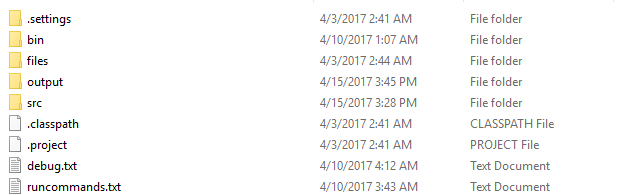
2. K-means clustering on imagesIn this problem, you will use K-means clustering for image compression. We have provided you  
with two images.

We have provided you java template KMeans.java which implements various image  
input/output operations. You have to implement the function k-means in the template. See the  
file for more details. Note that your program must compile and we should be able to replicate  
your results. Otherwise no credit will be given.  
What to turn in:  
1. Your code and datasets (in ARFF format)  
2. A README for your compiling/using your code  
3. A report (pdf or doc file) containing answers to the questions posed.

Readme:

**Source code Execution:**

Source Code Structure:  
Means is the base folder, it has 4 subfolders.  
1) files which contains input image files.  
2) output which contains output image files.  
3) src which contains the source code



1.To compile the code.  
open the command line go to Kmeans folder and src execute “javac \*.java”



To execute the code,

1) move to src.

use the below format.

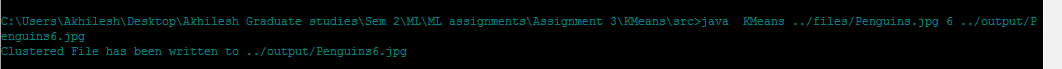
java KMeans <input folder> K <outputfolder>

There are some sample files in previous folder "files".

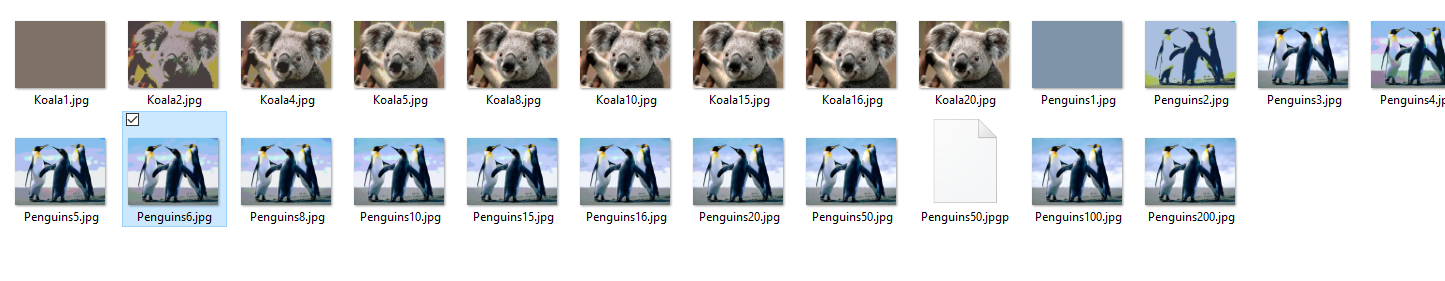
java KMeans ../files/Penguins.jpg 200 ../output/Penguins200.jpg

java KMeans ../files/Koala.jpg 10 ../output/Koala10.jpg

Alternatively if on windows use runme.bat to execute the compiling and running the code.



Go back Kmeans folder and check the /output :  
you will see penguin image clustering into 6 colors.



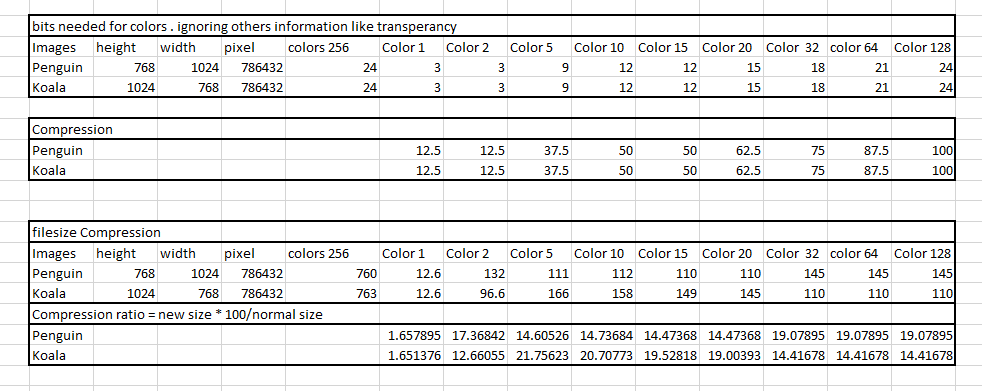
• Display the images after data compression using K-means clustering for different values of K  
(2, 5, 10, 15, 20).

KMean Clustered Images (reduced to 1 inch)

|  |  |  |
| --- | --- | --- |
|  | Penguin | Koala |
| Original | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Penguins.jpg | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Koala.jpg |
| 1 | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Penguins1.jpg | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Koala1.jpg |
| 2 | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Penguins2.jpg | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Koala2.jpg |
| 5 | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Penguins5.jpg | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Koala5.jpg |
| 10 | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Penguins10.jpg | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Koala10.jpg |
| 15 | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Penguins15.jpg | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Koala15.jpg |
| 20 | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Penguins20.jpg | C:\Users\Akhilesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Koala20.jpg |

• What are the compression ratios for different values of K? Note that you must repeat the  
experiment multiple times with different initializations and report the average as well as  
variance in the compression ratio.

In the code that we exercised the compression technique still stores the values of R, G, B values as 3 bit and relevance we can see is only of the lossless compression in the JPG.  
However for the calculation purpose, we use if K colors then we use log 2 K (base2) bits each for R,G,B forgetting the value of Transparency to compare the ratio.



Interpretation Is for color 1: The image is reduced to 12.5% of the size. i.e. compressed image is 12.5 size unit if the original was 100. (lower number the better)

Comparing file sizes won’t give good result as still the image will be represented in jpg format using the 8bits (256 colors), even though we have lots of less colors. However, since jpg uses Huffman coding, this might give a better result. The compression ratio for one color is too less, but for remaining colors, it is the same size 100-150KB.

When we reduce the number of colors from 256 colors to K colors, if we consider bit needed to represent an RGB values, it is log K (base 2) bits per color. For example, 2 values for each color Red, Green or Blue will require 1 bit and since there are 3 colors we need 3 bits. Similarly, color 64 needs 7bits and since there are 3 colors, 7 \* 3 = 21 bits (instead of Original 8 bits). The table 1 and table 2 displays the value obtained.  
  
For values of K =0,1 it is 12.5 % of original image. For values of K = 5 it is 37.5% of original image and it goes on.

• Is there a tradeoff between image quality and degree of compression? What would be a good  
value of K for each of the two images?

Comparing both compression and image quality:  
The images with K=1 has only one color no boundaries are seen. The Image is not clear, but compression is maximum.   
With k=2, Boundary of the images can be seen, but the real colors are not seen. The compression is 12.5% of original image. (8 colors in total)  
With k=5, Different base colors can be seen clearly, but similar colors like blue and deep blue are not differentiated. ( 3^5= 243 colors in total)  
Then the image clarity improves with K, but image is less compressed.  
for penguin image k = 50 , it shows most of the true colors.(decide by using previous image and seeing the change) and 87.5% of the original image.  
for Koala image k =20, it shows most of the true colors.(decided by using previous image and seeing the change.) and 62.5% if the original image.  
  
Depending on the what clarity is needed for the image we can choose different K values for a particular image.