Universidade Federal de Pernambuco

Centro de Informática

Programa de Pós-Graduação

**IN1102 - Aprendizagem de Máquina**

Projeto de Disciplina - 2018.1

Relatório de Resultados

Recife, 2018

Cleison Amorim

Dennys Azevedo

Dinaldo Pessoa

Projeto realizada para a disciplina de Aprendizagem de Máquina lecionada pelo Prof. Francisco de A. T. de Carvalho no primeiro semestre de 2018.

Sumário

[Introdução 4](#_Toc516609853)

[Conjunto de dados 5](#_Toc516609854)

[Layout dos dados 6](#_Toc516609855)

[Características dos dados 6](#_Toc516609856)

[Questão 1 8](#_Toc516609857)

[Algoritmo KCM-K-GH 8](#_Toc516609858)

[Preparação dos dados 9](#_Toc516609859)

[Execução do experimento 9](#_Toc516609860)

[Parâmetros de entrada 9](#_Toc516609861)

[Inicialização do algoritmo 10](#_Toc516609862)

[Execução 11](#_Toc516609863)

[Tempo de execução 12](#_Toc516609864)

[Reprodução do experimento 13](#_Toc516609865)

[Avaliação dos resultados 13](#_Toc516609866)

[Índice de Rand Corrigido 13](#_Toc516609867)

[Resultados 14](#_Toc516609868)

[Conclusão 16](#_Toc516609869)

[Questão 2 17](#_Toc516609870)

[Contexto de execução 17](#_Toc516609871)

[Preparação dos dados 17](#_Toc516609872)

[Execução do experimento 18](#_Toc516609873)

[Preparação 18](#_Toc516609874)

[Validação cruzada 19](#_Toc516609875)

[Avaliação dos resultados individuais 21](#_Toc516609876)

[Formatação dos dados 21](#_Toc516609877)

[Estimativa pontual e intervalo de confiança 22](#_Toc516609878)

[Comparação dos classificadores 23](#_Toc516609879)

[Conclusão 24](#_Toc516609880)

[Referências 25](#_Toc516609881)

# Introdução

Este trabalho tem como objetivo apresentar os detalhes da execução, abordagens selecionadas e resultados obtidos a partir do projeto da disciplina Aprendizagem de Máquina lecionada no primeiro semestre de 2018.

O projeto é composto por duas questões que envolvem técnicas de agrupamento e classificação de dados apresentadas no decorrer da disciplina. A primeira questão aborda a utilização do algoritmo de agrupamento baseado em Kernel Gaussiano KCM-K-GH e avaliação de similaridade com relação ao conjunto de dados original. A segunda questão, por sua vez, envolve a aplicação de técnicas de classificação Bayesiana Gaussiana e de classificação Bayesiana baseada em janelas de Parzen, bem como a avaliação de performance e comparação do desempenho desses classificadores.

# Conjunto de dados

Conforme instruções do projeto, foi utilizado o conjunto de dados “*Image Segmentation”* disponibilizado no repositório para aprendizagem de máquina da Universidade da Califórnia [1]. Este conjunto é composto por amostras extraídas randomicamente a partir de uma base de dados de 7 imagens obtidas ao ar livre, as quais foram segmentadas manualmente em regiões de 3 x 3.

O conjunto de dados é composto por 210 amostras de treinamento e 2100 amostras de testes. Essas amostras estão previamente rotuladas entre as 7 classes apresentadas abaixo, e a proporção de distribuição é de 30 instâncias por classe na base de treinamento e 300 instâncias por classe na base de testes.

1. *Brickface* (tijolos)
2. *Sky* (céu)
3. *Foliage* (folhagem)
4. *Cement* (cimento)
5. *Window* (janela)
6. Path (caminho)
7. Grass (grama)

Cada amostra, por sua vez, possui os 19 atributos apresentados a seguir, que descrevem características como centroide, densidade, contraste, intensidade de cores primárias, saturação e matiz dos pixels contidos nas regiões das amostras:

Tabela 1 - Atributos das amostras do conjunto de dados e agrupamento nas visões apresentadas

|  |  |  |  |
| --- | --- | --- | --- |
| **Visões** | | **Atributos** | |
| *Complete view* | *Shape view* | 1 | *REGION-CENTROID-COL* |
| 2 | *REGION-CENTROID-ROW* |
| 3 | *REGION-PIXEL-COUNT* |
| 4 | *SHORT-LINE-DENSITY-5* |
| 5 | *SHORT-LINE-DENSITY-2* |
| 6 | *VEDGE-MEAN* |
| 7 | *VEGDE-SD* |
| 8 | *HEDGE-MEAN* |
| 9 | *HEDGE-SD* |
| *RGB view* | 10 | *INTENSITY-MEAN* |
| 11 | *RAWRED-MEAN* |
| 12 | *RAWBLUE-MEAN* |
| 13 | *RAWGREEN-MEAN* |
| 14 | *EXRED-MEAN* |
| 15 | *EXBLUE-MEAN* |
| 16 | *EXGREEN-MEAN* |
| 17 | *VALUE-MEAN* |
| 18 | *SATURATOIN-MEAN* |
| 19 | *HUE-MEAN* |

Para este projeto, os atributos acima foram divididos em 3 visões:

1. *Complete view* – composta por todos os atributos;
2. *Shape view* – composta pelos 9 primeiros atributos;
3. *RGB view* – composta pelos 10 últimos atributos.

## Layout dos dados

Ao analisar os dados contidos no conjunto, é possível observar que o rótulo dos dados está disposto na primeira coluna da esquerda, e que os demais atributos estão dispostos nas colunas posteriores, na mesma ordem apresentada na seção acima.

Tabela 2 - Amostragem das 3 primeiras linhas e 11 primeiras colunas do conjunto de dados

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **REGION-CENTROID-COL** | **REGION-CENTROID-ROW** | **REGION-PIXEL-COUNT** | **SHORT-LINE-DENSITY-5** | **SHORT-LINE-DENSITY-2** | **VEDGE-MEAN** | **VEDGE-SD** | **HEDGE-MEAN** | **HEDGE-SD** | **INTENSITY-MEAN** |  |
| GRASS | 110 | 189 | 9 | 0 | 0 | 1,000000 | 0,666667 | 1,222222 | 1,186342 | 12,925926 |  |
| GRASS | 86 | 187 | 9 | 0 | 0 | 1,111111 | 0,720082 | 1,444444 | 0,750309 | 13,740741 |  |
| GRASS | 225 | 244 | 9 | 0 | 0 | 3,388889 | 2,195113 | 3,000000 | 1,520234 | 12,259259 |  |

Tabela 3 – Amostragem das 3 primeiras linhas e 9 últimas colunas do conjunto de dados

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RAWRED-MEAN** | **RAWBLUE-MEAN** | **RAWGREEN-MEAN** | **EXRED-MEAN** | **EXBLUE-MEAN** | **EXGREEN-MEAN** | **VALUE-MEAN** | **SATURATION-MEAN** | **HUE-MEAN** |
| 10,888889 | 9,222222 | 18,666668 | -6,111111 | -11,111111 | 17,222221 | 18,666668 | 0,508139 | 1,910864 |
| 11,666667 | 10,333334 | 19,222221 | -6,222222 | -10,222222 | 16,444445 | 19,222221 | 0,463329 | 1,941465 |
| 10,333334 | 9,333334 | 17,111110 | -5,777778 | -8,777778 | 14,555555 | 17,111110 | 0,480149 | 1,987902 |

## Características dos dados

Com o objetivo de analisar as características das amostras que serão utilizadas nas questões do projeto, foi utilizada a biblioteca *Pandas* [2], a qual possibilitou extrair as informações contidasna Tabela 4. As linhas da tabela correspondem aos atributos das amostras e cada coluna corresponde a uma característica obtida pela biblioteca.

É possível observar a partir das propriedades mínimo, máximo e média, que os atributos não possuem uma escala homogênea, e também apresentam média e desvio padrão diferentes. Isso pode influenciar os resultados de alguns algoritmos que se demonstram mais sensíveis a esse tipo de variação, e uma análise quanto à necessidade de normalização dos atributos das amostras deverá ser realizada para cada uma das questões a seguir.

Quando analisamos os atributos das amostras de forma individual, percebemos que alguns deles podem ser pouco significativos no processo de agrupamento e classificação das questões:

1. *REGION-PIXEL-COUNT* – apresenta média 9 e não há desvio padrão. Em outras palavras, é um valor constante que é irrelevante por não ser capaz de contribuir na distinção entre as amostras nos processos de agrupamento e classificação. Além disso, pode causar complicações na execução do experimento, como por exemplo, a aproximação de zero ou anulação dos valores calculados de diferença ou distância, potencializando assim erros aritméticos de divisão nos cálculos subsequentes;
2. *SHORT-LINE-DENSITY-5* – através da média, desvio padrão e principalmente dos percentis, é possível perceber que este atributo possui valores muito pequenos e próximos a zero. Uma análise na tabela de dados completa, permite-nos identificar que 87% das amostras (ou 1.838 de um total de 2.100) possuem valor zerado; os outros 13% possuem os valores iguais aos números 0,11111111, 0,22222222 ou 0,33333333. Apesar de não se tratar de um valor constante, este atributo tem pouco potencial de contribuir com o processo de distinção das amostras e pode causar complicações como no atributo apresentado anteriormente;
3. *SHORT-LINE-DENSITY-2* – através da mesma análise realizada para a propriedade acima, percebe-se que seu comportamento é equivalente. Observando a tabela completa de dados, identifica-se que 96% das amostras (ou 2.020 de todas as 2.100 amostras) possuem o valor dessa propriedade zerado; apenas 4% das demais possuem valores iguais aos números 0,11111111 ou 0,22222222. Por esse motivo, esta variável possui um potencial ainda menor de contribuir com os algoritmos de classificação e agrupamento.

Tabela 4 - Descrição das características estatísticas dos dados obtidas através da biblioteca Pandas [2]

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **cont.** | **média** | **desvio padrão** | **mín.** | **máx.** | **percentis** | | | |
| **25%** | **50%** | **75%** |
| **REGION-CENTROID-COL** | 2100 | 124,940476 | 72,858637 | 1,000000 | 254,000000 | 62,000000 | 121,000000 | 188,250000 |
| **REGION-CENTROID-ROW** | 2100 | 123,483333 | 57,431428 | 11,000000 | 251,000000 | 81,000000 | 122,000000 | 171,250000 |
| **REGION-PIXEL-COUNT** | 2100 | 9,000000 | 0,000000 | 9,000000 | 9,000000 | 9,000000 | 9,000000 | 9,000000 |
| **SHORT-LINE-DENSITY-5** | 2100 | 0,014921 | 0,041024 | 0,000000 | 0,333333 | 0,000000 | 0,000000 | 0,000000 |
| **SHORT-LINE-DENSITY-2** | 2100 | 0,004550 | 0,023573 | 0,000000 | 0,222222 | 0,000000 | 0,000000 | 0,000000 |
| **VEDGE-MEAN** | 2100 | 1,890820 | 2,649453 | 0,000000 | 29,222221 | 0,722222 | 1,277776 | 2,222221 |
| **VEDGE-SD** | 2100 | 5,708299 | 44,989359 | 0,000000 | 991,718400 | 0,349603 | 0,833333 | 1,807406 |
| **HEDGE-MEAN** | 2100 | 2,406772 | 3,469954 | 0,000000 | 44,722225 | 0,833332 | 1,444444 | 2,555556 |
| **HEDGE-SD** | 2100 | 7,904224 | 53,47107 | -1,59E-08 | 1386,329000 | 42,16377 | 98,974420 | 2,251852 |
| **INTENSITY-MEAN** | 2100 | 37,047654 | 38,135291 | 0,000000 | 143,444440 | 7,472222 | 21,666666 | 53,277778 |
| **RAWRED-MEAN** | 2100 | 32,806667 | 34,994538 | 0,000000 | 137,111110 | 7,000000 | 19,666668 | 47,333332 |
| **RAWBLUE-MEAN** | 2100 | 44,205556 | 43,510119 | 0,000000 | 150,888890 | 9,666667 | 27,777779 | 65,000000 |
| **RAWGREEN-MEAN** | 2100 | 34,130741 | 36,303768 | 0,000000 | 142,555560 | 6,222222 | 20,444445 | 46,388888 |
| **EXRED-MEAN** | 2100 | -12,722963 | 11,588214 | -49,666668 | 9,888889 | -18,583333 | -10,888889 | -4,222222 |
| **EXBLUE-MEAN** | 2100 | 21,473704 | 19,654107 | -12,444445 | 82,000000 | 4,305556 | 19,666666 | 36,111110 |
| **EXGREEN-MEAN** | 2100 | -8,750741 | 11,606996 | -33,888890 | 24,666666 | -17,000000 | -11,000000 | -3,222222 |
| **VALUE-MEAN** | 2100 | 45,162381 | 42,900582 | 0,000000 | 150,888890 | 11,777778 | 28,666666 | 65,000000 |
| **SATURATION-MEAN** | 2100 | 0,427259 | 0,228458 | 0,000000 | 1,000000 | 0,284934 | 0,375064 | 0,540228 |
| **HUE-MEAN** | 2100 | -1,365147 | 1,544278 | -3,044175 | 2,912480 | -2,188539 | -2,052625 | -1,565745 |

# Questão 1

Esta questão envolve um problema de classificação por meio de agrupamento dos dados apresentados na seção Conjunto de dados. A questão determina a utilização da variante KCM-K-GH do algoritmo de clusterização *c-means* de Kernel Gaussiano com computação automatizada dos hiperparâmetros de largura descrita no artigo [3].

## Algoritmo KCM-K-GH

A variante KCM-K-GH do algoritmo apresentado no artigo [3] difere-se porque utiliza um vetor global de hiperparâmetros, em vez de um vetor local. Sendo assim, cada atributo presente nas amostras do conjunto de dados acima possui um hiperparâmetro próprio, que é compartilhado entre todos os *clusters* (ou agrupamentos) e ajustado de forma automática pelo algoritmo.

O algoritmo KCM-K-GH utiliza como função objetivo a equação apresentada abaixo, a qual é minimizada iterativamente segundo os passos descritos a seguir para que seja encontrada a configuração ótima de agrupamentos para os dados.



Esta equação, por sua vez, baseia-se na utilização de uma função de Kernel Gaussiana com vetor global de hiperparâmetros, conforme abaixo:



Em linhas gerais, o algoritmo KCM-K-GH consiste na execução iterativa desses passos:

1. Representação – consiste em calcular os representantes de cada agrupamento, tomando como base os elementos nele contidos. O vetor de representantes (ou protótipos) é calculado segundo a função abaixo. Esses representantes serão utilizados posteriormente no passo de alocação, para identificação de semelhança para novos elementos e atualização dos agrupamentos atuais;



1. Cálculo dos hiperparâmetros de largura – consiste em prover um do vetor de hiperparâmetros de largura global, calculado conforme função abaixo, que será utilizado pela função objetivo. Após esta etapa, cada propriedade das amostras do conjunto de dados passará a ter associado a ele um hiperparâmetro de largura.



1. Alocação – tem por objetivo associar cada amostra do conjunto de dados a um agrupamento. Para isso, é necessário encontrar o agrupamento “vencedor”, ou seja, aquele que é capaz de minimizar mais a função objetivo, tomando como base o seu representante e o vetor global de hiperparâmetros de largura. Esse processo busca identificar o agrupamento cujo o elemento a alocar apresente-se mais semelhante.

## Preparação dos dados

Conforme apresentado na seção Características dos dados, os atributos *REGION-PIXEL-COUNT, SHORT-LINE-DENSITY-5* e *SHORT-LINE-DENSITY-2* são constantes ou zerados em sua maioria e não são capazes de contribuir de forma significativa para os resultados do algoritmo. Além disso, a utilização desses atributos em algumas tentativas fez com que o processamento do algoritmo apresentasse erros aritméticos de divisão por zero devido à variância nula ou pela tendência de igualar a zero os valores calculados para os representantes ou hiperparâmetros. Devido a isso, esses 3 atributos foram removidos do conjunto de dados utilizado no experimento realizado na questão atual.

Ainda conforme observação da seção Características dos dados, quanto à falta de homogeneidade na escala dos atributos, o experimento descrito para esta questão foi executado considerando duas versões dos dados: uma versão original sem normalização e uma versão onde os valores dos atributos foram normalizados para uma escala entre de 0 a 1 com base no somatório dos valores totais de cada atributo. O resultado que essa normalização exerceu sobre o desempenho do algoritmo será analisado na seção Avaliação dos resultados adiante.

## Execução do experimento

Esta seção descreve os passos para realização do experimento do algoritmo de agrupamento desta questão.

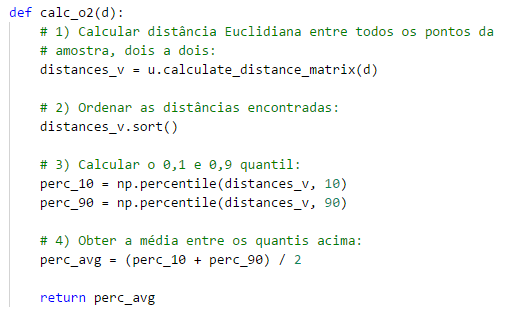
### Parâmetros de entrada

O algoritmo KCM-K-GH requer o conjunto de parâmetros de entrada a seguir, os quais foram obtidos como abaixo:

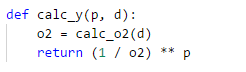
1. *D* (conjunto de dados) – definido na seção Conjunto de dados;
2. *c* (número de agrupamentos) – definido como **7**, conforme enunciado da questão;
3. *γ* (parâmetro de entrada) – calculado através da equação abaixo, onde *p* é o número de atributos das amostras (que para o nosso conjunto de dados é **19**) e σ² é calculado conforme conteúdo adiante:



Para realizar o cálculo do σ² foi implementada a seguinte função, que de acordo com as instruções do enunciado da questão, realiza o cálculo das distâncias entre as amostras, ordena as distâncias e calcula a média entre o seus 0,1 e 0,9 quantis. Mais detalhes sobre o cálculo desse parâmetro podem ser obtidos no artigo [3, p. 379].



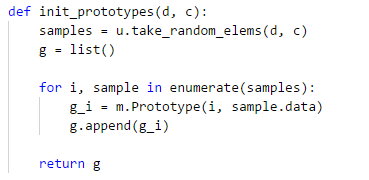
A partir do resultado dessa função é realizado o cálculo do parâmetro *γ* no algoritmo:



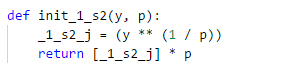
### Inicialização do algoritmo

Uma vez informados os parâmetros de entrada, o algoritmo precisa inicializar as seguintes variáveis para que possa executar as iterações:

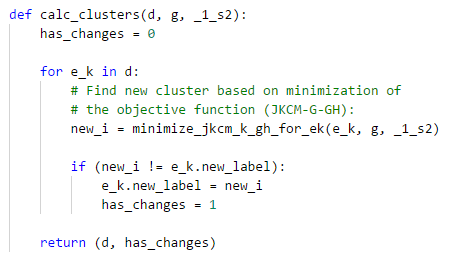
1. *g* (vetor de representantes) – o vetor de representantes (ou protótipos) é inicializado obtendo-se randomicamente *c* amostras distintas a partir do conjunto de dados *D*;



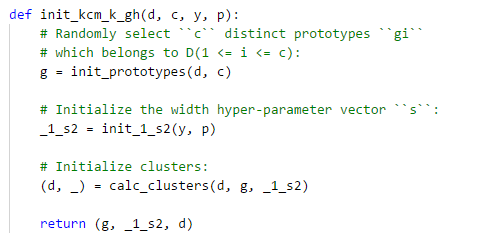
1. (vetor de hiperparâmetros) – o vetor global de hiperparâmetros de largura é inicializado fazendo-se . Nesse momento, os hiperparâmetros para todos os atributos são iguais, e serão ajustados a cada iteração.



1. *Pi* (agrupamentos) – uma vez inicializados os parâmetros acima, é possível executar o passo de alocação descrito na seção Algoritmo KCM-K-GH para prover um agrupamento inicial dos elementos entre os *c* grupos:

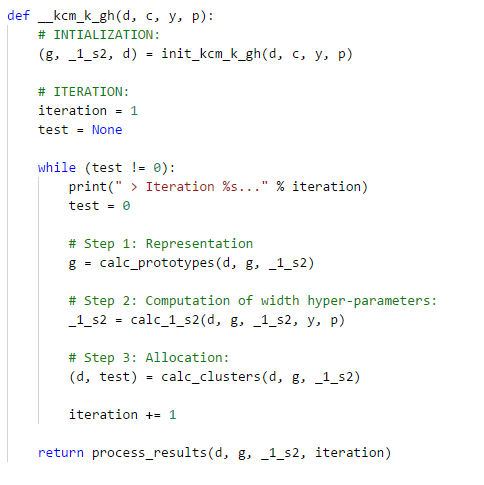


Dessa forma, a implementação da função de inicialização do KCM-K-GH ficou conforme a seguir:



### Execução

Uma vez obtidos os parâmetros de entrada e calculados os demais parâmetros, eles foram informados para a função que implementa o algoritmo do KCM-K-GH. De acordo com o enunciado da questão, o algoritmo deve ser executado **100** vezes para cada uma das visões “*Complete view*”, “*Shape view*” e “*RGB view*” descritas na seção Conjunto de dados. A partir dessas iterações, o melhor resultado deverá ser selecionado segundo a função objetivo do algoritmo (vide seção Algoritmo KCM-K-GH).



A execução do algoritmo acima foi realizada a partir do terminal do Windows PowerShell, pelo script Python *main.py* localizado na pasta “*\questao\_1\kcm-k-gh\kcm-k-gh”*. Esse script recebe como parâmetros o caminho do arquivo com o conjunto de dados, a pasta onde serão gravados os resultados, o número de iterações, o números de agrupamentos e um indicativo de que os dados devem ou não ser normalizados.

> python main.py "..\\data\\segmentation.test" "..\\output\\" 100 7 s

### Tempo de execução

O experimento foi executado em um computador com Processador Intel® Core™ i5-7300 CPU @ 2.60 GHz com 2 núcleos e 4 processadores lógicos, 8 GB de memória RAM. O tempo médio obtido para a execução com dados normalizados foi de 87 minutos (aproximadamente 1,5 horas) e para a execução com os dados não normalizados foi de 117 minutos (aproximadamente 2 horas). É importante salientar que é esperada uma oscilação nos tempos de execução devido ao fato do algoritmo inicializar aleatoriamente os representantes de cada agrupamento. Isso faz com que, inicializado de diferentes formas, o algoritmo necessite de mais ou menos iterações até convergir para uma solução ótima.

Com o intuito de reduzir o consumo de recursos computacionais, as seguintes abordagens foram tomadas na implementação do algoritmo:

1. Cache do conjunto de dados – para diminuir o tempo e o consumo de recursos com a transferência de dados do disco para a memória ao carregar e tratar os dados na execução das iterações, os conjuntos de dados foram guardados após a primeira carga para cada uma das visões. Iterações sob a mesma visão reutilizam os dados em cache;
2. Cache do parâmetro γ – o cálculo desse parâmetro corresponde a um dos passos que mais consome recursos computacionais, uma vez que para o conjunto de dados utilizado ele precisa iterar mais de 2 milhões de vezes calculando a diagonal superior da matriz de distâncias Euclidianas entre as 2.100 amostras. De forma semelhante ao cache anterior, uma vez que este parâmetro foi calculado para uma visão específica, ele é guardado associado a ela. Iterações posteriores sob a mesma visão reutilizaram o parâmetro em cache.

### Reprodução do experimento

Os arquivos fontes com a implementação do algoritmo apresentado aqui estão disponíveis num repositório do GitHub[[1]](#footnote-1) para download e execução.

## Avaliação dos resultados

Esta seção apresentada as técnicas e abordagens utilizadas para avaliar os resultados do experimento detalhado acima.

De acordo com o enunciado da questão, o algoritmo deve ser executado 100 vezes para cada uma das visões apresentadas. Em cada uma das iterações deve ser calculado o Índice de Rand Corrigido, que indicará o grau de similaridade dos agrupamentos produzidos em relação aos grupos originalmente fornecidos no conjunto de dados.

Além disso, deve ser selecionado o melhor resultado para cada uma das visões segundo a função objetivo apresentada na seção Algoritmo KCM-K-GH. Para cada um desses resultados selecionados, deve-se extrair também:

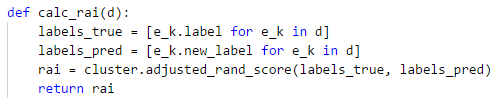
1. Os representantes de cada agrupamento;
2. O número de objetos em cada agrupamento;
3. O vetor de hiperparâmetros;
4. As amostras contidas em cada agrupamento;
5. O Índice de Rand Corrigido.

### Índice de Rand Corrigido

Cada uma das amostras está originalmente rotulada no conjunto de dados com uma classe específica. Dessa forma, pode-se assumir que este conjunto define um agrupamento prévio das amostras, onde cada uma delas pertence àquele grupo do rótulo específico.

Esse conhecimento foi utilizado para avaliar o grau de similaridade dos agrupamentos gerados pelo algoritmo KCM-K-GH com o agrupamento original. Em outras palavras, essa informação nos indica o quão correto estão os agrupamentos que estamos gerando. Para isso utilizou-se o Índice de Rand Corrigido [4] (ou *Adjusted Rand Index*) implementado na biblioteca ScikitLearn [5].

No experimento, cada uma das amostras foi carregada em memória mantendo sua classe original e a criando um novo atributo para sua nova classe, que é atualizado durante as iterações. Dessa forma, ao final da execução essas informações são utilizadas para calcular o Índice de Rand Corrigido:



### Resultados

Serão apresentados a seguir os melhores resultados obtidos na execução do algoritmo KCM-K-GH conforme descrito nas seções anteriores para cada uma das visões.

Foram realizadas duas execuções do experimento descrito acima: uma aplicando a normalização dos dados e outra sem aplicação desse procedimento. O objetivo disso é avaliar se houve perdas decorrentes da falta de homogeneidade na escala dos atributos, ou se o ajuste dos dados pode causar melhoras no processo de agrupamento.

#### Dados normalizados

Nesta seção serão analisados os resultados oriundos da utilização dos dados após um processo de normalização conforme seção Preparação dos dados. A Tabela 5 relaciona os melhores resultados selecionados segundo a minimização da função objetivo do KCM-K-GH, como solicitado no enunciado da questão, para a execução com dados normalizados. A partir dela pode-se perceber que a visão “*RGB view”* obteve o melhor resultado pelo Índice de Rand Corrigido (ou ARI[[2]](#footnote-2)) em comparação com as demais. Isso provavelmente está relacionado ao fato dessa visão considerar as propriedades mais significativas das amostras do conjunto de dados, o que faz com que seja favorecida a comparação entre elas, dado a maior facilidade de comparação. O ARI obtido para esta visão foi de aproximadamente 0,42. Este resultado, apesar de ser o melhor perante as demais visões, ainda denota uma similaridade menor que 50% com o agrupamento original dos dados.

Em segundo lugar, com uma diferença pequena, está a visão *“Complete view”* com um valor de ARI de aproximadamente 0,43. Há uma diferença significativa do segundo e para o terceiro colocado e, a justificativa para isso, é de que a *“Complete view”* considera todos os atributos da amostra (inclusive aqueles utilizados pela *“Shape view”*) e isso desempenha um papel bastante favorável para o segundo colocado. Em contrapartida, aqueles atributos considerados para a visão *“Shape view”* são poucos e ainda foram reduzidos após desconsiderados aqueles que eram constantes ou nulos.

Tabela 5 - Melhores resultados segundo a minimização da função objetivo do KCM-K-GH, com dados normalizados

| Visão | JKCM-K-GH | ARI | Iterações | Tempo | Objetos por agrupamento | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| c1 | c2 | c3 | c4 | c5 | c6 | c7 |
| Complete View | 0,627273 | 0,427593 | 16 | 00:00:23 | 437 | 136 | 14 | 472 | 580 | 164 | 297 |
| **RGB View** | **0,119319** | **0,416436** | **15** | **00:01:05** | **398** | **136** | **573** | **297** | **286** | **246** | **164** |
| Shape View | 1,793214 | 0,165376 | 19 | 00:00:58 | 305 | 10 | 500 | 82 | 341 | 443 | 419 |

Os demais dados referentes aos representantes de cada agrupamento, ao vetor de hiperparâmetros e aos objetos contidos em cada partição após a execução do algoritmo com os dados normalizados estão disponíveis no anexo abaixo.



Se quiséssemos analisar se o algoritmo foi capaz de alcançar um ARI maiores que os apresentados acima, poderíamos optar por selecionar as linhas que obtiveram os maiores valores de ARI para cada uma das visões a partir da tabela completa de resultados. Fazendo isso, obtemos a Tabela 6, que nos mostra que o algoritmo KCM-K-GH executado neste experimento foi capaz de fornecer valores de ARI superiores, e a visão *“Complete view”* conseguiu alcançar um grau de semelhança de aproximadamente 0,65, passando a assumir a primeira posição entre as visões.

Tabela 6 - Melhores resultados pela maximização do Índice de Rand Corrigido (ou ARI), com dados normalizados

| Visão | JKCM-K-GH | ARI | Iterações | Tempo | Objetos por agrupamento | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| c1 | c2 | c3 | c4 | c5 | c6 | c7 |
| **Complete View** | **0,764145** | **0,648560** | **14** | **00:00:20** | **43** | **541** | **581** | **297** | **15** | **323** | **300** |
| RGB View | 0,119314 | 0,415079 | 11 | 00:00:12 | 245 | 573 | 401 | 164 | 297 | 136 | 284 |
| Shape View | 1,772301 | 0,311697 | 37 | 00:00:14 | 45 | 405 | 398 | 317 | 362 | 10 | 563 |

#### Dados não-normalizados

Nesta seção realizaremos a análise dos resultados obtidos da utilização dos dados originais, sem aplicação do processo de normalização acima. Na Tabela 7 observamos a seleção dos melhores resultados para as visões segundo o enunciado da questão, que define a função objetivo do algoritmo como critério para isso. A partir dela temos que a visão com maior grau de similaridade é novamente a *“RGB view”*, cujovalor do ARI é de aproximadamente 0,51. Por esse comportamento, pode-se endossar a teoria proposta acima que afirma que o conjunto de propriedades consideradas por esta visão é o que apresenta as informações mais significativas acerca das amostras do conjunto completo, quando comparada com as demais visões.

Ainda de acordo com a tabela abaixo percebe-se que há uma diferença significativa do ARI do primeiro para o segundo e terceiro melhores resultados, figurados pelas visões *“Complete view”* e *“Shape view”*, respectivamente.

Tabela 7 - Melhores resultados segundo a minimização da função objetivo do KCM-K-GH, com dados sem normalização

| Visão | JKCM-K-GH | ARI | Iterações | Tempo | Objetos por agrupamento | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| c1 | c2 | c3 | c4 | c5 | c6 | c7 |
| Complete View | 1555,757001 | 0,396120 | 36 | 00:00:46 | 622 | 402 | 160 | 263 | 297 | 216 | 140 |
| **RGB View** | **835,720425** | **0,514135** | **28** | **00:00:23** | **300** | **237** | **182** | **403** | **297** | **396** | **285** |
| Shape View | 984,727884 | 0,248289 | 92 | 00:00:35 | 373 | 182 | 344 | 321 | 295 | 366 | 219 |

Os demais dados dos resultados selecionados segundo a função objetivo que se referem aos representantes de cada agrupamento, ao vetor de hiperparâmetros e aos objetos contidos em cada partição extraídos da execução do algoritmo com dados sem normalização estão disponíveis no anexo abaixo.



Se novamente quisermos verificar se houve valores de ARI ainda maiores que o melhor resultado acima, podemos selecionar as linhas com maiores valores segundo este critério. Fazendo isso, obtemos a tabela Tabela 8. Ela nos indica que dessa vez o melhor valor de ARI foi o da visão *“Complete view”*, seguido com pouca diferença pela visão *“RGB view”* e posteriormente pela *“Shape view”*. O primeiro colocado apresentou um valor de ARI de aproximadamente 0,56.

Tabela 8 - Melhores resultados pela maximização do Índice de Rand Corrigido (ou ARI), com dados sem normalização

| Visão | JKCM-K-GH | ARI | Iterações | Tempo | Objetos por agrupamento | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| c1 | c2 | c3 | c4 | c5 | c6 | c7 |
| **Complete View** | **1613,615013** | **0,558933** | **32** | **00:00:49** | **451** | **241** | **300** | **216** | **297** | **423** | **172** |
| RGB View | 835,720425 | 0,514135 | 28 | 00:00:23 | 300 | 237 | 182 | 403 | 297 | 396 | 285 |
| Shape View | 1034,254437 | 0,267030 | 58 | 00:00:26 | 96 | 342 | 327 | 332 | 390 | 339 | 274 |

## Conclusão

Através da execução do experimento e análise dos resultados apresentados nas seções acima, pode-se concluir que a visão *“RGB view”* apresentou os melhores resultados e maior grau de similaridade pelo Índice de Rand Corrigido (ou ARI) quando considerado o critério de seleção no enunciado da questão – pela minimização da função objetivo do KCM-K-GH. Isso se deve ao fato de que os atributos contidos nessa visão são capazes de fornecer informações mais significativas acerca das amostras do conjunto de dados, favorecendo assim a uma maior assertividade na distinção e alocação das amostras nos grupos equivalentes.

Além disso, também foi possível concluir que a utilização de um processo de normalização dos dados para contornar a falta de homogeneidade de escala dos parâmetros das amostras exerceu influência nos valores de ARI calculados, mas não foi capaz de alterar o cenário dos resultados extraídos. Quando observamos os resultados segundo a minimização da função objetivo, os maiores graus de similaridade pelo ARI estão entre os dados não normalizados; e quando avaliamos segundo a maximização do ARI, as maiores similaridades se dividem entre os dados normalizados e aqueles não normalizados. Isso nos mostra que a influência exercida não é capaz de transformar significativamente o cenário dos resultados extraídos. Além disso, observa-se pelo experimento que o ranking do resultado das visões com melhor desempenho permanece o mesmo, estando os dados normalizados ou não. Dessa forma, entende-se que o passo de normalização é dispensável no contexto do presente experimento.

# Questão 2

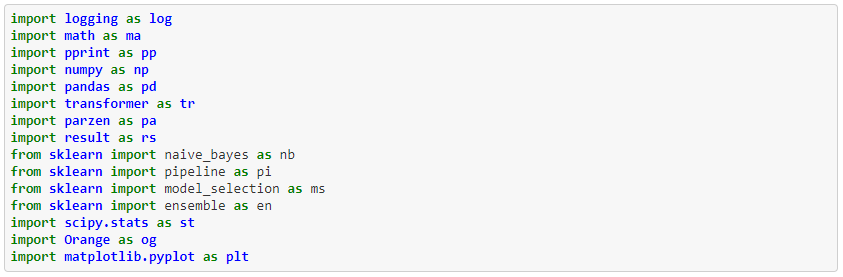
Esta questão aborda um problema de classificação multiclasses para o conjunto de dados apresentado na seção Conjunto de dados. Seu objetivo é comparar diversas combinações de classificadores e visões dos atributos em relação à taxa de acerto.

Os atributos dos exemplos neste conjunto de dados são informações de formas e cores extraídas da região de pixels de imagens aleatórias. Em outras palavras, os classificadores utilizarão informações de textura ao invés de usar diretamente os valores de RGB dos 9 pixels da região.

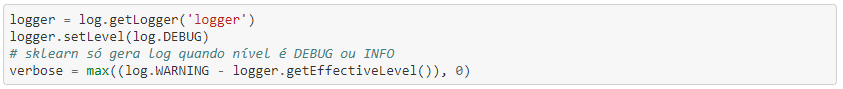
A seção atual foi estruturada da seguinte forma: Preparação dos dados, faz uma análise exploratória dos dados e realiza a seleção de variáveis; Execução do experimento, define a semente do estado aleatório, monta as combinações de classificadores e visões dos atributos e executa o experimento; Avaliação dos resultados individuais, avalia as médias e os intervalos de confiança das taxas de acerto das combinações; e Comparação dos classificadores, realiza os testes de Friedman e Nemenyi para identificar as combinações com resultados significativamente diferentes.

## Contexto de execução

Dependências



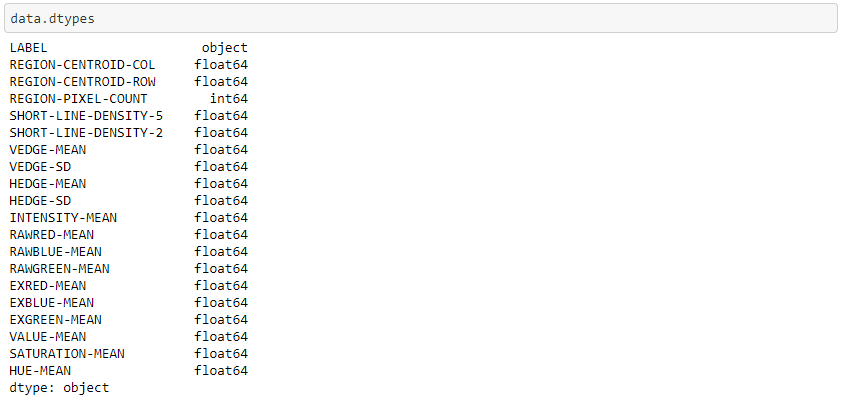
Log



## Preparação dos dados

Conforme apresentado na seção Características dos dados, os atributos das amostras estão representados em escalas diferentes. Isso, porém, não foi um problema para os classificadores que foram utilizados no experimento desta questão. O classificador bayesiano gaussiano (*GaussianNB*) e o classificador bayesiano baseado em janela de Parzen com função de kernel gaussiana (*ParzenWindowB*) são robustos às diferenças de escala entre as variáveis.

Antes de passar os dados para os classificadores, é preciso confirmar que os tipos das variáveis são numéricos. A tabela abaixo confirma isso, pois os tipos das variáveis se limitam a *float64* e *int64*. Além disso, a semântica das variáveis nos garante que não há dados categóricos. Conforme documentação do conjunto de dados, todas as variáveis são números ordinais. Nesse caso, as variáveis já estão em formatos compatíveis com os classificadores que serão usados mais à frente, sem a necessidade de pré-processamento.



A partir da análise realizada na seção Características dos dados, que conclui que o atributo *REGION-PIXEL-COUNT* refere-se a um valor constante, assume-se que ele é irrelevante para os classificadores aqui utilizados. Tal variável, inclusive, criou complicações na execução do experimento, como divisão por zero na validação cruzada que define o tamanho da janela de Parzen. Por isso, ela foi excluída do conjunto de dados com o comando abaixo.

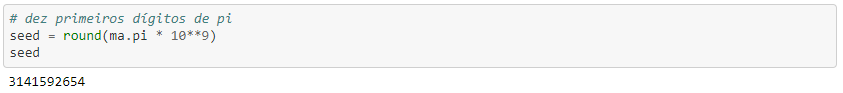


## Execução do experimento

O experimento consiste em executar a validação cruzada estratificada "*30 times 10-fold*" para cada combinação de classificador e visão das variáveis e para o classificador combinado. Para isso, vários passos de preparação foram realizados antes da validação cruzada propriamente dita dos classificadores. Todos os passos serão descritos a seguir.

### Preparação

1. Definir uma semente para estado aleatório do experimento.

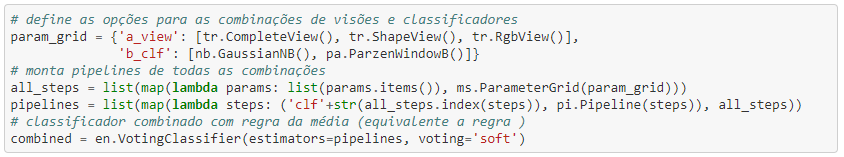


1. Separar o rótulo das variáveis.

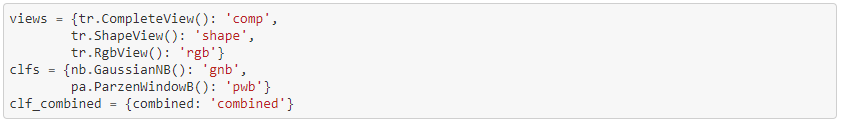


1. Criar o classificador combinado. Conforme enunciado, esse classificador consiste na combinação de seis combinações de visões e classificadores com a regra da soma:
   1. Complete view e *GaussianNB*;
   2. Shape view e *GaussianNB*;
   3. RGB view e *GaussianNB*;
   4. Complete view e *ParzenWindowB*;
   5. Shape view e *ParzenWindowB*;
   6. RGB view e *ParzenWindowB*.

Mais detalhes sobre a regra de combinação podem ser obtidos na documentação de *VotingClassifier* [3] e no artigo [4].



1. Definir dicionários dos nomes das visões e dos classificadores. Esses nomes serão vistos mais à frente, pois foram utilizados na formatação dos resultados.

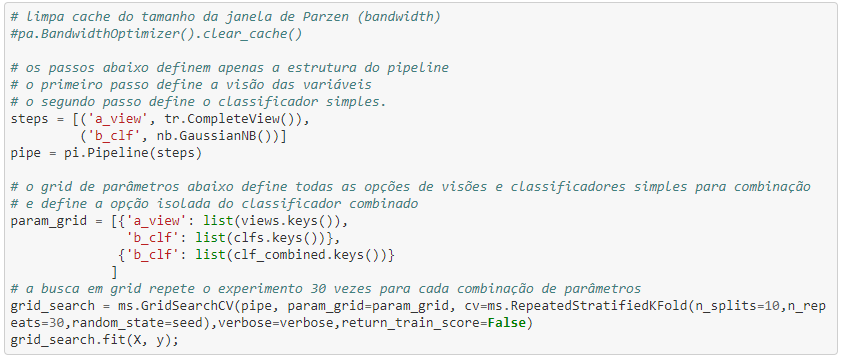


1. Unir dicionários de nomes em um único dicionário.



### Validação cruzada

Executa a validação cruzada estratificada com cada combinação de visão e classificador simples e com o classificador combinado. Conforme enunciado da questão, as visões das variáveis são *Complete*, *Shape* e *RGB* e os classificadores simples são bayesiano gaussiano e bayesiano baseado em janela de Parzen. Devido à diferença na estrutura do pipeline, o classificador combinado foi criado na seção Preparação e incluído de forma isolada no experimento para evitar novas combinações com as visões.





#### Classificadores utilizados

O classificador bayesiano gaussiano utilizado assume que as variáveis seguem uma distribuição normal multivariada e são independentes. Além disso, desconsidera as probabilidades *a priori*, já que o conjunto de dados possui uma distribuição uniforme entre as classes. Então, na etapa de treinamento, uma função de distribuição de probabilidade condicional é estimada para cada classe com base na forma paramétrica de uma função de distribuição normal multivariada. Na etapa de classificação, para cada classe, uma probabilidade *a posteriori* é calculada com base na função de densidade de probabilidade condicional da respectiva classe. Por fim, o exemplo é dito como da classe com maior probabilidade a posteriori. Mais detalhes sobre o classificador bayesiano gaussiano podem ser obtidos na documentação de GaussianNB [3].

O classificador bayesiano baseado em janela de Parzen utilizado não assume a forma de distribuição das variáveis e não assume que elas são independentes. Além disso, a função de janela foi substituída por uma função de kernel gaussiana. Assim como o classificador bayesiano gaussiano, esse classificador desconsidera as probabilidades *a priori*. Então, na etapa de treinamento, os dados são apenas carregados. Na etapa de classificação, esse classificador utiliza a regra do produto para combinar as funções de kernel gaussianas das variáveis e obter o resultado da função de densidade de probabilidade condicional de cada classe. Com esses valores, ele calcula as probabilidades *a posteriori* de cada classe. Por fim, o exemplo é dito como da classe com maior probabilidade *a posteriori*. Mais detalhes sobre o classificador bayesiano baseado em janela de Parzen com função de kernel gaussiana podem ser obtidos no código fonte de ParzenWindowB [3] e na documentação de KDEMultivariate [5].

#### Tamanho da janela de Parzen

Vale salientar que o classificador *ParzenWindowB* faz uma validação cruzada interna para encontrar o melhor tamanho de janela para o conjunto de dados de treinamento de cada rodada de validação cruzada do experimento. Os valores avaliados na validação cruzada interna são 3, 4 e 5.

Mais detalhes sobre a definição do tamanho da janela de Parzen podem ser obtidos no código fonte de *BandwidthOptimizer*.

#### Tempo de execução

Em um computador com Processador Intel® Core™ i7-7500U CPU @ 2.70GHz com 2 núcleos, 2 GB de memória RAM, a execução da validação cruzada levou 260,2 min (por volta de 4 horas). Foi verificado que a validação cruzada para definição do tamanho da janela de Parzen introduz bastante custo computacional e ela ocorre de forma idêntica nos classificadores simples e no combinado. Então, para evitar o recálculo desses valores dentro de uma mesma validação cruzada ou em reexecuções completas, foi introduzido um mecanismo de cache em *BandwidthOptimizer*. Essa otimização baixou o tempo de reexecução para 11,0 minutos. Porém, o tempo de execução de *ParzenWindowB* ainda permaneceu relativamente elevado. Enquanto que, utilizando todas as variáveis, *GaussianNB* executa uma rodada de validação cruzada em um tempo na ordem de 10-3 segundos, *ParzenWindowB* fica na ordem de 101 segundos (sem cache) e 10-1 segundos (com cache). Esse resultado é um indício de que *ParzenWindowB*, sem otimizações adicionais, pode não ser uma boa opção para um contexto que exige alta performance computacional.

#### Reprodução do experimento

Todo código fonte utilizado se encontra disponível em um repositório do GitHub[[3]](#footnote-3). Além disso, um ambiente de execução similar ao do experimento se encontra disponível no Binder[[4]](#footnote-4).

## Avaliação dos resultados individuais

Esta seção tem como objetivo formatar os dados gerados pelo experimento, obter uma estimativa pontual e um intervalo de confiança para a taxa de acerto de cada classificador.

### Formatação dos dados

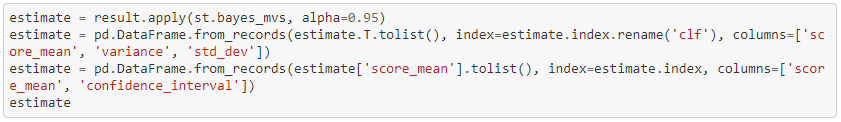
Os dados gerados pelo experimento que são relevantes para este trabalho consistem em uma matriz no formato "rodada de validação cruzada" x "classificador" (300 x 7), com um valor de "taxa de acerto" em cada célula. Porém, conforme orientado, é preciso calcular a média da taxa de acerto para cada repetição da validação cruzada antes de seguir em frente. Dessa forma, a nova matriz, exibida abaixo, ficou no formato "repetição" x "classificador" (30 x 7), com uma média da "taxa de acerto" em cada célula. Cada repetição corresponde a 10 rodadas de validação cruzada, pois a divisão do conjunto de dados é 10-fold. Vide a seção Preparação para obter o dicionário de nomes das abreviações referenciadas na matriz abaixo.



| ***repetition*** | ***comp\_gnb*** | ***comp\_pwb*** | ***shape\_gnb*** | ***shape\_pwb*** | ***rgb\_gnb*** | ***rgb\_pwb*** | ***combined*** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 0,798095 | 0,950952 | 0,478571 | 0,586190 | 0,794286 | 0,869048 | 0,937619 |
| **1** | 0,795714 | 0,951429 | 0,471905 | 0,583810 | 0,795238 | 0,867143 | 0,933810 |
| **2** | 0,796667 | 0,954286 | 0,479048 | 0,587143 | 0,790476 | 0,873333 | 0,936667 |
| **3** | 0,799048 | 0,954762 | 0,473333 | 0,581905 | 0,794762 | 0,871905 | 0,938571 |
| **4** | 0,794762 | 0,950000 | 0,479048 | 0,586667 | 0,800000 | 0,869524 | 0,938095 |
| **5** | 0,799048 | 0,953333 | 0,477143 | 0,588095 | 0,795238 | 0,870952 | 0,933810 |
| **6** | 0,795714 | 0,949048 | 0,477143 | 0,582857 | 0,792381 | 0,866667 | 0,936190 |
| **7** | 0,796190 | 0,951429 | 0,474286 | 0,589524 | 0,795238 | 0,868571 | 0,930000 |
| **8** | 0,796667 | 0,953333 | 0,473810 | 0,585238 | 0,798095 | 0,867619 | 0,935714 |
| **9** | 0,799048 | 0,954286 | 0,475714 | 0,587619 | 0,796667 | 0,870952 | 0,936667 |
| **10** | 0,796667 | 0,952857 | 0,479048 | 0,584762 | 0,794286 | 0,870000 | 0,934286 |
| **11** | 0,797143 | 0,953333 | 0,478571 | 0,581905 | 0,797143 | 0,865714 | 0,936667 |
| **12** | 0,796667 | 0,951905 | 0,472857 | 0,589048 | 0,795238 | 0,869048 | 0,934762 |
| **13** | 0,796190 | 0,950952 | 0,473333 | 0,589048 | 0,796190 | 0,865714 | 0,933810 |
| **14** | 0,796667 | 0,951429 | 0,480000 | 0,583810 | 0,796667 | 0,868095 | 0,931905 |
| **15** | 0,797143 | 0,948095 | 0,477143 | 0,592857 | 0,793810 | 0,870476 | 0,936667 |
| **16** | 0,795238 | 0,952857 | 0,474286 | 0,590476 | 0,796190 | 0,872381 | 0,935238 |
| **17** | 0,799524 | 0,954286 | 0,476190 | 0,584762 | 0,796667 | 0,868095 | 0,937619 |
| **18** | 0,795238 | 0,955238 | 0,479048 | 0,583333 | 0,791429 | 0,870952 | 0,935238 |
| **19** | 0,796667 | 0,954286 | 0,475238 | 0,584286 | 0,797143 | 0,869048 | 0,937143 |
| **20** | 0,796667 | 0,948095 | 0,476667 | 0,587619 | 0,792857 | 0,868095 | 0,934762 |
| **21** | 0,796667 | 0,953810 | 0,476667 | 0,582381 | 0,796190 | 0,871905 | 0,939524 |
| **22** | 0,798095 | 0,951429 | 0,476667 | 0,582381 | 0,794762 | 0,870952 | 0,935714 |
| **23** | 0,797143 | 0,953333 | 0,475714 | 0,596190 | 0,799524 | 0,869048 | 0,936190 |
| **24** | 0,794762 | 0,950000 | 0,474286 | 0,589524 | 0,792381 | 0,871905 | 0,933810 |
| **25** | 0,795714 | 0,954286 | 0,474286 | 0,582381 | 0,792857 | 0,870000 | 0,934286 |
| **26** | 0,796190 | 0,950000 | 0,474762 | 0,587143 | 0,795714 | 0,869524 | 0,932857 |
| **27** | 0,798095 | 0,953810 | 0,477143 | 0,579524 | 0,796667 | 0,872381 | 0,937619 |
| **28** | 0,797619 | 0,950000 | 0,477143 | 0,580952 | 0,796667 | 0,868095 | 0,934286 |
| **29** | 0,796667 | 0,951905 | 0,474762 | 0,592857 | 0,794762 | 0,868571 | 0,937619 |

### Estimativa pontual e intervalo de confiança

A tabela abaixo contém os resultados individuais dos classificadores. "*score\_mean*" é a estimativa pontual da média da taxa de acerto e "*confidence\_interval*" é o intervalo no qual há 95% confiança de que o "*score\_mean*" esteja presente. Como a amostra utilizada para calcular "*score\_mean*" é de tamanho 30, é possível assumir que a distribuição de "*score\_mean*" se aproxima de uma distribuição normal, conforme afirma o Teorema do Limite Central. Com base nesse tipo de distribuição, foi calculado o intervalo de confiança da estimativa pontual. A combinação "*comp\_pwb*" (*Complete view* e *ParzenWindowB*) foi o classificador que obteve o melhor resultado.



|  |  |  |
| --- | --- | --- |
| **clf** | **score\_mean** | **confidence\_interval** |
| **comp\_gnb** | 0,796857 | (0,796384152991, 0,797330132723) |
| **comp\_pwb** | 0,952159 | (0,95140761877, 0,952909841548) |
| **shape\_gnb** | 0,476127 | (0,475321249765, 0,476932718489) |
| **shape\_pwb** | 0,586143 | (0,584677061536, 0,58760865275) |
| **rgb\_gnb** | 0,795317 | (0,794500505578, 0,796134415057) |
| **rgb\_pwb** | 0,869524 | (0,868781066783, 0,870266552264) |
| **combined** | 0,935571 | (0,934791244694, 0,936351612449) |
| **comp\_gnb** | 0,796857 | (0,796384152991, 0,797330132723) |

## Comparação dos classificadores

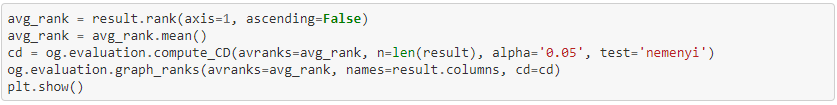
Os classificadores foram comparados por meio de um teste estatístico para confirmar que a diferença entre eles é significante. O teste utilizado foi o teste de Friedman [6]. A hipótese nula foi: as médias das taxas de acerto dos classificadores no conjunto de dados utilizado possuem a mesma distribuição, ou seja, as diferenças entre os classificadores para o conjunto de dados utilizado são insignificantes. A hipótese alternativa foi: as diferenças entre os classificadores para o conjunto de dados utilizado são estatisticamente significantes.

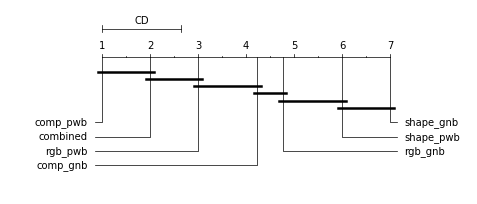
Mais detalhes sobre comparação de classificadores podem ser obtidos no artigo [6]. Vale salientar que este artigo utilizou vários conjuntos de dados, enquanto que este trabalho utilizou apenas um. Dessa forma, a comparação de classificadores realizada neste trabalho está limitada ao conjunto de dados apresentado na seção Conjunto de dados, não podendo ser generalizada para outros cenários. Se a hipótese nula deste trabalho fosse generalizada, o risco de cometer o erro Tipo 1 seria elevado, pois as variâncias das médias das taxas de acerto dependem unicamente da amostragem de dados de cada repetição da validação cruzada. Nesse caso, as chances da performance do melhor classificador da primeira repetição de validação cruzada se manter nas demais repetições é muito alta.

O teste de Friedman foi realizado e o *p-value* é exibido abaixo. O resultado do teste foi a rejeição da hipótese nula com *p-value* < 0.05.



Dado que a hipótese nula foi rejeitada, o passo seguinte foi aplicar o teste de Nemenyi para identificar quais pares de classificadores possuem diferenças significativas e quais não possuem. O resultado do teste é exibido abaixo de forma gráfica, com o *CD diagram* [6] (diagrama de diferença crítica), com *p-value* < 0.05. A partir desse diagrama, é possível observar que, por exemplo, "*comp\_pwb*" e "*combined*" não possuem diferenças significativas, pois estão conectados por uma barra; enquanto que "*comp\_pwb*" e "*rgb\_pwb*" possuem diferenças significativas, pois não estão conectados por uma barra. A ordenação dos classificadores é do melhor resultado (esquerda) para o pior (direita). Por fim, a barra acima do gráfico com a sigla CD representa a diferença crítica para *p-value* < 0.05.





## Conclusão

A partir dos resultados individuais e das comparações dos classificadores, é possível concluir que "*comp\_pwb*" possui a melhor estimativa pontual de média de taxa de acerto, mas a comparação dos classificadores não confirma diferença significativa entre esse classificador e "*combined*". Por outro lado, como "*combined*" é a junção de seis combinações de visões e classificadores e "*comp\_pwb*" é um classificador simples, o segundo classificador torna-se a melhor opção quando o custo computacional é um dos critérios de escolha. Por fim, vale reforçar que essas conclusões se limitam ao conjunto de dados apresentado na seção Conjunto de dados.

# Referências

|  |  |
| --- | --- |
| [1] | University of California, Irvine, “Image Segmentation Data Set,” Machine Learning Repository, 1990. [Online]. Available: http://archive.ics.uci.edu/ml/machine-learning-databases/image/. [Acesso em 09 Junho 2018]. |
| [2] | Pandas, “pandas.DataFrame,” [Online]. Available: https://pandas.pydata.org/. [Acesso em 09 Junho 2018]. |
| [3] | F. d. A. T. Carvalho, E. C. Simões, L. V. C. Santana e M. R. P. Ferreira, “Gaussian kernel c-means hard clustering algorithms with automated computation of the width hyper-parameters,” *Pattern Recognition,* nº 79, pp. 370-368, Julho 2018. |
| [4] | F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot e E. Duchesnay, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research,* nº 12, pp. 2825-2830, 2011. |
| [5] | J. Kittler, M. Hatef, R. P. Duin e J. Matas, “On Combining Classifiers,” *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE,* vol. 20, nº 3, pp. 226-239, 1998. |
| [6] | Stats Model, "Non Parametric Models: KDEMultivariate," [Online]. Available: http://www.statsmodels.org/dev/generated/statsmodels.nonparametric.kernel\_density.KDEMultivariate.html. [Accessed 10 Junho 2018]. |
| [7] | J. Demsar, “Statistical comparisons of classifiers over multiple data sets,” *Journal of Machine Learning Research,* nº 7, pp. 1-30, 2006. |

1. GitHub – repositório do projeto da disciplina IN1102 (Aprendizagem de Máquina). Disponível em <https://github.com/amorim-cleison/cin_am> [↑](#footnote-ref-1)
2. ARI – sigla em inglês para a tradução do termo Índice de Rand Ajustado (*Adjusted Rand Index*). [↑](#footnote-ref-2)
3. GitHub – repositório do projeto da disciplina IN1102 (Aprendizagem de Máquina). Disponível em <https://github.com/amorim-cleison/cin_am> [↑](#footnote-ref-3)
4. Binder – ambiente contendo o código fonte para a questão 2. Disponível em <https://hub.mybinder.org/user/amorim-cleison-cin_am-jjv18om5/notebooks/questao_2/questao2.ipynb> [↑](#footnote-ref-4)