REPORT- PROJECT01

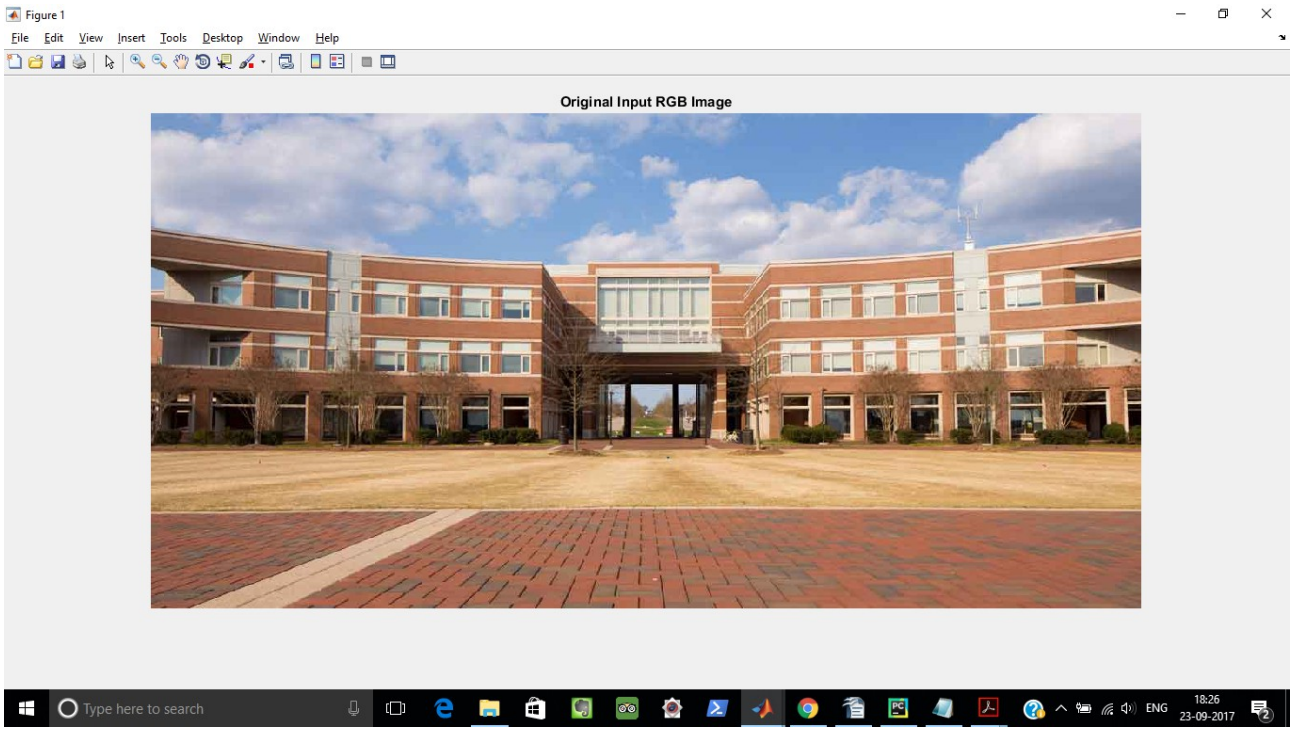
Team Members:
1. Kalyan Ghosh(unity_id:kghosh)
2. Bhargav(unity_id:bmysore)

Explanations:

1. Loading an Image:

1.1
In this project, we have used the image of NCSU Engineering Building campus and as shown below:

The original image is an RGB image with 3 color channels.



The image was loaded with below MATLAB code (**imread**):

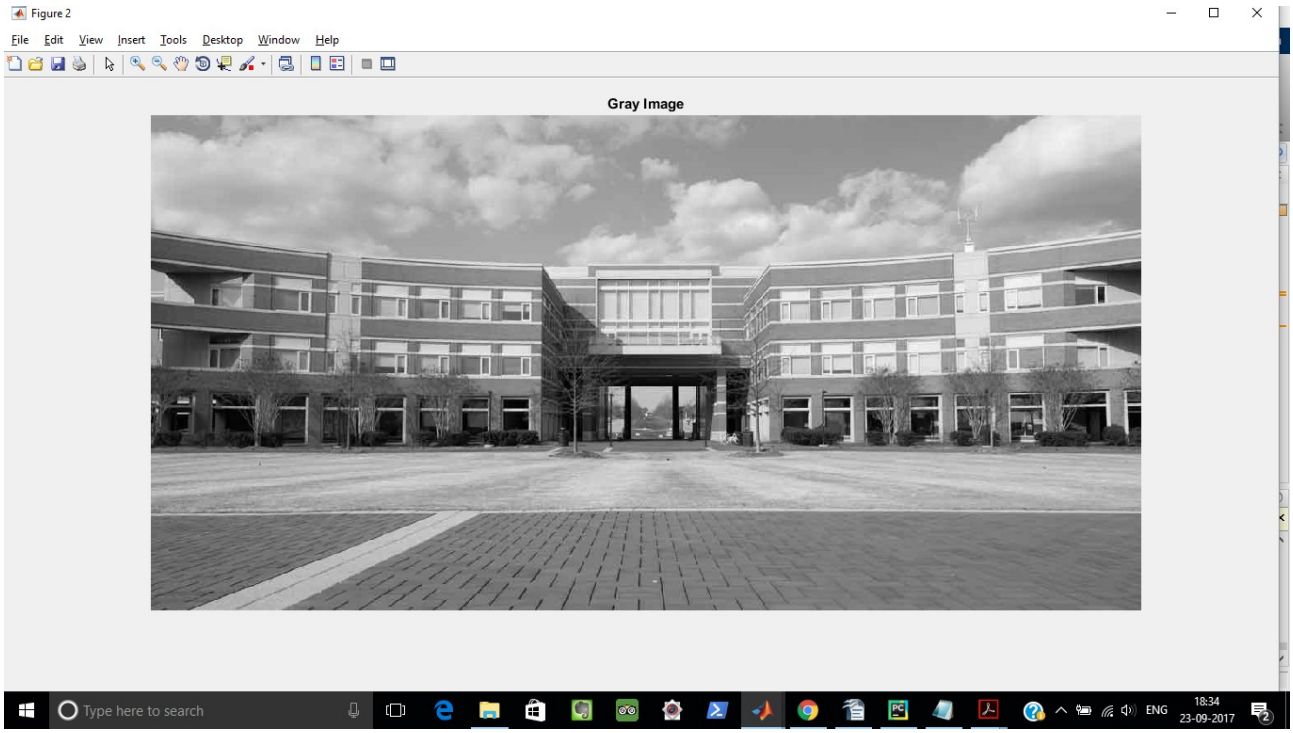
```
%1.Loading the image(View of the EB):
I=imread('ncsu.jpg');
figure(1),
imshow(I);
title('Original Input RGB Image');
```

1.2

The input RGB image is then converted into a gray scale image by using the below MATLAB code (**rgb2gray**):

```
%Converting to gray image from RGB:
rgbI=rgb2gray(I);
figure(2),
imshow(rgbI);
title('Gray Image');
```

The grayscale image is shown below:



1.3

Now ,we are resizing the image into appropriate dimensions for mathematical simplicity,where we are choosing min{M,N} to be a power of 2 and and max{M,N} is a multiple of that power of 2 . In our case we have taken M=512 and N=2X512.

We are resizing the rgb image using the below MATLAB code:

```
%Processinng the image to get the correct dimensions
%Getting the dimension of the gray image:
[M,N]=size(rgbI);

%Reshaping the image to dimensions M=512,N=2*512:
J=imresize(rgbI,[512,2*512]);
[m,n]=size(J);
```

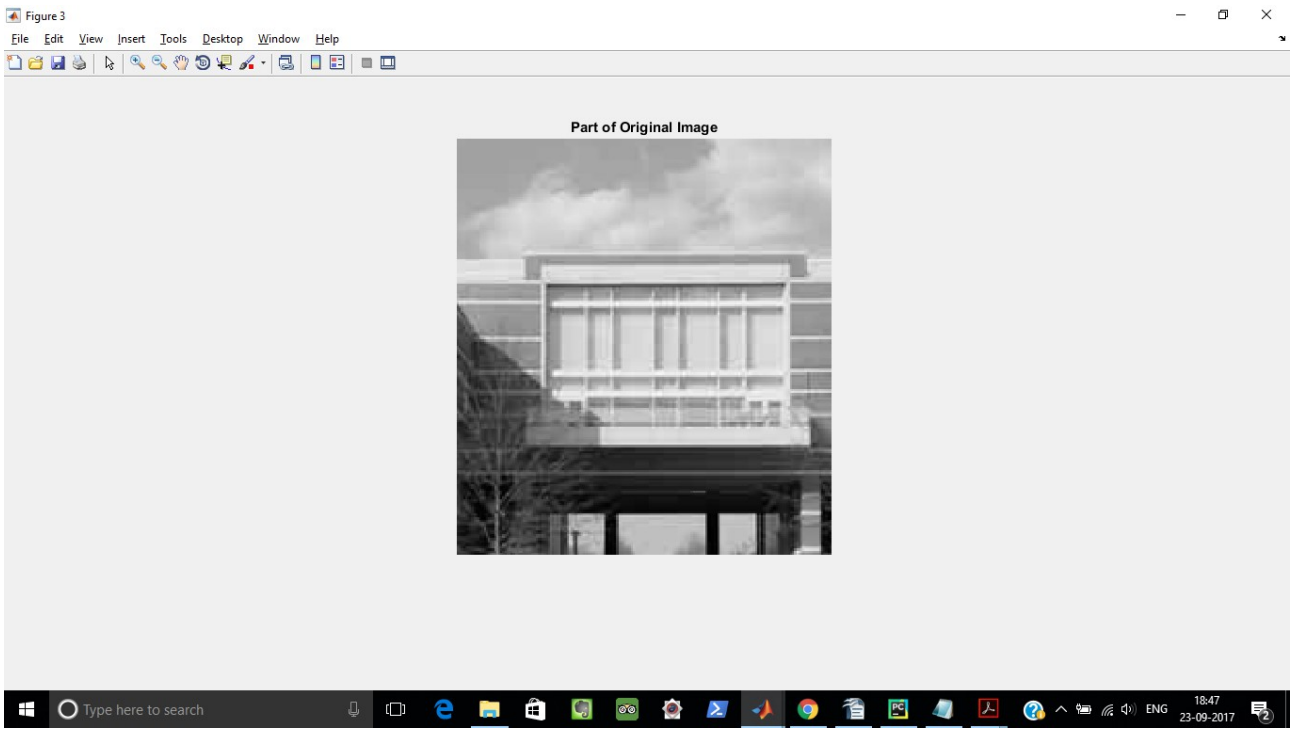
1.4

Now we select a subset of the M X N image that looks nice.This subset of the image will be compared with its version after Vector Quantization.

The subset of the image is selected using the below MATLAB code:

```
%Selecting a good subset of the image:
p=[100:300];
q=[420:600];

subset=J(p,q,:);
figure(3),
imshow(subset);
title('Part of Original Image');
```

The part of the original image is shown below:



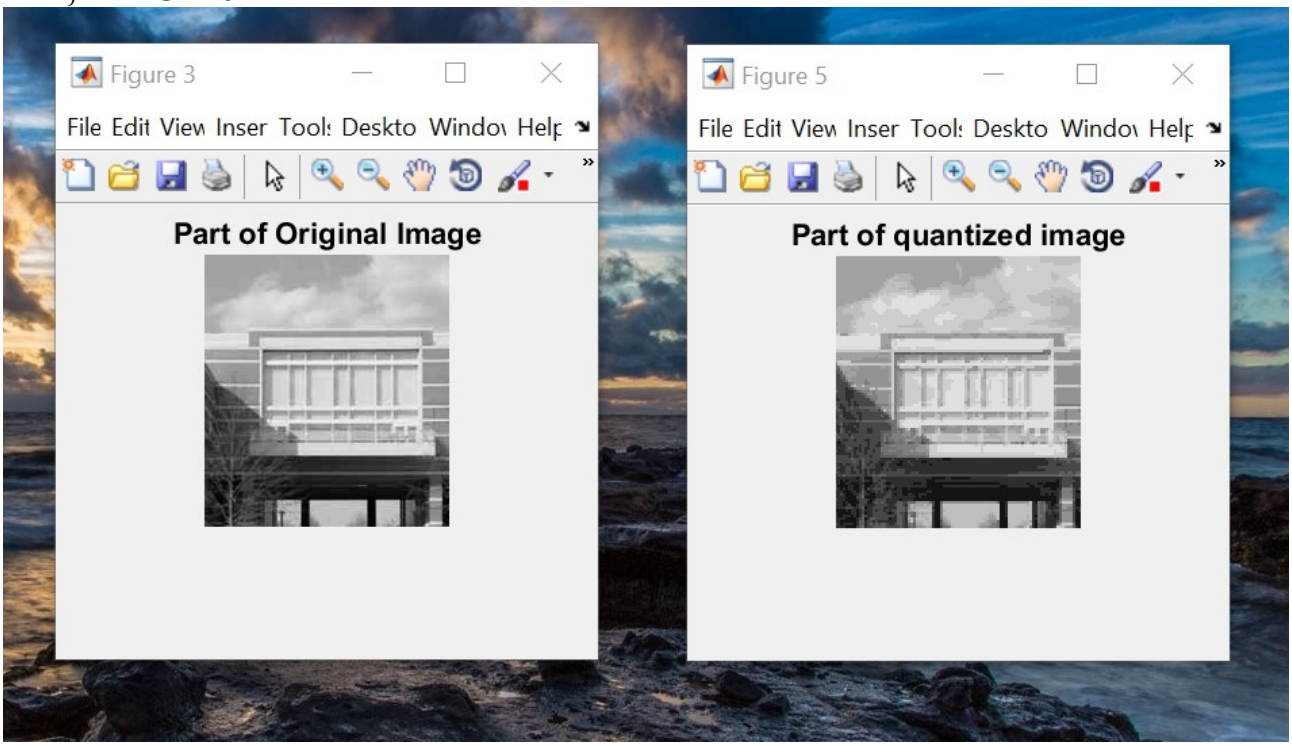
2. **Clustering and Vector Quantization:**

2.1

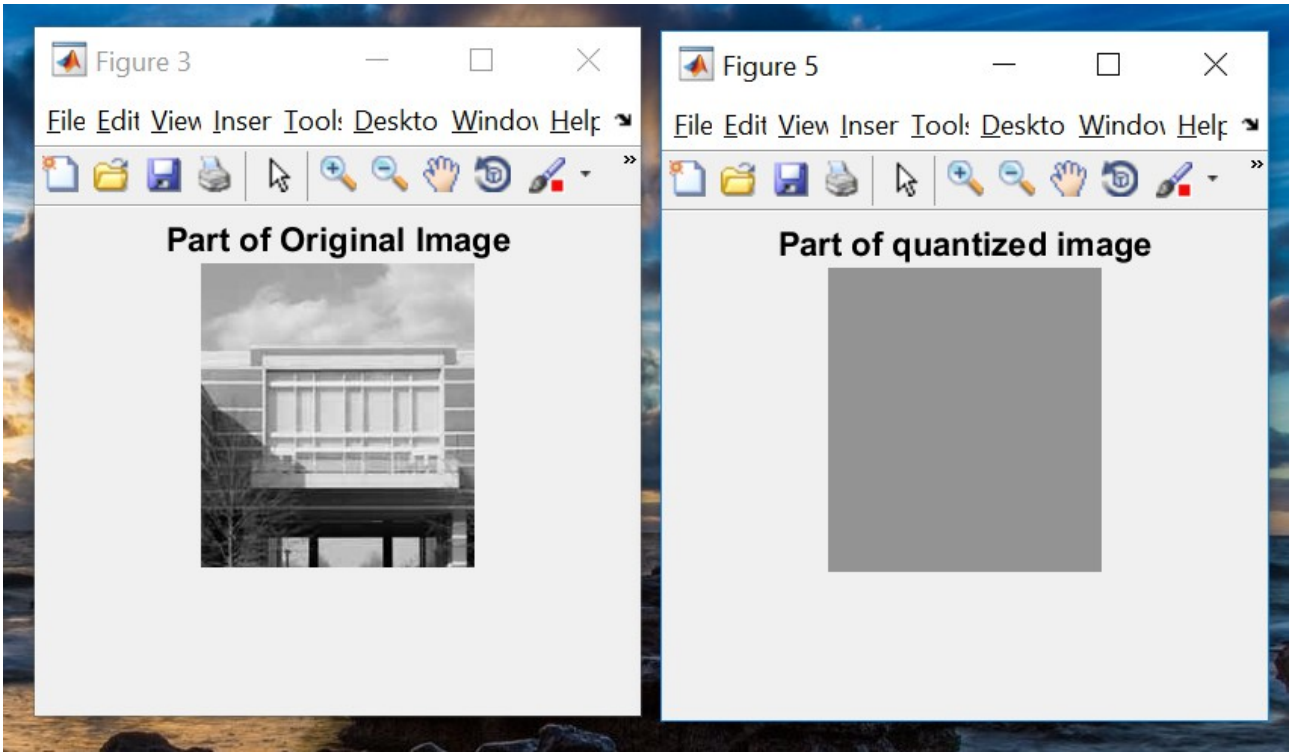
- Now after the image preprocessing is done in the first step, we define a "patch" in the image with dimensions PxP, where P=2. This value of will work well for the dimensions of the above M X N image since they are also powers of 2.
- In total,there would be (MxN)/P^2 patches. We then cluster these patches into 16 clusters using k-means clustering algorithm. We use the kmeans command in MATLAB for this.
- The value of the number of clusters (C) is chosen by choosing the value of R as 1, where R (rate) as 1. This means, each pixel is represented by RP^2 number of bits. Total number of clusters C is obtained using, $C = 2^{(RP^{\wedge}2)}$.
- When k-means algorithm is applied, each patch in the image would be assigned to a particular cluster. Replace each patch by the values of the cluster to which it belongs.
- We observe that as the number of patches decreases the quality of the compressed image deteriorates(at cluster=1, image is totally destroyed).
- We now observe that the whole image is now represented using only those values in the cluster. Hence, we will be able to represent the whole image in lesser number of bits than before.
- Thus, image compression has been achieved.

Below is the plot of part of the image and its quantized version:

P=2,R=1 C=16



When number of clusters=1



Code:

```
%Topics in Data Science Project1
%Project Name:Image Compression via Clustering
%Project Members:
%Kalyan Ghosh(kghosh)
%Bhargav Mysore(bmysore)
%1.Loading the image(View of the EB):
I=imread('ncsu.jpg');
%figure(1),
%imshow(I);
%title('Original Input RGB Image');

%Converting to gray image from RGB:
rgbI=rgb2gray(I);
%figure(2),
%imshow(rgbI);
%title('Gray Image');



%Processinng the image to get the correct dimensions
%Getting the dimension of the gray image:
[M,N]=size(rgbI);

%Reshaping the image to dimensions M=512,N=2*512:
J=imresize(rgbI,[512,2*512]);
[m,n]=size(J);


%Selecting a good subset of the image:
p=[100:300];
```

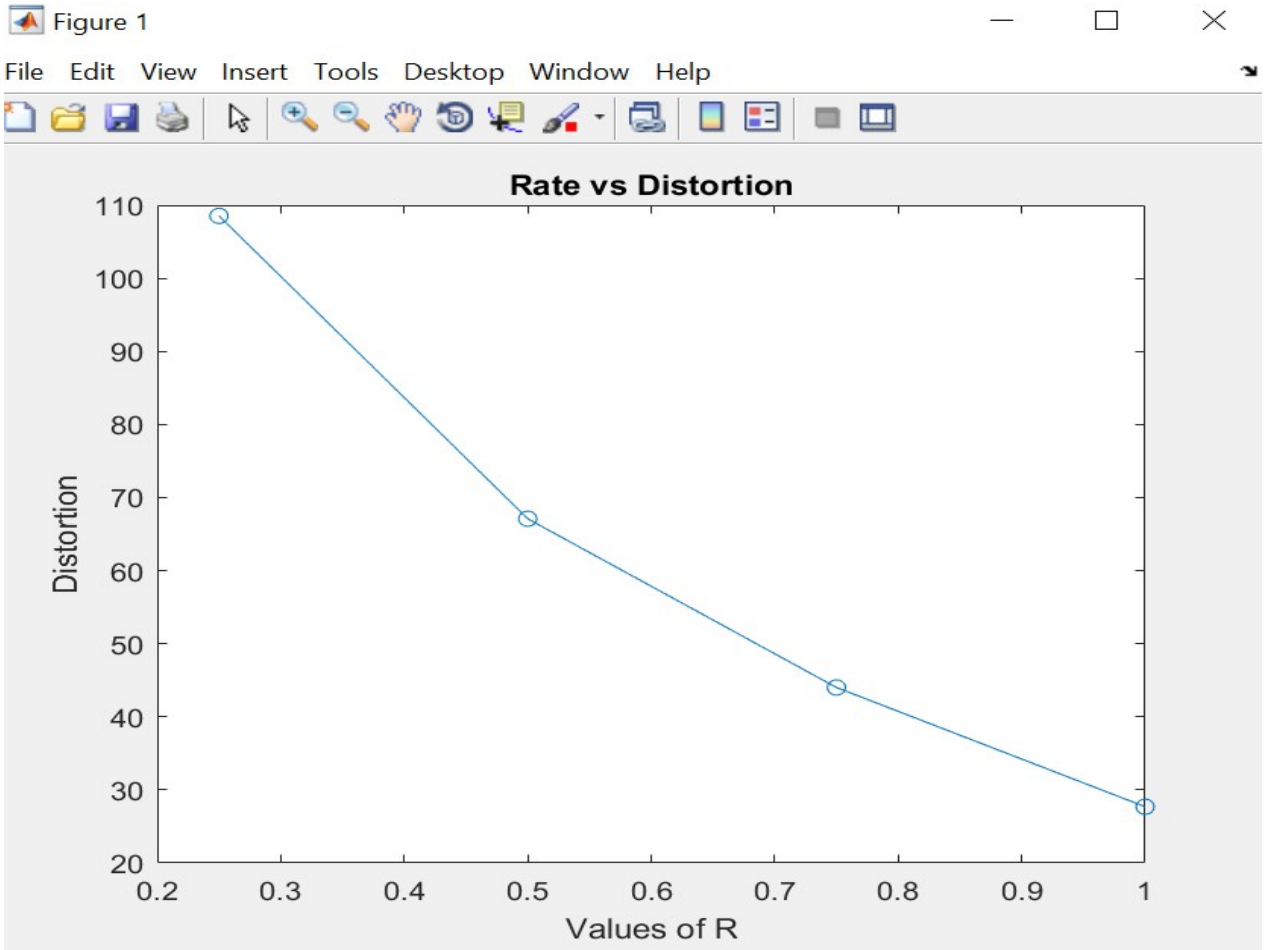
```
q=[420:600];

subset=J(p,q,:);
%figure(3),
%imshow(subset);
%title('Part of Original Image');

%Choosing the Patch Size,P=2,resulting in 4 pixels per patch:
hor_dim=2;
ver_dim=2;


%Initializing xnew parameterized matrix:
xnew=zeros(hor_dim*ver_dim,(m*n)/(hor_dim*ver_dim));
[a,b]=size(xnew);

k=1;
%Looping through the image and storing the patches in rows
%of xnew.Note:Each Row of xnew stores the pixel of a patch and there are
%MN/P^2 number of such patches
for i=1:hor_dim:m
    for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        xnew(:,k)=patch(:);
        k=k+1;
    end
end


xtranspose=xnew(:);

%Applying KMeans Clustering on the patches of the image

[idx,C]=kmeans(xtranspose,16);




%Now running Vector Quantization
%Replacing the patch with the cluster centroid to which it belongs

a=1;
for i=1:hor_dim:m
    for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        cluster_number=idx(a);
        cluster_mean=C(cluster_number);
        J(i:i+hor_dim-1,j:j+ver_dim-1)=cluster_mean;
        a=a+(hor_dim*ver_dim);
    end
end


figure(4),
imshow(J);
title('Vector Quantized whole image');

quantized_image=J(p,q,:);
figure(5),
imshow(quantized_image)
title('Part of quantized image');
```

3 Rate vs. Distortion

In this part, we vary the R(rate) value from 0 to 1. As we vary the R value, number of clusters C also changes because $C=2^{(R*P\hat{}2)}$. For each value of R (and thereby C) we do the clustering and the Vector Quantization again. For each of these R we calculate the distortion between the original image and the quantized image using the formula: $D(x,y)=(\frac{1}{MN}\sum_{M=1,N=1}^{M,N}(x-y)^2)$

Where M is the number of rows in the image, N is the number of columns in the image, x is the pixel value of original image and y is the pixel value of the quantized image.

P=2



- We observe that as we increase the rate R, the distortion is monotonically decreasing. This happens because as we increase R keeping patch size constant, the number of clusters C increase.
- Distortion is the measure of how close the quantized image is to the original image.
- As the more and more clusters are used to represent the same image, the reconstructed image goes closer and closer to the original image and hence the distortion is less.
- Basically we are using more number of values we are using to represent the same image, so the values will be more closer to the values of original image.

- The trade-off here is that as R increases the number of bits required to represent also increases.

Note: We need to choose R values such that $R*P^2$ should be an integer. Here we have taken R=[0.25,0.5,0.75,1.0]

Code:

```
%Topics in Data Science Project1
%Project Name:Image Compression via Clustering
%Project Members:
%Kalyan Ghosh(kghosh)
%Bhargav()
close all;
clear all;
clc;
%1.Loading the image(View of the EB):
I=imread('ncsu.jpg');
% figure(1),
% imshow(I);
% title('Original Input RGB Image');

%Converting to gray image from RGB:
rgbI=rgb2gray(I);
% figure(2),
% imshow(rgbI);
% title('Gray Image');



%Processinng the image to get the correct dimensions
%Getting the dimension of the gray image:
[M,N]=size(rgbI);

%Reshaping the image to dimensions M=512,N=2*512:
J=imresize(rgbI,[512,2*512]);
J1=imresize(rgbI,[512,2*512]);
[m,n]=size(J);


%Selecting a good subset of the image:
p=[100:300];
q=[420:600];

subset=J(p,q,:);
% figure(3),
% imshow(subset);
% title('Part of Original Image');

%Choosing the Patch Size,P=2,resulting in 4 pixels per patch:
hor_dim=2;
ver_dim=2;

%Initializing x_new parameterized matrix:
xnew=zeros(hor_dim*ver_dim,(m*n)/(hor_dim*ver_dim));
[a,b]=size(xnew);

k=1;
%Looping through the image and storing the patches in rows
%of xnew.Note:Each Row of xnew stores the pixel of a patche and there are
%N number of such patches
```

```
for i=1:hor_dim:m
    for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        xnew(:,k)=patch(:);
        k=k+1;
    end
end

xtranspose=xnew(:);

%Applying KMeans Clustering on the image patches
R=[0.25,0.5,0.75,1.0];
for g=1:1:length(R)
    T=2^(floor(R(g)*hor_dim*ver_dim));
    [idx,C]=kmeans(xtranspose,T);



    %Now running Vector Quantization
    %Replacing the patch with the cluster centroid to which it belongs

   a=1;
   for i=1:hor_dim:m
      for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        cluster_number=idx(a);
        cluster_mean=C(cluster_number);
        J(i:i+hor_dim-1,j:j+ver_dim-1)=cluster_mean;
        a=a+(hor_dim*ver_dim);
      end
  end
   err=J1-J;
   error(g)=((1/(m*n))*(sum(sum(err.^2))));
end

plot(R,error);
xlabel(' Values of R ');ylabel( ' Distortion');
title('Rate vs Distortion Plot for P=2');
```

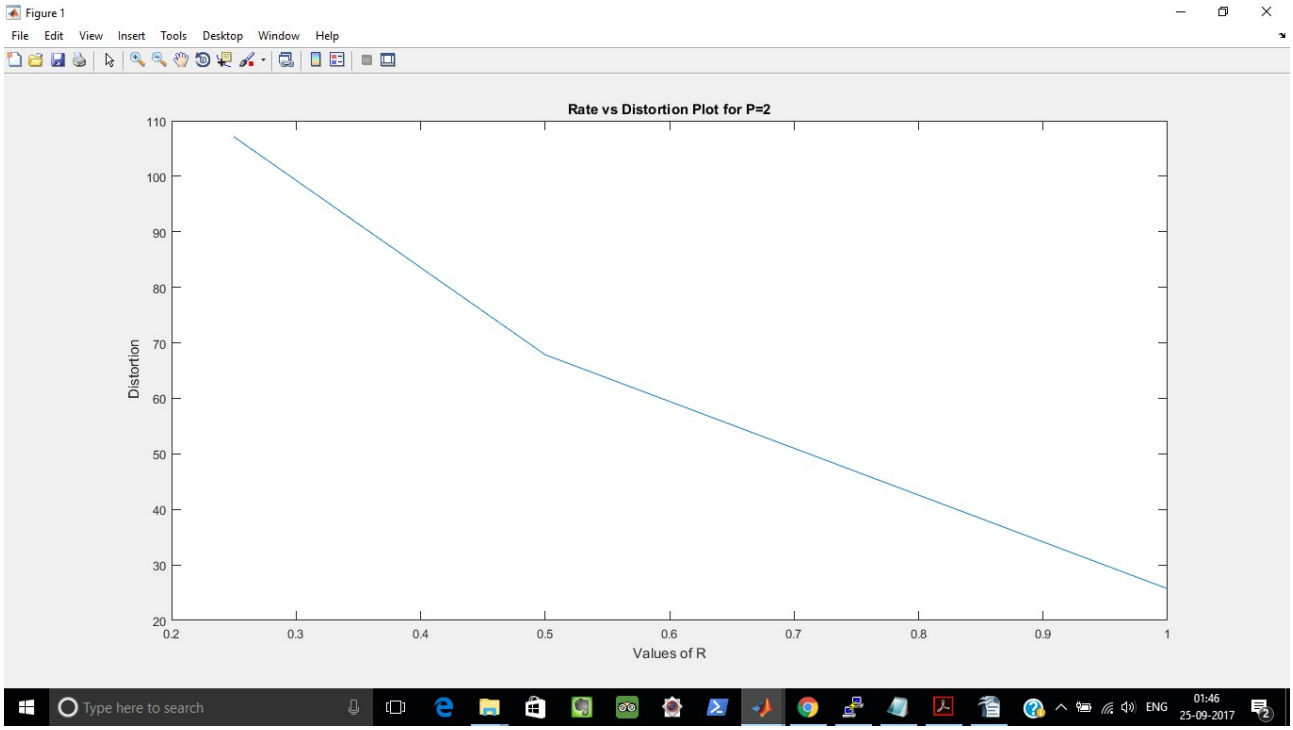
4.**Patch Size:**

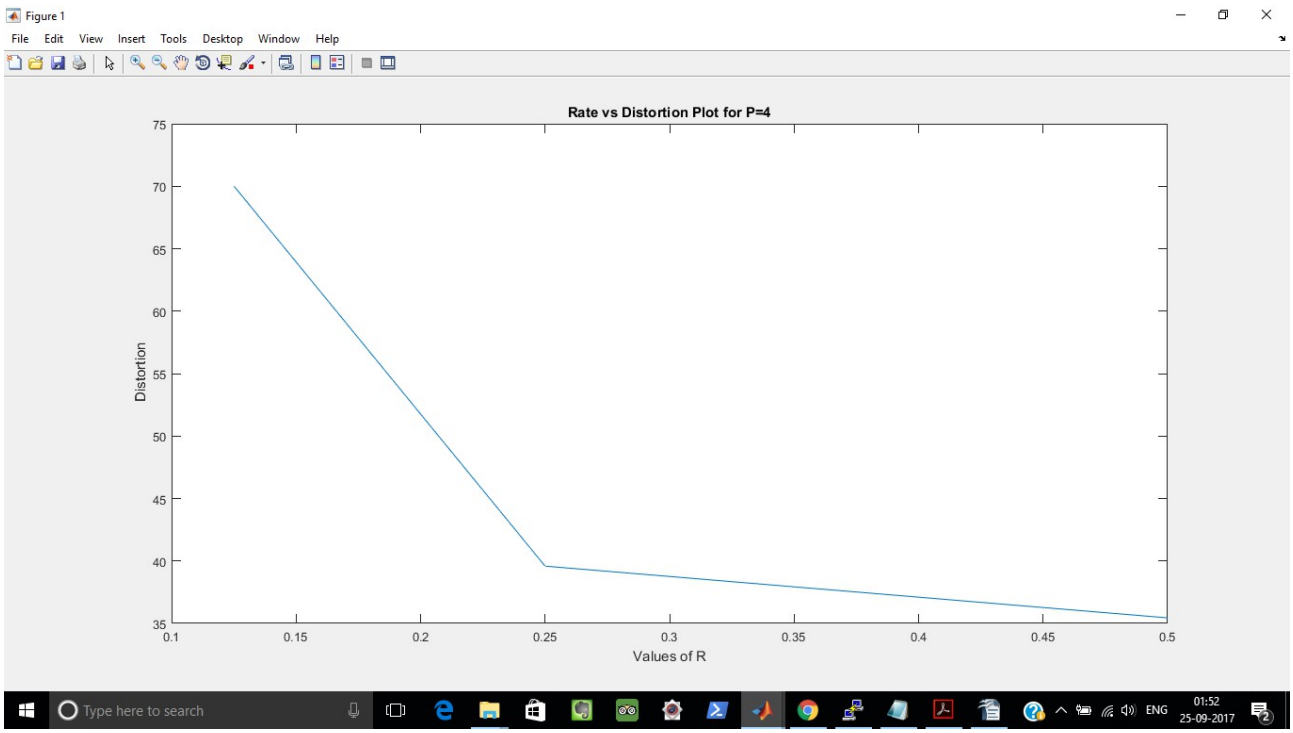
In this part, we vary the patch size of the patches.Earlier we had used P=2 to generate a 2x2 patch size.Now, in this part we use some value of P that is a powe of 2.For our example we have used P=4 to generate a patch size of 4x4 and plotted the Rate vs Distortion plot.

We now compare the Rate vs Distortion plot for P=2 and P=4 to see that if the RD tradeoff improved or degraded.

Plot 1: R=[0.25 0.5 1] ,P=2 and Plot 2 :R=[0.125 0.25 0.5] and P=4.

The values of R are so chosen that the value 2^k where(k=RP^2) returns integral values(Clusters).





Explanation:

We see, that with increasing values of R and P,the Distortion decreases.This can be explained by the below points:

- When P=2 and R is [0.25 0.5 1],the values of k=RP^2 are (1,2,4) and consequently the values of 2^k are (2,4,16) clusters
- When P=4 and R=[0.125 0.25 0.5],the values of k=RP^2 are(2,4,8) and consequently the values of 2^k are (4,16,256) clusters.

- So,we see that with increasing patch size in powers of 2,the value of 2^k i.e the numbers of clusters also increases.
- We know with increasing Patch Size,the distortion value goes up ,because we are approximating a larger portion of the image.But on the other hand,with increasing value of patch size,the cluster number also goes up and we know that increasing the number of clusters decreases the distortion because we are dividingthe image into more number of clusters and hence decreasing the level of approximation of the image.
- In the above example,for P=2 and R=0.25 and 0.5,the values of Distortion are:110 and 68 respectively and the cluster numbers are 2 and 4 respectively.Similarly, for P=4 and R=0.25 and R=0.5,the values of Distortion are 40 and 35 respectively and the cluster numbers are 16 and 256 respectively.
- It is evident from the above values that for P=4,the number of clusters is much larger that the number of clusters for P=2 for same value of R.So,even though P=4 is greater that P=2 which should have increased the distortion,we see that the distortion for P=4 is less than P=2 for same values of R.
- So,it is evident from the above discussion that,the number of clusters dominates the increase of patch size in reducing the distortion.

Code:

```
P=2

%Topics in Data Science Project1
%Project Name:Image Compression via Clustering
%Project Members:
%Kalyan Ghosh(kghosh)
%Bhargav(bmysore)
close all;
clear all;
clc;
%1.Loading the image(View of the EB):
I=imread('ncsu.jpg');
% figure(1),
% imshow(I);
% title('Original Input RGB Image');

%Converting to gray image from RGB:
rgbI=rgb2gray(I);
% figure(2),
% imshow(rgbI);
% title('Gray Image');



%Processinng the image to get the correct dimensions
%Getting the dimension of the gray image:
[M,N]=size(rgbI);

%Reshaping the image to dimensions M=512,N=2*512:
J=imresize(rgbI,[512,2*512]);
J1=imresize(rgbI,[512,2*512]);
[m,n]=size(J);


%Selecting a good subset of the image:
p=[100:300];
q=[420:600];

subset=J(p,q,:);
% figure(3),
% imshow(subset);
```

```
% title('Part of Original Image');

%Choosing the Patch Size,P=2,resulting in 4 pixels per patch:
hor_dim=2;
ver_dim=2;

%Initializing x_new parameterized matrix:
xnew=zeros(hor_dim*ver_dim,(m*n)/(hor_dim*ver_dim));
[a,b]=size(xnew);

k=1;
%Looping through the image and storing the patches in rows
%of xnew.Note:Each Row of xnew stores the pixel of a patche and there are
%N number of such patches
for i=1:hor_dim:m
    for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        xnew(:,k)=patch(:);
        k=k+1;
    end
end

xtranspose=xnew(:);
%disp(length(xtranspose))
%disp(xnew);
%Applying KMeans Clustering on the
R=[0.25,0.5,1];
for g=1:1:length(R)
    T=2^(floor(R(g)*hor_dim*ver_dim));
    [idx,C]=kmeans(xtranspose,T);



    %Now running Vector Quantization
    %Replacing the patch with the cluster centroid to which it belongs

   a=1;
   for i=1:hor_dim:m
      for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        cluster_number=idx(a);
        cluster_mean=C(cluster_number);
        J(i:i+hor_dim-1,j:j+ver_dim-1)=cluster_mean;
        a=a+(hor_dim*ver_dim);
      end
  end
   err=J1-J;
   error(g)=((1/(m*n))*(sum(sum(err.^2))));
end

plot(R,error);
xlabel(' Values of R ');ylabel( ' Distortion');
title('Rate vs Distortion Plot for P=4');
```

Code

P=4

```
%Topics in Data Science Project1
%Project Name:Image Compression via Clustering
%Project Members:
%Kalyan Ghosh(kghosh)
%Bhargav(bmysore)
close all;
clear all;
clc;
%1.Loading the image(View of the EB):
I=imread('ncsu.jpg');
% figure(1),
% imshow(I);
% title('Original Input RGB Image');

%Converting to gray image from RGB:
rgbI=rgb2gray(I);
% figure(2),
% imshow(rgbI);
% title('Gray Image');



%Processinng the image to get the correct dimensions
%Getting the dimension of the gray image:
[M,N]=size(rgbI);

%Reshaping the image to dimensions M=512,N=2*512:
J=imresize(rgbI,[512,2*512]);
J1=imresize(rgbI,[512,2*512]);
[m,n]=size(J);


%Selecting a good subset of the image:
p=[100:300];
q=[420:600];

subset=J(p,q,:);
% figure(3),
% imshow(subset);
% title('Part of Original Image');

%Choosing the Patch Size,P=2,resulting in 4 pixels per patch:
hor_dim=4;
ver_dim=4;

%Initializing x_new parameterized matrix:
xnew=zeros(hor_dim*ver_dim,(m*n)/(hor_dim*ver_dim));
[a,b]=size(xnew);

k=1;
%Looping through the image and storing the patches in rows
%of xnew.Note:Each Row of xnew stores the pixel of a patche and there are
%N number of such patches
for i=1:hor_dim:m
    for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        xnew(:,k)=patch(:);
        k=k+1;
    end
end

xtranspose=xnew(:);
```

```
%disp(length(xtranspose))
%disp(xnew);
%Applying KMeans Clustering on the
R=[0.125,0.25,0.5];
for g=1:1:length(R)
    T=2^(floor(R(g)*hor_dim*ver_dim));
    [idx,C]=kmeans(xtranspose,T);



    %Now running Vector Quantization
    %Replacing the patch with the cluster centroid to which it belongs

   a=1;
   for i=1:hor_dim:m
      for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        cluster_number=idx(a);
        cluster_mean=C(cluster_number);
        J(i:i+hor_dim-1,j:j+ver_dim-1)=cluster_mean;
        a=a+(hor_dim*ver_dim);
      end
  end
   err=J1-J;
   error(g)=((1/(m*n))*(sum(sum(err.^2))));
end

plot(R,error);
xlabel(' Values of R ');ylabel( ' Distortion');
title('Rate vs Distortion Plot for P=4');
```

5 Better Compression

Till now we have considered that all the clusters have equal probability of appearing in the image. However in reality, some clusters appear more often than the others. Therefore, in practical applications we usually represent the clusters which appear frequently with lesser number of bits. Information theory tells us that we can get an average coding length of $H=\sum_i pi\ log2(\frac{1}{pi})$ if we take into consideration that some symbols appear more often than others.
In this part we compare the average coding rate for uniform probability and the actual probability.

R	H	Actual R(H/P^2)
1.0	3.4893	0.8723
0.75	1.8284	0.4571
0.50	1.1798	0.2950
0.25	0.6984	0.1746

We have calculated H and R(actual coding rate) for different values of R. We see from the first entry that we could save 1-0.8723=0.1277 bits per patch if we had not assigned equal coding length for every patch. Also the average number of bits required for each patch is 3.4893 instead of 4. Hence, in practical applications we observe that always the frequency of appearance of one symbol will be more than others. We can decrease the coding length by representing it using lesser number of bits. We assign more bits for representing symbols which appear less frequently.

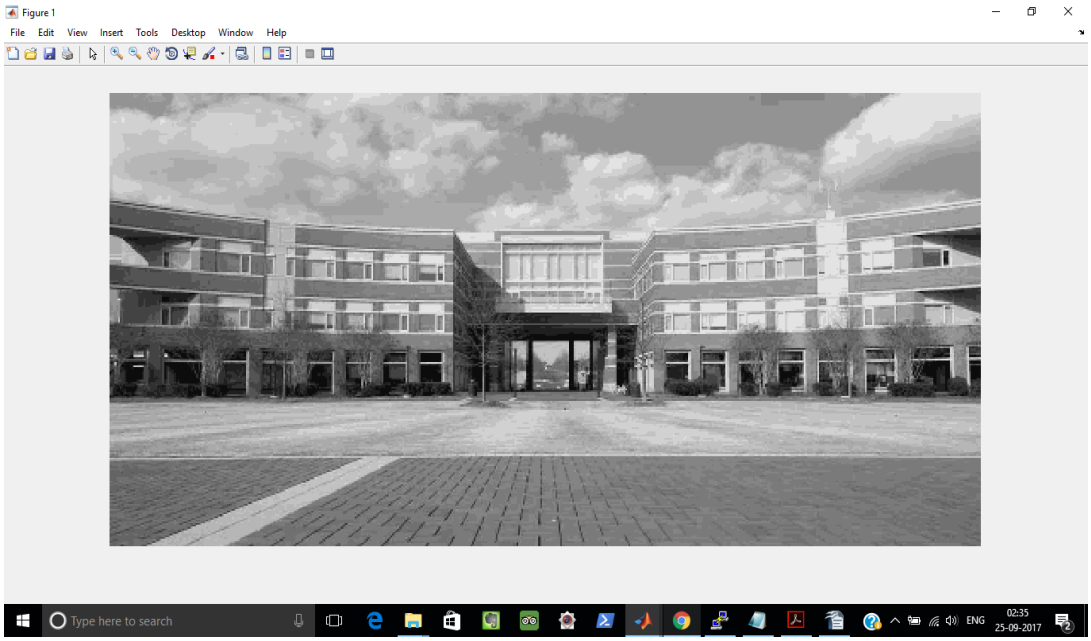
6.Extra credit part:

Explanation:

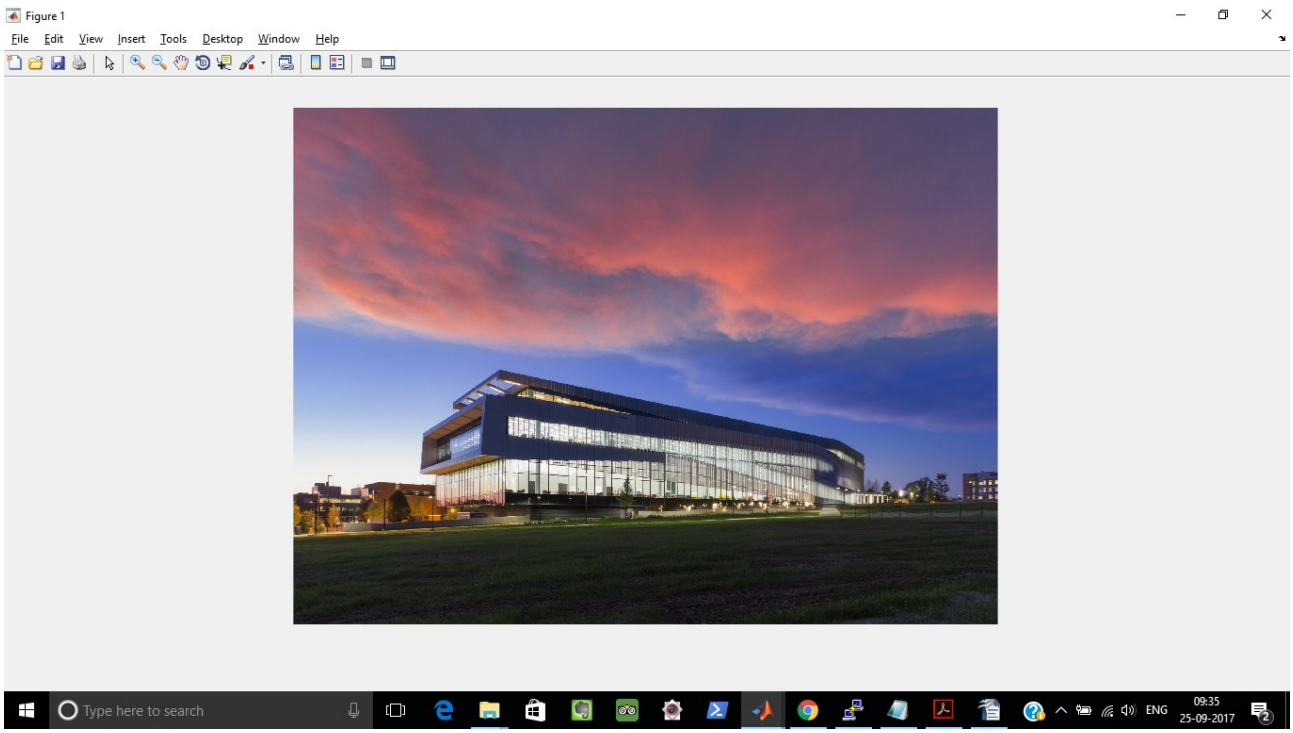
Here we run the k means clustering algorithm on a training image,collect the cluster means and then test it on a testing image.

Our findings can be discussed in the below points.

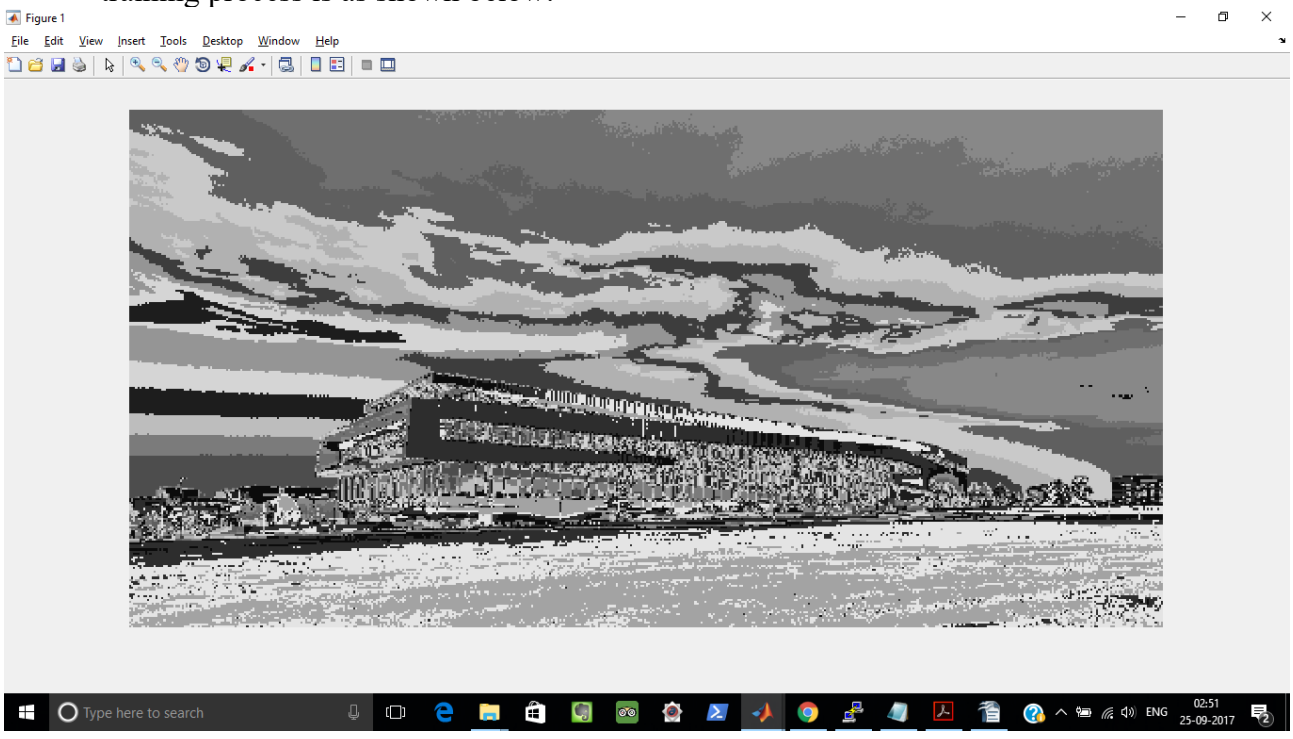
- Here ,first of all we run the k means clustering algorithm for P=2,R=1 and k=16 on our training image.
- After runnnig the kmeans .we store the 16 cluster centroids.
- The quantized version of the training image is shown as below.



- The quantized version of the training image is shown as below.
- We find that error is (Error of Training Image for R=1 and P=2 and k=16 is =**25.431574**
- We then take a new test image.The original version of the test image is shown as below:



- We then run k means clustering on the test image with R=1,patch size P=2(same as in training process) and k=16(same as in training process) and replace the representation patch with values as received in the training process.
- The quantized version of the test image using representation patch values obtained from the training process is as shown below:



- The error in the test image(Error of Testing Image for R=1 and P=2 and k=16 is =**158.945526**)
- We see that the Testing error is much larger than the Training error which is expected because,each image has its own randomness in pixel values which can be best represented or approximated if the patches are represented with the representation patches derived by

running k means algorithm on the same test image.However the extent of testing error can be reduced by employing techniques like Cross Validation etc which can lead to better generalization.

Code:

```
%Topics in Data Science Project1
%Project Name:Image Compression via Clustering
%Project Members:
%Kalyan Ghosh(kghosh)
%Bhargav(bmysore)
close all;

clc;
%1.Loading the image(View of the EB):
I=imread('ncsu.jpg');
% figure(1),
% imshow(I);
% title('Original Input RGB Image');

%Converting to gray image from RGB:
rgbI=rgb2gray(I);
% figure(2),
% imshow(rgbI);
% title('Gray Image');



%Processinng the image to get the correct dimensions
%Getting the dimension of the gray image:
[M,N]=size(rgbI);

%Reshaping the image to dimensions M=512,N=2*512:
J=imresize(rgbI,[512,2*512]);
J1=imresize(rgbI,[512,2*512]);
[m,n]=size(J);


%Selecting a good subset of the image:
p=[100:300];
q=[420:600];

subset=J(p,q,:);
% figure(3),
% imshow(subset);
% title('Part of Original Image');

%Choosing the Patch Size,P=2,resulting in 4 pixels per patch:
hor_dim=2;
ver_dim=2;

%Initializing x_new parameterized matrix:
xnew=zeros(hor_dim*ver_dim,(m*n)/(hor_dim*ver_dim));
[a,b]=size(xnew);

k=1;
%Looping through the image and storing the patches in rows
%of xnew.Note:Each Row of xnew stores the pixel of a patche and there are
%N number of such patches
for i=1:hor_dim:m
    for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        xnew(:,k)=patch(:);
        k=k+1;
```

```
    end
end

xtranspose=xnew(:);
%disp(length(xtranspose))
%disp(xnew);
%Applying KMeans Clustering on the



    [idx,C]=kmeans(xtranspose,16);
    codebook=C(:);


    %Now running Vector Quantization
    %Replacing the patch with the cluster centroid to which it belongs

   a=1;
   for i=1:hor_dim:m
      for j=1:ver_dim:n
        patch=J(i:i+hor_dim-1,j:j+ver_dim-1);
        cluster_number=idx(a);
        cluster_mean=C(cluster_number);
        J(i:i+hor_dim-1,j:j+ver_dim-1)=cluster_mean;
        a=a+(hor_dim*ver_dim);
      end
  end
   err=J1-J;
   error=((1/(m*n))*(sum(sum(err.^2))));
   fprintf('Error of Training Data for R=1 and P=2 is =%f \n',error)
   imshow(J)

   %**********************************************************************

I2=imread('hunt.png');
rgbI2=rgb2gray(I2);
J2=imresize(rgbI2,[512,2*512]);
J2_temp=imresize(rgbI2,[512,2*512]);
[m1,n1]=size(J2);

p1=[100:300];
q1=[420:600];

xnew1=zeros(hor_dim*ver_dim,(m*n)/(hor_dim*ver_dim));
[a1,b1]=size(xnew1);

%Running K Means on the new image and replacing the Cluster Means with the
%entries we have from the training data


k=1;
%Looping through the image and storing the patches in rows
%of xnew.Note:Each Row of xnew stores the pixel of a patch and there are
%MN/P^2 number of such patches
for i=1:hor_dim:m1
    for j=1:ver_dim:n1
        patch=J2(i:i+hor_dim-1,j:j+ver_dim-1);
        xnew1(:,k)=patch(:);
        k=k+1;
    end
end

%disp(xnew)
xtranspose1=xnew1(:);
```

```
%disp(length(xtranspose1))

[idx1]=kmeans(xtranspose1,16);

a=1;
for i=1:hor_dim:m
    for j=1:ver_dim:n
        patch=J2(i:i+hor_dim-1,j:j+ver_dim-1);
        cluster_number=idx1(a);
        cluster_mean=codebook(cluster_number);
        J2(i:i+hor_dim-1,j:j+ver_dim-1)=cluster_mean;
        a=a+(hor_dim*ver_dim);
    end
end
err1=J2-J2_temp;
   error=((1/(m*n))*(sum(sum(err1.^2))));
   fprintf('Error of Test Data for R=1 and P=2 is =%f \n',error)
imshow(J2)
```