Ahmet Can Ozbek

Texture Analysis and Classification
(a)Texture Classification: Two Classes + Minimum Mean Distance Classifier

Introduction:

In this problem, our goal is to analyze and classify different textures. Our methods will consist of the techniques from filtering image, signal processing, machine learning and pattern recognition.

Methodology:

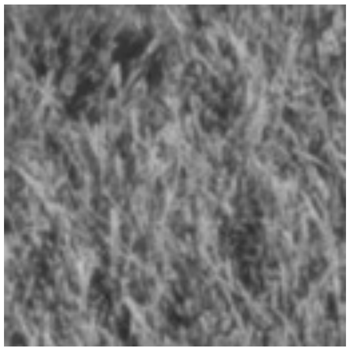
In this problem our methodology will consist of first feature extraction and then applying various machine learning and pattern recognition techniques such as Minimum Mean Distance to Classification using Mahalanobis distance. We will also investigate the effect of the number of dimensions by reducing the dimensionality by Principal Component Analysis(PCA) and Linear Discriminant Analysis(LDA). We can summarize the following procedure as follows:

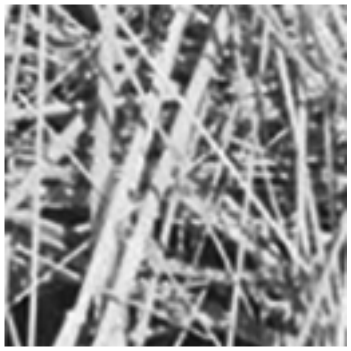
1)Feature Extraction

2)Minimum Mean Distance to Classification using Mahalanobis distance (both with PCA and LDA)

1)Feature Extraction:

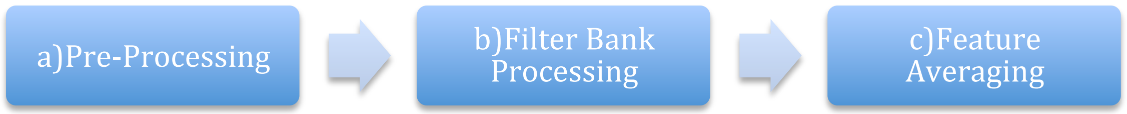
For feature extraction, we are going to apply the Law's filters. The theory behind using Laws filters for feature extraction is related to signal processing concept of the energy levels of the image signal in different frequency ranges. By applying Laws filters, we are trying to get a feature vector representing the energy levels of the image in different frequency ranges. Let's take a look at the two images below to get a better understanding of the concept of using energy levels in different frequency ranges to classify different textures. The image below on the left is the Grass texture image and the one on the right is Straw texture image.


Grass


Straw

By direct visual observation of the texture images above, we can say that grass image has a smoother texture compared to the straw image. If we traverse a direction through the pixels of the image, we will see a more rapid change in the pixel values for the straw texture image, therefore we can say that Straw texture image will be more likely to contain higher energy in higher frequency ranges in general. In this problem, we are going to use this concept to classify different texture images.

In the process diagram below, we have the steps to be done for feature extraction:



We are going to explain each step in the following:

a)Pre-Processing: In the pre-processing part our goal is to remove the dc component of the signal image. The reason why we want to get rid of the dc component is that we are only interested in the AC component of the signal image. We would like to see how much energy is present for each frequency subdivision ranges. If we do not kill DC component before applying anything, we will be misinformed about the energy levels in the low frequency ranges.

To remove the DC component, we are performing local mean subtraction. Local mean subtraction is done as follow: We are visiting each pixel of the image and get the 5x5 window of the neighbor pixels with the visited pixel at the center of the window, we get the mean value of this 5x5 window and subtract it from the center pixel. As a result, we get rid of the DC component from the image.

b)Filter Bank Processing: This is the part where we apply the Law's filters. Before talking about the law's filters, let's look at the 1-D Kernels:

1D Kernel Name	Filter Coefficients
L5(Level)	[1 4 6 4 1]
E5(Edge)	[-1 -2 0 2 1]
S5(Spot)	[-1 0 2 0 -1]
W5(Wave)	[-1 2 0 -2 1]
R5(Ripple)	[1 -4 6 -4 1]

For filter bank processing, we need a bank of Laws filters created from the 1D Kernels. For this purpose, we are getting the tensor product of every possible two element permutations of 1D Kernels. This tensor product procedure gives us a bank of 25 (=5*5) Laws filters each of which has 5x5 dimension.

We can get a better understanding for the reason why Laws filter is useful for extracting features of the energy levels in different frequency ranges by signal processing and frequency analysis concepts. In the figure below, we can observe 2 dimensional Fourier transforms of each 25 Laws filters. As we can see, each 25 Laws filter filters a certain area(range) of the frequencies of the image. With this way, we can get information about how much energy is present in different frequency ranges. This plot is done by me using the fft2() function in MATLAB

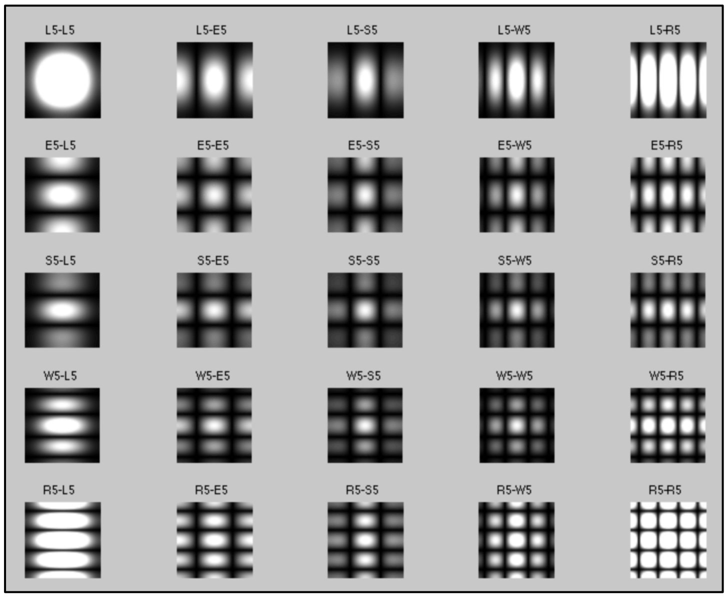
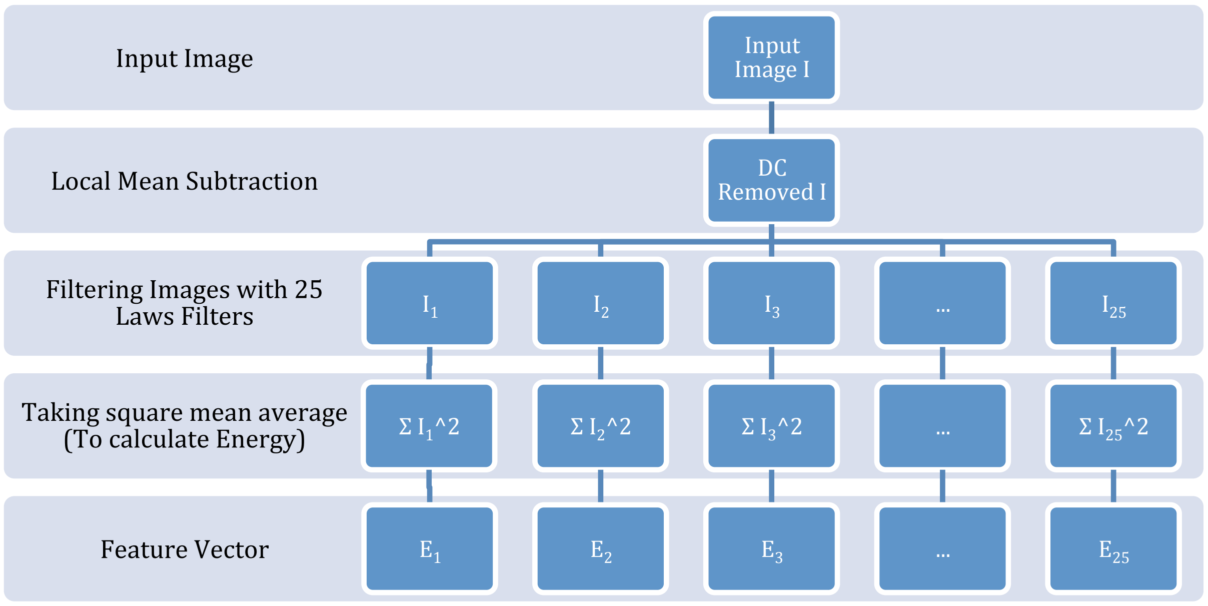


Figure 1 - Frequency responses of each Laws filter (Created with fft2() function of MATLAB)

c)Feature Averaging: After applying, 25 Laws filters to our input image, we get 25 filtered images with Laws filter. As we know that each image contains only one texture, it is okay to take the square mean average of each filtered image. For taking the square mean of one filtered image, we simply take the square of each pixel value, sum them up and then divide this sum by the size of the image. The purpose of doing this procedure is to get the average energy value in each 25 Laws filtered image.

If we do this procedure we get just one numerical value from each 25 Laws filtered images. As a result, we get a 25 dimensional vector from one input image. This 25 dimensional output vector is our result of feature extraction from the image and each element of this vector is just a number representing the enery values in specific frequency ranges.

In the process figure below we can see the summary of feature extraction:



2)Minimum Mean Distance to Classification using Mahalanobis distance (both with PCA and LDA):

In this part, we are applying the pattern recognition and machine learning to classify between different texture images. For performing classification we need training data and test data. Training data is used to train our classifiers and Test data is for measuring how accurate the trained classifiers is. We have two classes: Grass and Straw

-Training Data: 36 Grass Images, 36 Straw Images

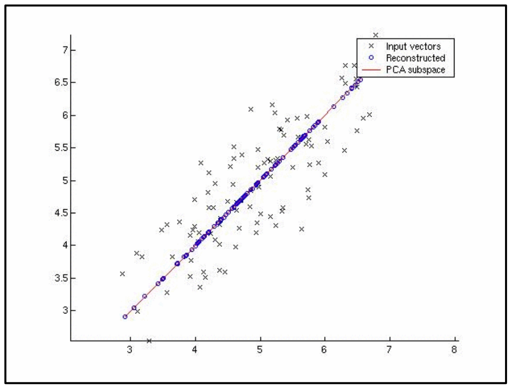
-Test Data: 24 Unknown images

In this part of the problem the classifier that is going to be used is Minimum Mean Distance to Classification using Mahalanobis distance. The classification method of Minimum Mean Distance to Classification using Mahalanobis distance is simple. It classifies and input sample to the class which has the closest mean. However, this distance is not measure in terms of Euclidean distance, instead our distance measure is Mahalanobis distance which takes into account of the way how the data is distributed and the variance of the data distribution. Therefore for calculation of Mahalanobis distance we need the information of variences of the data which is contained in the "S" matrix which is also called 'covarience matrix'. And 'μ' represents the class means. The equation below shows us how to calculate the Mahalanobis distance.

$$\boxed{D_M(x) = \sqrt{(x-\mu)^T S^{-1}(x-\mu)}.}$$

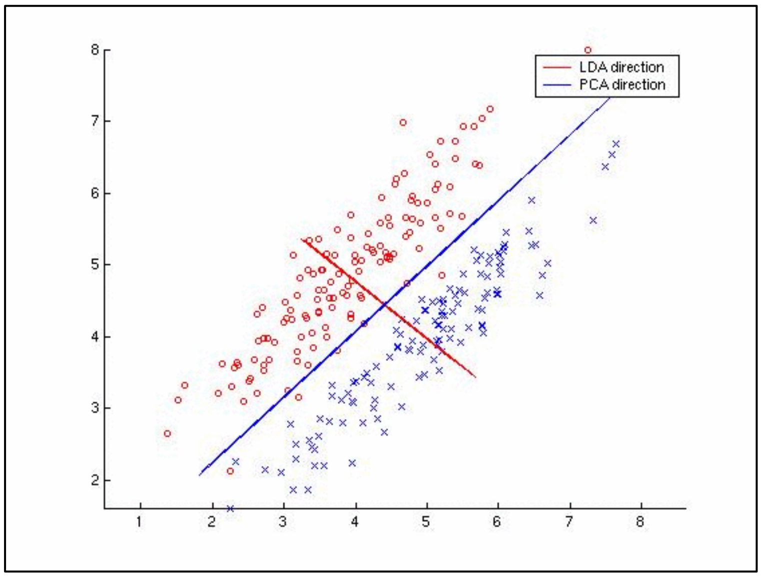
In this part of the problem, we are also asked to investigate the dimensionality of the feature space. From the feature extraction process, we get 25 features for each sample image. We are asked to reduce this dimensionality to 1 dimension by PCA and LDA. Let us have a brief explanation of PCA, LDA and their comparison.

PCA (Principal Component Analysis): In our data sets, some features are highly correlated. Therefore some features might be redundant. PCA aims to find highly correlated features and to get rid of redundant features. PCA also tries to find a subspace to project all the sample onto this subspace, this subspace is along the direction of the variance of the samples are the largest. To calculate the subspace to project the samples, we find the eigenvectors and the eigenvalues of the covariance matrix, the eigenvectors that correspond to the largest eigenvalues give us the subspace that we need the project the samples on. As we can see from the figure below, in 2 dimensional space, the PCA subspace is found to be in the direction of the largest varience of the samples.



LDA (Linear Discriminant Analysis); As we can understand from the above, the explanation of PCA, PCA does not take the class labels into account while choosing the subspace the samples will project onto. However, LDA does take the class labels into account, the goal of LDA is to find a subspace that the projected labels will have minimum inter class variance and the samples of different classes will be as far as possible from each other. We can say that LDA is trying to maximize the function below:

$$J = \frac{(Distance\ between\ projected\ class\ means)^\wedge 2}{Some\ measure\ of\ varience\ of\ each\ projected\ class, combined}$$



In the figure above, we can see the difference between PCA and LDA, as we can see LDA take the class labels into account and finds the direction of subspace so that classes can be separated.

Results:

To get results I coded in C++, using OpenCV 3.0.0 library.

Before doing any classification all features are normalized to the range [0,1] by applying this transformation:

$$X' = \ (X - Xmin)/(Xmax - Xmin)$$

Output of my code:

```
Minimum Mahalanobis Distance To Class Means Classifier:
Number of Errors in Training Data: 0
Error Rate in Training Data: 0
Number of Errors in Test Data: 0
Error Rate in Test Data: 0
*Dimension Reduction*
PCA:
(PCA)Number of Errors in Training data with dimensions reduced to 1: 0
(PCA)Number of Errors in Test data with dimensions reduced to 1: 0
*Dimension Reduction*
LDA:
(LDA)Number of Errors in Training data with dimensions reduced to 1: 0
(LDA)Number of Errors in Test data with dimensions reduced to 1: 0
```

Results in table:

	Minimum Mean Distance Classification (25 Dimension)	Minimum Mean Distance Classification (1 Dimension) (PCA)	Minimum Mean Distance Classification (1 Dimension) (LDA)
Error rate in Training Data	0	0	0
Error rate in Test Data	0	0	0

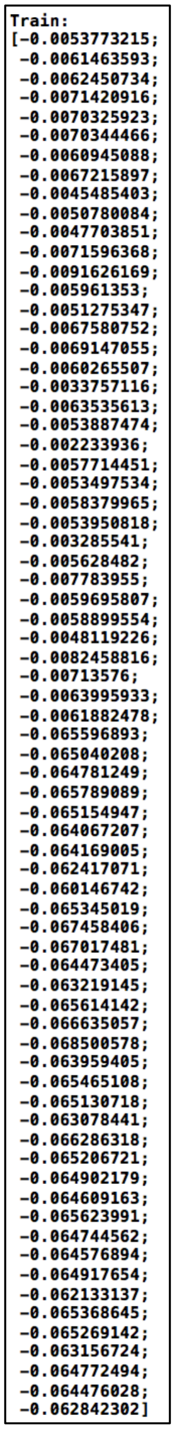
As we can see from the table above, all classification methods classify training and test data with zero error.

Feature Dimension Reduction Using PCA:

With 25 Dimensions, we get 0 error rate, after applying PCA we get 0 error rate again. Therefore we can conclude that in 25 features, some features are highly correlated with each other and we can say that not 25 features are equally important.

Feature Dimension Reduction using LDA

After reducing the 25 feature dimensions to 1 feature dimension, we also get 0 error rate. For justification of 0 error rate, I am going to show the feature values when the features are reduced and projected to 1 dimension, so our reduced dimension feature has dimension 72x1 for the training data, as there are 72 samples in training data.



```
Train:
[-0.0053773215;
 -0.0061463593;
 -0.0062450734;
 -0.0071420916;
 -0.0070325923;
 -0.0070344466;
 -0.0060945088;
 -0.0067215897;
 -0.0045485403;
 -0.0050780084;
 -0.0047703851;
 -0.0071596368;
 -0.0091626169;
 -0.005961353;
 -0.0051275347;
 -0.0067580752;
 -0.0069147055;
 -0.0060265507;
 -0.0033757116;
 -0.0063535613;
 -0.0053887474;
 -0.002233936;
 -0.0057714451;
 -0.0053497534;
 -0.0058379965;
 -0.0053950818;
 -0.003285541;
 -0.005628482;
 -0.007783955;
 -0.0059695807;
 -0.0058899554;
 -0.0048119226;
 -0.0082458816;
 -0.00713576;
 -0.0063995933;
 -0.0061882478;
 -0.065596893;
 -0.065040208;
 -0.064781249;
 -0.065789089;
 -0.065154947;
 -0.064067207;
 -0.064169005;
 -0.062417071;
 -0.060146742;
 -0.065345019;
 -0.067458406;
 -0.067017481;
 -0.064473405;
 -0.063219145;
 -0.065614142;
 -0.066635057;
 -0.068500578;
 -0.063959405;
 -0.065465108;
 -0.065130718;
 -0.063078441;
 -0.066286318;
 -0.065206721;
 -0.064902179;
 -0.064609163;
 -0.065623991;
 -0.064744562;
 -0.064576894;
 -0.064917654;
 -0.062133137;
 -0.065368645;
 -0.065269142;
 -0.063156724;
 -0.064772494;
 -0.064476028;
 -0.062842302]
```

On the right, we see the feature values of the training data after normalizing and reducing its dimension from 25 to 1 by LDA and the matrix on the right has dimension 72x1. In the training data, first 36 samples correspond to the grass images and the second half of 36 samples correspond to the straw images. As we can observe from the numerical values, the first 36 samples have values ranging between (-0.005) and (-0.007) approximately. And the second half of 36 samples feature values are in the range between (-0.061) and (-0.066). Therefore there is a very noticeable distance between the feature values of the two classes on 1 dimension. As a result, minimum mean distance classification by mahalanobis distance classifies the training samples with 0 error rate.

Performance Comparison:

In the results of:
(i)No feature reduction
(ii)Feature reduction with no class label
(iii)Feature reduction with labeled data

We do not see any performance difference between those results because all of them classify with zero error rate.

We can conclude that not all of the features are equally important in our dataset, because even we reduce the dimensionality from 25 to 1, we still get zero error rate.

However, in an another dataset, reducing the dimensions might give different error rates than the classification without feature reduction.

If we compare PCA and LDA, we might expect for LDA to perform better classification and less error rate than using LDA, because PCA does not class labels into account, whereas LDA takes class labels into account and LDA tries to project it to a subspace that the samples of different classes are more likely to be separable from class to class.

(b) Advanced Texture Classification: Multi-Classes + SVM

<u>Introduction:</u>

In this problem, our goal is to classify different textures as in part (a) . We need to implement four different classifiers:

1)Grass VS. Non-grass classifier
2)Straw VS. Non-straw classifier
3)Sand VS. Non-sand classifier
4)Leather VS. Non-leather classifier

To implement these classifiers, we are going to use minimum distance to means classifier, Support Vector Machine (SVM) and Principal Component Analysis (PCA).

<u>Methodology:</u>

To do any classification, we first need to do feature extraction. Feature extraction is done as the same way as it is done in part(a). After feature extraction, we get 25 features for each sample.

As mentioned in the introduction part, we need to implement four binary classifiers. Each classifier is treated, and analyzed separately.

Our dataset consists of:
-48 Grass Images
-48 Straw Images
-48 Sand Images
-48 Leather Images

Here is how we get training and test data for these independent four classifiers

*For Grass VS. Non-grass classifier:	-Training Data:	36 Grass, 12 Straw, 12 Sand, 12 Leather
	-Test Data:	12 Grass, 3 Straw, 3 Sand, 3 Leather
*For Straw VS. Non-straw classifier:	-Training Data:	36 Straw, 12 Grass, 12 Sand, 12 Leather
	-Test Data:	12 Straw, 3 Grass, 3 Sand, 3 Leather
*For Sand VS. Non-sand classifier:	-Training Data:	36 Sand, 12 Grass, 12 Straw, 12 Leather
	-Test Data:	12 Sand, 3 Grass, 3 Straw, 3 Leather
*For Leather VS. Non-leather classifier:	-Training Data:	36 Leather, 12 Grass, 12 Straw, 12 Sand
	-Test Data:	12 Leather, 3 Grass, 3 Straw, 3 Sand

As result, each of the four classifiers have 72(=36 + 12*3) samples in training data
24(=12 + 4*3) samples in the test data

The methods we apply for each of the four classifiers is as follows:
1)Normalize all the feature according to the equation below:

$$X^{'} = (X - Xmin)\ (Xmax - Xmin)$$

2)Classify with minimum distance to means classifier and SVM (25 Features (without dimension reduction)
3)Classify with minimum distance to means classifier and SVM (3 Features (with dimension reduction) (With PCA)

<u>Minimum distance to means classifier:</u> This classification method is simple. We just calculate the class means and we assign the sample to the class which has the closest class mean to the sample.

<u>Support Vector Machine(SVM):</u> SVM is a non-probabilistic binary classifier. SVM tries to separate the classes by a space that is as much as possible. The samples closest to the boundary are called support vectors. SVM also transforms the space to a space that has higher dimensions by Kernel functions. In this question we are going to use Linear Kernels.

Results:

Output of my code:

```
1)GRASS VS NON-GRASS CLASSIFICATION
*Minimum Distance To Means Classifier (25 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0
*SVM (25 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0
*Minimum Distance To Means Classifier (3 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0
*SVM (3 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0

2)STRAW VS NON-STRAW CLASSIFICATION
*Minimum Distance To Means Classifier (25 Dimensions)
--(Training)Number of Errors: 12
--(Training)Error Rate: 0.166667
--(Test)Number of Errors: 4
--(Test)Error Rate: 0.166667
*SVM (25 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0
*Minimum Distance To Means Classifier (3 Dimensions)
--(Training)Number of Errors: 12
--(Training)Error Rate: 0.166667
--(Test)Number of Errors: 4
--(Test)Error Rate: 0.166667
*SVM (3 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0

3)SAND VS NON-SAND CLASSIFICATION
*Minimum Distance To Means Classifier (25 Dimensions)
--(Training)Number of Errors: 12
--(Training)Error Rate: 0.166667
--(Test)Number of Errors: 4
--(Test)Error Rate: 0.166667
*SVM (25 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0
*Minimum Distance To Means Classifier (3 Dimensions)
--(Training)Number of Errors: 12
--(Training)Error Rate: 0.166667
--(Test)Number of Errors: 4
--(Test)Error Rate: 0.166667
*SVM (3 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0

4)LEATHER VS NON-LEATHER CLASSIFICATION
*Minimum Distance To Means Classifier (25 Dimensions)
--(Training)Number of Errors: 16
--(Training)Error Rate: 0.222222
--(Test)Number of Errors: 4
--(Test)Error Rate: 0.166667
*SVM (25 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0
*Minimum Distance To Means Classifier (3 Dimensions)
--(Training)Number of Errors: 16
--(Training)Error Rate: 0.222222
--(Test)Number of Errors: 4
--(Test)Error Rate: 0.166667
*SVM (3 Dimensions)
--(Training)Number of Errors: 0
--(Training)Error Rate: 0
--(Test)Number of Errors: 0
--(Test)Error Rate: 0
```

Results represented in tables:

Grass VS. Non-grass	Minimum Distance To Means Classifier(25 Dimension)	SVM(25 Dimension)	Minimum Distance To Means Classifier(3 Dimension)	SVM(3 Dimension)
Error in Training Data	0	0	0	0
Error In Test Data	0	0	0	0

Straw Vs. Non-straw	Minimum Distance To Means Classifier(25 Dimension)	SVM(25 Dimension)	Minimum Distance To Means Classifier(3 Dimension)	SVM(3 Dimension)
Error in Training Data	0.166667	0	0.166667	0
Error In Test Data	0.166667	0	0.166667	0

Sand VS. Non-sand	Minimum Distance To Means Classifier(25 Dimension)	SVM(25 Dimension)	Minimum Distance To Means Classifier(3 Dimension)	SVM(3 Dimension)
Error in Training Data	0.166667	0	0.166667	0
Error In Test Data	0.166667	0	0.166667	0

Leather VS. Non-leather	Minimum Distance To Means Classifier(25 Dimension)	SVM(25 Dimension)	Minimum Distance To Means Classifier(3 Dimension)	SVM(3 Dimension)
Error in Training Data	0.222222	0	0.222222	0
Error In Test Data	0.166667	0	0.166667	0

As we can see the results from the tables above, dimension reduction from 25 to 3 does not make any effect on the error rate.

If we compare the performance of Minimum Distance to Means Classifier and SVM, we can conclude that SVM performs better because the classification with SVM has less error rate. The reason why SVM performs better is because SVM is a very advanced technique whereas Minimum Distance to Means classifier is a simple technique.