CENTRO UNIVERSITÁRIO FEI

VICTOR BIAZON

RA: 119.115-4

RELATÓRIO III – TÓPICOS ESPECIAIS DE APRENDIZAGEM
K-MEANS

SÃO BERNARDO DO CAMPO

2019

Sumário:

1. Objetivo 3

2. Teoria 3

Análise de componente principal........ Erro! Indicador não definido.

3. Implementação 4

4. Resultados 6

5. Conclusão 11

6. Referências bibliográficas........ 12

1. Objetivo

Implementar o algoritmo clustering k-means e testar em diferentes datasets.

2. Teoria

K – Means

Segundo MacQueen(1967) o algoritmo k-means consiste na classificação de um número n de observações com i varíaveis em k classes distintas, tratando-se de um algoritmo de clustering (agrupamento). O algoritmo tem diversas variações para melhorar seu desempenho e confiabilidade do agrupamento, no entanto, sua forma mais simples consiste atribuir um número k de pontos aleatoriamente no espaço entre os valores mínimos e máximos de cada variável e em seguida calcular a distância de cada ponto observado para os pontos "k-means" que são os centroides dos grupos. Para cada ponto é atribuída a classe daquele centroide que está mais perto dele. Após a atribuição de todos os pontos, se calcula novamente a posição dos centroides sendo a nova localização a média dos valores dos pontos atribuídos a sua classe. Realocando o centroide para o centro médio dos pontos a ele atribuídos. Em seguida é calculada novamente a distância de cada ponto observado até a nova posição dos centroides, estes sendo novamente classificados como pertencentes ao grupo do centroide mais próximo e recalculada a posição do centroide até que sua posição não se modifique de uma iteração a outra ou não haja mudança na atribuição de pontos a grupos diferentes, concluindo assim o algoritmo.

Segundo Lloyd(1982) existem previstas na literatura diversas formas de se calcular a distância dos pontos até o centróide. Entre elas: Euclidiana, Manhattan, quadrática, etc. Também para avaliar o grau de qualidade da separação dos grupos pode-se utilizar conceitos de LDA para verificar o espalhamento e distância entre clusters.

3. Implementação

A implementação partiu do seguinte fluxo:

Para implementação foram criadas as seguintes funções:

```
#calcula a distância Euclidiana de um ponto a outro
def DistEuclid(pontoA, pontoB, eixos):
    accum = 0
    for i in range(0, eixos):
        accum += (pontoA[i]-pontoB[i])**2
    return math.sqrt(accum)



def K_means(data, K, dim, dimx, dimy):

    #separando variáveis independentes e dependentes
    X = np.asanyarray(data[:,:]) #separa as variáveis independentes no vetor X

    #Atribuindo K's aleatorios
    Km = np.zeros((K,len(X[0])),float)

    #Atribuindo os centroides a pontos aleatórios nas observações
    Xsamples = np.copy(X)
    for i in range(0, len(Km)):
        Index = random.randint(0,len(Xsamples)-1)
        Km[i] = Xsamples[Index]
        np.delete(Xsamples, Index, 0)
```

```
    del Xsamples


    #   plot de dados sem divisao de classes e ponto Kmean aleatórios
    plt.figure()
    plt.scatter(X[:,dimx], X[:,dimy], color = 'red', cmap = 'rainbow')
    plt.scatter(Km[:,dimx], Km[:,dimy], color = 'black', marker = 'X')
    plt.title('K-means - Data')
    plt.xlabel('X1')
    plt.ylabel('X2')
    plt.show()


    execute_kmean = True


    #executa enquanto nao se estabiliza a posição dos centroides
    while(execute_kmean):
        Km2 = np.copy(Km)
        # atribuição de classes de acordo com a proximidade do ponto
de outros pontos Kmean
        Y_class = np.zeros_like(X[:,0], float)
        for i in range(0, len(X)):
            minD = math.inf
            for j in range(0, len(Km)):
               D = DistEuclid(X[i,:], Km[j], dim)
               if minD > D:
                   minD = D
                   Y_class[i] = j + 1
```

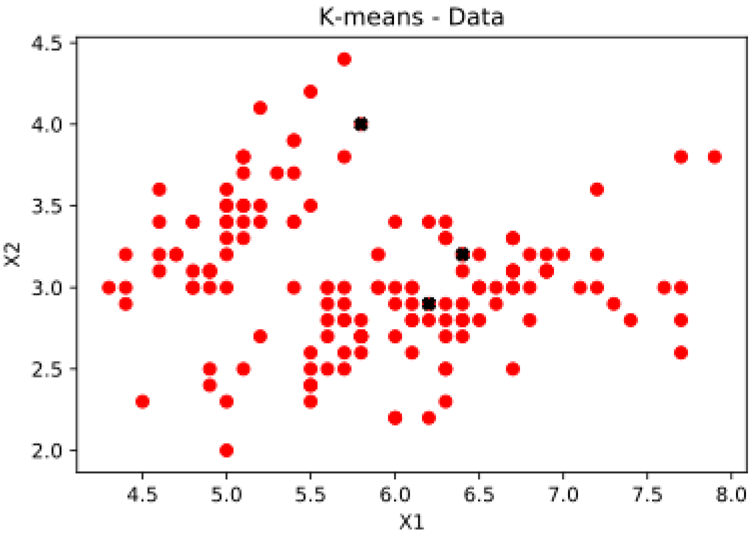
```
        #ajuste das posições dos pontos Kmean
        for i in range(0, len(Km)):
            for j in range(0, len(X[0])):
                Km[i,j] = np.mean(X[Y_class == i + 1][:,j])

        #  plot de dados com divisão de classes e ponto Kmean ajustados
        plt.figure()
        plt.scatter(X[:,dimx],  X[:,dimy],  c  =  Y_class,  cmap  =
'rainbow')
        plt.scatter(Km[:,dimx], Km[:,dimy], color = 'black', marker =
'X')
        plt.title('K-means - Data')
        plt.xlabel('X1')
        plt.ylabel('X2')
        plt.show()

        #verifica se a posição anterior dos centroides foi modificada.
        if (Km == Km2).all(): execute_kmean = False

    return
```

4. Resultados

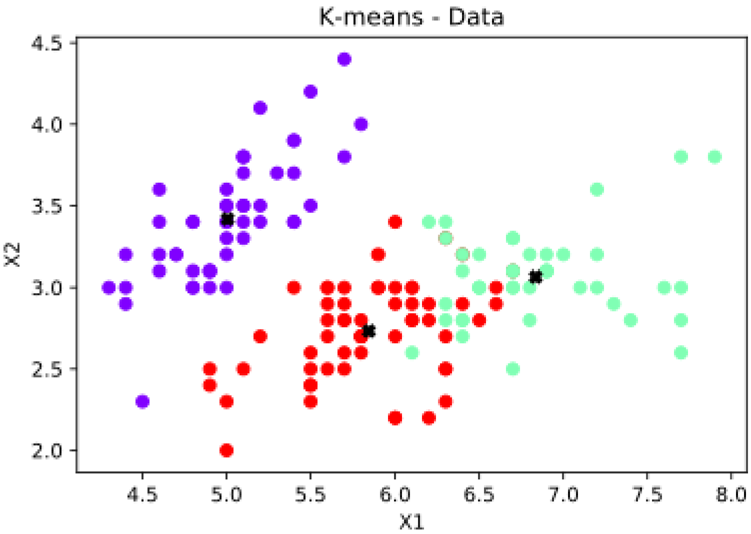
Para se testar o algoritmo do K-means foram testados três datasets: O Iris Fisher de agrupamento de flores, o Mall_Custumers de agrupamentos de tipo de compradores de um shopping, e o Wine da UCI de tipos de vinho.

IRIS:

O Iris do Fisher tem em seu dataset a seguinte distribuição:

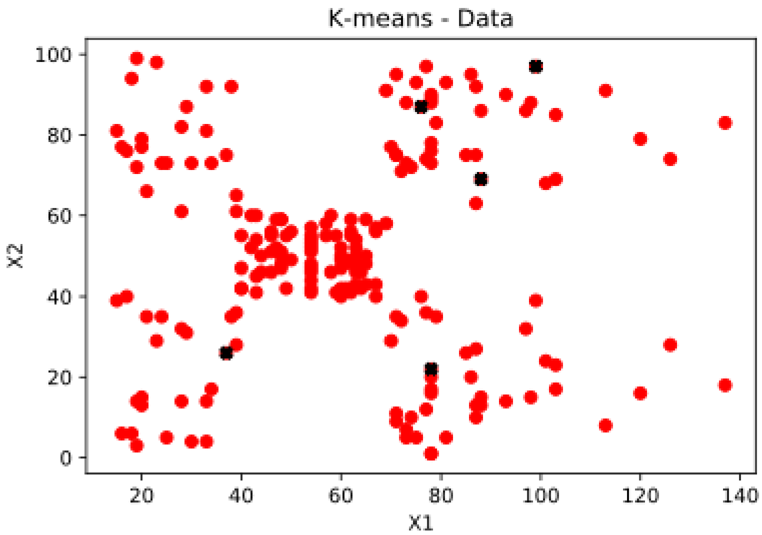


Foram alocados 3 centroides aleatoriamente neste e após algumas iterações resultou na seguinte divisão considerando-se 3 classes e 4 dimensões

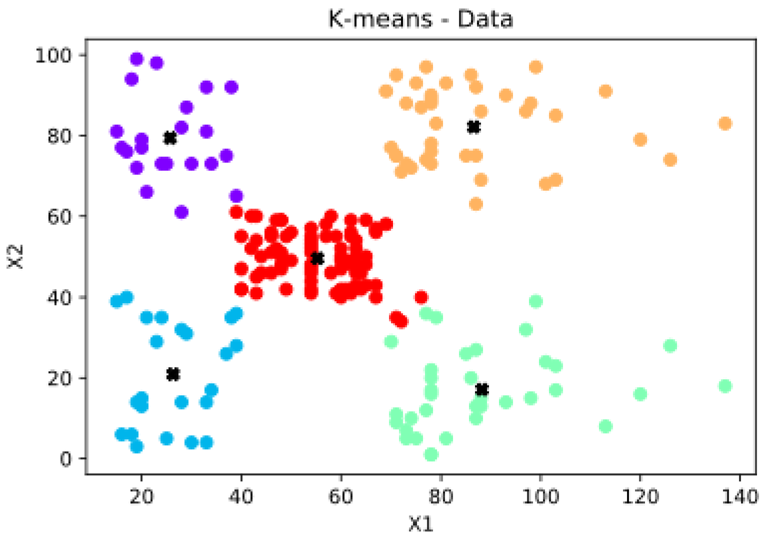


MALL CUSTOMERS:

O Mall_Customers relaciona o salário das pessoas com seu índice de consumo no shopping e resulta em 5 classes diferentes da seguinte forma:



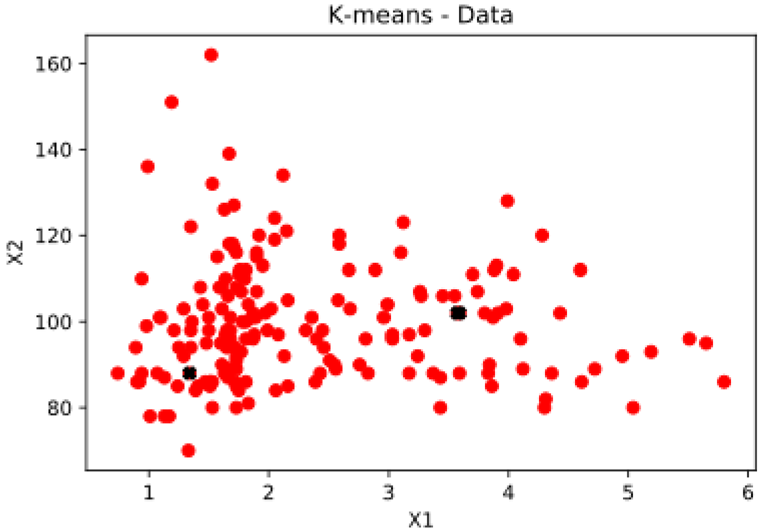
Atribuição aleatória dos centroides.



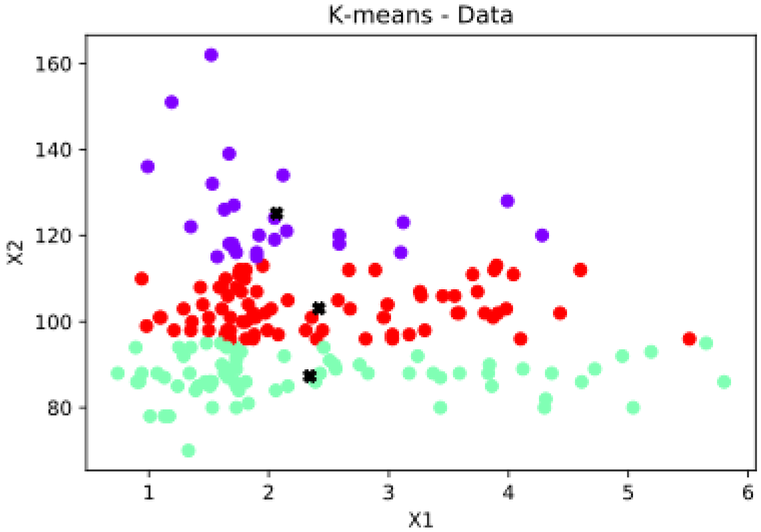
Classes separadas em 5 grupos.

WINE:

O Wine dataset consiste em 13 variáveis de observações sobre 3 tipos de vinho. E seu analise pelo K-means consiste na seguinte distribuição:

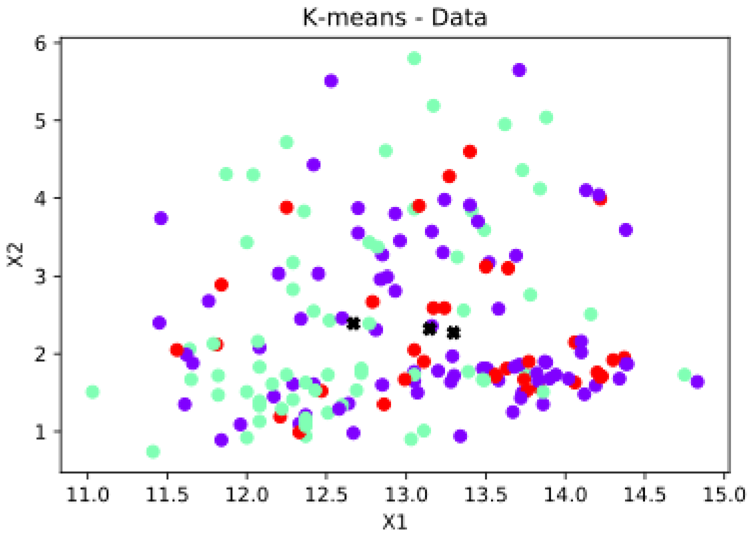


Dataset sem divisão com centroides aleatoriamente distribuídos.



Divisão dos pontos em 3 classes distintas.

Especialmente para datasets com muitas dimensões é necessário se analisar após serem feitas as divisões das classes como plotar os dados em um gráfico. Por exemplo, no gráfico acima, fica clara a diferenciação das classes, mas se escolhermos outra dimensão para impressão o resultado não é tao claro como no gráfico a seguir para o mesmo dataset.



5. Conclusão

Com estes experimentos pode-se verificar a eficácia e limitações do algoritmo k-means, sendo este muito versátil e de certa forma até simples sua implementação. Por ter sido feita uma analise superficial não foram avaliadas métricas de qualidade da separação, mas aliando LDA, PCA e K-means pode-se construir um algoritmo de clustering extremamente eficiente para os mais diversos dataset.

Notou-se que a atribuição aleatória dos centroides tem algumas limitações, onde caso um centroide não venha a ser o mais próximo de nenhum ponto de observação pode causar problemas de convergência do algoritmo.

6. Referências bibliográficas

[1] MacQueen, J. B. (1967). *Some Methods for classification and Analysis of Multivariate Observations*. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press.

[2] Lloyd, Stuart P. (1982), "Least squares quantization in PCM". IEEE Transactions on Information Theory