Universidade Federal de Minas Gerais
Engenharia Elétrica
Sistemas Nebulosos

Trabalho Prático 1
Fuzzy K-Means

Aluno: Thiago Lages Rocha | 2015.123.320

Professor: Cristiano Leite de Castro

Belo Horizonte 2019.2

Sumário

List of Figures

1 Introdução

A ideia do algoritmo K-Means (também chamado de K-Médias) é fornecer uma classificação de informações de acordo com os próprios dados. Esta classificação, como será vista a seguir, é baseada em análise e comparações entre os valores numéricos dos dados. Desta maneira, o algoritmo automaticamente vai fornecer uma classificação automática sem a necessidade de nenhuma supervisão humana, ou seja, sem nenhuma pré-classificação existente. Por causa desta característica, o K-Means é considerado como um algoritmo de mineração de dados não supervisionado.

Para entender como o algoritmo funciona, vamos imaginar que temos uma tabela com linhas e colunas que contêm os dados a serem classificados. Nesta tabela, cada coluna é chamada de dimensão e cada linha contém informações para cada dimensão, que também são chamadas de ocorrências ou pontos. Geralmente, trabalha-se com dados contínuos neste algoritmo, mas nada impede que dados discretos sejam utilizados, deste que eles sejam mapeados para valores numéricos correspondentes.

Como foi dito, o algoritmo vai analisar todos os dados desta tabela e criar classificações. Isto é, o algoritmo vai indicar uma classe (cluster) e vai dizer quais linhas pertencem a esta classe. O usuário deve fornecer ao algoritmo a quantidade de classes que ele deseja. Este número de classes que deve ser passada para o algoritmo é chamado de k e é daí que vem a primeira letra do algoritmo: K-Means.

Para gerar as classes e classificar as ocorrências, o algoritmo faz uma comparação entre cada valor de cada linha por meio da distância. Geralmente utiliza-se a distância euclidiana para calcular o quão 'longe' uma ocorrência está da outra. A maneira de calcular esta distância vai depender da quantidade de atributos da tabela fornecida. Após o cálculo das distâncias o algoritmo calcula centroides para cada uma das classes. Conforme o algoritmo vai iterando, o valor de cada centroide é refinado pela média dos valores de cada atributo de cada ocorrência que pertence a este centroide. Com isso, o algoritmo gera k centroides e coloca as ocorrências da tabela de acordo com sua distância dos centroides.

Para simplificar a explicação de como o algoritmo funciona vou apresentar o algoritmo K-Means em cinco passos:

1. Fornecer valores para os centroides: Neste passo os k centroides devem receber valores iniciais. No início do algoritmo geralmente escolhe-se os k primeiros pontos da tabela. Também é importante colocar todos os pontos em um centroide qualquer para que o algoritmo possa iniciar seu processamento.

2. Gerar uma matriz de distância entre cada ponto e os centroides: Neste passo, a distância entre cada ponto e os centroides é calculada. A parte mais 'pesada' de cálculos ocorre neste passo pois se temos N pontos e k centroides teremos que calcular N x k distâncias neste passo.

3. Colocar cada ponto nas classes de acordo com a sua distância do centroide da classe: aqui, os pontos são classificados de acordo com sua distância dos centroides de cada classe. A classificação funciona assim: o centroide que está mais perto deste ponto vai 'incorporá-lo', ou seja, o ponto vai pertencer à classe representada pelo centroide que está mais perto do ponto. É importante dizer que o algoritmo termina se nenhum ponto 'mudar' de classe, ou seja, se nenhum ponto for 'incorporado' a uma classe diferente da que ele estava antes deste passo.

4. Calcular os novos centroides para cada classe: neste momento, os valores das coordenadas dos centroides são refinados. Para cada classe que possui mais de um ponto o novo valor dos centroides é calculado fazendo-se a média de cada atributo de todos os pontos que pertencem a esta classe.

5. Repetir até a convergência: o algoritmo volta para o PASSO 02 repetindo iterativamente o refinamento do cálculo das coordenadas dos centroides. Notem que desta maneira teremos uma classificação que coloca cada ponto em apenas uma classe. Desta maneira dizemos que este algoritmo faz uma classificação hard (hard clustering) uma vez que cada ponto só pode ser classificado em uma classe. Outros algoritmos trabalham com o conceito de classificação soft onde existe uma métrica que diz o quão 'dentro' de cada classe o ponto está.

Já o algoritmo de K-Médias em lógica Fuzzy (Nebulosa) parte do pressuposto que cada amostra não pertence apenas a um único grupo (cluster), não apresentando nenhuma relação com os outros grupos. É assumido que há um grau de pertencimento a cada um dos clusters, que vai de 0 a 1. Dessa forma, é possível montar problemas mais próximos da linguagem natural, em definições como "alto", "muito quente", "bem próximo", entre outros. Nestes casos, não há um limite bem definido entre as amostras de dados e os grupo especificados, devendo portanto, ser considerados que há um gra de pertencimento entre a amostra e o grupo.

2 Exercícios

2.1 Exercício 1

A primeira parte do trabalho é mostrada a seguir:

Fuzzy C-Means: Implemente o algoritmo de agrupamento Fuzzy C-Means (FCM). Caso seja conveniente, modifique o cóodigo do algoritmo K-Means fornecido no Moodle;

Validação do FCM: Valide o algoritmo FCM com a base de dados 'FCM-Dataset.mat'. Para a validação, plote os centros dos clusters encontrados pelo algoritmo FCM sobre a base de dados fornecida. Compare os resultados obtidos pelo FCM com aqueles obtidos pelo algoritmo K-Means. A comparação deve ser em termos de: (i) número médio de iterações até a convergência(ii) número de vezes que o algoritmo encontra valores adequados para os centros dos clusters;

Algoritmo	Número médio de execuções	Vezes que encontra valores adequados (em 50)
K-Means Tradicional	6.6	24
K-Means Fuzzy	11.4	35

Table 1: Comparação entre o K-Means tradicional e K-Means Fuzzy

Para coleta desses dados, execute os algoritmos N vezes (onde N $>$ 30) com os mesmos valores de inicialização.

Como se pode observar nos resultados da tabela 1, e analisando os gráficos 1 e 2, pode-se perceber que o algoritmo de K-Means Fuzzy geralmente demora mais tempo para convergir do que o algoritmo tradicional. Além disso, a avaliação da sua função de custo é similar ao algoritmo tradicional, porém apresenta alguns valores mais altos (os picos no gráfico 1), indicando que não encontra clusters adequados em algumas ocasiões.

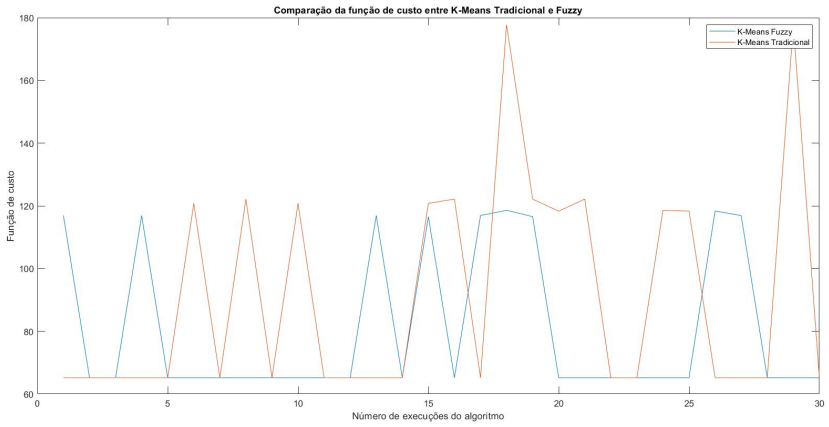

Figure 1: Comparação entre a função de custo entre os dois algoritmos.

Na implementação do algortimo Fuzzy K-Means, foi-se inicializada a matriz U com valores aleatórios entre 0 e 1, de maneira que cada amostra fica em cada linha, e possui K colunas, sendo a soma de cada linha igual a 1. Utilizou-se a equação 3 para calcular a nova posição de cada cluster:

E para a atualização da matriz U, utilizou-se a expressão contida na figura 4:

2.2 Exercício 2

A segunda parte é apresentada a seguir: **Segmentação de Imagens por Região:** Use o algoritmo FCM para segmentar por região as imagens RGB fornecidas no diretório ImagensTeste do Moodle. Para cada imagem, escolha o número de clusters de forma empírica, com base na observação das cores das

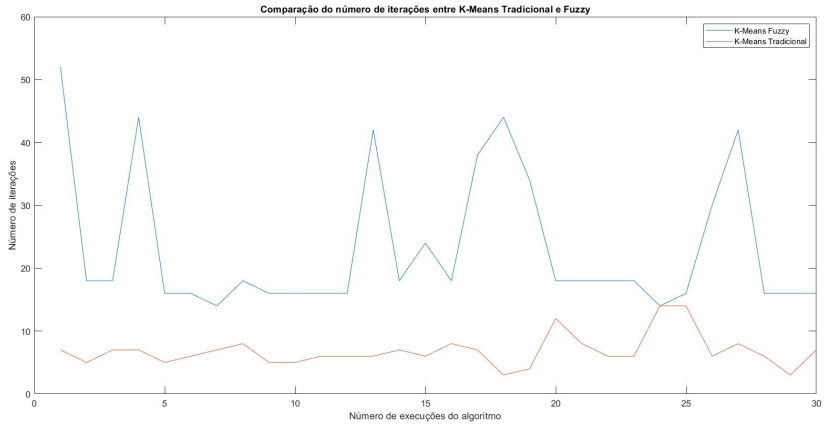

Figure 2: Comparação entre o número de iterações entre os dois algoritmos.

$$\vec{c}_k = \frac{\sum_{i=1}^{n} u_{i,k}^{m} \vec{x}_i}{\sum_{i=1}^{n} u_{i,k}^{m}}$$

Figure 3: Fórmula utilizada para calcular cada centroide.

$$u_{i,k} = \frac{1}{\left(\frac{(\vec{x}_i - \vec{c}_k)^2}{\sum_{t=1}^{K} (\vec{x}_i - \vec{c}_t)^2} \right)^{2/(m-1)}}$$

Figure 4: Fórmula utilizada para calcular a matriz U a cada iteração

diferentes regiões. Após obter a matriz de partição U, resultado da aplicação do FCM em cada imagem, use esta matriz para colorir cada região (cluster) com a tonalidade do pixel que corresponde ao centro da região. Os pixels que apresentarem maior grau de compatibilidade (pertinência) a uma dada região devem ser coloridos com a tonalidade do pixel central daquela região.

A seguir são apresentados os resultados obtidos a partir da segmentação de imagens usando o algoritmo de Fuzzy K-Means. Os números de clusters escolhidos para cada uma das 11 imagens apresentadas a seguir foram 2,2,5,6,6,3,3,6,5,2,3, respectivamente. Nas figuras a seguir, a imagem da esquerda é a imagem original, a do meio é o resultado da segmentação pelo algoritmo e o gráfico tridimensional à esquerda é a representação de cada pixel da imagem original no espaço RGB (Red, Green, Blue), cuja coloração corresponde à sua localização

no espaço. A localização dos centróides podem ser vistos como círculos verdes maiores do que os pontos no espaço.

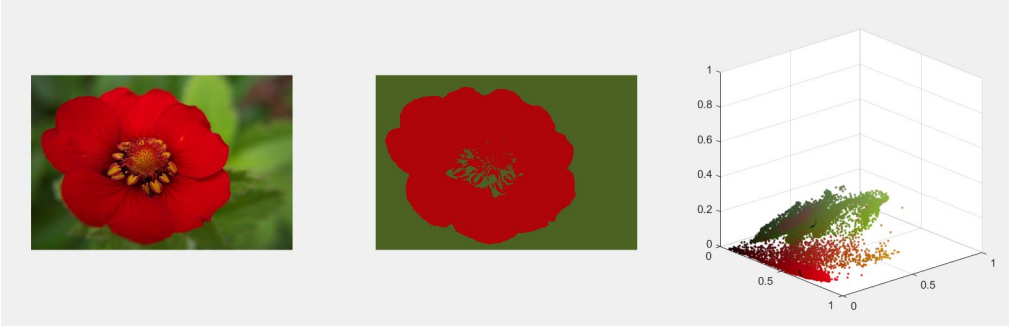

Figure 5: Resultado de segmentação na primeira imagem.

Em alguns casos, como no da figura 6, escolhendo-se um número aparentemente adequado de clusters (dois), o algoritmo não convergiu como esperado, ficando apenas em um curto espectro de cores. Desta forma, o resultado parece conter apenas um espectro de cor verde, de maneira que não obteve o espectro cinza esperado. Dessa forma, aumentando-se o número de clusters para três, o resultado observado na figura 7 é melhor, onde o animal foi identificado de maneira mais consistente.

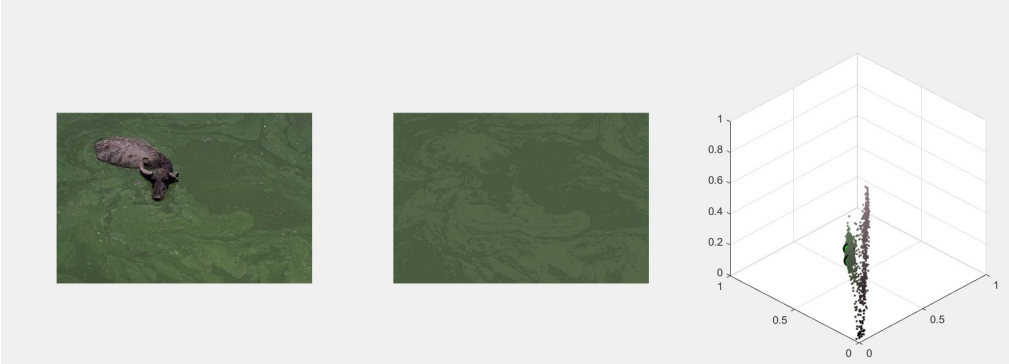

Figure 6: Resultado de segmentação na segunda imagem, com número inadequado de clusters.

Na figura 10, mesmo escolhendo-se vários clusters, o algoritmo não foi capaz de pegar o espectro de azul pertencente a uma parte da roupa na manga esquerda, e no pescoço da modelo. O algoritmo ficou preso em locais cujo espectro era próximo do preto e amarelado.

Na figura 11, mesmo escolhendo-se três clusters, o algoritmo não foi capaz de capturar a imagem da lua, ficando preso em um espectro de azul, como pode

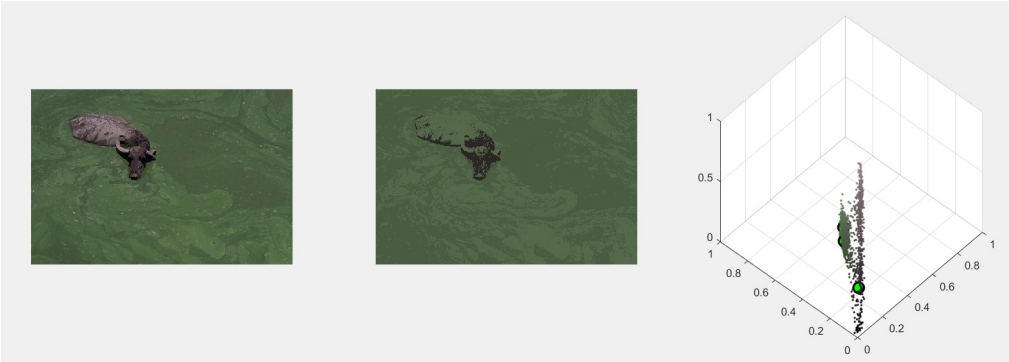

Figure 7: Resultado de segmentação na segunda imagem, com número mais adequado de clusters

Figure 8: Resultado de segmentação na terceira imagem.

Figure 9: Resultado de segmentação na quarta imagem.

Figure 10: Resultado de segmentação na quinta imagem.

ser observado no gráfico tridimensional.

Figure 11: Resultado de segmentação na sexta imagem.

A imagem de número oito, figura 13, inicialmente havia sido assimilada 4 centroides, o que não se mostrou ser suficiente para fazer uma coloração adequada à imagem, visto que todos eles estavam se localizando perto do espectro azul da imagem, que se mostrou ser muito amplo. Desta forma, aumentou-se o número de centroides para 6, de maneira que os espectros laranja e branco foram também encontrados, dando uma coloração mais adequada à imagem, na figura 14.

O resultado da imagem 12 surpreende pois, mesmo com um número relativamente baixo de clusters (seis) para a aparente complexidade da imagem, os resultados obtidos mostram que o algortimo foi capaz de cobrir muito bem o espectro de cores apresentado na imagem, e sua segmentação se mostrou muito boa, apesar de possuir regiões muito pequenas que mudam de coloração muito bruscamente.

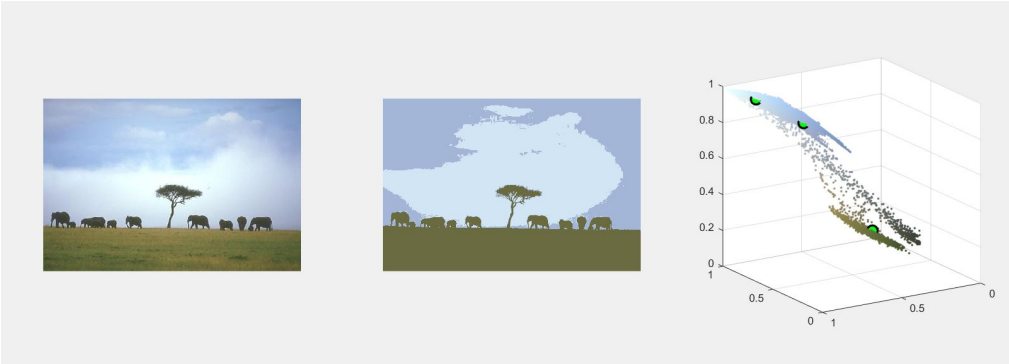

Figure 12: Resultado de segmentação na sétima imagem.

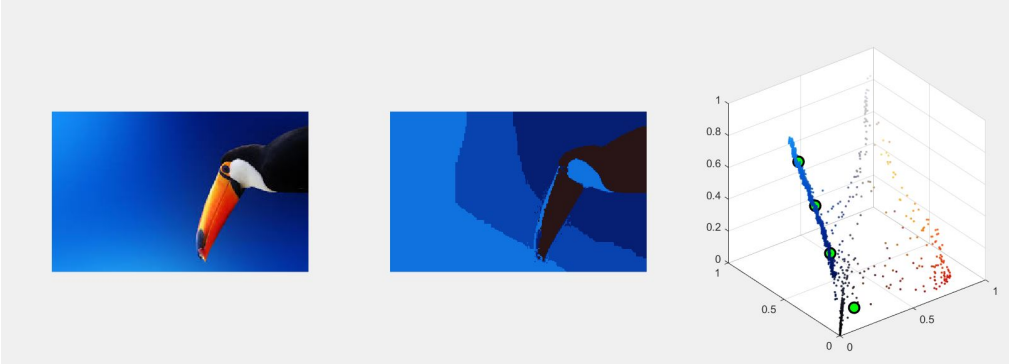

Figure 13: Resultado de segmentação na oitava imagem, com número de clusters indevido.

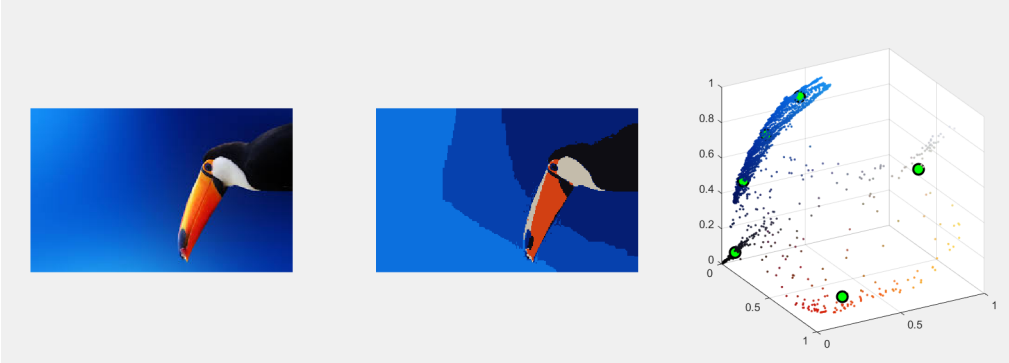

Figure 14: Resultado de segmentação na oitava imagem, com número mais apropriado de clusters.

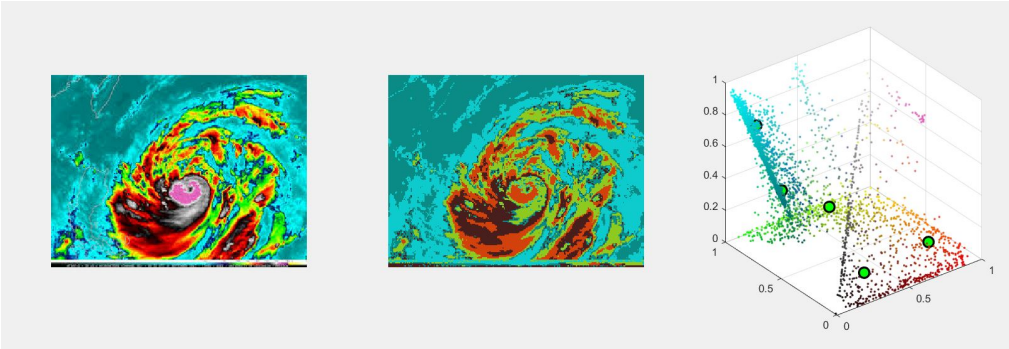

Figure 15: Resultado de segmentação na sétima imagem.

Nas figuras 16 e 17 a seguir, por terem regiões mais bem definidas, foi relativamente fácil o algoritmo encontrar soluções para elas, convergindo de maneira correta às regiões de interesse.

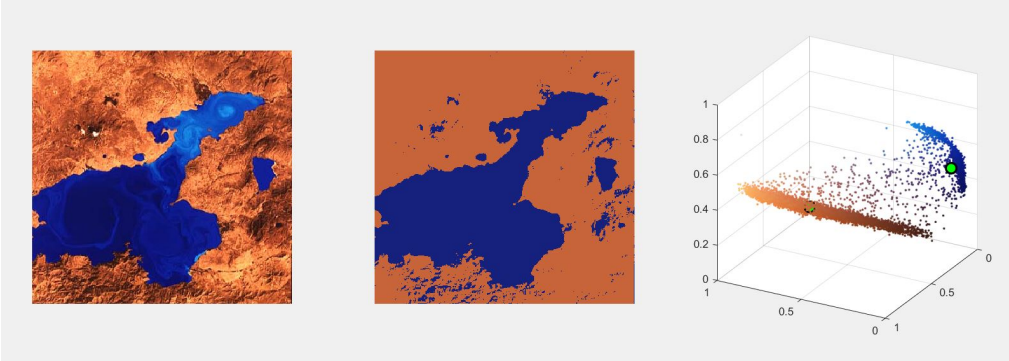

Figure 16: Resultado de segmentação na sétima imagem.

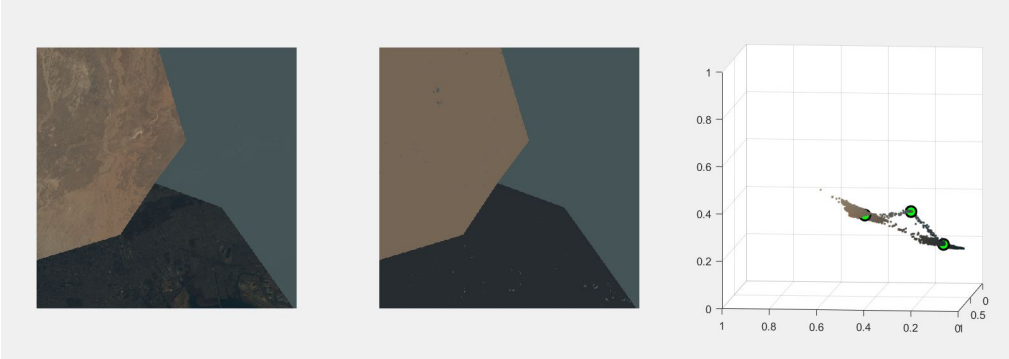

Figure 17: Resultado de segmentação na sétima imagem.