Centro de Informática
UFPE

UNIVERSIDADE FEDERAL DE PERNAMBUCO – UFPE

PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Prof. Dr. Francisco de Assis Tenório de Carvalho

PROJETO DA DISCIPLINA APRENDIZAGEM DE MÁQUINA

EQUIPE:

Carlos Antônio Alves Júnior (caaj@cin.ufpe.br)

Marcos de Souza Oliveira (mso2@cin.ufpe.br)

Matheus Johann Araújo (mja@cin.ufpe.br)

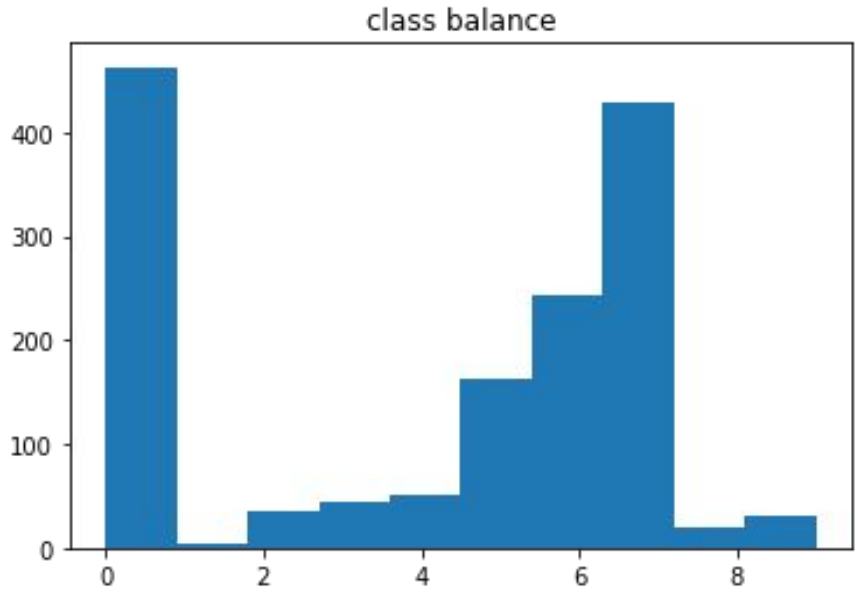
Sumário:

- Análises Iniciais do Conjunto de Dados
- Parte 1: Algoritmo FCM-DFCV e Resultados
- Parte 2: Análise de Classificadores e Resultados

Análises Iniciais do Conjunto de Dados

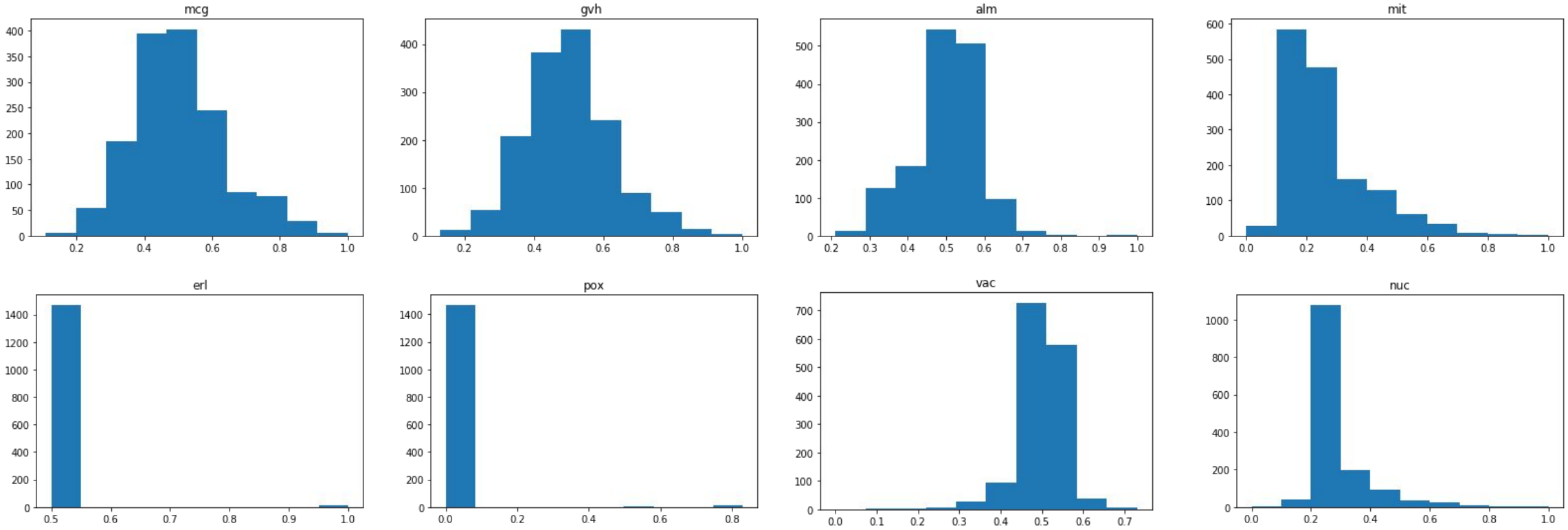
Análises Iniciais do Conjunto de Dados

- Distribuição das Classes:



Análises Iniciais do Conjunto de Dados

- Distribuição dos Atributos:



Análises Iniciais do Conjunto de Dados

Com o objetivo de avaliar a **robustez** dos métodos ao lidar com os **problemas** apresentados, neste trabalho **não** foi empregado um **pré-processamento** no conjunto de dados.

Parte 1: Algoritmo FCM-DFCV e Resultados

Parte 1: Algoritmo FCM-DFCV

- 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 cada partição fuzzy, calcule o Modified partition coefficient e o Partition entropy.
- Para cada partição fuzzy, produza uma partição crisp em 10 grupos e calcule o índice de Rand corrigido e a F-measure.
- Para o melhor resultado imprimir:
  - os protótipos;
  - a matriz de confusão da partição crisp versus a partição a priori;
  - o Modified partition coefficient e o Partition entropy;
  - o índice de Rand corrigido, a F-measure e erro de classificação.

Parte 1: Algoritmo FCM-DFCV

- Considere os hiperparâmetros:
  - c = 10;
  - m = {1.1, 1.6, 2.0};
  - T = 150;
  - e = 10^−10;
- Sequência de passos utilizada:
  - passo 1: geração de protótipos aleatoriamente
  - passo 2: geração matriz U inicial
    - passo 3: geração dos protótipos a partir da matriz U
    - passo 4: geração da matriz M a partir da matriz U e protótipos
    - passo 5: atualize a matriz U a partir da matriz M e protótipos
    - passo 6: calcule a função de custo

Parte 1: Algoritmo FCM-DFCV

- Para os protótipos iniciais:
  - foram escolhidos, aleatoriamente, um único exemplo
  - para ser o representante de cada grupo.
- Matriz U inicial:

$$u_{ik} = \left[\sum_{h=1}^{c} \left\{ \frac{\sum_{j=1}^{p} (x_k^j - g_i^j)^2}{\sum_{j=1}^{p} (x_k^j - g_h^j)^2} \right\}^{1/(m-1)} \right]^{-1} \quad \text{for } i = 1, \ldots, c.$$

Parte 1: Algoritmo FCM-DFCV

- Geração dos protótipos a partir da matriz U:

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

Parte 1: Algoritmo FCM-DFCV

- Geração da matriz M a partir da matriz U e protótipos:

$$\mathbf{M}_i = \begin{pmatrix} \lambda_i^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_i^p \end{pmatrix} \qquad \lambda^j = \frac{\{\prod_{h=1}^{p}[\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m(x_k^h - g_i^h)^2]\}^{1/p}}{\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m(x_k^j - g_i^j)^2} \quad (j = 1, \ldots, p).$$

Parte 1: Algoritmo FCM-DFCV

- Atualização da matriz U:

$$u_{ik} = \left[\sum_{h=1}^{c} \left(\frac{d^2_{\mathbf{M}_i}(\mathbf{x}_k, \mathbf{g}_i)}{d^2_{\mathbf{M}_h}(\mathbf{x}_k, \mathbf{g}_h)} \right)^{1/(m-1)} \right]^{-1} = \left[\sum_{h=1}^{c} \left(\frac{\sum_{j=1}^{p} \lambda_i^j (x_k^j - g_i^j)^2}{\sum_{j=1}^{p} \lambda_h^j (x_k^j - g_h^j)^2} \right)^{1/(m-1)} \right]^{-1}.$$

Parte 1: Algoritmo FCM-DFCV

- Cálculo a função de custo:

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

Parte 1: Algoritmo FCM-DFCV

- Algoritmo:

Algorithm 1 Algoritmo FCM-DFCV

Input O conjunto de dados X

Inicialização T=150, $e = 10^{-10}$, m=[1.1, 1.6, 2.0], J_{old}=99999, J=99999.

1: Construa a matriz **G** aleatoriamente;
2: Construa a matriz **U** a partir de **G** utilizando a Equação 4;
3: **while** $t < T$ and J_{old} - $J > e$ **do**
4: $J_{old} = J$
5: Gere novos protótipos **G** a partir da matriz **U** utilizando a Equação 3.
6: Construa a matriz **M** a partir de **G** e da matriz **U** utilizando a Equação 19.
7: Construa a matriz **U** a partir de **G** e da matriz **M** utilizando a Equação 20.
8: Atualize **J** utilizando a Equação 17.
9: $t = t + 1$
10:
11: **return** J, U, G.

Parte 1: Resultados

- Modified Partition Coefficient (MPC):

	MPC (m=1.1)	MPC (m=1.6)	MPC (m=2.0)
count	100	100	100
mean	0.993981	0.957169	0.951405
std	0.017853	0.115885	0.117852
min	0.900054	0.339813	0.511110
25%	0.999472	0.997256	0.990008
50%	1.000000	0.999804	0.998209
75%	1.000000	1.000000	0.999956
max	1.000000	1.000000	1.000000

Obs.: quanto mais próximo de 1 (um) melhor.

Parte 1: Resultados

- Partition Entropy (PE):

	PE (m=1.1)	PE (m=1.6)	PE (m=2.0)
count	100	100	100
mean	9.388324e-03	6.428793e-02	7.631151e-02
std	2.774241e-02	1.696796e-01	1.719051e-01
min	1.236375e-08	1.407474e-14	2.651796e-11
25%	1.611522e-05	1.069152e-06	1.041061e-04
50%	4.150377e-05	4.053279e-04	2.805545e-03
75%	8.261557e-04	5.679193e-03	2.295916e-02
max	1.479648e-01	1.035130e+00	6.945560e-01

Obs.: quanto mais próximo de 0 (zero) melhor.

Parte 1: Resultados

- Índice de Rand Corrigido (IRC) - Partição Crisp vs Partição a Priori:

	IRC (m=1.1)	IRC (m=1.6)	IRC (m=2.0)
count	100	100	100
mean	0.008086	0.011955	0.010284
std	0.010038	0.014384	0.007058
min	-0.005273	-0.002199	-0.004375
25%	0.002980	0.003713	0.008557
50%	0.008545	0.011632	0.011615
75%	0.011632	0.011632	0.011632
max	0.075881	0.113643	0.057885

Obs.: quanto mais próximo de 1 (um) melhor.

Parte 1: Resultados

- F-Measure (FM) - Partição Crisp vs Partição a Priori:

	FM (m=1.1)	FM (m=1.6)	FM (m=2.0)
count	100	100	100
mean	0.110195	0.098726	0.091678
std	0.108505	0.108695	0.103957
min	0.000000	0.000000	0.000000
25%	0.024090	0.020889	0.021395
50%	0.040094	0.032345	0.034030
75%	0.166611	0.164420	0.129043
max	0.312668	0.311321	0.318059

Obs.: quanto mais próximo de 1 (um) melhor.

Parte 1: Resultados (melhor partição)

- **Menor *J5***: 8.90926821735237e-30
- **Protótipos:**

0	0.498955	0.498834	0.499937	0.259248	0.5	2.089737e-145	0.499494	0.276900
1	0.776150	0.733249	0.411142	0.273819	0.5	2.833927e-107	0.522439	0.220020
2	0.440003	0.529996	0.520002	0.229994	0.5	8.300000e-01	0.509999	0.220000
3	0.502000	0.496667	0.511333	0.253333	0.5	7.420000e-01	0.508667	0.237333
4	0.573723	0.593120	0.376120	0.224092	0.5	1.457722e-56	0.518729	0.227973
5	0.591429	0.577143	0.492857	0.252857	1.0	2.386426e-118	0.525714	0.279286
6	0.523599	0.547891	0.507444	0.561874	0.5	1.579188e-108	0.509171	0.230861
7	0.611038	0.584205	0.493854	0.218669	0.5	4.014276e-102	0.510490	0.228726
8	0.475059	0.490149	0.488930	0.226901	0.5	1.646073e-119	0.522499	0.695924
9	0.649557	0.612392	0.493696	0.214594	0.5	9.386093e-12	0.513643	0.223253

Parte 1: Resultados (melhor partição)

➢ Matriz de Confusão:

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	459	0	4	0	0	0	0
1	0	0	0	0	0	5	0	0	0	0
2	0	0	0	35	0	0	0	0	0	0
3	0	0	0	44	0	0	0	0	0	0
4	0	0	0	50	0	1	0	0	0	0
5	0	0	0	162	0	1	0	0	0	0
6	0	0	0	244	0	0	0	0	0	0
7	0	0	0	426	0	3	0	0	0	0
8	0	0	0	20	0	0	0	0	0	0
9	0	0	0	30	0	0	0	0	0	0

Parte 1: Resultados (melhor partição)

- Índices:

Métrica	Valor
Acurácia*	0.3032
F-Measure	0.0069
IRC	0.0029
MPC	1
PE	0

*Foi utilizada a métrica: *accuracy_score*
Disponível em:
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html#sklearn.metrics.accuracy_score

Parte 2: Análise de Classificadores e Resultados

Parte 2: Classificadores (Bayes Gaussiano)

➢ Cálculo da máxima verossimilhança:

$$p(\mathbf{x}_k|\omega_i,\theta_i) = (2\pi)^{-\frac{d}{2}}(|\boldsymbol{\Sigma}_i^{-1}|)^{\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_k-\mu_i)^{tr}\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}_k-\mu_i)\right\}$$

- Resultados:

Bayes Gaussiano		
z = 1.96, K-Folds = 5	Média	Desvio Padrão
Erro	43.82%	2.314%
Cobertura (Recall)	43.7%	2.1819%
Precisão	43.82%	6.8210%
F-Score (F-Measure)	43.70%	3.8437%

Parte 2: Classificadores (KNN)

➢ Melhor K definido: 4

- Resultados:

K-NN		
z = 1.96, K-folds = 5, k = 4	Média	Desvio Padrão
Erro	40.61%	0.1862%
Cobertura (Recall)	26.29%	2.1819%
Precisão	15.23%	0.0%
F-Score (F-Measure)	18.65%	0.02165%

Parte 2: Classificadores (Janela de Parzen)

➢ Melhor bandwidth definido: 0.5

- Resultados:

Janela de Parzen		
z = 1.96, K-Folds = 5, Bandwidth = 0.5	Média	Desvio Padrão
Erro	43.60%	2.387%
Cobertura (Recall)	41.60%	1.654%
Precisão	43.48%	4.378%
F-Score (F-Measure)	42.01%	1.334%

Parte 2: Classificadores (Regressão Logística)

➢ Função de ativação sigmóide.

- Resultados:

Regressão Logística		
z = 1.96, K-Folds = 5	Média	Desvio Padrão
Erro	43.69%	4.296%
Cobertura (Recall)	43.69%	4.023%
Precisão	43.62%	4.33%
F-Score (F-Measure)	43.76%	3.646%

Parte 2: Classificadores (Ensemble)

➢ Predição feita por voto majoritário.

- Resultados:

Ensemble		
z = 1.96, K-Folds = 5	Média	Desvio Padrão
Erro	15.03%	0.638%
Cobertura (Recall)	28.2%	2.694%
Precisão	8.19%	0.205%
F-Score (F-Measure)	11.11%	0.3801%

Parte 2: Resultados (Teste de Friedman)

➢ Cálculo do F estatístico:

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

- N = número de amostras (5 folds) e k = número de classificadores (5)
- χ_F^2 :

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

R^2: ranking médio do classificador j nos 5-folds

Parte 2: Resultados (Teste de Friedman)

- F Crítico: obtido em qualquer tabela F
- Com $5-1 = 4$ e $(5-1)\times(5-1) = 16$ graus de liberdade.

O valor crítico de $F(4,16)$ para $\alpha = 0.05$ é **3.238872**

	metric	F_f	F_critical	result
0	error_rate	21.316456	3.238872	reject H0
1	f_measure	80.000000	3.238872	reject H0
2	precision	79.333333	3.238872	reject H0
3	recall	37.666667	3.238872	reject H0

Parte 2: Resultados (Pós-teste)

- Aplicando o Nemenyi teste, sendo a **diferença crítica (CD)** definida como:

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

temos:

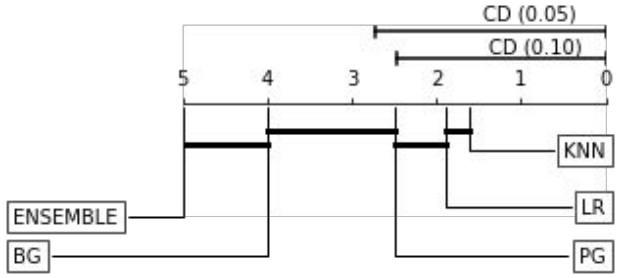
CD (5%) = 2.728

CD (10%) = 2.459

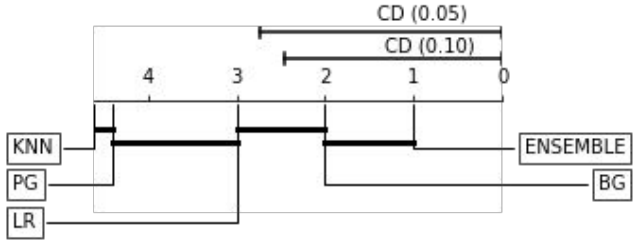
$q\alpha$: é obtido através da Tabela de nemenyi.

Parte 2: Resultados (Pós-teste)

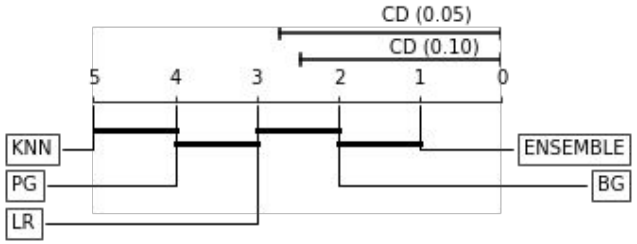
Taxa de erro



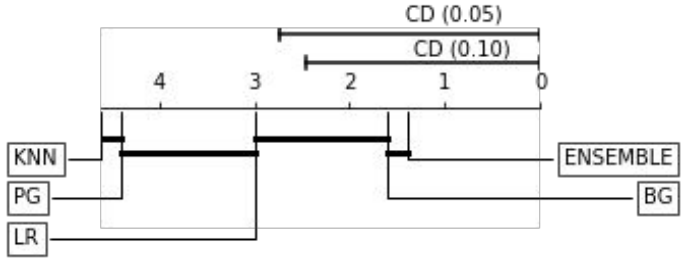
Precisão



F-Measure



Cobertura



Repositório

Código do projeto disponível em:

https://github.com/marcosd3souza/ML-Experiments_FuzzyClustering_ProbClassifiers