



Universidade Federal de Pernambuco

Programa de Pós-Graduação em Ciência da Computação

Projeto - Aprendizagem de Máquina

Professor: Francisco de Assis Tenório de Carvalho

Grupo:

Diógenes Wallis de França Silva

Johnny Marcos Silva Soares

Rodrigo Vitor Castro Alves de Mello

1. Introdução
2. Apresentação dos dados
 - 2.1. Significado dos dados
 - 2.2. Pré-processamento
3. Questão 1
 - 3.1. FCM-DFCV
 - 3.2. Problemas de Implementação
 - 3.3. Busca por melhores parâmetros
 - 3.4. Métricas
 - 3.5. Análise do melhor resultado
4. Questão 2
 - 4.1. Classificador Bayesiano Gaussiano:
 - 4.1.1. Fundamentação teórica
 - 4.1.2. Desempenho
 - 4.2. k-Vizinhos:
 - 4.2.1. Fundamentação teórica
 - 4.2.2. Obtenção de hiperparâmetros
 - 4.2.3. Desempenho
 - 4.3. Janela de Parzen:
 - 4.3.1. Fundamentação teórica
 - 4.3.2. Obtenção de hiperparâmetros
 - 4.3.3. Desempenho
 - 4.4. Regressão Logística:
 - 4.4.1. Fundamentação teórica
 - 4.4.2. Desempenho
 - 4.5. Voto Majoritário:
 - 4.5.1. Fundamentação teórica
 - 4.5.2. Desempenho
 - 4.6. Para a comparação dos modelos (testes, etc.)
 - 4.6.1. Teste de Friedman
 - 4.6.2. Teste de Nemenyi
5. Referências

1. Introdução

Este documento trata do projeto da disciplina Aprendizagem de Máquina ministrada no programa de pós-graduação em Ciência da Computação na Universidade Federal de Pernambuco durante o primeiro semestre letivo de 2021.

O projeto abarcou duas atividades (Questão 1 e Questão 2) a serem realizadas sobre o mesmo dataset. A Questão 1 teve como tópico aprendizagem não-supervisionada; especificamente, um algoritmo de clusterização. A Questão 2 teve como tópico aprendizagem supervisionada, abarcando diversos algoritmos.

Após esta introdução, abordamos o conjunto específico de dados (seção 2), explicando o conjunto em si (2.1) e o pré-processamento que realizamos sobre ele (2.2). Então, mergulharemos nas questões (seções 3 e 4). Para cada algoritmo abarcado pelas questões, apresentamos sua fundamentação teórica, explicamos como obtivemos seus hiperparâmetros e relatamos o seu desempenho. Por fim, realizamos os testes de hipótese pertinentes

2. Apresentação dos dados

Os dados foram obtidos do repositório UCI de aprendizagem de máquina, conforme especificação do projeto. Consideramos o “Yeast Data Set”, disponível em [<https://archive.ics.uci.edu/ml/datasets/Yeast>]

Trata-se de um dataset multivariado com oito atributos reais, aproximadamente 1500 instâncias e sem valores faltantes (vide Tabela 1)

Data Set Characteristics:	Multivariate	Number of Instances:	1484	Area:	Life
Attribute Characteristics:	Real	Number of Attributes:	8	Date Donated	1996-09-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	333591

Tabela 1: Características do conjunto de dados

2.1 Significado dos dados

Os dados de entrada, ou seja, as variáveis dependentes, descrevem diversas características de proteínas presentes no trigo (principalmente pontuações de diversos testes). A variável dependente, ou seja, a classe,

descreve a localização da proteína na composição da célula (incluindo uma classe para “extracelular”).

Índice	Atributo	Descrição
0	mcg	Método de McGeoch para reconhecimento de sequência de sinal.
1	gvh	Método de von Heijne para reconhecimento de sequência de sinal.
2	alm	Pontuação do programa de previsão da região de abrangência da membrana ALOM.
3	mit	Pontuação da análise discriminante do conteúdo de aminoácidos da região N-terminal (20 resíduos de comprimento) de proteínas mitocondriais e não mitocondriais.
4	erl	Presença de substring "HDEL" (suspeita-se que atua como um sinal para retenção no lúmen do retículo endoplasmático). Atributo binário.
5	pox	Sinal de direcionamento peroxissômico no terminal C.
6	vac	Pontuação da análise discriminante do conteúdo de aminoácidos de proteínas vacuolares e extracelulares.
7	nuc	Pontuação de análise discriminante de sinais de localização nuclear de proteínas nucleares e não nucleares.

Tabela 2: Atributos

Índice	Classe	Descrição
0	CYT	citossólica ou citoesquelética

1	NUC	nuclear
2	MIT	mitocondrial
3	ME3	proteína de membrana, sem sinal N-terminal
4	ME2	proteína de membrana, sinal não clivado
5	ME1	proteína de membrana, sinal clivado
6	EXC	extracelular
7	VAC	vacuolar
8	POX	peroxissomal
9	ERL	Lúmen do retículo endoplasmático

Tabela 3: Classes

2.2 Pré-processamento

O conjunto de dados já estava normalizado quando o obtivemos.

Analisando o histograma de cada feature, suspeitamos que duas delas (“erl” e “pox”) tinham sempre o mesmo valor. Contudo, uma análise mais minuciosa revelou que existem alguns exemplos com valores diferentes, o que é difícil de visualizar no histograma.

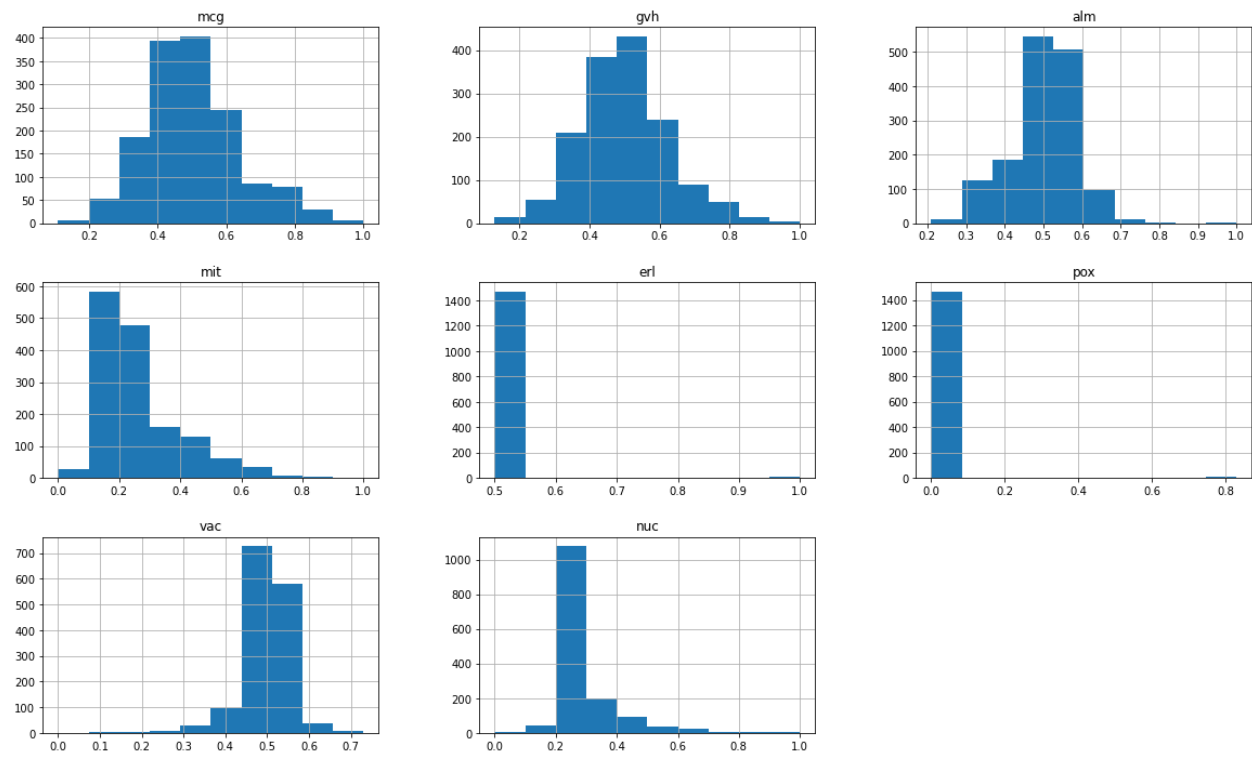


Figura 1: Histogramas dos atributos

3. Questão 1

A questão 1 consistiu das tarefas descritas na Figura 2

- Considere os dados "Yeast Data Set" do site uci machine learning repository (<http://archive.ics.uci.edu/ml/datasets/Yeast>).
 - Execute o algoritmo "FCM-DFCV" 100 vezes para obter uma partição fuzzy em 10 grupos e selecione o melhor resultado segundo a função objetivo.
 - Para detalhes do algoritmo "FCM-DFCV" veja o artigo: "Franciso de A.T. de Carvalho, Camilo P. Tenório, Nicomedes L. Cavalcanti Junior, Partitional fuzzy clustering methods based on adaptive quadratic distances, Fuzzy Sets and Systems, 157 (2006), 2833-2857". Implemente a seguinte variante desse algoritmo:
 - Função objetivo: equação (17),
 - Cálculo do prototipo: equação (03)
 - Cálculo dos pesos de relevância das variaveis: equação (19)
 - Cálculo do grau de pertinência de um objeto em um grupo: equação (20)

Figura 2: Enunciado da questão 1

Inicialmente foi visualizado a quantidade de amostras para cada classe, no qual é possível observar na Figura 3 . Existe um grande desbalanceamento nos dados, porém, isso não foi tratado neste projeto.

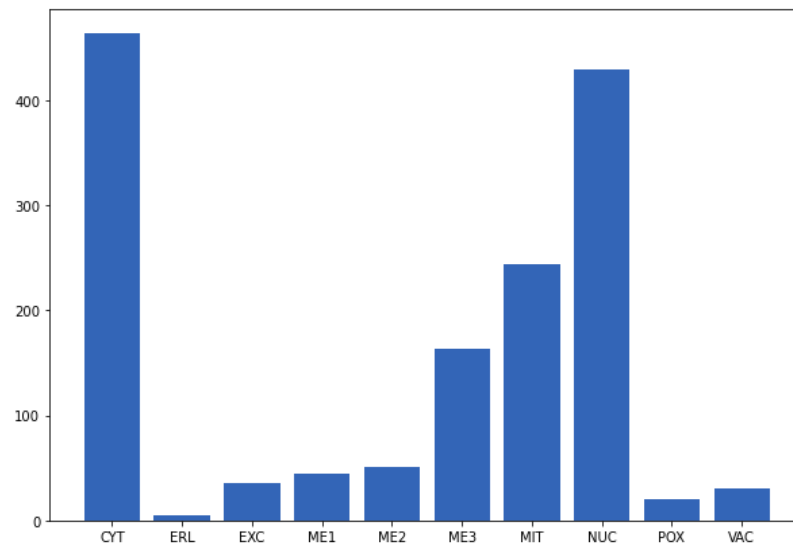


Figura 3: Quantidade de amostras para cada classe disponível

3.1 FCM-DFCV

A primeira etapa consiste na inicialização aleatória da matriz de pertinência u . Respeitando a regra $\sum_{i=1}^c u_{ik} = 1$. Ou seja, a soma dos

graus de pertinência de uma amostra para todas as classes tem que ser igual a 1. Para realizar a inicialização aleatória das probabilidades foi utilizada uma função softmax no eixo vertical da matriz de pertinência. O resultado é uma matriz com valores de probabilidade para cada amostra em cada cluster. Os valores criados aleatoriamente antes de passarem pelo softmax são multiplicados por uma constante w , que serve como fator que de acordo com w faz as probabilidades serem distribuídas de forma diferente, desde a distribuição igual para cada cluster até a criação de uma partição quase CRISP. Portanto, o valor w é considerado um parâmetro.

Função Objetivo:

$$J5 = \sum_{k=1}^n \sum_{i=1}^c (u_{ik})^m d_{\mathbf{M}_i}^2(\mathbf{x}_k, \mathbf{g}_i) = \sum_{k=1}^n \sum_{i=1}^c (u_{ik})^m (\mathbf{x}_k - \mathbf{g}_i)^T \mathbf{M}_i (\mathbf{x}_k - \mathbf{g}_i).$$

Cálculo dos protótipos:

$$\mathbf{g}_i = \frac{\sum_{k=1}^n (u_{ik})^m \mathbf{x}_k}{\sum_{k=1}^n (u_{ik})^m}.$$

Cálculo dos pesos de relevância:

$$\mathbf{M}_i = \begin{pmatrix} \lambda_i^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_i^p \end{pmatrix} = [\det(\mathbf{C}_i)]^{1/p} \mathbf{C}_i^{-1}, \quad \mathbf{C}_i = \text{diag} \left(\sum_{k=1}^n (u_{ik})^m (\mathbf{x}_k - \mathbf{g}_i)(\mathbf{x}_k - \mathbf{g}_i)^T \right)$$

$$\lambda_i^j = \frac{\{\prod_{h=1}^p [\sum_{k=1}^n (u_{ik})^m (x_k^h - g_i^h)^2]\}^{1/p}}{\sum_{k=1}^n (u_{ik})^m (x_k^j - g_i^j)^2} \quad (j = 1, \dots, p).$$

Cálculo do grau de pertinência:

$$u_{ik} = \left[\sum_{h=1}^c \left(\frac{d_{\mathbf{M}_i}^2(\mathbf{x}_k, \mathbf{g}_i)}{d_{\mathbf{M}_h}^2(\mathbf{x}_k, \mathbf{g}_h)} \right)^{1/(m-1)} \right]^{-1}$$

$$d_{\mathbf{M}_i}^2(\mathbf{x}_k, \mathbf{g}_i) = (\mathbf{x}_k - \mathbf{g}_i)^T \mathbf{M}_i (\mathbf{x}_k - \mathbf{g}_i)$$

3.2 Problemas de Implementação

Cálculo dos protótipos:

No cálculo dos protótipos podem existir problemas de inconsistência, como por exemplo, quando nenhuma amostra pertence a um certo cluster, tornando o denominador igual a zero. Portanto, para tratar esse problema, caso não existam amostras no cluster, será mantido o valor anterior do protótipo.

Cálculo dos pesos de relevância:

Quando não existem amostras para um certo cluster ou a distância do protótipo para a amostra é zero, pode existir inconsistência, pois aparecerá uma divisão por zero. Além disso, pode existir overflow no cálculo dos lambdas. Por isso, realizamos o cálculo normalmente e utilizamos o numpy para tratar as inconsistências e estouros com a verificação da ocorrência de NaN (Not a Number). Quando ocorre NaN utilizamos a matriz M anterior na matriz M atual.

Cálculo do grau de pertinência:

Quando ocorre alguma distância muito próxima à 0, acontecerá uma inconsistência no cálculo. Para contornar esse problema, consideramos que quando ocorre uma distância próxima a zero, utilizamos o valor 1 na pertinência da amostra na classe com distância próxima a 0. E utilizamos o valor 0 nas outras classes dessa amostra.

3.3 Busca por melhores parâmetros

● Observações:

● Parametros: $c = 10$; $m = \{1.1, 1.6, 2.0\}$; $T = 150$; $\epsilon = 10^{-10}$;

Figura 4: Parâmetros utilizados

Para realizar a busca dos parâmetros que obtêm os melhores resultados, utilizamos a ferramenta Optuna. O Optuna realiza a busca de parâmetros de várias formas, como otimização para encontrar os parâmetros em uma faixa de valores definidos, categorias ou uma lista de valores. Como também a busca em GRID, que consiste na combinação de todos os parâmetros selecionados. Além de realizar outros tipo de otimização, como uma otimização multi objetivo. No nosso problema utilizamos a busca em GRID para testar todas as combinações dos parâmetros que serão avaliados.

Além dos parâmetros mostrados na Figura 4, foram utilizados os valores de 1 a 100 no parâmetro w , que realiza a inicialização aleatória da matriz de pertinência. Ou seja, foram executadas 100 iterações para cada modificação dos parâmetros da Figura 4.

3.4 Métricas

- Para cada partição fuzzy, calcule o Modified partition coefficient e o Partition entropy. Comente.
- Para cada partição fuzzy, produza uma partição crisp em 10 grupos e calcule o índice de Rand corrigido, e a F-measure. Comente.

Figura 5: Métrica para cada partição

Modified Partition Coefficient (MPC):

Essa métrica avalia o quão próximo de uma partição CRISP é a partição fuzzy encontrada. Quando mais próximo de 1 mais próximo de uma partição CRISP. Podemos calcular com as expressões abaixo.

$$PC = \sum_{i=1}^N \sum_{k=1}^K u_{ik}^2 / N$$

$$\text{MPC} = 1 - \frac{K}{K-1}(1 - \text{PC})$$

Na Figura 6 é mostrado o histograma da distribuição dos valores de Modified Partition Coefficient encontrados nas 300 iterações realizadas. O resultado mostra que muitos dos resultados obtiveram um partição tendendo à uma partição CRISP.

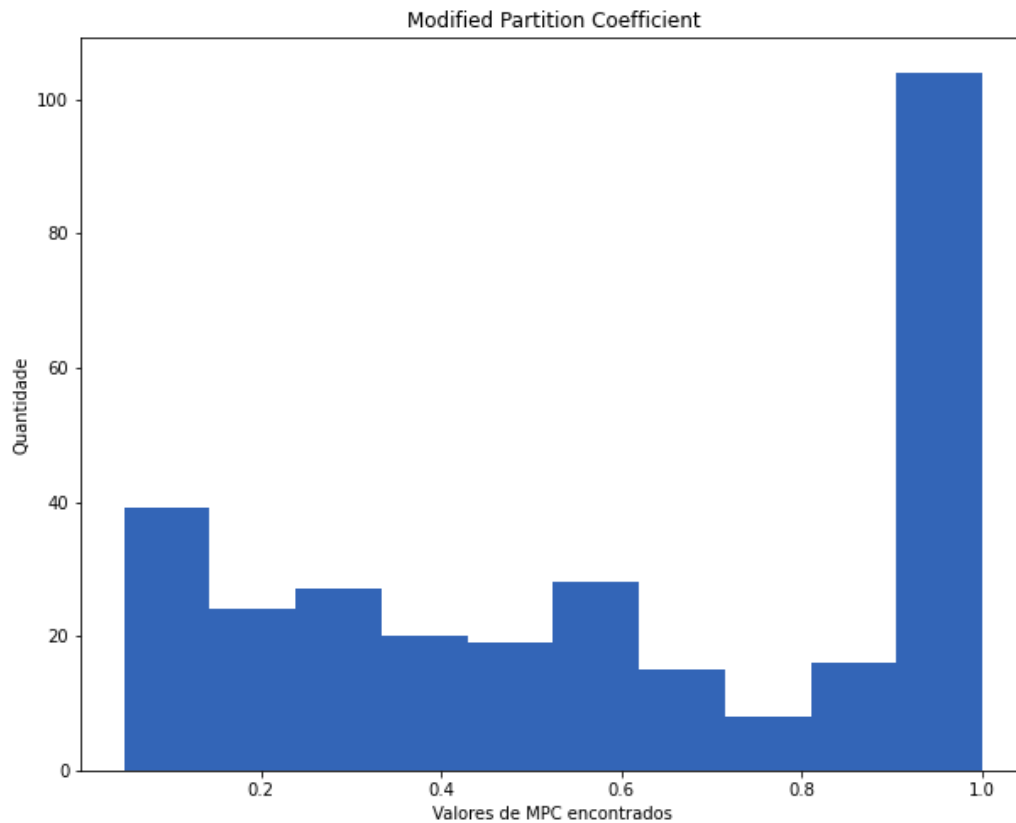


Figura 6: Histograma de MPC de todas as execuções e inicializações aleatórias

Partition Entropy (PE):

A entropia de uma partição considera a certeza ao qual os dados são separados, ou seja, o quão próximo de uma partição CRISP tende a partição Fuzzy, semelhante ao Modified Partition Coefficient. Quanto mais próximo de 0 é o PE, mais próximo de uma partição CRISP é a partição avaliada. E quanto mais próximo de $\log(c)$, onde c é o número de clusters

utilizados, maior a incerteza da clusterização. No exemplo, c é 10, então o valor máximo é 3,32.

Partition Entropy pode ser calculado utilizando a formula abaixo.

$$PE = -\frac{1}{n} \sum_{i=1}^c \sum_{j=1}^n (u_{ij}) \log_a(u_{ij})$$

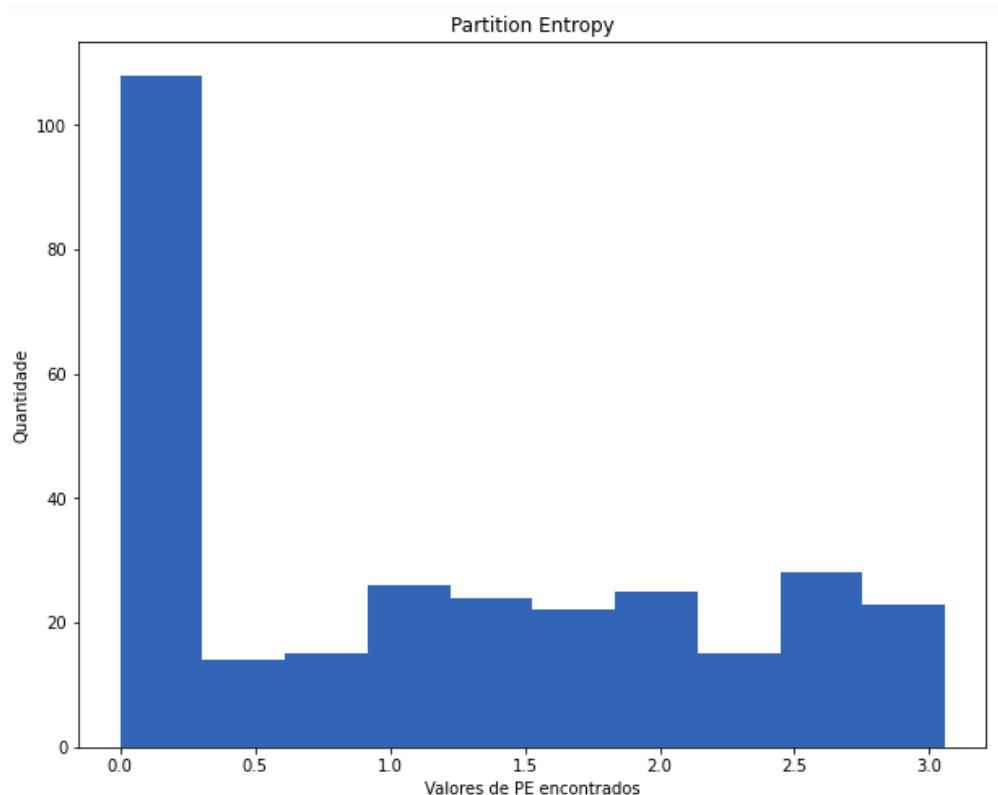


Figura 7: Histograma de PE de todas as execuções e inicializações aleatórias

Adjusted Rand Index (ARI):

O ARI é uma medida de similaridade entre dois agrupamentos. A pontuação de similaridade é entre -1 e 1, sendo que uma clusterização aleatória tem um ARI próximo ao valor 0. A métrica pode ser calculada utilizando a fórmula descrita abaixo. Porém, utilizamos a implementação disponível do Scikit Learning .

$$ARI = \frac{m - \frac{m_1 m_2}{M}}{\frac{1}{2} m_1 + \frac{1}{2} m_2 - \frac{m_1 m_2}{M}};$$

$$m = \sum_{i,j} C_{nij}^2; m_1 = \sum_i C_{ni\bullet}^2; m_2 = \sum_j C_{n\bullet j}^2; M = C_{n\bullet\bullet}^2;$$

No experimento realizado foram constatados os valores de ARI mostrados no histograma da Figura 8. Inicialmente, foram selecionados os clusters com maior valor de pertinência para cada amostra para montar a clusterização CRISP dos dados. Em seguida, esse valor foi avaliado com a separação a priori dos dados. O gráfico da Figura 8 mostra que existem muitos valores próximos a 0, ou seja, a clusterização dos dados não possui muita semelhança com a separação a priori dos dados.

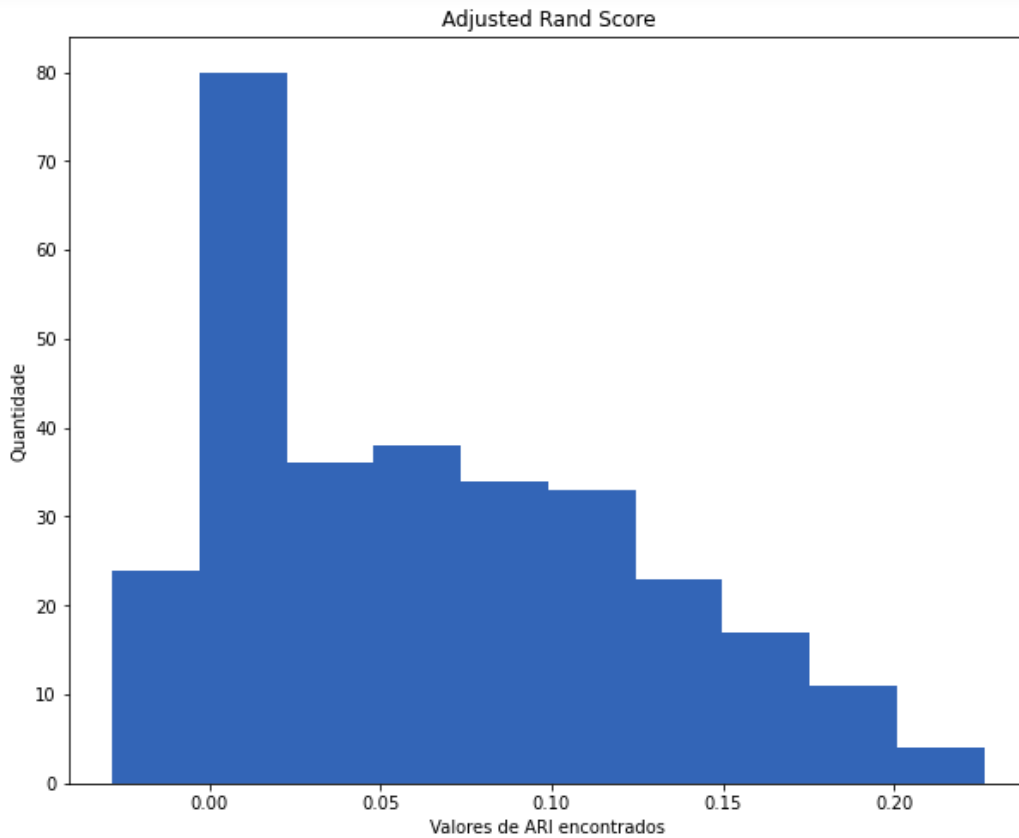


Figura 8: Histograma de ARI de todas as execuções e inicializações aleatórias

F-Measure:

O F-Measure é uma ponderação entre o Recall e a Precision da clusterização, sendo que essa métrica utiliza uma matriz de contingência para retirar as informações úteis. A matriz de contingência é igual a matriz de confusão das classes a priori e as partições encontradas.

A Figura 9 mostra o histograma dos valores de F-Measure no experimento realizado. Vale ressaltar que para calcular o F-Measure é preciso criar uma partição CRISP, como a criada para calcular o ARI.

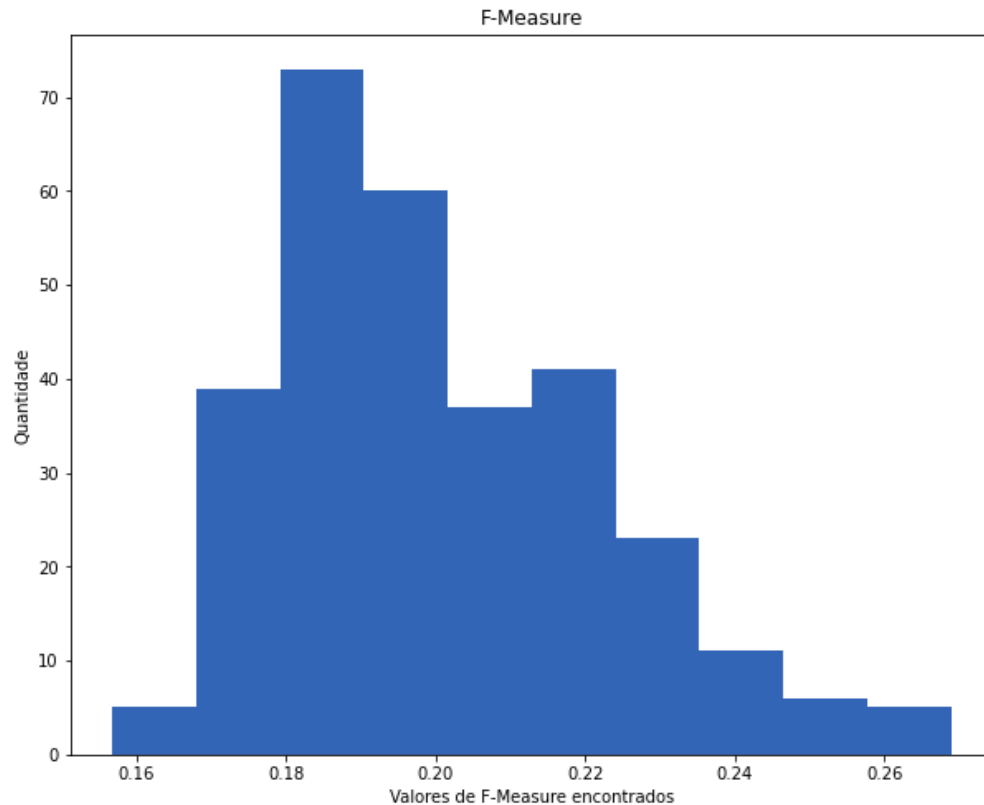


Figura 9: Histograma de F-Measure de todas as execuções

Escolha do melhor modelo:

Para selecionar o melhor resultado para cada mudança de parâmetro, foi escolhida a inicialização aleatória com o menor valor da função de custo. A Tabela 1 mostra os melhores resultados considerando o menor valor de J encontrado nas 100 inicializações aleatórias da matriz de pertinência, além de informar os resultados das métricas avaliadas.

Avaliando o resultado e comparando com os histogramas apresentados, é notório que houve melhores resultados considerando apenas as métricas avaliadas. Por exemplo, o maior valor de F-Measure

mostrado na tabela é 0,1824, porém, encontramos no histograma da F-Measure valores até cerca de 0,27 com um valor de **J** maior que os selecionados como melhores.

Portanto, selecionamos a partição com valor **m** igual a 1.6 como a partição com melhor resultado, pois ela obteve resultados levemente maiores na F-Measure e no ARI. Porém, avaliamos a separação dos dados com algumas partições com os melhores nas métricas.

Valor de m utilizado	J - Função de custo	F-Measure	ARI	PE	MPC	w (aleatoriedade na inicialização)
1.1	1.10493	0.182480	0.01019	0.000825	0.99967	93
1.6	0.449938	0.181859	0.013080	0.11267	0.96255	22
2.0	0.064732	0.17879	0.00887	0.080185	0.98184	98

Tabela 3: Valores dos parâmetros do melhor resultado para cada uma das métricas avaliadas

3.5 Análise do melhor resultado

- Para o melhor resultado imprimir: i) os protótipos ii) a matrix de confusão da partição crisp versus a partição a priori; iii) o Modified partition coefficient e o Partition entropy v) O índice de Rand corrigido, a F-measure e erro de classificação.

Figura 10: Descrição da atividade de visualização das informações do melhor resultado

i) Protótipos do melhor resultado

Na Tabela 4 é apresentada a posição de cada protótipo nas 8 features disponíveis dos dados.

Protótipos	Feature 0	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7
Protótipo 0	0.495432	0.495256	0.501227	0.258262	0.5	0.0	0.499054	0.275499
Protótipo 1	0.662582	0.649665	0.492881	0.290595	0.99886	0.0	0.526019	0.25268
Protótipo 2	0.394642	0.383293	0.497410	0.151234	0.999973	0.0	0.530750	0.362294
Protótipo 3	0.571249	0.521402	0.460401	0.203001	0.5	0.829541	0.519781	0.40331
Protótipo 4	0.454812	0.526925	0.466934	0.174350	0.5	0.0	0.526506	0.772470

Protótipo 5	0.478467	0.557906	0.509643	0.636973	0.5	1.7e-06	0.514823	0.221632
Protótipo 6	0.467212	0.472401	0.525431	0.287125	0.5	0.584104	0.487815	0.230274
Protótipo 7	0.600729	0.587460	0.472981	0.487699	0.5	0.0	0.532463	0.220378
Protótipo 8	0.782893	0.760238	0.417798	0.272557	0.5	0.0	0.524617	0.220728
Protótipo 9	0.510468	0.511318	0.512704	0.232428	0.5	0.826378	0.519828	0.221666

Tabela 4: Valores das posições de cada protótipo do melhor resultado

Na Figura 11 é mostrado as posições dos protótipos em uma visualização par a par com todas as features, onde cada marcação com uma cor é a posição de um protótipo. É possível perceber que a posição dos protótipos estão muito próximas uma das outras.

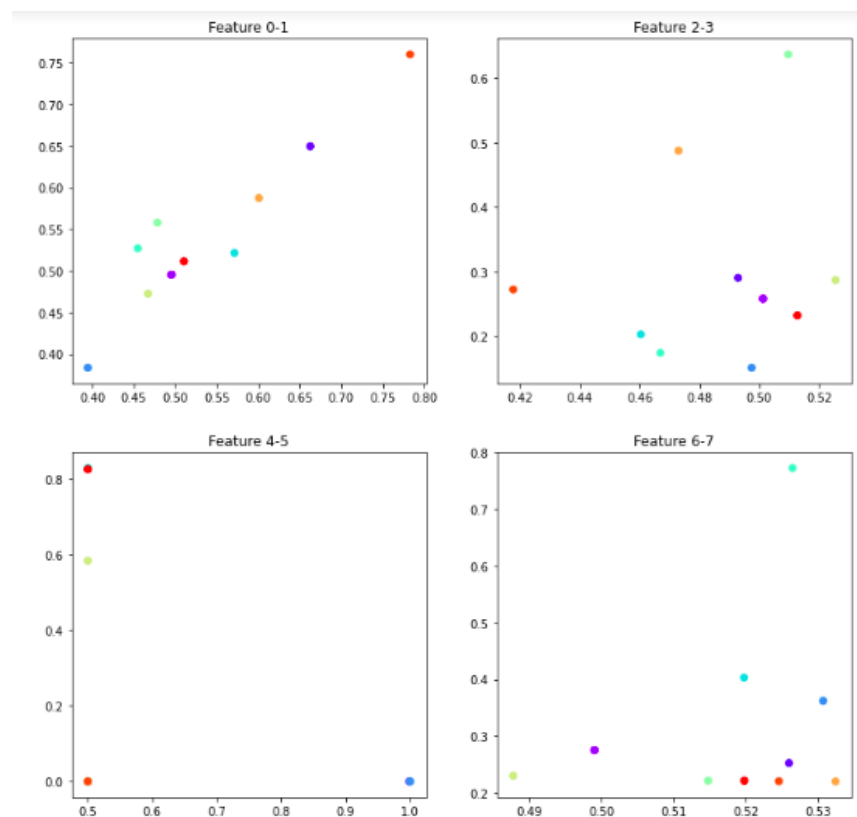


Figura 11: Visualização dos protótipos feature a feature

Analisando a Figura 11, que mostra gráficos da visualização das amostras em *plots* par a par entre as features, agrupando as amostras por

cor em cada cluster, vemos que existe um grande desbalanceamento na quantidade de amostras em um cluster. É possível perceber que a clusterização não funcionou bem. Na Figura 12 é mostrado as posições das amostras com a separação baseada na classe a priori dos dados. A visualização apresenta uma separação não muito clara dos dados disponíveis.

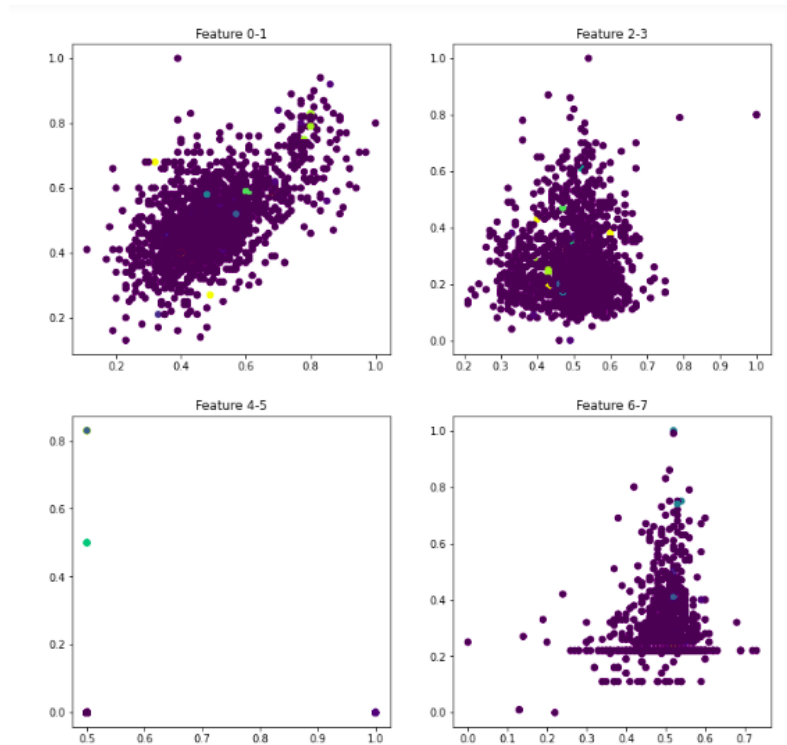


Figura 12: Visualização das amostras separando os clusters por cor da melhor partição

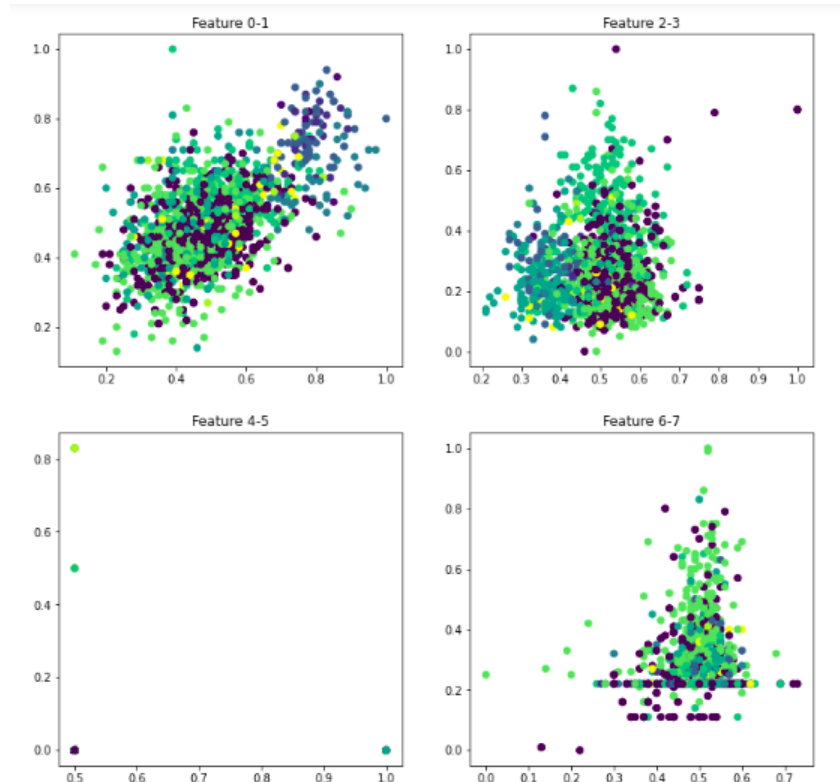


Figura 13: Visualização das classes a priori separando os clusters por cor

Considerando a partição com maior valor de F-Measure, foi observado uma separação mais próxima do esperado. Como observado na Figura 14.

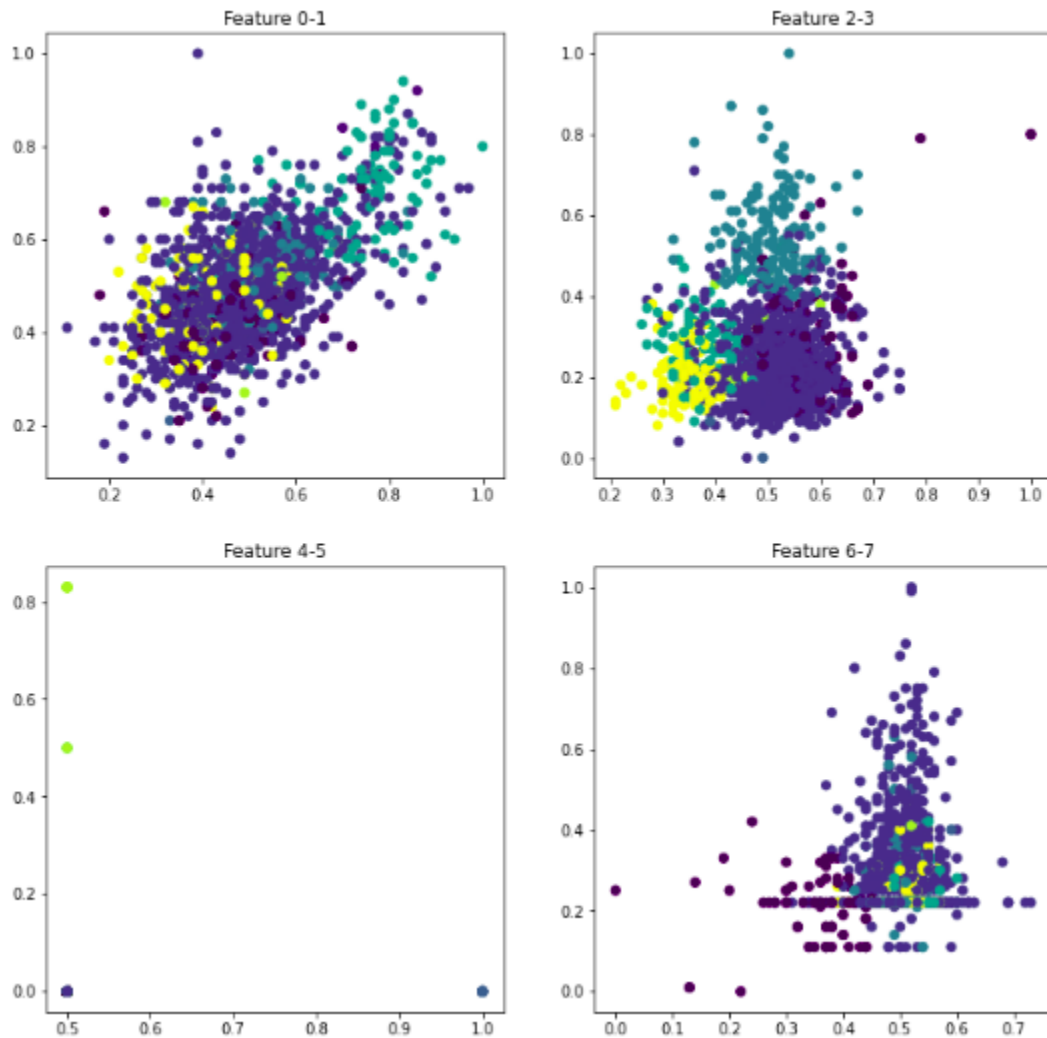


Figura 14: Visualização das classes da partição com valor mais alto de F-Measure

Para visualizar as separações de forma mais simples, fizemos uma redução de dimensionalidade nas 8 features dos dados para 2 features, utilizando Principal Component Analysis (PCA). Na Figura 15 são mostradas as amostras com a separação do método com menor valor na função de custo. Já na Figura 16 é mostrado a separação baseada nas classes a priori, onde vemos que as separações são muito difíceis de serem realizadas. E por fim, temos na Figura 17 a separação predita pela partição com maior valor de F-Measure, com separação visualmente fáceis de entender.

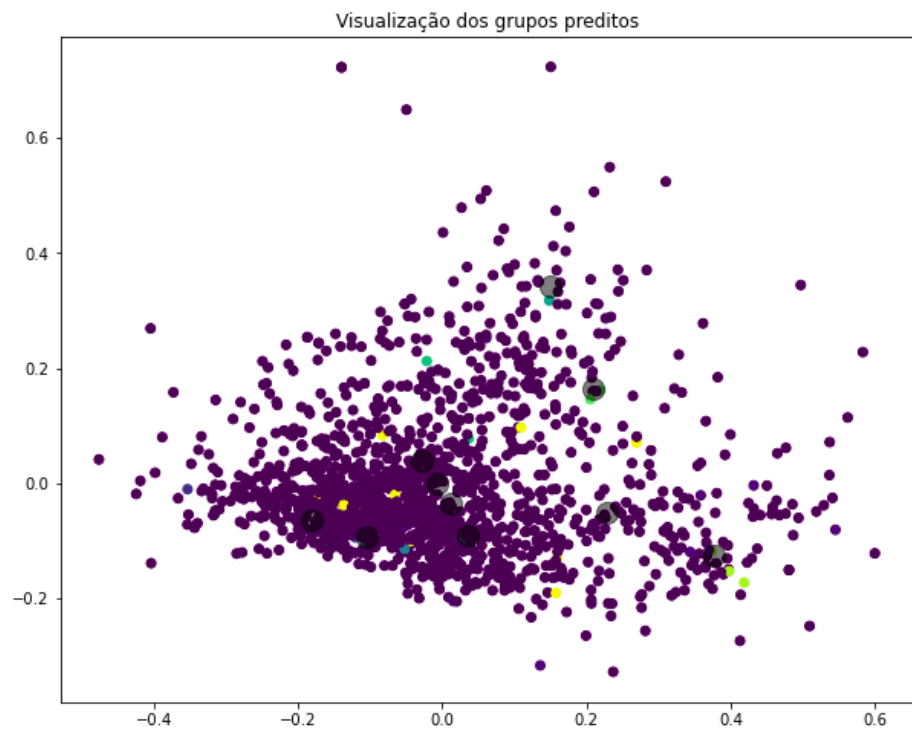


Figura 15: Visualização das amostras com os clusters preditos utilizando PCA

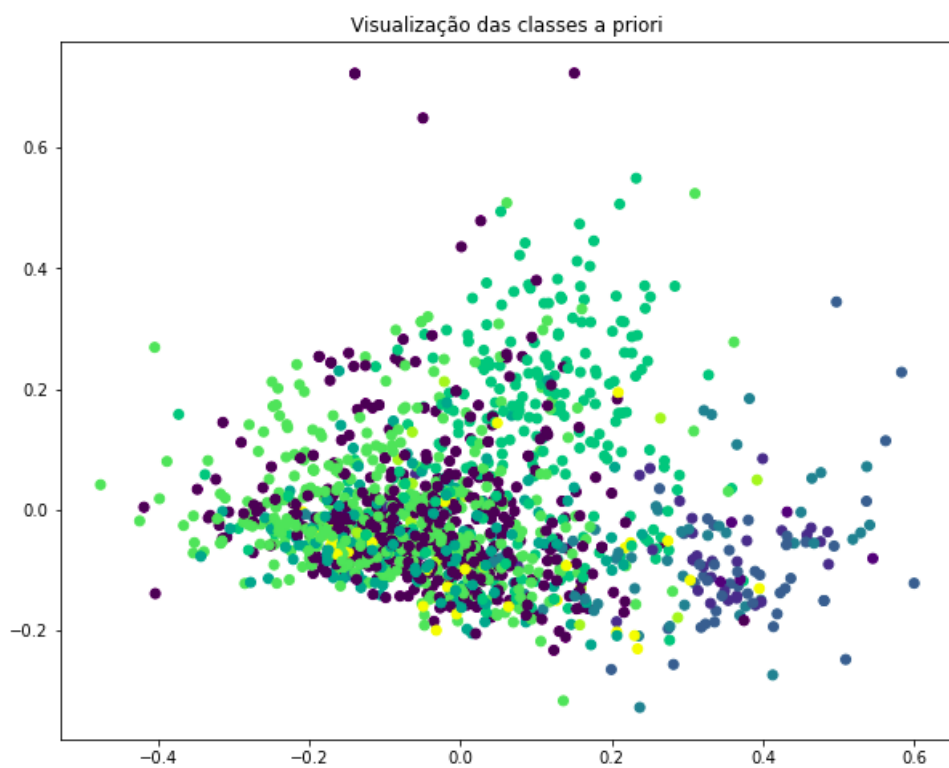


Figura 16: Visualização das amostras com as classes a priori utilizando PCA

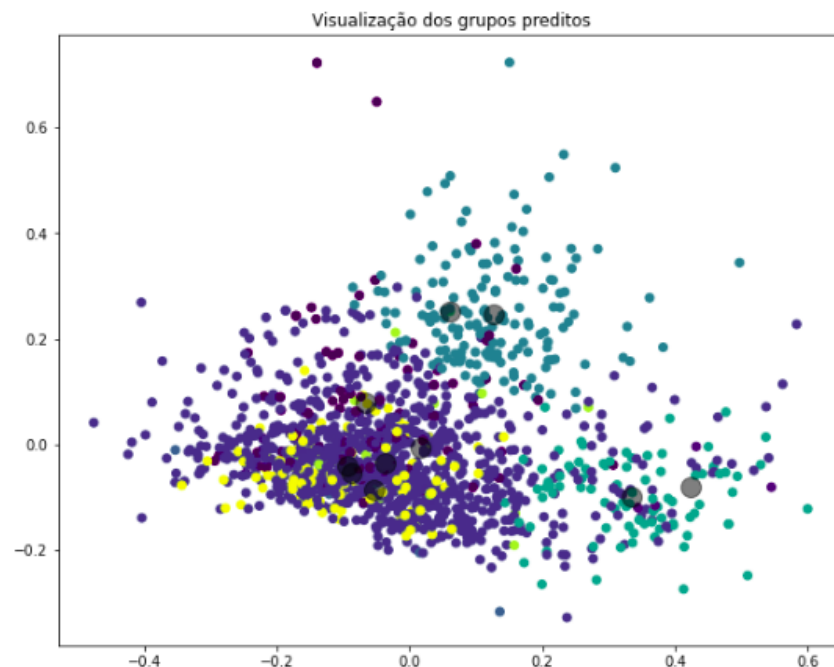


Figura 17: Visualização das amostras com a partição com maior F-Measure utilizando PCA

ii) Matriz de confusão da partição CRISP vs Partição a Priori

A matriz de confusão da partição com menor valor na função de custo comprova que a clusterização jogou todas as amostras no mesmo cluster. Como é visto na Figura 18 as amostras foram agrupadas no mesmo cluster.

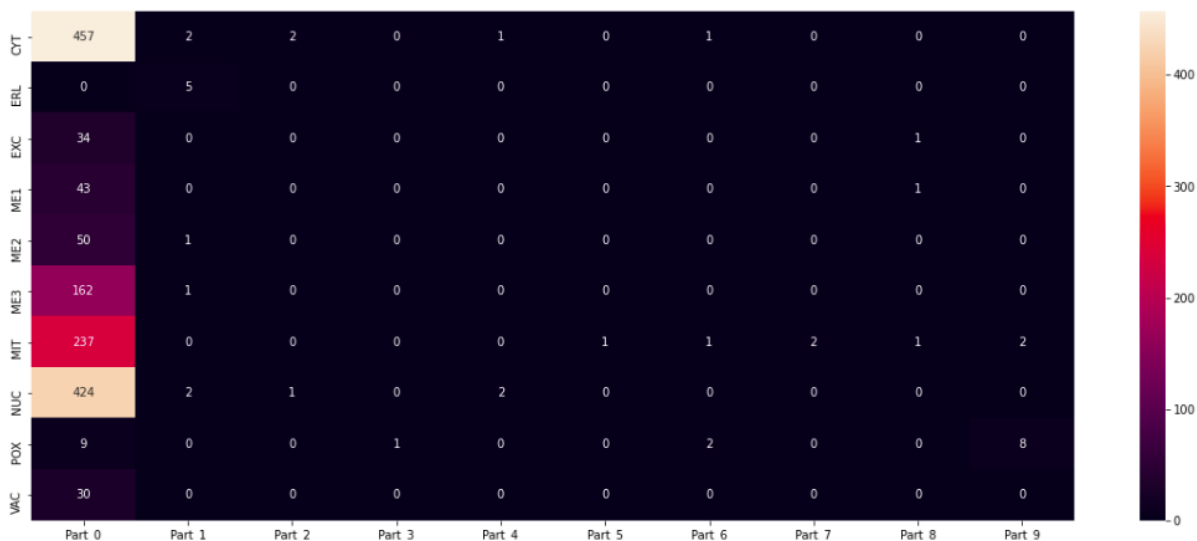


Figura 18: Matriz de confusão da partição com menor valor na função de custo vs partição a priori

Já a Figura 19 mostra a matriz de confusão entre a partição com melhor valor de F-Measure. Nesse exemplo, as amostras são melhor separadas entre os clusters.

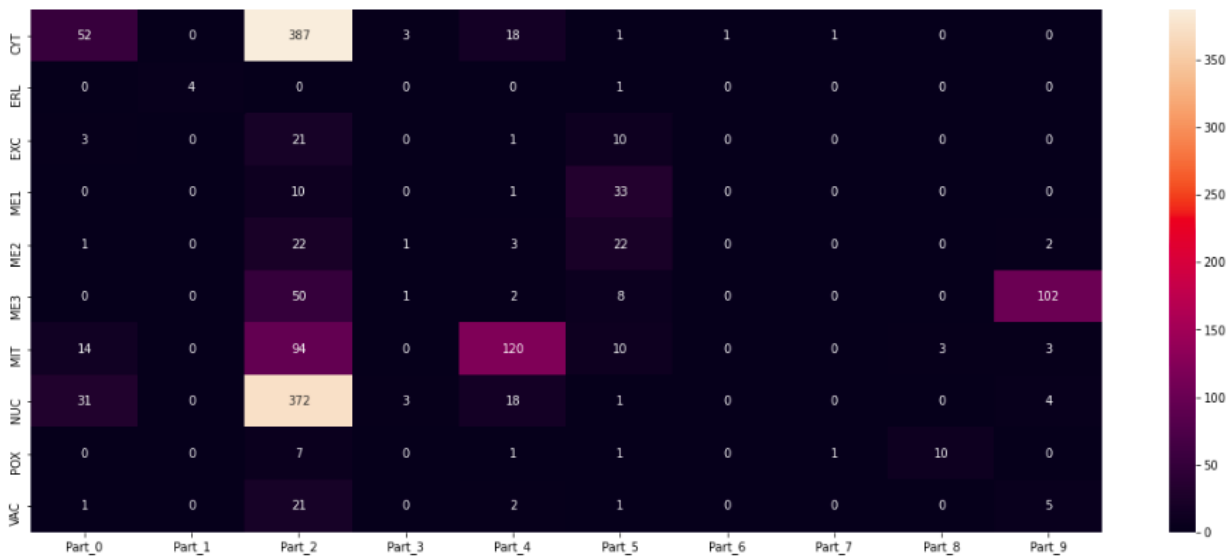


Figura 19: Matriz de confusão da partição com maior F-Measure vs partição a priori

iii) Modified Partition Coefficient e Partition Entropy

A partição com menor valor na função objetivo obteve o valor de 0.99967 em Modified Partition Coefficient, ou seja, a partição está tendendo a partição CRISP. Já o Partition Entropy foi de 0.000825, ou seja, as amostras foram separadas com grande certeza. As duas métricas têm objetivos similares.

Avaliando a partição com melhor valor de F-Measure, foram obtidos 0.9504 em Modified Partition Coefficient, 0.1127 em Partition Entropy. Ou seja, algumas amostras não obtiveram muita certeza na seleção do cluster.

iv) O Índice de Rand Corrigido, F-Measure e o Erro de Classificação

O Índice de Rand Corrigido foi 0.01019, ou seja, considerando a separação a priori, praticamente foi um resultado aleatório, pois foi próximo a zero. Já a F-Measure foi 0.1824, ou seja, 18% na proporção de Recall e Precisão do agrupamento. Consideramos que o Erro de Classificação é o *Overall Error Rate of Classification (OERC)*, ou seja, é a comparação entre a quantidade de amostras juntas nas classes a priori com a partição

CRISP. O OERC da partição com menor valor na função de custo foi 0.01347.

Além disso, avaliamos também a acurácia da partição comparado com a classificação a priori. Nesse caso, foi utilizado uma abordagem de otimização para encontrar a permutação que maximiza a soma da diagonal. E por fim, foi computado a acurácia de 0.3207 na partição com menor valor na função de custo.

Considerando a partição com melhor valor de F-Measure, foram constatados os valores de 0.1932 no Índice de Rand Corrigido, F-Measure de 0.26887, OERC de 0.26415 e acurácia de 0.4676. Ou seja, essa partição foi melhor em quase todas as situações, porém, o valor na função de custo é muito alto, por isso, seguindo os critérios, ele não foi selecionado como melhor. Além disso, o valor do OERC foi menor na partição com menor valor na função de custo, pois essa partição agrupou todas as amostras no mesmo grupo e essa métrica avalia quantas amostras estão no mesmo cluster para cada classe, ou seja, não avalia a qualidade da separação entre os grupos.

4. Questão 2

A questão 2 consistiu das tarefas descritas na Figura abaixo:

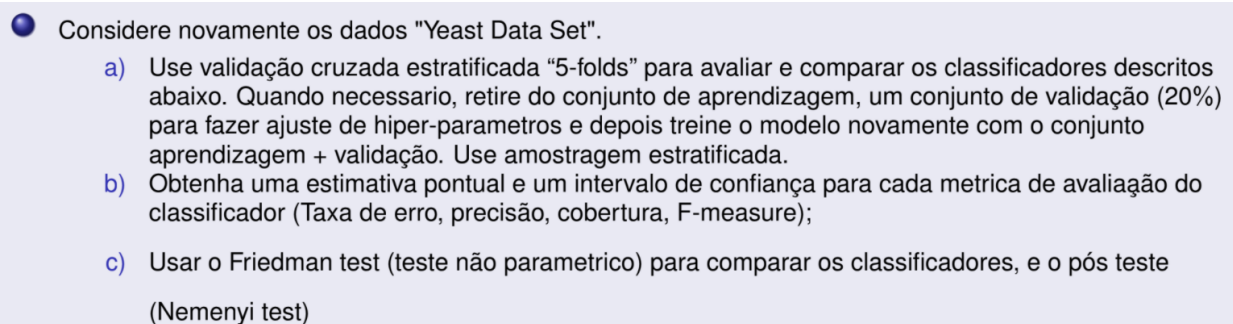
- 
- Considere novamente os dados "Yeast Data Set".
 - a) Use validação cruzada estratificada "5-folds" para avaliar e comparar os classificadores descritos abaixo. Quando necessario, retire do conjunto de aprendizagem, um conjunto de validação (20%) para fazer ajuste de hiper-parametros e depois treine o modelo novamente com o conjunto aprendizagem + validação. Use amostragem estratificada.
 - b) Obtenha uma estimativa pontual e um intervalo de confiança para cada metrica de avaliação do classificador (Taxa de erro, precisão, cobertura, F-measure);
 - c) Usar o Friedman test (teste não parametrico) para comparar os classificadores, e o pós teste (Nemenyi test)

Figura 20: o enunciado da questão 2

O restante dessa seção é organizado ao redor de cada algoritmo (subseções 4.1-4.5), compreendendo cada uma das letras a e b, enquanto a letra c é apresentada na subseção 4.6

4.1 Classificador bayesiano gaussiano

- i) Classificador bayesiano gaussiano: considere a seguinte regra de decisão: afetar o exemplo \mathbf{x}_k à classe ω_l se $P(\omega_l|\mathbf{x}_k) = \max_{i=1}^{10} P(\omega_i|\mathbf{x}_k)$ com $P(\omega_i|\mathbf{x}_k) = \frac{p(\mathbf{x}_k|\omega_i)P(\omega_i)}{\sum_{r=1}^c p(\mathbf{x}_k|\omega_r)P(\omega_r)}$ ($1 \leq l \leq 10$)
- a) Use a **estimativa de máxima verossimilhança** para $P(\omega_i)$
- b) Para cada classe ω_i ($i = 1, 2$) use a seguinte estimativa de máxima verossimilhança de $p(\mathbf{x}_k|\omega_i) = p(\mathbf{x}_k|\omega_i, \theta_i)$, supondo uma normal multivariada:
- $$p(\mathbf{x}_k|\omega_i, \theta_i) = (2\pi)^{-\frac{d}{2}} (|\Sigma_i^{-1}|)^{\frac{1}{2}} \exp \left\{ -\frac{1}{2} (\mathbf{x}_k - \mu_i)^T \Sigma_i^{-1} (\mathbf{x}_k - \mu_i) \right\}, \text{ onde}$$
- $$\theta_i = \left(\begin{matrix} \mu_i \\ \Sigma_i \end{matrix} \right), \Sigma_i = \text{diag}(\sigma^2, \dots, \sigma^2)$$
- $$\mu_i = \frac{1}{n} \sum_{k=1}^n \mathbf{x}_k, \mu_{ij} = \frac{1}{n} \sum_{k=1}^n x_{kj}$$
- $$\sigma^2 = \frac{1}{d \times n} \sum_{k=1}^n \|\mathbf{x}_k - \mu_i\|^2 = \frac{1}{d \times n} \sum_{k=1}^n \sum_{j=1}^d (x_{kj} - \mu_{ij})^2 \quad (1 \leq j \leq d)$$

Figura 21: descrição do classificador bayesiano gaussiano

4.2.1 Fundamentação Teórica

O classificador bayesiano consiste em atribuir uma classe a um dado vetor de entrada através da fórmula de Bayes:

$$P(\omega_j|x) = \frac{p(x|\omega_j)P(\omega_j)}{p(x)}$$

Em que (no caso de termos duas classes)

$$p(x) = \sum_{j=1}^2 p(x|\omega_j)P(\omega_j)$$

Note que se tratando de c classes a soma iria de $j = 1$ até o valor c .

Os termos na fórmula de Bayes podem ser lidos como:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

Sendo *posterior* a probabilidade a posteriori, *likelihood* é a densidade condicional, *prior* é a probabilidade a priori e *evidence* é o fator de

evidência, que atua como um fator de escala para garantir que a soma das probabilidades a posteriori seja 1 (Duda et al. 2001).

No classificador em questão, além de utilizar a fórmula de Bayes, determina-se o valor da densidade condicional supondo uma normal multivariada:

$$p(\mathbf{x}_k | \omega_i, \theta_i) = (2\pi)^{-\frac{d}{2}} (|\Sigma_i^{-1}|)^{\frac{1}{2}} \exp \left\{ -\frac{1}{2} (\mathbf{x}_k - \mu_i)^T \Sigma_i^{-1} (\mathbf{x}_k - \mu_i) \right\}, \text{ onde}$$

$$\theta_i = \begin{pmatrix} \mu_i \\ \Sigma_i \end{pmatrix}, \Sigma_i = \text{diag}(\sigma^2, \dots, \sigma^2)$$

$$\mu_i = \frac{1}{n} \sum_{k=1}^n \mathbf{x}_k, \mu_{ij} = \frac{1}{n} \sum_{k=1}^n x_{kj}$$

$$\sigma^2 = \frac{1}{d \times n} \sum_{k=1}^n \|\mathbf{x}_k - \mu_i\|^2 = \frac{1}{d \times n} \sum_{k=1}^n \sum_{j=1}^d (x_{kj} - \mu_{ij})^2 \quad (1 \leq j \leq d)$$

Figura 22: Equações do classificador gaussiano

A probabilidade a priori é estimada usando a estimativa de máxima verossimilhança que consiste na frequência em que cada classe está presente no conjunto de dados.

Uma vez determinadas as probabilidades a priori e as densidades condicionais, calculam-se os valores das probabilidades a posteriori para cada classe, em que a classe atribuída é aquela com maior probabilidade a posteriori para aquele dado vetor de entrada.

4.1.2 Desempenho

Uma vez que o classificador bayesiano gaussiano não possui hiperparâmetros a serem escolhidos, foi necessário apenas aplicar a validação cruzada estratificada. Foram escolhidos 5 folds conforme o enunciado e para cada fold calcula-se as métricas de acurácia, F-measure, cobertura e precisão. Ao final da validação cruzada, tira-se a média de cada métrica obtida em cada *fold*, também é calculado o desvio padrão e determina-se o intervalo de confiança, assim são obtidos os valores das respectivas métricas. O intervalo de confiança é determinado como:

$$\bar{x} \pm Z_c \frac{\sigma}{\sqrt{n}}$$

O Z_c é obtido na tabela normal e vale 1,96 num intervalo de 95% de confiança.

Métrica	Média	Desvio Padrão	Intervalo de confiança
Acurácia	0.557	0.029	0.557 ± 0.025
Precisão	0.519	0.047	0.519 ± 0.041
Cobertura	0.541	0.055	0.541 ± 0.048
F-Measure	0.513	0.051	0.513 ± 0.045

Tabela 5: desempenho do classificador bayesiano gaussiano

4.2 K-vizinhos

Treine um classificador bayesiano baseados em k-vizinhos. Use a distância Euclidiana para definir a vizinhança. Use conjunto de validação para fixar o o número de vizinhos k .

Figura 23: descrição do classificador baseado em k vizinhos

4.2.1 Fundamentação Teórica

Do ponto de vista bayesiano, o classificador baseado em k-vizinhos estima diretamente a probabilidade a posteriori $p(x|w_j)$ através da densidade de exemplos k_j associados a cada classe com total de exemplos n_j num volume V do espaço de atributos:

$$^{\wedge}P(x|w_j) = \frac{k_j/n_j}{V}$$

O Classificador também estima a probabilidade a priori $p(w_j)$ através da quantidade de exemplos em cada classe n_j e da quantidade total n de exemplos:

$$^{\wedge}P(w_j) = \frac{n_j}{n}$$

Assim, a regra de decisão se resume, multiplicando e cancelando as equações acima, a: decidir pela classe i se:

$$k_i > k_j \forall i \neq j$$

4.2.2 Obtenção de Hiperparâmetros

O melhor k foi obtido através do uso de um conjunto de validação específico para este item com k -fold. O número de folds foi escolhido como cinco (20% de validação, conforme enunciado). Assim, não foi possível que o k -fold interno fosse estratificado, isto é, que houvesse ao menos um membro de cada classe em cada fold. Pois havia classes com apenas 5 representantes, ao total, e um deles já estava no conjunto de teste. Mas o k -fold externo foi estratificado, também de acordo com o enunciado.

Aplicamos a técnica de Grid Search, variando k de 2 a 40 com passo igual a 1.

Após a obtenção do melhor k , o desempenho do algoritmo em si foi avaliado de acordo com diversas métricas. Usamos a técnica Nested Cross Validation para evitar superestimação do desempenho do algoritmo. Em resumo, essa técnica evita que os dados usados para encontrar os hiperparâmetros sejam os mesmos usados para avaliar o modelo. Novamente, a avaliação de desempenho foi feita com cinco folds estratificados, conforme enunciado.

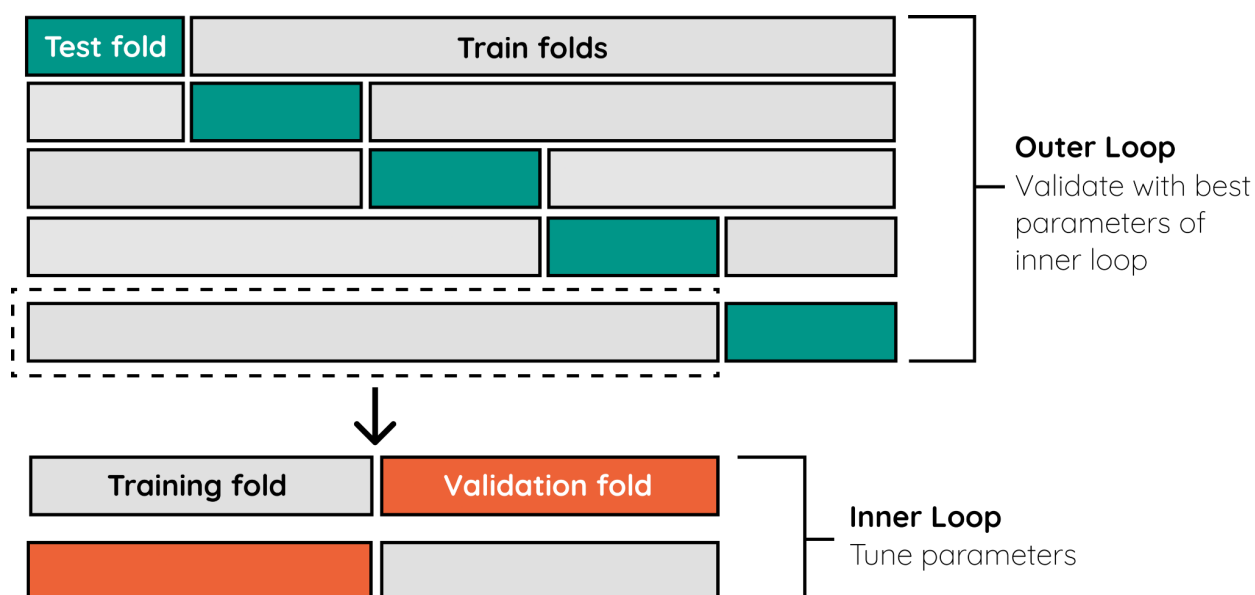


Figura 24: Nested cross-validation.
Fonte da imagem: [1]

4.2.3 Desempenho

Métrica	Média	Desvio Padrão	Intervalo de confiança
Acurácia	0.573	0.026	0.573 ± 0.023
Precisão	0.573	0.026	0.573 ± 0.026
Cobertura	0.524	0.072	0.524 ± 0.063
F-measure	0.516	0.063	0.516 ± 0.055

Tabela 6: desempenho kNN

4.3 Janela de Parzen

Treine um classificador bayesiano baseado em janela de Parzen. Use a função de kernel multivariada produto com o mesmo h para todas as dimensões e a função de kernel Gaussiana unidimensional. Use conjunto de validação para fixar o parâmetro h .

Figura 25: descrição do classificador baseado em janela de Parzen

4.3.1 Fundamentação Teórica

A janela de Parzen consiste em uma técnica para estimar a densidade condicional utilizando um parâmetro h . Em que h é utilizado para determinar um intervalo $(x-h, x+h)$, em que a densidade é a proporção de observações da amostra pertencentes ao intervalo determinado.

A estimação também pode ser realizada através da função janela w :

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

Em que w

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

A função w também pode ser substituída por uma função de kernel K , em que

$$\begin{aligned} K(x) &> 0 \\ \int_{-\infty}^{+\infty} K(x) dx &= 1 \\ K &\text{ é simétrica em torno de } 0 \end{aligned}$$

Este classificador, embora também gaussiano, estima a densidade condicional utilizando a janela de Parzen. Para tanto, utiliza-se a seguinte função kernel multivariada produto:

$$\hat{p}(\mathbf{x}) = \frac{1}{n} \frac{1}{h_1 \dots h_p} \sum_{i=1}^n \prod_{j=1}^p K_j\left(\frac{x_j - x_{ij}}{h_j}\right)$$

Sendo a função de kernel Gaussiana unidimensional:

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$$

Vale também mencionar que os valores de h são os mesmos para todas as dimensões. Uma vez calculadas as densidades condicionais, utiliza-se a fórmula de Bayes para realizar a classificação.

4.3.2 Obtenção de Hiperparâmetros

Foi utilizada a mesma forma de divisão dos dados no classificador KNN, exceto que o parâmetro h foi variado de 0.01 a 0.3 em passos de 0.032.

4.3.3 Desempenho

Métrica	Média	Desvio Padrão	Intervalo de confiança
Acurácia	0.575	0.035	0.575 ± 0.031
Precisão	0.614	0.045	0.614 ± 0.039
Cobertura	0.551	0.023	0.551 ± 0.020
F-Measure	0.559	0.024	0.559 ± 0.021

Tabela 7: desempenho da janela de Parzen

4.4 Regressão logística

Treine um classificador baseado em regressão logística para cada classe e use a bordagem “um contra todos” para classificar os exemplos

Figura 26: descrição do classificador baseado em regressão logística

4.4.1 Fundamentação Teórica

A técnica de regressão logística computa a função logística (também conhecida como sigmóide) de cada um dos atributos do problema, ponderados pelos parâmetros (pesos) do modelo.

$$h_{\theta}(\mathbf{x}^{\top} \boldsymbol{\theta}) = \frac{\exp\{\mathbf{x}^{\top} \boldsymbol{\theta}\}}{1 + \exp\{\mathbf{x}^{\top} \boldsymbol{\theta}\}} = \frac{1}{1 + \exp\{-\mathbf{x}^{\top} \boldsymbol{\theta}\}}$$

A decisão ocorre pela classe 1 se o resultado do cômputo for maior que 0.5 e pela classe 0 se o resultado for menor que 0.5. Frequentemente se trabalha com o log da função logística, especialmente para definir a função de custo necessária para o algoritmo de gradiente descendente. Esse algoritmo é um procedimento iterativo para ajustar os parâmetros do

modelo, o qual consiste em atualizar os pesos pouco a pouco seguindo o gradiente da função de custo a cada passo.

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y_i \ln \{h_{\theta}(\mathbf{x}_i^{\top} \theta)\} + (1 - y_i) \ln \{1 - h_{\theta}(\mathbf{x}_i^{\top} \theta)\}]$$

$$\Theta_{n+1} = \Theta_n - \alpha \frac{\partial}{\partial \Theta_n} J(\Theta_n)$$

Acerca do método um-contra-todos, ele consiste em treinar vários classificadores binários, cada um tentando distinguir uma classe do conjunto de todas as demais. Então, para decidir a que classe o exemplo de treinamento pertence, basta escolher o classificador binário mais confiante.

4.4.2 Desempenho

Métrica	Média	Desvio Padrão	Intervalo de confiança
Acurácia	0.542	0.025	0.542 ± 0.022
Precisão	0.407	0.021	0.407 ± 0.018
Cobertura	0.311	0.016	0.311 ± 0.014
F-Measure	0.331	0.015	0.331 ± 0.013

Tabela 8: desempenho da regressão logística

4.5 Voto majoritário

Treine um classificador usando a regra do voto majoritario a partir dos classificadores i) a iv).

Figura 27: classificador baseado em regra do voto majoritário

4.5.1 Fundamentação Teórica

Este último classificador é composto por todos os 4 antecedentes. Nos classificadores KNN e janela de Parzen foram utilizados o melhor hiperparâmetro usando a acurácia como referência.

Cada classificador foi treinado em todo o conjunto de dados, cada um fazendo sua classificação, isto é, o seu voto. De acordo com o voto da maioria, a classe que mais aparece nas predições é a classe atribuída no processo de voto majoritário. Em caso de empate, seleciona-se a primeira classe após a ordenação de forma crescente.

4.5.2 Desempenho

Métrica	Média	Desvio Padrão	Intervalo de confiança
Acurácia	0.583	0.030	0.580 ± 0.026
Precisão	0.604	0.045	0.609 ± 0.039
Cobertura	0.538	0.041	0.537 ± 0.036
F-Measure	0.549	0.039	0.549 ± 0.034

Tabela 9: desempenho do classificador de voto majoritário

4.6 Comparação de Classificadores

4.6.1 Teste de Friedman

Para realizar o teste de Friedman, escolheu-se as 4 métricas como referências para os classificadores, a seguinte tabela contém o valor médio de cada métrica e também o respectivo ranqueamento:

Metricas	Bayseano Gaussiano	KNN	Parzen	Regressao Logistica	Ensemble
Acuracia	0.557	0.573	0.575	0.542	0.583
Precisao	0.519	0.573	0.614	0.407	0.604
Cobertura	0.541	0.524	0.551	0.311	0.538
F-Measure	0.513	0.516	0.559	0.331	0.549
Ranqueamento	Bayseano Gaussiano	KNN	Parzen	Regressao Logistica	Ensemble
Acuracia	4	3	2	5	1
Precisao	4	3	1	5	2
Cobertura	2	4	1	5	3
F-Measure	4	3	1	5	2
Average	3.5	3.25	1.25	5	2

Tabela 10: métricas do teste de Friedman

Uma vez determinada a tabela, calcula-se o chi quadrado, em que:

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_{j=1}^k R_j^2 - \frac{k(k+1)^2}{4} \right]$$

Para o caso em questão, $N = 4$ e $k = 5$. O R refere-se ao ranqueamento médio, cujos valores estão na linha *average*.

Calculado chi quadrado, agora determina-se o valor de F_F de modo que

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}$$

Agora que a estatística foi obtida, compara-se com o valor crítico $F_{k-1,(k-1)(N-1)}$, em que a hipótese nula deve ser rejeitada caso F_F seja maior que o valor crítico, neste caso há diferença entre os classificadores.

Com o valor de chi quadrado igual a 13.4 e F_F valendo 15.46, utiliza-se a tabela F para verificar o valor crítico, em que a distribuição possui 4 e 12 graus de liberdade e considerando $\alpha = 0.05$, sendo assim o valor crítico pela tabela é 3.2592. Este valor é inferior a 15.46, logo a hipótese nula é rejeitada, neste caso é necessário realizar o pós-teste.

4.6.2 Teste de Nemenyi

No teste de Nemenyi calcula-se a diferença entre os ranqueamentos médios entre todos os classificadores, em que caso essa diferença seja maior ou igual a CD (distância crítica), pode-se dizer que os dois classificadores são significativamente diferentes. A CD é expressa da seguinte forma:

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

O termo q_α é baseado na seguinte tabela (para $\alpha = 0.05$):

k	2	3	4	5	6	7	8	9	10
Nemenyi	1.960	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164

Para $k = 5$, o q_α vale 2.728.

Para os classificadores deste trabalho apenas o janela de Parzen e Regressão Logística obtiveram uma diferença maior que CD, sendo assim classificadores significativamente diferentes.

5. Referências

[1] GROOTENDORST, Maarten. Validating your Machine Learning Model. Towards Data Science, 2019. Disponível em: <https://towardsdatascience.com/validating-your-machine-learning-model-25b4c8643fb7> Acesso em 29 de Julho de 2021.

Branco, Diogo Philippini Pontual. *Agrupamento fuzzy c-medoids semi-supervisionado de dados relacionais representados por múltiplas matrizes de dissimilaridade*. MS thesis. Universidade Federal de Pernambuco, 2017.

Takahama, Tetsuyuki, and Setsuko Sakai. "Fuzzy c-means clustering and partition entropy for species-best strategy and search mode selection in

nonlinear optimization by differential evolution." *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011)*. IEEE, 2011.

de Carvalho, Francisco de AT, Camilo P. Tenório, and Nicomedes L. Cavalcanti Junior. "Partitional fuzzy clustering methods based on adaptive quadratic distances." *Fuzzy Sets and Systems* 157.21 (2006): 2833-2857.