Ydatalytics Case Study: Document Classification

Due on Friday, September 1, 2017

For Data Scientists

Dr. Marcello Cacciato

Contents

Section 1

Listing 2 shows a Perl script.

Listing 1: A sample

```
# coding: utf-8

# In[1]:


from sklearn import cluster
from scipy.spatial import distance
import sklearn.datasets
from sklearn.preprocessing import StandardScaler
import numpy as np


# In[2]:

def compute_bic(kmeans,X):
    """
    Computes the BIC metric for a given clusters

    Parameters:
    -----------------------------------------
    kmeans:  List of clustering object from scikit learn

    X     :  multidimension np array of data points

    Returns:
    -----------------------------------------
    BIC value
    """
    # assign centers and labels
    centers = [kmeans.cluster_centers_]
    labels  = kmeans.labels_
    #number of clusters
    m = kmeans.n_clusters
    # size of the clusters
    n = np.bincount(labels)
    #size of data set
    N, d = X.shape

    #compute variance for all clusters beforehand
    cl_var = (1.0 / (N - m) / d) * sum([sum(distance.cdist(X[np.where(labels == i)
        ], [centers[0][i]],
             'euclidean')**2) for i in range(m)])

    const_term = 0.5 * m * np.log(N) * (d+1)

    BIC = np.sum([n[i] * np.log(n[i]) -
               n[i] * np.log(N) -
             ((n[i] * d) / 2) * np.log(2*np.pi*cl_var) -
             ((n[i] - 1) * d/ 2) for i in range(m)]) - const_term
```

Section 1 continued on next page. . .

```
    return(BIC)


# In[3]:

# IRIS DATA
iris = sklearn.datasets.load_iris()
X = iris.data[:, :4]  # extract only the features
#Xs = StandardScaler().fit_transform(X)
Y = iris.target


# In[4]:

ks = range(1,10)


# In[5]:

# run 9 times kmeans and save each result in the KMeans object
KMeans = [cluster.KMeans(n_clusters = i, init="k-means++").fit(X) for i in ks]


# In[6]:

# now run for each cluster the BIC computation
BIC = [compute_bic(kmeansi,X) for kmeansi in KMeans]

print BIC
```

Listing 2: Sample Perl Script With Highlighting

```
#!/usr/bin/perl

use strict;
use warnings;

for (1..99) { print $_." Luftballons\n"; }

# This is a commented line

my $string = "Hello World!";

print $string."\n\n";

$string =~ s/Hello/Goodbye/;

print $string."\n\n";

test();

exit;
```

```
    sub test { print "All good.\n"; }
```