ON COMBINING PRINCIPAL COMPONENTS WITH FISHER'S LINEAR DISCRIMINANTS FOR SUPERVISED LEARNING

Mykola PECHENIZKIY*, Alexey TSYMBAL**, Seppo PUURONEN*

Abstract. "The curse of dimensionality" is pertinent to many learning algorithms, and it denotes the drastic increase of computational complexity and classification error in high dimensions. In this paper, principal component analysis (PCA), parametric feature extraction (FE) based on Fisher's linear discriminant analysis (LDA), and their combination as means of dimensionality reduction are analysed with respect to the performance of different classifiers. Three commonly used classifiers are taken for analysis: *k*NN, Naïve Bayes and C4.5 decision tree. Recently, it has been argued that it is extremely important to use class information in FE for supervised learning (SL). However, LDA-based FE, although using class information, has a serious shortcoming due to its parametric nature. Namely, the number of extracted components cannot be more that the number of classes minus one. Besides, as it can be concluded from its name, LDA works mostly for linearly separable classes only. In this paper we study if it is possible to overcome these shortcomings adding the most significant principal components to the set of features extracted with LDA. In experiments on 21 benchmark datasets from UCI repository these two approaches (PCA and LDA) are compared with each other, and with their combination, for each classifier. Our results demonstrate that such a combination approach has certain potential, especially when applied for C4.5 decision tree learning. However, from the practical point of view the combination approach cannot be recommended for Naïve Bayes since its behavior is very unstable on different datasets.

Keywords: Principal component analysis (PCA), linear discriminant analysis (LDA), feature extraction, supervised learning

* Department of Computer Science and Information Systems, University of Jyväskylä, P.O. Box 35, 40351 Jyväskylä, Finland, (phone: +358-14-2602472; fax: +358-14-2603011; e-mail: {mpechen,sepi}@cs.jyu.fi).

** Department of Computer Science, Trinity College Dublin, College Green, Dublin 2, Ireland (e-mail: Alexey.Tsymbal@cs.tcd.ie).

1. Introduction

Fayyad [5] introduced KDD as "the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data". The process comprises several steps, which involve data selection, data pre-processing, data transformation, application of machine learning techniques, and the interpretation and evaluation of patterns.

In this paper we analyse issues related to data transformation, which needs to be undertaken before applying machine learning techniques [8]. These issues are often considered from two different perspectives. The first one is related to the so-called "curse of dimensionality" problem [3] and the necessity of dimensionality reduction [1]. The second perspective comes from the assumption that in many datasets to be processed some individual features, being irrelevant or indirectly relevant to the purpose of analysis, form a poor problem representation space. This may be the case even when there is no dimensionality problem. The corresponding ideas of constructive induction that assume the improvement of problem representation before application of any learning technique have been presented in [9].

Feature extraction (FE) for classification is aimed at finding such a transformation of the original space in order to produce new features, which would preserve class separability as much as possible [6] and to form a new lower-dimensional problem representation space. Thus, FE accounts for both the perspectives, and, therefore, we believe that FE, when applied either on datasets with high dimensionality or on datasets including indirectly relevant features, can improve the performance of a classifier.

We consider Principal Component Analysis (PCA), which is perhaps the most commonly used FE technique, and class-conditional LDA-based FE in the context of supervised learning (SL). Recently, it has been argued that it is extremely important to use class information in FE for SL. Thus, in [10] it was shown that PCA gives high weights to features with higher variabilities irrespective of whether they are useful for classification or not. However, LDA-based FE, although using class information, also has a serious shortcoming due to its parametric nature. Namely, the number of extracted components cannot be more that the number of classes minus one. Besides, as it can be concluded from its name, LDA works mostly for linearly separable classes only. We discuss the main limitations of PCA and parametric LDA-based FE and search for ways to overcome those limitations. Our approach is to combine linear discriminants (LDs) with principal component (PCs) for SL. Particularly, in this paper we study if it is possible to overcome the shortcomings of both approaches, and construct a better representation space for SL adding the most significant principal components to the set of features extracted with LDA.

We conduct a number of experiments on 21 UCI datasets, analyzing the difference in the impact of these three FE approaches (PCA, LDA and combined use of LDs and PCs) on the classification performance of the nearest neighbour classification, Naïve Bayes, and C4.5 decision tree learning. We compare PCA and LDA with each other, and with their combination for each classifier. The results of these experiments show that linear discriminants (LDs) when combined together with principal components (PCs) can significantly improve the supervised learning process in comparison with the case when either of approaches is used alone for FE. The results support our intuitive reasoning that when PCs are added to LDs the representation space for SL is improved providing more information about variance in data and when LDs are added to PCs the representation space is improved providing more information about data variance within and between classes. Indeed, our results show that such a combination approach has certain potential, especially when applied for C4.5 decision tree learning. However, from the practical point of view the combination approach cannot be recommended for Naïve Bayes since its behaviour is very unstable on different datasets.

To the best of our knowledge, this is the first attempt to analyse the combination of Fisher's linear discriminants and principal components.

The rest of the paper is organised as follows. In Section 2 we first consider PCA (Section 2.1) and class-conditional LDA-based FE (Section 2.2) in the context of SL. Then, Section 2.3 introduces the combination of PCA and LDA-based FE approaches. In Section 3 we consider our experiments and present the main results. In Section 4 we summarise our work with the main conclusions and further research directions.

2. Feature Extraction for Supervised Learning

Generally, FE for classification can be seen as a search process among all possible transformations of the original feature set for the best one, which preserves class separability as much as possible in the space with the lowest possible dimensionality [6]. In other words, having n independent observations of the variables $\mathbf{x}^T = (x_1,...,x_d)$ centred about the mean, we are interested in finding a projection **w**:

$$\mathbf{y} = \mathbf{w}^T\mathbf{x} \tag{1}$$

where **y** is a $k \times 1$ transformed data point (presented by k features), **w** is a $d \times k$ transformation matrix, and **x** is a $d \times 1$ original data point (presented by d features).

2.1. Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a well-known statistical method, which extracts a lower dimensional space by analyzing the covariance structure of multivariate statistical observations [7].

The main idea behind PCA is to determine the features that explain as much of the total variation in the data as possible with as few features as possible. The computation of the PCA transformation matrix is based on the eigenvalue decomposition of the covariance matrix **S** (formula 2 below) and therefore is rather expensive computationally:

$$\mathbf{w} \leftarrow eig\_decomposition\left(\mathbf{S} = \sum_{i=1}^{n}(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T\right) \tag{2}$$

where n is the number of instances, $\mathbf{x}_i$ is the i-th instance, and **m** is the mean vector of the input data: $\mathbf{m} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$.

Computation of the PCs can be presented with the following algorithm:

1. Calculate the covariance matrix **S** from the input data.
2. Compute the eigenvalues and eigenvectors of **S** and sort them in a descending order with respect to the eigenvalues.
3. Form the actual transition matrix by taking the predefined number of components (eigenvectors).
4. Finally, multiply the original feature space with the obtained transition matrix, which yields a lower-dimensional representation.

The necessary cumulative percentage of variance explained by the principal axes is used commonly as a threshold, which defines the number of components to be chosen.

PCA has been perhaps one of the most popular FE techniques because of its many appealing properties. Thus, PCA maximises the variance of the extracted features and the extracted features are uncorrelated. It finds best linear approximation in the mean-square sense, because the truncation error is the sum of the lower eigenvalues. Information that is contained in the extracted features is maximised. Besides this the model parameters for PCA can be computed directly from the data, for example by diagonalizing the sample covariance of the data. For more detailed description of PCA and its properties see, for example [16].

In [10] it was shown that PCA has an important drawback, namely the conventional PCA gives high weights to features with higher variabilities irrespective of whether they are useful for classification or not. This may give rise to the situation where the chosen principal component corresponds to the attribute with the highest variability but without any discriminating power (compare two different situations in Figure 1).

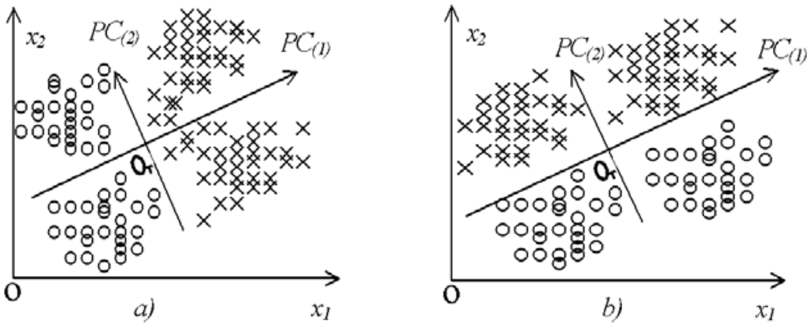


Figure 1. PCA for classification: a) effective work of PCA, b) the case where an irrelevant PC was chosen from the classification point of view (O denotes the origin of initial feature space *x_1*, *x_2* and O_T – the origin of transformed feature space *$PC_{(1)}$*, *$PC_{(2)}$*).

2.2. Class-Conditional Eigenvector-Based FE

A usual approach to overcome the above problem is to use some class separability criterion [2], e.g. the criteria defined in Fisher's linear discriminant analysis and based on the family of functions of scatter matrices:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \tag{3}$$

where $\mathbf{S}_B$ in the parametric case is the between-class covariance matrix that shows the scatter of the expected vectors around the mixture mean, and $\mathbf{S}_W$ is the within-class covariance, that shows the scatter of samples around their respective class expected vectors:

$$\mathbf{S}_W = \sum_{i=1}^{c} n_i \sum_{j=1}^{n_i} (\mathbf{x}_j^{(i)} - \mathbf{m}^{(i)})(\mathbf{x}_j^{(i)} - \mathbf{m}^{(i)})^T \tag{4}$$

$$\mathbf{S}_B = \sum_{i=1}^{c} n_i (\mathbf{m}^{(i)} - \mathbf{m})(\mathbf{m}^{(i)} - \mathbf{m})^T \tag{5}$$

where c is the number of classes, n_i is the number of instances in a class i, $\mathbf{x}_j^{(i)}$ is the j-th instance of i-th class, $\mathbf{m}^{(i)}$ is the mean vector of the instances of i-th class, and $\mathbf{m}$ is the mean vector of all the input data.

The total covariance matrix shows the scatter of all samples around the mixture mean. It can be shown analytically that this matrix is equal to the sum of the within-class and between-class covariance matrices [6]. In this approach the objective is to maximise the distance between the means of the classes while minimizing the variance within each class. A number of other criteria were proposed by Fukunaga [6].

The criterion (4) is optimised using the simultaneous diagonalization algorithm [6]. The basic steps of the algorithm include eigenvalues decomposition of $\mathbf{S}_W$; transformation of original space to intermediate $\mathbf{x}_W$ (whitening); calculation of $\mathbf{S}_B$ in $\mathbf{x}_W$; the eigenvalue decomposition of $\mathbf{S}_B$ and then transformation matrix $\mathbf{w}$ finally can be produced by simple multiplication:

$$\begin{aligned} &\mathbf{w}_{S_W} \leftarrow eig\_decomposition(S_W), \quad x_W = \mathbf{w}_{S_W} x; \\ &\mathbf{w}_{S_B} \leftarrow eig\_decomposition(S_B \mid x_W) \\ &\mathbf{w} = \mathbf{w}_{S_W} \mathbf{w}_{S_B} \end{aligned} \tag{6}$$

It should be noticed that there is a fundamental problem with the parametric nature of the covariance matrices. The rank of $\mathbf{S}_B$ is at most the *number of classes-1*, and hence no more than this number of new features can be obtained.

Some attempts have been done to overcome the shortcomings of LDA-based FE. One approach is to increase the number of degrees of freedom in the between-class covariance matrix, measuring the between-class covariances on a local basis. The *k*-nearest neighbor (*k*NN) technique can be used for this purpose. Such approach is known as nonpaprametric FE [6]. In our previous works (as for example [14, 11, and 13]) we experimentally compared these approaches for one-level supervised learning and for dynamic integration of classifiers. Our results demonstrated that measuring the between-class covariance on a local basis does increase the performance of FE for supervised learning with many datasets. Yet, the nonparametric approach is much more time consuming.

Another way to improve the performance of LDA-based FE is to discover the structure of (possibly heterogeneous) data and apply FE locally for each region (cluster/partition) independently. For example in [12] we applied so-called natural clustering approach to perform local dimensionality reduction also using parametric LDA-based FE.

In this paper we propose another approach which tries to improve the parametric class-conditional LDA by adding a few principal components (PCs). We describe this approach in the following section.

2.3. Combination of Principal Components and Linear Discriminants for Supervised Learning

The basic idea behind combination of PCs with LDs for SL is rather intuitive. The added PCs are selected so that they cover most variance in the data. By adding those PCs it is possible to have more than the number of classes minus one extracted features. The approach is illustrated in Figure 2.

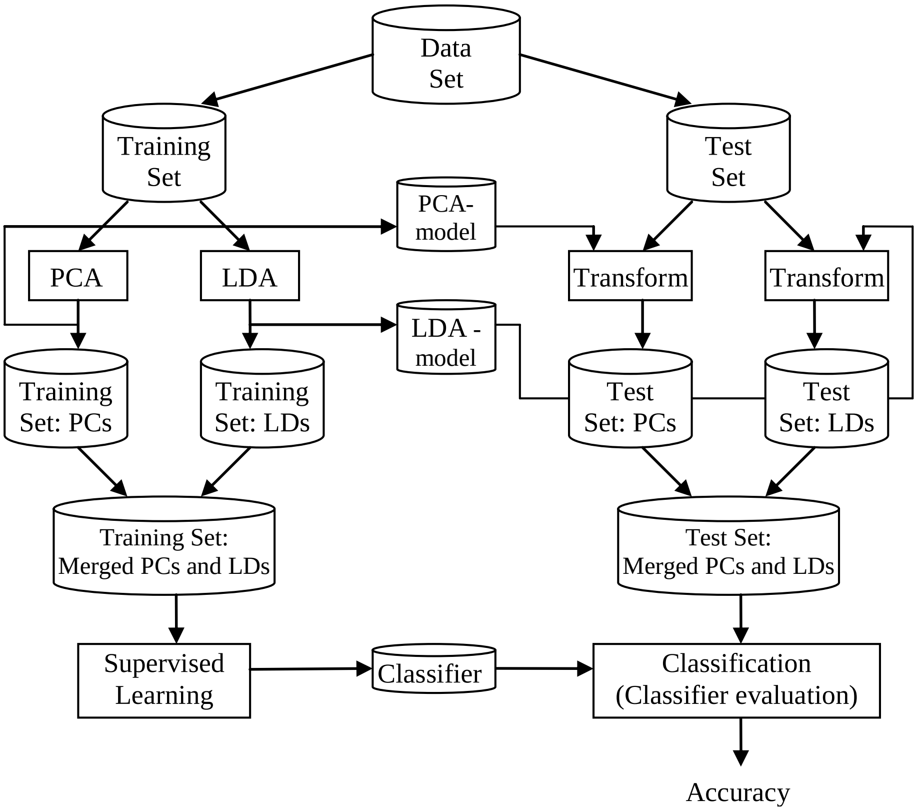


Figure 2. Combining PCs and LDs for SL

A dataset is divided into training and test sets. Both PCA and parametric LDA-based FE (LDA) are applied independently to training data, producing PCA-model and LDA-model (transformation matrices) correspondingly. Then, original data in training set is mapped onto the lower-dimensional spaces (also independently with each approach). Thus, two transformed training sets are produced, one of which contains PCs instead of original features and the other LDs. Then, these transformed (lower-dimensional spaces) datasets are merged, so that the resulting training sets contains both LDs and PCs and class attribute from the original training set. The constructed new representation space is used for learning a classifier.

To be able to evaluate the learnt classifier test set should also be transformed to the same format. This is done in the similar way, so that test set is transformed independently with PCA and LDA models, and PCs and LDs are constructed. Then, LDs are merged with PCs and class attribute from original test set.

3. Experiments and Results

The experiments were conducted on 21 datasets with different characteristics (see Table 1) taken from the UCI machine learning repository [4]. For FE each categorical feature was replaced with a redundant set of binary features, each corresponding to a value of the original feature.

Table 1. Datasets characteristics

Dataset	instances	classes	Features			
			num.	categ.	binarised	num+bin.
Balance	625	3	0	4	20	20
Breast	286	2	0	9	38	38
Car	1728	4	0	6	21	21
Diabetes	768	2	8	0	0	8
Glass	214	6	9	0	0	9
Heart	270	2	13	0	0	13
Ionosphere	351	2	34	0	0	33
Iris Plants	150	3	4	0	0	4
LED	300	10	0	7	7	7
LED17	300	10	0	17	24	24
Liver	345	2	6	0	0	6
Lymph	148	4	3	15	33	36
Monk-1	432	2	0	6	15	15
Monk-2	432	2	0	6	15	15
Monk-3	432	2	0	6	15	15
Tic	958	2	0	9	27	27
Vehicle	846	4	18	0	0	18
Kr-vs-kp	3196	2	0	36	38	38
Waveform	3772	4	7	23	27	34
Vovel	990	11	10	2	16	26
Hypothyroid	5000	3	21	0	0	21

In the experiments, the accuracies of the classifiers produced by 3-nearest neighbour classification (3*NN*), Naïve-Bayesian (NB) learning algorithm, and C4.5 decision tree (C4.5) [15] were calculated. All they are well known in the DM community and represent three different approaches to learning from data. The main motivation to use the 3 different types of classifiers is that we expect different impact of FE on the representation space not only with respect to different datasets but also with respect to different classifiers. In particular, for *k*NN FE can produce a better neigbourhood, for C4.5 it produces better (more informative) individual features, and for Naïve Bayes it produces uncorrelated features.

For each dataset 30 test runs of Monte-Carlo cross validation were made to evaluate classification accuracies with and without FE. In each run, the dataset is first split into the training set and the test set by stratified random sampling to keep class distributions approximately same. Each time 30 percent instances of the dataset are first randomly taken to the test set. The remaining 70 percent instances form the training set, which is used for finding the FE transformation matrix **w**. The test environment was implemented within the WEKA framework (the machine learning library in Java) [15]. The classifiers from this library were used with their default settings.

We took all the features extracted by parametric FE as it was always equal to *number of classes-1*. In the experiments we added 1, 2, and 3 PCs and the number of principal components that cover 85% of variance. We found that patterns of performance were quite similar for all the ways of combining LDs and PCs. Therefore, in this paper we limit our presentation with the case where the first three PCs were added to the set of features extracted with PCA.

The basic accuracy results for each dataset for each of the three classifiers are presented in Table 2. For each classifier three different accuracies are presented: for the case when parametric LDA-based FE was used (denoted as LDA), for the case when PCA was used as FE technique (PCA), and for the case when both LDA and PCA were used to produce new features (denoted as LDA + PCA).

Main results are presented in Table 3 (placed after Figure 3 to prevent the table break) in qualitative form. Each cell contains information whether one method was better than another. Notations used are: '++' denotes a significant win of LDA (or LDA+PCA) according to the paired Student's *t*-test with 0.95 level of significance, '+' denotes win (average improvement more than 0.5%, but without statistical significance) of LDA (or LDA+PCA), '=' denotes tie (average difference less than 0.5%), '–' denotes loss (average decrease of accuracy is more than 0.5%, but non significant according to the paired Student's t-test with 0.95 level of significance) of LDA (or LDA+PCA), and '– –' denotes significant loss of LDA (or LDA+PCA) also according to the paired Student's *t*-test with 0.95 level of significance.

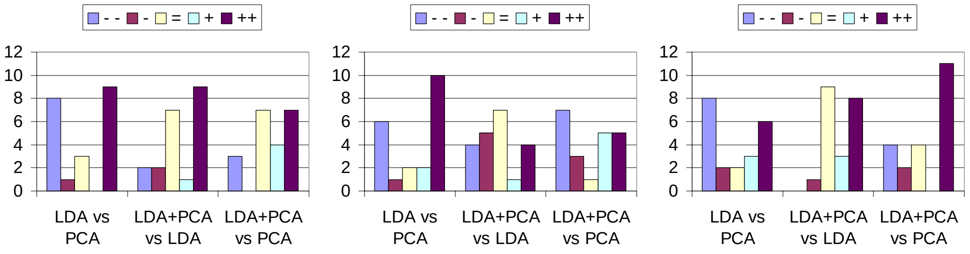
It can be seen from the table that there are many common patterns in the behavior of techniques for the 3 different classifiers, yet there are some differences with some datasets too.

First we tried to analyse how many wins, losses and ties occurred for each pair of compared approaches. These results are presented in Figure 3. It can be seen from the figure that, according to the ranking results, PCA in comparison with LDA works better for C4.5, and LDA is suited better for NB. The accuracy results of 3NN classifier are similar with either PCA or LDA FE. This kind of behaviour was observed and discussed earlier in

[11]. The combination approach work very well with C4.5 classifier, and it is also good with 3NN classifier. However, the combination approach with NB classifier demonstrates surprisingly very poor behaviour.

Table 2. Basic accuracy results

Dataset	3NN			NB			C4.5		
	LDA	PCA	LDA+ PCA	LDA	PCA	LDA+ PCA	LDA	PCA	LDA+ PCA
balance	.881	.707	.829	.901	.804	.890	.903	.668	.902
breast	.715	.686	.702	.755	.723	.737	.745	.697	.743
car	.930	.842	.926	.914	.825	.902	.935	.835	.939
diabetes	.621	.711	.709	.672	.673	.674	.663	.703	.700
glass	.633	.638	.635	.464	.492	.462	.614	.640	.609
heart	.602	.640	.635	.619	.663	.649	.630	.663	.670
ionosphere	.771	.853	.860	.787	.891	.835	.800	.881	.853
iris	.944	.916	1	.903	.894	.979	.931	.930	1
led	.690	.697	.703	.715	.735	.714	.683	.690	.682
led17	.549	.318	.554	.633	.492	.642	.541	.359	.534
liver	.593	.577	.597	.568	.537	.511	.652	.594	.651
lymph	.619	.749	.762	.626	.787	.770	.619	.738	.733
monk1	.681	.962	.946	.720	.751	.716	.740	.896	.854
monk2	.651	.519	.552	.679	.634	.666	.677	.671	.676
monk3	.965	.985	.987	.971	.960	.969	.967	.944	.974
tic	.689	.926	.836	.707	.709	.699	.745	.802	.757
vehicle	.552	.528	.601	.487	.445	.465	.554	.545	.601
kr-vs-kp	.806	.920	.893	.836	.857	.801	.841	.857	.880
waveform	.848	.768	.842	.838	.820	.838	.862	.853	.858
vowel	.914	.927	.920	.667	.538	.664	.780	.754	.789
hypothyroid	.954	.951	.954	.927	.850	.857	.952	.953	.954
average	**.743**	**.753**	**.783**	**.733**	**.718**	**.735**	**.754**	**.746**	**.779**



a) 3NN

b) NB

d) C4.5

Figure 3. Ranking of the FE techniques according to the results on 21 UCI datasets

Table 3. Comparison of accuracy results

Dataset	LDA vs PCA			LDA+PCA vs LDA			LDA+PCA vs PCA		
	3*NN*	NB	C4.5	3*NN*	NB	C4.5	3*NN*	NB	C4.5
Balance	++	++	++	– –	–	=	++	++	++
Breast	++	++	++	–	–	=	+	+	++
Car	++	++	++	–	–	=	+	++	++
Diabetes	– –	=	– –	=	=	++	=	=	=
Glass	=	– –	– –	=	=	=	=	– –	– –
Heart	– –	– –	– –	++	++	++	=	–	++
Ionosphere	– –	– –	– –	++	++	++	+	– –	– –
Iris Plants	++	+	=	++	++	++	++	++	++
LED	=	–	–	+	=	=	=	– –	–
LED17	++	++	–	=	+	–	++	+	++
Liver	++	++	++	=	– –	=	++	– –	++
Lymph	– –	– –	– –	++	++	++	+	–	–
Monk-1	– –	– –	– –	++	=	++	– –	– –	– –
Monk-2	++	++	+	– –	–	=	++	++	=
Monk-3	– –	+	++	++	=	+	=	+	++
Tic	– –	=	– –	++	–	+	– –	–	– –
Vehicle	++	++	+	++	– –	++	++	– –	++
Kr-vs-kp	– –	– –	– –	++	– –	++	– –	– –	++
Waveform	++	++	+	=	=	=	++	+	=
Vovel	–	++	++	=	=	+	=	++	++
Hypothyroid	=	++	=	=	– –	=	=	+	=
++	9	10	6	9	4	8	7	5	11
+	0	2	3	1	1	3	4	5	0
=	3	2	2	7	7	9	7	1	4
-	1	1	2	2	5	1	0	3	2
- -	8	6	8	2	4	0	3	7	4

The effect of combining PCs with LDs on classification accuracy of each supervised learner with regard to the use of only PCs or only LDs has been estimated with the help of state transition diagrams. Each transition from a state to the same state (i.e. no transition or a loop) gives no score to an estimate. Each transition from state “– –“ to “–”, from “–” to “=”, from “=” to “+”, and from “+” to “++” gives score +1; correspondingly, transition from state “++“ to “+”, from “+” to “=”, from “=” to “–”, and from “–” to “– –” gives score -1. Each transition from “– –” to “=”, from “–” to “+”, and from “=” to “++” gives score +2; correspondingly, each transition from “++” to “=”, from “+” to “–”, and from “=” to “– –” gives score -2. Each transition from “– –” to “+” and from “–” to “++” gives score +3; and from “++” to “–” and from “+” to “– –” gives score -3. Finally, each transition from “– –” to “++” gives score +4, and from “++” to “– –” score -4 correspondingly. We summarise this information in Table 4.

Thus, the effect of combining LDs with PCs is estimated according to the following formula:

$$effect = \sum_{i} \sum_{j} score_i \cdot tr_j \ , \qquad (7)$$

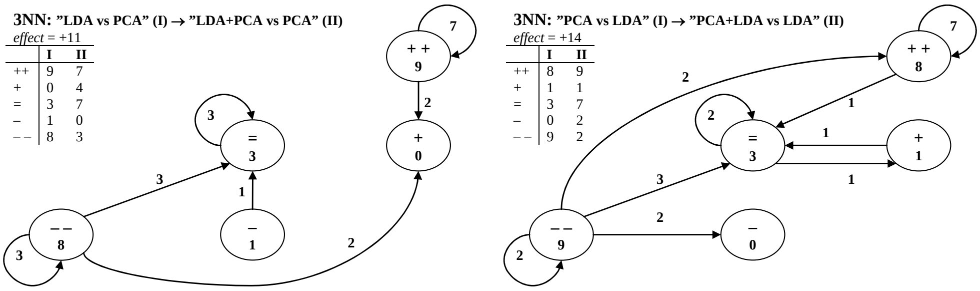
where $score_i$ is the score for the transition from state I to state II and tr_j is the number of such transitions.

Table 4. Scores for transition from state I to state II

state I	– –	++	– –	–	++	+	– –	–	=	++	+	=
state II	++	– –	+	++	–	– –	=	+	++	=	–	– –
score	+4	–4	3	3	–3	–3	2	2	2	–2	–2	–2
state I	– –	–	=	+	++	+	=	–	++	+	–	=
state II	–	=	+	++	+	=	–	– –	++	+	–	=
score	1	1	1	1	–1	–1	–1	–1	0	0	0	0

We present state transition diagrams for each classifier in Figure 4. On the left side we place diagrams that show the gain for a classifier due to the use of PCs with LDs for a new representation space construction instead of only PCs, and on the right side – the gain for a classifier due to use of LDs with PCs instead of only LDs.

In Table 5 we summarise the analysis of diagrams giving the corresponding scores of the effect of merging LDs with PCs for each classifier.



3NN: ”LDA vs PCA” (I) → ”LDA+PCA vs PCA” (II)

effect = +11

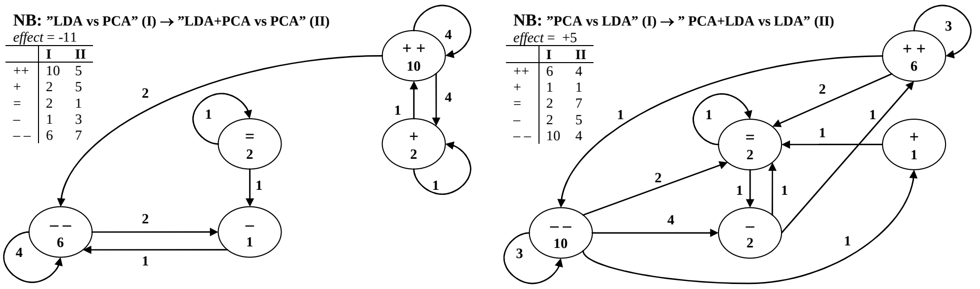
	I	II
++	9	7
+	0	4
=	3	7
–	1	0
– –	8	3

3NN: ”PCA vs LDA” (I) → ”PCA+LDA vs LDA” (II)

effect = +14

	I	II
++	8	9
+	1	1
=	3	7
–	0	2
– –	9	2

Figure 4a. State transition diagrams for 3NN classifier



NB: ”LDA vs PCA” (I) → ”LDA+PCA vs PCA” (II)

effect = -11

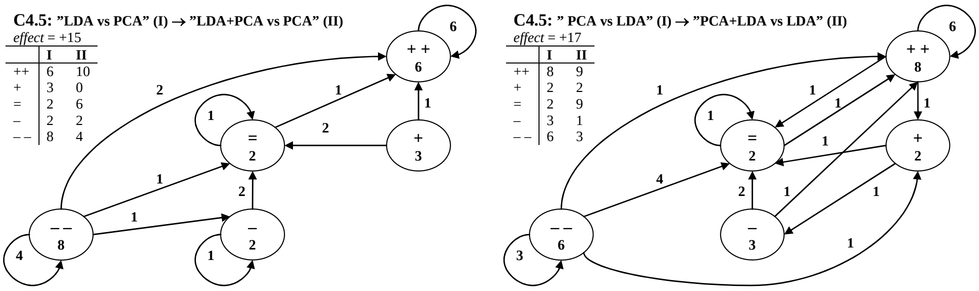
	I	II
++	10	5
+	2	5
=	2	1
–	1	3
– –	6	7

NB: ”PCA vs LDA” (I) → ” PCA+LDA vs LDA” (II)

effect = +5

	I	II
++	6	4
+	1	1
=	2	7
–	2	5
– –	10	4

Figure 4b. State transition diagrams for NB classifier


C4.5: ”LDA vs PCA” (I) → ”LDA+PCA vs PCA” (II)
effect = +15
C4.5: ” PCA vs LDA” (I) → ”PCA+LDA vs LDA” (II)
effect = +17
+ + 6
= 2
+ 3
– – 8
– 2
+ + 8
= 2
+ 2
– – 6
– 3

	I	II
++	6	10
+	3	0
=	2	6
–	2	2
– –	8	4

	I	II
++	8	9
+	2	2
=	2	9
–	3	1
– –	6	3

Figure 4c. State transition diagrams for C4.5 classifier

Figure 4. State transition diagrams for 3NN (a), NB (b) and C4.5 (c) classifiers: from “LDA vs PCA” to “LDA+PCA vs PCA”(on the left side) and from “PCA vs LDA” to “LDA+PCA vs LDA ”(on the right side)

Table 5. Effect of combining PCs with LDs according to the state transition diagrams

	LDA+PCA vs PCA wrt LDA vs PCA	PCA+ LDA vs LDA wrt PCA vs LDA
kNN	+11	+14
NB	-11	+5
C4.5	+15	+17

In Table 6, averaged over the 21 datasets, accuracy results are presented for each classifier. Also we show the averaged increase of accuracy due to the use of both LDs and PCs with respect to PCA and with respect to LDA. It can be seen from the table that the combination approach significantly outperforms the use of LDs or PCs alone.

Table 6. Averaged over 21 datasets accuracy results

	PCA	LDA	LDA+PCA	LDA+PCA vs PCA	LDA+PCA vs LDA
kNN	.753	.743	.783	3.0%	4.0%
NB	.718	.733	.735	1.7%	.2%
C4.5	.746	.754	.779	3.3%	2.5%

In Figure 5 we visualise these averaged accuracies with a histogram.

However, from the practical point of view we are especially interested in those cases where the use of both LDs and PCs together outperforms both a supervised learning algorithm that uses only LDs and the same learning algorithm that uses only PCs to construct a model (e.g. the case with Iris dataset with any classifier or the case with Vehicle for 3NN and C4.5). We present the corresponding results for 3NN, NB and C4.5 in Figure 6. (Significant) loss is counted if either PCA or LDA outperforms the combination approach. And a tie is counted if either PCA or LDA has a tie with the combination approach even if one approach loses (significantly or not) to the combination approach.

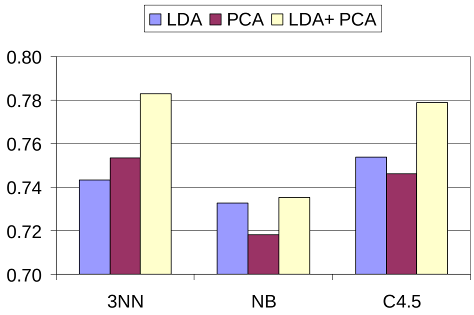


Figure 5. The accuracy of classifiers averaged over 21 datasets.

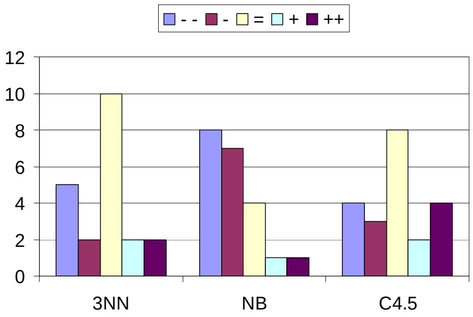


Figure 6. When combination of PCs and LDs is practically useful.

It can be seen from the figure that from the practical point of view the use of both LDs and PCs for supervised learning is safer and more beneficial for C4.5 in comparison with NB and 3NN. For this learning algorithm only the number of wins is similar to the number of losses. On the contrary, for 3NN there were two significant wins against five significant losses. And for NB the situation is ever worse, since there are only one significant and one non significant wins against seven non significant and 8 significant losses. Therefore the combination approach cannot be recommended for use with NB.

It is very interesting to analyse why the behaviour is so different for considered classifiers. Our hypothesis for the better behaviour with C4.5 is that decision trees use inherent feature selection, and thus implicitly select LDs and/or PCs, that are useful for classification, from the combined set of features, discarding the less relevant and duplicate ones. Moreover, this feature selection is local, based on the recursive partitioning principle of decision tree construction.

4. Conclusions

“The curse of dimensionality” is a serious problem for machine learning and data mining especially with present-day multi-feature datasets. Classification accuracy decreases and processing time increases dramatically in high dimensions literally with any learning technique, which was demonstrated with a number of experiments and analytical studies. FE is a common way to cope with this problem. Before applying a learning algorithm the space of instances is transformed into a new space of a lower dimensionality, trying to preserve the distances among instances and class separability.

A classical approach (that takes into account class information) is Fisher’s LDA, which tries to minimise the within class covariance and to maximise the between class covariance in the extracted features. This approach is well studied, and commonly used, however it has one serious drawback. It has a parametric nature, and it extracts no more than the number of classes minus one features. It works well with linearly separable classes, and often provides informative features for classification in more complex problems.

However, due to the limited number of extracted features, it often fails to provide reasonably good classification accuracy even with fairly simple datasets, where the intrinsic dimensionality exceeds that number. A number of ways has been considered to solve this problem. Many approaches suggest non-parametric variations of LDA, which lead to greater numbers of extracted features.

In this paper we consider an alternative way to improve LDA-based FE for classification, which consists of combining the extracted LDs with a few PCs. Our experiments with the combination of LDs with PCs have demonstrated that the discriminating power of LDA features can be improved by PCs for many datasets and learning algorithms. The best performance is exhibited with C4.5 decision trees. A possible explanation for the good behaviour with C4.5 is that decision trees use implicit local feature selection, and thus implicitly select LDs and/or PCs, useful for classification, from the combined set of features, discarding the less relevant and duplicate ones.

This hypothesis and the fact that every dataset needs a different FE technique(s) suggest a direction for future work. It would be interesting to consider a combination of the presented approach with explicit automatic feature selection (filter or wrapper-based), which could lead to an increase in accuracy and better FE through selection of a set of appropriate features of different nature. Local feature selection might be especially useful.

Another important direction for future work is the evaluation of this technique with real-world multidimensional datasets of different nature including images and texts.

Acknowledgments: We would like to thank the UCI ML repository of databases, domain theories and data generators for the datasets, and the WEKA ML library in Java for the source code used in this study. This research is partly supported by the COMAS Graduate School of the University of Jyväskylä, Finland. This material is based also upon works supported by the Science Foundation Ireland under Grant No. S.F.I.-02IN.1I111.

References

[1] Aivazyan S.A., *Applied statistics: classification and dimension reduction,* Finance and Statistics, Moscow, 1989.

[2] Aladjem M., Multiclass discriminant mappings, *Signal Processing*, **35**, 1994, 1-18.

[3] Bellman R., *Adaptive Control Processes: A Guided Tour*, Princeton University Press, 1961.

[4] Blake C.L., Merz C.J., *UCI Repository of Machine Learning Databases*, Dept. of Information and Computer Science, University of California, Irvine CA, 1998.

[5] Fayyad U.M., Data Mining and Knowledge Discovery: Making Sense Out of Data, *IEEE Expert*, **11**, **5**, 1996, 20-25.

[6] Fukunaga K., *Introduction to statistical pattern recognition,* Academic Press, London 1990.

[7] Jolliffe I.T., *Principal Component Analysis*, Springer, New York, NY, 1986.

[8] Liu H., *Feature Extraction, Construction and Selection: A Data Mining Perspective,* ISBN 0-7923-8196-3, Kluwer Academic Publishers, 1998.

[9] Michalski R.S., Seeking Knowledge in the Deluge of Facts, *Fundamenta Informaticae*, **30**, 1997, 283-297.

[10] Oza N.C., Tumer K., *Dimensionality Reduction Through Classifier Ensembles*, Technical Report NASA-ARC-IC-1999-124, Computational Sciences Division, NASA Ames Research Center, Moffett Field, CA, 1999.

[11] Pechenizkiy M., Impact of the Feature Extraction on the Performance of a Classifier: kNN, Naïve Bayes and C4.5, in: B.Kegl, G.Lapalme (eds.), *Proc. of 18th CSCSI Conference on Artificial Intelligence AI'05*, LNAI 3501, Springer Verlag, 2005, 268-279.

[12] Pechenizkiy M., Tsymbal A., Puuronen S., Local Dimensionality Reduction within Natural Clusters for Medical Data Analysis, in *Proc. 18th IEEE Symp. on Computer-Based Medical Systems CBMS'2005*, IEEE CS Press, 2005, 365-37.

[13] Tsymbal A., Pechenizkiy M., Puuronen S., Patterson D.W., Dynamic integration of classifiers in the space of principal components, in: L.Kalinichenko, R.Manthey, B.Thalheim, U.Wloka (eds.), *Proc. Advances in Databases and Information Systems: 7th East-European Conf. ADBIS'03*, LNCS 2798, Heidelberg: Springer-Verlag, 2003, 278-292.

[14] Tsymbal A., Puuronen S., Pechenizkiy M., Baumgarten M., Patterson D., Eigenvector-based feature extraction for classification, in: *Proc. 15th Int. FLAIRS Conference on Artificial Intelligence*, Pensacola, FL, USA, AAAI Press, 2002, 354-358.

[15] Witten I. and Frank E., *Data Mining: Practical machine learning tools with Java implementations*, Morgan Kaufmann, San Francisco, 2000.

[16] William D.R., Goldstein M., *Multivariate Analysis. Methods and Applications*, ISBN 0-471-08317-8, John Wiley & Sons, 1984.

[17] Quinlan, J.R., *C4.5 programs for machine learning*, San Mateo CA: Morgan Kaufmann, 1993.