IMAGE PROCESSING ASSIGNMENT -I

Submitted by
Shabeeb Ali O
M.Tech. Signal Processing
Semester -I
No.15

October 2, 2017

Contents

0.1 Questions

0.1.1 Question 1:

The folder contains 4 sub folders containing classes of images and one test folder. Determine the Euclidean distance between pair of images in which one is from the test folder and the other is from other folders. Using a suitable strategy determine the class membership of each image in the test folder. Evaluate your methodology and results.

0.1.2 Question 2:

Determine the classmembership of each image in the test folder using Mahalanobis distance and considering each class of images belong to a Gaussian distribution. Evaluate your metodology and results. The report must contain detailed procedure, theory, methodology, observations, results comparisons

0.2 Theory

0.2.1 Euclidean Distance

In our problem there are four classes are given classes namely mountain , opencountry, street and tallbuilding. Each one of them are having 200+ colour mages belonging to them in their folder names.Also we are given a folder named test which contains 8 colour images in which each 2 of them are from a class.

The features are extracted for every image in each class and they are combined to form a feature matrix for each class. A feature vector is also calculated for test case image. The Euclidean image is found by finding distance between each image vector in a class to the test image vector. Since our test image itself is included in its own class we will get a distance of zero. So by getting zero for Euclidena distance we can easily classify the test image.

Euclidean distance calculation: A central problem in image recognition and computer vision is determining the distance between images. All the M by N images are easily discussed in an MN dimensional Euclidean space, called image space.
An image $x = (x_1, x_2, x_3 \cdot \cdots \cdot x_{MN})$, where x_{kN+1} is the gray level at the $(k, l)^{th}$ pixel, is represented as a point in the image space.
The squared Euclidean distance of two vectors x,y can be found as

$$\boldsymbol{d_E^2(x, y) = \sum_{k=1}^{MN} (x^k - y^k)^2}$$

Taking the square root of $\boldsymbol{d_E^2}$ we get the Euclidean distance.

0.2.2 Mahalanobis Distance

Here there are four classes namely mountain , opencountry, street and tallbuilding. Each one of them are having 200+ colour mages belonging to them in their folder names. Also we are given a folder named test which contains 8 colour images in which each 2 of them are from a class.

Each classes are considered as distribution of 80 feature vectors. In order to build a feature matrix which represents the class, 80 features of each image in the class are calculated and combine them row-wise to form a matrix having 80 columns. If there are 300 images in a class, we get a matrix of size 300 x 80. So we are calculating Mahalanobis distnace in 80-dimensional space.

Mahalanois distance Calculation: The Mahalanobis distance of an observation $\boldsymbol{x = (x_1, x_2, x_3 \cdot \cdots \cdot, x_N)^T}$ from a set of observations with mean $\boldsymbol{\mu = (\mu_1, \mu_2, \mu_3 \cdot \cdots \cdot, \mu_N)^T}$ and covariance matrix $\boldsymbol{\Sigma}$ is defined as:

$$\boldsymbol{D_M(x) = \sqrt{(x - \mu)^T \Sigma^{-1} (x - \mu)}}$$

Mahalanobis distance (or "generalized squared inter point distance" for its squared value) can also be defined as a dissimilarity measure between two random vectors and of the same distribution with the covariance matrix $\boldsymbol{\Sigma}$:

$$\boldsymbol{d(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})^T \Sigma^{-1} (\vec{x}, \vec{y})}}$$

If the covariance matrix is the identity matrix, the Mahalanobis distance reduces to the Euclidean distance. If the covariance matrix is diagonal, then the resulting distance measure is called a *normalized Euclidean distance*:

$$\boldsymbol{d(\vec{x},\vec{y}) = \sqrt{\sum_{i=1}^{N}\frac{(x_i-y_i)^2}{s_i^2}}}$$

where $\boldsymbol{s_i}$ is the standard deviation of the $\boldsymbol{x_i}$ and $\boldsymbol{y_i}$ over the sample set.

Matlab Functions : The covariance matrix can be calculated by using $\boldsymbol{cov(X)}$ function.
The inverse of the covariance matrix can calculated by $\boldsymbol{pinv(X)}$ function which gives pseudo-inverse of the given matrix **X**

0.2.3 Confusion Matrix

In the field of machine learning and specifically the problem of statistical classification, a confusion matrix, also known as an error matrix,[4] is a specific table layout that allows visualization of the performance of an algorithm
Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class (or vice versa).[2] The name stems from the fact that it makes it easy to see if the system is confusing two classes (i.e. commonly mislabelling one as another).

Example:

If a classification system has been trained to distinguish between class 1, class 2 and class 3, a confusion matrix will summarize the results of testing the algorithm for further inspection. Assuming a sample of 27 test cases 8 cases belongs to class 1, 6 cases belongs to class 2 , and 13 cases belongs to class 3, the resulting confusion matrix could look like the table below:

	Class 1	Class 2	Class 3
Class 1	5	2	0
Class 2	3	3	2
Class 3	0	1	11

In this confusion matrix, of the 8 actual cases that belongs to class 1 , the system predicted that three were belongs to class 2, and of the six actual cases that belongs to class 2, it predicted that three were belongs to class 1, three were belongs to class 2 and two were belongs to class 3. We can see from the matrix that the system in question has trouble distinguishing between class 1 and class 2. All correct predictions are located in the diagonal of the table, so it is easy to visually inspect the table for prediction errors, as they will be represented by values outside the diagonal.

Classification Accuracy:

$$\text{classification accuracy } (\%) = \frac{\text{sum of all diagonal elements of confusion matrix}}{\text{sum of all elements of confusion matrix}} \times \mathbf{100}$$

0.2.4 Feature Extraction:

Features simply represents some information relative to an image, or a local ROI inside the image.In the field of images, features might be raw pixels for simple problems like digit recognition of well-known dataset. However, in natural images, usage of simple image pixels are not descriptive enough. Instead there are two main stream to follow. One is to use hand engineered feature extraction methods (e.g. SIFT, VLAD, HOG, GIST, LBP) and the another stream is to learn features that are discriminative in the given context (i.e. Sparse Coding, Auto Encoders, Restricted Boltzmann Machines, PCA, ICA, K-means).
Features are extracted in order to differentiate between the images. Extracted features represents some useful information contained in the image. Since features extracted are normally less in number compared to the total pixel count of an image, image classification algorithm can work faster on features rather than operating on raw pixels.

In this assignment, 2-Dimensional Fourier Transform method is used for feature extraction. In MATLAB 'fft2' method is used for this operation which computes 2-Dimensional DFT using Fast Fourier Algorithm (FFT). 'fft2' method gives the output with same size of the image. Among the DFT coefficients strongest 80 values are selected and combined to form a $\mathbf{1 \times 80}$ vector for each image. Before computing 2-D DFT all images are scaled to $\mathbf{256 \times 256}$ and converted to grayscale for computaional simplicity.

A brief explanation of 2–D Fourier Transform: The Fourier Transform is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image.

The Fourier Transform is used in a wide range of applications, such as image analysis, image filtering, image reconstruction and image compression.

As we are only concerned with digital images, we will restrict this discussion to the Discrete Fourier Transform (DFT).

The DFT is the sampled Fourier Transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. The number of frequencies corresponds to the number of pixels in the spatial domain image, i.e. the image in the spatial and Fourier domain are of the same size.

For a square image of size $\boldsymbol{N \times M}$, the two-dimensional DFT is given by:

$$\boldsymbol{F(k,l) = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} f(i,j) e^{-2\pi(\frac{ki}{N}+\frac{lj}{M})}}$$

0.3 Procedure

0.3.1 Euclidean Distance:

1. Select a class from given classes of images.
2. Read images from the selected class and collect the pixel values in matrix so that each row represents each image of the class.
3. Repeat Step 1 and Step 2 until matrices for all classes are built.
4. Take each matrix and extract feature vector for each row which represents the image of the class.
5. Collect all feature vectors of the class and form a matrix.(*class feature matrix*)
6. Repeat Step 5 for all classes.
7. Select an image from test folder and extract its feature vector.
8. Calculate the euclidean distance from test image feature vector and each row of the *class feature matrix*
9. Repeat Step 8 for all classes.
10. Determine minimum value from calculated euclidean distances.
11. The class which belongs to minimum euclidean distance with test image determines the class of the test image

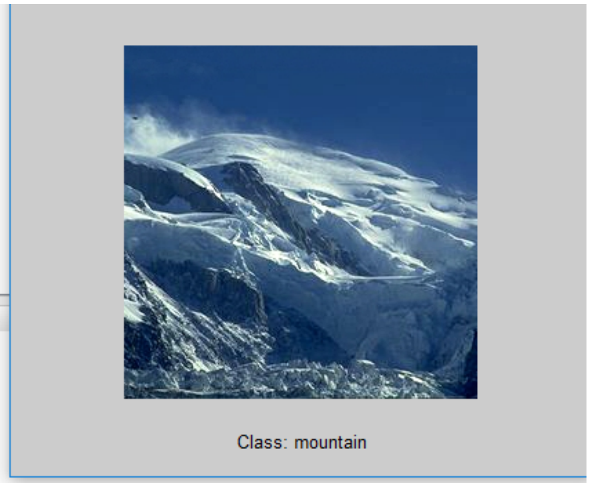
0.3.2 Mahalanobis Distance:

1. Select a class from given classes of images
2. Read images from the selected class and collect the pixel values in matrix so that each row represents each image of the class.
3. Repeat Step 1 and Step 2 until matrices for all classes are built.
4. Take each matrix and extract feature vector for each row which represents the image of the class.
5. Collect all feature vectors of the class and form a matrix.(*class feature matrix*)
6. Repeat Step 5 for all classes.
7. Select an image from test folder and extract its feature vector.
8. Calculate mahalanobis distance between test image feature vector and each *class feature matrix.*
9. Find the minimum value among the mahalanobis distances calculate from above step.
10. Class which belongs to the minimum mahalanobis distance determines the class of the test image.

0.4 Observations

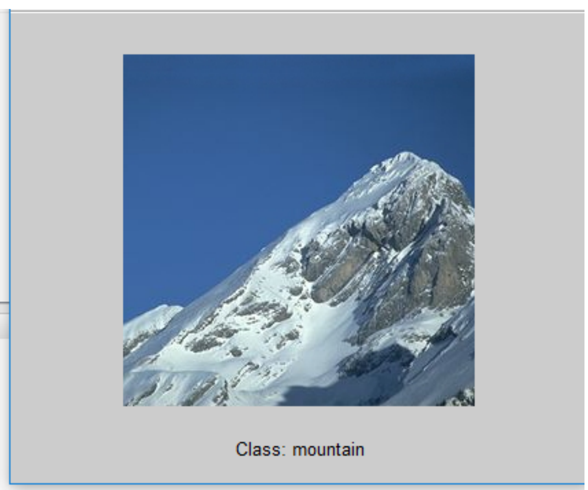
0.4.1 Question 1 (Euclidean Distance):

Output for test image 1



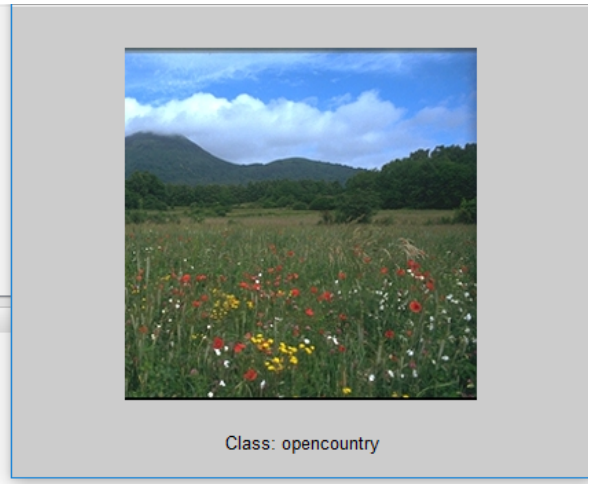

```
Command Window
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested mountain_land172.jpg
Minimum distance from collection of Euclidean distances with images of mountain class: 0
Minimum distance from collection of Euclidean distances with images of opencountry class: 442473.3569
Minimum distance from collection of Euclidean distances with images of street class: 494550.4682
Minimum distance from collection of Euclidean distances with images of tallbuilding class: 371993.2294
mountain_land172.jpg belongs to mountain class
>>
```

Output for test image 2



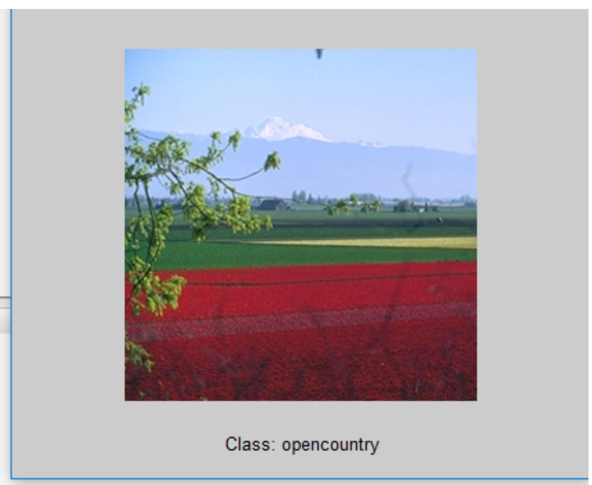

```
Command Window
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested mountain_land179.jpg
Minimum distance from collection of Euclidean distances with images of mountain class: 0
Minimum distance from collection of Euclidean distances with images of opencountry class: 456653.5453
Minimum distance from collection of Euclidean distances with images of street class: 301824.1935
Minimum distance from collection of Euclidean distances with images of tallbuilding class: 493542.157
mountain_land179.jpg belongs to mountain class
>>
```

Output for test image 3



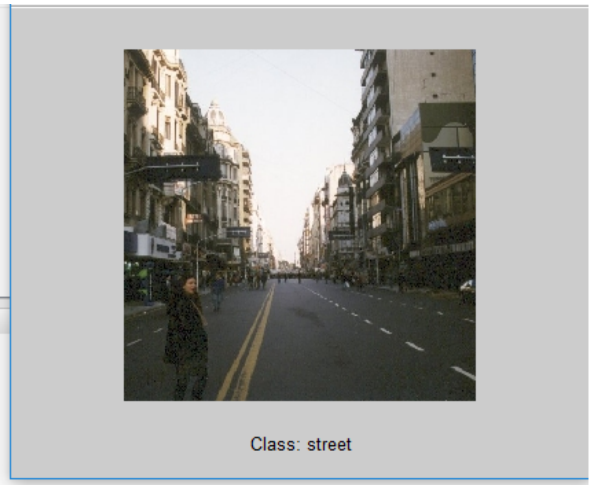

```
Command Window
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested opencountry_fie28.jpg
Minimum distance from collection of Euclidean distances with images of mountain class: 583989.049
Minimum distance from collection of Euclidean distances with images of opencountry class: 0
Minimum distance from collection of Euclidean distances with images of street class: 664761.9093
Minimum distance from collection of Euclidean distances with images of tallbuilding class: 682215.1611
opencountry_fie28.jpg belongs to opencountry class
fx >>
```

Output for test image 4




```
Command Window
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested opencountry_fie30.jpg
Minimum distance from collection of Euclidean distances with images of mountain class: 579023.9619
Minimum distance from collection of Euclidean distances with images of opencountry class: 0
Minimum distance from collection of Euclidean distances with images of street class: 418858.7129
Minimum distance from collection of Euclidean distances with images of tallbuilding class: 1218086.2208
opencountry_fie30.jpg belongs to opencountry class
fx >>
```

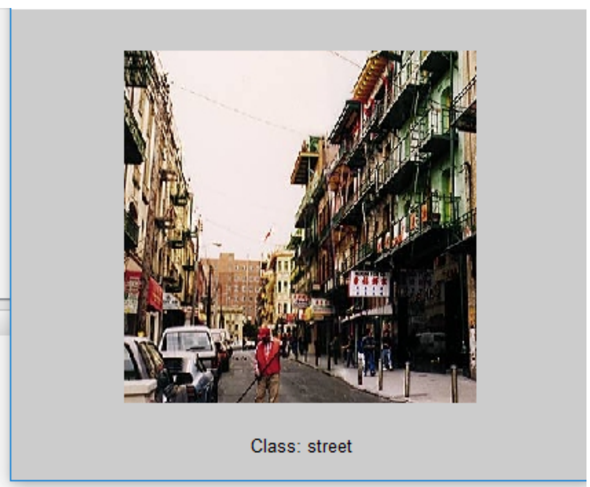
Output for test image 5



Command Window

```
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested street_art779.jpg
Minimum distance from collection of Euclidean distances with images of mountain class: 620592.5788
Minimum distance from collection of Euclidean distances with images of opencountry class: 820662.1581
Minimum distance from collection of Euclidean distances with images of street class: 0
Minimum distance from collection of Euclidean distances with images of tallbuilding class: 597979.5248
street_art779.jpg belongs to street class
>>
```

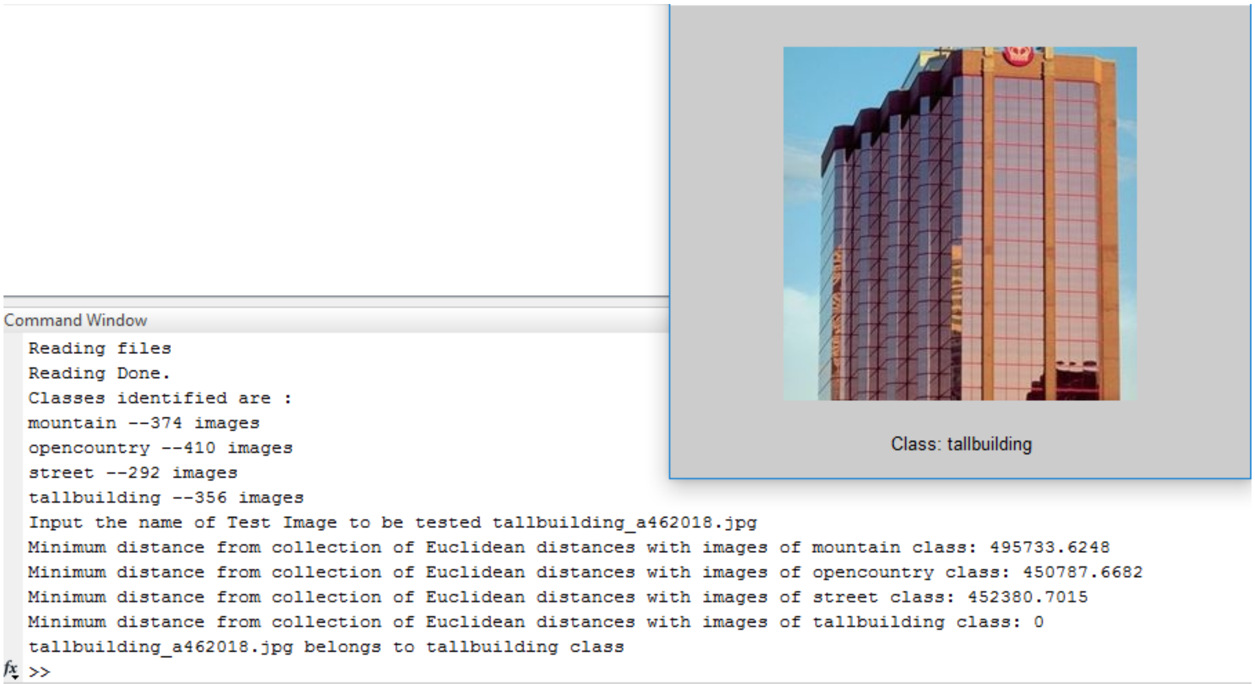
Output for test image 6



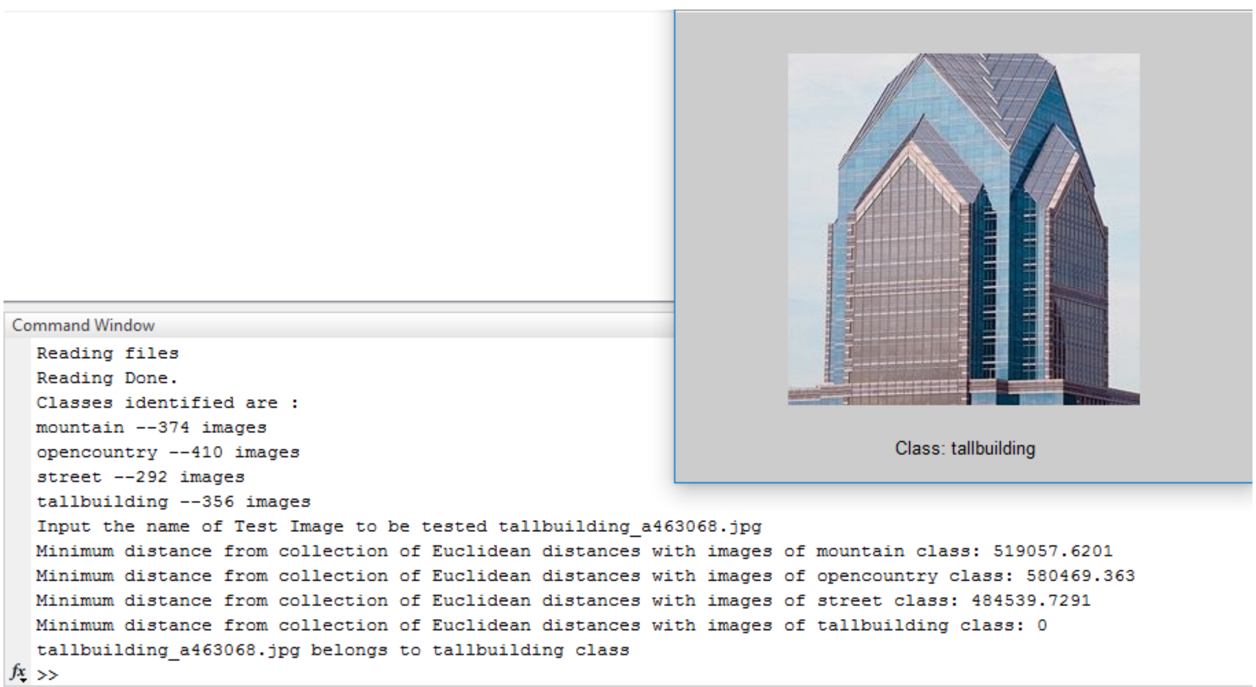
Command Window

```
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested street_art793.jpg
Minimum distance from collection of Euclidean distances with images of mountain class: 711691.7016
Minimum distance from collection of Euclidean distances with images of opencountry class: 880037.9849
Minimum distance from collection of Euclidean distances with images of street class: 0
Minimum distance from collection of Euclidean distances with images of tallbuilding class: 705382.9229
street_art793.jpg belongs to street class
>>
```

Output for test image 7

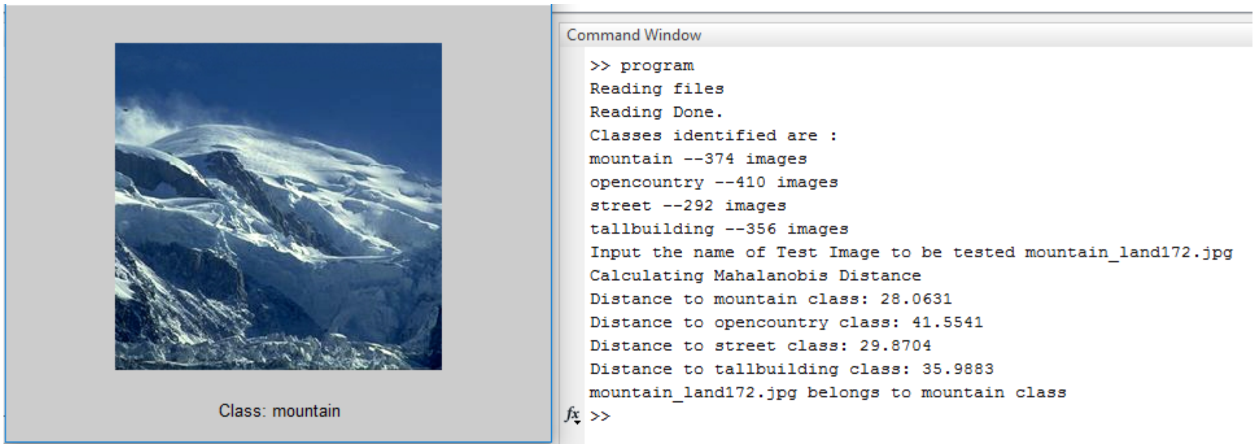


Output for test image 8

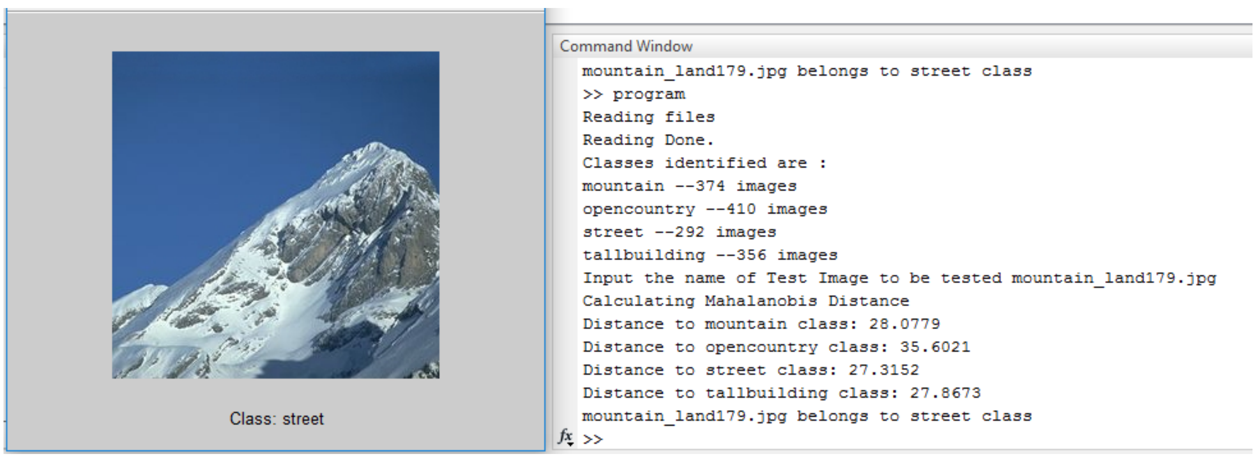


0.4.2 Question 2 (Mahalanobis Distance):

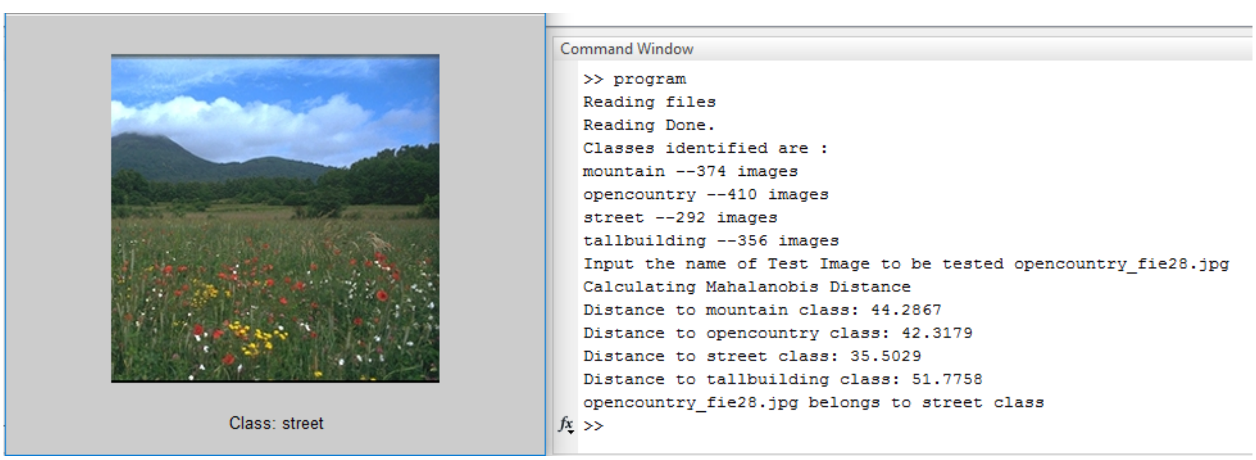
Output for test image 1



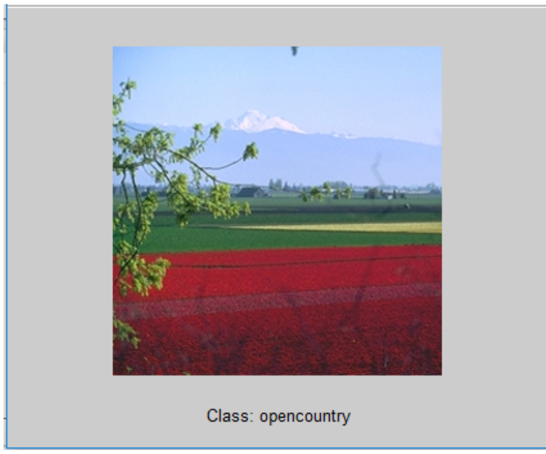
Output for test image 2



Output for test image 3



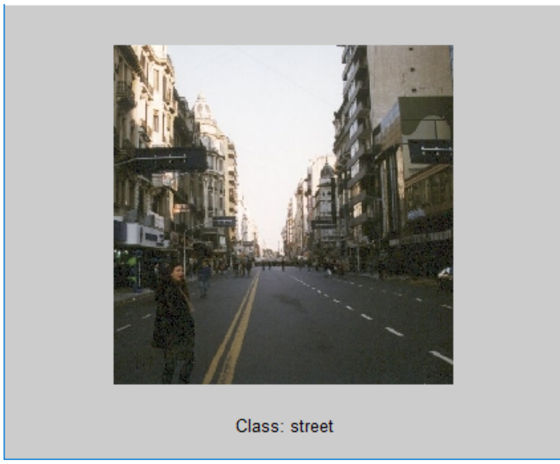
Output for test image 4



Command Window

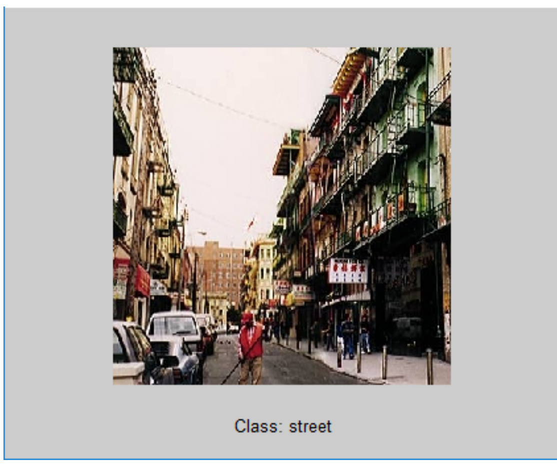
```
>> program
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested opencountry_fie30.jpg
Calculating Mahalanobis Distance
Distance to mountain class: 26.4442
Distance to opencountry class: 25.4904
Distance to street class: 33.5003
Distance to tallbuilding class: 46.6935
opencountry_fie30.jpg belongs to opencountry class
>>
```

Output for test image 5



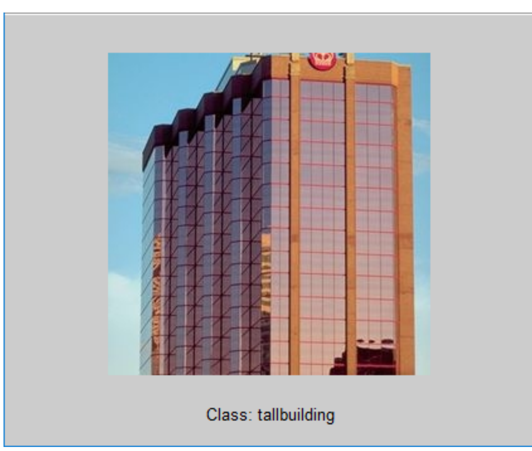

```
Command Window
>> program
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested street_art779.jpg
Calculating Mahalanobis Distance
Distance to mountain class: 69.0185
Distance to opencountry class: 105.5712
Distance to street class: 50.0158
Distance to tallbuilding class: 84.2721
street_art779.jpg belongs to street class
fx >>
```

Output for test image 6




```
Command Window
>> program
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested street_art793.jpg
Calculating Mahalanobis Distance
Distance to mountain class: 61.0865
Distance to opencountry class: 96.9693
Distance to street class: 42.7859
Distance to tallbuilding class: 56.911
street_art793.jpg belongs to street class
fx >>
```

Output for test image 7




```
Command Window
>> program
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested tallbuilding_a462018.jpg
Calculating Mahalanobis Distance
Distance to mountain class: 26.3985
Distance to opencountry class: 35.3298
Distance to street class: 27.7191
Distance to tallbuilding class: 25.7595
tallbuilding_a462018.jpg belongs to tallbuilding class
fx >>
```

Output for test image 8

Command Window

```
>> program
Reading files
Reading Done.
Classes identified are :
mountain --374 images
opencountry --410 images
street --292 images
tallbuilding --356 images
Input the name of Test Image to be tested tallbuilding_a463068.jpg
Calculating Mahalanobis Distance
Distance to mountain class: 42.8943
Distance to opencountry class: 56.3376
Distance to street class: 42.5062
Distance to tallbuilding class: 35.2191
tallbuilding_a463068.jpg belongs to tallbuilding class
>>
```

0.5 Result:

0.5.1 Question 1 (Euclidean Distance):

All outputs for Euclidean distance algorithm gave perfect result and all test images are classified to actual class where it belongs
No. of tested images: 8
No. of tested images classified in actual class : 8

Confusion Matrix:

	Mountain	Opencountry	Street	Tallbuilding
Mountain	2	0	0	0
Opencountry	0	2	0	0
Street	0	0	2	0
Tallbuilding	0	0	0	2

Classification Accuracy:

$$\begin{aligned}\text{classification accuracy (\%)} &= \frac{\mathbf{2+2+2+2}}{\mathbf{2+2+2+2}} \times \mathbf{100} \\ &= \mathbf{100\%}\end{aligned}$$

0.5.2 Question 2 (Mahalanobis Distance):

No. of tested images: 8
No. of tested images classified in wrong class : 2
No. of tested images classified in actual class : 6

Confusion Matrix:

	Mountain	Opencountry	Street	Tallbuilding
Mountain	1	0	1	0
Opencountry	0	1	1	0
Street	0	0	2	0
Tallbuilding	0	0	0	2

Classification Accuracy:

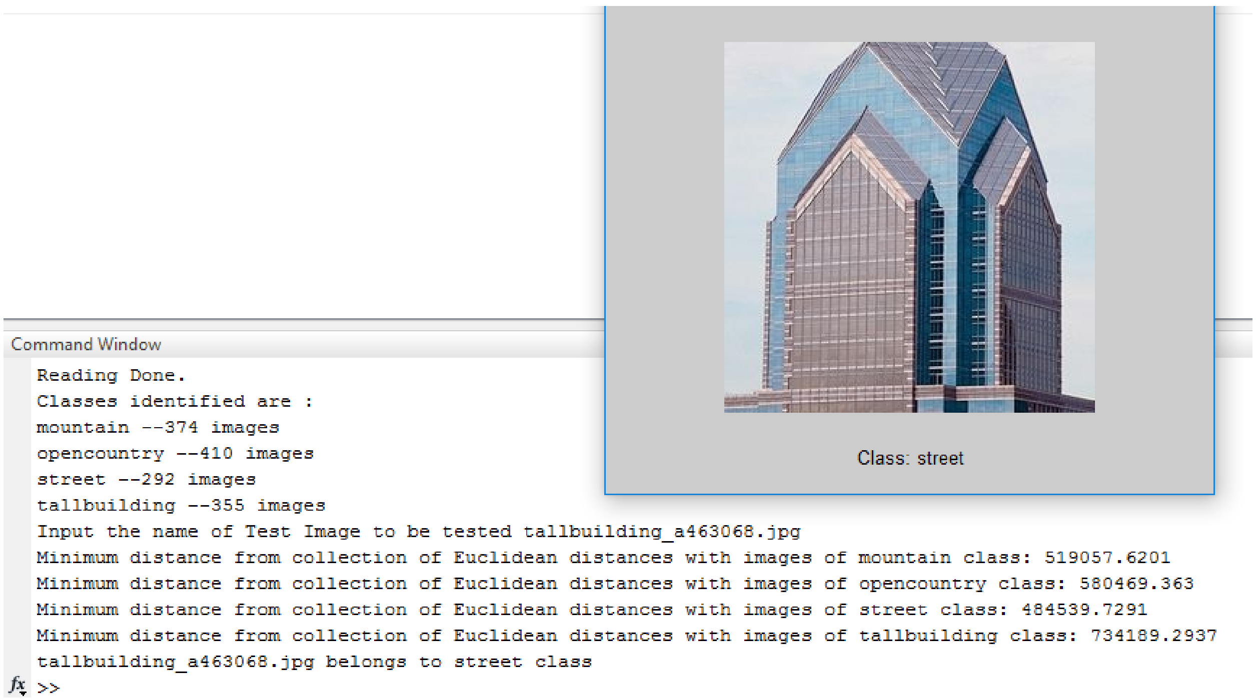
$$\begin{aligned}\text{classification accuracy (\%)} &= \frac{\mathbf{1+1+2+2}}{\mathbf{2+2+2+2}} \times \mathbf{100} \\ &= \mathbf{75\%}\end{aligned}$$

0.6 Observation

0.6.1 Euclidean Distance

Since each test image itself is contained its actual class we will get a an euclidean distance of zero alteast with one image from the actual class and test image. Here we are actually doing image matching algorithm. If the exact image matching is occurred we get zero euclidean distance and then we determine the class by finding the class from which the zero distance is came. It can be seen from the output for all test images given in the result section of Euclidean Distance. For example: test image ***'tallbuilding_a463068.jpg'*** has shown that it has zero distance from an image in the ***'tallbuilding'*** class. Which means the actual ***'tallbuilding_a463068.jpg'*** is present in the ***'tallbuilding'*** class.

If the ***'tallbuilding_a463068.jpg'*** is removed from ***'tallbuilding'*** class, the classification gets wrong. The out of this case shown below:



0.6.2 Mahalanobis Distance

Mahalanobis distance algorithm works by considering each class of images as distribution. So the similarity between images plays an important role in the distance calculation. Because the algorithm computes the distance between each feature of test image with the mean of same feature variable in the distribution. A features of images determines the similarity between images. So to get accurate results of classification by mahalanobis distance algorithm we need perfect feature extraction methods. Here we employed 2D dimensional DFT for feature extraction. and we take strongest 80 DFT coefficients as features. Which merely means that we are taking frequency of most frequent intensity values of an image. Also we converted all images in grayscale spatial domain. So a perfect red and blue images shows same DFT values. So if we classify test image with colour as class we will get poor results.