Exam Assignment

Machine Learning (BSc Data Science) Fall 2023,
IT University of Copenhagen

1 Introduction and formalities

This is the project description for the exam project in the Machine Learning course for the BSc program in Data Science at the IT University of Copenhagen. The project must be submitted electronically via LearnIT no later than 14.00 on 3rd January 2024.

Groups. You need to work in groups of 2–3 people for this project. Please register as a group in LearnIT by 14.00 on Thursday 23th November. The course manager reserves the right to modify the grouping if necessary. Only one person for each group should submit the project.

1.1 What should be handed in?

You must hand in both a report and the source code you have developed during the project. Note that the project's evaluation will be based almost entirely on the report; the source code should be seen as a supporting document.

Make sure to use correct references to works of other people in your report, including references to any of the course textbooks. This also applies to code; if you copy or take inspiration from code developed by other people, this should be stated clearly in your report. Note that any work based on a previous exam submission for this course – even if it is your own work – needs to be clearly marked as such and cited.

Report. The report should be submitted as a single PDF file. There is a strict limit of 15 pages, including figures, tables, code snippets, references, and appendixes, but excluding the front page. The project must be typeset with at least 11pt font size and margins of at least 2 cms. The report must be in PDF format and have a front page that meets the ITU requirements.[1]

Implementation and code. Your implementation has to be in Python. The code must be handed in as a single file (either a zip or tar archive). Except where explicitly stated, there are no restrictions as to which Python libraries you may use.

Your code should be organised such that it is easy to read, i.e. you have to use descriptive names for files, functions, variables, etc. The code may be organised in regular Python source files (.py files) or Jupyter notebooks.

[1] Found at `https://itustudent.itu.dk/study-administration/exams/submitting-written-work`

2 Problem and data set

In this project we will explore different methods for determining the type of clothing from an image of the item.

The data for the project consists of 15,000 labelled images of clothing based on images from the Zalando website (Xiao et al., 2017). Each image is a grayscale 28x28 picture of either a t-shirt/top, trousers, a pullover, a dress, or a shirt (see Figure 1).

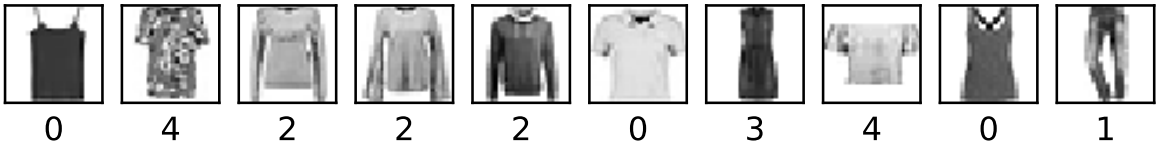

Figure 1: A random sample of 10 images from the training dataset.

Table 1: Categories of clothes

Type of clothing	T-shirt/top	Trouser	Pullover	Dress	Shirt
Label	0	1	2	3	4

The images are divided into a training set of 10,000 images and a test set of 5,000 images. The images and associated labels are available in NPY format as:

`fashion_train.npy` and `fashion_test.npy`.

Each line describes a piece of clothing. The first 784 columns are the pixel values of the 28x28 grayscale image, each taking an integer value between 0 and 255. The last column, number 785, is the category of clothing and takes values in $\{0, 1, 2, 3, 4\}$ (see Table 1).

3 Scientific requirements to the project

This project aims to investigate methods for determining the type of clothing from an image. You should carry out and report an analysis of the fashion data while making sure that you cover all of the tasks set out below.

3.1 Visualisation and exploratory data analysis

In this first part of the project, you will get acquainted with the data through principal component analysis and linear discriminant analysis. You have seen already linear discriminant analysis in the context of classification, but indeed it is also a popular method for dimensionality reduction (see Appendix A for details).

The minimal requirements to your analysis are:

- The report should contain at least one scatterplot of data projected to the first two principal components, and one scatterplot of data projected to the first two linear discriminant variables. Please include a thorough comparison of these two plots.
- The LDA dimensionality reduction should be implemented from scratch using only Python standard libraries and the numerical libraries NumPy and SciPy. However, you are highly encouraged to use the scikit-learn library to reproduce and validate the results of your own implementation.
- Give an interpretation of the first few principal components in terms of how each of these directions correspond to a specific transformation of an image.

3.2 Classification

Regarding the problem of determining the type of clothing from an image of the item, you should explore at least three classification methods:

- A Naive Bayes classifier based on non-parametric estimates of the marginal class conditionals as described in Appendix B. The classifier should use the first two linear discriminant variables as the features.
- A further two classification methods of your own choice.

The Naive Bayes classifier should be implemented from scratch (training and prediction) using only Python standard libraries and the numerical libraries NumPy and SciPy.

Thus, for the Naive Bayes classifier you cannot use machine learning libraries such as scikit-learn, TensorFlow, PyTorch, or Keras. Note that the restriction only applies to the implemenation of the method itself (training and prediction), but not the further interpretations of the results, such as visualisations.

For the classification methods of your own choice, you may use any library you wish with no restrictions.

You are recommended to consider applying some kind of feature scaling to the pixel values as part of your analyses.

3.3 Report

Exploratory data analysis and visualization of the data. Your report should carefully introduce the reader to the data by illustrating selected aspects of the data. Remember to include some comments on what the reader can learn from the visualizations and other things you present.

Details on machine learning methods. For each method please make sure to include

- A brief technical introduction to the method in general.
- A thorough description of how you applied the method to the data, including all details needed for an independent reproduction of your results. This also includes a description of how you have gone about selecting any hyperparameters for the method.

Details on implementations. Your report should describe in detail how you have implemented the linear discriminant analysis and the naive Bayes classifier. Note that the report should be self-contained, and that the assessment of your implementation is based on your description in the report. Please include a discussion of how you have asserted the correctness of your implementations.

Interpretation and discussion of the results. Your report should include a thorough discussion of the performance of each of the methods applied. In particular, you should compare the methods' performance and guide the reader in interpreting the results. Use your expert knowledge to explain the results; for instance, why do particular methods perform better than others?

Appendix A Linear discriminant analysis for dimensionality reduction

Linear discriminant analysis (LDA) is sometimes used also as a dimensionality reduction method. Where the PCA produces principal components, the LDA produces a set of most discriminative directions. Where PCA is *unsupervised* and finds directions of maximal variance in data, LDA is *supervised* and uses the class information to find a view—a projection—that spreads out the class means as much as possible while minimizing the scatter around the means. The new variables are often called *linear discriminants*; we may refer to them instead as *discriminant variables* not to confuse them with linear discriminant functions in classification.

It can be shown that **there are at most $\min\{p, k-1\}$ discriminant variables** for data with p features and k classes.

A detailed description of how to compute the discriminant variables can be found in Section 6.8 of Alpaydin (2014), included below for convenience.

The method `LinearDiscriminantAnalysis` from `sklearn` can be used to train an LDA classifier. The associated `transform` method gives the data projected to the discriminant variables. While you need to make your own implementation, it can be useful to compare results to assert correctness of your implementation.

mon stress and is defined as

$$E(\theta|\mathcal{X}) = \sum_{r,s} \frac{(\|\mathbf{z}^r - \mathbf{z}^s\| - \|\mathbf{x}^r - \mathbf{x}^s\|)^2}{\|\mathbf{x}^r - \mathbf{x}^s\|^2}$$

$$= \sum_{r,s} \frac{(\|\mathbf{g}(\mathbf{x}^r|\theta) - \mathbf{g}(\mathbf{x}^s|\theta)\| - \|\mathbf{x}^r - \mathbf{x}^s\|)^2}{\|\mathbf{x}^r - \mathbf{x}^s\|^2} \quad (6.37)$$

One can use any regression method for $\mathbf{g}(\cdot|\theta)$ and estimate θ to minimize the stress on the training data $\mathcal{X}$. If $\mathbf{g}(\cdot)$ is nonlinear in $\mathbf{x}$, this will then correspond to a nonlinear dimensionality reduction.

In the case of classification, one can include class information in the distance (see Webb 1999) as

$$d'_{rs} = (1-\alpha)d_{rs} + \alpha c_{rs}$$

where c_{rs} is the "distance" between the classes $\mathbf{x}^r$ and $\mathbf{x}^s$ belong to. This interclass distance should be supplied subjectively and α is optimized using cross-validation.

6.8 Linear Discriminant Analysis

LINEAR DISCRIMINANT ANALYSIS

Linear discriminant analysis (LDA) is a supervised method for dimensionality reduction for classification problems. We start with the case where there are two classes, then generalize to $K > 2$ classes.

Given samples from two classes C_1 and C_2, we want to find the direction, as defined by a vector $\mathbf{w}$, such that when the data are projected onto $\mathbf{w}$, the examples from the two classes are as well separated as possible. As we saw before,

$$z = \mathbf{w}^T\mathbf{x} \quad (6.38)$$

is the projection of $\mathbf{x}$ onto $\mathbf{w}$ and thus is a dimensionality reduction from d to 1.

$\mathbf{m}_1$ and m_1 are the means of samples from C_1 before and after projection, respectively. Note that $\mathbf{m}_1 \in \Re^d$ and $m_1 \in \Re$. We are given a sample $\mathcal{X} = \{\mathbf{x}^t, r^t\}$ such that $r^t = 1$ if $\mathbf{x}^t \in C_1$ and $r^t = 0$ if $\mathbf{x}^t \in C_2$.

$$m_1 = \frac{\sum_t \mathbf{w}^T\mathbf{x}^t r^t}{\sum_t r^t} = \mathbf{w}^T\mathbf{m}_1$$

$$m_2 = \frac{\sum_t \mathbf{w}^T\mathbf{x}^t (1-r^t)}{\sum_t (1-r^t)} = \mathbf{w}^T\mathbf{m}_2 \quad (6.39)$$

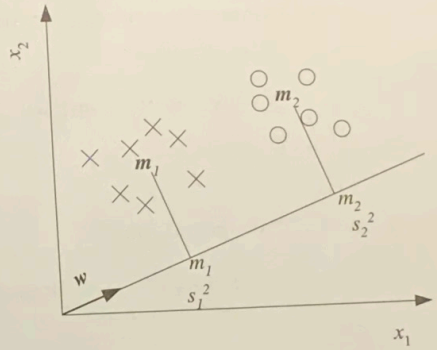

Figure 6.10 Two-dimensional, two-class data projected on $\mathbf{w}$.

SCATTER

The *scatter* of samples from C_1 and C_2 after projection are

$$s_1^2 = \sum_t (\mathbf{w}^T\mathbf{x}^t - m_1)^2 r^t$$

$$s_2^2 = \sum_t (\mathbf{w}^T\mathbf{x}^t - m_2)^2 (1-r^t) \quad (6.40)$$

After projection, for the two classes to be well separated, we would like the means to be as far apart as possible and the examples of classes be scattered in as small a region as possible. So we want $|m_1 - m_2|$ to be large and $s_1^2 + s_2^2$ to be small (see figure 6.10). *Fisher's linear discriminant* is $\mathbf{w}$ that maximizes

FISHER'S LINEAR DISCRIMINANT

$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} \quad (6.41)$$

Rewriting the numerator, we get

$$(m_1 - m_2)^2 = (\mathbf{w}^T\mathbf{m}_1 - \mathbf{w}^T\mathbf{m}_2)^2$$

$$= \mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T\mathbf{w}$$

$$= \mathbf{w}^T\mathbf{S}_B\mathbf{w} \quad (6.42)$$

BETWEEN-CLASS SCATTER MATRIX

where $\mathbf{S}_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T$ is the *between-class scatter matrix*. The

denominator is the sum of scatter of examples of classes around their means after projection and can be rewritten as

$$\begin{aligned} s_1^2 &= \sum_t (\mathbf{w}^T\mathbf{x}^t - m_1)^2 r^t \\ &= \sum_t \mathbf{w}^T(\mathbf{x}^t - \mathbf{m}_1)(\mathbf{x}^t - \mathbf{m}_1)^T\mathbf{w} r^t \\ &= \mathbf{w}^T\mathbf{S}_1\mathbf{w} \end{aligned} \tag{6.43}$$

where

$$\mathbf{S}_1 = \sum_t r^t(\mathbf{x}^t - \mathbf{m}_1)(\mathbf{x}^t - \mathbf{m}_1)^T \tag{6.44}$$

WITHIN-CLASS SCATTER MATRIX

is the *within-class scatter matrix* for C_1. $\mathbf{S}_1/\sum_t r^t$ is the estimator of Σ_1. Similarly, $s_2^2 = \mathbf{w}^T\mathbf{S}_2\mathbf{w}$ with $\mathbf{S}_2 = \sum_t (1-r^t)(\mathbf{x}^t - \mathbf{m}_2)(\mathbf{x}^t - \mathbf{m}_2)^T$, and we get

$$s_1^2 + s_2^2 = \mathbf{w}^T\mathbf{S}_W\mathbf{w}$$

where $\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2$ is the total within-class scatter. Note that $s_1^2 + s_2^2$ divided by the total number of samples is the variance of the pooled data. Equation 6.41 can be rewritten as

$$J(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} = \frac{|\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)|^2}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} \tag{6.45}$$

Taking the derivative of J with respect to $\mathbf{w}$ and setting it equal to 0, we get

$$\frac{\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} 2\left((\mathbf{m}_1 - \mathbf{m}_2) - \frac{\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}}\mathbf{S}_W\mathbf{w}\right) = 0$$

Given that $\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)/\mathbf{w}^T\mathbf{S}_W\mathbf{w}$ is a constant, we have

$$\mathbf{w} = c\mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \tag{6.46}$$

where c is some constant. Because it is the direction that is important for us and not the magnitude, we can just take $c = 1$ and find $\mathbf{w}$.

Remember that when $p(\mathbf{x}|C_i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \Sigma)$, we have a linear discriminant where $\mathbf{w} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$, and we see that Fisher's linear discriminant is optimal if the classes are normally distributed. Under the same assumption, a threshold, w_0, can also be calculated to separate the two classes. But Fisher's linear discriminant can be used even when the classes are not normal. We have projected the samples from d dimensions to 1,

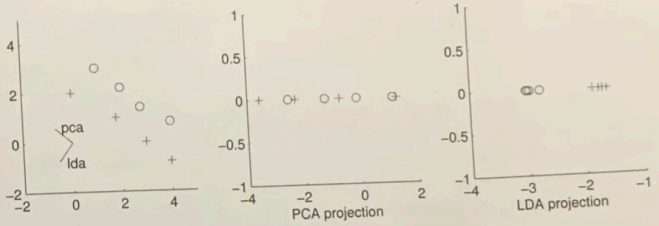

Figure 6.11 Two-dimensional synthetic data, directions found by PCA and LDA and projections along these directions are shown. LDA uses class information and as expected, does a much better job in terms of class separation.

and any classification method can be used afterward. In figure 6.11, we see two-dimensional synthetic data with two classes. As we see, and as expected, because it uses the class information, LDA direction is superior to the PCA direction in terms of the ease of discrimination afterwards.

In the case of $K > 2$ classes, we want to find the matrix $\mathbf{W}$ such that

$$\mathbf{z} = \mathbf{W}^T\mathbf{x} \tag{6.47}$$

where $\mathbf{z}$ is k-dimensional and $\mathbf{W}$ is $d \times k$. The within-class scatter matrix for C_i is

$$\mathbf{S}_i = \sum_t r_i^t(\mathbf{x}^t - \mathbf{m}_i)(\mathbf{x}^t - \mathbf{m}_i)^T \tag{6.48}$$

where $r_i^t = 1$ if $\mathbf{x}^t \in C_i$ and 0 otherwise. The total within-class scatter is

$$\mathbf{S}_W = \sum_{i=1}^K \mathbf{S}_i \tag{6.49}$$

When there are $K > 2$ classes, the scatter of the means is calculated as how much they are scattered around the overall mean

$$\mathbf{m} = \frac{1}{K}\sum_{i=1}^K \mathbf{m}_i \tag{6.50}$$

and the between-class scatter matrix is

$$\mathbf{S}_B = \sum_{i=1}^K N_i(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T \tag{6.51}$$

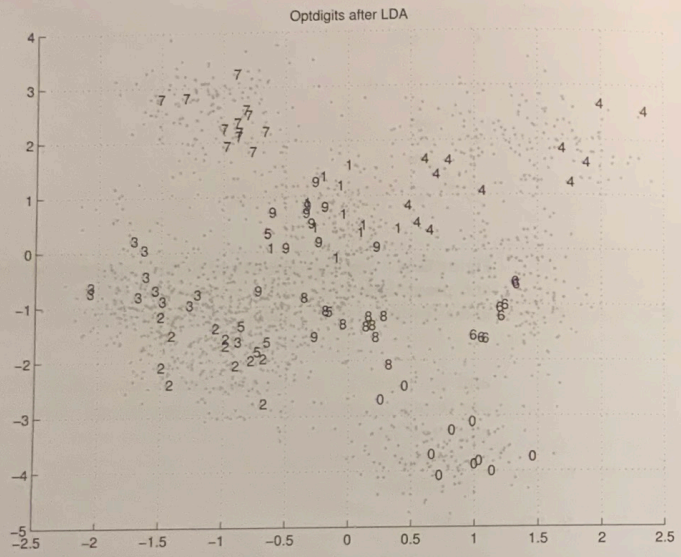

Figure 6.12 Optdigits data plotted in the space of the first two dimensions found by LDA. Comparing this with figure 6.5, we see that LDA, as expected, leads to a better separation of classes than PCA. Even in this two-dimensional space (there are nine altogether), we can discern separate clouds for different classes.

with $N_i = \sum_t r_i^t$. The between-class scatter matrix after projection is $\mathbf{W}^T\mathbf{S}_B\mathbf{W}$ and the within-class scatter matrix after projection is $\mathbf{W}^T\mathbf{S}_W\mathbf{W}$. These are both $k \times k$ matrices. We want the first scatter to be large, that is, after the projection, in the new k-dimensional space we want class means to be as far apart from each other as possible. We want the second scatter to be small, that is, after the projection, we want samples from the same class to be as close to their mean as possible. For a scatter (or covariance) matrix, a measure of spread is the determinant, remembering that the determinant is the product of eigenvalues and that an eigenvalue gives the variance along its eigenvector (component). Thus we

are interested in the matrix W that maximizes

$$J(\mathbf{W}) = \frac{|\mathbf{W}^T\mathbf{S}_B\mathbf{W}|}{|\mathbf{W}^T\mathbf{S}_W\mathbf{W}|} \tag{6.52}$$

The largest eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$ are the solution. $\mathbf{S}_B$ is the sum of K matrices of rank 1, namely, $(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T$, and only $K-1$ of them are independent. Therefore, $\mathbf{S}_B$ has a maximum rank of $K-1$ and we take $k = K-1$. Thus we define a new lower, $(K-1)$-dimensional space where the discriminant is then to be constructed (see figure 6.12). Though LDA uses class separability as its goodness criterion, any classification method can be used in this new space for estimating the discriminants.

We see that to be able to apply LDA, $\mathbf{S}_W$ should be invertible. If this is not the case, we can first use PCA to get rid of singularity and then apply LDA to its result; however, we should make sure that PCA does not reduce dimensionality so much that LDA does not have anything left to work on.

6.9 Canonical Correlation Analysis

In all the methods discussed previously, we assume we have a single source of data returning us a single set of observations. Sometimes, for the same object or event, we have two types of variables. For example, in speech recognition, in addition to the acoustic information, we may also have the visual information of the lip movements while the word is uttered; in retrieval, we may have image data and text annotations. Frequently, these two sets of variables are correlated, and we want to take this correlation into account while reducing dimensionality to a joint space. This is the idea in *canonical correlation analysis* (CCA) (Rencher 1995).

CANONICAL CORRELATION ANALYSIS

Let us say we have a dataset with two sets of variables $\mathcal{X} = \{\mathbf{x}^t, \mathbf{y}^t\}_{t=1}^N$ where $\mathbf{x}^t \in \Re^d$ and $\mathbf{y}^t \in \Re^e$. Note that both of these are inputs and this is an unsupervised problem; if there is a required output for classification or regression, that is handled afterward as in PCA (section 6.3).

The *canonical correlation* is measured as the amount of correlation between the $\mathbf{x}$ dimensions and the $\mathbf{y}$ dimensions. Let us define a notation: $\mathbf{S}_{xx} = \text{Cov}(\mathbf{x}) = E[(\mathbf{x} - \boldsymbol{\mu}_x)^2]$ is the covariance matrix of the $\mathbf{x}$ dimensions and is $d \times d$—this is the $\boldsymbol{\Sigma}$ matrix that we use frequently, in PCA for example. Now, we also have the $e \times e$ covariance matrix of the $\mathbf{y}$, namely, $\mathbf{S}_{yy} = \text{Cov}(\mathbf{y})$. We also have the two cross-covariance matrices, namely,

Appendix B Naive Bayes classifiers

Naive Bayes classifiers model the joint (multivariate) distribution of features given class as a product of the marginal (univariate) distributions for each feature given the class. In this project, where it is of interest to predict the clothing type Y from two continuous features X_1, X_2, we have

$$f(x_1, x_2|y) = f(x_1|y)\, f(x_2|y).$$

In this project, you should estimate the univariate feature distributions $f(x_1|y)$ and $f(x_2|y)$ using one of the non-parametric methods described below.

Estimating the univariate feature distributions

Given n observations, the histogram with binwidth h estimates a pdf $f(x)$ by the proportion of observations in the bin containing x scaled by the width of the bin (h):

$$\hat{f}(x) = \frac{\text{No. of observations in bin containing x}}{nh}$$

For the histogram, the bins are at *fixed positions*.

Alternatively, we can make a kernel density estimate using a "sliding window" so that $\hat{f}(x)$ is instead estimated by the proportion of observations in the window $[x - h; x + h]$ containing x, where h is the *bandwidth*.

$$\hat{f}(x) = \frac{\text{No. of observations in } [x - h; x + h]}{n \cdot 2h}.$$

So here the window is always *centered around the value* x that we are interested in finding the value $f(x)$ for, rather than non-overlapping bins at fixed values as in the histogram. The formula corresponds to placing a *kernel* at every single observation and then summing up their values measured at the point of interest x – hence the name. In this case the kernel is constant 1 in the interval $[x_0 - h; x_0 + h]$ for every single observation. So summing over all observations will exactly count the number of observations that are at most a distance h away from the point x of interest.

If you prefer a smooth density estimate, then you can use instead the Gaussian kernel. That would place a Gaussian bell curve in each observation rather than a constant function, so the further away from x they are, the less they contribute to the estimated pdf value $f(x)$.

References

Alpaydin, E. (2014). *Introduction to Machine Learning.* The MIT Press.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms.