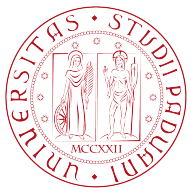


UNIVERSITY OF PADUA

MASTER'S THESIS

Partial Least Squares for Classification: a new point of view

Department of Mathematics
Master's Degree in Data Science

Author:
Martino DE NARDI
1206029

Supervisor:
Prof. Bruno SCARPA
Co-Supervisor:
Matteo STOCCHERO, PhD

Academic Year 2019/2020

Abstract

Nowadays data are everywhere and it becomes increasingly important to collect and analyze them in the correct way in order to obtain useful information, since a broad number of fields on a scientific and industrial level need data analysis to solve a wide range of problems.
With the advent of highly performing computers as well as measurement and processing systems, data are exploding progressively in both size and complexity, therefore requiring careful analysis to address the issues arising from this phenomenon: just think of the sensors that can make thousands of measurements in a few moments and that subsequently need a proper analysis to extract the information.
This scenario introduces the so-called *high dimensional* data, characterized by a number of predictor variables which is (possibly much) larger than that of observations: this type of data can be found in different areas such as economy, bioinformatics, astronomy, geology, chemistry, physics, and so on.
This phenomenon poses several problems when using the traditional approaches, so it is necessary to apply some methods that are suited and adapted to this context: one of this is Partial Least Squares regression (PLS), a technique initially designed for linear regression that exploits a dimensionality reduction approach to find a few orthogonal components which explain as much variance of the predictors as possible while being correlated to the response.

The focus of this thesis is to adapt PLS for classification from a new point of view with respect to those are now present in the literature, since in most cases PLS is used as a *discriminatory* tool rather than a *classifier*, meaning that it only separates the classes of the response variable and does not effectively perform the final classification (delegated to an additional classifier).
This is our starting point: indeed, the aim of this work is to design a new classification method purely based on PLS. To achieve this, there are two main ingredients: the first one involves the formulation of PLS as an iterative procedure that minimizes the distance between response and modelled response (that in the Euclidean space corresponds to the least squares problem) through the steepest descent method; the second one is the use of compositional data, through which

it is possible to consider the response as compositions (and therefore probabilities), giving a rigorous mathematical justification to the classification criterion used by the model, and make use of proper transformations that allow to perform calculations that link spaces with different structures.

Exploiting these factors, we developed a new approach that adapts PLS for classification providing a clear theoretical foundation, focusing on the binary response case. The case of $G > 2$ class is presented in its general framework but it requires further studies for a more detailed discussion.
Different procedures are proposed which share the underlying approach but differ in the space in which the calculations are made and in the transformation applied to the data.
These classification techniques have the same performance of Partial Least Squares - Discriminant Analysis (PLS-DA) , which is the most used state-of-the-art tool to perform classification using PLS; nevertheless, PLS-DA is not a purely PLS-based method since it also requires additional classifiers, as Linear Discriminant Analysis (LDA), to predict the classes of the observations.
Moreover, the proposed methods present a good predictive ability also in traditional scenarios, that is when the number of X-variables is much lower than that of observations and the collinearity between predictors is mild or moderate: in this setting, the results are comparable to those of logistic regression.
The classification procedures are tested against both simulated and real datasets, also giving the evidence of their theoretical properties.

Acknowledgements

First and foremost, I would like to thank my Supervisor, Prof. Bruno Scarpa, and my co-Supervisor, Matteo Stocchero PhD, for their generous availability and the never-ending professional support, with which they helped me patiently during these months.

I would like to express my gratitude towards my family, and to my parents in particular, for the continued financial and moral support that they provided me with during my university years and for never missing an opportunity to encourage me.

I would like to thank my classmates, especially Damiano, Domenico, Laura, Francesco, Árpád, and Paolo, for all the great time spent together, at the University as well as elsewhere, for being available in the times of need and for always reciprocating support.

Thanks to all my friends for being part of my everyday life and for having aided me directly or indirectly, and to my uncles, cousins and grandmothers, for their constantly-demonstrated affection and closeness to me.

Last but not least, I send a thought to my grandfathers, Augusto and Ivano, who, even if not physically present, have always been watching me from above.

I miei ringraziamenti vanno innanzitutto al mio Relatore, Prof. Bruno Scarpa, e Correlatore, Dott. Matteo Stocchero, per la generosa disponibilità dimostrata, la professionalità e il costante supporto che mi hanno fornito pazientemente in questi mesi.

Desidero ringraziare la mia famiglia e in particolar modo i miei genitori per il sostegno economico e morale che mi hanno sempre dato in questi anni, non facendomi mai mancare il loro incoraggiamento.

Ringrazio i miei compagni, in particolar modo Damiano, Domenico, Laura, Francesco, Árpád e Paolo, per il fantastico tempo passato assieme sia all'interno dell'Università che fuori e per essere sempre stati presenti nei momenti di reciproco aiuto.

Un grazie a tutti i miei amici per essere parte della mia vita quotidiana ed avermi direttamente o indirettamente sostenuto e a zii, cugini e nonne, per la vicinanza e l'affetto sempre dimostrati.

Infine, ma non di minore importanza, un pensiero ai miei due nonni Augusto e Ivano che, sebbene non presenti fisicamente, mi hanno sempre seguito dall'alto.

Contents

List of Figures

List of Abbreviations

PLS	**P**artial **L**east **S**quares
PCA	**P**rincipal **C**omponent **A**nalysis
PCR	**P**rincipal **C**omponent **R**egression
OLS	**O**rdinary **L**east **S**quares
RMSE	**R**oot **M**ean **S**quare **E**rror
IDA	**I**terative **D**eflation **A**lgorithm
PLS-DA	**PLS** - **D**iscriminant **A**nalysis
VIP	**V**ariable **I**nfluence on **P**rojection
SR	**S**electivity **R**atio
LDA	**L**inear **D**iscriminant **A**nalysis
QDA	**Q**uadratic **D**iscriminant **A**nalysis
CRA	**C**anonical **R**idge **A**nalysis
CCA	**C**anonical **C**orrelation **A**nalysis
KL-PLS	**K**ernel **L**ogistic - **PLS**
RR	**R**idge **R**egression
IRLS	**I**teratively **R**eweighted **L**east **S**quares
IRPLS	**I**teratively **R**eweighted **PLS**
R-PLS	**R**idge - **PLS**
RIRLS	**R**idge **IRLS**
RSS	**R**esidual **S**um of **S**quares
SVD	**S**ingular **V**alue **D**ecomposition
ALR	**A**dditive **L**og **R**atio Transformation
CLR	**C**entred **L**og **R**atio Transformation
ILR	**I**sometric **L**og **R**atio Transformation

Chapter 1

Guideline

The aim of this thesis is to fit PLS in the classification context, defining a new method to perform it with a precise mathematical justification and without using any additional classifier during the procedure.
Thus, first of all PLS is presented as it was designed, i.e. an algorithm to solve linear regression problems with continuous response variables, as well as some other post-processing steps that can be applied to it; then, some methods related to PLS are described to give a wider view of the model.
After this introductory part, the most common techniques that exploit PLS in the classification scenario are illustrated in Chapter 4: at the state of the art, PLS is mainly used as an initial discriminatory step, which means that it does not effectively assign a class to an observation, but more simply separates the samples so that subsequently a classifier can distinguish the classes and perform the classification.
Nevertheless, there are some methods that approach the use of PLS as a pure classifier: an overview about them is also given.
The new methods we designed for classification are presented in Chapter 6, where also the technical details are described. Finally, those techniques are tested using different datasets and the results are commented in Chapter 7.
The specifications regarding the mathematical notation employed in this work are given in the Appendix A, while in Appendix B proofs and additional clarifications about some details of the new methods explained in Chapter 6 are reported.

Chapter 2

Partial Least Squares regression

In this Chapter, Partial Least Squares regression (PLS) is presented in its original formulation, i.e. as a means of solving linear regression problems using a latent variable approach, and the motivation behind its ability to handle high-dimensional problems is highlighted. Then, different versions of the algorithmic procedure used to build the model are given, as well as a description of a post-processing technique to extract the predictive part of the model. An alternative and new formulation of PLS based on gradient descent will be introduced in Chapter 6.

2.1 PLS and reasons behind it

Partial Least Squares regression (PLS) is a method for relating two data matrices, the predictors X ($N \times P$) and the responses Y ($N \times M$) by a multivariate model; the regression is not performed using the measured predictors and responses, but using the so called latent variables that are obtained projecting the measured features along suitable directions. The latent variables span the latent space where the linear regression model is built. As a result, a bilinear decomposition is generated for both X and Y, while a matrix of regression coefficients is calculated to predict Y from X. The projection into the latent space reduces the dimensionality of the problem acting as a sort of regularization [43] [30].

A great advantage of PLS lies in its ability to handle data in high dimensional scenarios, especially when the number of predictors (which are affected by collinearity and are often noisy) is greater than that of observations.
The traditional approach to regression involves modelling Y by means of X using ordinary least squares (OLS), which works well if the X-variables are few and fairly uncorrelated (that means X has full rank): with the advent of sophisticated and technological measuring instruments (spectrometers, chromatographs, sensor batteries, etc), the number of explanatory variables increased

considerably in many fields and they are no longer by their nature uncorrelated to each other.
This fact poses a problem, since OLS is no more capable of providing models and estimates suited to the context (more about this in Section 5.1).
Consider a standard linear regression to be used when $n > p$: X is a $N \times P$ matrix, $y \in \mathcal{R}^N$ is the response vector and e is a vector of independently and identically distributed $N(0, \sigma^2)$ errors.
The common model is $y = X\beta + e$ and the aim is to find the optimum vector $\hat{\beta}$ to minimize the residual sum of squares:

$$\hat{\beta} = \underset{\beta \in \mathcal{R}^P}{\operatorname{argmin}} \|y - X\beta\|_2^2 \tag{2.1}$$

The well-known solution of this problem is

$$\hat{\beta} = (X^\top X)^{-1} X^\top y$$

This closed form solution exists if the columns of X are linearly independent and $X^\top X$ is invertible; it appears obvious that in the $p > n$ scenario, the variables are correlated, X cannot have linearly independent columns and $X^\top X$ is not invertible.

On the contrary, PLS regression is able to solve the "large p small n" problem since it exploits dimensional reduction using orthogonal latent variables as new predictors in the model and therefore allows to analyze data composed of many correlated predictors: for this reason it now occupies an important position in the chemometric literature.

2.2 Historical notes about PLS

The first papers concerning PLS were published around 1975 by Herman Wold, a Norwegian-born econometrician and statistician who applied this technique to multiple blocks of data from the economic and social sciences [38]: indeed, in those years he realized the power of the latent variable concept in multivariate modelling, starting to show interest in Principal Component Analysis (PCA), which he then generalized to path models in latent variables using the PLS approach [42].
Subsequently, also his son Svante Wold (chemometrician), was attracted by the

PLS philosophy: at the beginning of the 80's he started to work on PLS regression with Harald Martens and after several issues (and with the help of the professor of numerical analysis Axel Ruhe), they defined the well-known 2-block PLS regression approach based on the Nonlinear Iterative Partial Least Squares (NIPALS) algorithm developed by H. Wold for path modelling.
The PLS version of Wold and Martens is called PLS2 to distinguish it from other implementations such as SIMPLS [15] and UPLS [9]. PLS2 was presented at a conference in Oslo in 1982 [39]; after that, he published together with his father and Harald Martens the first papers about PLS and multivariate calibration in 1983 [41] and an application for the analytical chemistry community [17]. In Martens' opinion, "there is no other statistical method with comparable versatility" with respect to PLS regression [19], since it provides both cognitive access to the relevant information and statistical tools for the reliability of the results. S. Wold suggested to use Projection to Latent Structures by partial least squares instead of Partial Least Squares to give a more descriptive meaning to PLS.

2.3 PLS2

To truly understand why PLS can overcome the high-dimensional problem, let's see how regression is performed.
The PLS model aims at finding a few new latent variables that captures the variance of the X matrix[1] but at the same time are correlated with the Y block. In fact, for each latent variable PLS finds the direction that maximizes the covariance between X and Y, in a sense adjusting the PCA directions to better predict the Y.
Thus, the idea is to find some variables, whose values are expressed by the score matrix T ($N \times A$), that focus both on describing as much variance as possible of the X block while being correlated to Y.

In general, given a matrix X of predictors $[x_1, x_2, \ldots, x_P]$ and a matrix Y of responses, the aim of PLS is to compute the regression coefficients matrix B ($P \times M$) defining the following model:

$$Y = XB + F_A \tag{2.2}$$

where A is the number of latent variables and F_A is the matrix of residuals. Moreover, PLS does not require assumptions on the statistical distribution of the errors because regression is performed minimizing the norm of the matrix

[1] As the Principal Component Analysis does, more about this in 3.1

of the residuals in an algebraic fashion. If X is a full rank matrix and A is chosen to be the rank of X, than the regression coefficients are the same of the OLS estimates. Since PLS is usually applied in cases when $P > N$, a value of A smaller than the rank of X is typically adopted.

PLS is an iterative algorithm where at each step i a weight vector w_i is calculated; the weight vector is used to project the residual matrix to obtain the score t_i. Thus, the weight vectors give the information about how much an explanatory variable x_i contributes to form a latent variable. Note that the terms *latent variable* and *component* are used with the same meaning in the following.

The vector w_i is found maximizing the covariance of the residual matrix of the X-block and Y; the matrix T obtained juxtaposing the score vectors $t_i, (i = 1, \ldots, N)$ is used to decompose X and Y such that:

$$X = TP^\top + E_A \tag{2.3}$$

and

$$Y = TQ^\top + F_A \tag{2.4}$$

where E_A $(N \times P)$ and F_A $(N \times M)$ are respectively the residuals of the X and Y block after A iterations and P $(P \times A)$ and Q $(M \times A)$ are the loadings, that can be expressed using T and the original data X and Y. Indeed, $P = X^\top T(T^\top T)^{-1}$ and $Q = Y^\top T(T^\top T)^{-1}$.
The general scheme of the matrix decomposition generated by PLS is shown in Figure 2.1.

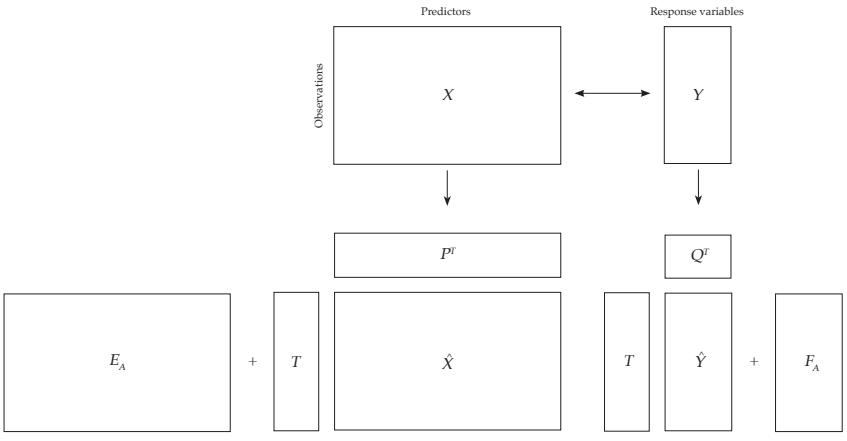


Figure 2.1: Matrix decomposition of PLS model

We now present the most common formulation of PLS introduced by Martens

and Wold [41], based on the Nonlinear Iterative Partial Least Squares (NIPALS) algorithm. PLS is called PLS2 in that case. Unless otherwise noted, PLS will be used to stand PLS2 in the following.
Let X be the predictor matrix, Y the response matrix, A the number of components of the model and ϵ the parameter for the convergence check.

Algorithm 1: NIPALS - PLS

1 $E_0 = X$, $F_0 = Y$;
2 **for** $i = 1, \ldots, A$ **do**
3 Set $\tilde{u}_i$ the first column of F_{i-1};
4 $w_i = \frac{E_{i-1}^\top \tilde{u}_i}{(\tilde{u}_i^\top E_{i-1} E_{i-1}^\top \tilde{u}_i)^{\frac{1}{2}}}$;
5 $t_i = E_{i-1} w_i$;
6 $q_i = \frac{F_{i-1}^\top t_i}{(t_i^\top F_{i-1} F_{i-1}^\top t_i)^{\frac{1}{2}}}$;
7 $u_i = F_{i-1} q_i$;
8 If $\|\tilde{u}_i - u_i\| \leq \epsilon$, then go to 9, else $\tilde{u}_i \leftarrow u_i$ and go to 4;
9 $p_i = \frac{E_{i-1}^\top t_i}{t_i^\top t_i}$;
10 $b_i = \frac{u_i^\top t_i}{t_i^\top t_i}$;
11 $E_i = E_{i-1} - t_i p_i^\top$;
12 $F_i = F_{i-1} - b_i t_i q_i^\top$
13 **end**

Note that if we introduce the orthogonal projection matrix $\hat{Q}_{t_i} = I_N - \frac{t_i t_i^\top}{t_i^\top t_i}$ (that projects a given matrix M onto the space orthogonal to the score vector t_i), the last two steps (11 and 12) can be written as:

$$E_i = E_{i-1} - t_i p_i^\top = E_{i-1} - \frac{t_i t_i^\top}{t_i^\top t_i} E_{i-1} = \hat{Q}_{t_i} E_{i-1}$$

$$F_i = F_{i-1} - b_i t_i q_i^\top = F_{i-1} - \frac{u_i^\top t_i}{t_i^\top t_i} t_i q_i^\top = \hat{Q}_{t_i} F_{i-1}$$

This calculations are important because also in the following algorithms $\hat{Q}_{t_i}$ is used to simplify the notation in the deflation steps, since it helps in defining the part of information orthogonal to the score vector and therefore to be explained in the following components. This particular form of the deflation step assures that the score vectors t_i are a set of orthogonal scores.

Another formulation of the algorithm can be given since Höskuldsson observed that steps 3-8 are equivalent to solve the following eigenvalue equation [14]:

$$E_{i-1}^\top F_{i-1} F_{i-1}^\top E_{i-1} w_i = \lambda_i w_i \tag{2.5}$$

Thus, let X be the matrix of predictors, Y be the matrix of responses and A the number of latent variables; the algorithm can be re-formulated as follows.

Algorithm 2: NIPALS - PLS, Eigenvalue formulation

$E_0 = X$, $F_0 = Y$;
for $i = 1, \dots, A$ **do**
  ${E_{i-1}}^\top F_{i-1} {F_{i-1}}^\top E_{i-1} w_i = \lambda_i w_i$;
  $t_i = E_{i-1} w_i$;
  $E_i = \hat{Q}_{t_i} E_{i-1}$;
  $F_i = \hat{Q}_{t_i} F_{i-1}$;
end

where the matrix $\hat{Q}_{t_i}$ is used for the deflation steps of the X and Y block; at each step, indeed, the procedure removes the information explained by the i^{th} latent variable to define the residuals and use them at the next iteration to explain the part of data that is left. It is essentially the same algorithm as before, written in a more compact way and exploiting Equation 2.5.
It is worth noting that the Y-deflation step is not necessary to calculate the weights since F_{i-1} can be substituted by Y in 2.5 without to modify the solution. Again, the eigenvalue formulation is more suitable for theoretical investigations than the original form based on NIPALS as suggested by Rayens and Andersen [28] .
From a geometrical point of view, the solution of equation 2.5 is the solution of the problem of finding the versors w_i and c_i such that

$$w_i, c_i = \underset{w,c}{\operatorname{argmax}} \operatorname{cov}(E_i w, F_i c) \tag{2.6}$$

If the score vectors $t_i = E_i w_i$ and $u_i = F_i c_i$ are introduced, at each step of PLS the covariance between the scores of the X and Y block is maximized. This point of view is useful to develop new versions of PLS, as done in the case of constrained PLS [32] [34].

The score matrix $T = [t_1, \dots, t_A]$ can be calculated from the original measured predictors by $T = XW^*$ where W^* is a transformation of the matrix $W = [w_1, \dots, w_A]$ to write the scores as a function of the original variables; the matrix W^* can be computed as $W^* = W(P^\top W)^{-1}$.

As pointed out in 2.4 and using what we have just noticed, the Y matrix can be written as

$$Y = TQ^\top + F_A = XW^*Q^\top + F_A = XB + F_A \tag{2.7}$$

where $B = W^*Q^\top = W(W^\top X^\top XW)^{-1}W^\top X^\top Y + F_A$ is therefore the matrix of regression coefficients of Y against X. It is worth noting that B depends only on W, X and Y.

It is important to specify how the number of latent variables is chosen when applying PLS regression. In general, when performing a prediction of a response variable using a set of predictors, the aim is to get the lowest possible error: to achieve this, the model must be not too complex (otherwise it tends to overfit the data, giving high variance) but also not too simple, because in that case it misses the relevant patterns between features and output and thus underfits the data, causing bias. For this reason, usually a tradeoff between bias and variance must be found: however, in PLS scenario there is no analytical form of the prediction error and it is not known how errors vary as a function of the number of latent variables, so the cross-validation procedure on training data must be used in order to find the number of latent variables with the maximum predictive power, which coincides with the minimum error in prediction.

The validation of the choice of the final model can be done through a permutation test, that is a procedure that builds sampling distribution by resampling the observed data, in particular shuffling the y values for n times without replacement: this is done assigning different response values to each observation from among the set of actually observed outcomes. This procedure is usually applied in experimental studies to test the null hypothesis that two treatment groups do not differ on the outcome: in fact, we want to assess whether the several Q^2 values[2] after the shuffling are similar with respect to the original one. The distribution of the null hypothesis is therefore the distributions of the n Q^2 calculated after the shuffling: if the values of the Q^2 were close to the original one, it would mean that the model would predict those outcome values anyway and that it fails to define specific patterns for the prediction (it would not have credibility because the result does not change even if the values in the response variable are exchanged).
The p-value of a permutation test consists in the proportion of samples for which the statistics (e.g. the Q^2) is higher than the original one, with respect to the total number of permutations. Since in most cases it is not possible for computational reasons to consider all the possible permutations (which are $n!$), a reasonably large number of permutations is applied (usually 1000 or more).

[2]The Q^2 represents the goodness of prediction of a model and it is calculated as the R^2 but using the validation data or cross-validation

Finally, the predictions of new observations can be calculated as:

$$Y_{new} = X_{new}B$$

or calculating the score matrix T ($T_{new} = X_{new}W^*$) and then multiplying by the transposed of the Y-loadings:

$$Y_{new} = T_{new}Q^\top = X_{new}W^*Q^\top$$

2.4 PLS1

Considering the simplest case of univariate y response, which means that the Y block is a $N \times 1$ vector, the algorithm aims at finding b such that $y = Xb + f_A$, where f_A are the residuals after A iterations.

The algorithm starts looking for a vector w_1 that is the result of the maximization problem:

$$\max_{\|w\|=1} \operatorname{cov}(Xw, y) \tag{2.8}$$

Then, the score vector t_1 is defined as projection of the X data along w_1 and the matrix X is deflated using the information gathered up by this component; the process that is repeated iteratively for A times.
In its NIPALS formulation, the algorithm is:

Algorithm 3: NIPALS - PLS1

1 **for** $i = 1, \ldots, A$ **do**
2 $\quad w_i = {X_{i-1}}^\top y_{i-1}$;
3 $\quad w_i = \dfrac{w_i}{\|w_i\|}$;
4 $\quad t_i = X_{i-1} w_i$;
5 $\quad p_i = \dfrac{{X_{i-1}}^\top t_i}{{t_i}^\top t_i}$;
6 $\quad q_i = \dfrac{{y_{i-1}}^\top t_i}{{t_i}^\top t_i}$;
7 $\quad X_i = X_{i-1} - t_i {p_i}^\top$;
8 $\quad y_i = y_{i-1} - t_i q_i$;
9 $\quad$ save w_i, t_i, p_i, q_i
10 **end**

At line 2, the direction vector that maximizes the covariance between the predictors and the response is found. Then, after its normalization, the original data (X) are projected along it, resulting in scores t (line 4).

The two final steps consist in the deflation of the X and y matrices, in such a way to remove the information explained by the current component.
Thus, the response values therefore can be modelled as

$$\begin{aligned} y &= \sum_{i=1}^{A} \hat{y}_i + f_A = q_1 t_1 + q_2 t_2 + \cdots + q_A t_A + f_A = \\ &= T q^\top + f_A \end{aligned} \tag{2.9}$$

where T is a $N \times A$ matrix ($[t_1, t_2, \ldots, t_A]$, with $t_i \in \mathcal{R}^N$) and $q \in \mathcal{R}^A$.
Again, the vector of coefficients b can be defined as $b = W^* q^\top$, with $W^* = W(P^\top W)^{-1}$.

2.5 What is PLS model?

Often, traditional statistical methods involve a parameter estimation based on distribution assumptions (e.g. require knowledge about the distribution of the errors). PLS exploits a different approach: indeed, it is an algorithmic procedure based on linear algebra that does not make probabilistic assumptions and therefore has no assumptions about the statistical model of the data.
Instead, it deterministically minimizes a given loss function in a vector space provided with distance; thus, the term "model" in the PLS context refers to the matrix decomposition[3].
Moreover, the use of latent variables in PLS provides a mathematical frame which can be used to explore and relate data in a new domain, as well as to assess the number and nature of influencing phenomena, acting as interface between the experimental (measurable) data and the conceptual world, meaning the underlying structures existing as latent information [13]. This is of primary importance because one of the most important goals of all branches of science is to provide a description of the phenomena in a domain in terms of a small number of concepts [13], that is what PLS tries to achieve: nowadays, in many fields high dimensional data are emerging and methods that use latent variables try to convert them into communicable and accessible information.

2.6 Iterative deflation algorithm (IDA)

PLS regression is a special case of a more general strategy for regression that uses the so called Iterative Deflation Algorithm [30].
Given:

[3] Recall that using PLS the X matrix can be decomposed as $X = TP + E_A$

- the matrices X and Y of the data
- the number A of latent variables (or equivalently the number of iterations)
- the weight matrix $W = [w_1, \ldots, w_A]$

The algorithm is the following:

Algorithm 4: Iterative Deflation Algorithm

$E_0 = X$;
$F_0 = y$;
for $i = 1, \ldots, A$ **do**
    $t_i = E_{i-1} w_i$;
    $E_i = \hat{Q}_{t_i} E_{i-1}$;
    $F_i = \hat{Q}_{t_i} F_{i-1}$;
end

where the matrices E_{i-1} and F_{i-1} of the residuals are deflated using the orthogonal projection matrix $\hat{Q}_{t_i}$.
The algorithm produces orthogonal vectors t_i and for a non-trivial choice of W leads to:

$$\begin{aligned} Y &= XW(W^\top X^\top XW)^{-1} W^\top X^\top Y + F_A = XB + F_A \\ X &= XW(W^\top X^\top XW)^{-1} W^\top X^\top X + E_A \end{aligned} \tag{2.10}$$

where $B = W(W^\top X^\top XW)^{-1} W^\top X^\top Y$.
When w_i is calculated using equation 2.5, PLS is obtained. If w_i is the loading PCA-vector, Principal Component Regression (PCR) is obtained.

Notice that in this case we already have the matrix W before the application of the algorithm[4]: this might sound strange but actually the IDA is very useful for the transformation technique presented in Section 2.7.
In fact, an interesting property of this algorithm is that if a nonsingular matrix $L(A \times A)$ is used to transform the weight W, i.e.:

$$\tilde{W} = WL$$

and if those weights are used instead of W, the same residuals and the same B are obtained. This property is used to apply the post-transformation.

[4] That can be found e.g. by running a PLS algorithm before the application of the IDA

2.7 Post-transformation of PLS (ptPLS)

The post-transformation of PLS is a procedure through which it is possible to divide the latent space in a predictive part (meaning that it is useful in explaining the response) and the remaining non-predictive one, useless in describing the Y-block [34]: the strength of this technique lies in the fact that a smaller number of components is needed to describe the data and potential sources of noise can be detected.
Post-transformation has been introduced because PLS sometimes generates unsatisfactory decompositions with a number of latent variables that exceeds the rank of Y. This behavior is observed when X data variation that is not correlated to Y is used to build the latent variables: that X data variation is called *structured noise*.

It is basically a three-step process:

1. A PLS model is built to obtain the weight matrix W
2. The weights are subjected to a suitable orthogonal transformation through a matrix G
3. IDA is applied with $\tilde{W} = WG$ as weight matrix

In step 2, the matrix $G = [G_o G_p]$ is defined as a juxtaposition of the columns g_{oi} and the columns g_{pi}, related to the non-predictive and the predictive part respectively.
The block G_o gives weight vectors $w_{oi} = W g_{oi}$, while G_p defines the weight vectors of the predictive part $w_{pi} = W g_{pi}$: let's now see how this matrix G is built.
The block G_o is composed of M vectors, i.e. $G_o = [g_{o1}, \ldots, g_{oM}]$ and they are calculated as follows.
Consider the singular value decomposition of $Y^\top XW$: $Y^\top XW = USV^\top$; g_{oi} are the M eigenvectors extracted from:

$$\hat{Q}_V g_{oi} = \lambda_{oi} g_{oi} \tag{2.11}$$

with $\lambda_{oi} > 0$. The block G_p is made of $A - M$ columns ($[g_{p1}, \ldots, g_{p(A-M)}]$), computed as the eigenvectors of:

$$\hat{Q}_{G_o} g_{pi} = \lambda_{pi} g_{pi} \tag{2.12}$$

with $\lambda_{pi} > 0$ and A is the total number of columns of W.
The minimum number of predictive latent variables is less or equal to $\min(\text{rank}(Y), A)$.

Using $\tilde{W}$ in the iterative deflation algorithm, the X-block and Y-block are decomposed such that:

$$\begin{aligned} X &= T_p P_p^\top + T_o P_o^\top + E_A \\ Y &= T_p Q_p^\top + F_A \end{aligned} \tag{2.13}$$

which means that Y can be modelled using only the predictive part.
In this case, $P_p = X^\top T_p (T_p^\top T_p)^{-1}$, $P_o = X^\top T_o (T_o^\top T_o)^{-1}$ and $Q_p = Y^\top T_p (T_p^\top T_p)^{-1}$.
Now, the regression model becomes:

$$Y = (X - T_o P_o^\top) B_{pt} + F_A$$

where the original matrix of coefficients B and the new B_{pt} are linked by this relationship:

$$B = [I_p - W_o (P_o^\top W_o)^{-1} P_o^\top] B_{pt}$$

Thus, it is possible to define B_{pt} as:

$$B_{pt} = [I_p - W_o (P_o^\top W_o)^{-1} P_o^\top]^{-1} B$$

Analyzing the predictive part of X (i.e. $T_p P_p^\top$) is significantly important because it can be used to examine the X-variation that influence Y, while the non-predictive one (i.e. $T_o P_o^\top$) can be used to analyze variation of the X-block that do not affect the response block but generates the structured noise.

2.8 Model interpretation

One of the main advantages of the PLS model lies in its interpretation: in fact, it is possible to extract information between predictors and response and also between predictors and latent scores by direct model interpretation, e.g. using plots [32].
In presence of modest correlation between predictors, the vector of regression coefficients can be used for model interpretation: in this setting, when a PLS model is made of few latent components, one possible way to investigate the relationship between predictors and responses is through the w^*q plot, which shows their relation using a single plot (also in the case of more than one response variable).
However, when the correlation between predictors is strong, this is not true and the attention should be focused on other parameters like Variable Influence on Projection (VIP) and Selectivity Ratio (SR) or procedures as stability selection

[32]. There is no best parameter to interpret a model, since each of them is designed considering specific properties of the model, so they should be chosen according to the case.
As mentioned above, if the multicollinearity is not too strong and the model has few latent variables, a first tool to discover the relationship between predictors and responses is the w^*q plot: $w_i^*, i = 1, \dots, A$ are the column of the W^* matrix[5] and q_i, $i = 1, \dots, A$ are the loadings of the Y-block. This graph is based on the relationship $B = W^*Q^\top$ and so the w_i^* and the loading q_i are reported in the same plot.

Another useful tool to investigate the relationship between (predictive) latent variables and predictors and between (predictive) latent variables and responses is the correlation loading plot, in which the Pearson's correlations between each latent variable and both the predictors and the responses are plotted together in the same graph. Indeed, this plot can be helpful for the interpretation of the model since the scores of the latent variables are orthogonal; when a strong collinearity is present, this tool gives a qualitative explanation and visualization of the relation between predictors and responses and it is recommendable over the previous one [32]. In fact, predictors that are positively or negatively correlated to the response of interest are the ones for which the related points in the graph are close to that of the response or to its image obtained by origin reflection.

To give an example, we consider a real dataset which is divided during its construction in training and test sets. Each observation of the dataset consists in a ^{1}H Nuclear Magnetic Resonance (NMR) spectroscopy of post-mortem aqueous humor (the clear fluid filling the space in the front of the eyeball between the lens and the cornea) collected from sheep with both closed and opened eyes. Data processing was applied to obtain a dataset whose features are 43 quantified metabolites.
The y values consist in the post-mortem intervals (PMI) after which the samples are collected, expressed in minutes (ranging from 118 to 1429).
The training set is composed of 38 observations and 43 quantified metabolites as predictors, while the test set for the prediction has 21 observations. More details about sample collection, experimental procedure and data pre-processing in [18].
The correlation loading plot of a model with 4 latent variables is shown in Figure 2.2: $pcor[Tp]$ denotes the Pearson's correlation between the predictive latent

[5] Recall that $W^* = W(P^\top W)^{-1}$ is a transformation of the weight matrix W that allows to write the latent scores as a function of the original variables, i.e. $T = XW^*$

variable and each predictor and between the predictive latent variable and the response. The same holds for $pcor[To1]$, but the first orthogonal latent variable is considered instead of the predictive one. The points associated to the 43 metabolites are plotted in green, while the one indicating the response (PMI) in blue.

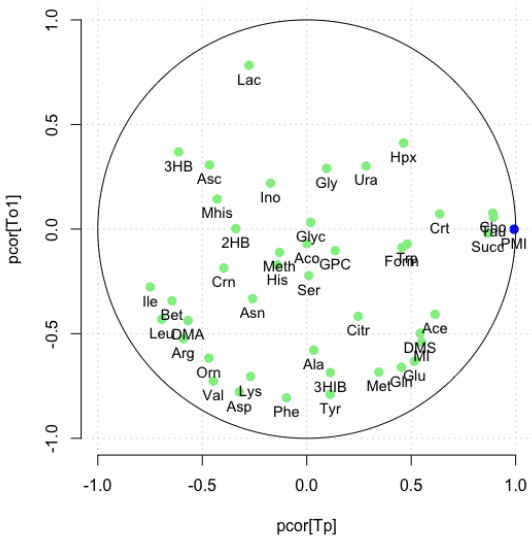


Figure 2.2: Correlation loading plot of a PLS model with 4 latent variables

Predictors whose points are close to the response (Taurine, Choline, Succinate) or to its projection by origin reflection (Isoleucine, Leucine, Betaine) are those that are most correlated to the response variable PMI.

Kvalheim in 2010 introduced a new parameter, the SR, which can be used only when there is a single predictive latent variable [16]: it is based on the possibility to decompose the latent space into two orthogonal subspaces (as described in 2.13) and on the assumption that the similarity between a predictor and a latent variable gets bigger with the increasing of the variance explained by that latent variable. In particular, SR is defined for each predictor i as the proportion between its variance explained by the predictive latent variable (computed using $X_p = t_p p_p^\top$, where t_p and p_p are the predictive latent score vector and the predictive X-loading vector respectively) and its variance not explained by that predictive latent variable (i.e. related to $X_o + E_A = T_o P_o^\top + E_A$, as in Equation 2.13), that is

$$SR_i = \frac{\text{var}(X_{pi})}{\text{var}(X_{oi} + E_i)} \tag{2.14}$$

The predictors with the highest SR are the most informative in explaining the predictive latent variable. A parametric test based on the F-distribution has been developed to select the most relevant X-variables.
Using the same dataset as before, it is possible to calculate the SR for each X-variable (Figure 2.3): it is worth noting that the predictors with the highest values are the same that resulted to be (positively or negatively) correlated with the response in the correlation loading plot (Figure 2.2).

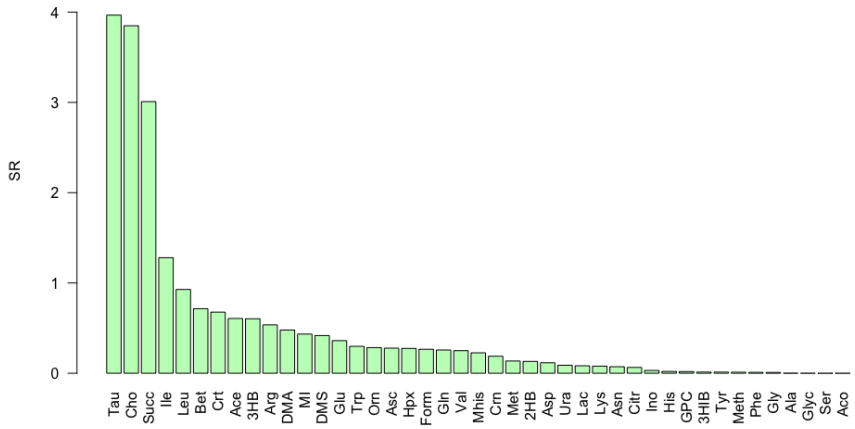


Figure 2.3: Selectivity Ratio of a PLS model with 4 latent variables

Nevertheless, one of the most common parameter to investigate the importance of a variable in a PLS model is the VIP: for each measured X-variable, the VIP is computed as

$$VIP_i = \sqrt{\frac{P}{SSY}\sum_{j=1}^{A} W_{ij}^2 SSY_j} \tag{2.15}$$

with A equal to the number of latent variables, P the number of X-variables, SSY_j corresponding to the sum of squares of Y explained by component j and SSY the total sum of squares of Y.
VIP provides a ranking for the contribution of each X-variable in building the latent space [40]. Unfortunately, parametric tests for assessing if a variable with a certain VIP is significant have not been developed. VIP is mainly used for variable selection and model refinement.
An example of the VIP is given in Figure 2.4, where it is possible to note that the most-influencing X-variables in defining the latent space correspond to the previously mentioned predictors.

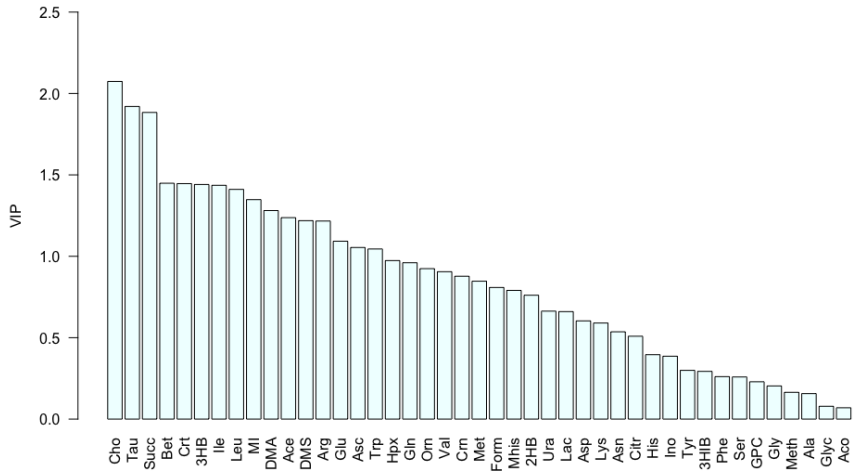


Figure 2.4: VIP of a PLS model with 4 latent variables

Finally, instead of evaluating the importance of predictors through parameters, it is possible to define a procedure, called *stability selection*, which aims at finding a subset of predictors that are helpful for the model. This technique is implemented drawing a consistent number of random subsamples of the data and then applying a PLS with VIP selection to each subsample: the most valuable predictors are those selected in more than half of the sub-models generated for each subsample [3]. The procedure has been recently improved including new measures of importance and a well-founded procedure for assessing which variable is relevant and which are irrelevant [31].

Chapter 3

PLS and related methods

PLS has many relationships with other regression techniques developed to solve linear problems where collinearity and noise affect the data: in fact, efforts have been made to include PLS in a more general system of regression methods.
To give a wider view of the model, in this Chapter we want to give a picture of those relationships, the common elements and those which instead distinguish PLS with respect to these.

Basically, when dealing with high-dimensional data, there are two main strategies that can be used to address the issue of multicollinearity: the first one is regularization, which imposes a penalization on the magnitude of regression coefficients, while the second consists in dimensionality reduction, a transformation of the original data from a high-dimensional space into a low-dimensional space in such a way that the new space representation retains the informative properties of the original data; this is what PLS does when looking for few latent components describing the data. We now present some famous methods related to PLS that implement dimensionality reduction or regularization, and then general frameworks that include PLS regression.

3.1 Principal Component Regression (PCR)

The first method related to PLS that comes to mind is PCA, a statistical procedure for dimensionality reduction that allows to summarize the information content in a large dataset by means of a smaller set of latent variables, which are linear combination of the original ones and try to describe as much as possible the variance of the data [1].
The orthogonal scores of PCA can be used as new variables for OLS, defining the PCR; unlike PLS, PCA in the regression context does not use the response variable to define the components and this is the main difference between the

two approaches: in fact, only the predictors are used to define the new latent variables.

The first component of PCA is the direction along which the variance of the data is maximized; the second one is the direction of maximum variance of the data subject to the condition of being orthogonal to the first. In general, each new component is defined as the direction that maximizes the variance of the data with the constraint of being orthogonal to the previous ones.
Indeed, the first weight vector is found solving the following maximization problem:

$$w_1 = \underset{\|w\|=1}{\operatorname{argmax}}\{\|Xw\|\}^2 = \underset{\|w\|=1}{\operatorname{argmax}}\{w^\top X^\top Xw\} \tag{3.1}$$

When w_1 is found, the first score vector is computed as $t_1 = Xw_1$.
Generalizing, the a^{th} component is calculated subtracting the contribution to explain the data of the previous $a-1$ principal components from X:

$$X_a = X - \sum_{s=1}^{a-1} Xw_s w_s^\top$$

and then finding the vector w_a that maximizes the variance of X_a:

$$w_a = \underset{\|w\|=1}{\operatorname{argmax}}\{\|X_a w\|\}^2 = \underset{w}{\operatorname{argmax}}\{\frac{w^\top X_a^\top X_a w}{w^\top w}\} \tag{3.2}$$

to calculate the a^{th} score vector as $t_a = Xw_a$.

In other words, the weight vectors are eigenvectors of the matrix $X^\top X$[1] whose associated eigenvalues correspond to the variance of the components.
Thus, the original matrix X can be written as:

$$T = XW \tag{3.3}$$

If X is a $N \times P$ matrix and A is the number of components of the model, T is defined as a $N \times A$ matrix and W has dimensions $P \times A$.
PCA is nowadays one of the most common techniques to perform dimensionality reduction, especially in high dimensional scenarios, such as those in which PLS is applied.

[1] The w_h weight vector is the h^{th} eigenvector of $X^\top X$

3.2 Ridge regression

Ridge regression (RR) is a regression technique that imposes a regularization on the magnitude of the coefficients, in such a way to reduce their variance and to mitigate the problem of multicollinearity. In RR, the OLS loss function is augmented by a penalty factor regarding the coefficients in order to minimize the sum of squared residuals as well as the size of parameter estimates, in such a way to shrink them towards zero to reduce the complexity of the model[2].
Thus, given a matrix $X \in \mathcal{R}^{N \times P}$ of predictors and a response vector $y \in \mathcal{R}^N$, RR solves the following optimization problem:

$$\min_{\beta_0, \beta} \{ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{P} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{P} \beta_j^2 \} \tag{3.4}$$

where β_0 is the intercept, β the vector of coefficients of the predictors and λ is the parameter that controls the amount of regularization of the model: setting λ to zero implies performing a OLS regression, while as λ increases, the penalty becomes more and more marked, shrinking the coefficients values towards zero.
Too high values of λ are associated to a strong penalization and therefore can lead to low variance but high bias at the same time. On the other side, too low values of λ can be not sufficient to properly reduce the variance of the estimates: thus, λ must be chosen in such a way to find a tradeoff balancing variance and bias (for example performing a cross-validation).
It is important to note that RR is characterized by a L2 penalization (λ is multiplied by the sum of squared coefficients) which tends to evenly shrink the coefficient to zero.

3.3 Lasso

The Lasso (Least Absolute Shrinkage and Selection Operator) is very similar to RR, with the only difference consisting in the penalty term. In fact, the Lasso adds to the OLS loss function the sum of the absolute values of the coefficients: this difference has several implications, since the L1 regularization creates sparsity in the model, which means that the less important features' coefficients are set to zero, therefore removing those variables from the model. From this point of view, in contrast to RR, Lasso also acts as a variable selection procedure.

[2] When multicollinearity affects the data, the OLS estimates of the regression coefficients tends to be very imprecise since they are characterized by high variance

Lasso solves the following optimization problem:

$$\min_{\beta_0,\beta} \{\frac{1}{2N}\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{P}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{P}|\beta_j|\} \tag{3.5}$$

If the outcome values y_i are centered, the intercept term β_0 can be omitted and the Lagrangian form of the problem becomes:

$$\min_{\beta} \frac{1}{2N}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

Actually, Lasso and RR can be seen as special cases of a more general setting, called Elastic Net, which combines the two methods and solves problem of the form:

$$\min_{\beta_0,\beta} \{\frac{1}{2N}\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{P}\beta_j x_{ij})^2 + \lambda_1 \sum_{j=1}^{P}|\beta_j| + \lambda_2 \sum_{j=1}^{P}\beta_j^2\}$$

with λ_1 and λ_2 corresponding to Lasso and RR penalty parameters respectively. The same problem can be also written as:

$$\min_{\beta_0,\beta} \{\frac{1}{2N}\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{P}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{P}[\frac{1}{2}(1-\alpha)\beta_j^2 + \alpha|\beta_j|]\} \tag{3.6}$$

where α is the parameter controlling how much Ridge regularization is used with respect to Lasso, and viceversa. In fact, $\alpha = 1$ consists in applying the L1 regularization (and therefore the Lasso procedure), while $\alpha = 0$ means using RR. If $0 < \alpha < 1$, a convex combination of the two methods is applied.

3.4 Canonical Ridge Analysis

An overview of the relation between different regression techniques can be introduced through the so-called *Canonical Ridge Analysis* (CRA) , which provides a unique formulation to collect different procedures like OLS, Canonical Correlation Analysis (CCA), RR and PLS as special cases of it [10].
Given a matrix X of predictors and the response matrix Y, the CRA in general solves the following optimization problem:

$$\max_{\|w\|=\|c\|=1} \frac{[\text{cov}(Xw, Yc)]^2}{[(1-\eta_X)\text{var}(Xw) + \eta_X][(1-\eta_Y)\text{var}(Yc) + \eta_Y]} \tag{3.7}$$

where $\eta_X \geq 0$, $\eta_Y \leq 1$ are the regularization terms and w, c are respectively the weights of the X and Y blocks to get the scores of the latent variables (among

which the covariance is maximized).
Some special cases can be highlighted to show how CRA includes well-known procedures:

- $\eta_X = 0$ and $\eta_Y = 0$ implies CCA, which solves the following problem:

$$\max_{\|w\|=\|c\|=1} \frac{[\text{cov}(Xw, Yc)]^2}{[\text{var}(Xw)][\text{var}(Yc)]} = \text{corr}^2(Xw, Yc)$$

- $\eta_X = 1$ and $\eta_Y = 1$ implies PLS since the problem becomes of the form:

$$\max_{\|w\|=\|c\|=1} [\text{cov}(Xw, Yc)]^2$$

This clearly applies also in the case of univariate response (y is a vector), getting:

$$\max_{\|w\|=1} [\text{cov}(Xw, y)]^2 \tag{3.8}$$

Moreover, in this setting and if $0 \leq \eta_X \leq 1$, RR is obtained. In fact, the optimization problem becomes

$$\max_{\|w\|=1} \frac{[\text{cov}(Xw, y)]^2}{(1-\eta_X)\text{var}(Xw) + \eta_X} \tag{3.9}$$

which can be seen as:

$$\begin{aligned}
&\max_{\|w\|=1} \frac{[\text{cov}(Xw, y)]^2}{\text{var}(Xw) - \eta_X \text{var}(Xw) + \eta_X} = \\
&\max_{\|w\|=1} \frac{[\text{cov}(Xw, y)]^2}{\text{var}(Xw) + \eta} = \\
&\max_{\|w\|=1} \frac{[\text{corr}(Xw, y)]^2 \text{var}(Xw)}{\text{var}(Xw) + \eta}
\end{aligned}$$

where $\eta_X \in [0, 1]$ and $\eta \leq 0$ representing the regularization terms (*ridge* parameters) and $\eta = -\eta_X \text{var}(Xw) + \eta_X$.
Finally, imposing $\eta = 0$ consists in considering the OLS case.
In fact, the optimization problem is the following:

$$\begin{aligned}
&\max_{\|w\|=1} \frac{[\text{corr}(Xw, y)]^2 \text{var}(Xw)}{\text{var}(Xw)} = \\
&\max_{\|w\|=1} [\text{corr}(Xw, y)]^2
\end{aligned}$$

which is equal to:

$$\max_{\|w\|=1} \frac{[\text{cov}(Xw, y)]^2}{\text{var}(Xw)} \tag{3.10}$$

that is exactly the OLS approach.
This is evident as imposing $\eta = 0$ means not adding a constant to the diagonal of the $X^\top X$ matrix.
Thus it is possible to summarize some methods and the quantities they optimize:

PLS: $\text{var}(Xw) \cdot \text{corr}^2(Xw, Yc) \cdot \text{var}(Yc)$
PCA: $\text{var}(Xw)$
CCA: $\text{corr}^2(Xw, Yc)$

As it can be seen from Equations (3.8), (3.9) and (3.10), the optimization approaches differ for the denominators, since in RR the ridge parameter η_X defines a convex combination between the PLS and OLS ones ($\eta_X = 1$ implies considering the PLS case while $\eta_X = 0$ leads to OLS setting).

3.5 Continuum approach

Actually, different continuum approaches have been designed in such a way to find a common ground among PLS and other regression techniques. One of these [26], is designed as a method to encompass OLS, PLS and RR as special cases, developed for the case of a single response variable.
In general, since the aim is to determine a latent variable $t = Xw$ having a good prediction ability, the vector of weights w must be chosen in such a way to maximize the correlation between t and y, i.e.:

$$\text{corr}(t, y) = \frac{t^\top y}{\sqrt{t^\top t}\sqrt{y^\top y}} = \frac{w^\top X^\top y}{\sqrt{w^\top X^\top X w}\sqrt{y^\top y}} \tag{3.11}$$

The vector of weights w which is the solution of the latter is proportional to $(X^\top X)^{-1} X^\top y$. This is basically the OLS model:

$$y = X(X^\top X)^{-1} X^\top y + e$$

where e is the vector of errors.
This procedure, as known, requires the inversion of matrix $X^\top X$, which can be a problem in case of presence of multicollinearity, phenomenon that leads to a bad quality of prediction and poor interpretability of the model.

In order to deal with this issue, PLS aims to find the vector w that maximize the covariance between t and y: in this case, w is proportional to $X^\top y$ and not to $(X^\top X)^{-1}X^\top y$ as in the OLS approach.
Thus, from this point of view PLS consists in a shrinkage of the matrix $(X^\top X)^{-1}$ towards I, the identity matrix. From this point of view, a *continuum* range of possibility between the two extreme points can be defined.
Let α be a constant such that $\alpha \in [0,1]$: instead of using only $(X^\top X)^{-1}$ or I, a convex combination can be defined: in this way, w is proportional to $[(1-\alpha)(X^\top X)+\alpha I]^{-1}X^\top y$.

The relationship between X and y can be modelled using a set of latent variables: in particular, the first one is $t_1 = Xw_1$, where w_1, as previously said, is proportional to $[(1-\alpha)(X^\top X)+\alpha I]^{-1}X^\top y$.
Then, the matrix of predictors X and the response variable y are regressed upon t_1, in such a way to determine the residual matrix of the predictors (E) and the residual vector f related to the y.
In a second step, the latent variable t_2 is created as $t_2 = Ew_2$, where in this case w_2 is proportional to $[(1-\alpha)(E^\top E)+\alpha I]^{-1}E^\top f$. This procedure can be repeated as many times as the number of latent variables that must be computed.

Note that in this setting, due to the construction of the approach, $\alpha = 0$ gives the OLS regression while $\alpha = 1$ corresponds to PLS regression.
Thus, the first step leads to the model:

$$\hat{y} = \gamma X[(1-\alpha)(X^\top X)+\alpha I]^{-1}X^\top y \tag{3.12}$$

where γ is the regression coefficient of y upon the first latent variable t_1.
With a slight modification, it can be written as:

$$\hat{y} = \frac{\gamma}{(1-\alpha)} X[(X^\top X)+k(\alpha)I]^{-1}X^\top y$$

where $k(\alpha) = \dfrac{\alpha}{1-\alpha}$ and $k(\alpha) \in [0,+\infty)$.
As it can be noticed, the first-factor model of this approach is equivalent to the RR with a modification of a constant.
Indeed, one can test this type of regression with different values of α and look for the one which provide the minimization of the error (or maximization of the Q^2) with a numerical method.
For this purpose, a dataset has been defined in such a way to appreciate the influence of the parameter α on the model. Basically the construction of the dataset starts from a PCA model of a matrix M of random numbers (a 100×100

matrix of numbers drawn from a Gaussian function with $\mu = 0$ and $\sigma = 1$) from which the loadings P are extracted as right singular vectors.
The data in M are projected along those directions to find the orthogonal scores T ($T = MP$) and one of those scores is chosen as response, with additional noise; the loadings are then used again to compute the X matrix as $X = TP^\top$.
Figure 3.1 represents the Root Mean Square Error (RMSE) obtained in a 7-fold cross validation using the first component of the model and different values of α.

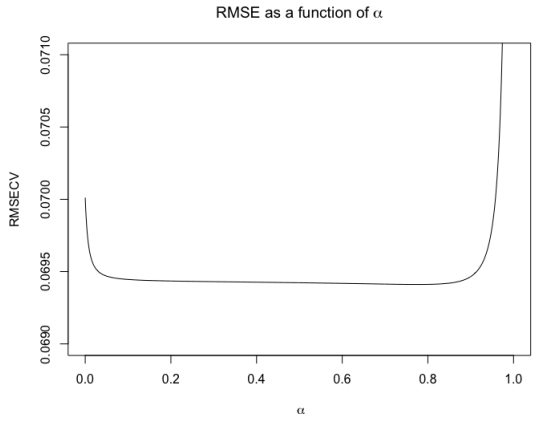


Figure 3.1: RMSE in cross-validation (RMSECV) as a function of α

In Figure 3.2, instead, the RMSE in cross-validation for different numbers of components for PLS is shown.

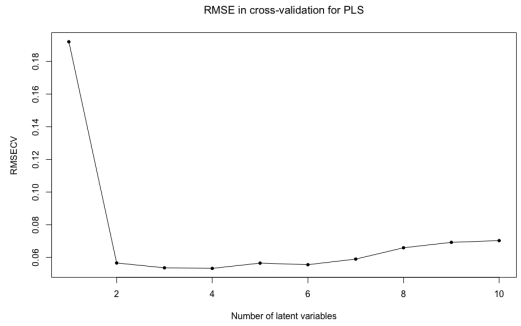
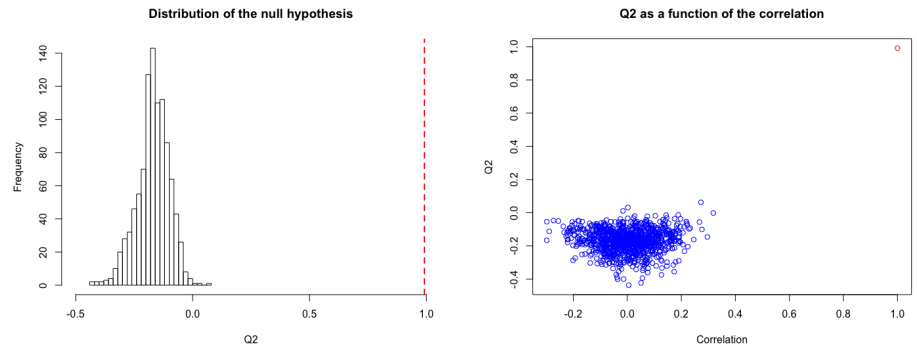


Figure 3.2: RMSE in cross-validation (RMSECV) for PLS

Thus, the minimization of the error occurs for a fairly large value of α, precisely for $\alpha = 0.81$, that indicates that the model is closer to PLS than OLS.
To validate this result, we apply a permutation test: in this case ($n = 1000$ permutations) it is evident that the null hypothesis is rejected since the original Q^2 is at the extreme of the distribution, giving a p-value of 0.001 (Figure 3.3).



(a) Null hypothesis distribution (b) Q^2 and correlation for each permutation

Figure 3.3: Permutation test ($n = 1000$)

To sum up, PLS has many relationships and common elements with other regression techniques (primarily PCA as it is defined), which however differ from each other based on the quantity that is maximized in determining the weights and latent variables. Furthermore, PLS can be seen as the borderline case of a continuous approach that can go up to the classic OLS, including inside the RR. All these techniques are usually applied to perform linear regression, which is also the original aim of PLS; therefore, predictions are made on quantitative response variables. In Chapter 4 several methods that use PLS in the classification scenario are described.

Chapter 4

PLS for discrimination

Classification is a cornerstone in statistics; in many application areas, in fact, the variable to be predicted is qualitative and therefore PLS requires an appropriate adaptation of the algorithm to the categorical case, which is *not the original purpose* for which this technique was designed (i.e., to perform linear regression).
There are many ways in which PLS can be applied within a classification framework; in most cases PLS is used as a *discriminatory tool* to separate the observation, but it needs a subsequent additional model to perform the actual classification (so it *discriminates* rather than *classify*): this is a crucial point since it is the reason why we want to develop a new technique that exploit PLS directly as a classifier.
A classification procedure based on PLS as discriminator is therefore commonly built following two main steps: first of all, the response variable is coded through an indicator Y-matrix containing the information about the class of each instance and a PLS model is built on it using the predictors. Then, the latent scores are used as input for a classifier (many types of classifiers or classification criteria can be used in this step).
In the following sections, the most common approaches are described.

4.1 PLS-Discriminant Analysis (PLS-DA)

PLS has been developed to perform regression: then, it cannot be applied to responses that are categorical variables. The idea underlying PLS-DA is to use a numerical representation of the categorical response that can be modelled by PLS. In classical regression, categorical variables are represented using a special coding that uses indicator variables: specifically, two coding systems are the most used, both using indicator variables composed of 0 and 1.
The first one, in case of a G-class problem, uses a indicator matrix with G

columns, where each column is associated to a particular class. An observation belonging to class i is codified with 1 in column i and zero in the other columns: the obtained representation is overdetermined.
The other one uses a class as a control and an indicator matrix with $G-1$ columns. If the observation belongs to the control class, it is codified using 0 in all the columns, while if the observation belongs to class i that is not the control class, it shows 1 in column i and zero in the other columns.
Both systems can be used (and are used) to codify the categorical response in PLS-DA. After the coding, the indicator Y-matrix is centred or autoscaled[1] and submitted to PLS. This approach is a heuristic approach to discrimination that is usually justified recalling that the directions discovered by Linear Discriminant Analysis (LDA) can be calculated applying CCA to relate the X-matrix of the predictors and the indicator Y-matrix specifying the class membership [6]. Here, since the predictors are correlated and noisy, CCA is substituted with PLS.
When the G classes are equally populated and X is centred, the objective function of PLS at each iteration is

$$\underset{\substack{w^\top w=1\\ c^\top c=1}}{\operatorname{argmax}}\, w^\top E_{i-1}^\top F_{i-1} c \implies w_i : \operatorname{argmax}\, t_i^\top (\hat{Y}\hat{Y}^\top) t_i \propto \sum_{j=1}^{G} \bar{t_{ij}}^2 \tag{4.1}$$

where $\bar{t_{ij}}$ is the mean value of the score t_i calculated considering the observations of class j. In other words, the among-groups sum of squares for each latent variable is maximised. In the general case of classes with a different number of observations, the maximum of the objective function depends on the number of observations of each class and the among-groups sum of squares is not maximised. In the case of a 2-class problem the differences between the mean scores of the two classes is always maximised: these properties can be proved with simple algebra.
Barker and Rayens [5] proposed a modification of the scaling factor used to scale the indicator Y-matrix in order to obtain a maximum of the objective function equal to the among-groups sum of squares both for equally and not equally populated classes.

Then, the representation of the observations in the latent space obtained by PLS-DA should be more suitable for distinguishing the groups of observation than that obtained by PCA [8]. However, the within-group variation is not taken into account.

[1] Each variable of the dataset is standardized, subtracting the mean of that variable and dividing by its standard deviation

The matrix of the responses is modelled considering a linear regression model. As a consequence, the calculated responses are not 0 or 1 but they are in the whole real axis and the prediction of PLS-DA cannot be directly interpreted as class membership. For this reason, a second step must be performed to transform the results of PLS-DA into class membership: this is a drawback of PLS-DA. Several approaches can be applied to assess class membership: the traditional approach is that to classify an observation on the basis of the class response with the maximum value or to apply a some sort of classification rule optimized by cross-validation. Other approaches use the scores of the PLS-DA model as predictors to train classical classifiers, such as Naive Bayes classifiers and LDA.
Model interpretation is performed taking into account only the PLS part of the procedure and the tools described in Section 2.8 are applied. The lack of a method to assess the importance of the predictors in the whole procedure used for classification is another drawback of PLS-DA. It is interesting to note that most of the studies published in literature that use PLS-DA do not specify which scaling is applied to the indicator Y-matrix or which rule or technique is used to transform the results of the PLS regression in class membership, while PLS-DA is a two-step procedure. In the next section, PLS-DA is presented using a real dataset and simulated datasets.

4.2 Some examples

4.2.1 Classification through LDA

To give a practical example, a dataset with three classes is chosen; in particular, the observations are ^{1}H-NMR olive oil spectra used for cultivar classification: the dataset is made of 45 observations, which are almost equally distributed among the three classes (16 observations for class 1, 15 for class 2 and 14 for class 3), and 221 predictors. The dataset is divided in a training set and a test set, to perform the prediction on previously unseen samples: the test set includes 15 observations equally distributed among the classes.
The three possible outcomes of the response variables are "Coratina" (originally from the city of Corato, is an olive cultivar typical of Puglia and cultivated throughout the North Bari countryside), "Ogliarola" (one of the most widespread squeezing olive cultivars in the south of Italy, particularly present in Puglia and Basilicata) and "Peranzana" (an olive cultivar whose production area is the north-west area of the province of Foggia). More details about the dataset are provided in [25].

A first glance at the data can be given by applying a PCA on the autoscaled matrix of predictors X and assessing which is the proportion of the total variance that the first latent variables are able to explain.

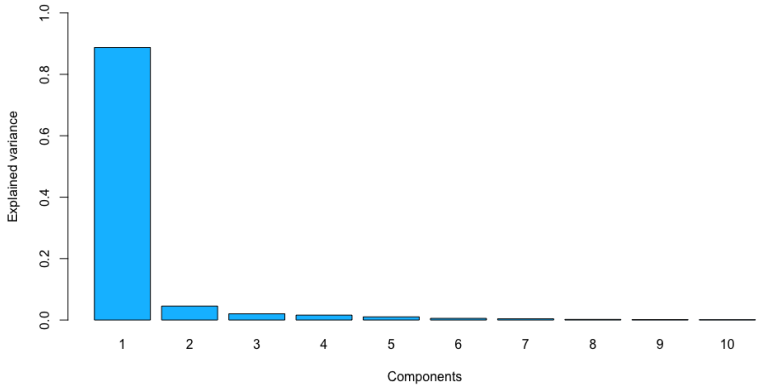


Figure 4.1: Proportion of explained variance of each component using PCA

These components captures most of the variance of the data (Figure 4.1): in fact, the first latent variable is able to explain more than 88% of the total variance, followed by a 4.50% of the second one; the first four components explain the 97% of the total variance of the data.
In Figure 4.2 data separation using the first two components of PCA is illustrated.

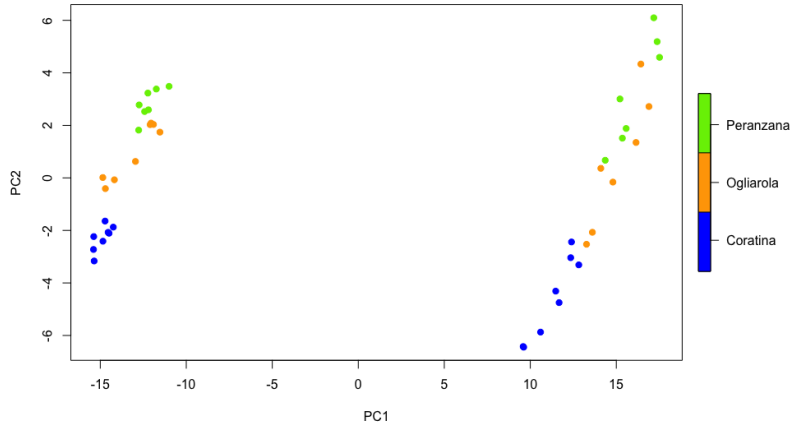


Figure 4.2: Scores of the first two latent variables

The first component is clearly not useful in separating the groups of observations. PLS-DA has been performed considering the autoscaled matrix of X-variables and autoscaled indicator Y-matrix with 3 columns and LDA on the score vectors in the second step.
In Figure 4.3 is presented an example of the discriminatory power of PLS and

in particular of the scores upon which LDA is performed: more precisely, those values are the scores of the two predictive components (post-transformation has been applied as explained in Section 2.7) of the model[2]. Unlike the first components of PCA (Figure 4.2), in this case the two predictive components of the model are useful in separating the classes.

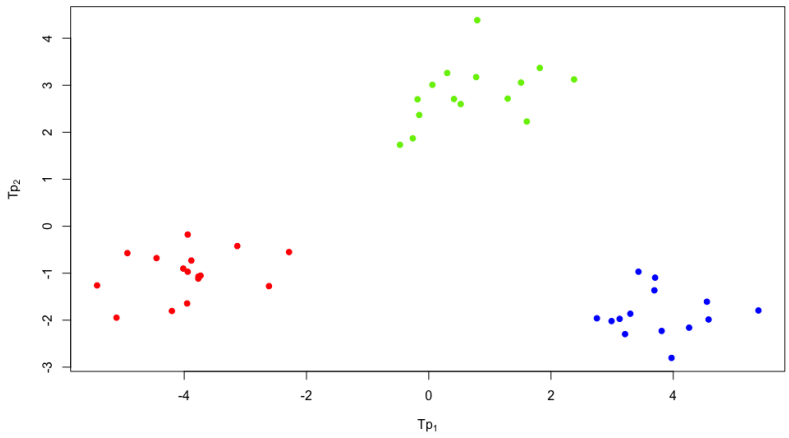


Figure 4.3: Scores of the predictive components (3-class classification problem)

Indeed, three PCA components are necessary to separate the groups: Figure 4.4 shows the clusters of observations of the three cultivars created by the application of PCA with the first three components.

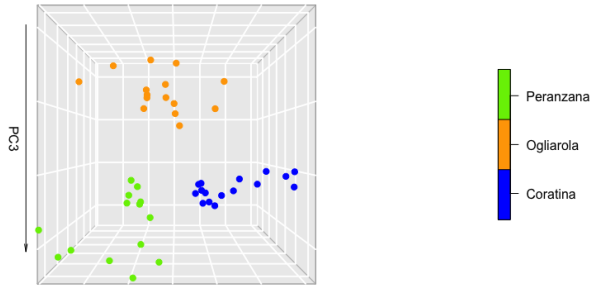


Figure 4.4: Scores of the first three latent variables

In this case it can be seen that PLS is able to markedly separate the samples, which is the best possible case: a traditional classifier will not have problem in

[2]An example with three classes is shown since two predictive component are sufficient to discriminate between three classes and so they are easy to represent in a 2D plot

correctly classifying the samples.
Using LDA on the scores of a PLS model, the following results are obtained for different numbers of components (Figure 4.5).

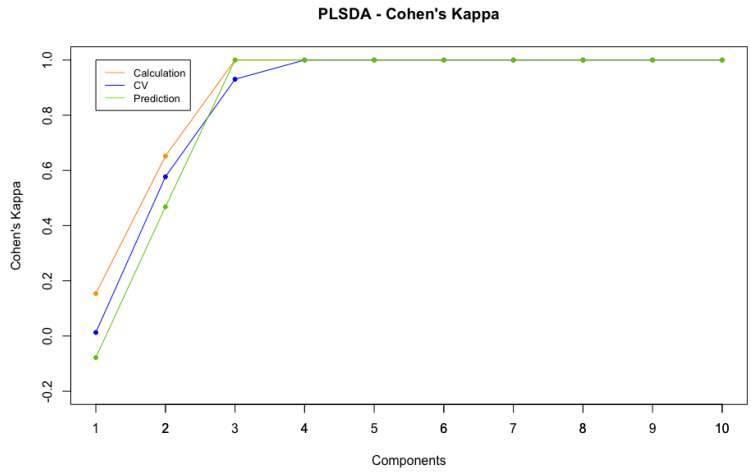


Figure 4.5: Cohen's Kappa in calculation, cross-validation and prediction

We use the Cohen's Kappa as statistic since it represents the degree of accuracy and reliability in a statistical classification, it takes into account the possibility of the agreement occurring by chance and it can handle multi-class problems [21].
The Cohen's Kappa is defined as:

$$\kappa = \frac{P(a) - P(e)}{1 - P(e)}$$

where $P(a)$ is the proportion of values in the confusion matrix that are in agreement, calculated as the sum of the diagonal terms (the ones for which the samples are correctly classified) divided by the total number of observations. Instead, $P(e)$ represents the expected proportion of chance agreement and it is computed as the sum of the multiplication of the marginal proportions for each class.
The Cohen's Kappa can take values from -1 (worst situation) to 1 (perfect classification): if $P(a) = 1$, $\kappa = 1$ and the non-diagonal terms of the confusion matrix are all 0; therefore, the model correctly classifies all the observations.
If $P(a) = P(e)$, than the numerator is equal to 0 and so is κ, meaning that the result is that of a random classifier; finally, if $P(a) < P(e)$, the κ value is lower than 0 and the model performs worse than a random classifier.
Note that PLS-DA can certainly also be used in 2-class problems (the simplest

case).

As it can be seen, the Cohen's Kappa does not present good values until three components are used in the model; from the fourth one, every instance is assigned to its true class in calculation, prediction and CV, which is performed with 6 folds on the training set.

4.2.2 Classification through other approaches

Another procedure consists in exploiting a Bayesian approach to classify the observations (we start from 2-class problem scenario): indeed, one can train a PLS model on the training set with a binary response, collecting all the values of the calculated y-value for the two classes. A normal distribution is subsequently fitted on the two set of values and finally a threshold can be set in the point where the two Gaussian curves cross, since in that point the probability of belonging to the two classes is assumed to be equal: using it, a new instance can be classified depending on whether its predicted value in the test phase exceeds the threshold or not. PLS is exploited in this case to discriminate between classes and the samples are than classified using a specific criterion; the threshold is used to assign a test sample to a specific class [24].
To exemplify this approach, a dataset is built starting from the definition of a matrix $D \sim \mathcal{N}(0,1)$ with 100 rows and 100 columns of random numbers drawn from a Gaussian distribution, from which the loadings (P) are extracted as right singular vectors (the first three vectors are kept, so P is a 100×3 matrix). The y variable is defined as a sequence of 50 values equal to 1 and other 50 equal to 0, defining the classes of the observations. The matrix of the predictors is made of a column z_1 that contains the information regarding the classes (since it is composed of a set of 50 random numbers from a Gaussian with $\mu = 1, \sigma = 0.1$, and other 50 random numbers from a Gaussian with $\mu = 0, \sigma = 0.1$), and other two columns (z_2, z_3) that do not contain any information (they can be extracted from a Gaussian with a given mean and standard deviation, or other distributions)[3]. The matrix $Z = [z_1, z_2, z_3]$ is mean centred obtaining Z' and, since we want to have a matrix T with orthogonal components, it is multiplied by its loadings P' found through the application of the PCA, resulting in $T \in \mathcal{R}^{100\times3}$. Finally, T is multiplied by $P^\top$ to get the final X matrix (100×100).
In Figure 4.6 the matrices generation scheme is reported.

[3] There is no particular reason why the columns that do not carry information are two, another number could be used

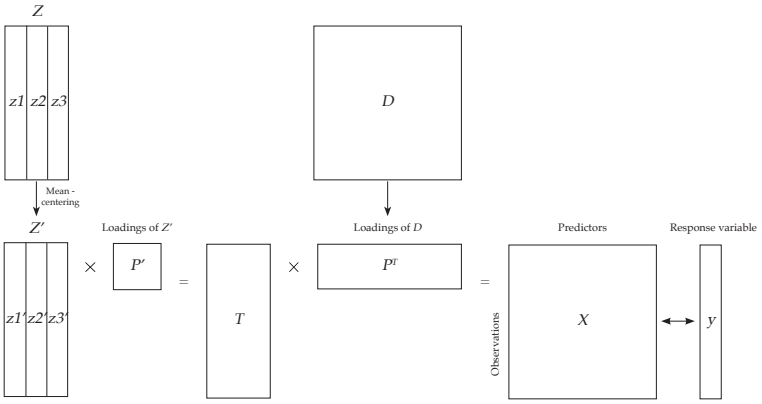


Figure 4.6: Matrices generation scheme

Figure 4.7 illustrates an example of the Normal distributions fitted on the calculated values during the training step where the observation are easily separable.

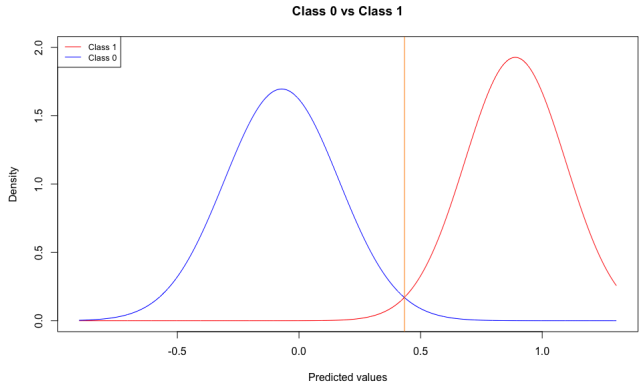


Figure 4.7: Gaussian distributions and threshold with $\sigma = 0.1$

In this case the two classes are easily separable because the variance of the two groups in $z1$ is quite low (0.1). The more the samples are easily separable, the more the PLS model is able to discriminate between them and the probability of making mistakes during the classification phase decreases.
In Figure 4.8 the standard deviation of the two groups of samples in $z1$ is set to 0.25, so the classes are less easily distinguishable than in the previous case. With respect to the previous example, the two curves are less tightly clustered around the mean, thus the probability that in the prediction phase an observation of a class may be beyond the threshold is higher.

Other examples are given in Figure 4.9 where $\sigma = 0.5$ and $\sigma = 1$ are considered:

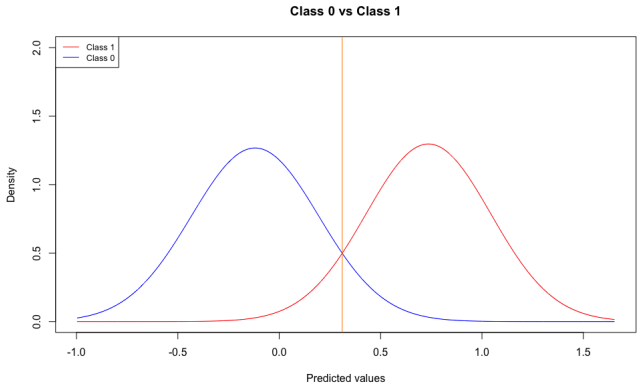
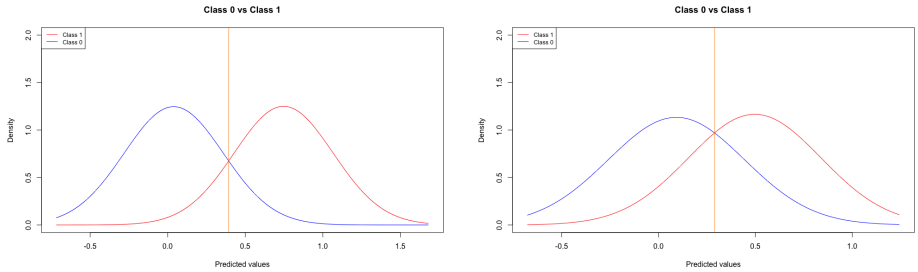


Figure 4.8: Gaussian distributions and threshold with $\sigma = 0.25$

the higher the value of the standard deviation, the more the spread of the observations around the mean increases, implying more problems in separating the samples.



(a) $\sigma = 0.5$ (b) $\sigma = 1$

Figure 4.9: Gaussian distributions and threshold with $\sigma = 0.5$ (left) and $\sigma = 1$ (right)

This method can be generalized to $G > 2$ classes, starting from training a PLS model on the indicator Y-matrix; then, for each class, the samples of that class are separated from those of all the others classes using the threshold as for the 2-class procedure previously mentioned. A threshold is set using the calculated values in the training phase in such a way to separate that class from the others; in this setting, if the predicted value of a new test instance for a given class is greater than the threshold, the observation is assigned to that class [4].
However, there is a non-negligible drawback: using this criterion it can happen that two or more estimated class values exceed the related thresholds; in this case, the sample is defined as *not assigned*.
The same applies in the case in which no one of the estimated class values exceeds the related thresholds: in this case, the sample would not be recognised

as member of any class, so it is labeled as *not assigned* too.
In the ideal situation, only one of the predicted class values exceeds the threshold and the observation is assigned to that class.
In Figure 4.10 the number of *not assigned* samples and the Cohen's Kappa in calculation using the olive oil dataset with three response classes can be observed.

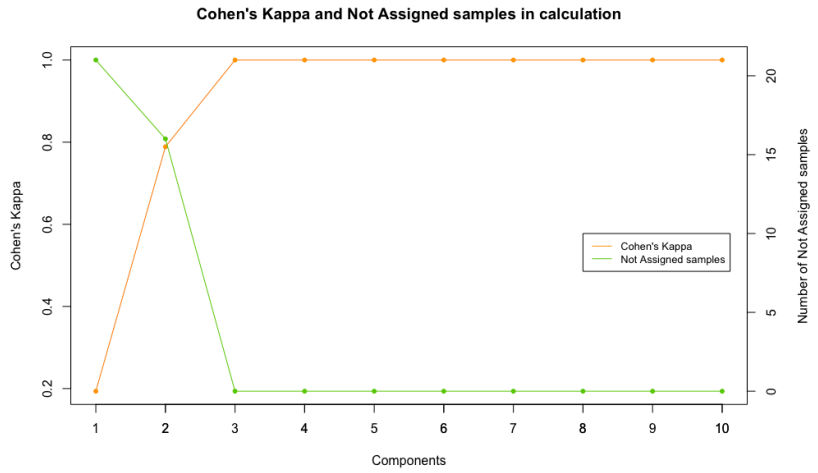


Figure 4.10: Cohen's Kappa and number *not assigned* samples in calculation

As it can be seen, the value of the Cohen's Kappa rises quickly and it is an increasing function of the number of latent variables, while at the same time the number of *not assigned* samples decreases, reaching zero after three components. The result obtained in prediction using the test set (which presents an equal number of observations between classes) is reported in Figure 4.11.

The number of *not assigned* samples has a minimum (as it should) when four latent variables are used, which also corresponds to the optimal number of components, that represent a compromise between a too simple model and a model that incurs in overfitting. When more than five component are used, the number of *not assigned* samples increase because the model overfits and values of bad predictions return to exceed more than one threshold, or no one at all, as for a very small number of latent variables.

A more direct procedure, instead, involves the training of a PLS model on the indicator Y-block and then use it to predict an instance assigning it to the class with the highest estimated class value [4].

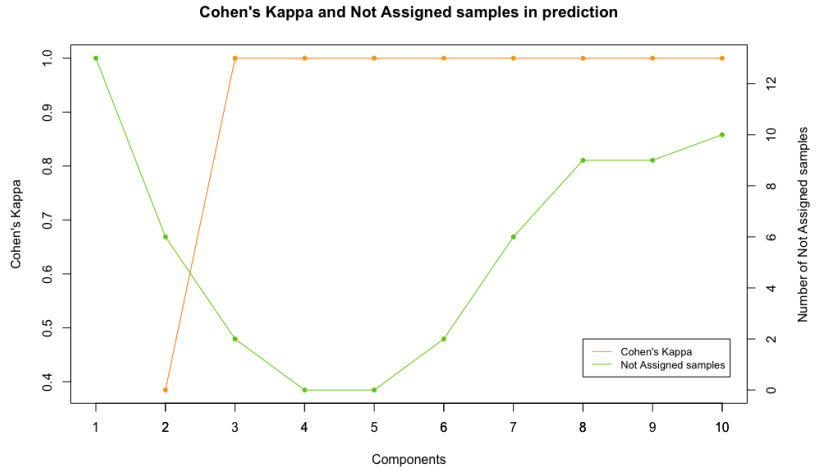


Figure 4.11: Cohen's Kappa and number *not assigned* samples in prediction

It is important to underline that using this approach, all the samples are assigned to one and only one class, without having *not assigned* samples[4].
In Figure 4.12 is shown the result using this method applied to the dataset of olive cultivar, which shows in this case a good prediction ability starting from a model with three components; as can be expected, with a smaller number of components the model is not able to easily distinguish the classes and the results in prediction are worse than the ones in calculation.

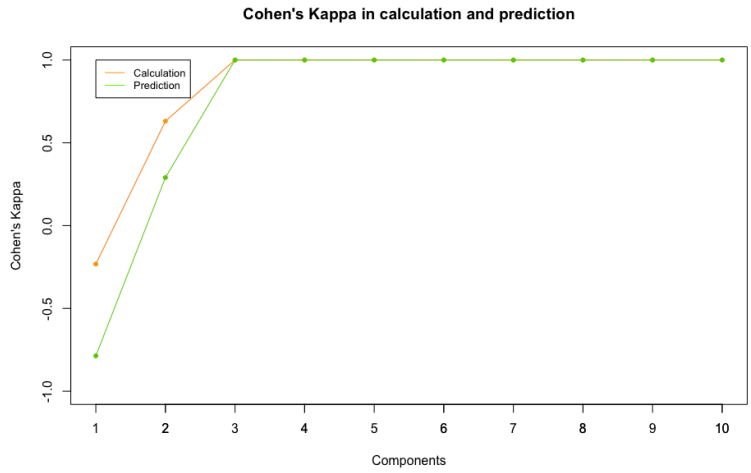


Figure 4.12: Cohen's Kappa in calculation and prediction

[4]In contrast to what happens for the methods outlined in 4.1 that fit the Gaussian distributions on the calculated value in the training phase and then classify using a threshold, with the risk of having *not assigned* samples

The intuition to assign the observation to the class with the highest predicted value is empirical and we want to give this procedure a well-defined mathematical justification: this is possible through the introduction of compositional data (presented in Chapter 6).

Chapter 5

Towards classification

The use of PLS in the context of classification as self-consistent classifier has been presented so far in some papers, even if in general it has not been explored so much in the literature. Some techniques that try to move towards the use of PLS as an effective classifier are now described.
However, first of all it is important to explain why traditional methods, like logistic regression, are not suitable to a high-dimensional scenario.

5.1 Logistic regression in high dimensional setting

When the response variable is categorical, one of the most common approaches to classification is to use logistic regression; however, it is important to understand why conventional techniques, as logistic regression, are not adequate when $p > n$.
Logistic regression is a statistical technique used when the response variable is binary (e.g. success vs failure, alive vs dead, etc) that models the logarithm of the ratio between the probability of success (event $Y = 1$) and the probability of failure (event $Y = 0$) as a linear combination of the predictors. The logarithm of this proportions is called *log-odds* and it corresponds to the *logit* function whose input is the probability of success. Let's define the probability of success $p = P(Y = 1)$. Then, the model is:

$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right) = \beta_0 + x^\top \beta \tag{5.1}$$

where x is the vector containing the values of the covariates of an observation and β is the vector of regression coefficients of the model.
In this way, the link function (i.e. the logit) maps values in $(0, 1)$ to $(-\infty, +\infty)$, which is the same domain of the linear combination of the predictors.

Using the inverse function (logistic) the probability p can be written as:

$$p = \frac{1}{1 + e^{-(\beta_0 + x^\top \beta)}} \tag{5.2}$$

Thus, once β is estimated in the training phase, the prediction of a new observation x_i can be made using [5.2] to find the probability of success and then using a threshold to effectively assign it to one of the two classes. In other words, logistic regression squeezes the output of a linear equation between 0 and 1.
Indeed, the name of the method derives from the fact that the probability of the response being 1 is equal to the value of the logistic function whose input is the linear combination of the X-variables.
The β coefficients of the model may be found via maximum likelihood estimation. Specifically, they can be estimated through Iteratively Re-weighted Least Squares (IRLS) procedure. IRLS is a method to solve optimization problems (as the Least Squares problem) through an iterative procedure that updates a weight vector; in fact, at each iteration the parameters are re-estimated as

$$\hat{\beta}^{t+1} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} w_i(\hat{\beta}^t)|y_i - f_i(\beta)|^2$$

where the optimization problems has an objective function of the form

$$\underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} |y_i - f_i(\beta)|^p$$

with p indicating a p-norm.
This technique is widely applied in several fields, but as for the other traditional methods, a high-dimensional setting poses different problems when trying to build a model. Indeed, the problems caused by the high-dimensional scenario highlighted in Section 2.1 hold also for this method.
In fact, when the number of predictors is (possibly much) larger than the number of observations, the existence and unicity of the vector of coefficients is not guaranteed and also this technique needs a regularization method (as the ridge penalty) to handle the high-dimensionality of the data.
This is one of the reasons why we want to adapt PLS also for the classification purpose. In the following sections, some adaptations of PLS for classification are discussed.

5.2 PLS logistic regression

In this section some classification techniques that try to approach PLS in their formulation are illustrated.
A work by Michel Tenenhaus [36] was written in 2000 to adapt the PLS algorithm to the case of logistic regression.
It was the first attempt to extend PLS towards classification based on a serious thought about what PLS is. Tenenhaus observed that the components of the weights of PLS1 can be calculated considering single OLS regression between the y-response and the residuals of each single predictor. This new formulation of PLS1 driven the development of the logistic version of PLS discussed in the following.
In Tenenhaus [36], a new estimation method is presented to fit a model such that

$$\ln\left(\frac{p}{1-p}\right) = Xb \tag{5.3}$$

where b is the vector of regression coefficients.
The algorithm starts with the computation of the latent variables (m components t_h with $h = 1, 2, \ldots, m$), working on the individual explanatory variables. Given the matrix X made of P centred predictor variables, the first component is found by computing the regression coefficient a_{1j} of x_j in the simple logistic regression of the response variable y on x_j for each variable x_j, with j going from 1 to P.
A vector a_1 is made up using these coefficients, then it is subsequently normalized and w_1 is computed:

$$w_1 = \frac{a_1}{\|a_1\|}$$

Now the scores of the first component are ready to be computed:

$$t_1 = \frac{Xw_1}{{w_1}^\top w_1}$$

In computing the second component of the model, the information captured by the first one must be taken into account; in fact, first of all the residual matrix $E_1 = [e_{11}, e_{12}, \ldots, e_{1P}]$ of the linear regression of X on t_1 is computed. Then, the regression coefficients a_{2j} of each e_{1j} are calculated in the multiple logistic regression of y on t_1 and e_{1j}, $j = 1, \ldots, p$.
As before, the vector is normalized and w_2 is calculated as $w_2 = \frac{a_2}{\|a_2\|}$.

At this step, the second score vector of the model is:

$$t_2 = \frac{X_1 w_2}{{w_2}^\top w_2}$$

The other components are calculated in the same way, adding the information of the previous component at each step.
For the h^{th} latent variable, the residual matrix E_{h-1} is computed on the latent variables $t_1, \ldots, t_{h-1}$.
Then, a_{hj} are the regression coefficients of $e_{h-1,j}$ in the multiple logistic regression of y on $[t_1, \ldots, t_{h-1}, e_{h-1,j}]$, $j = 1, \ldots, P$.
Then, $w_h = \frac{a_h}{\|a_h\|}$ and $t_h = \frac{E_{h-1} w_h}{{w_h}^\top w_h}$.
This procedure assures that the score vectors t_i are a set of orthogonal vectors.
After h iterations, the set of weight vectors $w_i, i = 1, \ldots, h$ and score vectors t_i, $i = 1, \ldots, h$ are obtained, which compose the matrices W and T.
The orthonality of the score vectors is then exploited when making a prediction, according to Equation 5.3: in fact, the following relationship holds

$$\text{logit}(p) = X W^* q^\top = Xb \tag{5.4}$$

where Q is matrix of loadings of the Y-block and $b = W^* q^\top$.
Usually the number of components of the model is choosen via cross-validation in order to have a sufficient number of latent variables to explain the response but at the same time avoid overfitting.
The technique has not been adapted to the general case of G classes: the G-class classification problem can be solved using pairs of logistic-PLS models according to the standard approach used for logistic regression.

5.3 Other methods

Other methods have been designed towards the classification using PLS [7]; one of these was formulated by Nguyen and Rocke [22] and, given a number of latent components A, applies a PLS model on Y using the X-variables as predictors, finding the score matrix T. Then parameters are estimated by running on Y and T the IRLS in the classical logistic regression framework.

A second method called Iteratively Re-weighted PLS (IRPLS) was developed by Marx [20] and tries to extend the concept of PLS into the framework of generalized linear models.

IRPLS is essentially an IRLS algorithm in which the PLS regression is used instead of the weighted least squares regression; it is worth noting that using the maximum number of PLS components corresponds to the least squares case. Moreover, in [20] the number of components is chosen to be equal to rank(X): this choice can be not appropriate in the "large p small n" scenario since in this case the algorithm never converges. Other authors [23] use a similar procedure with $A <$ rank(X). The principal drawback of this method is that convergence is not guaranteed.

Another method was designed in a work by Fort and Lambert-Lacroix [12], in which a new procedure is proposed to combine PLS and Ridge penalized logistic regression to extend PLS to the logistic regression model.
They suggest to replace the binary y response with a pseudoresponse variable (indicated as z^{∞} at convergence of the algorithm) that has an expected value with a linear relationship with the covariates. This new dependent variable has some properties; in fact, at convergence of the Ridge-IRLS (RIRLS), which is a version of IRLS that substitutes the weighted regression with a weighted RR, it can be written as $z^{\infty} = X\hat{\gamma}^R + e$, where $\hat{\gamma}^R$ is the vector of regression coefficients. The method is called R-PLS since a ridge approach is applied to PLS: in this case, two parameters must be found by cross-validation, i.e. λ (the ridge parameter) and A (the number of latent variables).

A different approach to classification using PLS for a multiclass problem is that of Wang et al. [37] , in which a classification of four stages of cancer development ("normal", "hyperplasia", "dysplasia" and "early cancer") is performed in two steps.
First of all, the PLS components are extracted using the original explanatory variables and the response block, coded in three indicator variables (assumed as unordered).
Then, the following is assumed:

$$\ln\left(\frac{p_k}{p_1}\right) = \beta_{0k} + {\beta_k}^{\top} s_k \tag{5.5}$$

where $p_k = P(\text{class } k|s_k)$ and s_k is the vector of values of the PLS components for an instance belonging to class k.
The regression coefficients are computed through maximum likelihood and a sample is classified according to the class with the highest predicted probability from the logistic regression.

Finally, a work regarding the use of PLS in the context of classification was

presented in 2007 by Tenenhaus A. et al [35] and focused on a new algorithm, called KL-PLS (Kernel Logistic PLS), which essentially is a tool for supervised nonlinear dimensionality reduction and binary classification.
Indeed, it can be seen as a supervised dimensionality reduction method followed by a classification based on logistic regression.
The main idea of KL-PLS is to look for a discriminant space spanned by the KL-PLS components $(t_1, \ldots, t_m)$, where a simple model, such as the logistic regression may become efficient for classification.
In order to find such space, a kernel matrix is used to map the data into a higher-dimensional space (with respect to the original) where the probability of finding the hyperplane increases with the dimension of the space.
The scheme of KL-PLS is quite intuitive:

1. The kernel matrix is computed
2. The m KL-PLS components are calculated
3. Logistic regression of the response matrix Y on the m retained latent variables is performed

Let X be a $N \times P$ matrix, with N observations and P explanatory variables and y the dependent variable. A kernel matrix K associated to X is a $N \times N$ matrix in which each cell k_{ij} represent the inner product between individuals i and j in the feature space $\mathcal{F}$; it follows that each column k_j, $j = 1, \ldots, N$ represent the similarity between the individual j and the whole dataset.
To determine the first KL-PLS component, a sequence of N simple logistic regressions must be carried out:

Step 1: Compute the coefficients a_{1j} of k_j in the binary logistic regression of y on each k_j, $j = 1, \ldots, N$.
Step 2: Normalization step, $w_1 = \dfrac{a_1}{\|a_1\|}$
Step 3: Calculate the first score vector $t_1 = Kw_1$

The h^{th} component tries to capture as much as possible the discriminant information not explained by the previous components.
Thus, at each step the residual matrix must be computed, that is $k_{h-1,1}, \ldots, k_{h-1,N}$ are calculated from the multiple regression of each k_j on $t_1, \ldots, t_{h-1}$, giving K_{h-1} as result.
Again, the h^{th} latent variable is determined through N logistic regressions:

Step 1: Compute a_{hj} of $k_{h-1,j}$ by performing a binary logistic regression of y on $[t_1, \ldots, t_{h-1}, k_{h-1,j}]$, with $j = 1, \ldots, N$

Step 2: Normalize the vector, i.e. $w_h = \frac{a_h}{\|a_h\|}$

Step 3: Calculate the h^{th} score vector: $t_h = K_{h-1} w_h$

Often it is useful to express the values of the score vector of a latent variable in terms of the original variables, that is finding a vector ${w_h}^*$ such that $t_h = K{w_h}^*$: this allows to compute directly the KL-PLS components for a new data set, using the relationship $t_{test} = K_{test} W^*$.
It must be noticed that the first component is already expressed in terms of the original variables, because ${w_1}^* = w_1$ (and $t_1 = Kw_1$).
The second component and the following ones are expressed in terms of the residuals, so ${w_h}^* \neq w_h$, for $h > 2$.
In the case of the second component, let p_{1j} be the regression coefficient of t_1 in the regression of k_j on t_1, then $K = t_1 {p_1}^\top + K_1$ and $t_2 = K_1 w_2$.
Thus we get:

$$\begin{aligned} t_2 = K_1 w_2 = (K - t_1 {p_1}^\top) w_2 = (K - K w_1 {p_1}^\top) w_2 = \\ = K(I - w_1 {p_1}^\top) w_2 = K {w_2}^* \end{aligned} \tag{5.6}$$

where ${w_2}^* = (I - w_1 {p_1}^\top) w_2$.
Extending this reasoning to the general case of the h^{th} component, t_h can be expressed as:

$$\begin{aligned} t_h = K_{h-1} w_h = (\sum_{i=1}^{k-1} t_i {p_i}^\top) w_h = (\sum_{i=1}^{k-1} K {w_i}^* {p_i}^\top) w_h = \\ = K(I - \sum_{i=1}^{k-1} {w_i}^* {p_i}^\top) w_h = K {w_h}^* \end{aligned} \tag{5.7}$$

where ${w_h}^* = (I - \sum_{i=1}^{k-1} {w_i}^* {p_i}^\top) w_h$.
Also in this case, the choice of the number of components m is usually guided by cross-validation.

However, as it can be noticed those methods are designed in such a way that PLS is initially used as first step for the discriminatory phase (calculating the components and trying to separate the classes); subsequently, an analysis is made on the PLS scores.
For this reason, we want to design a new technique purely based on PLS, which acts as a classification tool (which is not the case of PLS-DA).

Chapter 6

PLS for Classification

The methods described so far (especially PLS-DA in Chapter 4) are nowadays widely used and although they work and often give consistent results, their approach is empirical or PLS is used only as an initial discriminatory step and needs other classifier to effectively assign an observation to a specific class.
Thus, in this Chapter we want to present the use of PLS with a different approach: to understand how this is possible, let's start to see PLS from a new point of view, through whose extension we can get to the new methods, which are then illustrated after the introductory part.

6.1 PLS and Gradient Descent

In addition to how it has already been described in Chapter 2, PLS can be seen as an algorithm that solves the least squares problem through an iterative approach where at each step the residuals of the X-block and Y-block are regressed using the first approximated solution obtained by steepest descent method. From this point of view, the concept of latent variables by projection are introduced only later since it is not strictly necessary to define the method (actually they are only two different versions to solve the same problem).
The objective function of PLS[1] arises from the application of the steepest descent method maximizing the directional derivative and it is fully justified in the framework, as it is explained in the following sections.
Introducing PLS as a means of solving the least squares is very important because starting from this point of view it is possible to modify the algorithm for more general purposes, in the direction in which we want to move, therefore trying to adapt it to classification. Given this general introduction, let's proceed with the first case, i.e. PLS1.

[1] Maximization of covariance between the scores in the X and Y spaces

6.1.1 A new formulation for PLS1

PLS1 is the special case of PLS in which the response variable has only one dimension, so y is a $N \times 1$ vector, where N is the number of observations.
Given a matrix of predictors X ($N \times P$) and a vector y ($N \times 1$), that we consider mean-centred and scaled, let's consider the linear regression problem:

$$y = Xb + e \tag{6.1}$$

where b ($P \times 1$) is the vector of regression coefficients and e ($N \times 1$) is the vector of errors (no assumptions regarding the errors are done).
The core of the iterative procedure here described is the application of the gradient descent to reduce the distance between y or its residuals and its approximation obtained considering a subspace of X.
Let's define f_i the vector of residuals of y at step i[2] and E_i a subspace of the column space of X at iteration i; the aim is to find the weight vectors such that:

$$\tilde{w}_{i+1} = \underset{w}{\operatorname{argmin}} \|f_i - E_i w\|_F^2 \tag{6.2}$$

being the solution $\tilde{w}_{i+1} = \tilde{w}_{i+1}^{(0)} + \Delta\tilde{w}_{i+1} + \dots$, where $\Delta\tilde{w}_{i+1} = \alpha_{i+1} w_{i+1}$ and $\|w_{i+1}\|_F = 1$. The Frobenius norm is used since in this Chapter we have to deal also with norm of matrices.
Starting from the initial guess $\tilde{w}_{i+1}^{(0)} = 0_p$, the solution at the first iteration of the steepest descent algorithm is:

$$\tilde{w}_{i+1} \approx \Delta\tilde{w}_{i+1} = \alpha_{i+1} w_{i+1} \tag{6.3}$$

where

$$w_{i+1} \propto -\frac{\partial \|f_i - E_i \tilde{w}_{i+1}\|_F^2}{\partial \tilde{w}_{i+1}}\Big|_{\tilde{w}_{i+1}^{(0)} = 0_p} = E_i^\top f_i \text{ i.e. } w_{i+1} = \frac{E_i^\top f_i}{\|E_i^\top f_i\|} \tag{6.4}$$

and

$$\alpha_{i+1} = \underset{\alpha}{\operatorname{argmin}} \|f_i - \alpha E_i w_{i+1}\|_F^2 \text{ , i.e. } \alpha_{i+1} = \frac{f_i^\top E_i E_i^\top f_i}{f_i^\top E_i E_i^\top E_i E_i^\top f_i} \|E_i^\top f_i\| \tag{6.5}$$

[2] At the beginning of the algorithm, f_i is exactly y

It is worth noting that the same solution is obtained by considering the direction w_{i+1} that maximizes the following directional derivative:

$$\underset{w_{i+1}}{\operatorname{argmax}} \nabla_{w_{i+1}} \|f_i - E_i \tilde{w}_{i+1}\|_F^2 = \underset{w_{i+1}}{\operatorname{argmax}} \nabla \|f_i - E_i \tilde{w}_{i+1}\|_F^2 \Big|_{\tilde{w}_{i+1}=0_p} \quad w_{i+1} \propto \underset{w_{i+1}}{\operatorname{argmax}} f_i^\top E_i \tilde{w}_{i+1}$$

that is $w_{i+1} \propto E_i^\top f_i$ and $\alpha_{i+1} = \underset{\alpha}{\operatorname{argmin}} \|f_i - \alpha E_i w_{i+1}\|_F^2$.

The algorithm is the following:

Algorithm 5: PLS1 - "Gradient descent" PLS1 algorithm

$f_0 = y$;
$E_0 = X$;
for $i = 1, \ldots, A$ **do**
  $w_i = \frac{E_{i-1}^\top f_{i-1}}{(f_{i-1}^\top E_{i-1} E_{i-1}^\top f_{i-1})^{\frac{1}{2}}}$;
  $\alpha_i = \frac{w_i^\top E_{i-1}^\top f_{i-1}}{w_i^\top E_{i-1}^\top E_{i-1} w_i}$;
  $E_i = \hat{Q}_{E_{i-1} w_i} E_{i-1}$;
  $f_i = f_{i-1} - \alpha_i E_{i-1} w_i$;
end

where $\hat{Q}_{E_{i-1}w_i} = I_N - (w_i^\top E_{i-1}^\top E_{i-1} w_i)^{-1} E_{i-1} w_i w_i^\top E_{i-1}^\top$ is the orthogonal projection matrix that projects a vector v $(P \times 1)$ onto the space orthogonal to $E_{i-1}w_i$. After A iterations, the response variable y is decomposed as:

$$y = X\beta_A + f_A \tag{6.6}$$

where f_A is the vector of residuals after A iterations and the vector β_A is:

$$\beta_A = W(W^\top X^\top X W)^{-1} W^\top X^\top y \tag{6.7}$$

being W the matrix of weights, i.e. $W = [w_1, \ldots, w_A]$.
The regression model in 6.1 is obtained if we consider $b = \beta_A$ and $f_A = e$.
In Appendix B.1 it is proved that the algorithm solves the least squares problem.

Moreover, one can prove that the algorithm is the PLS1 algorithm. Introducing the vector

$$t_i = E_{i-1} w_i \tag{6.8}$$

the residuals of the y-block can be written as:

$$f_i = f_{i-1} - \alpha_i E_{i-1} w_i = \hat{Q}_{t_i} f_{i-1} \tag{6.9}$$

and the algorithm becomes:

Algorithm 6: "Eigenvalue" PLS1 algorithm

1 $f_0 = y$;
2 $E_0 = X$;
3 **for** $i = 1, \ldots, A$ **do**
4 $\quad w_i = \frac{E_{i-1}^\top f_{i-1}}{(f_{i-1}^\top E_{i-1} E_{i-1}^\top f_{i-1})^{\frac{1}{2}}}$;
5 $\quad t_i = E_{i-1} w_i$
6 $\quad E_i = \hat{Q}_{t_i} E_{i-1}$;
7 $\quad f_i = \hat{Q}_{t_i} f_{i-1}$;
8 **end**

The "gradient descent" PLS1 algorithm will be used to solve the classification problem with 2-classes. It is interesting to note that steps 4 and 5 characterize the algorithm in the least squares sense.
On the other hand, steps 4 and 7 in the "eigenvalue" PLS1 algorithm are the ones that characterize the algorithm as a least squares solver, since Step 4 is used to find the direction that minimizes the least squares loss function ($\|f_i - E_i \tilde{w}_{i+1}\|_F^2$) and Step 7 is a least squares regression considering the latent variable t_i.

6.1.2 A new formulation for PLS

Let's now come to the most general form of PLS algorithm (recall that PLS is used to stand PLS2), which is able to handle more than one response variable at the same time. Given a multiple Y-response ($N \times K$) and the matrix X ($N \times P$) of the predictors, that are mean-centred and scaled, the residual matrix F_i and the matrix E_i belonging to the column subspace of X, the objective function to be minimized in this case at iteration i is:

$$\|F_i - E_i \tilde{W}_{i+1}\|_F^2 = \text{tr}(F_i^\top F_i) - 2\text{tr}(F_i^\top E_i \tilde{W}_{i+1}) + \text{tr}(\tilde{W}_{i+1}^\top E_i^\top E_i \tilde{W}_{i+1}) \tag{6.10}$$

The idea consists in considering the directional derivative

$$\nabla_{W_{i+1}} \|F_i - E_i \tilde{W}_{i+1}\|_F^2 \tag{6.11}$$

and looking for W_{i+1} that maximizes 6.11, i.e.:

$$\underset{\|W_{i+1}\|_F^2 = 1}{\text{argmax}} \nabla_{W_{i+1}} \|F_i - E_i \tilde{W}_{i+1}\|_F^2 = \underset{\|W_{i+1}\|_F^2 = 1}{\text{argmax}} \text{tr}\left(\nabla \|F_i - E_i \tilde{W}_{i+1}\|_F^2 \Big|_{\tilde{W}_{i+1}=0} W_{i+1} \right)$$

Since $\nabla f(X) = \left(\frac{\partial f}{\partial X_{ij}}\right)$, we get:

$$\nabla \|F_i - E_i \tilde{W}_{i+1}\|_F^2 \Big|_{\tilde{W}_{i+1}=0} = F_i^\top E_i \tag{6.12}$$

and also

$$\underset{\|W_{i+1}\|_F^2=1}{\operatorname{argmax}} \nabla_{W_{i+1}} \|F_i - E_i \tilde{W}_{i+1}\|_F^2 = \underset{\|W_{i+1}\|_F^2=1}{\operatorname{argmax}} \operatorname{tr}(F_i^\top E_i W_{i+1}) \tag{6.13}$$

If the SVD of $F_i^\top E_i$ is introduced as

$$F_i^\top E_i = C_i S_i V_i^\top \tag{6.14}$$

with $C_i = [c_1^i, \ldots, c_a^i]$, $V_i = [w_1^i, \ldots, w_a^i]$, $S_i = \operatorname{diag}(s_j^i)$, we get:

$$\underset{\|W_{i+1}\|_F^2=1}{\operatorname{argmax}} \nabla_{W_{i+1}} \|F_i - E_i \tilde{W}_{i+1}\|_F^2 = \underset{\|W_{i+1}\|_F^2=1}{\operatorname{argmax}} \operatorname{tr}(C_i S_i V_i^\top W_{i+1}) \tag{6.15}$$

The maximum of this quantity is obtained when $W_{i+1} = w_1^i c_1^{i\top}$; in fact, in this case

$$\operatorname{tr}(F_i^\top E_i W_{i+1}) = \operatorname{tr}(F_i^\top E_i w_1^i c_1^{i\top}) = c_1^{i\top} F_i^\top E_i w_1^i = s_1^i \tag{6.16}$$

We recall that w_1^i satisfies the following equation

$$E_i^\top F_i F_i^\top E_i w_1^i = s_1^{(i)2} w_1^i \tag{6.17}$$

and also

$$c_1^i = \frac{1}{s_1^i} F_i^\top E_i w_1^i \tag{6.18}$$

Therefore, in this setting the first approximation of $\tilde{W}_{i+1}$ if $\tilde{W}_{i+1}^{(0)} = 0$ is considered is

$$\tilde{W}_{i+1} = \alpha_{i+1} w_1^i c_1^{i\top} \tag{6.19}$$

The term α_{i+1} is a scalar equal to $\operatorname{argmin} \|F_i - \alpha_{i+1} E_i w_1^i c_1^{i\top}\|_F^2$ and it is defined as:

$$\alpha_{i+1} = \frac{\operatorname{tr}(F_i^\top E_i w_1^i c_1^{i\top})}{\operatorname{tr}(c_1^i w_1^{i\top} E_i^\top E_i w_1^i c_1^{i\top})} = \frac{s_1^i}{w_1^{i\top} E_i^\top E_i w_1^i} \tag{6.20}$$

Thus, the proposed algorithm in this case is:

Algorithm 7: "Gradient descent" PLS algorithm

$F_0 = Y$;
$E_0 = X$;
for $i = 1, \dots, A$ **do**
  solve $E_{i-1}^\top F_{i-1} F_{i-1}^\top E_{i-1} w_i = s_i^2 w_i$;
  $c_i = \frac{1}{s_i} F_{i-1}^\top E_{i-1} w_i$;
  $\alpha_i = \frac{s_i}{w_i^\top E_{i-1}^\top E_{i-1} w_i}$;
  $E_i = \hat{Q}_{E_{i-1} w_i} E_{i-1}$;
  $F_i = F_{i-1} - \alpha_i E_{i-1} w_i c_i^\top$;
end

Thus after A iterations of the procedure, the response is decomposed as:

$$Y = X\beta_A + F_A \tag{6.21}$$

where

$$\beta_A = W(W^\top X^\top X W)^{-1} W^\top X^\top Y \tag{6.22}$$

with $W = [w_1, \dots, w_A]$.
When $A = \text{rank}(X)$, given the SVD of X: $X = USV^\top$,

$$\beta_A = VS^{-1}U^\top Y \tag{6.23}$$

and the least squares problem is solved.
Moreover, introducing the score vectors t_i, $i = 1, \dots, A$ defined as in 6.8, the residuals of the X and Y blocks can be written as:

$$\begin{aligned} E_i &= \hat{Q}_{t_i} E_{i-1} \\ F_i &= F_{i-1} - \alpha_i E_{i-1} w_i c_i^\top = F_{i-1} - \frac{s_i}{t_i^\top t_i} t_i \frac{1}{s_i} t_i^\top F_{i-1} = \hat{Q}_{t_i} F_{i-1} \end{aligned} \tag{6.24}$$

and noting that the vectors w_i are equal to the PLS-weight vectors, we can conclude that the algorithm is equivalent to the PLS algorithm introduced in section 2.3.
So, the matrix of regression coefficients of PLS is the one presented in 6.22 and the blocks are decomposed as follows:

$$X = TP^\top + E_A \tag{6.25}$$

$$Y = T(T^\top T)^{-1}T^\top Y + F_A = XB + F_A \tag{6.26}$$

where $T = [t_1, \ldots, t_A]$, $P = X^\top T(T^\top T)^{-1}$ and $B = \beta_A$, according to PLS.
It is worth noting that the proposed derivation of PLS does not require the introduction of the concept of latent variable and that the objective function of PLS has been derived minimizing the least squares loss function without assumptions about the covariance of X and Y. Moreover, steps 4 and 5 of the algorithm are the solution of the problem

$$w_i, c_i = \underset{w,c}{\operatorname{argmax}} \, (w^\top E_i^\top F_i c) \tag{6.27}$$

that is the maximization problem 2.6 used in the standard formulation of PLS. This formulation will be used to solve the G-class classification problem with $G > 2$ thanks to the capability of PLS to handle more than one response at the same time.

6.2 "Gradient descent" PLS and regularization

The iterative procedure used to define "gradient descent" PLS (and that of PLS1 in the case of single y-response) generates a series of estimations of the solution of the least squares problem and converges to the least squares solution when the maximum number of iterations is performed. However, the least squares solution may be a suboptimal solution in the case of real data. Indeed, the variance of the coefficients may result too large and model generalization may be hindered due to over-fitting. As a consequence, regularization is introduced to balance bias and variance in order to obtain good performance in predictions and improve generalization. In the case of PLS, regularization is not performed constraining the norm of the regression coefficients as for Ridge or Lasso regression but applying the so-called early stopping that is a general methods usually applied to control the complexity of the solution obtained by iterative methods [27]. In early stopping, the algorithm used to solve the problem is stopped at a certain iteration on the basis of some stopping rules. In PLS, strategies of cross-validation are usually applied to determine the iteration that generates the minimum of the estimated error in prediction since an analytical expression of the error in prediction is not known, and the algorithm is stopped. As a results, the regression coefficients are regularized because they belong to a subspace of the column space of X with a dimension much smaller than $\text{rank}(X)$.

6.3 Transformation of the Y-response and PLS

The formulation of PLS based on gradient descent is suitable to solve the problem of regression when functions are used to transform the Y-response. We consider the function

$$g : \mathcal{R}^K \to \mathcal{R}^L \tag{6.28}$$

that transforms the response y into $g(y) = [g_1(y), \ldots, g_L(y)]^\top = z \in \mathcal{R}^L$. Moreover, we assume that the inverse $g^{-1} : \mathcal{R}^L \to \mathcal{R}^K$ of g exists, i.e. $g^{-1}g(y) = y$ and $gg^{-1}(z) = z$.
The following regression model is considered

$$g(Y) = XB + E \tag{6.29}$$

where $B(P \times L)$ is the matrix of regression coefficients and $E(N \times L)$ is the matrix of residuals. We assume that X is mean-centred and scaled. The model parameters can be estimated by the following algorithm

Algorithm 8: "Gradient descent" PLS - X-space

1 $F_0 = g(Y)$;
2 $E_0 = X$;
3 **for** $i = 1, \ldots, A$ **do**
4 solve $\underset{\|W_i\|_F^2=1}{\text{argmax}} \nabla_{W_i} \|F_{i-1} - E_{i-1}\tilde{W}_i\|_F^2 \Big|_{\tilde{W}_i=0}$;
5 calculate $\alpha_i = \underset{\alpha}{\text{argmin}} \|F_{i-1} - \alpha E_{i-1} W_i\|_F^2$;
6 $F_i = F_{i-1} - \alpha_i E_{i-1} W_i$;
7 Deflation step $E_i \leftarrow E_{i-1}$;
8 **end**

that corresponds to the "gradient descent" PLS algorithm where $g(Y)$ is used instead of Y.
Specifically, the solution of step 4 is

$$\begin{aligned} &W_i = w_i c_i^\top \text{ such that } E_{i-1}^\top F_{i-1} F_{i-1}^\top E_{i-1} w_i = s_i^2 w_i \\ &c_i = \frac{1}{s_i} F_{i-1}^\top E_{i-1} w_i \end{aligned} \tag{6.30}$$

while

$$\alpha_i = \frac{s_i}{w_i^\top E_{i-1}^\top E_{i-1} w_i} \tag{6.31}$$

and the deflation step 7 is

$$E_i = \hat{Q}_{E_{i-1}w_i} E_{i-1} \tag{6.32}$$

After A iterations, the matrix of the regression coefficients is

$$B = W(W^\top X^\top XW)^{-1} W^\top X^\top Y \tag{6.33}$$

and

$$E = F_A$$

It is interesting to note that step 4 and 5 are solved in the Euclidean space where the matrix of the predictors is defined. We called this approach "regression in the X-space".
When the inverse transformation g^{-1} is taken into account, the regression model

$$g(Y) = XB + E$$

can be solved considering the following iterative algorithm

Algorithm 9: "Gradient descent" PLS - Y-space

1 $F_0 = Y$;
2 $E_0 = X$;
3 **for** $i = 1, \ldots, A$ **do**
4 solve $\underset{\|W_i\|_F^2=1}{\operatorname{argmax}} \nabla_{W_i} \|F_{i-1} - g^{-1}(E_{i-1}\tilde{W}_i)\|^2$;
5 calculate $\alpha_i = \underset{\alpha}{\operatorname{argmin}} \|F_{i-1} - g^{-1}(\alpha E_{i-1} W_i)\|^2$;
6 $F_i = g^{-1}(g(F_{i-1}) - \alpha_i E_{i-1} W_i)$;
7 Deflation step $E_i \leftarrow E_{i-1}$;
8 **end**

that corresponds to the "gradient descent" PLS algorithm where g^{-1} is used to transform the matrices in the X-space into its representation in the Y-space. We called this approach "regression in the Y-space". It is worth noting that the differences in steps 4 and 5 are calculated in the Y-space that could not be a Euclidean space (e.g. it could be a simplex).
After A iterations, the following Y-block decomposition is obtained

$$F_A = g^{-1}\left(g(Y) - \sum_{i=1}^{A} \alpha_i E_{i-1} W_i\right) = g^{-1}\left(g(Y) - XB\right) \text{ i.e. } g(Y) = XB + g(F_A) \tag{6.34}$$

having E_i the general form $E_i = XZ_i$ where Z_i depends on the deflation step 7. In this case we have

$$E = g(F_A) \tag{6.35}$$

If g^{-1} is a linear function, we have

$$Y = g^{-1}(XB) + F_A \tag{6.36}$$

A special case of functions used to transform Y is the link function used in the logistic regression that is discussed in the next section.

6.4 PLS for classification

We now present the central part of this thesis, focused on explaining how the classification problem can be solved by PLS through various methods, each exploiting different properties. Both the 2-class problem and the framework of its generization to $G > 2$ classes are considered.

6.4.1 2-class classification problem

In the following, we shall consider two classes A and B, and the related conditional probabilities

$$0 < P(class = j|x_i) < 1 \text{ with } j = \text{A, B} \tag{6.37}$$

being $x_i \in \mathcal{R}^P$ the vector of predictors of observation i. Given a training set of N_A observations of class A and N_B observations of class B, we shall assume that all the observations of the same class show the same conditional probability and specifically that

$$P(class = A|x_i) = 1 - \epsilon \text{ if the observation } i \text{ belongs to class A}$$

and

$$P(class = A|x_i) = \epsilon \text{ if the observation } i \text{ belongs to class B}$$

for $0 < \epsilon < \frac{1}{2}$. Moreover, we shall consider the regression model

$$g(y_i) = x_i^\top b + e_i \tag{6.38}$$

where $y_i = P(class = A|x_i)$, the vector of regression coefficients is b and e_i is the vector of errors.

6.4.2 2-class classification problem solved by logistic-like method in the X-space

In this section, the transformation of the Y-response called logit function

$$g(y_i) = \ln\left(\frac{y_i}{1-y_i}\right), \text{ with } y_i \in (0,1) \tag{6.39}$$

whose inverse is the logistic function

$$g^{-1}(x_i) = \frac{1}{1+e^{-x_i}}, \text{ with } x_i \in (-\infty, +\infty) \tag{6.40}$$

is considered.
For the observations of the training set, the values of the logit function are

$$g(y_i) = \ln\left[\frac{P(class = A|x_i)}{1-P(class = A|x_i)}\right] = \ln\left(\frac{1-\epsilon}{\epsilon}\right) > 0 \text{ for observations of class A}$$

and

$$g(y_i) = \ln\left[\frac{P(class = A|x_i)}{1-P(class = A|x_i)}\right] = -\ln\left(\frac{1-\epsilon}{\epsilon}\right) < 0 \text{ for observations of class B}$$

Here, we estimate the vector of the regression coefficients using the PLS "regression in the X-space" approach. The "regression in the Y-space" approach will be considered only after the introduction of the compositional data. Compositional data theory is required to properly take into account the different geometry of the X and Y spaces. It is worth noting that we can directly apply the "regression in the X-space" approach to the transformed response ignoring the geometry of the y-transformed space because the logit function is an isomorphism. Indeed, it corresponds to the additive-logratio transformation for a binary mixture, as it will be discussed in section 6.4.4.

The vector $g(y)$ of the responses is

$$g(y) = \ln\left(\frac{1-\epsilon}{\epsilon}\right)\begin{bmatrix} 1_{N_A} \\ -1_{N_B} \end{bmatrix} \tag{6.41}$$

where 1_{N_i} is the vector with N_i elements equal to 1.

After mean centering (being the mean vector $\bar{g} = \ln(\frac{1-\epsilon}{\epsilon})(\frac{N_A-N_B}{N_A+N_B})$), the vector of the responses is

$$\tilde{g}(y) = 2\ln\left(\frac{1-\epsilon}{\epsilon}\right)\frac{1}{N_A+N_B}\begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix} \tag{6.42}$$

and the solution of step 4 at iteration i is the weight vector

$$w_i \propto E_{i-1}^{\top} \begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix}, \text{ i.e. } w_{ij} \propto (e_{jA}^{-(i-1)} - e_{jB}^{-(i-1)})$$

where $e_{jk}^{-(i-1)}$ is the mean of the residuals of the predictor j at the iteration $i-1$ calculated using the observations of class k. It is interesting to note that the weight vectors are independent of ϵ and that they are the same obtained by PLS-DA for a 2-class problem. Moreover, they are a set of orthonormal vectors[3]. After A iterations the following decompositions are obtained

$$X = TP^{\top} + E_A \tag{6.43}$$

and

$$\tilde{g}(y) = \sum_{i=1}^{A} \alpha_i t_i + F_A = XW^*\alpha + F_A \tag{6.44}$$

The vector of the regression coefficients is

$$b = W^*\alpha = 2\ln\left(\frac{1-\epsilon}{\epsilon}\right) \frac{1}{N_A + N_B} W^*(T^{\top}T)^{-1}T^{\top} \begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix} \tag{6.45}$$

and the vector of the residuals is F_A. The matrices T, P, W^* are defined according to the standard PLS.

The class of a new observation x is predicted estimating the logit function by

$$\begin{aligned} \ln\left(\frac{P(class = A|x_{new})}{P(class = B|x_{new})}\right) &= (x_{new} - \bar{x})^{\top} L_{scaling} b + \bar{g} \\ &\propto (x_{new} - \bar{x})^{\top} L_{scaling} W^*(T^{\top}T)^{-1}T^{\top} \begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix} + \frac{1}{2}(N_A - N_B) \end{aligned}$$

where $\bar{x}$ is the vector of the mean values of the columns of X and $L_{scaling} = \text{diag}(\frac{1}{f_i})$ is the scaling matrix (f_i is a parameter that depends on the type of scaling that is applied). If the class membership is attributed on the basis of the maximum conditional probability, the predicted class is independent of ϵ because it depends only on the sign of the logit function.

[3]Since for $j < i$ we have $E_{i-1} = Z_i \hat{Q}_{t_j} E_{j-1}$, then, $w_j^{\top} w_i = w_j^{\top} E_{i-1}^{\top} c_i = w_j^{\top} E_{j-1}^{\top} \hat{Q}_{t_j} Z_j c_i = t_j^{\top} \hat{Q}_{t_j} Z_j c_i = 0$; for $j = i$, we have $w_j^{\top} w_i = w_i^{\top} w_i = 1$

Again, the score latent space can be post-transformed according to the post-transformation procedure proposed for PLS [33][29] obtaining a single predictive latent variable and the following model of $g(y)$

$$g(y) = t_p q_p + \bar{g} + f_A$$

where $q_p = \frac{t_p^\top \tilde{g}(y)}{t_p^\top t_p}$, is obtained. It allows the calculation of the conditional probability

$$P(class = A|x_i) = \frac{1}{1 + e^{-(t_{pi} q_p + \bar{g})}} \tag{6.46}$$

6.4.3 Some notes about compositional data

Compositional data theory is one of the keys of this work. It allows us to consider the real nature of the probabilities to belong to a given set of classes, that is the constraint to sum to 1. From an algebraic point of view, the Y-space of the probability and the X-space of the predictors are treated with the right geometry and their relationships studied using suitable isomorphisms.

Background

Compositional data are vectors whose components are the proportion of some whole, carrying relative information; their fundamental property is that the sum is constrained to be equal to some constant (e.g. 1 for proportions, 100 for percentages or other constant c for other cases).
Such type of data must be properly processed, since they are not simple vectors of real numbers: in fact, strange behaviours can occur when statistical methods for unconstrained data are applied to compositional data.
This is due to the fact that the space of compositional data is different from the real Euclidean space associated to uncostrained data.

Compositional data are present in many areas, such as petrology or geology (e.g. for geochemical or mineral compositions of rocks), economics (portfolio compositions), geography (land use compositions), agriculture, biology, medicine, etc. [2]

As previously said, standard statistical methods lose their applicability and interpretation when dealing with compositional data, so a new approach was designed by J. Aitchison in the 80's, defining a theory based on log-ratios: a mathematical foundation of the analysis of compositional data is based on the

definition of a specific geometry on the simplex, through which it is possible to perform rigorous analysis using such data.

The sample space of compositional data is the simplex:

$$\mathcal{S}^G = \{x = [x_1, x_2, \ldots, x_G] | x_i > 0, i = 1, \ldots, G, \sum_{i=1}^{G} x_i = \kappa\} \tag{6.47}$$

which means that x is a G-part composition whose components are stricly positive real numbers that sum to κ.

Moreover, for any vector of G real strictly positive component of the form

$$z = [z_1, z_2, \ldots, z_G] \in \mathcal{R}_+^G \tag{6.48}$$

the closure of z is:

$$\mathcal{C}(z) = \left[\frac{\kappa \cdot z_1}{\sum_{i=1}^{G} z_i}, \frac{\kappa \cdot z_2}{\sum_{i=1}^{G} z_i}, \ldots, \frac{\kappa \cdot z_G}{\sum_{i=1}^{G} z_i}\right] \tag{6.49}$$

resulting in the same vector rescaled so that the sum of the components is equal to κ, with κ depending on the measurements.

Euclidean geometry is not a proper tool when dealing with compositional data; as an example, let's consider two couples of compositions:

$$[10, 80, 10], [20, 70, 10]$$

and

$$[60, 30, 10], [70, 20, 10]$$

The Euclidean distance between each couple is undoubtedly the same, but in the first couple the first component is doubled, while in the second case it increases by only 16%, so since we are dealing with proportions, the percentage increase seems to be much more suitable for the analysis of compositions than the absolute difference and thus the difference between those two couples does not appear to be the same.
This is only one possible example that motivates the need of defining a new geometry to analyze this type of data.
First of all, two operations which give the simplex a vector space structure must be defined.

Vector space structure

The perturbation operation (analogous to addition in the Euclidean space) of a composition $x \in \mathcal{S}^G$ by a composition $y \in \mathcal{S}^G$ is defined as:

$$x \oplus y = \mathcal{C}[x_1 y_1, x_2 y_2, \ldots, x_G y_G] \tag{6.50}$$

Power transformation (equivalent to multiplication by a scalar in real space) of a composition $x \in \mathcal{S}^G$ by a constant $\alpha \in \mathcal{R}$ is defined as:

$$\alpha \odot x = \mathcal{C}[x_1^\alpha, x_2^\alpha, \ldots, x_G^\alpha] \tag{6.51}$$

The simplex $(\mathcal{S}^G, \oplus, \odot)$ with the perturbation operation and power transformation is a vector space.
The properties are reported in Appendix B.3. Definitions of inner product, norm and distance in this space to get a linear vector space structure are given in Appendix B.4.

Mapping the simplex and the Euclidean space

Transformations are essential to achieve the goal of performing classification, because they allow us to make calculations by relating spaces with different structures. There are three main transformations, each with properties that characterize and differentiate it from the others: these properties will then be exploited in the algorithms based on the purpose and type of data.

- Additive-logratio transformation (alr): once a component (*ref*) is chosen as reference, the transformation is defined as:

$$alr : \mathcal{S}^G \to \mathcal{R}^{G-1} \text{ s.t. } y \to \left(\ln \frac{y_1}{y_{ref}}, \ldots, \ln \frac{y_k}{y_{ref}} \right) \tag{6.52}$$

  The Additive-logratio transformation is an isomorphism but *it is not* an isometry because the distances in the simplex are not the same of the ones in the transformed space.

- Centred-logratio transformation (clr):

$$clr : \mathcal{S}^G \to \mathcal{R}^G \text{ s.t. } y \to \left(\ln \frac{y_1}{g(y)}, \ldots, \ln \frac{y_G}{g(y)} \right) \tag{6.53}$$

where $g(y) = (\prod_{i=1}^{G} y_i)^{\frac{1}{G}}$, that is the geometric mean of the composition. The inverse of the clr transformation is the softmax transformation:

$$\text{softmax} : \mathcal{R}^G \to \mathcal{S}^G \text{ s.t. } y \to \left(\frac{e^{y_1}}{\sum_{i=1}^{G} e^{y_i}}, \ldots, \frac{e^{y_G}}{\sum_{i=1}^{G} e^{y_i}} \right) \tag{6.54}$$

The clr is both an isomorphism and an isometry, but the representation in $\mathcal{R}^G$ does not use an orthogonal basis and the observations belong to a plane in $\mathcal{R}^G$.
It is important to point out that it is possible to apply all the methods developed in the uncostrained $\mathcal{R}^G$ space to analyze the relationships between data in the simplex if clr is applied to the data (softmax must then be applied on the data in $\mathcal{R}^G$ to bring the data back to the simplex $\mathcal{S}^G$).

- Isometric-logratio transformation (ilr): this transformation is an isometry (as the name suggests) and an isomorphism. Moreover, it uses an orthogonal basis to represent the data of the simplex ($\mathcal{S}^G$) in the Euclidean space $\mathcal{R}^{G-1}$: for this reason, ilr is the *natural transformation to be used* if the goal is to apply algebraic methods for regressing classes (starting from compositional data) on predictors in the real space, because after the application of ilr both X and Y variables belong to space with the same norm and scalar product (this is fundamental to obtain consistent results).
  Finally, there is an important relation between clr and ilr:

$$\forall \alpha \in \mathcal{S}^G, ilr(\alpha) = V^\top clr(\alpha) = V^\top \ln(\alpha) \tag{6.55}$$

where V is a $G \times G-1$ matrix such that $V^\top V = I_{G-1}$ and $VV^\top = I_G - \frac{1}{G} 1_G 1_G^\top$.

6.4.4 2-class classification problem solved by logistic-like method in the Y-space

Given the two classes A and B, if we assume that

$$0 < P(class = j|x_i) < 1$$

we recognize that P belongs to the simplex $\mathcal{S}^2$.
If $P(class = B|x_i)$ is considered the reference, the alr-transformation generates

the following $\mathcal{R}$-representation

$$\text{alr: } \mathcal{S}^2 \to \mathcal{R} \text{ s.t. } P(class = A|x_i) \to \ln\left(\frac{P(class = A|x_i)}{1 - P(class = A|x_i)}\right) \tag{6.56}$$

i.e. $y_i \to \ln(\frac{y_i}{1-y_i})$, that is equivalent to apply the logit function to $y_i = P(class = A|x_i)$. Its inverse is

$$alr^{-1}(x) = \mathcal{C}(exp(x, 0)) = \mathcal{C}(e^x, 1) = \left(\frac{1}{1+e^{-x}}, \frac{e^{-x}}{1+e^{-x}}\right) \tag{6.57}$$

that corresponds to the logistic function. Taking this is mind, we can now apply the PLS "regression in the Y-space" approach to estimate the parameters of the regression model

$$g(y_i) = x_i^\top b + e_i$$

Recalling the iterative algorithm introduced in section 6.3 for the "regression in the Y-space", it is important to note that the norm in steps 4 and 5 is not the norm of a vector in $\mathcal{R}^N$, but the norm of a compositional-data vector in $\mathcal{S}_N^2$ (which is a set of N compositions each belonging to $\mathcal{S}^2$), and that the differences have to be calculated in the simplex because the vectors involved belong to the simplex and are not vectors in the Euclidean space. Moreover, alr is not an isometry and the solution of steps 4 and 5 must be obtained operating in the simplex.
Given $y \in \mathcal{S}_N^2$ mean centred[4] and X scaled and mean-centered, the algorithm can be re-written as

Algorithm 10: Logistic-like method for 2 classes in the y-space

1 $f_0 = y$;
2 $E_0 = X$;
3 **for** $i = 1, \ldots, A$ **do**
4 $\quad w_i = \frac{d}{d\tilde{w}_i} \| f_{i-1} \diamond alr^{-1}(E_{i-1}\tilde{w}_i) \|_S^2 \Big|_{\tilde{w}_i = 0_p}$;
5 $\quad t_i = E_{i-1} w_i$;
6 $\quad \alpha_i = \underset{\alpha}{\text{argmin}} \, \| f_{i-1} \diamond alr^{-1}(\alpha t_i) \|_S^2$;
7 $\quad f_i = alr^{-1}(alr(f_{i-1}) - \alpha_i t_i)$;
8 $\quad E_i = \hat{Q}_{t_i} E_{i-1}$
9 **end**

[4] The mean of a set of N compositional data vectors $a_i \in \mathcal{S}^D$ is defined as

$$\bar{m} = \text{mean}(a_1, \ldots, a_N) = C(\sqrt[N]{a_{11}, \ldots, a_{N1}}, \ldots, \sqrt[N]{a_{1D}, \ldots, a_{ND}})$$

in the case of $a_i \in \mathcal{S}^2$ we have $alr(\bar{y}) = \text{mean}(alr(y_1), \ldots, alr(y_N))$

where y and f_i are compositional-data vectors with elements in $\mathcal{S}^2$, $\diamond$ is the difference between vectors in $\mathcal{S}_N^2$, $\|x\|_S^2$ is the norm of the vector $x \in \mathcal{S}_N^2$, X and E_i are matrices in $\mathcal{R}^{N\times P}$, w_i and $\tilde{w}_i$ are vectors in $\mathcal{R}^P$ and $\alpha_i \in \mathcal{R}$.
Specifically, in $\mathcal{S}_N^2$ we have

$$\|a\|_S^2 = \sum_{i=1}^{N} \|a_i\|_S^2 = \frac{1}{2}\sum_{i=1}^{N}\left[\ln\left(\frac{a_{i1}}{a_{i2}}\right)\right]^2,$$

$$\text{with } a = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \in \mathcal{S}_N^2 \text{ and } a_i = (a_{i1}, a_{i2}) \in \mathcal{S}^2$$

whereas the difference $\diamond$ is defined as

$$\forall x, y \in \mathcal{S}_N^2,\ x \diamond y = \begin{bmatrix} x_1 \oplus ((-1)\odot y_1) \\ \vdots \\ x_N \oplus ((-1)\odot y_N) \end{bmatrix} = \begin{bmatrix} \mathcal{C}(\frac{x_{11}}{y_{11}}, \frac{x_{12}}{y_{12}}) \\ \vdots \\ \mathcal{C}(\frac{x_{N1}}{y_{N1}}, \frac{x_{N2}}{y_{N2}}) \end{bmatrix}$$

being $\mathcal{C}(\cdot)$ the closure operator, i.e. $\mathcal{C}(a,b) = (\frac{a}{a+b}, \frac{b}{a+b})$.

To obtain the explicit form of the algorithm, we calculate the derivative in step 4, solve step 5 and simplify the expression of the residuals in step 6. Firstly, we recognize that

$$\begin{aligned}
\|f \diamond alr^{-1}(E\tilde{w})\|_S^2 &= \sum_{i=1}^{N} \|f_i \oplus (-1)\odot alr^{-1}((E\tilde{w})_i)\|_S^2 = \\
&= \sum_{i=1}^{N} \|\mathcal{C}\left(\frac{f_{i1}}{alr^{-1}((E\tilde{w})_i)_1}, \frac{f_{i2}}{alr^{-1}((E\tilde{w})_i)_2}\right)\|_S^2 = \\
&= \frac{1}{2}\sum_{i=1}^{N}\left[\ln\left(\frac{f_{i1}}{f_{i2}}\frac{alr^{-1}((E\tilde{w})_i)_2}{alr^{-1}((E\tilde{w})_i)_1}\right)\right]^2 = \\
&= \frac{1}{2}\sum_{i=1}^{N}\left[\ln\left(\frac{f_{i1}}{f_{i2}}e^{-(E\tilde{w})_i}\right)\right]^2 = \frac{1}{2}\sum_{i=1}^{N}\left[\ln\left(\frac{f_{i1}}{f_{i2}}\right) - (E\tilde{w})_i\right]^2 = \\
&= \frac{1}{2}\sum_{i=1}^{N}[alr(f_i) - (E\tilde{w})_i]^2
\end{aligned}$$

Then, we have

$$w_i = \frac{d}{d\tilde{w}_i}\|f_{i-1} \diamond alr^{-1}(E_{i-1}\tilde{w}_i)\|_S^2\Big|_{\tilde{w}_i = 0_p} \propto E_{i-1}^{\top} alr(f_{i-1})$$

Step 6 can be re-written as

$$\alpha_i = \underset{\alpha}{\operatorname{argmin}} \|f \diamond alr^{-1}(\alpha t)\|_S^2 = \underset{\alpha}{\operatorname{argmin}} \sum_{i=1}^{N} [alr(f_i) - \alpha(E\tilde{w})_i]^2 = \underset{\alpha}{\operatorname{argmin}} f(\alpha)$$

and the minimum is

$$\alpha_i = \frac{\sum_{j=1}^{N} alr(f_{i-1,j}) t_j}{\sum_{i=1}^{N} t_i^2} = \frac{t_i^\top alr(f_{i-1})}{t_i^\top t_i}$$

because $f''(\alpha) = \sum_{i=1}^{N} t_i^2 > 0$.
Step 6 can be re-written as

$$f_i = alr^{-1}(alr(f_{i-1}) - \alpha_i t_i) = f_{i-1} \diamond alr^{-1}(\alpha_i t_i)$$

because alr and alr^{-1} are linear transformations, i.e.

$$alr(\alpha \odot a \oplus \beta \odot b) = \alpha alr(a) + \beta alr(b)$$

and

$$alr^{-1}(\alpha a + \beta b) = \alpha \odot alr^{-1}(a) \oplus \beta \odot alr^{-1}(b)$$

Finally, the algorithm to solve a 2-class classification problem using the logistic-like regression method in the Y-space is

Algorithm 11: Regression in the Y-space - alr

1 $f_0 = y$;
2 $E_0 = X$;
3 **for** $i = 1, \ldots, A$ **do**
4 $\quad w_i = \frac{E_{i-1}^\top alr(f_{i-1})}{[alr(f_{i-1})^\top E_{i-1} E_{i-1}^\top alr(f_{i-1})]^{\frac{1}{2}}}$;
5 $\quad t_i = E_{i-1} w_i$;
6 $\quad \alpha_i = \frac{t_i^\top alr(f_{i-1})}{t_i^\top t_i}$;
7 $\quad E_i = \hat{Q}_{t_i} E_{i-1}$;
8 $\quad f_i = f_{i-1} \diamond alr^{-1}(\alpha_i t_i)$
9 **end**

The weight vectors are a set of orthonormal vectors and the score vectors are orthogonal to each other. After A iterations the following decompositions are

obtained

$$X = TP^\top + E_A$$

and

$$f_A = y \diamond alr^{-1}\left(\sum_{i=1}^{A} \alpha_i t_i\right)$$

Using the inverse transformation of alr^{-1}, i.e. alr, we have

$$\ln\left(\frac{y_i}{1-y_i}\right) = alr(y_i) = \sum_{j=1}^{A} \alpha_j t_{ji} + alr(f_{Ai}) = x_i^\top W^* \alpha + alr(f_{Ai})$$

Then, the vector of the regression coefficients is $b = W^*\alpha$ and $e_i = alr(f_{Ai})$. The matrices T, P, W^* are defined according to standard PLS. The class of a new observation x_{new} is predicted estimating the probabilities

$$(P(class = A|x_{new}), P(class = B|x_{new})) = alr^{-1}((x_{new} - \bar{x})^\top L_{scaling} W^* \alpha) \oplus \bar{y}$$

or the value of the logit function

$$\ln\left(\frac{y_{new}}{1-y_{new}}\right) = (x_{new} - \bar{x})^\top L_{scaling} W^* \alpha + alr(\bar{y})$$

where $\bar{x}$ is the vector of the mean values of the columns of X, $L_{scaling} = \text{diag}(\frac{1}{f_i})$ is the scaling matrix and $\bar{y}$ is the mean of y for the training set calculated in the simplex.

6.4.5 2-class classification problem solved in the framework of compositional data

The case of using alr as Y-transformation has been discussed in the previous sections. Here we summarise the results obtained using clr and ilr as Y-transformation distinguishing the case of regression in the X- and Y-space.

Regression in the X-space

The case of ilr-transformation is similar to that of alr because both transformation map $\mathcal{S}^2$ into $\mathcal{R}$ and the PLS regression follows the same lines of PLS1. Specifically, the mean centred y-response is

$$\tilde{g}(y) = \frac{v_1 - v_2}{N_A + N_B} \ln\left(\frac{1-\epsilon}{\epsilon}\right) \begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix}$$

where v_1 and v_2 depend on the orthogonal basis chosen for $\mathcal{S}^2$, and the weight vectors are

$$w_i \propto E_{i-1}^{\top} \begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix} \text{ i.e. } w_{ij} \propto (e_{jA}^{-(i-1)} - e_{jB}^{-(i-1)})$$

being $e_{jk}^{-(i-1)}$ the mean of the residuals of the predictor j at the iteration $i-1$ calculated using the observations of class k.

The case of clr-transformation is only apparently different. Indeed, since clr maps $\mathcal{S}^2$ into a plane in $\mathcal{R}^2$, the PLS regression follows the lines of PLS with more than one response variable. Specifically, after mean centring of the Y-response we obtain

$$\tilde{g}(y) = \frac{1}{(N_A + N_B)} \ln\left(\frac{1-\epsilon}{\epsilon}\right) \begin{bmatrix} N_B 1_{N_A} & -N_B 1_{N_A} \\ -N_A 1_{N_B} & N_A 1_{N_B} \end{bmatrix}$$

When the PLS algorithm for the regression in the X-space is applied, the weight vector is

$$w_i \propto E_{i-1}^{\top} \begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix} \text{ i.e. } w_{ij} \propto (e_{jA}^{-(i-1)} - e_{jB}^{-(i-1)})$$

being $e_{jk}^{-(i-1)}$ the mean of the residuals of the predictor j at the iteration $i-1$ calculated using the observations of class k. Indeed, introducing the vector

$$z = E_{i-1}^{\top} \begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix}$$

the eigenvalue problem to be solved in step 4 becomes

$$E_{i-1}^{\top} \tilde{g}(y) \tilde{g}(y)^{\top} E_{i-1} w_i = s_i^2 w_i \implies z z^{\top} w_i = s_i^2 w_i$$

whose solution is z.

Then, we can conclude that the weight vectors generated by PLS regression in the X-space are independent of the Y-transfomation applied and are equal to those obtained by standard PLS-DA for a 2-class problem. Moreover, the weight vectors are independent of ϵ.
If the class membership is attributed considering the class with the greatest conditional probability, the prediction of a new observation is independent of ϵ and it is the same for all the three transformations, as proven in Appendix B.2.

Regression in the Y-space

The case of alr-transformation has been discussed in the section 6.4.4. Here, we consider the case of ilr-transformation for the sake of simplicity. The case of clr-transformation requires a more complex solution since it maps $\mathcal{S}^2$ into $\mathcal{R}^2$ (it follows the same lines of the case of G-class problem solved in the Y-space discussed in section 6.4.9). We recall the algorithm for PLS regression in the Y-space in the case of simplex as Y-space and Euclidean space as X-space.

Algorithm 12: Compositional data framework - Y-space (2-class problem)

1 $f_0 = y$;
2 $E_0 = X$;
3 **for** $i = 1, \ldots, A$ **do**
4 solve $\underset{\|W_i\|_F^2=1}{\text{argmax}} \nabla_{W_i} \|f_{i-1} \diamond g^{-1}(E_{i-1}\tilde{W}_i)\|_S^2$;
5 calculate $\alpha_i = \underset{\alpha}{\text{argmin}} \|f_{i-1} \diamond g^{-1}(\alpha E_{i-1}W_i)\|_S^2$;
6 $f_i = g^{-1}(g(f_{i-1}) - \alpha_i E_{i-1}W_i)$;
7 Deflation step $E_i \leftarrow E_{i-1}$;
8 **end**

where g^{-1} is a function that maps the Euclidean space in the simplex, y and f_i are compositional-data vectors with components in $\mathcal{S}^2$, $\diamond$ is the difference between vectors in $\mathcal{S}_N^2$ and $\|x\|_S^2$ is the norm of the vector $x \in \mathcal{S}_N^2$, X and E_i are matrices in $\mathcal{R}^{N\times P}$, W_i and $\tilde{W}_i$ are matrices in $\mathcal{R}^{P\times 2}$ or $\mathcal{R}^{P\times 1}$ depending on g, and $\alpha_i \in \mathcal{R}$.
The main advantage to use ilr instead of alr is that the application of ilr to the probabilities in $\mathcal{S}^2$ allows us to solve the problems in steps 4 and 5 using the properties of $\mathcal{R}^N$ because ilr is an isometry. Indeed, the minimization of the loss function in steps 4 and 5 can be performed in the Euclidean space being the norm the same of that calculated in the simplex. Then, we can use the Frobenius norm considering the ilr-transformation of f_i and $\alpha_i E_{i-1}W_i$. This is not possible in the case of alr because it is not an isometry and we have to apply norm and the operations of the simplex.

If the vector $t_i = E_{i-1}w_i$ is introduced, step 7 becomes $E_i = \hat{Q}_{t_i}E_{i-1}$.
Considering y mean centred and X mean centred and scaled, Algorithm 13 is obtained.

Step 4 generates a set of orthonormal weight vectors $w_i \propto E_{i-1}^{\top} ilr(f_{i-1})$ whereas

Algorithm 13: Ilr transformation - Y-space (2-class problem)

$f_0 = y$;
$E_0 = X$;
for $i = 1, \ldots, A$ **do**
  $w_i = \underset{w^\top w=1}{\text{argmax}} \nabla \|ilr(f_{i-1}) - E_{i-1}w\|_F^2$;
  $t_i = E_{i-1}w_i$;
  $\alpha_i = \underset{\alpha}{\text{argmin}} \|ilr(f_{i-1}) - \alpha t_i\|_F^2$;
  $f_i = f_{i-1} \diamond ilr^{-1}(\alpha_i t_i)$;
  $E_i = \hat{Q}_{t_i} E_{i-1}$
end

the coefficients are $\alpha_i = \frac{w_i^\top E_{i-1}^\top ilr(f_{i-1})}{w_i^\top E_{i-1}^\top E_{i-1} w_i}$ according to PLS1.
After A iterations, the following matrix decompositions are obtained

$$X = E_0 = TP^\top + E_A$$

and

$$y = ilr^{-1}\left(\sum_{i=1}^{A} \alpha_i t_i\right) \oplus f_A = ilr^{-1}\left(E_0 W^* \alpha\right) \oplus f_A = ilr^{-1}(Xb) \oplus f_A$$

where $b = W^*\alpha$. T, P, W^* are defined according to standard PLS.

A new observation x_{new} is predicted by

$$y_{new} = ilr^{-1}((x_{new} - \bar{x})^\top L_{scaling} b) \oplus \bar{y} \tag{6.58}$$

where $\bar{x}$ is the vector of the mean values of the columns of X, $L_{scaling} = diag(\frac{1}{\bar{f}_i})$ is the scaling matrix and $\bar{y}$ is the mean of y for the training set calculated in the simplex.

6.4.6 G-class classification problem

In the following sections the lines to solve the general G-class classification problem by PLS will be drawn. We will not discuss in details every aspects of the presented methods, but we shall limit to present the general framework. Further studies are required for a more detailed discussion.

6.4.7 G-class classification problem solved by logistic-like method

The logistic regression based on PLS can be applied to solve a G-class problem according to the standard approach for multinomial logistic regression. Specifically, given G classes of observations, $G-1$ logistic models are built considering the class k as reference to calculate the conditional probabilities

$$P(class = j|x_i) = P(class = k|x_i)e^{x_i^\top b_{j|k}+c_{j|k}}, \forall j \neq k$$

where $b_{j|k}$ and $c_{j|k}$ are the vector of regression coefficients and the constant term for the logistic-like PLS regression of class j against class k, respectively.
Since $\sum_{j=1}^{G} P(class = j|x_i) = 1$, we have

$$\begin{aligned} P(class = k|x_i) &= 1 - \sum_{\substack{j=1 \\ j\neq k}}^{G} P(class = j|x_i) = \\ &= 1 - P(class = k|x_i)\sum_{\substack{j=1 \\ j\neq k}}^{G} e^{x_i^\top b_{j|k}+c_{j|k}} \end{aligned} \tag{6.59}$$

then

$$P(class = k|x_i) = \frac{1}{1 + \sum_{\substack{j=1 \\ j\neq k}}^{G} e^{x_i^\top b_{j|k}+c_{j|k}}}$$

and

$$P(class = j|x_i) = \frac{e^{x_i^\top b_{j|k}+c_{j|k}}}{1 + \sum_{\substack{j=1 \\ j\neq k}}^{G} e^{x_i^\top b_{j|k}+c_{j|k}}}, \forall j \neq k$$

The estimation of the regression parameters can be performed both by regression in the X- or in the Y-space. It is worth noting that $b_{j|k}$ and $c_{j|k}$ depend only on the two classes j and k and are independent of the other $G-2$ classes (assumption of *independence of irrelevant alternatives*). Moreover, in the case of the regression in the X-space the set of $G-1$ predictive latent scores could not be a set of orthogonal vectors and the classification model depends in principle on the choice of ϵ.

6.4.8 G-class classification problem solved in the framework of compositional data in the X-space

The approach used in 6.4.1 and 6.4.5 to solve the 2-class classification problem using the compositional data theory can be extended to the general case of G-class classification problems. Given a training set of G classes, we assume that if observation i belongs to class j one has

$$P(class = j|x_i) = 1 - (G-1)\epsilon$$

and for the other classes ($\forall k \neq j$)

$$P(class = k|x_i) = \epsilon$$

being $0 < \epsilon < \frac{1}{G}$.
The training set can be represented by the response compositional-data matrix

$$\begin{bmatrix} y_1 & y_2 & \cdots & y_G \end{bmatrix} = \begin{bmatrix} [1-(G-1)\epsilon]1_{N_1} & \epsilon 1_{N_1} & \cdots & \epsilon 1_{N_1} \\ \epsilon 1_{N_2} & [1-(G-1)\epsilon]1_{N_2} & \cdots & \epsilon 1_{N_2} \\ \vdots & \vdots & \ddots & \vdots \\ \epsilon 1_{N_G} & \epsilon 1_{N_G} & \cdots & [1-(G-1)\epsilon]1_{N_G} \end{bmatrix}$$

where N_i is the number of observation for the class i, $\forall i = 1, \ldots, G$.
After clr-transformation of the matrix, we have

$$clr\left(\begin{bmatrix} y_1 & y_2 & \cdots & y_G \end{bmatrix}\right) = \frac{1}{G} \ln\left[\frac{1-(G-1)\epsilon}{\epsilon}\right] \begin{bmatrix} (G-1)1_{N_1} & -1_{N_1} & \cdots & -1_{N_1} \\ -1_{N_2} & (G-1)1_{N_2} & \cdots & -1_{N_2} \\ \vdots & \vdots & \ddots & \vdots \\ -1_{N_G} & -1_{N_G} & \cdots & (G-1)1_{N_G} \end{bmatrix}$$

Applying mean-centring we obtain

$$\begin{aligned} \tilde{Y}_{clr} &= \ln\left[\frac{1-(G-1)\epsilon}{\epsilon}\right] \begin{bmatrix} (\frac{1-N_1}{N})1_{N_1} & -\frac{N_2}{N}1_{N_1} & \cdots & -\frac{N_G}{N}1_{N_1} \\ -\frac{N_1}{N}1_{N_2} & (\frac{1-N_2}{N})1_{N_2} & \cdots & -\frac{N_G}{N}1_{N_2} \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{N_1}{N}1_{N_G} & -\frac{N_2}{N}1_{N_G} & \cdots & (\frac{1-N_G}{N})1_{N_G} \end{bmatrix} \\ &= \ln\left[\frac{1-(G-1)\epsilon}{\epsilon}\right] Y_{DActr} \end{aligned}$$

where $N = \sum_{i=1}^{G} N_i$ is the total number of observations and Y_{DActr} is the Y-indicator matrix used in PLS-DA after mean-centering. Since

$$E_{i-1}^{\top}\tilde{Y}_{clr}\tilde{Y}_{clr}^{\top}E_{i-1} \propto E_{i-1}^{\top}Y_{DActr}Y_{DActr}^{\top}E_{i-1}$$

the weight vectors obtained applying the PLS "regression in the X-space" are the same generated by PLS-DA if only mean centring is applied as data pre-treatment. Then, the weight matrix $W = [w_1, \ldots, w_A]$ can be used to calculate the matrix of the regression coefficients

$$B = W(W^{\top}X^{\top}XW)^{-1}W^{\top}X^{\top}\tilde{Y}_{clr} \tag{6.60}$$

and the class of a new observation x_{new} is predicted by

$$\text{class } i = y_{i,new} = \max(y_{1,new}, \ldots, y_{G,new}) \tag{6.61}$$

being

$$\begin{bmatrix} y_{1,new} & y_{2,new} & \cdots & y_{G,new} \end{bmatrix} = \text{softmax}((x_{new} - \bar{x})^{\top}L_{scaling}B + \bar{y}_{clr})$$

where $\bar{y}_{clr,i} = \frac{1}{G}(\frac{N_i}{N}(G-1))\ln\left[\frac{1-(G-1)\epsilon}{\epsilon}\right]$, $\bar{x}$ is the vector of the mean values of the columns of X and $L_{scaling} = \text{diag}(\frac{1}{f_i})$ is the scaling matrix.
The weight matrix is independent of ϵ, the predictive latent scores are a set of orthogonal vectors and the presence of a particular class influences the probabilities of all the other classes (no assumption of *independence of irrelevant alternatives*).
In general, because

$$(x_{new} - \bar{x})^{\top}L_{scaling}B + \bar{y}_{clr} = at$$

where $a = \ln\left(\frac{1-(G-1)\epsilon}{\epsilon}\right) > 0$ and t a row vector in $\mathcal{R}^G$, if

$$e^{at_i} = max(e^{at_1}, \ldots, e^{at_G}) \implies e^{\tilde{a}t_i} = max(e^{\tilde{a}t_1}, \ldots, e^{\tilde{a}t_G}), \forall \tilde{a} > 0$$

the prediction is independent on ϵ.
In the case of ilr transformation, each vector of $\mathcal{S}^G$ is transformed in a vector of $\mathcal{S}^{G-1}$. The PLS regression in the X-space follows the same lines discussed in the case of clr, with the difference of using a different Y-matrix.

6.4.9 G-class classification problem solved in the framework of compositional data in the Y-space

In the case of a 2-class classification problem, the ilr transformation allows us to solve steps 4 and 5 using PLS1. In the general case of a G-class classification problem, those steps are solved using the same method discussed in section 6.1.2 for PLS. Both ilr and clr can be applied as isomorphism because they are isometries and, then, they allow the minimization of the norm in the simplex considering the Frobenius norm in the Euclidean space. Given a training set of N observations belonging to G classes, we represent the training set using the compositional-data vector

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \in \mathcal{S}_N^G$$

where the composition of observation i of class j is

$$P(class = j|x_i) = 1 - (G-1)\epsilon$$

and

$$P(class = k|x_i) = \epsilon, \forall k \neq j$$

with $0 < \epsilon < \frac{1}{G}$.
For mean centered y and mean centered and scaled X, if ilr is applied, the following algorithm is obtained.

Algorithm 14: Regression in the Y-space - ilr

1 $f_0 = y$;
2 $E_0 = X$;
3 **for** $i = 1, \ldots, A$ **do**
4 $\quad W_i = \underset{\|W\|_F^2=1}{\operatorname{argmax}} \nabla \|ilr(f_{i-1}) - E_{i-1}W\|_F^2$;
5 $\quad \alpha_i = \underset{\alpha}{\operatorname{argmin}} \|ilr(f_{i-1}) - \alpha E_{i-1}W_i\|_F^2$;
6 $\quad f_i = f_{i-1} \diamond ilr^{-1}(\alpha_i E_{i-1}W_i)$;
7 $\quad$ deflation step $E_i \leftarrow E_{i-1}$
8 **end**

Steps 4 and 5 generate the weight vectors

$$w_i \text{ s.t. } E_{i-1}^{\top} ilr(f_{i-1}) ilr(f_{i-1})^{\top} E_{i-1} w_i = s_i^2 w_i$$
$$c_i = \frac{1}{s_i} ilr(f_{i-1})^{\top} E_{i-1} w_i$$

being $W_i = w_i c_i^{\top}$ and the coefficient

$$\alpha_i = \frac{s_i}{w_i^{\top} E_{i-1}^{\top} E_{i-1} w_i}$$

Step 7 becomes

$$E_i = \hat{Q}_{t_i} E_{i-1}$$

where $t_i = E_{i-1} w_i$.
After A iterations, the following matrix decompositions are obtained

$$X = TP^{\top} + E_A \tag{6.62}$$

and

$$y = ilr^{-1}\left(\sum_{i=1}^{A} \alpha_i t_i c_i^{\top}\right) \oplus f_A = ilr^{-1}(XW^{*}\bar{C}^{\top}) \oplus f_A = ilr^{-1}(XB) \oplus f_A \tag{6.63}$$

where $B = W^{*}\bar{C}^{\top}$ being $\bar{C} = [\alpha_1 c_1, \ldots, \alpha_A c_A]$.
A new observation x_{new} is predicted by

$$y_{new} = ilr^{-1}((x_{new} - \bar{x})^{\top} L_{scaling} B) \oplus \bar{y} \tag{6.64}$$

where $\bar{x}$ is the vector of the mean values of the columns of X, $L_{scaling} = diag(\frac{1}{f_i})$ is the scaling matrix and $\bar{y}$ is the mean of y for the training set calculated in the simplex.
The same approach can be applied considering clr, substituting clr for ilr.

Chapter 7

Applications

In this Chapter the proposed algorithms are tested against both simulated and real datasets, giving the practical evidence of their properties.

7.1 Low dimensional scenario ($p < n$)

First of all, a traditional logistic regression problem is considered to evaluate how the proposed methods behave in this context, i.e. we use datasets in which the number of predictors is smaller than that of observations and the collinearity between predictors is mild or moderate.
The first dataset is a real dataset used for banknote authentication [11]; it contains 1372 observations, 4 continuous predictors and a binary response variable which indicates whether a banknote is genuine (0) or forged (1).
The values of the X-variables are extracted from images that are taken from genuine and forged banknote-like specimens: an industrial camera for print inspection is used for digitalization of the images and gray-scale pictures with a resolution of about 660 dots per inch (dpi) are obtained. Finally, the features are extracted from images through a Wavelet Transform tool.
The predictors are the variance, skewness and curtosis of the Wavelet Transformed image and the entropy of the image.
Using this dataset, we want to compare logistic regression with the methods we presented in the previous Chapter.

First of all, we can evaluate how the Cohen's Kappa changes, both in calculation and cross-validation (10 folds are used), as the number of latent variables increases. Figure 7.1 shows the results using the autoscaled X matrix, ilr transformation in the X-space and $\epsilon = 10^{-6}$ (the default value where not specified in the following). However, the predictions (and therefore the κ values for each component) are the same for all the three types of transformations (alr, clr, ilr)

and also for the regression in the X and Y space. This confirms what we mentioned in Section 6.4.5 and proved in B.2 for PLS regression in the X-space.

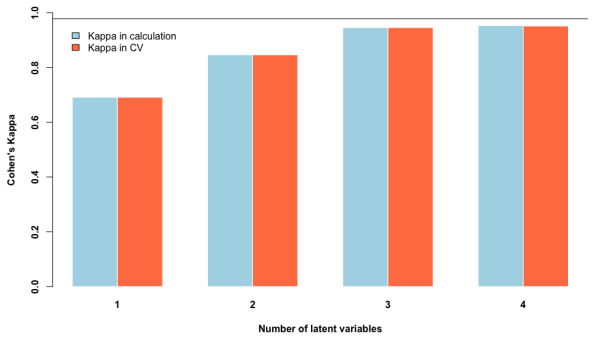


Figure 7.1: Cohen's κ as a function of the number of latent variables

The horizontal black line indicates the κ value obtained from logistic regression in a 10-fold cross-validation, which is just above the κ of the new methods with three and four components. This highlights how the proposed techniques provide fairly accurate predictions even in a traditional scenario (i.e. when $n > p$). The κ increases both in calculation (as it is expected) and in cross-validation, and their difference is subtle. The predictions in calculation and cross-validation are almost the same, as illustrated in Table 7.2, which represent the confusion matrices obtained using four latent variables in calculation and cross-validation. Only one more observation is misclassified in cross-validation.

		classCalculation	
		0	1
class	0	730	32
	1	0	610

		classCV	
		0	1
class	0	729	33
	1	0	610

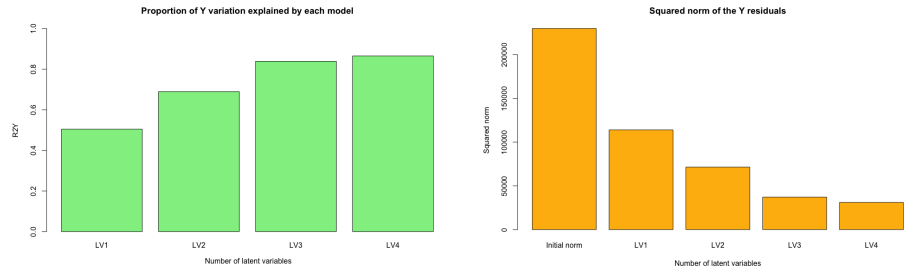
Figure 7.2: Confusion matrices in calculation (left) and cross-validation (right) with 4 latent variables

Furthermore, as we expected also the weight vectors produced by the models are equal using all the three transformations and they are a set of orthonormal vectors: an example of weight matrix W is given in Figure 7.3, where the values of $w_i, i = 1, \ldots, 4$ of a model with four latent variables are shown.

Figure 7.4 shows on the left the cumulative fraction of the y variation explained by each model ($R2Y$) which is equal performing the regression in the X and Y space, while through the regression in the Y-space it is possible to appreciate the decreasing of the squared norm of the y-residuals as the number of latent variable grows (right).

	LV1	LV2	LV3	LV4
variance	-0.8380989	-0.2866997	-0.2723704	-0.37577642
skewness	-0.5141696	0.1490185	0.5859094	0.60838577
curtosis	0.1802398	-0.9323777	0.2994326	0.09233445
entropy	-0.0270836	-0.1620761	-0.7020432	0.69291644

Figure 7.3: Weight matrix of the model with four latent variables



(a) Cumulative fraction of y variation explained by each model

(b) Squared norm of the y residuals for each number of latent variables

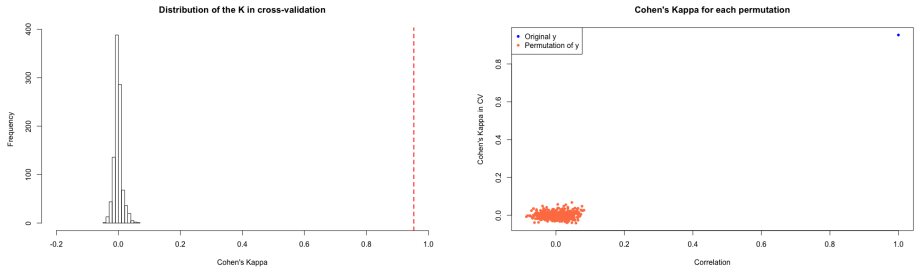
Figure 7.4: Cumulative fraction of the y variation (left) and squared norm of the Y residuals (right) as a function of the latent variables in the model

The first latent variable explains half of the variation of y, while the others add progressively smaller increments in the $R2Y$, as it should be when dealing with dimensionality reduction techniques like PLS.
Finally, a permutation test is applied: specifically, we considered the model with 4 components and shuffled the response variable 1000 times: for each permutation, a 10-fold cross-validation is performed using the autoscaled X matrix considering that permutation of y and the k value is calculated. Note that also in this case the choice of the transformation to be used or the type of method (i.e. whether to regress in the space of X or Y) is indifferent as they all lead to the same prediction.
The right panel of Figure 7.5 shows the Cohen's Kappa in cross-validation for each permutation as a function of the Pearson's correlation between the original y and the related permutation of the response variable[1].

As it can be seen in the left panel, there is no permutation for which the κ in cross-validation is higher than the original one, so the p-value is 0.001, which

[1]Since y and its permutations are binary vectors, also the Tanimoto coefficient can be used to calculate the correlation. Given two vectors $a, b \in \mathcal{R}^N$, the Tanimoto coefficient ranges from 0 to +1 (+1 is the highest similarity) and it is computed as:

$$T(a,b) = \frac{a \cdot b}{\|a\|^2 + \|b\|^2 - a \cdot b} \tag{7.1}$$



(a) Distribution of the Kappa values

(b) Cohen's Kappa in cross-validation as a function of the Pearson's correlation

Figure 7.5: Permutation test (1000 permutations)

means that the parameter κ is statistically significant.

Let's now consider the case of a simulated dataset with 500 observations and 10 predictors: the values of each predictor are drawn from a Gaussian distribution $\mathcal{N}(\mu = 0, \sigma = 1)$ and a regression coefficient is assigned to each of them (β_0 is the intercept and β the vector of coefficients of the variables). Then, the probability of the response being 1 (p) of each observation x_i is modelled as

$$p_i = \frac{1}{1 + e^{-(\beta_0 + \beta^\top x_i)}} \tag{7.2}$$

and the response variable y is defined as vector in which each value y_i is drawn from a Bernoulli distribution having an expected value equal to $p_i, i = 1, \ldots, 500$. First, we investigate the pairwise correlation of the predictors (Figure 7.6), which always has a low absolute value: thus, no particular correlation is detected among variables.

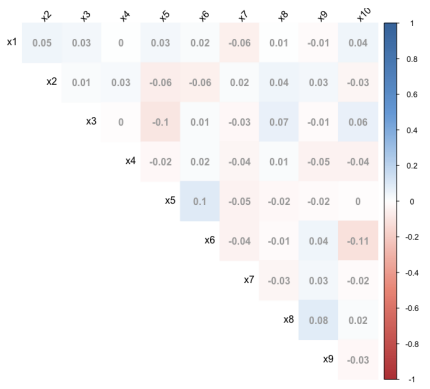


Figure 7.6: Pairwise correlation of the X-variables

A standard logistic regression is then applied. The true coefficients in the underlying model and the estimated ones are reported in Figure 7.7.

	Estimated coefficients	Coefficients
Intercept	-2.296	-2.0
x1	0.806	1.0
x2	3.132	3.0
x3	-1.005	-1.0
x4	1.245	1.0
x5	-2.402	-2.0
x6	0.953	1.0
x7	-1.968	-1.5
x8	-0.390	-0.5
x9	2.597	2.0
x10	-0.926	-1.0

Figure 7.7: Coefficients estimated by logistic regression and true ones for each X-variable

The logistic regression presents a κ value in a 10-fold cross-validation equal to 0.774; actually, the proposed techniques perform slightly better, since the κ in cross-validation goes up to 0.781 with three latent variables. The results in calculation and cross-validation are shown in Figure 7.8. This confirms the ability of the methods to properly adapt to traditional problems as well, i.e. in the "small p, large n" scenario.

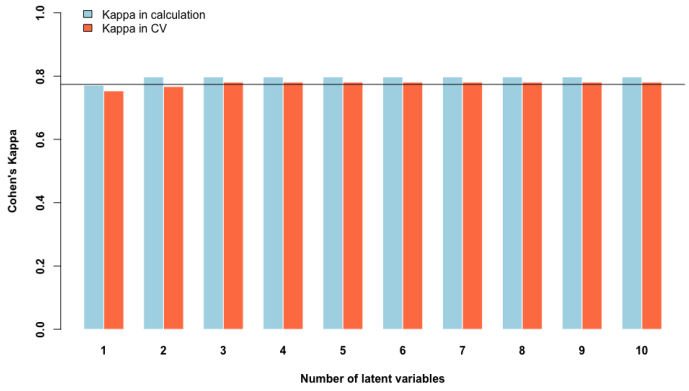


Figure 7.8: Cohen's κ in calculation and cross-validation for each number of latent variables

Also in this case, the weight vectors are the same for all the three transformations and both performing the regression in the Y or X-space. The same holds for the predicted class.
An example of the comparison of the regression coefficients is given in Figure 7.9, where the first column represents the coefficients of the regression in the Y-space of a model with three latent variables using clr transformation. The

second columns presents the coefficients estimated by the logistic regression and the last one the true coefficients. Recall that the intercept is not present in the proposed model.

	Y-space (clr)	Logistic Regression Coefficients	
x1	1.092	0.806	1.0
x2	3.854	3.132	3.0
x3	-1.069	-1.005	-1.0
x4	1.090	1.245	1.0
x5	-2.653	-2.402	-2.0
x6	1.148	0.953	1.0
x7	-2.457	-1.968	-1.5
x8	-0.587	-0.390	-0.5
x9	2.898	2.597	2.0
x10	-0.894	-0.926	-1.0

Figure 7.9: Comparison of regression coefficients

Actually, it is interesting to evaluate how the regression coefficients vary as a function of the number of latent variables in the model. To do that, we calculate the regression coefficient vector for each number of latent variables from one to ten, and compare them to the original β and to the vector of coefficients estimated by the logistic regression. To facilitate the visualization of the data, a PCA has been applied in such a way that each model can be represented by a point in a three-dimensional space (the first three principal components are considered).

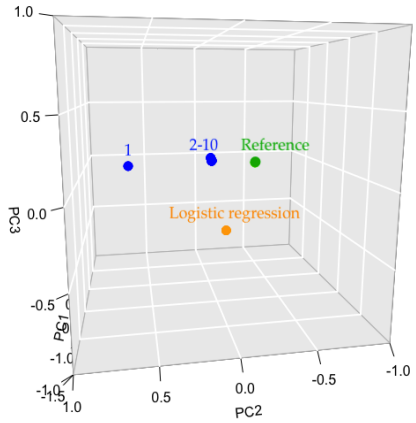
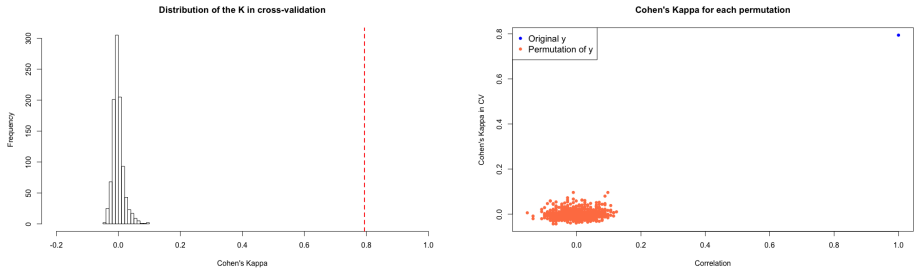


Figure 7.10: Vector of regression coefficients in the space extracted by the first three PCA components for all the considered models

The vectors of regression coefficients of the model with one latent variable is far from the others, indicating that it presents different values for the coefficients, which stabilize from the model with two components. Also the vector estimated by logistic regression in not really close to the reference one, that is the vector of original coefficients β.

Again, we apply a permutation test on the data. As before, we consider 1000 permutations of y and a model with three components, evaluating how the Cohen's Kappa in a 10-fold cross-validation changes when the response variable is shuffled.
Figure 7.11 illustrates the results of the permutation test, that also in this case gives a p-value of 0.001.



(a) Distribution of the Kappa values

(b) Cohen's Kappa in cross-validation as a function of the Pearson's correlation

Figure 7.11: Permutation test (1000 permutations)

7.2 High-dimensional scenario ($p > n$)

In this Section the new methods are tested in the scenario for which they are designed, i.e. when the number of predictors are bigger than the number of observations, causing a multicollinearity phenomenon. Moreover, we want to compare these classification procedures with the well-known PLS-DA[2]. Let's start by testing those methods against the simulated dataset used in Section 4.2. First of all it is possible to check that the data are affected by collinearity, as shown in Figure 7.12: the correlation has values close to +1 or -1 for a large part of the X-variable pairs, indicating a (positive or negative) high correlation.

The proposed techniques present the same weight vectors of PLS-DA (as mentioned in 6.4.5). An example is given in Figure 7.13, which shows the first elements of the weights vectors of a model trained on the autoscaled X matrix using a regression in the X-space with 10 latent variables and clr as transformation[3] and those of a PLS-DA model, fitted on the same X matrix and with the same number of components.

[2]PLS is used for the class separation and Linear Discriminant Analysis for the classification

[3]It has already been proved that the weight vectors are the same for all the three transformations, so it clr can be replaced also by ilr or alr

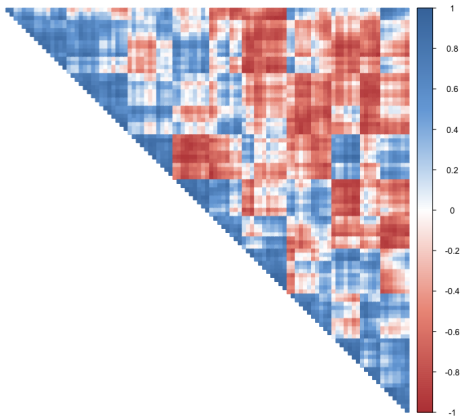


Figure 7.12: Pairwise correlation between X-variables

```
> PLS
             [,1]        [,2]        [,3]       [,4]       [,5]        [,6]        [,7]        [,8]       [,9]       [,10]
[1,] -0.16988538 -0.03698163 -0.03563586  0.12909418  0.13306385 -0.09513226  0.003300831  0.01346862  0.11203470  0.23046436
[2,]  0.13655739  0.02613562 -0.04286073 -0.03296190  0.12760657  0.06225275  0.010632504  0.11162671  0.05126729 -0.09093332
[3,]  0.15206318 -0.01970482 -0.01240528 -0.17025634 -0.04616791  0.10297742  0.138899740  0.13189198  0.03587686 -0.03444184
[4,]  0.10240115  0.16952698  0.01793532  0.08840550 -0.14769830  0.01022004  0.049484683  0.04606736  0.02820108 -0.13385485
[5,] -0.17633098 -0.07836210 -0.06697701 -0.01836796 -0.03599360 -0.07421001 -0.101051450 -0.08398505 -0.07039899 -0.07356657
[6,]  0.03074737  0.03260319  0.14224405  0.09095274 -0.07109562 -0.13933813 -0.089524975 -0.13814985  0.01573118  0.04109677
> PLS-DA
             [,1]        [,2]        [,3]       [,4]       [,5]        [,6]        [,7]        [,8]       [,9]       [,10]
[1,] -0.16988538 -0.03698163 -0.03563586  0.12909418  0.13306385 -0.09513226  0.003300831  0.01346862  0.11203470  0.23046436
[2,]  0.13655739  0.02613562 -0.04286073 -0.03296190  0.12760657  0.06225275  0.010632504  0.11162671  0.05126729 -0.09093332
[3,]  0.15206318 -0.01970482 -0.01240528 -0.17025634 -0.04616791  0.10297742  0.138899740  0.13189198  0.03587686 -0.03444184
[4,]  0.10240115  0.16952698  0.01793532  0.08840550 -0.14769830  0.01022004  0.049484683  0.04606736  0.02820108 -0.13385485
[5,] -0.17633098 -0.07836210 -0.06697701 -0.01836796 -0.03599360 -0.07421001 -0.101051450 -0.08398505 -0.07039899 -0.07356657
[6,]  0.03074737  0.03260319  0.14224405  0.09095274 -0.07109562 -0.13933813 -0.089524975 -0.13814985  0.01573118  0.04109677
```

Figure 7.13: Weights of the new approach and PLS-DA

The same applies also in cross-validation: indeed, the methods we developed and PLS-DA share the same predicted classes, and therefore also the value of κ for each number of latent variables in the model.
Figure 7.14 summarizes the results in calculation: the Cohen's Kappa grows significantly up to the model with five latent variables and then tends to stabilize.

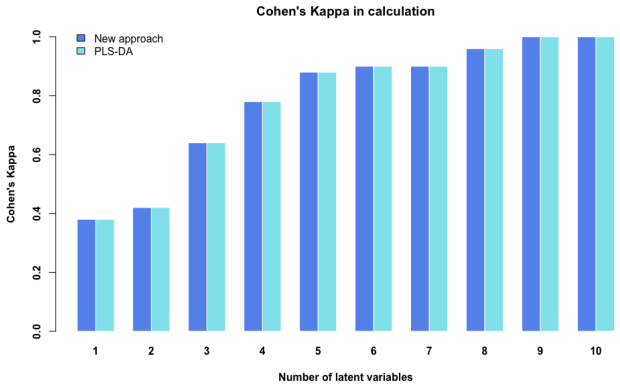
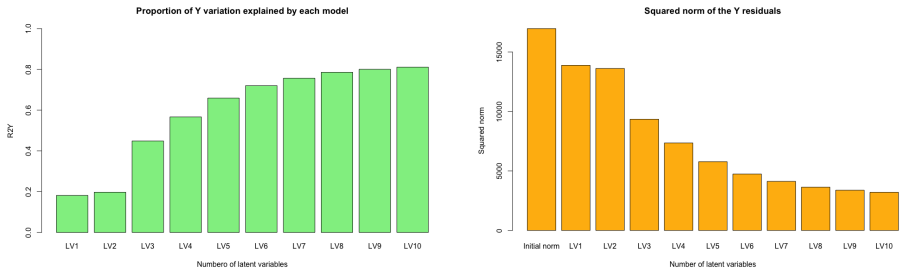


Figure 7.14: Cohen's Kappa in calculation for the new approach and PLS-DA

The values of κ in calculation reflects the proportion of variation of y explained by each model (Figure 7.15, left). In the right panel the squared norm of the residuals for each number of latent variables is illustrated.



(a) Cumulative fraction of y variation explained by each model

(b) Squared norm of the y residuals for each number of latent variables

Figure 7.15: Cumulative fraction of the y variation (left) and squared norm of the Y residuals (right) as a function of the latent variables in the models

In cross-validation the values of κ are definitely lower than in calculation but still the two approaches share the performance also when predicting new observations (Figure 7.16). In contrast to the calculation case, in cross-validation the model that performs best (i.e. balances underfitting and overfitting) seems to be the one with two latent variables.

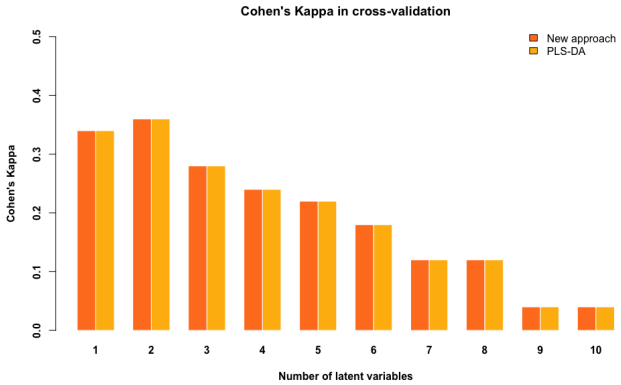
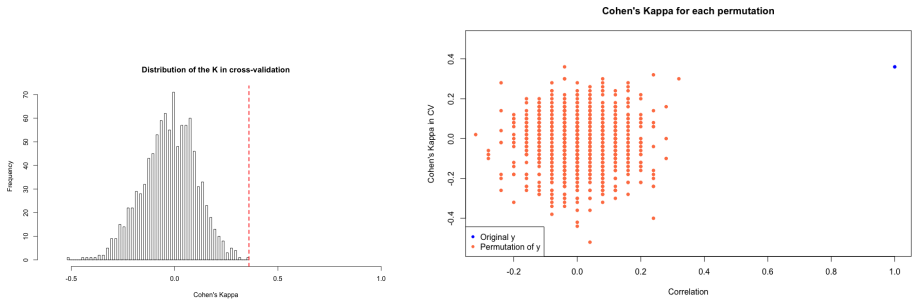


Figure 7.16: Cohen's Kappa in cross-validation for the new approach and PLS-DA

Thus, a permutation test is applied considering a model with two components. Here, the κ value in cross-validation of the original model is closer to the ones of the permutations with respect to the previous cases, but still higher than all the others, giving a p-value of 0.001 and therefore providing statistical significance

to the model. As can be observed in Figure 7.17, the Cohen's Kappa of the original model lies at the end of the right tail of the distribution.



(a) Distribution of the Kappa values (b) Cohen's Kappa in cross-validation as a function of the Pearson's correlation

Figure 7.17: Permutation test (1000 permutations)

Another interesting parameter to investigate is ϵ, and in particular whether it influences the predictions. As it is noted in Chapter 6, the weight vectors calculated in the models should be independent on the value of $\epsilon \in (0, \frac{1}{2})$ for the regression in the X-space, and they should be the same for all the three transformations.
This is confirmed by the evidence on the data: computing the Cohen's Kappa in calculation using the autoscaled X matrix, it is possible to check that any value of ϵ in that interval does not affect the predictions (and also the weight vectors) using all the transformation and for any number of latent variables.

	LV1	LV2	LV3	LV4	LV5	LV6	LV7	LV8	LV9	LV10
1e-05	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.01	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.06	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.11	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.16	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.21	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.26	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.31	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.36	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.41	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1
0.49999	0.38	0.42	0.64	0.78	0.88	0.9	0.9	0.96	1	1

Figure 7.18: Cohen's Kappa depending on ϵ and the number of components

In Table 7.18, the type of transformation is kept fixed (clr with a regression in the X-space is used) and the Cohen's Kappa depending on the number of latent variables and ϵ is reported: given a number of components, the Cohen's Kappa is independent on ϵ.
In Table 7.19 the number of latent variables is kept fixed (a model with two components is used) and the Cohen's Kappa depending on ϵ and the type of

transformation is shown: as can be seen, the Cohen's Kappa is the same for any type of transformation and value of ϵ.

	X-space, alr	X-space, clr	X-space, ilr	Y-space, alr	Y-space, clr	Y-space, ilr
1e-05	0.42	0.42	0.42	0.42	0.42	0.42
0.01	0.42	0.42	0.42	0.42	0.42	0.42
0.06	0.42	0.42	0.42	0.42	0.42	0.42
0.11	0.42	0.42	0.42	0.42	0.42	0.42
0.16	0.42	0.42	0.42	0.42	0.42	0.42
0.21	0.42	0.42	0.42	0.42	0.42	0.42
0.26	0.42	0.42	0.42	0.42	0.42	0.42
0.31	0.42	0.42	0.42	0.42	0.42	0.42
0.36	0.42	0.42	0.42	0.42	0.42	0.42
0.41	0.42	0.42	0.42	0.42	0.42	0.42
0.49999	0.42	0.42	0.42	0.42	0.42	0.42

Figure 7.19: Cohen's Kappa depending on ϵ and the type of transformation

The same holds also in cross-validation. On the contrary, having a value of $\epsilon > \frac{1}{2}$ means inverting the probability of the two classes, assigning more probability to the wrong one. This leads to a specular prediction[4] and therefore the value of the Cohen's Kappa has the same absolute value but with a negative sign.

From the running time point of view, the regression in the X-space and PLS-DA present very similar times of execution, while the regression in the Y-space takes more time: 100 10-fold-cross-validations were run 50 times, collecting the 50 averages of execution times (summarized in Figure 7.20).
The regressions in the X- and Y-space were performed using the autoscaled X matrix, 10 latent variables and ilr as transformation. However, the type of transformation negligibly affects execution times. PLS-DA were run using the autoscaled X matrix, 10 folds and 10 latent variables, as for the other methods.

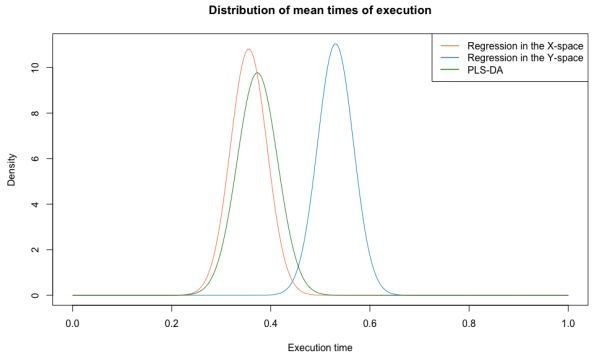


Figure 7.20: Distribution of mean execution times of a 10-fold cross-validation

[4] A sample that is assigned to class 1 with $\epsilon \in (0, \frac{1}{2})$ is predicted as 0 with $\epsilon \in (\frac{1}{2}, 1)$ and viceversa

The regression in the Y-space takes more time of execution than that of the X-space (+ 49%) because at each iteration the values of the response variables must be mapped (more than once) from the simplex to the Euclidean space and subsequently re-transformed through the inverse function. Furthermore, operations like sum, difference, norm must be performed as they are defined in the simplex.
On the contrary, when regressing in the X-space the y variable is transformed at the beginning from the simplex to the Euclidean space and then used as input for PLS; only when the execution of PLS is completed, the values are mapped back into the original space.
The classification procedures are tested on a MacBook Pro 2016 with 2.9 GHz Intel Core i5 dual-core CPU, 8 GB 2133 MHz LPDDR3 RAM and Intel Iris Graphics 550 (1536 MB) graphic card.

The new methods discussed in this thesis are then tested against the real dataset presented in Section 2.8. However, the response variable of interest now corresponds to the state of the eye of each sample, which can takes two possible values, i.e. "opened" or "closed"; thus, the binary y variable (called "EYE") is coded with 0 and 1 depending on the state of the eye.
The two classes are balanced in both training and test sets: indeed, the training contains 19 samples belonging to class 0 and 19 instances for class 1, while the test set has 10 observations for class 0 and 11 samples for class 1.

First, it is worth noting that having the same weight vectors in our approach and in PLS-DA implies that also the score vectors are the same (and this holds for both PLS regression in the X and Y space and for all three transformations). An example of the score vectors is illustrated in Figure 7.21, which shows the ability of the first two latent variables to separate the groups.

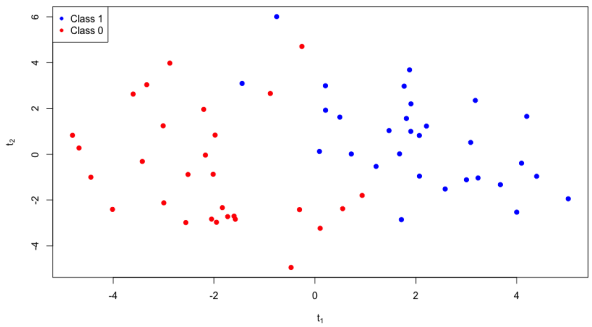
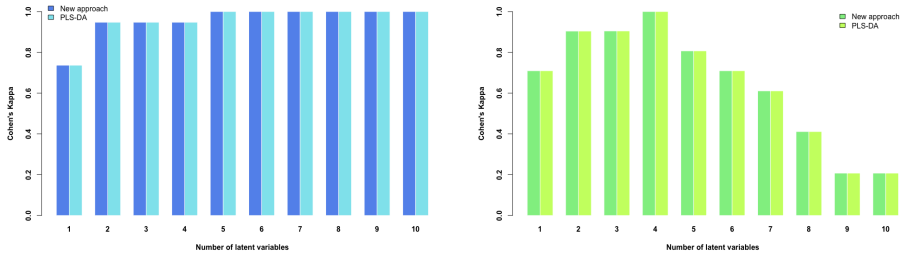


Figure 7.21: Separation of the groups in the training set

We want to compare the new techniques with PLS-DA in calculation and prediction. The matrix of predictors X is autoscaled and the three transformations produce the same results. As before, they show the same prediction ability in classifying the samples (Figure 7.22).



(a) Cohen's Kappa in calculation for the new approach and PLS-DA

(b) Cohen's Kappa in prediction for the new approach and PLS-DA

Figure 7.22: New approach and PLS-DA in calculation (left) and prediction (right)

The right panel shows the performance of both procedures when a model with a given number of latent variables is fitted on the autoscaled training set and then used to predict new samples (i.e. the test set). The model with four latent variables is the one that performs best, while simpler models or more complex models result in showing respectively high bias and high variance, which decrease their performance in prediction.

As for model interpretation, if the regression in the X-space is applied, one can use post-transformation, VIP, SR and other parameters because the method corresponds to a standard PLS with a suitably transformed response variable. On the other hand, for the regression in the Y-space, VIP and standard plots can be used, whereas the post-transformation has not yet been developed, so SR and related methods do not yet exist.

Thus, to give a practical example of model interpretation in the classification scenario, a PLS model with 4 latent variables is calculated using the autoscaled matrix X of predictors and the transformed response variable (alr transformation is used).
The correlation loading plot obtained from the model is reported in Figure 7.23, where $pcor[Tp]$ indicates the Pearson's correlation between the predictive latent variable and each predictor, as well as the response. The same holds for $pcor[To1]$, but the first orthogonal latent variable is considered instead of the predictive one.

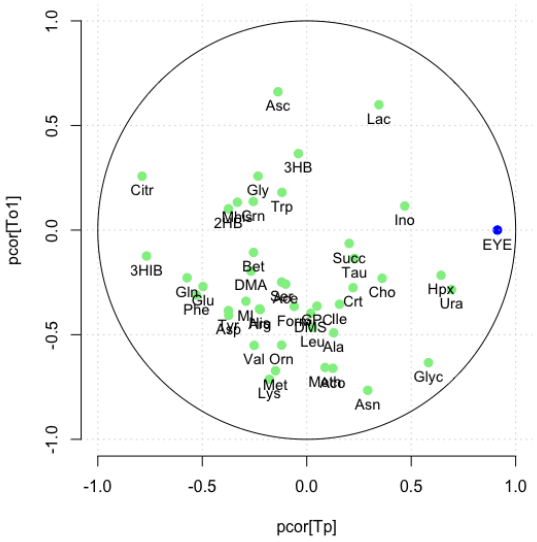


Figure 7.23: Correlation loading plot of a PLS model with 4 latent variables

The predictors whose points are close to that of the response (or its image by origin reflection) and therefore are (positively or negatively) highly correlated with the response seem to correspond to metabolites like Hypoxanthine, Uracil, Inosine, 3-Hydroxyisobutyrate, Citrate, Glutamine, Glutamate.
This is confirmed by the VIP parameter, whose aim is to identify the most important X-variables in the model (Figure 7.24): the variables that show the highest VIP correspond exactly to the aforementioned predictors.

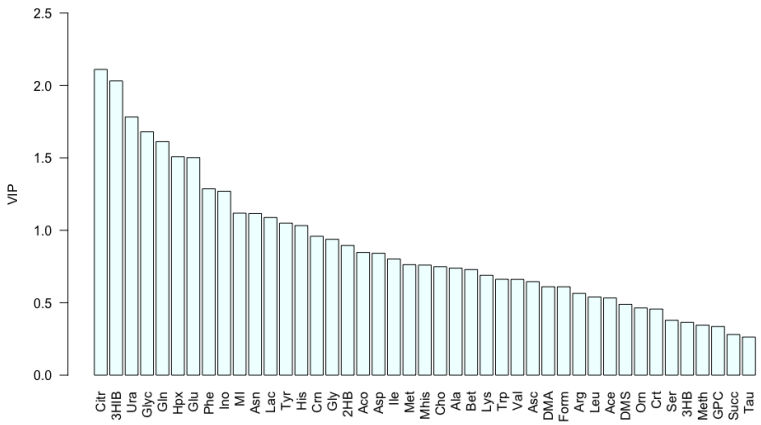


Figure 7.24: VIP of a PLS model with 4 latent variables

Chapter 8

Conclusions

In this work new procedures to perform classification exclusively through PLS are presented. Indeed, at the state-of-the-art most of PLS-based classification techniques require an additional classifier to effectively assign the classes to the observations. Therefore, PLS is used as a *discriminatory* tool to separate the samples rather than a classifier.
The development of those methods starts from looking at PLS from a different point of view, that is an iterative procedure that minimizes the distance between response and modelled response (that in the Euclidean space corresponds to the least squares problem) through the steepest descent method. Moreover, compositional data theory is used to consider the response variables as probabilities (i.e. compositions that sum up to 1) providing a theoretical foundation to the procedure; the proposed Y-transformations allow to perform calculations that link spaces with different structures.

Two main categories of models are presented: the first one involves a regression in the X-space, since the response variable is mapped through proper transformations in the same space of the X-variables, i.e. the Euclidean space, and minimization is performed in that space.
On the other side, the regression in the Y-space involves taking into account at each iteration the specific structure of the response variable space and using suitable operations to perform the calculations in that space since minimization is performed in the Y-space, i.e. the simplex. Thus, this opens up the possibility of choosing the space with the simplest geometry to perform the minimization depending on the type of response and predictors.

The theory of these algorithms for both 2-class and G-class ($G > 2$) problems is presented, even if the focus is on the binary case.
In that setting, we can assert that the proposed classification procedures present

the same performance as PLS-DA even not requiring further classifiers (our approach is purely PLS-based).
Moreover, the properties mentioned in Chapter 6 are confirmed through tests on simulated and real data. In fact, the weight vectors and the predictions are identical for both the regression in the X and Y space and for all three transformations; moreover, the ϵ value does not influence the weight vectors and the predictions if it lies in the interval $(0, \frac{1}{2})$ and if the class membership is attributed considering the class with the greatest conditional probability.
The regression in the X-space generates models that can be post-transformed according to the standard methods developed for PLS whereas for the regression in the Y-space the post-transformation has still not been developed.
The new methods behave properly also in a low-dimensional setting and have comparable results with respect to those of logistic regression.
As for running times, the regression in the X-space has slightly shorter times than PLS-DA regardless of the type of transformation, although the difference is negligible.
The strategy to use for solving a general G-class problem with $G > 2$ in the framework of PLS has been presented. However, the behavior of the proposed methods could be different from the case of 2 classes. Indeed, the value of ϵ could influence the predictions and the model could depends on the transformation used to map the simplex into the Euclidean space.
Then, further investigations are required to clarify the effect of the choice of ϵ and the role played by the transformation used. Moreover, in the case of not equally populated classes, a suitable scaling factor for the response is probably required to balance the different variance of the class-response during minimization since PLS is scale sensitive.

Bibliography

[1] A. Adelchi and B. Scarpa. *Data Analysis and Data Mining: An Introduction*. Oxford University Press, 2012.

[2] J. Aitchison. *A Concise Guide to Compositional Data Analysis*. MIT Press.

[3] A. Arredouani, M. Stocchero, N. Culeddu, J. El-Sayed Moustafa, J. Tichet, B. Balkau, T. Brousseau, M. Manca, and M. Falchi. "Metabolomic Profile of Low Copy-Number Carriers at the Salivary Alpha-Amylase Gene Suggests a Metabolic Shift Towards Lipid-Based Energy Production". In: *Diabetes* 65 (July 2016), db160315. DOI: 10.2337/db16-0315.

[4] D. Ballabio and V. Consonni. "Classification tools in chemistry. Part 1: Linear models. PLS-DA". In: *Analytical methods* 5 (Aug. 2013), pp. 3790–3798. DOI: 10.1039/c3ay40582f.

[5] M. Barker and W. Rayens. "Partial Least Squares For Discrimination". In: *Journal of Chemometrics* 17 (Mar. 2003), pp. 166–173. DOI: 10.1002/cem.785.

[6] M. S. Bartlett. "Further aspects of the theory of multiple regression". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 34.1 (1938), pp. 33–40. DOI: 10.1017/S0305004100019897.

[7] C. Bazzoli and S. Lambert-Lacroix. "Classification based on extensions of LS-PLS using logistic regression: Application to clinical and multiple genomic data". In: *BMC Bioinformatics* 19 (Dec. 2018). DOI: 10.1186/s12859-018-2311-2.

[8] R. Brereton and G. Lloyd. "Partial least squares discriminant analysis: Taking the magic away". In: *Journal of Chemometrics* 28 (Apr. 2014). DOI: 10.1002/cem.2609.

[9] A. Burnham, R. Viveros, and M. J.F. "Frameworks for latent variable multivariate regression". In: *J. Chemometrics* 18 (1996), pp. 31–45.

[10] S. Del Zotto. "The PLS regression model: algorithms and application to chemometric data". In: (2013).

[11] D. Dua and C. Graff. *UCI Machine Learning Repository*. Owner: Volker Lohweg. 2017. URL: http://archive.ics.uci.edu/ml.

[12] G. Fort and S. Lambert-Lacroix. "Classification using partial least squares with penalized logistic regression". In: *Bioinformatics* 21.7 (Nov. 2004), pp. 1104–1111. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bti114. URL: https://doi.org/10.1093/bioinformatics/bti114.

[13] P. Gemperline and O. Kvalheim. "History, philosophy and mathematical basis of the latent variable approach – from a peculiarity in psychology to a general method for analysis of multivariate data". In: *Journal of Chemometrics* 26 (June 2012). DOI: 10.1002/cem.2427.

[14] A. Höskuldsson. "PLS regression method". In: *Journal of Chemometrics* 2 (June 1988), pp. 211–228. DOI: 10.1002/cem.1180020306.

[15] S. de Jong. "SIMPLS: An alternative approach to partial least squares regression". In: *Chemometrics and Intelligent Laboratory Systems* 18.3 (Mar. 1993), pp. 251–263. DOI: 10.1016/0169-7439(93)85002-X.

[16] O. Kvalheim. "Interpretation of partial least squares regression models by means of target projection and selectivity ratio plots". In: *Journal of Chemometrics* 24 (July 2010), pp. 496–504. DOI: 10.1002/cem.1289.

[17] W. Lindberg, J. Persson, and S. Wold. "Partial least-squares method for spectro-fluorimetric analysis of mixtures of humic acid and ligninsulfonate". In: *Anal. Chem.* 55 (1983), pp. 643–648.

[18] E. Locci, M. Stocchero, A. Noto, A. Chighine, L. Natali, R. Caria, F. DeGiorgio, M. Nioi, and E. d'Aloja. "A ^{1}H NMR metabolomic approach for the estimation of the time since death using aqueous humour: An animal model". In: *Metabolomics* (2019).

[19] H. Martens and M. Martens. "Multivariate Analysis of Quality. An Introduction". In: *Measurement Science Technology - MEAS SCI TECHNOL* 12 (Oct. 2001), pp. 1746–1746. DOI: 10.1088/0957-0233/12/10/708.

[20] B. D. Marx. "Iteratively Reweighted Partial Least Squares Estimation for Generalized Linear Regression". In: *Technometrics* 38.4 (1996), pp. 374–381. ISSN: 00401706. URL: http://www.jstor.org/stable/1271308.

[21] M. McHugh. "Interrater reliability: The kappa statistic". In: *Biochemia medica* 22 (Oct. 2012), pp. 276–82.

[22] D. Nguyen and D. Rocke. "Tumor Classification by Partial Least Squares Using Microarray Gene Expression Data". In: *Bioinformatics (Oxford, England)* 18 (Feb. 2002), pp. 39–50. DOI: 10.1093/bioinformatics/18.1.39.

[23] P. Park, L. Tian, and I. Kohane. "Linking expression data with patient survival times using partial least squares". In: *Bioinformatics (Oxford, England)* 18 Suppl 1 (Feb. 2002), S120–7. DOI: 10.1093/bioinformatics/18.suppl_1.S120.

[24] N. Pérez, J. Ferré, and R. Boqué. "Calculation of the reliability of classification in Discriminant Partial Least-Squares classification". In: *Chemometrics and Intelligent Laboratory Systems* 95 (Feb. 2009), pp. 122–128. DOI: 10.1016/j.chemolab.2008.09.005.

[25] S. Piccinonna, R. Ragone, M. Stocchero, L. Del Coco, S. De Pascali, F. P. Schena, and F. Fanizzi. "Robustness of NMR-based metabolomics to generate comparable data sets for olive oil cultivar classification. An interlaboratory study on Apulian olive oils". In: *Food Chemistry* (Dec. 2015). DOI: 10.1016/j.foodchem.2015.12.064.

[26] E. M. Qannari and M. Hanafi. "A simple continuum regression approach". In: *Journal of Chemometrics* (2005).

[27] G. Raskutti, M. Wainwright, and B. Yu. "Early stopping for non-parametric regression: An optimal data-dependent stopping rule". In: *2011 49th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2011* (Sept. 2011). DOI: 10.1109/Allerton.2011.6120320.

[28] W. Rayens and A. Andersen. "Oriented partial least squares". In: *J. Appl. Stat.* 15 (Jan. 2003), pp. 367–388.

[29] M. Stocchero. "Exploring the latent variable space of PLS2 by post-transformation of the score matrix (ptLV)". In: *Journal of Chemometrics* 34 (Sept. 2018). DOI: 10.1002/cem.3079.

[30] M. Stocchero. "Iterative deflation algorithm, eigenvalue equations, and PLS2". In: *Journal of Chemometrics* 33 (Aug. 2019). DOI: 10.1002/cem.3144.

[31] M. Stocchero. "Relevant and irrelevant predictors in PLS2". In: *Journal of Chemometrics* (Apr. 2020). DOI: 10.1002/cem.3237.

[32] M. Stocchero, E. Locci, E. Baraldi, G. Giordano, E. d'Aloja, and M. Nioi. "PLS2 in metabolomics". In: *Metabolites* 9 (Mar. 2019), p. 51. DOI: 10.3390/metabo9030051.

[33] M. Stocchero and D. Paris. "Post-transformation of PLS2 (ptPLS2) by orthogonal matrix: A new approach for generating predictive and orthogonal latent variables". In: *Journal of Chemometrics* DOI: 10.1002/cem.2780 (Jan. 2016). DOI: 10.1002/cem.2780.

[34] M. Stocchero, S. Riccadonna, and P. Franceschi. "Projection to latent structures with orthogonal constraints for metabolomics data". In: *Journal of Chemometrics* 32 (Feb. 2018).

[35] A. Tenenhaus, A. Giron, E. Viennet, M. Bera, G. Saporta, and B. Fertil. "Kernel logistic PLS: A tool for supervised nonlinear dimensionality reduction and binary classification". In: *Computational Statistics and Data Analysis* 51 (Feb. 2007), pp. 4083–4100. DOI: 10.1016/j.csda.2007.01.004.

[36] M. Tenenhaus. “La regression logistique PLS”. In: (Jan. 2000).

[37] C. Wang, C. Chen, C. Chiang, S. Young, S. Chow, and H. Chiang. “A Probability-Based Multivariate Statistical Algorithm for Autofluoresence Spectroscopic Identification of Oral Carcinogenesis”. In: *Photochemistry and Photobiolog* 69 (Apr. 1999), pp. 471–477.

[38] H. Wold. “Path Models with Latent Variables: The NIPALS Approach”. In: *Quantitative Sociology: International Perspectives on Mathematical and Statistical Model Building* (1975), pp. 307–357.

[39] S. Wold, C. Albano, W. Dunn, K. Esbensen, S. Hellberg, and E. Johansson. “Pattern recognition: finding and using patterns in multivariate data”. In: *H. Martens, H. Russwurm Jr. (Eds.), Food Research and Data Analysis* (Oct. 1983), pp. 147–188.

[40] S. Wold, E. Johansson, and M. Cocchi. *3D QSAR in Drug Design; Theory, Methods, and Applications*. ESCOM, 1993.

[41] S. Wold, H. Martens, and H. Wold. “The multivariate calibration problem in chemistry solved by the PLS method”. In: *Proc. Conf. Matrix Pencils. Lecture Notes in Mathematics* (1983), pp. 286–293.

[42] S. Wold. “Personal memories of the early PLS development”. In: *Chemometrics and Intelligent Laboratory Systems - CHEMOMETR INTELL LAB SYST* 58 (Oct. 2001), pp. 83–84. DOI: 10.1016/S0169-7439(01)00152-6.

[43] S. Wold, M. Sjostrom, and L. Eriksson. “PLS-regression: A Basic Tool of Chemometrics”. In: *Chemometrics and Intelligent Laboratory Systems* 58 (Oct. 2001), pp. 109–130. DOI: 10.1016/S0169-7439(01)00155-1.

Appendix A

Mathematical notation

In this work, the common notation where column vectors are written in lower case characters (e.g. a) and the matrices in upper case characters (e.g. A) is used.
The transpose of a given matrix A is denoted $A^\top$, the scalar product between two vectors a and b is indicated as $a^\top b$ and the matrix product between two matrices A and B as AB. The juxtaposition of two matrices A and B is $[AB]$ and the identity matrix of size N is written as I_N.
A vector with p elements equal to zero or one is written as 0_p or 1_p respectively.
The Frobenius norm of a matrix $A \in \mathcal{R}^{n \times m}$, which means it has n observations and m columns, is defined as $\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$ and it corresponds to the Euclidean norm when A is a vector.
The matrix $\hat{Q}_t = I_N - t(t^\top t)^{-1}t^\top$ is defined as the orthogonal projection matrix that projects any matrix A onto the space orthogonal to the vector $t \in \mathcal{R}^N$.

Appendix B

Chapter 6

B.1 Algorithm 5

Let's prove that Algorithm 5 solves the least squares problem.
Consider the vector w_i that has the general form $w_i = Vh_i$, with h_i $(A \times 1)$: $h_i^\top h_j = \delta_{ij}$ being $A = \text{rank}(X)$ and $X = USV^\top$ the singular value decomposition.
When $A = \text{rank}(X)$, the matrix $H = [h_1, \ldots, h_A]$ is orthogonal and $W = VH$; therefore,

$$\beta_A = VH(H^\top V^\top X^\top XVH)^{-1}H^\top V^\top X^\top y = VS^{-1}U^\top y \tag{B.1}$$

and so the least squares problem is solved since $VS^{-1}U^\top$ is the Moore-Penrose inverse of X given the SVD of X and β_A is the vector of the regression coefficients with minimum norm.

B.2 Equivalence of the prediction of a new observation using alr, ilr, clr transformations

Firstly, we prove that using ilr-transformation the prediction of a new observation is equal to that obtained using the clr-transformation. Defining

$$\tilde{g}(y)_{clr} = \frac{1}{(N_A + N_B)} \ln\left(\frac{1-\epsilon}{\epsilon}\right) \begin{bmatrix} N_B 1_{N_A} & -N_B 1_{N_A} \\ -N_A 1_{N_B} & N_A 1_{N_B} \end{bmatrix}$$

and

$$\tilde{g}(y)_{ilr} = \frac{(v_1 - v_2)}{(N_A + N_B)} \ln\left(\frac{1-\epsilon}{\epsilon}\right) \begin{bmatrix} N_B 1_{N_A} \\ -N_A 1_{N_B} \end{bmatrix}$$

the mean centred Y-matrices obtained in the case of clr and ilr, respectively, we have

$$\tilde{g}(y)_{clr} = \frac{1}{(v_1 - v_2)} \begin{bmatrix} \tilde{g}(y)_{ilr} & -\tilde{g}(y)_{ilr} \end{bmatrix}$$

The vector of the regression coefficients b_{ilr} in the case of ilr is related to the matrix of the regression coefficients B_{clr} obtained by clr by

$$B_{clr} = \frac{1}{(v_1 - v_2)} W(W^\top X^\top X W)^{-1} W^\top X^\top \begin{bmatrix} \tilde{y}_{ilr} & -\tilde{y}_{ilr} \end{bmatrix} = \frac{1}{(v_1 - v_2)} \begin{bmatrix} b_{ilr} & -b_{ilr} \end{bmatrix}$$

Then in the Euclidean space, we have

$$\begin{aligned} y_{new,clr} \in \mathcal{R}^2, y_{new,clr} &= (x_{new} - \bar{x})^\top L_{scaling} B_{clr} + \bar{y}_{clr} = \\ &= \frac{1}{(v_1 - v_2)} \begin{bmatrix} (x_{new} - \bar{x})^\top L_{scaling} b_{ilr} + \bar{y}_{ilr} & -(x_{new} - \bar{x})^\top L_{scaling} b_{ilr} - \bar{y}_{ilr} \end{bmatrix} = \\ &= \frac{1}{(v_1 - v_2)} \begin{bmatrix} y_{new,ilr} & -y_{new,ilr} \end{bmatrix} \end{aligned}$$

where $y_{new,ilr} = (x_{new} - \bar{x})^\top L_{scaling} b_{ilr} + \bar{y}_{ilr} \in \mathcal{R}$ and $\bar{y}_{ilr} = \frac{(v_1 - v_2)}{2} \left(\frac{N_A - N_B}{N_A + N_B}\right) \ln\left(\frac{1-\epsilon}{\epsilon}\right)$.
Since

$$y_{new} = \text{softmax}(y_{new,clr}) \in \mathcal{S}^2$$

we have

$$ilr(y_{new}) = V^\top clr(\text{softmax}(y_{new,clr})) = V^\top y_{new,clr} = \frac{1}{(v_1 - v_2)} \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} y_{new,ilr} \\ -y_{new,ilr} \end{bmatrix} = y_{new,ilr}$$

Then, $ilr^{-1}(y_{new,ilr}) = ilr^{-1}(ilr(y_{new})) = y_{new}$.
The equivalence of the prediction of alr and clr is proven observing that

$$\tilde{g}(y)_{clr} = \frac{1}{2} \begin{bmatrix} \tilde{g}(y)_{alr} & \tilde{g}(y)_{alr} \end{bmatrix}$$

and

$$\begin{aligned} y_{new,clr} \in \mathcal{R}^2, y_{new_clr} &= (x_{new} - \bar{x})^\top L_{scaling} B_{clr} + \bar{y}_{clr} = \\ &= \frac{1}{2} \begin{bmatrix} (x_{new} - \bar{x})^\top L_{scaling} b_{alr} + \bar{y}_{alr} & -(x_{new} - \bar{x})^\top L_{scaling} b_{alr} - \bar{y}_{alr} \end{bmatrix} = \\ &= \frac{1}{2} \begin{bmatrix} y_{new,alr} & -y_{new,alr} \end{bmatrix} \end{aligned}$$

Indeed, we have

$$y_{new} = \text{softmax}(y_{new,clr}) = \left[\frac{e^{\frac{1}{2}y_{new,alr}}}{e^{\frac{1}{2}y_{new,alr}}+e^{-\frac{1}{2}y_{new,alr}}} \quad \frac{e^{-\frac{1}{2}y_{new,alr}}}{e^{\frac{1}{2}y_{new,alr}}+e^{-\frac{1}{2}y_{new,alr}}}\right] =$$
$$= \left[\frac{1}{1+e^{-y_{new,alr}}} \quad \frac{e^{-y_{new,alr}}}{1+e^{-y_{new,alr}}}\right] = alr^{-1}(y_{new,alr})$$

B.3 Properties of the Aitchison geometry on the simplex

The simplex $(\mathcal{S}^G, \oplus, \odot)$ with the perturbation operation and power transformation is a vector space; therefore the following holds:

$(\mathcal{S}^G, \oplus)$ has a commutative group structure, that is for $x, y, z \in \mathcal{S}^G$ the following properties hold:

1. Commutative property: $x \oplus y = y \oplus x$
2. Associative property: $(x \oplus y) \oplus z = x \oplus (y \oplus z)$
3. Neutral element: $n = \mathcal{C}[1, 1, \ldots, 1] = [\frac{1}{G}, \frac{1}{G}, \ldots, \frac{1}{G}]$ (n is the barycenter of the simplex and it is unique)
4. Inverse of x: $x^{-1} = \mathcal{C}[x_1^{-1}, x_2^{-1}, \ldots, x_G^{-1}]$; $x \oplus x^{-1} = n$

For the power transformation, given $x, y \in \mathcal{S}^G$, $\alpha, \beta \in \mathcal{R}$:

1. Associative property: $\alpha \odot (\beta \odot x) = (\alpha \cdot \beta) \odot x$
2. Distributive property 1: $\alpha \odot (x \oplus y) = (\alpha \odot x) \oplus (\alpha \odot y)$
3. Distributive property 2: $(\alpha + \beta) \odot x = (\alpha \odot x) \oplus (\beta \odot x)$
4. Neutral element: $1 \odot x = x$.

B.4 Inner product, norm and distance

To get a linear vector space structure, the following inner product, norm and distance are defined:

- Inner product of $x, y \in \mathcal{S}^G$:

$$\langle x, y \rangle_a = \frac{1}{2G} \sum_{i=1}^{G} \sum_{j=1}^{G} \ln \frac{x_i}{x_j} \ln \frac{y_i}{y_j} \tag{B.2}$$

- Norm of $x \in \mathcal{S}^G$:

$$\|x\|_a = \sqrt{\frac{1}{2G} \sum_{i=1}^{G} \sum_{j=1}^{G} \left(\ln \frac{x_i}{x_j} \right)^2} \tag{B.3}$$

- Distance between two compositions $x, y \in \mathcal{S}^G$:

$$d_a(x, y) = \|x \ominus y\|_a = \sqrt{\frac{1}{2G} \sum_{i=1}^{G} \sum_{j=1}^{G} \left(\ln \frac{x_i}{x_j} - \ln \frac{y_i}{y_j} \right)^2} \tag{B.4}$$

When referring to $(\mathcal{S}^G, \oplus, \odot)$ as an Euclidean linear vector space, we can call it *Aitchison geometry on the simplex*.