Exercício 4

Mateus Augusto Bellomo Agrello Ruivo RA:147338

28 de outubro de 2016

1 Código

```
from __future__ import division
import sklearn
import csv
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt
from numpy import genfromtxt
from sklearn.cluster import KMeans
from sklearn import metrics
from sklearn.metrics import pairwise_distances

X = genfromtxt('cluster-data.csv', delimiter=',', skip_header=1)
y = genfromtxt('cluster-data-class.csv', delimiter=',', skip_header=1)

xGraph = []
yGraph_adjustedRand = []
yGraph_adjustedMutualInfo = []
yGraph_vmeasure = []
yGraph_silhouette = []
yGraph_calinskiHarabaz = []
for k in range(2, 11):
    xGraph.append(k)

    kmeans = KMeans(n_clusters=k, n_init=5).fit(X)

    yGraph_adjustedRand.append(metrics.adjusted_rand_score(y, kmeans.labels_))
    yGraph_adjustedMutualInfo.append(metrics.adjusted_mutual_info_score(y, kmeans.labels_))
    yGraph_vmeasure.append(metrics.v_measure_score(y, kmeans.labels_))

    yGraph_silhouette.append(metrics.silhouette_score(X, kmeans.labels_, metric='euclidean'))
    yGraph_calinskiHarabaz.append(metrics.calinski_harabaz_score(X, kmeans.labels_))


plt.xlabel('number of clusters (k)')
plt.ylabel('score value')
plt.plot(xGraph, yGraph_adjustedRand, 'r--', label="adjusted rand score", linewidth=2.0)
plt.plot(xGraph, yGraph_adjustedMutualInfo, 'b--', label="adjusted mutual info score", linewidth=2.0)
plt.plot(xGraph, yGraph_vmeasure, 'g--', label="v-measure score", linewidth=2.0)
plt.legend(bbox_to_anchor=(1, 1), bbox_transform=plt.gcf().transFigure)

plt.savefig('medidaExterna.png')


# normalize calinski-harabaz so we can compare it with silhouette score
yGraph_calinskiHarabaz = yGraph_calinskiHarabaz/np.linalg.norm(yGraph_calinskiHarabaz)

plt.gcf().clear()
plt.xlabel('number of clusters (k)')
plt.ylabel('score value')
plt.plot(xGraph, yGraph_silhouette, 'r--', label="silhouette coefficient", linewidth=2.0)
plt.plot(xGraph, yGraph_calinskiHarabaz, 'b--', label="Calinski-Harabaz index", linewidth=2.0)
plt.legend(bbox_to_anchor=(1, 1), bbox_transform=plt.gcf().transFigure)

plt.savefig('medidaInterna.png')
```

2 Métrica interna

O gráfico 1 compara a utilização de 2 métricas internas diferentes (*silhouette coefficient*, *Calinski-Harabaz index*) com o número de clusters variando no intervalo [2,10]. Pela avaliação do gráfico é possível observar que obtêm-se um melhor score para 2 clusters no método *silhouette coefficient* e para 3 clusters no método *Calinski-Harabaz index*, o que não está de acordo também com a métrica externa (descrita posteriormente).

Vale ressaltar aqui que a métrica *Calinski-Harabaz* não é normalizada. Para poder compará-la com a métrica *silhouette coefficient* no gráfico 1 realizei normalização do score.

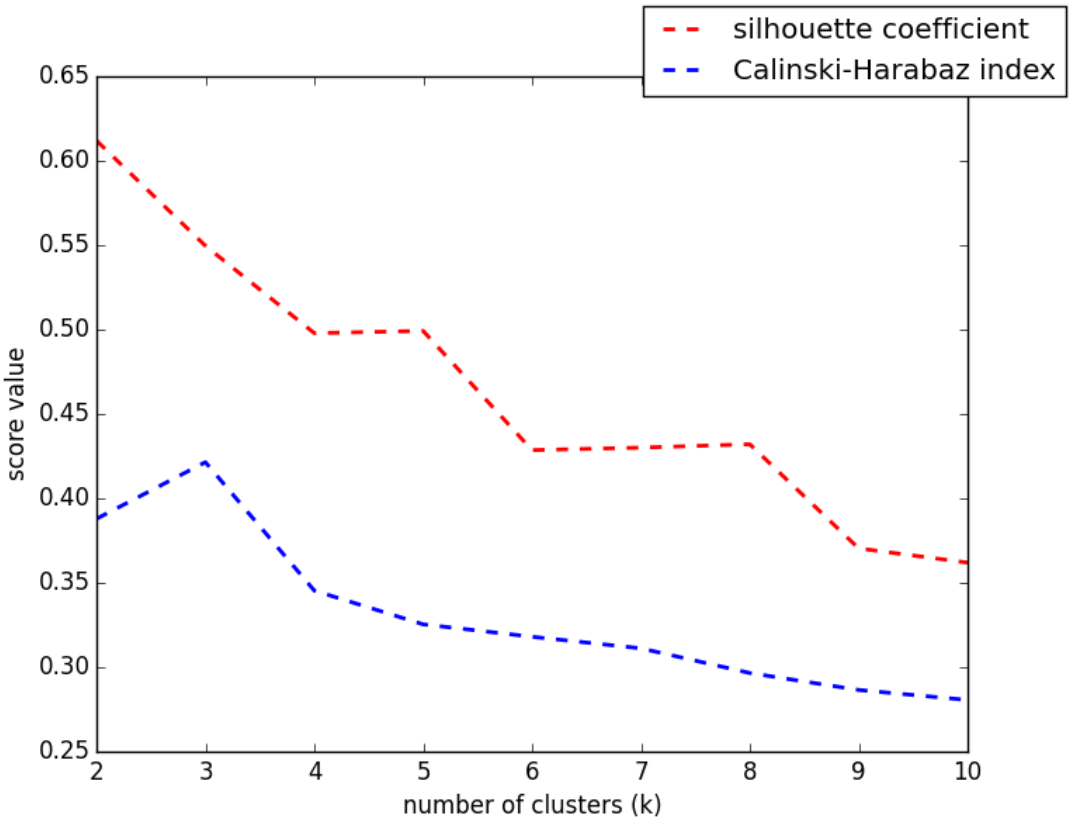

Figura 1: Métrica interna com variação de k no intervalo [2,10].

3 Métrica externa

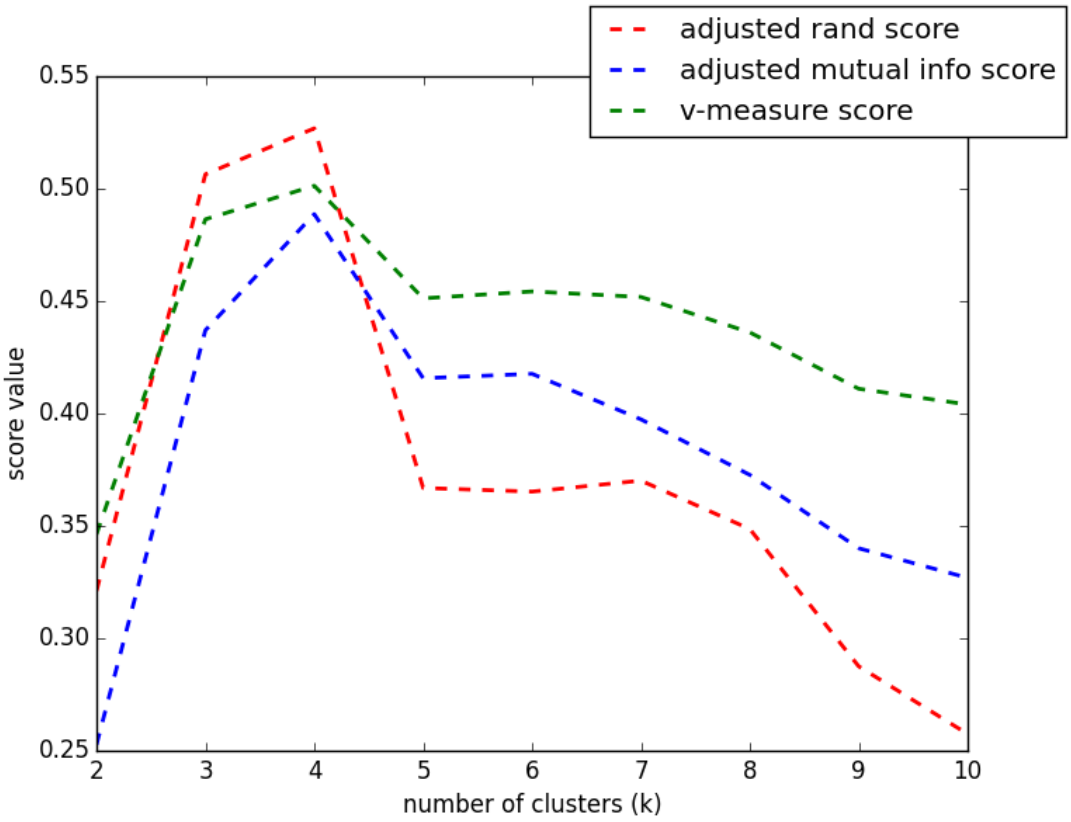

Figura 2: Métrica externa com variação de k no intervalo [2,10].

O gráfico 2 compara a utilização de 3 métricas externas diferentes (*adjusted rand score*, *adjusted mutual info score*, *v-measure score*) com o número de clusters variando no intervalo [2,10]. Pela avaliação do gráfico é possível observar que obtêm-se um melhor score para 4 clusters, e isso ocorre utilizando qualquer uma das três métricas externas.

4 Referências

[1] http://matplotlib.org/users/
[2] http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[3] http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation