratamentos (algoritmos).
N é o número total de comparações possíveis entre os tratamentos.

Essa fórmula é usada para calcular a distância crítica entre os *rankings* dos tratamentos, e é usada para determinar se há uma diferença estatisticamente significativa entre os pares de tratamentos em comparações múltiplas.

3.7 Conjunto de dados "*Multiple features*"

Este conjunto de dados consiste em características de numerais manuscritos ($'0' - '9'$) extraídos de uma coleção de mapas utilitários da Holanda (DUIN, 2024). 200 padrões por classe (para um total de 2.000 padrões) foram digitalizados em imagens binárias. Os dados já estão pré-processados, significando que os coeficientes de Fourier, Karhunen-Loève e momentos Zernike foram calculados a partir das imagens dos dígitos e fornecidos diretamente no conjunto de dados. Os dados podem ser acessados por meio do site *uci machine learning repository* (<https://archive.ics.uci.edu/dataset/72/multiple+features>).

Os dígitos são organizados através de seis conjuntos de arquivos dataset (em que apenas 3 foram usados neste estudo: `mfeat-fou`, `mfeat-fac` e `mfeat-zer`):

- **`mfeat-fou`**: 76 coeficientes de Fourier das formas dos caracteres;
- **`mfeat-fac`**: 216 correlações de perfil;
- `mfeat-kar`: 64 coeficientes de Karhunen-Love;
- `mfeat-pix`: médias de 240 pixels em janelas 2 x 3;
- **`mfeat-zer`**: 47 momentos Zernike;
- `mfeat-mor`: 6 características morfológicas.

A principal aplicação dessa base de dados é para problemas de classificação de padrões, onde o objetivo é identificar qual dígito (de 0 a 9) corresponde a uma determinada amostra baseada nos diferentes conjuntos de características.

3.7.1 `mfeat-fou` (Coeficientes de Fourier)

Descrição: Esse conjunto de dados contém os coeficientes de Fourier de cada um dos 2.000 dígitos manuscritos.

Tipo de Dados: Classificação;

Estrutura: Matriz de 2.000 x 76.

Linhas: Cada linha representa um dígito manuscrito.

Colunas: Existem 76 colunas representando os coeficientes de Fourier. Estes coeficientes são extraídos das representações contornadas dos dígitos e são usados para capturar características invariantes à translação e rotação.

3.7.2 `mfeat-fac` (Coeficientes de Karhunen-Loève)

Descrição: Esse conjunto de dados contém os coeficientes de Karhunen-Loève dos dígitos manuscritos.

Tipo de Dados: Classificação.

Estrutura: Matriz de 2.000 x 216.

Linhas: Cada linha representa um dígito manuscrito.

Colunas: Existem 216 colunas representando os coeficientes de Karhunen-Loève, também conhecidos como componentes principais. Estes coeficientes são derivados da Análise de Componentes Principais (PCA) aplicada às imagens dos dígitos, reduzindo a dimensionalidade dos dados.

3.7.3 `mfeat-zer` (Momentos Zernike)

Descrição: Esse conjunto de dados contém os momentos de Zernike dos dígitos manuscritos.

Tipo de Dados: Classificação.

Estrutura: Matriz de 2.000 x 47.

Linhas: Cada linha representa um dígito manuscrito.

Colunas: Existem 47 colunas que representam os momentos de Zernike. Momentos de Zernike são usados para descrever a forma dos dígitos de uma maneira que é invariável a rotação, escala e translação.

Por fim, podemos esclarecer que "*Multiple Features Dataset*" é uma base de dados rica em diferentes representações de dígitos manuscritos, oferecendo várias perspectivas e técnicas para a tarefa de reconhecimento de padrões e classificação. Os conjuntos de dados `mfeat-fou`,`mfeat-fac` e `mfeat-zer` fornecem diferentes tipos de transformações e reduções de dimensionalidade que são úteis para entender e classificar as formas dos dígitos.

4 Resultados e Conclusões

4.1 Análise 1: Implementação do Algoritmo KFCM-K-W.1 e Métodos de Avaliação de Partição

Aqui, foi implementado o algoritmo KFCM-K-W.1, que é uma variação do algoritmo Fuzzy C-Means (FCM). Ele utiliza um kernel gaussiano e largura global para realizar a clusterização fuzzy de dados (veja a Seção 3.1).

O algoritmo final foi ajustado com os parâmetros da Tabela 3. O parâmetro c indica o número de clusters; T refere-se ao número máximo de iterações; ϵ é uma tolerância para a convergência; m é o parâmetro de fuzzificação; e n_runs é o número de execuções. Os parâmetros foram escolhidos após várias verificações, por exemplo, para base de dados `mdeat-fac`, foi utilizado o valor de $m = 1.2$, pois o algoritmo não estava conseguindo fazer a partição correta em 10 cluster, devido a complexidade da base de dados. Para corrigir isso, após uma investigação na literatura, decidimos diminuir o valor de m a fim de obter os agrupamentos conforme foram solicitados.

Para cada conjunto de dados, o KFCM-K-W.1 foi executado 50 vezes a fim de minimizar a função objetivo e encontrar uma solução ótima. Durante cada execução, os centros dos clusters e os parâmetros de largura global foram atualizados iterativamente até que a convergência fosse alcançada.

Tabela 3 – Parâmetros utilizados em cada conjunto de dados

Parâmetro	**Dataset**		
	`mfeat-fou`	**`mfeat-fac`**	**`mfeat-zer`**
c	10	10	10
T	100	100	100
ϵ	10^{-6}	10^{-6}	10^{-6}
m	1.6	1.2	1.6
n_runs	50	50	50

4.1.1 Análise `mfeat-fou`

Para o conjunto de dados `mfeat-fou`, a função foi minimizada com um valor de $J_{\text{KFCM-K-W.1}} = 540.31$, como ilustra a Tabela 4. Para o melhor resultado da função objetivo foram obtidos os melhores centróides dos grupos, que representam os pontos centrais de cada cluster. Para acessar esses vetores, acesse o link a seguir: <https://raw.githubusercontent.com/rdmff/IN1102/main/Francisco/dataset/result/mfeat_fou.txt>; cada linha na lista de protótipos corresponde a um cluster, e cada valor na linha representa a coordenada média

do cluster em uma determinada dimensão dos dados. No geral, ao analisar os valores dos protótipos para esta base de dados, observa-se que cada cluster tem uma distribuição de coordenadas diferente, indicando uma boa separação dos dados em 10 grupos distintos.

Os parâmetros de largura global indicam a largura dos clusters em cada dimensão dos dados. Para esta base de dados, nota-se que eles variam consideravelmente. Cada valor corresponde a um ponto de dados e representa o raio de alcance ao redor desse ponto, dentro do qual outros pontos são considerados vizinhos. Esses valores nos dão uma visão de quais pontos serão agrupados juntos com base em sua proximidade.

```
Melhores parâmetros de largura global:
[0.60 0.29 0.37 0.39 0.32 0.68 0.34 0.40 0.38 0.53 0.62 0.61 0.59 0.46 0.88
 1.22 0.84 0.67 0.81 1.16 1.34 1.09 1.12 0.90 1.37 1.53 1.39 1.37 1.06 1.57
 1.59 1.69 1.59 1.68 1.66 1.82 1.83 1.89 1.73 1.85 1.98 2.05 2.19 2.12 1.86
 2.09 1.88 1.91 1.75 1.72 1.65 1.57 1.71 1.42 1.38 1.45 1.35 1.19 1.34 1.01
 1.04 1.04 0.91 1.03 1.09 0.72 0.74 0.73 0.61 0.52 0.61 0.59 0.47 0.41 0.53
 0.43]
```

O MPC foi calculado a fim de avaliar a qualidade dos clusters. Neste caso, de acordo com a Tabela 4, o MPC tem um valor de aproximadamente 0.214, o que sugere que os clusters gerados pelo algoritmo têm uma correspondência moderada com as classes.

O ARI é outra medida de validação que compara as associações entre os dados e os clusters gerados pelo algoritmo com as associações verdadeiras. Aqui, o ARI é aproximadamente 0.830, como mostra a Tabela 4, indicando uma boa correspondência entre os agrupamentos.

Como o MPC é baixo e o ARI é alto, isso sugere que embora os clusters possam estar bem definidos e bem separados uns dos outros, a distribuição dos pontos dentro desses clusters pode não ser uniforme ou ideal. Assim, quando o MPC é baixo, significa que os clusters podem não ser muito compactos, enquanto um ARI alto indica que os clusters encontrados pelo algoritmo são semelhantes às classes reais dos dados. Isso indica que os grupos podem estar bem separados uns dos outros, mas a distribuição interna dos pontos dentro desses clusters pode não ser muito homogênea. É possível que alguns clusters tenham regiões mais densas e outras mais esparsas, o que resulta em um baixo valor de MPC. No entanto, o ARI alto indica que, apesar disso, os clusters encontrados são semanticamente significativos e correspondem bem às classes reais dos dados.

Tabela 4 – Saídas das medidas do algoritmo KFCM-K-W.1 para o dataset `mfeat-fou`

	Dataset
	mfeat-fou
$J_{\textbf{KFCM-K-W.1}}$	540.311
MPC	0.21435
ARI	0.83023

Finalmente, a matriz de confusão entre a partição crisp (obtida a partir dos valores de pertinência) e a partição a priori (obitida pela primeira execução do algoritmo) fornece uma visão de como os clusters se comparam as classes reais. Cada elemento da matriz representa o número de pontos que foram atribuídos a um determinado cluster em comparação com os rótulos verdadeiros. Os valores da diagonal da matriz indica uma correspondência perfeita entre os clusters e os rótulos verdadeiros. Para a base `mfeat-fou`, a matriz de confusão mostra uma correspondência fraca entre os clusters gerados e as verdadeiras classes, com poucos pontos sendo atribuídos corretamente aos seus clusters correspondentes. Isso pode ocorrer devido a algumas limitações do algoritmo ou até mesmo pela complexidade dos dados. A matriz de confusão pode ser visualizada abaixo.

Matriz de confusão: partição crisp vs a priori - `mfeat-fou`

$$\begin{bmatrix} 190 & 0 & 0 & 5 & 0 & 5 & 0 & 0 & 2 & 1 \\ 0 & 6 & 1 & 0 & 245 & 1 & 8 & 1 & 0 & 1 \\ 0 & 3 & 15 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 12 & 1 & 0 & 52 & 0 & 20 & 0 & 11 \\ 0 & 1 & 0 & 0 & 0 & 0 & 243 & 0 & 0 & 0 \\ 0 & 3 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 240 & 0 & 9 & 0 & 0 & 0 & 0 \\ 212 & 0 & 0 & 1 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 440 & 1 & 0 & 0 & 12 & 1 & 0 & 0 & 5 \\ 0 & 0 & 0 & 0 & 0 & 35 & 0 & 0 & 211 & 0 \end{bmatrix}$$

4.1.2 Análise `mfeat-fac`

Para o conjunto de dados `mfeat-fac`, a função foi minimizada com um valor de $J_{\text{KFCM-K-W.1}} = 0.00055$, como apresenta a Tabela 5. Os melhores centróides dos grupos obtidos com o melhor resultado da função objetivo podem ser acessados por meio do link a seguir: <https://raw.githubusercontent.com/rdmff/IN1102/main/Francisco/dataset/result/mfeat_fac.txt>. Aqui também o cluster tem uma distribuição de coordenadas diferente, indicando uma boa separação dos dados em 10 grupos distintos.

Para esta base de dados, os parâmetros de largura global resultaram em valores consideravelmente grandes, o que sugere que algoritmo não conseguiu calcular de forma correta os valores desses parâmetros, isso pode ocorrer por exemplo, quando existem valores muito pequenos e valores grandes na base de dados, fazendo com a divisão do cálculo tenda a infinito. A alta dispersão também sugere que existem componentes ou relações no sistema que estão muito dispersos ou distantes uns dos outros. A distribuição dos valores não é uniforme; alguns são muito maiores do que a maioria, enquanto outros são relativamente menores. Isso indica uma estrutura complexa ou heterogênea no sistema.

```
Melhores parâmetros de largura global:
```

```
[25074079.00 37912661.90  59233677.60 32678164.70  5978886.77 21563195.40
   414814.45   163222.70    226705.54    50424.74    47180.70    45864.91
 18310963.00 23836732.20  25904970.60 18027245.80  5630006.42 26790280.30
   338162.69   137522.90    156952.47    75223.70    37675.10    51584.59
  8218982.96 36630762.10  13161552.80 12313892.30 37826285.50 16392307.60
   241035.99   133493.09    367022.86    68000.33    75421.79    88894.05
 23148793.40 45834996.30  37855193.70 29509171.20 15947334.90 15703604.90
   156996.10   166967.50    272232.72    58827.72    93020.53    91731.44
 11411129.50 44242531.90  36565256.80 26892642.50 25890666.20 15632566.10
   382125.69   213345.24    367022.86    58080.14    91602.76    17368.91
 10718622.40 17830425.60  37756223.90 24519206.90 43621493.40 18298926.10
   364494.62   121560.09    315759.63    64976.01    35242.64   111194.27
  9680452.89 18715750.98  24903863.60 25857028.00  9807640.89 34143980.37
   217624.65   157201.77    256160.67    90275.90    35325.40    87478.03
  9844879.91 34086369.30  24379391.50 32993132.00 13307668.19 24221099.87
   271870.74    75406.37    334092.50    76897.62    21345.66    58140.92
 37235589.90 20666074.50  16440019.00 42071165.20  6536539.06 22815366.20
   218603.74   247756.17    358186.98    67177.42    61208.68    64220.35
 36317581.60 16402407.30  71848169.50 10736619.10 28893669.00 18490982.40
   318885.95    55342.54    303789.60    41611.85    89287.97    91979.17
  7440569.54 12808651.84  59306925.97 22764756.50 30002653.63 15543310.27
   291585.58    67850.78    274818.42    63470.54    19365.41    77257.03
 20899325.41 18188885.34  71850278.84 29857195.99 12900091.40 12758236.28
   162119.77   164298.25    211472.94    61373.61    35654.56    89387.60
  9895684.80 34582946.71  47760440.00 42058847.10 13693555.64 16433426.14
   185391.08   153321.68    204220.21    67085.03    40180.41    46274.81
 22323405.70 20169838.50  17285299.30 25520120.10 11331197.10 30031411.13
   176855.22   103442.92    443191.15    85999.95    75421.79    20795.58
 13161509.10 18819029.10  30577614.20 12183692.80 33118603.10 17677441.20
   202874.76   115906.24    416209.87    23024.69    75126.93   104756.91
 49948164.00 18077859.80  58512473.50 25780702.40 44568335.70 25722947.00
   291585.58   231216.91    281110.43    49506.55    80789.14   107169.17
 29861919.70 42498642.29  36760879.54 35033005.14  5868528.08 39350731.67
   425234.60   242186.05    338290.10    77724.73    52069.24    39905.35
 11854344.52 11931000.52  62941867.47 38323174.84  7701692.88 16977429.67
    70932.23   193687.65     93427.85    17651.34     9873.76    69046.75]
```

Para avaliar a qualidade dos clusters, foi calculado o MPC. Para esta base de dados, de acordo com a Tabela 5, o MPC tem um valor de aproximadamente 0.67, o que indica que os clusters gerados pelo algoritmo têm uma correspondência boa com as classes. O ARI foi de aproximadamente 0.96, que indica uma alta concordância entre os agrupamentos, sugerindo que os grupos são bem definidos e similares.

Portanto, essas medidas indicam que o modelo está fazendo previsões com uma confiança moderada e que os agrupamentos feitos pelo modelo são altamente semelhantes aos agrupamentos reais.

Tabela 5 – Saídas das medidas do algoritmo KFCM-K-W.1 para o dataset `mfeat-fou`

	Dataset
	mfeat-fou
$J_{\textbf{KFCM-K-W.1}}$	0.00055
MPC	0.67510
ARI	0.95944

Por fim, a matriz de confusão entre a partição crisp vs partição a priori apresenta que os valores da diagonal da matriz indica uma fraca correspondência entre os clusters e os rótulos verdadeiros, com poucos pontos sendo atribuídos corretamente aos seus clusters correspondentes. Isso pode ocorrer devido a algumas limitações do algoritmo ou até mesmo pela complexidade dos dados. A matriz de confusão pode ser visualizada abaixo.

Matriz de confusão: partição crisp vs a priori - `mfeat-fac`

$$\begin{bmatrix} 0 & 0 & 0 & 13 & 0 & 0 & 0 & 0 & 0 & 225 \\ 0 & 0 & 176 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 145 & 0 & 5 & 1 & 1 & 3 & 0 \\ 238 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 176 & 0 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 203 & 0 & 0 & 0 \\ 4 & 213 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 168 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 209 & 0 \\ 0 & 0 & 0 & 2 & 0 & 208 & 0 & 0 & 0 & 1 \end{bmatrix}$$

4.1.3 Análise `mfeat-zer`

Para o conjunto de dados `mfeat-zer`, a função foi minimizada com um valor de $J_{\text{KFCM-K-W.1}} = 0.00055$, como apresenta a Tabela 6. Os melhores centróides dos grupos obtidos com o melhor resultado da função objetivo podem ser acessados por meio deste link: <https://raw.githubusercontent.com/rdmff/IN1102/main/Francisco/dataset/result/mfeat_zer.txt>. Neste caso, o cluster também tem uma distribuição de coordenadas diferente, indicando uma boa separação dos dados em 10 grupos distintos, fazendo uma boa representação dos grupos.

Para esta base de dados, os parâmetros de largura global resultaram em valores grandes, o que indica que algoritmo não conseguiu calcular de forma correta os valores desses parâmetros. Pode ter ocorrido o que aconteceu para base de dados `mfeat-fac`: a

alta dispersão e distribuição dos valores não uniforme. Isso sugere uma estrutura complexa ou heterogênea no sistema.

```
Matriz de confusão entre a partição crisp e a primeira execução:
    [16.37    4275.85  415069.81  3680443.75  5912154.96 36166074.80 20988141.80
     26.85    7610.85  794710.96  6270005.04  8748805.38  9786820.53       68.03
  40658.87 1464027.00 7430240.86 11451001.20 29472306.00      141.35    93190.15
2666046.52 8000935.44 9996557.48      896.01   218439.45  3909952.56 12753995.50
27502375.4    2439.73      44.00     3124.48  5835590.02  9082248.61     7118.00
 767415.26 8178515.75 17092211.1    16544.50  1694763.83  8927126.07    34506.52
3246930.13 21306470.7  103645.80  5869652.07   289616.02 11723018.00]
```

Em sequência, foi calculado o MPC a fim de avaliar a qualidade dos clusters. Para o conjunto de dados `mfeat-zer`, de acordo com a Tabela 6, o MPC tem um valor de aproximadamente 0.5, o que indica que os clusters gerados pelo algoritmo têm uma correspondência média com as classes; e o ARI foi de aproximadamente 0.63, que sugere uma concordância moderada entre os agrupamentos observados e esperados.

Portanto, esses valores indicam uma precisão moderada do algoritmo de agrupamento, onde aproximadamente metade das instâncias estão sendo classificadas corretamente, e há uma concordância moderada entre os agrupamentos observados e esperados.

Tabela 6 – Saídas das medidas do algoritmo KFCM-K-W.1 para o dataset `mfeat-zer`

	Dataset
	mfeat-fou
$J_{\textbf{KFCM-K-W.1}}$	0.00930
MPC	0.50493
ARI	0.63219

Por fim, a matriz de confusão entre a partição crisp vs partição a priori apresenta que os valores da diagonal da matriz indica uma fraca correspondência entre os clusters e os rótulos verdadeiros, com poucos pontos sendo atribuídos corretamente aos seus clusters correspondentes. Isso pode ocorrer devido a algumas limitações do algoritmo ou até mesmo pela complexidade dos dados. A matriz de confusão pode ser visualizada abaixo.

Matriz de confusão: partição crisp vs a priori - `mfeat-zer`

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 50 & 0 & 0 & 0 & 0 & 0 \\ 0 & 176 & 11 & 0 & 0 & 0 & 0 & 0 & 19 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 46 & 0 & 5 & 0 \\ 0 & 0 & 0 & 43 & 0 & 0 & 0 & 0 & 0 & 399 \\ 0 & 0 & 0 & 111 & 0 & 0 & 0 & 186 & 0 & 7 \\ 1 & 0 & 119 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 43 & 0 & 0 & 0 & 181 & 0 & 22 & 0 & 0 \\ 0 & 0 & 0 & 222 & 0 & 76 & 0 & 4 & 0 & 5 \\ 93 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 57 & 116 & 0 \end{bmatrix}$$

4.2 Análise 2: Desempenho de Classificadores

Aqui foi realizada uma validação cruzada estratificada com 30 repetições e 10 *folds* para comparar quatro classificadores (bayesiano Gaussiano, bayesiano baseado em K-NN, bayesiano baseado na Janela de Parzen e Regressão Logística) usando a regra do voto majoritário. Primeiramente, foi configurado os parâmetros necessários por meio da descrição de cada algoritmo, como solicitado; e os rótulos foram obtidos por meio do algoritmo k-means. Após isso, as bases de dados foram divididas entre os conjuntos de treinamento e teste. Em seguida, foi feito um treinamento (*ensemble*) para cada algoritmo baseado em 3 algoritmos de cada um dos classificadores, isto é, um treinamento em 3 algoritmos bayesiano Gaussiano, um treinamento em 3 algoritmos bayesiano baseados em K-NN, e assim por diante, para 3 bases de dados com dimensões distintas. Com isso, foi feita a validação cruzada estratificada com ajuste de hiperparâmetros, quando necessário, e foram obtidas as métricas de avaliação como acurácia, precisão, recall e F1-score para cada classificador. Ademais, o código calcula uma estimativa pontual e um intervalo de 95% de confiança para cada métrica de avaliação de cada classificador, conforme ilustra a Tabela 7.

Os passos específicos para cada classificador envolvem treinar o classificador em cada um dos três conjuntos de dados e, em seguida, usar a regra do voto majoritário para combinar as previsões dos classificadores treinados e atribuir uma classe a cada exemplo.

Nesta Seção, foram obtidas as métricas de cada algoritmo analisado e, em seguida, foram aplicados os métodos descritos na Seção 3.6 para comparação do desempenho dos classificadores. DEMŠAR (2006) apresenta os testes ANOVA e Friedman para comparar o desempenho de múltiplos classificadores. Todavia, o teste ANOVA baseia-se em suposições que provavelmente são violadas ao analisar o desempenho de algoritmos de aprendizado de máquina. Por esse motivo, o teste de Friedman foi o escolhido para aplicação neste estudo. Por conseguinte, foi aplicado o teste *post-hoc* de Nemenyi para realizar comparações detalhadas entre os classificadores após a hipótese nula (H_0) ser rejeitada.

As métricas de acurácia, precisão, *recall* e *F1-score* obtidas para os quatro classifi-

Tabela 7 – Métricas e intervalos de confiança do desempenho de cada algoritmo

Algoritmos	Métricas			
	ACURÁCIA	PRECISÃO	RECALL	F1-SCORE
GAUSSIANO	0.8824 [0.8809; 0.8838]	0.8849 [0.8835; 0.8863]	0.8824 [0.8809; 0.8838]	0.8823 [0.8808; 0.8837]
K-NN	0.9243 [0.9231; 0.9255]	0.9253 [0.9241; 0.9265]	0.9243 [0.9231; 0.9255]	0.9241 [0.9229; 0.9253]
PARZEN	0.4524 [0.4509; 0.4538]	0.8566 [0.8549; 0.8584]	0.4524 [0.4509; 0.4538]	0.5259 [0.5246; 0.5274]
REGRESSÃO LOGÍSTICA	0.9311 [0.9299; 0,9323]	0.9321 [0.9309; 0.9333]	0.9311 [0.9299; 0,9323]	0.9310 [0.9298; 0.9322]

cadores estão apresentadas na Tabela 8. Já na Tabela 9 é possível identificar os postos de cada métrica em relação aos modelos.

Tabela 8 – Resultados das métricas para diferentes algoritmos

Métricas	Classificadores			
	GAUSSIANO	K-NN	PARZEN	REGRESSÃO LOGÍSTICA
Acurácia	0.8824	0.9243	0.4524	0.9311
Precisão	0.8849	0.9253	0.8566	0.9321
Recall	0.8824	0.9243	0.4524	0.9311
F1-score	0.8822	0.9240	0.5259	0.9310
Média	**0.8829**	**0.9244**	**0.5718**	**0.9313**

Tabela 9 – Ranks dos classificadores para diferentes algoritmos

Métricas	Classificadores			
	GAUSSIANO	K-NN	PARZEN	REGRESSÃO LOGÍSTICA
Acurácia	3	2	4	1
Precisão	3	2	4	1
Recall	3	2	4	1
F1-score	3	2	4	1
Rank Médio	**3**	**2**	**4**	**1**

4.2.1 Aplicação dos Testes Não-paramétricos

Aplicando os valores dos ranks médios da Tabela 9 na Equação 3.40, e considerando os valores de $N = 4$, $k = 4$, para $\alpha = 0.05$ e $\alpha = 0.01$, obtém-se uma estatística do teste $\chi^2_{\text{calculado}} = 12$.

De acordo com a Tabela qui-quadrado (consulte a Tabela A4 em (SHESKIN, 2003)), os valores críticos com $(k - 1 = 3)$ graus de liberdade, para os níveis de significância $\alpha = 0.05$ e $\alpha = 0.01$, respectivamente, são $\chi^2_{(0.05,3)} = 7.81$ e $\chi^2_{(0.01,3)} = 11.34$. Como o valor de $\chi^2_{calculado}$ é maior do que os valores tabelados de $\chi^2_{(0.05,3)}$ e $\chi^2_{(0.01,3)}$, rejeitamos a hipótese nula H_0, a um nível de significância de $\alpha = 0.05$ e $\alpha = 0.01$, respectivamente. Isso sugere que existe uma diferença estatisticamente significativa entre, pelo menos, dois dos quatro classificadores.

A fim de identificar os algoritmos com diferença significativa, foi aplicado o teste *post-hoc* de Nemenyi em cada par de classificadores. Nesse teste, é preciso obter a diferença

entre os ranks médias (linha da média dos ranks) entre todos os classificadores (comparando por pares). Se essa diferença for maior ou igual a uma distância crítica CD, pode-se dizer que estes dois classificadores são significativamente diferentes entre si. CD é mostrado na Equação 3.41. Como nosso $k = 4$, assumindo $\alpha = 0.05$ e, de acordo com a Tabela 2a, o valor de $q_{0.05} = 2.569$, a distância crítica correspondente é $CD = 2.569 \times \sqrt{\frac{4(4+1)}{6\times4}} = 2.569 \times \sqrt{\frac{20}{24}} \approx 2.345$.

Note que apenas a diferença entre o algoritmo de melhor e o pior desempenho do rank médio é maior do que a distância crítica ($3 > 2.345$). Logo, podemos concluir que a performance do classificador bayesiano baseado na Janela de Parzen é significativamente pior do que a performance do algortimo de Regressão Logística. Mais ainda, se considerarmos, agora, $\alpha = 0.10$, com o valor de $q_{0.10} = 2.291$ da Tabela 2a, o resultado da região crítica $CD = 2.291 \times \sqrt{\frac{20}{24}} \approx 2.091$ diminui, entretanto ainda assim apenas a diferença entre os algoritmos bayesiano baseado na Janela de Parzen e Regressão Logística foram signficativamente diferentes, a um nível $\alpha = 0.10$.

Referências

ARPIT512512, G. de. *K-Means-Clustering*. 2024. Disponível em: <https://github.com/arpit512512/K-Means-Clustering/tree/master/stKFCM>. Citado na página 4.

CARVALHO, F. d. A. D.; LECHEVALLIER, Y. Partitional clustering algorithms for symbolic interval data based on single adaptive distances. *Pattern Recognition*, Elsevier, v. 42, n. 7, p. 1223–1236, 2009. Citado na página 1.

CARVALHO, F. d. A. D. et al. Adaptive hausdorff distances and dynamic clustering of symbolic interval data. *Pattern Recognition Letters*, Elsevier, v. 27, n. 3, p. 167–179, 2006. Citado na página 1.

CARVALHO, F. d. A. de; NETO, E. d. A. L.; SILVA, K. C. da. A clusterwise nonlinear regression algorithm for interval-valued data. *Information Sciences*, Elsevier, v. 555, p. 357–385, 2021. Citado na página 1.

CARVALHO, F. d. A. de; SANTANA, L. V.; FERREIRA, M. R. Gaussian kernel-based fuzzy clustering with automatic bandwidth computation. In: SPRINGER. *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*. [S.l.], 2018. p. 685–694. Citado na página 4.

CHAVENT, M. A monothetic clustering method. *Pattern Recognition Letters*, Elsevier, v. 19, n. 11, p. 989–996, 1998. Citado na página 1.

CONGALTON, R. G.; GREEN, K. *Assessing the accuracy of remotely sensed data: principles and practices*. [S.l.]: CRC press, 2019. Citado na página 14.

DAVE, R. N. Validating fuzzy partitions obtained through c-shells clustering. *Pattern recognition letters*, Elsevier, v. 17, n. 6, p. 613–623, 1996. Citado 2 vezes nas páginas 2 e 7.

DAVISEK20, G. de. *Kernelized General Fuzzy c-Means Clustering*. 2024. Disponível em: <https://github.com/Avisek20/kernelized_general_fuzzy_c_means>. Citado na página 4.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, JMLR. org, v. 7, p. 1–30, 2006. Citado 6 vezes nas páginas 2, 3, 4, 16, 18 e 27.

DITSKIH, G. de. *Fuzzy-c-Means-Algorithms*. 2024. Disponível em: <https://github.com/xiefan-guo/Fuzzy-c-Means-Algorithms>. Citado na página 4.

DITSKIH, G. de. *Projects of Praditya Nugraha*. 2024. Disponível em: <https://github.com/Ditskih/Project>. Citado na página 4.

DUIN, R. *Multiple Features*. 2024. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5HC70. Citado na página 18.

EZUGWU, A. E. et al. A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 110, p. 104743, 2022. Citado na página 1.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, Taylor & Francis, v. 32, n. 200, p. 675–701, 1937. Citado 2 vezes nas páginas 2 e 16.

FRIEDMAN, M. A comparison of alternative tests of significance for the problem of m rankings. *The annals of mathematical statistics*, JSTOR, v. 11, n. 1, p. 86–92, 1940. Citado na página 16.

GARCIA, S.; HERRERA, F. An extension on"statistical comparisons of classifiers over multiple data sets"for all pairwise comparisons. *Journal of machine learning research*, v. 9, n. 12, 2008. Citado na página 16.

HUANG, J. Z. et al. Automated variable weighting in k-means type clustering. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 27, n. 5, p. 657–668, 2005. Citado na página 1.

HUBERT, L.; ARABIE, P. Comparing partitions. *Journal of classification*, Springer, v. 2, p. 193–218, 1985. Citado 2 vezes nas páginas 2 e 8.

KITTLER, J. et al. On combining classifiers. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 20, n. 3, p. 226–239, 1998. Citado 2 vezes nas páginas 2 e 4.

LOPEZ, C. et al. An unsupervised machine learning method for discovering patient clusters based on genetic signatures. *Journal of biomedical informatics*, Elsevier, v. 85, p. 30–39, 2018. Citado na página 1.

LUQUE, A. et al. The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, Elsevier, v. 91, p. 216–231, 2019. Citado na página 14.

MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. *Introduction to information retrieval*. [S.l.]: Cambridge university press, 2008. Citado na página 2.

MONICO, J. F. G. et al. Acurácia e precisão: revendo os conceitos de forma acurada. *Boletim de Ciências Geodésicas*, Universidade Federal do Paraná, v. 15, n. 3, p. 469–483, 2009. Citado na página 15.

NEMENYI, P. B. *Distribution-free multiple comparisons*. [S.l.]: Princeton University, 1963. Citado 2 vezes nas páginas 2 e 17.

SHESKIN, D. J. *Handbook of parametric and nonparametric statistical procedures*. [S.l.]: Chapman and hall/CRC, 2003. Citado 3 vezes nas páginas 4, 17 e 28.

SIMÕES, E. C.; CARVALHO, F. d. A. de. Gaussian kernel fuzzy c-means with width parameter computation and regularization. *Pattern Recognition*, Elsevier, v. 143, p. 109749, 2023. Citado 2 vezes nas páginas 1 e 4.

SYAMSUL336, G. de. *Gaussian-Kernel-Fuzzy-C-Means-with-Example-Data*. 2024. Disponível em: <https://github.com/Syamsul336/Gaussian-Kernel-Fuzzy-C-Means-with-Example-Data>. Citado na página 4.

VANKAYALAPATI, R. et al. K-means algorithm for clustering of learners performance levels using machine learning techniques. *Revue d'Intelligence Artificielle*, v. 35, n. 1, 2021. Citado na página 1.

ZHANG, D.-Q.; CHEN, S.-C. A novel kernelized fuzzy c-means algorithm with application in medical image segmentation. *Artificial intelligence in medicine*, Elsevier, v. 32, n. 1, p. 37–50, 2004. Citado na página 2.

ZHANG, R.; HAJJAR, J.; SUN, H. Machine learning approach for sequence clustering with applications to ground-motion selection. *Journal of Engineering Mechanics*, American Society of Civil Engineers, v. 146, n. 6, p. 04020040, 2020. Citado na página 1.