





Course Title:	Intelligent Systems
Course Number:	ELE888
Semester/Year (e.g. F2016)	W2021

Instructor:	Dr. Xiao-Ping Zhan
-------------	--------------------

Assignment/Lab Number:	4
Assignment/Lab Title:	Unsupervised Learning

Submission Date:	April 11, 2021
Due Date:	April 11, 2021

Student Last Name	Student First Name	Student Number	Section	Signature
Paramsothy	Tharsigan	500757799	04	
Luu	Austin	500774611	04	

*By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: <http://www.ryerson.ca/senate/current/pol60.pdf>

Table of Contents

Introduction	3
Procedure	4
Discussion	5
Conclusion	9
Appendix B : Matlab Code	10

Introduction

The objective of this lab was to classify unlabeled data using a K-means algorithm. This is an unsupervised learning technique, which assumes that the underlying probabilities are known and so only the value of an unknown parameter vector must be determined. K-means algorithm, in other words is used to determine “k” mean vectors or cluster centers in an unlabeled data set. This algorithm has six steps as shown in Figure 1. It consists of first randomly initializing the “k” mean vectors and using the Euclidean distance to assign the data set to the nearest cluster (grouping to nearest cluster). Afterwards, the mean vector of each cluster is recalculated. This process is then iterated/repeated until there is no change after recomputing the “k” mean vector. In this lab, an unlabeled data set with each data point representing a color content of a pixel was given. This data set was grouped into “k” clusters and the final “k” mean vectors were to be determined. Lastly, the quality of the clustering solution was assessed using the Xie-Beni (XB) Index.

1. begin initialize $n, c = k, \mu_1, \mu_2, \dots, \mu_c$
2. do classify n samples according to nearest μ_i
3. recompute μ_i
4. until no change in μ_i
5. return $\mu_1, \mu_2, \dots, \mu_c$
6. end

Figure 1. Pseudo Code of the K-means Algorithm

$$XB(c) = \frac{1}{N} \cdot \sum_{k=1}^N \sum_{j=1}^c \frac{\mathbf{u}_{jk} \|\mathbf{x}_k - \mu_j\|}{\min_i \|\mu_i - \mu_j\|}$$

where $\mathbf{u}_{jk} = \hat{P}(\omega_j | \mathbf{x}_k) = \begin{cases} 1 & \text{if } x_k \in j^{th} \text{ cluster} \\ 0 & \text{if } x_k \notin j^{th} \text{ cluster} \end{cases}$

Figure 2. Xie-Beni Index formula

Procedure

1. Run the K-means algorithm on the “house.tiff” image file, Given the parameters: $C=2$, and Initial state: randomly generated.
 - a. Record the initial means used.
 - b. Plot the Error Criterion, J .
 - c. Plot the cluster means for at least 2 of the process stages, and the final stage.
 - d. Plot the data set with respect to its unique RGB value or color.
 - e. Plot the whole image formed after successfully identifying each pixel's colour.
2. Run 2 independent K-means algorithms on the “house.tiff” image file, Given the parameters: $C=5$, Initial state 1: random, Initial State 2: random.
 - a. Record the initial means used.
 - b. Plot the labelled data set with respect to its colour or RGB value for each algorithm separately.
 - c. Record any discrepancies between the two plots.
3. Assess the quality of the two clustering solutions from K-means algorithm found in Step 2 using the Xie-Beni (XB) index.
 - a. Justify the results.

*The code is included in the appendix section with comments to explain each section

Discussion

Lab 4: Part A

For the problem of K-means clustering, image processing and rendering is accomplished. Given a sample image, "house.tiff" the K most dominant colors from the sample image are discovered. For Part A, C is set to 2 for determining the 2 most dominant colors in an RGB space. Initially the image is reshaped into a 256 x 256px image where each pixel is thereafter re-constructed as RGB vectors (p_i). Using the restructured image, K-means algorithm is run to determine the 2 most dominant colors using pre-set initial means of [0.1 0.1 0.1].

As can be seen from figure 3 depicting the Error Criterion function J across iterations, after approx. 2×10^8 iterations the error quickly falls off before reaching no changes in the mean after approx. 6×10^8 iterations, signalling convergence of the K-means algorithm. Figure 4 depicts the plot of the cluster means during various iterations of K-mean clustering. A plot of the converged K-means algorithm for the dominant colors can be shown in figure 5 where the data samples are plotted in RGB space. Using the determined dominant colors, the "house.tiff" image is rebuilt and rendered in figure 6.

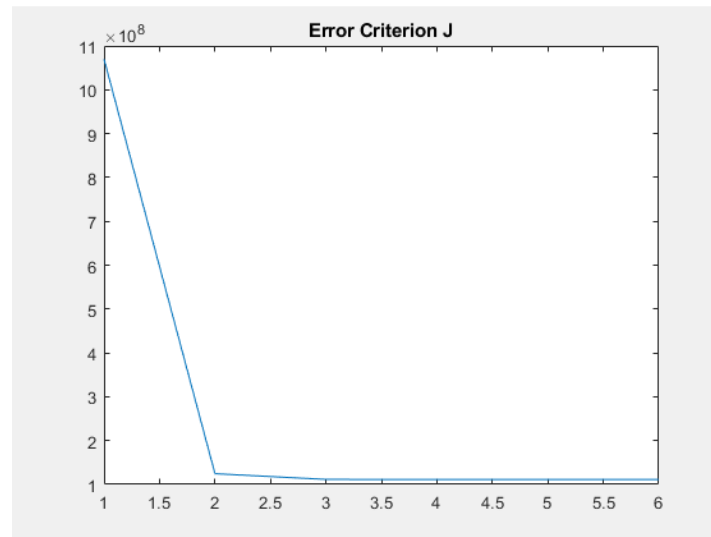


Figure 3: Plot of Error Criterion, J



Figure 4: Plot of Cluster Mean During Clustering and Final Value

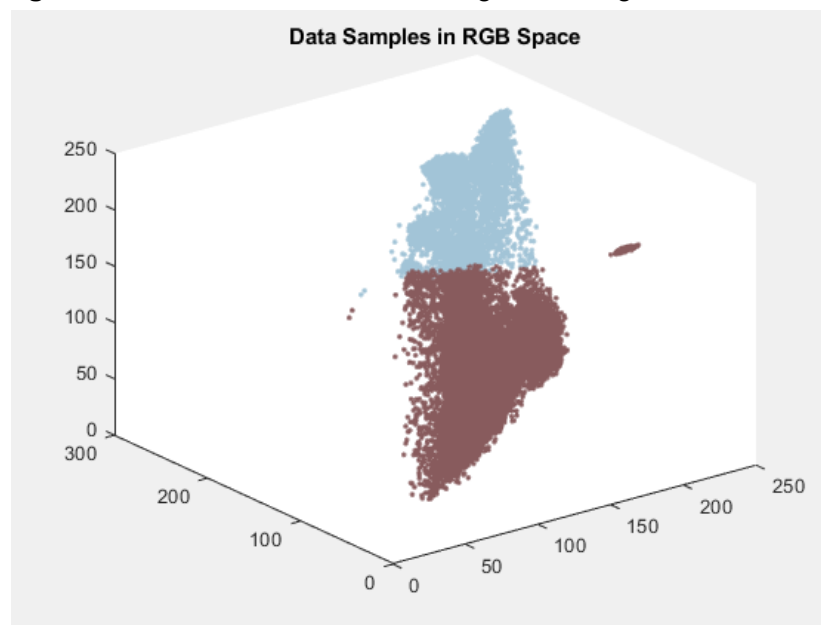


Figure 5: Plot of Data Samples in RGB Space using 2 Clusters



Figure 6: Render of image in labelled form using 2 Clusters

Lab 4: Part B

For the second part of the lab the problem was expanded to $C = 5$ to incorporate more colors for the K-means clustering algorithm. To speed up and potentially increase the accuracy of the algorithm's potential to convergence, the 5 initial states of the means being uniquely set. Two independent runs with varying means were set. For figure 7 below, the plot of the labeled data samples with means set as m1 listed in Appendix B were used to run the K-means clustering algorithm. Whereas figure 8, means set as m2 Appendix B were used. As can be seen by the varying means, a discrepancy in the colors and clustering distribution occurs. The most notable of which is the hue and saturation of the dark brown almost burgundy segment at the lower indexed data points in figure 8 in comparison to the larger light brown cluster shown in figure 7. This is likely largely due to two means being closely set for the second run, resulting in two distinct hues of brown shown in figure 6. Overall from a visual perspective figure 8 appears more accurate to the original image due to capturing the various hues of brown across the center and lower-right of the images bricking.

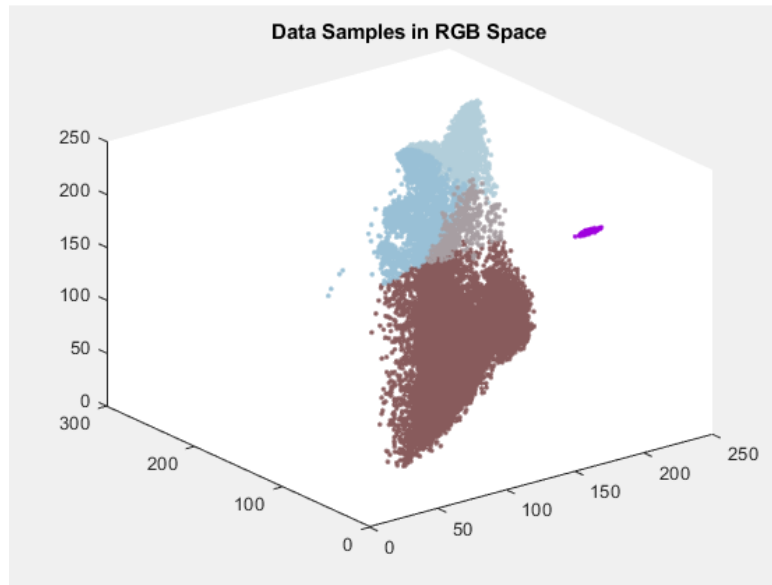


Figure 7: Plot of Data Samples in RGB Space using 5 Clusters

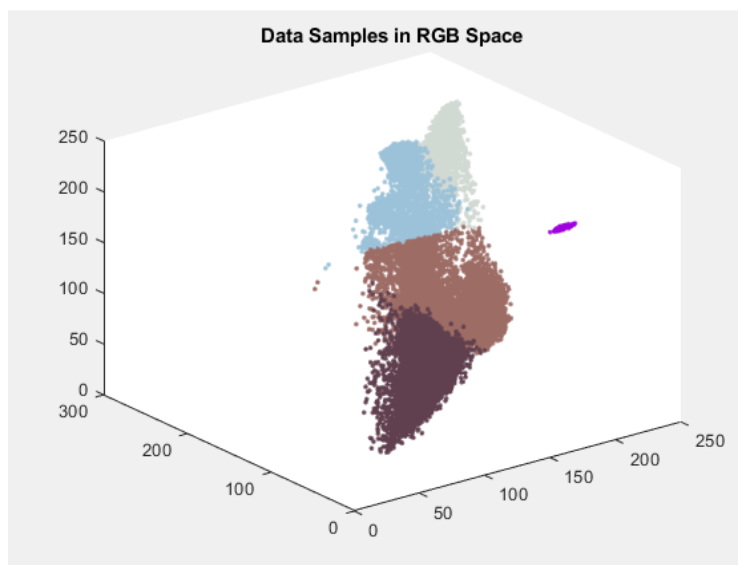


Figure 8: Plot of Data Samples in RGB Space using 5 Clusters

Lab 4: Part C

With the lack of certainty over accuracy with unlabelled data in unsupervised learning, the need to accurately assess a model's performance is preeminent. Rather than assessing various solutions by Error Criterion (J), or iterations to convergence (ln), the Xie-Beni (XB) index is used. The XB index is used to quantify and assess the quality of various K-means clustering solutions, where the smaller the index number the better the model performance in accurately clustering data. For the case of using the K-means algorithm with $C = 5$ as in part B, the XB index has been used to determine the performance of both. The results of the XB index for both models in Part B are shown in figure 9 below.

As mentioned in Part B, the second group of means resulting from the second grouping of means resulted in a visually more accurate representation of the original "house.tiff" image. The accuracy is apparent by the apparent change of brown hues in the houses' brickwork which align nicely to the RGB plot shown in figure 8. This observation is supported by the XB index results suggesting a lower XB index for the second model. The XB index helps in identifying the varying accuracy of models apparent in unsupervised learning algorithms. Much like supervised learning algorithms, unsupervised algorithms may have varying accuracy determined on preset conditions as well as the possibility of converging at local minima rather than absolute minima which is likely what had occurred for the first model in Part B. To mitigate this error various initializations for parameters should be set and tested with an evaluation algorithm such as XB index or Calinski Harabasz (CH) index should be used as an evaluation metric when computationally capable and available. To improve experimentation in the future, a further algorithm for determining the amount of clusters, C can be implemented to accurately replicate an image. As well, expansion to the lab by incorporating image compression as reduction in the colors by clustering is a common usage of K-means clustering in industry and a good metric by which to evaluate an algorithm.

```
XB1 =  
    0.4920  
  
XB2 =  
    0.2627
```

Figure 9: Xie-Beni Index Results

Conclusion

In conclusion, this experiment was a success as we were able to understand and implement the K-means algorithm to label an unsupervised data set consisting of pixel colours that form an image. For the iteration with $C=2$ (two colors), the initial mean values used were $[0.1 \ 0.1 \ 0.1; 0.9 \ 0.9 \ 0.9]$. For this process the K-means algorithm was iterated for about $2e8$ iterations, where after “k” mean values do not change and thus the error decreases dramatically. When plotting the data set with the final “k” mean values, the image of a house (Figure 6) was roughly formed with only 2 colours showcasing a satisfactory degree of accuracy with labelling the pixels. For the next part, with $C=5$, the randomly generated initial mean values used were: $[173.8240 \ 60.4672 \ 169.4464; 162.7648 \ 30.5664 \ 197.1968; 241.9712 \ 155.4688 \ 89.6512; 53.4784 \ 115.2256 \ 169.4720; 181.5808 \ 117.4272 \ 106.5472]$. After comparing the resulting mean values and assessing the two clustering solutions using the Xie-Beni index formula, it was determined that data set from Figure 8 had a higher degree of quality and accuracy in correctly labelling the pixels. This can be noted from Figure 8, which shows 5 distinct clustering, while Figure 7 only shows 4 groups of colours. Lastly, for the data set seen in Figure 8, the Xie-Beni index was found to be 0.2627, which compared to 0.4920 from the data set in Figure 7, is much smaller representing a higher degree of quality.

Appendix A : Matlab Code

Lab4.m

```
clc
clear all
close all
%% Set Up

I= imread('house.tiff');
X = reshape(I, 256*256, 3);
X = double(X);
R = [0,0,1];G = [0,1,0];B = [1,0,0];
figure, plot3(X(:,1), X(:,2), X(:,3),'.','Color',[1, 0, 0])

mean1 = [.1 .1 .1];

%% a- c=2

M = [.1 .1 .1
      .9 .9 .9];

M = M*256;

pre_M = zeros(size(M));

J = [];
M1_history = [M(1,:)];
M2_history = [M(2,:)];

while (pre_M ~= M)

    pre_M = M;

    J1 = (X - repmat(M(1,:), size(X,1), 1));
    J1 = sum(J1.^2, 2);

    J2 = (X - repmat(M(2,:), size(X,1), 1));
    J2 = sum(J2.^2, 2);

    cluster1 = J1 < J2;
    cluster2 = ~cluster1;

    M(1,:) = sum(X(cluster1, :)) / sum(cluster1);
```

```

M(2,:) = sum(X(cluster2, :)) / sum(cluster2);

J = [J sum(min(J1, J2))];
M1_history = [M1_history; M(1,:)];
M2_history = [M2_history; M(2,:)];

M

end

% i
close all
plot(J)
title('Error Criterion J')
% ii
figure
plot3(M1_history(:, 1), M1_history(:, 2), M1_history(:, 3), '-*')
hold all
plot3(M2_history(:, 1), M2_history(:, 2), M2_history(:, 3), '-*')
title('Cluster Means Final Value')

% iii
figure
X1 = X(cluster1, :);
X2 = X(cluster2, :);

plot3(X1(:,1), X1(:,2), X1(:,3), '.', 'Color', M(1,+)/256)
hold all
plot3(X2(:,1), X2(:,2), X2(:,3), '.', 'Color', M(2,+)/256)
title('Data Samples in RGB Space')

% iv
figure
XX = repmat(M(1,:), size(X,1), 1) .* repmat(cluster1, 1, size(X,2));
XX = XX + repmat(M(2,:), size(X,1), 1) .* repmat(cluster2, 1,
size(X,2));
XX = reshape(XX, size(I, 1), size(I, 2), 3);

subplot(1,2,1);imshow(I)
subplot(1,2,2);imshow(XX/256)

%% b

```

```

c = 5;

initial_M_1 = [
    173.8240    60.4672   169.4464
    162.7648    30.5664   197.1968
    241.9712   155.4688    89.6512
     53.4784   115.2256   169.4720
    181.5808   117.4272   106.5472
];
XX=[];
M = initial_M_1;

pre_M = zeros(size(M));

while (pre_M ~= M)

    pre_M = M;

    J = zeros(size(X,1), c);

    for i = [1:c]
        j = (X - repmat(M(i,:), size(X,1), 1));
        j = sum(j.^2, 2);
        J(:,i) = j;
    end
    [a, cluster] = min(J, [], 2);

    for i = [1:c]
        this_cluster = (cluster==i);
        M(i, :) = sum(X(this_cluster, :)) / sum(this_cluster);
    end

    end
XX = zeros(size(X));
for i = [1:c]
    this_cluster = (cluster==i);
    XX = XX + repmat(M(i,:), size(X,1), 1) .* repmat(this_cluster, 1,
size(X,2));
end
XX = reshape(XX, size(I, 1), size(I, 2), 3);
subplot(1,2,1);imshow(I)
subplot(1,2,2);imshow(XX/256)

end

```

```

M1 = M
cluster1 = cluster;
%
figure
for i = [1:c]
    this_cluster = (cluster==i);
    Xi = X(this_cluster, :);
    plot3(Xi(:,1), Xi(:,2), Xi(:,3), '.', 'Color', M(i,:)/256)
    title('Data Samples in RGB Space')
    hold all
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%5
initial_M_2 = [
    215.5339    138.4293    240.5963
    213.2267    222.7049    165.2613
    65.6489     67.7834    122.7426
    157.0459     81.4270    163.6651
    149.0558     30.5189    139.4473
];

M = initial_M_2;

pre_M = zeros(size(M));

while (pre_M ~= M)

    pre_M = M;

    J = zeros(size(X,1), c);

    for i = [1:c]
        j = (X - repmat(M(i,:), size(X,1), 1));
        j = sum(j.^2, 2);
        J(:,i) = j;
    end
    [a, cluster] = min(J, [], 2);

    for i = [1:c]
        this_cluster = (cluster==i);
        M(i, :) = sum(X(this_cluster, :)) / sum(this_cluster);
    end
end

```

```

end
M2 = M
cluster2 = cluster;

%
figure
for i = [1:c]
    this_cluster = (cluster==i);
    Xi = X(this_cluster, :);
    plot3(Xi(:,1), Xi(:,2), Xi(:,3),'.','Color', M(i,:)/256)
    title('Data Samples in RGB Space')
    hold all
end

%% c

N = size(X,1);
XB1 = 0;

for i = [1:c]
    this_cluster = (cluster1==i);
    Xi = X(this_cluster, :);
    mui_j = sort(sum((M1 - repmat(M1(i,:), c, 1)).^2, 2).^5);
    XB1 = XB1 + sum(sum((Xi - repmat(M1(i,:), size(Xi,1), 1)).^2,
2).^5) / mui_j(2);
end

XB1 = XB1 / N

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5
XB2 = 0;

for i = [1:c]
    this_cluster = (cluster2==i);
    Xi = X(this_cluster, :);
    mui_j = sort(sum((M2 - repmat(M2(i,:), c, 1)).^2, 2).^5);
    XB2 = XB2 + sum(sum((Xi - repmat(M2(i,:), size(Xi,1), 1)).^2,
2).^5) / mui_j(2);
end

XB2 = XB2 / N

```