# A Comparisons of Feature Transformations For Classification

John L. Weatherwax*

January 11, 2006

**Abstract**

There currently exist a large number of algorithms aimed at reducing the dimensionality of a feature set used for classification. These algorithms make various assumptions about the underlying data. Two very common statistical technique are the well known Principle Component Analysis (PCA) and Multiple Discriminant Analysis (MDA) also known as Linear Discriminant Analysis (LDA). Of these two techniques, only the MDA algorithm explicitly uses class information. A more recent technique includes the maximization of the mutual information (MMI) between the projected features and their classlabels [13]. For a nice overview of existing techniques for feature selection and transformation see reference [3].

In this project I will empirically compare and contrast the classification and data visualization performance of several algorithms for feature transformation with the purpose of improving classification. Specifically I will consider PCA, MDA, and MMI on a variety of data sets and retaining different numbers of projected features. Classification performance will be considered using a variant of a nearest neighbor classifier called a learning vector quantization [4, 6].

## 1 Introduction

There currently exist a large number of algorithms aimed at reducing the dimensionality of a feature set used for classification by relying on the statistical properties of the underlying distribution of data. Two common and well known examples of this type of technique are Principle Component Analysis (PCA) and Multiple Discriminant Analysis (MDA) also known as Linear Discriminant Analysis (LDA). In particular, for classification, of the techniques mentioned, only the MDA algorithm explicitly uses class information. Recently there has been interest in comparing the performance of PCA against MDA [8]. It is generally felt that the projected feature subset produced by MDA is better for classification purposes since the MDA algorithm finds a linear transformation that maximizes a measure of class separation.

*wax@mit.edu

The ultimate goal of any pattern recognition system is to achieve the best possible performance on novel feature vectors. In practice this is often attempted by developing a classification algorithms that performs "optimally" on a given subsection of data (the training set). This can be done in many ways depending on the assumed classifier form but a common theme is that the proposed classifier depends on a set of a-priori unknown, but constant, parameters. In supervised machine learning it is assumed that the designer of the pattern classifier has at his disposal a set of measured features and their associated class labels. In one way or another all pattern recognition algorithms use these samples to compute the unknown parameters of the learning algorithm.

In practical applications the dimension of the pattern space can often become quite large without any obvious means for reducing this dimension. For instance, in facial recognition, usually the raw measurements represent an entire image and it is not uncommon for the natural corresponding feature vector to contain several hundred components. If probabilistic methods are used for classification the curse of dimensionality [4] makes it very difficult to guarantee enough representative samples to ensure sufficient confidence in the estimated probability densities. In addition, for computational reasons it is often desirable to project the feature vector into a space of smaller dimensional before the execution of the learning algorithm. These unfortunate situations creates the requirement that some sort of dimensionality reduction be preformed before the parameters in the classification algorithm are estimated. Several common techniques currently exist for performing dimensionality reduction. The goal of this project is to empirically consider the following dimensionality reduction techniques and how the affect classification performance: Principle Component Analysis (PCA), Multiple Discriminant Analysis (MDA), and Maximization of mutual information (MMI). We briefly describe each of these base techniques before describing the extensions developed to make them class specific in section 3

Principle Components Analysis (PCA), also called the Karhunen-Loeve Transformation [5], is a purely statistical technique aimed at projecting the original feature space into a smaller space, the individual components of which having zero correlation. Because PCA is a pure statistical technique it does not incorporate any class information. As such, it is commonly believed that the reduced feature space will not necessarily perform optimally under classification [8].

Multiple Discriminant Analysis (MDA) [1, 2] explicitly considers the class labels associated with each feature vector when computing its transformation. It obtains a projection that maximizes a functional measuring the ratio of the spread of features among different classes relative to the spread of the features within the individual classes.

Maximization of Mutual Information (MMI) is a technique that guides its transformations in such a way to optimized the decrease in the entropy of the transformed class variable. Since this technique is a cornerstone of this paper and will not be familiar to most it will be described in detail in subsection 3.3.

This paper will empirically measure the data visualization and classification performance of the above algorithms on a number of standard data sets. Specifically we will attempt to duplicate as many results as possible from [13] time permitting. The data set considered are described in section 3.4. For each of the projected spaces the performance under classification will be assessed using a variant of a nearest neighbor classifier call Learning Vector Quantization (LVQ) [4, 6] and provided by the software [7].

# 2 Classifier Descriptions

In this subsection we describe the classifier used for classification performance. This classifier was chosen because it was readily available on the web and was one of the classifiers used in [13].

## 2.1 The LVQ Classifier

The LVQ classifier is a technique due to Kohonen [6] and is somewhat similar to K-means clustering. The basic idea is that several prototypes of a given class are deposited in feature space. The training points attract prototypes of the correct class and repel those of the incorrect class. When the algorithm has finished, prototypes should be close to the training points representing their class. Often the starting location of the prototypes is given from the results of using a K-means clustering algorithm. Once the iteration has converged the classification is taken to be that of the nearest (in Euclidean distance) prototype point.

# 3 Algorithm Descriptions

For a concrete discussion in what follows, we assume that our data is given in $N_c$ classes given by $\{c_1, c_2, \ldots, c_{N_c}\}$ with original feature space given by $D$-dimensional feature vectors for $c_p$ (denoted by $x$) contained in the set $\mathcal{D}_i$. In all the techniques we consider below a projection matrix $W$ will be constructed that project our $D$ dimensional feature vector into a smaller $d$ dimensional space ($d < D$).

In general, as a preprocessing step it is good to do some data normalization[1]. Several common data normalization steps are often used. "Standardizing" involves subtracting from each feature the global mean (mean over all training points independent of class) and dividing by the global standard deviation. "Sphering" transforms the original data set into a new one with mean zero and a covariance of the identity matrix. In the results that follow the data was sphered (using all of the data) before further transformations took place.

After this process the PCA and MDA projections were computed using *all* of the training data points. Due to the computational complexity of MMI only 1500 randomly selected data points were used[2]. This is consistent with the results found in [13]. Below we present a brief description of the individual feature selection algorithms considered in this paper.

## 3.1 Principle Component Analysis (PCA)

Principle Component Analysis [1] is a method of computing, one vector, $x_0$, that best represents the complete data set. That is, $x_0$ is the vector that minimizes the squared error, $J_0(x_0)$, between itself and every sample, $x_k$, in all $\mathcal{D}_i$.

$$J_0(x_0) = \sum_{k=0}^{n} ||x_0 - x_k||^2. \tag{1}$$

[1] When presenting classification results it is very important to document *what* preprocessing was done, else it can be very difficult to reproduce an authors results as I found in this project.

[2] Sampled according the the empirical priors obtained from the entire dataset.

The $x_0$ that minimizes $J_0$ is determined by solving the eigenvalue equation for the scatter matrix, $S$, where

$$S = \sum_{k=1}^{n} (x_k - m)(x_k - m)^t, \tag{2}$$

where $m$ is the mean of the full data set,

$$m = \frac{1}{n} \sum_{k=1}^{n} x_k. \tag{3}$$

The size of $S$ is $d \times d$, and consequently there will be $d$ solutions to the eigenvalue problem:

$$ST = \lambda T. \tag{4}$$

$T_1$, the eigenvector corresponding to $\lambda_1$, the largest eigenvalue, is the $x_0$ that minimizes $J_0$ and therefore is the first principle axis of the new eigenspace. $T_2$ is the second principle axis, $T_3$ is the third, etc. The raw data, $x_k$, are then projected onto $T$, the eigenvector space. The dimensionality of the projected data may be changed by including only the desired eigenvectors in the transformation matrix.

## 3.2 Multiple Discriminant Analysis (MDA)

Multiple Discriminant Analysis [1] is a linear transformation ($y = Wx$) that seeks to maximize the function $J(W)$, the ratio of between-class scatter to within-class scatter:

$$J(W) = \frac{|\tilde{S}_B|}{|\tilde{S}_W|} = \frac{|W^t S_B W|}{|W^t S_W W|}. \tag{5}$$

$S_W$, the within-class scatter matrix, is the sum of each classes scatter matrix:

$$S_W = \sum_{i=1}^{N_c} \sum_{x \in \mathcal{D}_i} (x - m_i)(x - m_i)^t. \tag{6}$$

$S_B$, the between-class scatter matrix is defined as

$$S_B = \sum_{i=1}^{N_c} n_i (m_i - m)(m_i - m)^t, \tag{7}$$

where $m$ is again the mean of all the data (Eq. 3) and $m_i$ are the means of each class,

$$m_i = \frac{1}{n} \sum_{x \in \mathcal{D}_i} x_i. \tag{8}$$

The columns of $W$ that maximize $J$ are the eigenvectors that solve the generalized eigenvalue problem:

$$S_B w_i = \lambda_i S_W w_i. \tag{9}$$

The size of $S_W$ and $S_B$ are $N_c \times N_c$ and the number of eigenvectors that solve the generalized eigenvalue problem is $(N_c - 1)$. The raw data can then be projected onto the desired dimensions of the eigenspace. MDA takes into account class-wise information, however, this information is blurred when $S_W$ is summed over all classes.

## 3.3 Maximization of Mutual Information (MMI)

Because the metric of mutual information for improved class separability maybe unfamiliar to most people, in this subsection we explain this technique in detail. For additional, details please see the original references [12, 13, 14].

The motivation for the maximization of mutual information comes from its relation to the entropy of a random variable. For a discrete random variable the definition of entropy as defined by Shannon is given by

$$H(C) = -\sum_{c} P(c)\log(P(c)) \tag{10}$$

which is defined in terms of the class prior probabilities $P(c)$. A random variable that is deterministic has an entropy of zero. A random variable that has a uniform distribution has the largest possible entropy. When a continuous feature vector $Y$ is observed this information may change the probability distribution of the class variable. Mathematically the entropy of the discrete class random variable $C$ *after* observing the continuous variable $Y$ is given by

$$H(C|Y) = -\int_{y} p(y)\left(\sum_{c} p(c|y)\log(p(c|y))\right)dy \tag{11}$$

The amount of entropy decrease $(H(C) - H(C|Y))$ (which measures how much the knowledge of $Y$ influences the understanding of $C$) is called the mutual information and is given mathematically in our discrete continuous case by

$$I(C,Y) = \sum_{c}\int_{y} p(c,y)\log(\frac{p(c,y)}{P(c)p(y)})dy \tag{12}$$

As a special case, if $C$ and $Y$ are independent then $p(c,y) = P(c)p(y)$ and the mutual information between $C$ and $Y$ is zero.

The definition above can be seen to be effectively a type of metric between the joint class-feature probability distribution $p(c,y)$ and the product of the marginals $p(c)p(y)$. This analogy will come in handy later for motivating additional measures of distance between probability densities functions.

A motivation for the use of mutual information for classification comes from Fano's classification bound, which limits the bound on misclassification after observing a feature vector $Y$. Fano's bound is given by

$$\Pr(c \neq \hat{c}) \geq \frac{H(C|Y) - 1}{\log(N_c)} = \frac{H(C) - I(C,Y) - 1}{\log(N_c)} \tag{13}$$

Here $\hat{c}$ is the estimated class and $c$ is the true class. Fano's bound gives a lower bound on the probability of misclassification. We see that by maximizing the mutual information between $C$ and $Y$ we are able to decrease this lower bound and correspondingly *decrease* the probability of misclassification. Finding transformed features $Y$ that maximizes the mutual information $I(C,Y)$ is then a prescription for finding features that could possibly achieve good classifier performance.

With this background, a formalized objective for pursuing the idea of maximization of mutual information between the transformed features $Y$ and the class random variable $C$ can be stated as follows. Find a transformation $g$ of the input features $X$ into $Y$ that maximizes the mutual information between $C$ and $Y$. If the transformation

is parameterized by a variable $w$, i.e. $y = g(x; w)$ the parameters $w$ that enforce this maximization can be found with a simple gradient assent procedure as follows

$$w_{t+1} = w_t + \eta \sum_{i=1}^{N} \frac{\partial I}{\partial w} = w_t + \eta \sum_{i=1}^{N} \frac{\partial I}{\partial y_i} \frac{\partial y_i}{\partial w} \,. \tag{14}$$

Other optimization techniques such as Conjugate-Gradient, Levenberg-Marquardt or a stochastic method could also be used. In this report the performance of several variants on these maximization techniques will be reported on.

The evaluation of Eq. 12 is computationally difficult due to the logarithmic and fractional dependence on the probability density functions in their expressions. We can arrive at a more computationally efficient expressions by considering an alternative to the Shannon entropy; that given by the Renyi entropy. The Renyi entropy is defined for discrete and continuous random variables as

$$H_R(C) = -\log(\sum_c p(c)^2) \tag{15}$$

$$H_R(Y) = -\log(\int_y p(y)^2 dy) \tag{16}$$

The computational simplifications to using the above equations can be seen when the probability density function appearing in those expressions are approximated using Parzen density [10] estimation with Gaussian kernels. Gaussian kernels for the $d$ dimensional $y$ vectors are given by

$$G(y, \Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} \exp(-\frac{1}{2} y^t \Sigma^{-1} y) \,. \tag{17}$$

The Parzen approximation of a continuous probability density function $p(y)$ is given with a sum of these Gaussian kernels as

$$p(y) \approx \frac{1}{N} \sum_{i=1}^{N} G(y - y_i, \sigma^2 I) \,. \tag{18}$$

Here $I$ is the $d \times d$ dimensional identity function.

The computational simplifications in using the Renyi definition of entropy with a Gaussian Parzen probability density estimation is a consequence that of the fact that Gaussians' satisfy the following simple convolution like rule

$$\int_y G(y - a_i, \Sigma_1) G(y - a_j, \Sigma_2) = G(a_i - a_j, \Sigma_1 + \Sigma_2) \,. \tag{19}$$

Combining the definition 18 and 19 the quadratic integrations found in 15 simplify to

$$\int_y p(y)^2 dy = \frac{1}{N^2} \sum_{k=1}^{N} \sum_{j=1}^{N} G(y_k - y_j, 2\sigma^2 I) \tag{20}$$

Thus we see that an approximation to the Reyni entropy can be evaluated as a sum of pairwise interactions.

It remains now to find an approximate quadratic measure of mutual information. One such measure between two probability distribution $f$ and $g$ is given as follows and heuristically derived in [11]

$$K_T(f, g) = \int f(x)^2 dx + \int g(x)^2 dx - 2 \int f(x) g(x) dx \,. \tag{21}$$

Using the idea that we desire a computationally tractable approximate measure of mutual information between the probability density functions $p(y,c)$ and $p(y)p(c)$, we can use Eq. 21 with these PDF's to arrive at the following objective function to maximize on transformation.

$$I_T(C,Y) = \sum_c \int_y p(c,y)^2 dy + \sum_c \int_y p(c)^2 p(y)^2 dy - 2\sum_c \int_y p(c,y)p(c)p(y)dy \quad (22)$$

To notationally evaluate the gradient of this expression we define several subexpressions

$$V_{(cy)^2} \equiv \sum_c \int_y p(c,y)^2 dy \quad (23)$$

$$V_{c^2y^2} \equiv \sum_c \int_y p(c)^2 p(y)^2 dy \quad (24)$$

$$V_{cy} \equiv \sum_c \int_y p(c,y)p(c)p(y)dy\,. \quad (25)$$

Giving as a componentwise decomposition of $I_T$

$$I_T(C,Y) = V_{(cy)^2} + V_{c^2y^2} - 2V_{cy}\,. \quad (26)$$

For maximization the gradient of $I_T$ per sample $\frac{\partial I}{\partial y_i}$ is required and is given by the corresponding three term sum

$$\frac{\partial I_T}{\partial y_i} = \frac{\partial V_{(cy)^2}}{\partial y_i} + \frac{\partial V_{c^2y^2}}{\partial y_i} - 2\frac{\partial V_{cy^2}}{\partial y_i}\,. \quad (27)$$

With this framework we are in a position to develop the Parzen density approximations for these expressions. To do so we assume that we have $J_p$ samples for each class $c_p$ giving estimates of the prior class probabilities of

$$P(c_p) = \frac{J_p}{N}\,, \qquad \sum_{p=1}^{N_c} J_p = N \quad (28)$$

As a notion in what follows for a sample $y$ in the output space, when class is irrelevant the $i$-th sample will be written $y_i$ with $1 \le i \le N$, when it is relevant we will use $y_{pj}$ as the notation, where the class index $1 \le p \le N_c$ and the index within the class $1 \le j \le J_p$. Using a Parzen density estimation (Eq. 18) we estimate the class conditional densities as

$$p(y|c_p) = \frac{1}{J_p}\sum_{j=1}^{J_p} G(y - y_{pj}, \sigma^2 I)\,. \quad (29)$$

Since the joint probability density $p(c,y)$ can be decomposed as a conditional and a prior ($p(c,y) = p(c)p(y|c)$) we get for the joint PDF between class $c_p$ and the mapped features $y$

$$p(c_p, y) = \frac{1}{N}\sum_{j=1}^{J_p} G(y - y_{pj}, \sigma^2 I) \quad (30)$$

Marginalizing to obtain $p(y) = \sum_c p(c,y)$ we obtain

$$p(y) = \frac{1}{N}\sum_{i=1}^{N} G(y - y_i, \sigma^2 I) \quad (31)$$

With these expressions we are a situation where we can evaluate approximations to the individual expressions in Eq. 23, 24, and 25 as follows

$$V_{(cy)^2}(\{c_i, y_i\}) = \frac{1}{N^2}\sum_{p=1}^{N_c}\sum_{k=1}^{J_p}\sum_{l=1}^{J_p} G(y_{pk} - y_{pl}, 2\sigma^2 I) \tag{32}$$

$$V_{c^2y^2}(\{c_i, y_i\}) = \frac{1}{N^2}\left(\sum_{p=1}^{N_c}\left(\frac{J_p}{N}\right)^2\right)\sum_{k=1}^{N}\sum_{l=1}^{N} G(y_k - y_l, 2\sigma^2 I) \tag{33}$$

$$V_{cy}(\{c_i, y_i\}) = \frac{1}{N^2}\sum_{p=1}^{N_c}\frac{J_p}{N}\sum_{j=1}^{J_p}\sum_{k=1}^{N} G(y_{pj} - y_k, 2\sigma^2 I)\,. \tag{34}$$

In the same vein we require the derivative of the above expressions with respect to the transformed sample points $y_i$. After some simplification and using the following fact

$$\frac{\partial}{\partial y_i} G(y_i - y_j, 2\sigma^2 I) = G(y_i - y_j, 2\sigma^2 I)\frac{y_j - y_i}{2\sigma^2} \tag{35}$$

(note the transposition $y_i$ and $y_j$ on output) these are given by

$$\frac{\partial V_{(cy)^2}}{\partial y_{ci}} = \frac{1}{N^2\sigma^2}\sum_{k=1}^{J_c} G(y_{ck} - y_{ci}, 2\sigma^2 I)(y_{ck} - y_{ci}) \tag{36}$$

$$\frac{\partial V_{c^2y^2}}{\partial y_{ci}} = \frac{1}{N^2\sigma^2}\left(\sum_{p=1}^{N_c}\left(\frac{J_p}{N}\right)^2\right)\sum_{k=1}^{N} G(y_k - y_i, 2\sigma^2 I)(y_k - y_i) \tag{37}$$

$$\frac{\partial V_{cy}}{\partial y_{ci}} = \frac{1}{N^2\sigma^2}\sum_{p=1}^{N_c}\left(\frac{J_p + J_c}{2N}\right)\sum_{j=1}^{J_p} G(y_{pj} - y_{ci}, 2\sigma^2 I)(y_{pj} - y_{ci})\,. \tag{38}$$

With these expressions and a parameterized transformation $y_i = g(x_i; w)$ we have a well defined gradient assent procedure. In this paper, the specific transformation we seek is given by a linear transformation that projects from our original feature space onto of dimension $D$ to a lower dimensional space of dimension $d$. That is we seek the transformation $W$ such that

$$W = \text{argmax}_W(I(\{c_i, y_i\})); \qquad y_i = W^t x_i \tag{39}$$

Subject to the constraint $W^tW = I$. In this case the derivatives of $y_i$ with respect to $x_i$ are given by $\frac{\partial y}{\partial W} = x^t$. Here $W$ is a $D \times d$ orthonormal matrix. With these specifications a gradient assent method can be derived using Eq. 14. As given we will call this method the *matrix* gradient assent algorithm.

In the matrix formulation above we have $dD$ unknown parameters subject to the constraint that $W$ is an orthonormal matrix. This constraint can be simplified and the dimensionality reduced, by parameterizing the linear transformation with rotations. The idea is to start with a given orthogonal matrix $W_0$ and to perturb from this by rotating subspace spanned by the columns of the $W_0$ into the subspace perpendicular to that spanned by the columns of $W_0$.

Since any of the $d$ initial column vectors in $W_0$ can be rotated toward any of the $D-d$ dimensions of the orthogonal complement of the columns of $W_0$ we have $d(D-d)$ angles to parameterize by. The can be represented via Givens rotations.

Define $\hat{G}(o, u, \theta)$ as a $D \times D$ unit matrix with the exception of four elements, $\hat{G}_{o,o} = \cos(\theta)$, $\hat{G}_{o,u} = \sin(\theta)$, $\hat{G}_{u,o} = -\sin(\theta)$, and $\hat{G}_{u,u} = \cos(\theta)$. A specific rotation

plan is defined by the indices $o$ and $u$ and $\hat{G}(o,u,\theta)W_0$ rotates the columns of $W_0$ by an angle $\theta$ along that plane. Thus we can write

$$W = \left( \prod_{o=1}^{d} \prod_{u=d+1}^{D} \hat{G}(o,u,\theta_{od+(u-d)}) \right) W_0 = \left( \prod_{i=1}^{d(D-d)} G(i,\theta_i) \right) W_0 \tag{40}$$

where as in [13] we define a different indexing structure of $G(i,\theta) = \hat{G}(\text{floor}((i-1)/(D-d)) + 1, ((i-1) \text{mod}(D-d)) + d + 1, \theta)$ The required derivatives are then computed with this expression as

$$\frac{\partial W}{\partial \theta_k} = \left( \prod_{i=1}^{k-1} G(i,\theta_i) \right) G'(k,\theta_k) \left( \prod_{i=k+1}^{d(D-d)} G(i,\theta_i) \right) W_0 \tag{41}$$

where $G'(k,\theta_k) = \frac{\partial}{\partial \theta_k} G(k,\theta_k)$. This matrix is otherwise a zero matrix, except for the trigonometric functions in $\hat{G}(o,u,\theta))$ have been replaced by their derivatives. A gradient ascent algorithm for updating $\Theta = [\theta_1, \theta_2, \ldots, \theta_{d(D-d)}]^t$ is given by

$$\Theta_{t+1} = \Theta_t + \eta \sum_{i=1}^{N} \frac{\partial I}{\partial y_i} \frac{\partial y_i}{\partial W} \frac{\partial W}{\partial \Theta} \tag{42}$$

Please see Appendix A for a complete derivation of this expression. In what follows this will be called the *parametric* gradient ascent algorithm. It was this algorithm that will be reported on in this write up.

For this investigation the above algorithms were coded in a mix of Matlab and C using the Mathworks Mex API. The inclusion of the C language was critical because of the computational nature of the MMI approach. Even with these speed ups the MMI projection was computed on a subset of size 1500 chosen from the training set.

Once the objective of maximization has been formalized there are still some details to be worked out that are in no means trivial. The two main items that need discussion are the setting of the $\eta$ parameter and the choice of the bandwidth ($\sigma$) of the Gaussian kernels. These two parameter have a delicate balance between them that must be understood when performing gradient ascent iterations. The size of the Gaussian bandwidth determines how local the Gaussian Parzen estimate are with a larger bandwidth resulting in more non local interactions. When the iterations begin $\sigma$ should be taken to be "large" something like half the projected distance between the two farthest points. The consequence of this large bandwidth is that the Gaussian evaluations (and its derivatives) are very small. Thus initially the magnitude of $\eta$ which would need to be searched over looking for a maximum (via some line search technique) can be very large. As the iterations proceed and Parzen bandwidth decreased the corresponding search range of $\eta$ should be decreased since the magnitude of the derivatives increase. To perform these operations correctly an algorithm must use of the knowledge of $\sigma$ and the magnitude of the derivative to dynamically set the line search technique used for $\eta$. In this code, due to time considerations, a *very* simple line search consisting of the evaluation of the objective function at several discretized values of $\eta$. This seemed to work well in the two dimensional case since this algorithm was able to reproduce the classification and the data visualization results from [13]. This algorithm was less successful in reproduction of the classification results in higher dimension and the most likely discrepancy is due to the choice of used maximization technique. In higher dimensions one expects *more* local maximums and correspondingly requires a more refined search technique. As mentioned previously, other choices of nonlinear optimization routines maybe better suited for this problem and not suffer from these difficulties.

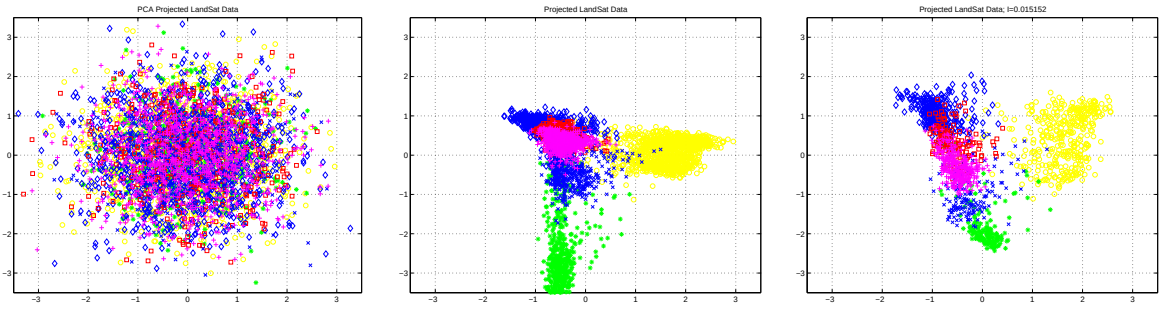


Figure 1: Examples of the PCA, MDA, and MMI projections for the LandSat data. The picture on the left is a the class independent PCA projection. The picture in the middle is the MDA projection and is equivalent to that shown in reference [13]. The picture on the right is the maximal mutual information projection. For details and discussion of these results see the main text.

### 3.4 Considered Data Sets

#### 3.4.1 UCI Landsat Database

This UCI data set [9] has 6 classes each with 36 features each (a 3 by 3 multi-spectral image with 4 spectral bands. This data was used in [13] to test the maximization of mutual information algorithms. This dataset was chosen due to it use in the above mentioned paper as well as it is intermediate size and complexity. Due to time constraints only results from this data set will be presented.

## 4 Computational Experiments

In this section I'll describe the computational experiments that I performed. For this project I focused mainly on replicating the results from [13]. In that paper, the MMI projection was found using the $\theta$ *parameterized* formulation above, coupled with gradient ascent. Since the gradient ascent by nature finds only a local maximum. This procedure was performed several times with different initial conditions and the configuration with the largest mutual information was returned. The reference [13] uses as initial conditions a PCA projection, a MDA projection, and several random projections, selecting the final structure to be the one with the largest mutual information.

To duplicate the results from [13] I present a study in data visualization where by I project the given data set into a plane and observe how well the points become separated under the different PCA, MDA, and MMI transformations.

In addition, an LVQ classifier as describe in [7] was used to classify the data sets projected into various subspaces. The one parameter required for that classifier is the number of code vectors. For the experiments presented here the number of code vectors was taken to be 200.

## 5 Results

### 5.1 UCI LandSat Database

We first present the results on the UCI at a high level and then discuss some details of the calculations to obtain these.

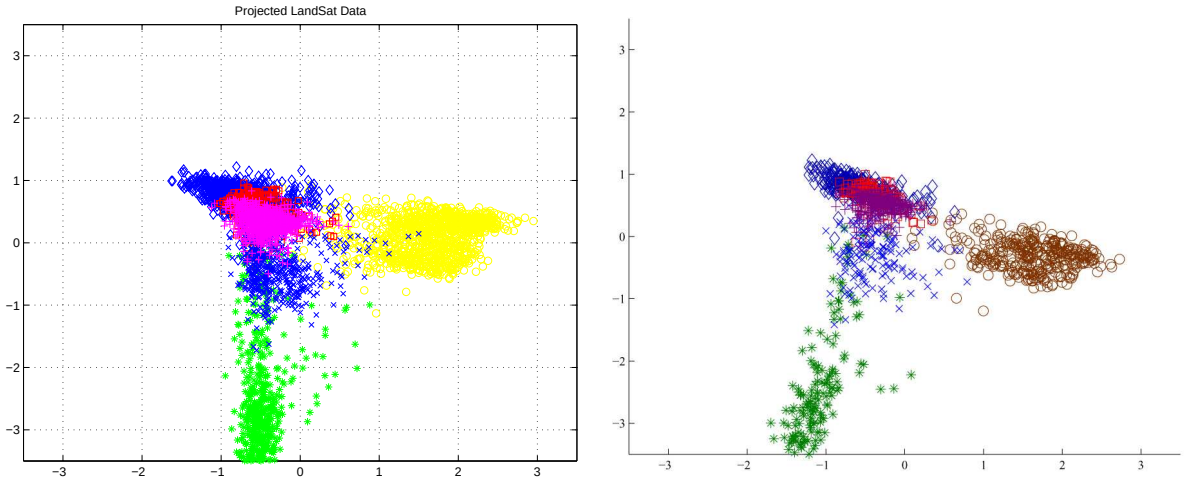


Figure 2: **Left:** The LDA projection of the LandSat data obtained during the course of this investigation (all training data is plotted). **Right:** The LDA projection results from reference [13]. These results are very nearly identical.

## 5.2 Visualization of Class Structure in Data

The first experiment considered has to do with data visualization. Here we consider the optimal projection defined by the three algorithms PCA, MDA, and MMI on a two dimensional plane. In both PCA and MDA this means projecting the feature vectors onto the eigenvectors with the two largest eigenvalues. The MMI projection is obtained by seeking the projection that maximizes the mutual information between the projected feature space and the class labels and is further described in subsection 3.3.

In Figure 1 we see the two dimensional projections (PCA, MDA, and MMI) obtained for the LandSat data, during this investigation. These are placed side by side to emphasize range of different projections available. How these compare with the projections found in [13] will be discussed below. From these pictures one can see that the result of the sphering operation results in the PCA projection having a very spherical looking structure. In fact the projected data all lie on top of each other. Thus one can conclude that the PCA projection will not be a good one for classification. The obtained projection for MDA matches exactly that found in reference [13], and in Figure 2 we present the results obtained in this study adjacent to a duplicate picture taken from [13]. One can see that the MDA graph does indeed maximize the criterion of between-class scatter to the within-class scatter, but (in this case) at the expense of placing several clusters on top of each other. The MMI projection matches very well with that from the original reference [13], and in Figure 3 we present the two results side by side.

In Figure 2 we can see that the two projected results are very similar. This is to be expected since the MDA algorithm is a deterministic algorithm. Any observed difference could be a result of the subsampling of plot points found in [13]. In general the agreement is quite good.

In Figure 3 we can see that while the MMI projection produced in this study has the same qualitative features as the MMI projection found in [13], there are some differences.

The results are similar in that we see the same general structure is formed between the clusters of corresponding classes. In addition, when comparing the classification accuracy with the LVQ classifier it was found that the accuracy that I obtained and

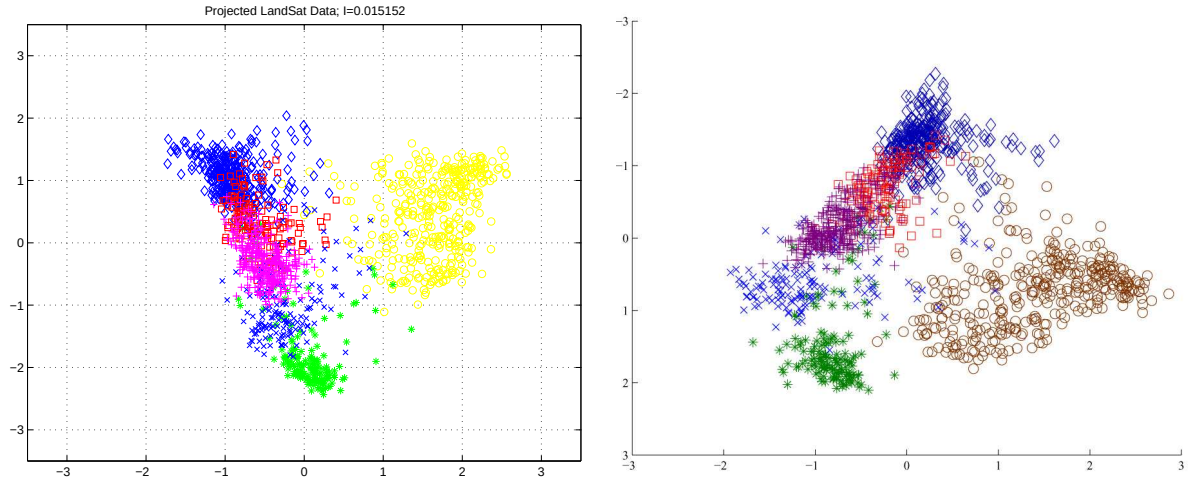


Figure 3: **Left:** The MMI projection of the LandSat data obtained during the course of this investigation. **Right:** The MMI projection results from reference [13]. For possible reasons for the slight differences please see the main text.

obtained in [13] were comparable, see below. The results might be different because of several factors.

- The MMI projection presented in [13] was the result of the mutual information maximization over a random set of 1500 instances drawn from the initial training set of size (∼4400). Is very unlikely that the samples used here in computing the MMI projection were the same as those used in [13].
- The MMI projection presented in [13] were the results of selecting the final projection with the largest mutual information. Several random initial conditions were used to derive the final projection. In particular, maximal mutual information projections were computed with the initial projection corresponding to PCA, MDA, and three random projections. This random initial condition start makes it difficult to compare the exact projections.
- As mentioned at the end of subsection 3.3 there exist several techniques to carry out the required maximization of these several readily suggest themselves. They are gradient ascent (implemented in this study), conjugate gradient, Levenberg-Marquet, and stochastic techniques. In the simple gradient ascent method implemented in this study a very simple strategy was used. This had the effect (in addition to possibly selecting a different local maximum) of limiting the accuracy to which maximum could be computed. An adaptive gradient ascent (that modifies the search range of $\eta$ as $\sigma$ changes) would need to be implemented to correct for this deficiency. Different optimization techniques will perform better or worse on depending on the problem they are applied to. Depending on the method used in [13] results could be different. It remains to be shown how the alternative ascent procedure perform.

As a visualization into the performance of the MMI algorithm, in Figure 4 we present the sequence of iterations that the MMI algorithm goes through beginning with an initial projection provided by MDA. One can see that the effect of the MMI maximization is to spread out the clusters that lie on top of each other in the MDA projection. In Figure 5 we present the sequence of iterations that the MMI algorithm goes through beginning with a random initial projection. We can see that very quickly the

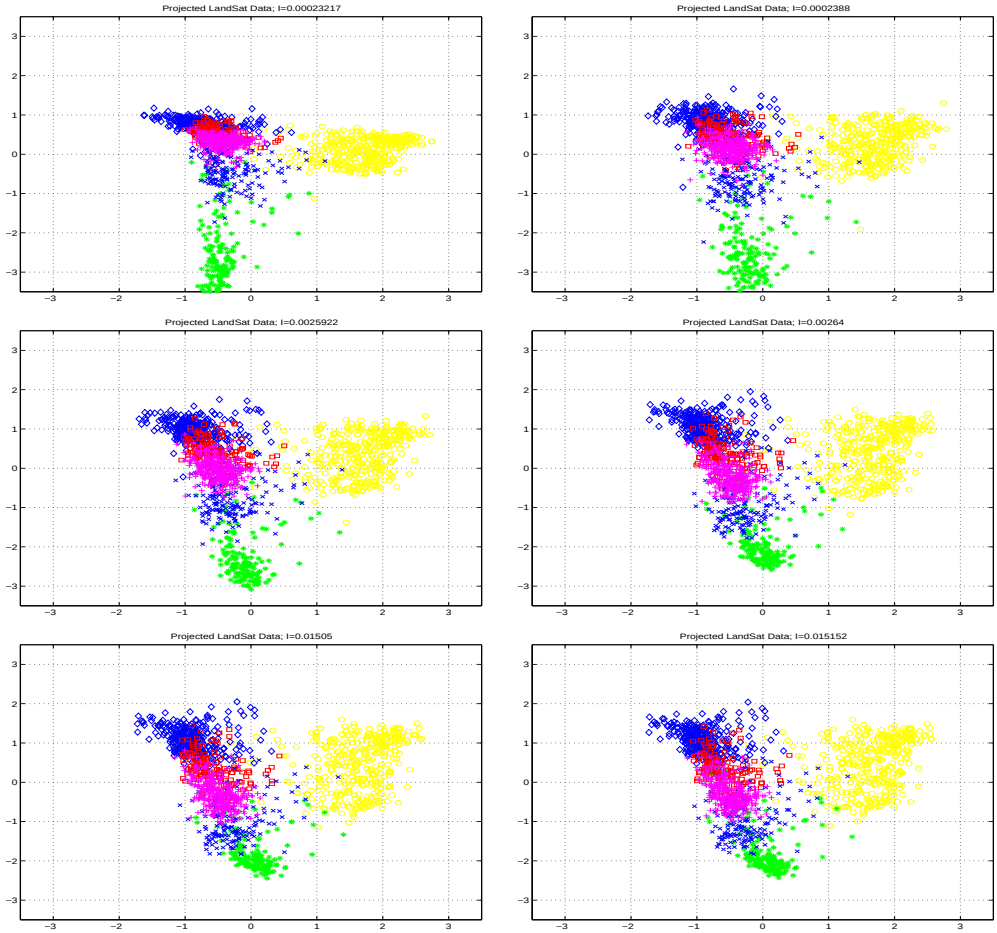


Figure 4: MMI iterations beginning with the MDA projection. The iterations are read from top to bottom, left to right. Note how the MMI projection pulls the classes apart.

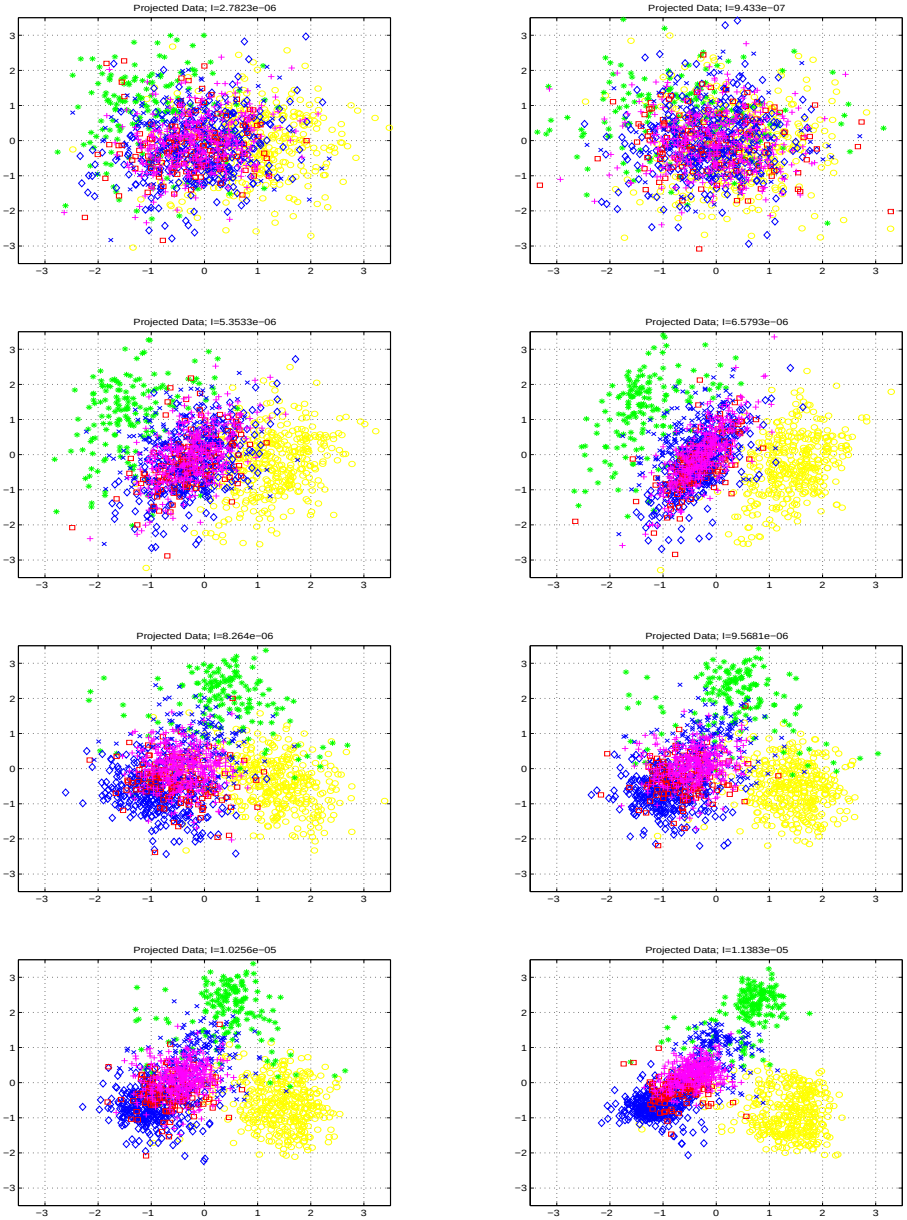


Figure 5: MMI iterations beginning with a random projection. The iterations are read from top to bottom from left to right. The last picture is the 11th iterate.

| Dim: | 2 | 3 |
|---|---|---|
| $\text{MMI}_T$ | 75.7 | 86.2 |
| $\text{MMI}_W$ | 79.0 | 80.5 |

Table 1: A comparison of LVQ classification accuracies under MMI projection. The row titled $\text{MMI}_T$ are results taken from [13]. The row titled $\text{MMI}_W$ were computed in this study.

| Region | Num. entries | d=2 | d=3 |
|---|---|---|---|
| 1 | 461 | 97.83 | 95.88 |
| 2 | 224 | 88.84 | 90.62 |
| 3 | 397 | 60.96 | 71.79 |
| 4 | 211 | 78.67 | 43.13 |
| 5 | 237 | 78.06 | 78.48 |
| 6 | 470 | 71.91 | 85.53 |

Table 2: Individual classification accuracies for the 6 regions in the LandSat data.

MMI projection algorithm is able to find a much better configuration (for classification) of the points in the plane.

### 5.3 LVQ Classification Results with the 2-D Projected Data

In this subsection we present results using the MMI projection and an LVQ classifier to compute classification accuracies under this projection. The LVQ classifier was run with 200 code vectors and the transformed testing data classified. When this was performed the obtained results are presented in Table 1. The classification accuracy of 2 and 3 dimensional projections are comparable. In addition to this information in Table 2 we present the accuracies of the LVQ classifier under projection into the dimensions $d = 2$ and $d = 3$.

## 6 Conclusions

In this paper we have duplicated a subset of the results found in [13]. In that paper a technique for feature projection was presented that relied on the idea of maximization of mutual information between the projected features and the class labels. This technique has been shown to be very promising at finding projections that are optimal with respect to classification. In this project we were able to duplicate a subset of the results presented in [13]. We were able to duplicate the MDA projection and the MMI projection for the Landsat data. In addition, while using a LVQ classifier we were able to duplicate some of the classification accuracy results. It is also concluded that the obtained projection depends to a great extend on the type of ascent technique used. In future work the sensitivity of the MMI projection under different optimization techniques will be investigated.

# A Required Partial Derivatives For Optimization

In this appendix we discuss the implementation of the required derivatives involved in the maximization of mutual information. Specifically this appendix develops in detail some pseudo-code expressing the calculation of the derivatives required in maximizing mutual information.

For the update of the $k$-th component of $\Theta_k$, the individual components in the derivative term in Eq. 42 can be written as

$$\sum_{i=1}^{N} \frac{\partial I}{\partial y_i} \frac{\partial y_i}{\partial W} \frac{\partial W}{\partial \Theta_k} = \sum_{i=1}^{N} \sum_{l=1}^{d} \sum_{m=1}^{D} \sum_{n=1}^{d} \frac{\partial I}{\partial y_{i,l}} \frac{\partial y_{i,l}}{\partial W_{m,n}} \frac{\partial W_{m,n}}{\partial \Theta_k} \tag{43}$$

$$= \sum_{i=1}^{N} \sum_{l=1}^{d} \frac{\partial I}{\partial y_{i,l}} \sum_{m=1}^{D} \sum_{n=1}^{d} \frac{\partial y_{i,l}}{\partial W_{m,n}} \frac{\partial W_{m,n}}{\partial \Theta_k} \tag{44}$$

Here $y_{i,l}$ is the $l$-th component ($l \in 1, \ldots, d$) of the $i$-th transformed vector $y_i$, and $W_{m,n}$ is the $m$-th row and $n$-th column of the transformation matrix $W$. Since the transformed points $y_i$ are given in terms of the $x_i$ via. $y_i = W^t x_i$ by writing out components of $y_i$ as follows (note the transposition of the components of $W$)

$$y_{i,l} = \sum_{m'=1}^{D} W_{m',l} x_{i,m'} = W_{1,l} x_{i,1} + W_{2,l} x_{i,2} + W_{3,l} x_3 + \ldots + W_{D,l} x_D \tag{45}$$

One can arrive at the following expression for the derivative of $y_{i,l}$ with respect to $W_{m,n}$ in terms of the components of $x_i$

$$\frac{\partial y_{i,l}}{\partial W_{m,n}} = x_{i,m} \delta_{n,l} \tag{46}$$

Inserting this into Eq. 43 we obtain

$$\sum_{i=1}^{N} \frac{\partial I}{\partial y_i} \frac{\partial y_i}{\partial W} \frac{\partial W}{\partial \Theta_k} = \sum_{i=1}^{N} \sum_{l=1}^{d} \frac{\partial I}{\partial y_{i,l}} \sum_{m=1}^{D} x_{i,m} \frac{\partial W_{m,l}}{\partial \Theta_k} \tag{47}$$

# References


[1] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley, 2001.

[2] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[3] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

[4] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

[5] I. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.

[6] T. Kohonen. Learning vector quantization. *Neural Networks*, 1((suppl. 1)):303, 1988.

[7] T. Kohonen, T. Kangas, J. Laaksonen, and K. Torkkola. LVQ_PAK. *The learning vector quantization program package version 2.1.* Laboratory of Computer and Information Science, Helsinki University of Technology, 1992. [Version 3.1 became available in 1995].

[8] A. M. Martinez and A. C. Kak. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):228–233, 2001.

[9] P. M. Murphy and D. W. Aha. UCI repository of machine learning databases, 1999.

[10] E. Parzen. On the estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[11] J. Principe, D. Xu, and J. Fisher. Information theoretic learning, 1999.

[12] K. Torkkola. On feature extraction by mutual information maximization, 2002.

[13] K. Torkkola and W. M. Campbell. Mutual information in learning feature transformations. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1015–1022, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[14] K. Torkkola and W. M. Campbell. Mutual information in learning feature transformations. In *Proc. 17th International Conf. on Machine Learning*, pages 1015–1022. Morgan Kaufmann, San Francisco, CA, 2000.