MARS 2020 test

The file rocks-assessment3.arff (included with this assessment brief) contains four measurements of each of the 185 samples analysed by the Opportunity rover. The four measurements are: reflective-red, reflective-blue, hard, and porous. Each one is normalised to be on a 0-10 scale.

Your assignment is to work out how many sample tubes will be needed by the Mars 2020 rover using the data collected by Opportunity. To maximise the scientific value of the samples, we want the rover to collect exactly one sample of each distinct type of rock. For example, if there were three types of rock, we would collect one sample of each and use three tubes.

In more detail, your task is to:

1. Perform k-means clustering for a set of suitable values of k. For each value of k, record (at least) the within cluster sum of squared errors and include it in your report.

2. Plot the within cluster sum of squared errors against k as a line plot. Include the plot in your report. You may use any suitable tool to make the plot. Spreadsheets such as Microsoft Excel, LibreOffice, and Google Sheets are able to make line plots.

3. Determine (using your plot) a suitable value of k for this dataset. Describe in your report how you determined this value, and why it is a suitable value.

4. Using your chosen value of k, read (from the output of WEKA) the number of examples allocated to each cluster, and include this in your report. Are the examples approximately equally split among the clusters?

5. Once again using your chosen value of k, test the k-means implementation in WEKA to see if it consistently produces the same set of cluster centroids. To do this, you will need to use several different values of seed (the seed of the random number generator). The value of seed can be changed using the same configuration window that is used to set the value of k. Report your results, and comment on whether the implementation of k-means produces consistent results on this data set.

MARS 2020 solution

I am going to use a machine learning technique called K-means clustering to analyse the data collected by Opportunity rover. In this way I should be able to group the data according to the rock types so that we can equip the new rover with the correct number of tubes which equal the number of types of rocks to collect.

1. Perform *k*-means clustering for a set of suitable values of *k*. For each value of *k*, record (at least) the *within cluster sum of squared errors* and include it in your report.

K=1

```
kMeans

======

Number of iterations: 1

Within cluster sum of squared errors: 45.0782191346161


Initial starting points (random):


Cluster 0: 2.777206,1.14027,8.325416,1.080847


Missing values globally replaced with mean/mode


Final cluster centroids:

                         Cluster#

Attribute        Full Data        0

                 (185.0)    (185.0)

=======================================

reflective-red        4.241      4.241

reflective-blue       2.0141     2.0141

hard                8.0454     8.0454

porous                5.3686     5.3686
```

```
Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      185 (100%)
```

K=2

```
kMeans
======

Number of iterations: 4
Within cluster sum of squared errors: 28.151838835749594

Initial starting points (random):

Cluster 0: 2.777206,1.14027,8.325416,1.080847
Cluster 1: 2.505611,3.188712,11.307323,9.732932

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute        Full Data        0        1
                   (185.0)   (139.0)    (46.0)
=====================================================
reflective-red       4.241    4.6864    2.8952
reflective-blue     2.0141    1.7043    2.9501
```

```
hard               8.0454     7.7244     9.0154

porous               5.3686     4.1331      9.102


Time taken to build model (full training data) : 0.01 seconds


=== Model and evaluation on training set ===


Clustered Instances


0      139 ( 75%)

1       46 ( 25%)
```

K=3

```
kMeans
======


Number of iterations: 6

Within cluster sum of squared errors: 17.21715674057593


Initial starting points (random):


Cluster 0: 2.777206,1.14027,8.325416,1.080847

Cluster 1: 2.505611,3.188712,11.307323,9.732932

Cluster 2: 3.548558,2.833865,10.025657,8.918419


Missing values globally replaced with mean/mode


Final cluster centroids:

                           Cluster#
```

```
Attribute        Full Data        0        1        2
                  (185.0)   (50.0)   (45.0)   (90.0)
============================================================
reflective-red       4.241    3.1045    2.8946    5.5456
reflective-blue      2.0141    0.7863    2.9559    2.2252
hard                 8.0454     7.716     9.026    7.7381
porous                5.3686    1.9777    9.1728    5.3503


Time taken to build model (full training data) : 0.01 seconds


=== Model and evaluation on training set ===


Clustered Instances


0       50 ( 27%)
1       45 ( 24%)
2       90 ( 49%)
```

K=4

```
kMeans
======


Number of iterations: 6
Within cluster sum of squared errors: 7.963945224325721


Initial starting points (random):


Cluster 0: 2.777206,1.14027,8.325416,1.080847
Cluster 1: 2.505611,3.188712,11.307323,9.732932
Cluster 2: 3.548558,2.833865,10.025657,8.918419

Cluster 3: 7.501657,2.106162,9.590414,5.38124


Missing values globally replaced with mean/mode


Final cluster centroids:

                        Cluster#

Attribute         Full Data        0        1        2        3

                 (185.0)    (50.0)    (44.0)    (46.0)    (45.0)

=====================================================================

reflective-red        4.241     3.1045      2.942     3.0929     7.9476

reflective-blue       2.0141     0.7863     2.9496     2.7192     1.7427

hard                 8.0454      7.716     9.0539      7.074     8.4183

porous                5.3686     1.9777     9.2623      5.352      5.346


Time taken to build model (full training data) : 0.02 seconds


=== Model and evaluation on training set ===


Clustered Instances


0       50 ( 27%)

1       44 ( 24%)

2       46 ( 25%)

3       45 ( 24%)
```

K=5

```
kMeans
======

Number of iterations: 7
Within cluster sum of squared errors: 7.19631512150502

Initial starting points (random):

Cluster 0: 2.777206,1.14027,8.325416,1.080847
Cluster 1: 2.505611,3.188712,11.307323,9.732932
Cluster 2: 3.548558,2.833865,10.025657,8.918419
Cluster 3: 7.501657,2.106162,9.590414,5.38124
Cluster 4: 3.719396,2.063613,8.985235,9.864693

Missing values globally replaced with mean/mode

Final cluster centroids:
                          Cluster#
Attribute       Full Data      0        1        2        3        4
                  (185.0)   (50.0)   (17.0)   (27.0)   (45.0)   (46.0)
=============================================================================
========
reflective-red      4.241   3.1045   2.7308   3.0749   7.9476   3.0929
reflective-blue    2.0141   0.7863   2.8722   2.9984   1.7427   2.7192
hard               8.0454    7.716   9.9976   8.4598   8.4183    7.074
porous             5.3686   1.9777   9.2201   9.2889    5.346    5.352
```

```
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       50 ( 27%)
1       17 (  9%)
2       27 ( 15%)
3       45 ( 24%)
4       46 ( 25%)
```

K=6

```
kMeans
======

Number of iterations: 9
Within cluster sum of squared errors: 6.944648885254871

Initial starting points (random):

Cluster 0: 2.777206,1.14027,8.325416,1.080847
Cluster 1: 2.505611,3.188712,11.307323,9.732932
Cluster 2: 3.548558,2.833865,10.025657,8.918419
Cluster 3: 7.501657,2.106162,9.590414,5.38124
Cluster 4: 3.719396,2.063613,8.985235,9.864693
Cluster 5: 3.363612,2.624417,9.329335,9.753074
```

```
Missing values globally replaced with mean/mode

Final cluster centroids:
                         Cluster#
Attribute         Full Data        0        1        2        3        4        5
                 (185.0)    (50.0)    (10.0)    (10.0)    (45.0)    (46.0)    (24.0)
==============================================================================
===================
reflective-red        4.241     3.1045     2.3289     3.3341     7.9476     3.0929     3.0341
reflective-blue       2.0141     0.7863     3.2773     2.4484     1.7427     2.7192     3.0219
hard                 8.0454      7.716    10.0507     9.6476     8.4183      7.074     8.3913
porous                 5.3686     1.9777     9.3963     9.1146      5.346      5.352     9.2681


Time taken to build model (full training data) : 0.02 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       50 ( 27%)
1       10 (  5%)
2       10 (  5%)
3       45 ( 24%)
4       46 ( 25%)
5       24 ( 13%)
```

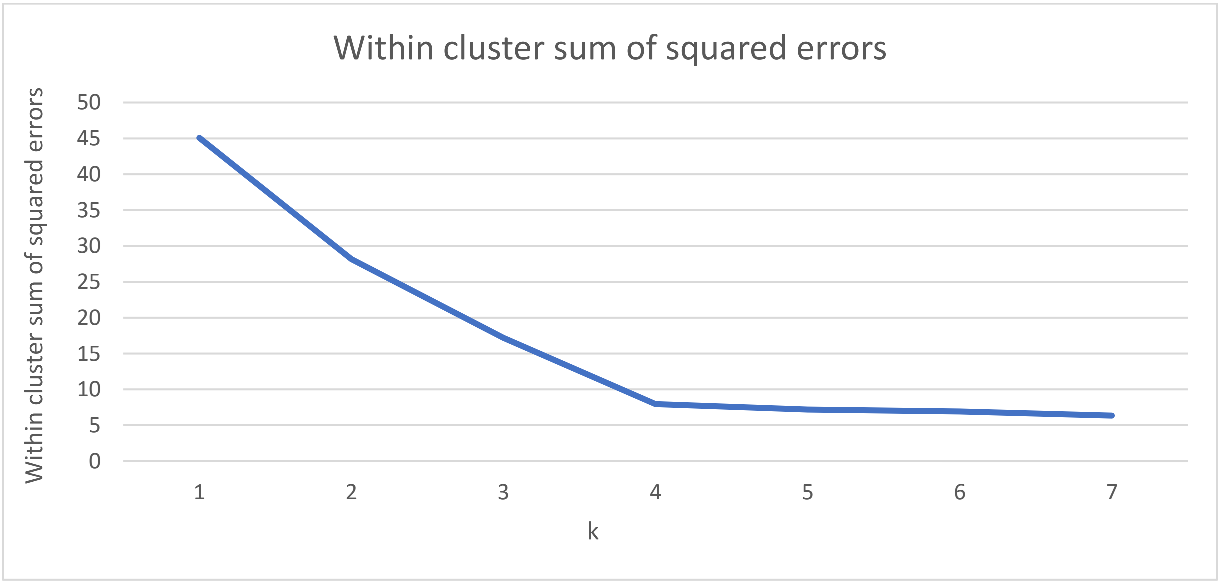
K=7

```
kMeans
======

Number of iterations: 11
Within cluster sum of squared errors: 6.3624118517906405

Initial starting points (random):

Cluster 0: 2.777206,1.14027,8.325416,1.080847
Cluster 1: 2.505611,3.188712,11.307323,9.732932
Cluster 2: 3.548558,2.833865,10.025657,8.918419
Cluster 3: 7.501657,2.106162,9.590414,5.38124
Cluster 4: 3.719396,2.063613,8.985235,9.864693
Cluster 5: 3.363612,2.624417,9.329335,9.753074
Cluster 6: 2.518608,2.738768,9.555985,8.819747

Missing values globally replaced with mean/mode

Final cluster centroids:
                     Cluster#
Attribute        Full Data       0        1        2        3        4        5        6
               (185.0)   (50.0)    (9.0)   (10.0)   (45.0)   (28.0)   (25.0)   (18.0)
==========================================================================
============================
reflective-red      4.241    3.1045   2.1898   3.3341   7.9476   3.3279   3.0559   2.7273
reflective-blue     2.0141   0.7863   3.1863   2.4484   1.7427   2.7434   3.0649   2.6815
hard               8.0454    7.716   10.1356   9.6476   8.4183   6.5757   8.4271   7.8491
porous              5.3686   1.9777   9.3778   9.1146    5.346   5.4187   9.2799   5.2483
```

```
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      50 ( 27%)
1       9 (  5%)
2      10 (  5%)
3      45 ( 24%)
4      28 ( 15%)
5      25 ( 14%)
6      18 ( 10%)
```

2. Plot the *within cluster sum of squared errors* against *k* as a line plot. Include the plot in your report. You may use any suitable tool to make the plot. Spreadsheets such as Microsoft Excel, LibreOffice, and Google Sheets are able to make line plots.

While performing the K-means clustering using different k each time I have noticed that the squared error diminished strongly and then continued to decrease but almost flattened meaning that we probably have reached the elbow point already. It's the time to plot the results and see.



3. Determine (using your plot) a suitable value of *k* for this dataset. Describe in your report how you determined this value, and why it is a suitable value

The best k to use for this dataset looks to be K=4 because as explained above that is where the substantial improvements from increasing k have occurred. Increasing k more than that would not be beneficial as it may risk to become to specific to the data sample which is made of 185 examples which is not a huge number.

4. Using your chosen value of *k*, read (from the output of WEKA) the number of examples allocated to each cluster, and include this in your report. Are the examples approximately equally split among the clusters?

Additionally, the data are divided almost equally among the clusters when using k=4 and no other value for k looks to give such equal division of examples among the clusters. That is highlighted in light blue in the below full output for k=4.

```
=== Run information ===


Scheme:         weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning
10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 4 -A "weka.core.EuclideanDistance -R first-last" -I
500 -num-slots 1 -S 10

Relation:     rocks

Instances:    185

Attributes:   4

              reflective-red

              reflective-blue

              hard

              porous

Test mode:    evaluate on training data


=== Clustering model (full training set) ===


kMeans

======
Number of iterations: 6

Within cluster sum of squared errors: 7.963945224325721


Initial starting points (random):


Cluster 0: 2.777206,1.14027,8.325416,1.080847

Cluster 1: 2.505611,3.188712,11.307323,9.732932

Cluster 2: 3.548558,2.833865,10.025657,8.918419

Cluster 3: 7.501657,2.106162,9.590414,5.38124


Missing values globally replaced with mean/mode


Final cluster centroids:

                         Cluster#

Attribute        Full Data        0        1        2        3

                 (185.0)    (50.0)    (44.0)    (46.0)    (45.0)

=====================================================================

reflective-red       4.241     3.1045      2.942     3.0929     7.9476

reflective-blue      2.0141     0.7863     2.9496     2.7192     1.7427

hard                 8.0454      7.716     9.0539      7.074     8.4183

porous               5.3686     1.9777     9.2623      5.352      5.346



Time taken to build model (full training data) : 0 seconds



=== Model and evaluation on training set ===
```

Clustered Instances

0 50 (27%)

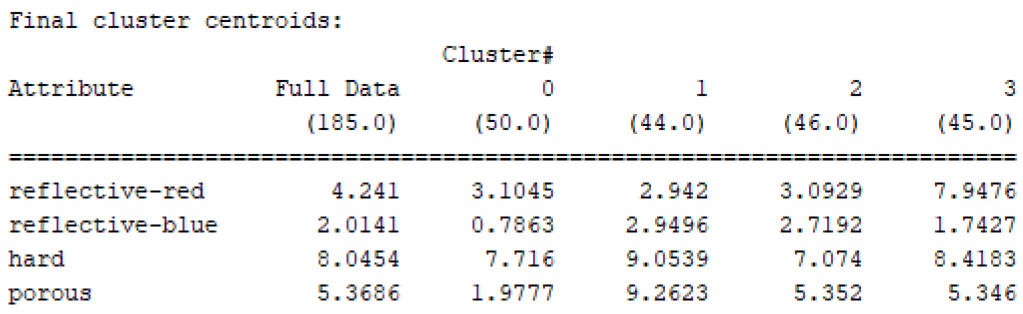
1 44 (24%)

2 46 (25%)

3 45 (24%)

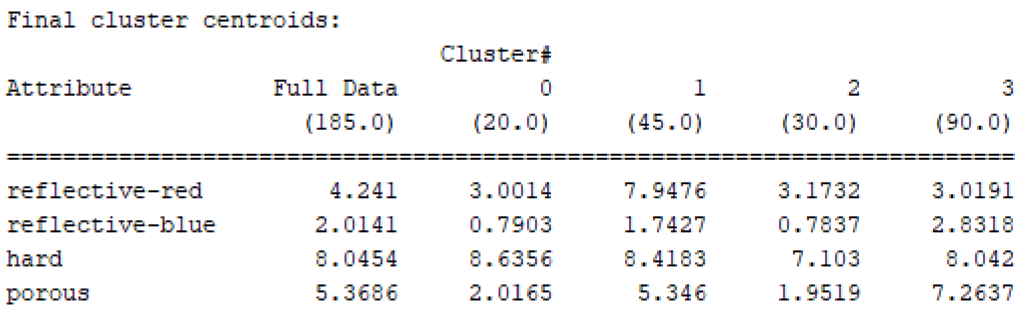
5. Once again using your chosen value of *k*, test the *k*-means implementation in WEKA to see if it consistently produces the same set of cluster centroids. To do this, you will need to use several different values of *seed* (the seed of the random number generator). The value of *seed* can be changed using the same configuration window that is used to set the value of *k*. Report your results, and comment on whether the implementation of *k*-means produces consistent results on this data set.

The final cluster centroids for the first run with k=4 (in a better format) were the below and had **seed=10**:



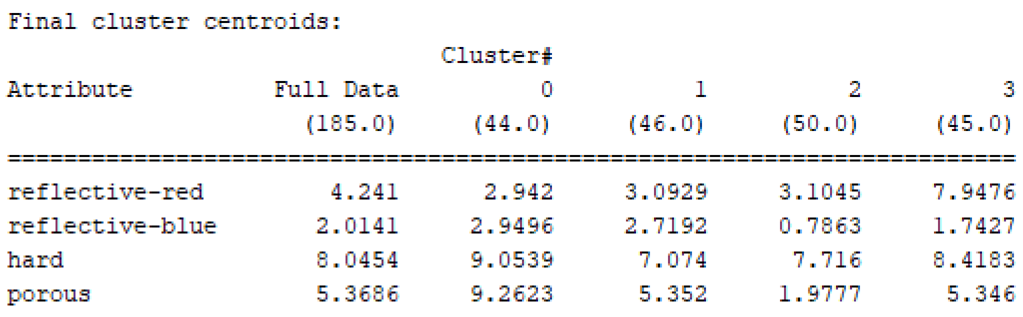
```
Final cluster centroids:
                                   Cluster#
Attribute            Full Data          0          1          2          3
                       (185.0)     (50.0)     (44.0)     (46.0)     (45.0)
==========================================================================
reflective-red           4.241     3.1045      2.942     3.0929     7.9476
reflective-blue         2.0141     0.7863     2.9496     2.7192     1.7427
hard                    8.0454      7.716     9.0539      7.074     8.4183
porous                  5.3686     1.9777     9.2623      5.352      5.346
```

Seed=11



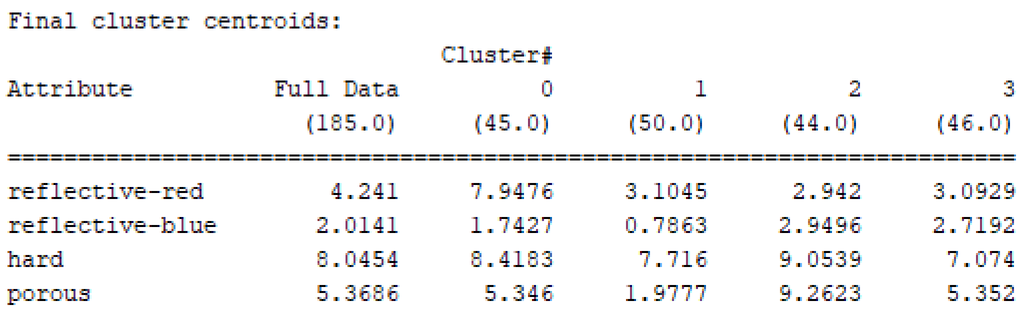
```
Final cluster centroids:
                                   Cluster#
Attribute            Full Data          0          1          2          3
                       (185.0)     (20.0)     (45.0)     (30.0)     (90.0)
==========================================================================
reflective-red           4.241     3.0014     7.9476     3.1732     3.0191
reflective-blue         2.0141     0.7903     1.7427     0.7837     2.8318
hard                    8.0454     8.6356     8.4183      7.103      8.042
porous                  5.3686     2.0165      5.346     1.9519     7.2637
```

Seed=12



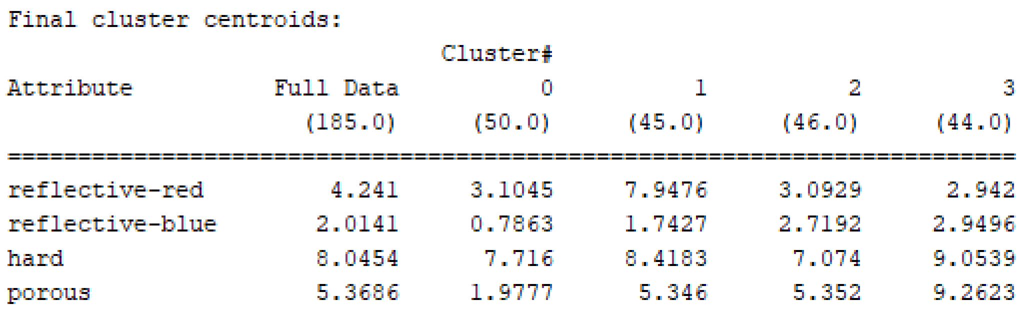
```
Final cluster centroids:
                                 Cluster#
Attribute            Full Data          0          1          2          3
                       (185.0)     (44.0)     (46.0)     (50.0)     (45.0)
==========================================================================
reflective-red           4.241      2.942     3.0929     3.1045     7.9476
reflective-blue         2.0141     2.9496     2.7192     0.7863     1.7427
hard                    8.0454     9.0539      7.074      7.716     8.4183
porous                  5.3686     9.2623      5.352     1.9777      5.346
```

Seed=13



```
Final cluster centroids:
                                 Cluster#
Attribute            Full Data          0          1          2          3
                       (185.0)     (45.0)     (50.0)     (44.0)     (46.0)
==========================================================================
reflective-red           4.241     7.9476     3.1045      2.942     3.0929
reflective-blue         2.0141     1.7427     0.7863     2.9496     2.7192
hard                    8.0454     8.4183      7.716     9.0539      7.074
porous                  5.3686      5.346     1.9777     9.2623      5.352
```

Seed=14



```
Final cluster centroids:
                                 Cluster#
Attribute            Full Data          0          1          2          3
                       (185.0)     (50.0)     (45.0)     (46.0)     (44.0)
==========================================================================
reflective-red           4.241     3.1045     7.9476     3.0929      2.942
reflective-blue         2.0141     0.7863     1.7427     2.7192     2.9496
hard                    8.0454      7.716     8.4183      7.074     9.0539
porous                  5.3686     1.9777      5.346      5.352     9.2623
```

The results show that K-means clustering produces optimal results only partially. In fact, cluster 1 and 3 are quite volatile meaning that there is some overlap in the data that do not allow clustering to be clean and optimal.

Also, when seed=11 the Within cluster sum of squared errors is 14.0971024947896 whereas with all the other seed values we get a similar Within cluster sum of squared errors to what we got when seed=10 which is 7.9639452243257205.

To conclude, we should bring 4 tubes on Mars with the new rover as chances are that we will need to bring home only 4 types of rocks.