UCLouvain

epl

École polytechnique de Louvain

Placebo Modeling

Predicting patient pain from personality traits

Authors: **Alexandre Gerlache, Vincent Vandervilt**
Supervisors: **Pierre Dupont, Samuel Branders**
Reader: **Siegfried Nijssen**
Academic year 2018–2019
Master [120] in Computer Science and Engineering

Abstract

The importance of properly evaluating the placebo effect becomes more and more important as the gap between new drugs effect and the placebo effect shrinks, leading to the refusal of multiple new treatments. This thesis is realized in collaboration with Tools4Patient, a Belgian company developing a tool named Placebell that predicts the Individual Placebo Response (IPR) of the patients by using their answers to a psychological questionnaire (representing their personality).

In this thesis, the relationship between the pain of patients and their personality is evaluated using predictive regression models. The evaluation of such a relationship is interesting for Tools4Patient as it could be used to improve their products as well as giving them new leads for further development. To measure the placebo effect, the pain before and after taking the placebo are needed. If these are wrongly assessed by the patients from the beginning, the placebo effect measured will not be correct. The focus is on the baseline pain[1]. It is possible for a patient to badly evaluate it, depending on external factors such as his mood or the weather for example. The ultimate goal is to predict this pain using personality in order to correct the reported pain.

Our work is divided into two parts. On one hand there is an investigation of the questionnaire data and the patient pain and on the other hand the prediction of the baseline pain of a patient using his personality. Our focus for the first task is feature extraction : finding which part of the personality is the most relevant when it comes to pain. A quite stable set of questions (rallying several parts of the personality) was found. As for the second task, our goal is to identify the promising ways of making predictions. The results of our experiments in this regard are indeed quite promising and show the existence of a relationship between the personality of a patient and his baseline pain.

[1]The pain before taking the placebo

Acknowledgments

Firstly, we would like to thank our families and friends who have supported us during all these years and most notably during the writing of this thesis.

Our special thanks go to our supervisors Pierre Dupont and Samuel Branders for their constructive criticism, the ideas they pushed us to explore, their guidance when we were feeling lost and their help through the process of writing a scientific document.

Lastly, we would like to thank all our teachers and teaching assistants from UCLouvain for their teaching, their availability when facing a problem and the quality of the multiple projects that helped us put theory into practice.

Contents

List of Figures

List of Tables

Notations

Throughout this document, symbols and abbreviations are used. Here, the most used ones are described so that they stay consistent and understandable.

$x \in \mathbb{R}^{n \times p}$	x is usually the data used to make predictions. It is a matrix with each row being a sample and each column describing a feature (n rows and p columns).
$x_{i*} \in \mathbb{R}^{p}$	The i^{th} row of x.
$x_{*j} \in \mathbb{R}^{n}$	The j^{th} column of x.
$x_{ij} \in \mathbb{R}$	The element of x from the i^{th} row and the j^{th} column.
$y \in \mathbb{R}^{n}$	y is the response vector, so the data that needs to be predicted. It is a column vector with each row being a sample.
$\hat{y} \in \mathbb{R}^{n}$	The predicted response vector. It is the response predicted by a model.
$x^T \in \mathbb{R}^{p \times n}$	The transposed of the matrix x
$\bar{x} \in \mathbb{R}$	The mean of a vector x.
$w \in \mathbb{R}^{p}$	It is the weight vector used in linear models. It is a column vector.
$w^* \in \mathbb{R}^{p}$	It is the optimal weight vector found by an optimization process.
$E \in \mathbb{R}$	It is a scalar value representing the error that need to be minimized to build a given model.

$\|v\|_2 \in \mathbb{R}$	It is the L2-norm of the vector v. It is equal to : $\sqrt{\sum_{i=1}^{n}(v_i)^2}$
$\|v\|_1 \in \mathbb{R}$	It is the L1-norm of the vector v. It is equal to : $\sum_{i=1}^{n}\|v_i\|$
$E[X] \in \mathbb{R}$	It is the expected value of the random variable (RV) X.
$\mathrm{Var}(X) \in \mathbb{R}$	It is the variance of the RV X.
$\sigma_X \in \mathbb{R}$	It is the standard deviation of the RV X. It is equal to : $\sqrt{\mathrm{Var}(X)}$
$\mathrm{Cov}(X, Y) \in \mathbb{R}$	It is the covariance computed between the two RVs X and Y.
$r = \mathrm{Cor}(X, Y) \in \mathbb{R}$	Pearson's correlation coefficient. It is equal to : $\frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y}$
$z = F(r) \in \mathbb{R}$	Fisher transformation of a correlation coefficient (F is the *atanh* function)
Z	A z-score for hypothesis testing. It is a normally distributed variable.
$U(v_1, v_2)$	This notation defines an uniform random variable bounded between v_1 and v_2

Introduction

For every treatment that is being developed, the second phase toward public marketing is designed to ensure the effectiveness of the treatment.[22] This phase obviously takes into account the placebo effect to ensure that it is actually effective. To assess the placebo effect for treatments trying to alleviate the pain of the patient, it is required to evaluate the pain of this patient. It is in fact the difference between the pain evaluated before and after taking a placebo. Measuring it nonetheless raises some challenges as pain is a subjective measurement. This means that having a biased estimation of any of the two measurements will put a bias on the value of the placebo effect leading to a possible rejection of actually effective treatments.

That is why in this document, the existence of a relationship between the personality of a patient and the pain they reported is studied, in order to see if the personality can be used to predict the pain and also to see which traits of the personality are important for such a prediction. The patients that are considered suffer from one of two distinct diseases : Osteoarthritis (OA) or Peripheral neuropathic pain (PNP).

This work is done in collaboration with Tools4Patient, a Belgian company developing a tool named Placebell that predicts the Individual Placebo Response (IPR) of the patients by using their answers to a psychological questionnaire (representing their personality).

Placebo effect The placebo effect denotes an effect where people have a response (positive or negative) to a false (i.e. not supposed to affect the patient) treatment known as a placebo. Such a treatment may be a sugar pill when testing a drug for example.[27] This effect is crucial in the medical industry as it is required to make sure that the treatment being developed is indeed more effective than a placebo. Properly evaluating the placebo effect becomes more and more important as some studies even suggest that the strength of the placebo effect has been increasing whereas the strength of drugs was stable[29], giving a possible explanation as to

why most new analgesics were failing the placebo phase of clinical trials.[19]

Diseases studied

- **Osteoarthritis**[2], also called degenerative arthritis, is the most common chronic joint disease and most often occurs to people older than 65. The disease is characterized by a degradation of the cartilage within joints which causes pain and loss of joint mobility.[31]
- **Peripheral neuropathic pain**[3] is a disease where nerves in the peripheral parts of the body get damaged. It is often due to diabetes but it can happen because of physical injuries, viral infection or some drugs.[23]

Personality surveys A personality survey is a survey given to people (in this case, patients with osteoarthritis or peripheral neuropathic pain) in order to determine as well as possible their personality traits. A set of questions are asked and the answers are ordinal (usually on a scale from 1 to 5). These questions are grouped together and each group assesses one trait of personality. A trait of personality can be optimism or kindness for example. The survey handed to the patients was made internally, at Tools4Patient.

Outline

The possibility of predicting the baseline pain can help having a better estimation of the real pain experienced by a patient. The purpose of the prediction is to know the pain felt by the patient better than himself. For example, if a patient reports a high level of pain only because he was in a bad mood, it would be appreciable to be able to correct this measurement because it doesn't correspond to the real pain that he is feeling. If the pain reported wasn't accurate, then neither will be the estimation of the placebo effect. Many factors can change the pain of a patient : the weather, his mood, his tiredness, etc.

The first chapter defines some predictive regression models. They will be used in the experiments to make predictions.

The second chapter describes the experimental settings : the software tools used, the data set provided by Tools4Patient and some concepts such as Pearson's correlation coefficient and hypothesis testing. Pearson's correlation coefficient is

[2]often abreviated OA
[3]often abreviated PNP

used a lot throughout our work, mainly as performance criterion for regression models. Hypothesis testing is also widely used in our work. It allows to tell if a correlation coefficient is significantly different than 0 or not. Other tests can be used to test the uniformity of a distribution.

The third chapter is about performance assessment of predictive regression models. Several performance assessment schemes are presented. A simulation is also carried out to compare them and choose a reasonable one from a bias and variance standpoint.

The fourth chapter explains dimensionality reduction techniques. The data contains a lot of features. Some of them might be useless and add noise. The goal of dimensionality reduction is to either find relevant features or create more relevant ones using the ones that are available. The predictive power of these chosen (or new) features is then assessed in the experiments.

In the fifth and final chapter, the experimental results are presented and discussed. The performance of the predictive regression models are reported and so are the most relevant features for predictions.

Chapter 1

Predictive regression models

The ultimate goal of this thesis is to assess whether it is possible to predict the pain of a patient using its answers to a psychological questionnaire. For this purpose, predictive regression models are needed.

In supervised machine learning, there are two types of models : classification models and regression models. In both cases, a model is learned using (a part of) the available input and output data, and then predictions are made using only input data. The difference between the two is that in the classification case, the model predicts a class, among a finite set of possible classes, for each input sample. And in the regression case, the model predicts a value contained in a continuous range for each input sample. For our task, regression models are used because pain measurements are continuous.

In a nutshell, a predictive regression model estimates an unknown target function $f(x)$ such that $y = f(x) + \text{noise}$ (x being the input and y the output). It is built through a learning algorithm. The estimate of f is f^*, the closest function to f in the search space of the algorithm (all the possible functions that can be learned by an algorithm). Two types of fitting problems can happen :

1. Underfitting : the approximated function f^* is too simple and is not a good approximation of f which leads to poor performances, even on the training set (see the upper part of figure 1.1).

2. Overfitting : the approximated function f^* is too complex and not only approximated f but $f^*(x) = f(x) + \text{noise}$. The performance on the training samples is high but is very likely to be poor on new samples as noise is not predictable. It is a dangerous problem if methods to evaluate the error on new samples are not put into place as it may look like the model is great (see the bottom right corner of figure 1.1) when looking at performances

on already seen data. It is often advisable to prefer simpler models giving acceptable performance to avoid overfitting.

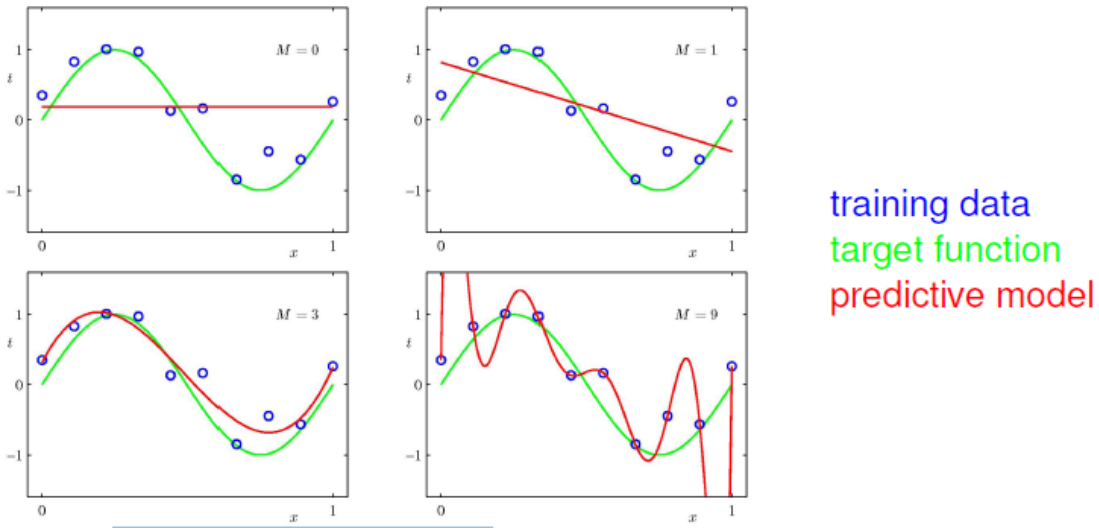


Figure 1.1: Four examples of models learned on the same data.[2] The ones on the upper part fit the data poorly. This illustrates the problem of underfitting. The one on the bottom left illustrates a good model fitting the data well but not perfectly, to be more general on new data. The one on the bottom right illustrates the concept of overfitting, the prediction will be poor for new samples.

1.1 Linear Least Square regression

The ordinary least square regression is probably the most well known regression technique. It is used in nearly every field, even by people not even aware about the concept of machine learning. This model assumes that the input data has a linear relationship with the response data (output). Let $x \in \mathbb{R}^{n \times p}$ be the input data, $y \in \mathbb{R}^{n}$ the output data and $w \in \mathbb{R}^{p}$ a vector of weights :

$$x = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix}, \; y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \text{ and } w = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_p \end{pmatrix}$$

The real unknown linear relation is defined as such :

$$y = xw + w_0$$

where w and w_0 are unknown. The leaning algorithm finds a model that is supposed to approximate the linear relationship :

$$\hat{y} = xw^* + w_0^*$$

Most of the time, to make notations simpler, it is assumed that the first feature of every sample of x is always 1 so that the term w_0^* is included in w^* :

$$x = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1p} \\ 1 & x_{21} & x_{22} & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix} \text{ and } w = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_p \end{pmatrix}$$

The equation becomes :

$$\hat{y} = xw^*$$

Basically, the goal of the method is to learn the vector w^* that minimizes the mean square error criterion E :

$$\begin{aligned} E &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \\ &= \frac{1}{n}\sum_{i=1}^{n}(y_i - x_{i*}w)^2 \\ &= \frac{1}{n}\|y - xw\|_2^2 \end{aligned}$$

Fortunately, the criterion is convex so there is only one optimum[1]. The optimum can be found by finding the point that makes the gradient of error null. And so :

$$\frac{\delta E}{\delta w} = \frac{2}{n}x^T(xw - y)$$

To optimize the criterion :

$$\frac{\delta E}{\delta w} = \frac{2}{n}x^T(xw - y) = 0$$

It is supposed that x is not a null matrix so :

$$x^T xw - x^T y = 0$$

[1] a global optimum

$$x^T xw = x^T y$$

And so the optimal solution for w^* is :

$$w^* = (x^T x)^{-1} x^T y$$

This solution has nonetheless a computational problem, as it features a matrix inversion to find the optimal weight vector. They are usually avoided because matrices are often ill-configured[2] which makes the inversion difficult or even impossible.

An other way to find the solution is to use gradient descent to iteratively minimize the error criterion. As the criterion is convex, the optimization will find the optimal solution. The principle of gradient descent is to go from a initial solution to a problem, and iteratively change the solution by moving in the (resp. opposite of the) direction of the gradient to (resp. minimize) maximize the objective. So here, an iteration to get w at time $t+1$ goes as follows:

$$\begin{aligned} w(t+1) &= w(t) - \alpha \frac{\delta E}{\delta w(t)} \\ &= w(t) - \alpha \frac{2}{n} x^T (xw - y) \end{aligned}$$

With α being the learning rate which should be low enough not to miss the optimal solution. The classical solution is to set $\alpha(t) = \frac{1}{t}$ so that the optimization process is fast at the beginning and precise at the end. With enough iterations, the weight vector will converge to the optimal solution w^*.

1.2 Ridge regression

The Ridge regression is a well established regularized technique for linear regression. It is very similar to the Least square regression. The only difference is that in Ridge regression, a regularization term is added to the error criterion. The term prevents the weight vector from having weights that are too great in order to improve the generalization on new samples. It is implemented with the L2-norm of w being added to the error criterion. This technique doesn't reduce the number of features used like the Lasso regression or the Elastic Net.[16]

In this model, the formulation of the error is :

$$E = \lambda \|w\|_2^2 + \|xw - y\|_2^2$$

[2]determinant close or equal to 0

with λ being a positive constant that changes the compromise between minimizing the error (small λ) and maximizing the regularization (big λ). The criterion is convex so there is only one global optimum and it can be optimized by finding the weight vector that sets the error gradient to 0 :

$$\begin{aligned}\frac{\delta E}{\delta w} &= \frac{\delta}{\delta w}(\lambda\|w\|_2^2 + \|xw - y\|_2^2) \\ &= \lambda\frac{\delta}{\delta w}(\|w\|_2^2) + 2x^T(xw - y) \\ &= 2\lambda w + 2x^T(xw - y)\end{aligned}$$

The gradient of error is set to 0 to optimize the error :

$$\begin{aligned}\lambda w^* + x^T x w^* - x^T y = 0 \\ (\lambda I + x^T x)w^* = x^T y\end{aligned}$$

The solution is easily obtained :

$$w^* = (x^T x + \lambda I)^{-1} x^T y$$

Which is nearly the same solution as LSE regression, with just λI added. This has as an advantage that the matrix is less likely to be ill-configured and in consequence easier to invert.[25]

1.3 Lasso regression

The Lasso regression is a variation of the "classical" ordinary least square regression in a comparable way than the Ridge regression. The constraint added to the error criterion isn't a L2-norm but a L1-norm of the weight vector. This technique has two desirable properties. It is used as a regularization technique much like the Ridge regression which may improve the generalization error and it is also used as an embedded variable selection technique to produce sparse models that are more interpretable. Indeed the L1-norm constraint tends to set a lot of weights to 0.[28]

In this model, the formulation of the error is :

$$E = \lambda\|w\|_1 + \|xw - y\|_2^2$$

Nonetheless, this error is not as easy to optimize as the one from the Ridge regression as it is not strictly convex. When the error is optimized, a linear model is

built with w and predictions are made the same way as in the ordinary least square regression.

Even if the Lasso has the interesting property of generating sparse models, it has multiple downsides[33] :

1. If there are more features than samples : $p > n$, a maximum of n features are selected by the Lasso.

2. If the pairwise correlation of a group of features is very high, only one feature of the group is selected and it is selected more or less randomly.

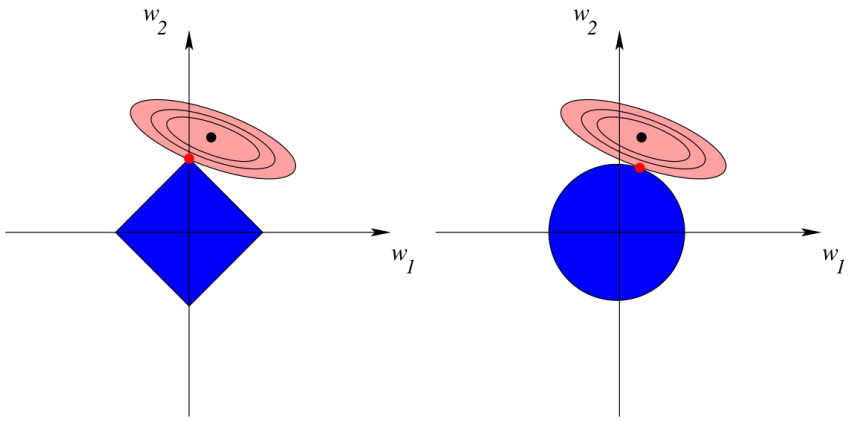


Figure 1.2: Geometric comparison of the regularization term of Ridge (on the right) and Lasso regression (on the left)[3]. The blue shapes represent the area of possible solutions if the regularization term is bounded. The red ellipsoids represent zones of equal Least Square Error (LSE). The center of the elipsoïds is the solution for the LSE regression. The solution for the Ridge and the Lasso is the intersection of the blue shape and the smallest ellipsoid that intersects with it. It is geometrically evident that it is likely for the solution of the Lasso to be sparse.

1.4 Elastic Net

The Elastic Net is a newer regularization technique that is a mixture between the Lasso and the Ridge regression as it uses a weighted sum between the L1-norm and L2-norm of the weight vector as a constraint. This technique addresses some

[3]This figure comes from slides about feature selection made by Mr. Pierre Dupont.

shortcomings of the Lasso regression while being as good as Ridge from a performance standpoint when Lasso does poorly.[33]

In this model, the formulation of the error is :

$$\begin{aligned} E &= a\|w\|_1 + b\|w\|_2^2 + \|xw - y\|_2^2 \\ &= \lambda(\alpha\|w\|_1 + (1-\alpha)\|w\|_2^2) + \|xw - y\|_2^2 \end{aligned}$$

This formulation indeed highlights the fact that this model is a mixture of the Ridge and the Lasso regression, with it being a simple Lasso if $\alpha = 1$ and a simple Ridge if $\alpha = 0$. From a feature selection standpoint, the Elastic Net has the advantage of selecting groups of highly correlated variables together rather than selecting only one randomly.

1.5 Support Vector Machine

1.5.1 SVM classification

The support vector algorithm was originally proposed for binary classification problems with the input data being $x \in \mathbb{R}^{n \times p}$ and the response being $y \in \{-1, 1\}^n$. The goal of the simple SVM is to find an optimal hyperplane defined by the weight vector w and the scalar w_0. It is the hyperplane that separates the response samples (-1 on one side and +1 on the other side) with the maximum **margin**.

This simple SVM (called a hard margin SVM) nonetheless assumes that the data is indeed perfectly linearly separable, meaning that there exists a separating hyperplane defined by the weight vector w and the scalar bias w_0 such that :

$$x_{i*}w + w_0 > 0 \iff y_i = +1 \ \forall i \in \{1..n\}$$

and

$$x_{i*}w + w_0 < 0 \iff y_i = -1 \ \forall i \in \{1..n\}$$

Or more simply :

$$(x_{i*}w + w_0)y_i > 0 \ \forall i \in \{1..n\}$$

With the separating hyperplane being the set of points $x' \in \mathbb{R}^p$ such that :

$$g(x') = x'w + w_0 = 0$$

The margin of the SVM is informally defined as the minimal distance between the hyperplane and a sample. Maximizing it usually yields better generalization as it avoids having the hyperplane too close to the samples which means unseen samples are less likely to be misclassified.

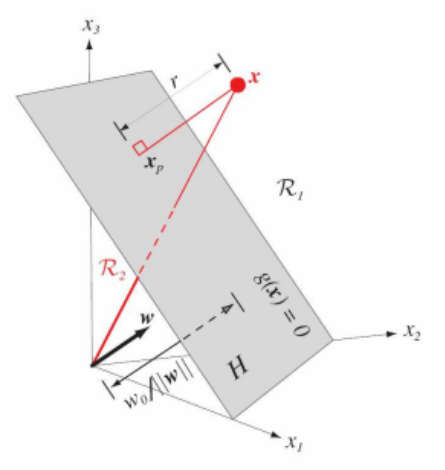


Figure 1.3: Figure illustrating the meaning of the distance of a point to a hyperplane[6]. x is a point, x_p is the projection of x on the hyperplane H and r is the distance between the point and the hyperplane.

The distance from a point to an hyperplane (figure 1.3) is the length of the vector $x - x_p$ where x_p is the projection of x on the hyperplane. This distance is given by :

$$r = \frac{xw + w_0}{\|w\|}$$

For the optimality criterion to make sense, it is evident that a constraint on either the size of the margin or the size of w has to bet set in place because the margin can be made arbitrarily big by multiplying w and w_0 by a constant. The usual solution is to fix $xw + w_0 = 1$ for support vectors and so to maximize the margin, $\|w\|$ or equivalently $\frac{1}{2}\|w\|^2$ needs to be minimized[7].

Consequently, the problem is formulated as such :

$$w^* = \arg\min_{w \in \mathbb{R}^p} \frac{1}{2}\|w\|_2^2$$

with the constraint :

$$(x_{i*}w^* + w_0)y_i \geq 1 \ \forall i \in \{1..n\}$$

This can be solved as a lagrangian optimization problem :

$$L(w, w_0, \alpha) = \frac{1}{2}\|w\|_2^2 - \sum_{i=1}^{n} \alpha_i (y_i(x_{i*}w + w_0) - 1)$$

The lagrangian needs to be minimized with respect to w and w_0 as it is the goal of the original optimization problem and maximized with respect to α. By computing the derivatives with respect to w and w_0, it is obtained that : $\sum_{i=1}^{n} \alpha_i y_i = 0$ and $w = \sum_{i=1}^{n} \alpha_i y_i x_{i*}^T$. It can also be transformed into the dual optimization problem by replacing those results in the original primal lagrangian form :

$$\alpha = \arg\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_{i*} x_{j*}^T$$

with :

$$\alpha_i \geq 0 \ \ \forall i \in \{1..n\} \quad \text{and} \ \sum_{i=1}^{n} \alpha_i y_i = 0$$

which needs to be solved in order to obtain the dual variables α that are necessary to compute w. Also, according to the Karush-Kuhn-Tucker conditions (as it maximizes L) :

$$\begin{aligned} \alpha_i = 0 &\iff y_i(x_{i*}w + w_0) > 1 \\ \alpha_i > 0 &\iff y_i(x_{i*}w + w_0) = 1 \text{ in which case } x_i \text{ is a support vector} \end{aligned}$$

So w_0 can be computed from the equality :

$$w_0 = y_i - x_{i*}w \text{ with } x_{i*} \text{ being a support vector}$$

To predict a new sample x' (a row vector) it is as simple as :

$$\begin{aligned} \hat{y}' &= \text{sign}(x'w + w_0) \\ &= \text{sign}(x' \sum_{x_{i*} \in \text{SV}} \alpha_i y_i x_{i*}^T + w_0) \quad \text{with SV being the set of all support vectors} \end{aligned}$$

It is evident from the equations that only the support vectors, α and w_0 are needed to make a prediction.

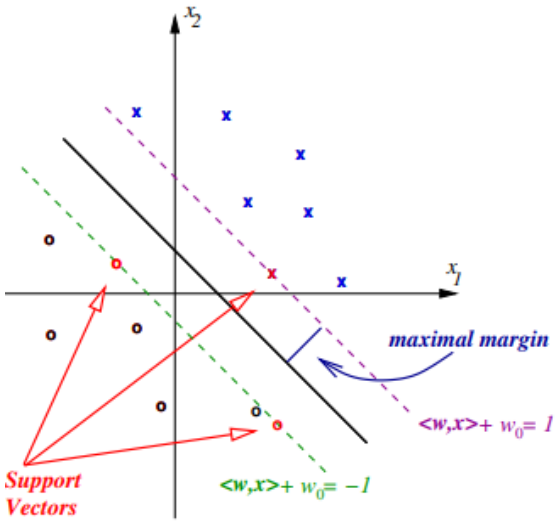


Figure 1.4: 2D example of hard margin SVM, data is linearly separable. The figure illustrates the maximal margin for a given set of samples. It shows that the support vectors are the samples that are the closest to the hyperplane (i.e. on the margin)[7]. $< w, x >$ is the scalar product of w and x.

Nonetheless, the hard margin SVM classification algorithm is deficient because it does not work with non linearly separable data. Moreover, it is sometimes better to misclassify a training sample for the sake of better generalization. The soft margin SVM algorithm was introduced to solve both of those problems.

The formulation in this case is :

$$w^* = \arg\min_{w \in \mathbb{R}^d} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n}(\max(0, 1 - y_i(x_{i*}w + w_0)))$$

with C being the cost. This value is used to tune the compromise between the margin maximization and the minimization of the number of misclassifications (margin error). If $C = \infty$, it is equivalent to the hard margin SVM. The $\max(0, 1 - y_i(x_{i*}w + w_0))$ term is here to encourage the model to avoid misclassification as it is equal to 0 if a sample satisfies the margin constraint $(y_i(x_{i*}w + w_0) \geq 1)$. The difference with the hard margin is that misclassifications are allowed if it makes the solution optimal for a given value of C. Also it is possible to have samples that are well classified but still violate the margin constraint $(1 > y_i(x_{i*}w + w_0) > 0)$.

The dual problem in this case is :

$$\alpha = \arg\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j x_{i*}x_{j*}^T$$

with :

$$0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0$$

In this case the Karush-Kuhn-Tucker conditions are :

$$\begin{aligned} \alpha_i = 0 &\iff y_i(x_i w + w_0) > 1 \text{ for well classified points} \\ 0 < \alpha_i < C &\iff y_i(x_i w + w_0) = 1 \text{ for support vectors} \\ \alpha_i = C &\iff y_i(x_i w + w_0) < 1 \text{ for misclassified samples or one that violate the margin} \end{aligned}$$

The expression of the weight is the same as the one in the hard margin SVM :

$$w = \sum_{i=1}^{n} \alpha_i y_i x_{i*}^T$$

1.5.2 SVM regression

SVM regression adapts the concept of using a margin to achieve a better regularization for regression problems. Instead of separating the samples, the margin or loss sensitivity ϵ is set and the goal is to enclose a maximum number of points between the margins while minimizing the error. Only the samples outside the margin will have an influence on the error that the algorithm minimizes. It reduces the algorithm sensitivity to small errors in order to avoid overfitting.

The hard margin SVR solution comes from solving this optimization problem :

$$w^* = \underset{w \in \mathbb{R}^p}{\arg\min} \frac{1}{2} \|w\|_2^2$$

under the constraints :

$$-\epsilon \leq y_i - x_{i*} w - w_0 \leq \epsilon \ \forall i \in \{1..n\}$$

There is a similar problem to the one in the hard margin SVM. If ϵ is too small, a solution may not be found because all points cannot all be between the margins. To alleviate this problem, a soft margin algorithm can also be set in place. Figure 1.5 illustrates the behavior of a SVR with soft margins. The constraints are removed and a loss term is added to the minimization. In this thesis, only two loss functions are considered with SVM regression, but other loss functions

may be used. The first one is the L2-Loss that is the same loss used in a simple least-square error regression. The second one is the L1-Loss that uses the distance between the hyperplane and the point instead of the squared distance. This type of loss usually makes the solution less influenced by outliers.[14]

$$w^* = \min_{w \in \mathbb{R}^p, w_0 \in \mathbb{R}} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n}(\max(0, |x_{i*}w + w_0 - y_i| - \epsilon)^2) \qquad \text{L2-Loss}$$

$$= \min_{w \in \mathbb{R}^p, w_0 \in \mathbb{R}} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n}(\max(0, |x_{i*}w + w_0 - y_i| - \epsilon)) \qquad \text{L1-Loss}$$

This problem can be transformed as a lagrangian dual optimization problem in order to solve it.

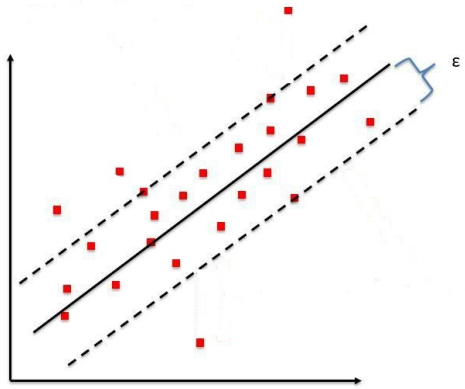


Figure 1.5: Example of SVR with soft margins[4]. The middle line $x'w + w_0$ is the regression line found by the algorithm. The dashed lines are the margins, they enclose as many training points as possible. Only training points outside the margins are considered in the loss function.

Those two types of formulations are interesting as the minimization of the L2-norm offers good regularization, ϵ makes the model insensitive to really small error and C controls the compromise between the regularization and the loss ($C = \infty$ makes a model that tries to fit a maximum of point at a distance inferior to ϵ and $C = 0$ makes a model with $w = 0$). Figure 1.6 is an example of different models learned on the same data. It shows how well regularization and the loss functions work.

It is also remarkable that the formulation is similar to the error minimized by the Ridge regression algorithm : in fact, the Ridge regression is a special case of the L2-Loss SVR with $\epsilon = 0$ and $\lambda = \frac{1}{2C}$.

[4]This illustration comes from this website : `https://medium.com/coinmonks/support-vector-regression-or-svr-8eb3acf6d0ff`

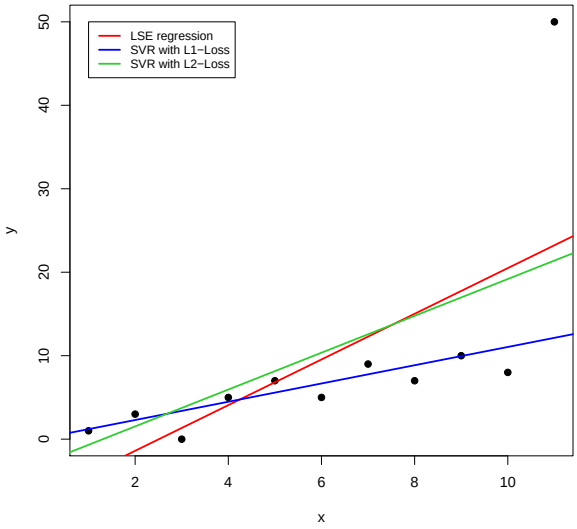


Figure 1.6: Comparison of three models learned by some linear regression algorithms on a toy example. In red : ordinary linear least square regression, in blue : SVR with L1-loss and in green : SVR with L2-loss. Least square regression is very sensitive to the outlier. SVR with L2-loss is a little less sensitive thanks to the regularization and SVR with L1-loss is almost insensitive to the outlier because the loss term isn't squared.

1.6 Random Forest

1.6.1 Decision trees

Decision trees are a well established technique to perform classification or regression. They have the good taste to be easily interpretable (at least if the size of the tree is reasonable) and can be graphically represented. They can also be reasonable models without too much overfitting depending on the way the tree was built.

In this section, the tree building part of the CART algorithm[13] for classification and regression problems is explained. This method is not directly used in the experiments but it is crucial in order to understand the Random Forest algorithm explained in a later sectionlater section.

A decision tree assigns a class (or a value in the case of regression) to a new sample by moving down the tree until reaching a leaf node, where these leaf nodes

each hold a class (or a value). Each non-leaf node assesses one feature and has multiple children[5], each child represents a (range of) value(s) for that feature. The new sample is moved toward a child depending on its feature value. As figure 1.7 illustrates, each node contains a question that separates the samples between several nodes. Classifying a new sample is as easy as following the path indicated by the questions depending on the sample values.

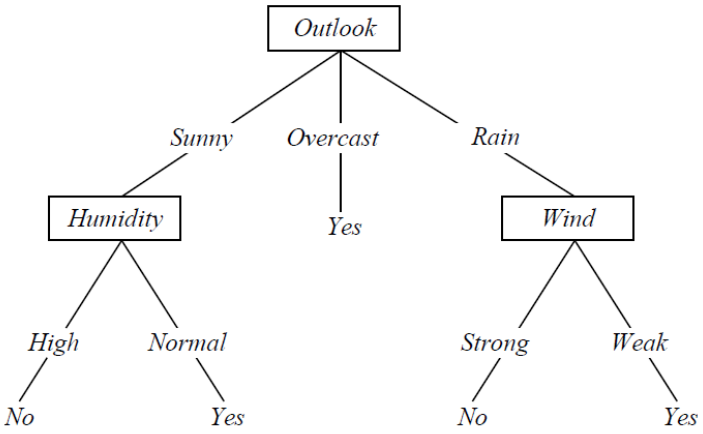


Figure 1.7: Graphical representation of an example of a classification tree. It is designed to decide whether or not to play tennis today.[18]

To build the tree, a greedy procedure is used to choose which feature will be used at each node to create a split. In CART, building the tree can be decomposed in 3 steps :

1. Build the maximal tree. It is done by recursively splitting the samples between 2 nodes. No more splits are done when nodes either contain only one sample or when a termination criterion is met. The tree that is produced at the end of this step is very likely to overfit the data.

2. Prune the maximal tree by iteratively deleting more and more important splits, which creates a sequence of trees going from the most complicated one to the most simple one.

3. Select the best tree from the sequence : the one that has the best performance without overfitting.

Only the first step of CART is explained, as Random Forest does not use the pruning capability of CART.

[5]two in case of CART as it only uses binary splits when building the tree

Building the tree

Building the regression tree is a recursive procedure. At each node, if the stopping criteria are not met[6], all the possible splits are evaluated and the one that optimizes a splitting criterion is chosen. In the classification case such criteria are "entropy", "gini" or "twoing".[13]

In the regression case, the Sum of Square Error (SSE) may be used. Let $S = \{1..n\}$ be the set of indices of the samples. The considered split, splits S into two distinct subset S_1 and S_2. c_i is the mean of the response of samples with indexes in S_i such that :

$$c_i = \frac{1}{|S_i|} \sum_{j \in S_i} y_j$$

The criterion to choose the best split is then :

$$SSE = \sum_{j \in S_1} (y_j - c_1) + \sum_{j \in S_2} (y_j - c_2)$$

The split giving minimal SSE is chosen. Splits are done until a leaf node is reached. The value of a leaf node is then the mean of the response values of the samples present in that node.

The tree at this point, probably overfits the data, but this can be solved by either pruning the tree which is done in the CART algorithm, or using ensemble learning techniques like Random Forest.

1.6.2 Random Forest, an ensemble tree model

Random Forest is an ensemble method to build models based on classification or regression trees. It builds a high number of trees and assigns a class (or value) to a new sample by taking a majority vote (or the mean) of the classes (values) assigned by all the trees. It usually offers a good generalization error, is robust to noise and keeps a lot of the advantages of decision tree models (compatible with classification and regression, compatible with ordinal, continuous or categorical input and missing values)[3]. It nonetheless loses the interpretability of decision trees as the high number of trees makes it hard to visualize.

Obviously, Random Forest doesn't work by just building a lot of trees on the same data as it would just be a lot of times the same tree. The building process

[6]only one sample left in the node, all samples have the same values for each feature, all samples have the same response or the maximum depth of the tree is reached

works by introducing randomization in it. There are two sources of randomization in the implementation of Random Forest used for our work. First, a modification of the splitting choice made by the CART algorithm. At each node when creating a new split, instead of choosing the best split among all the available ones, only splits on a random subset of F features are considered. The best split is chosen from the all the possible ones on any feature of the random subset. Secondly, sampling is also used to build trees. Instead of using the whole set of data to build a tree, only a part of it is used (i.e. sampled at random). Moreover, replacements can be introduced in the sampling, meaning that a sample can be taken more than once to build the tree.

Chapter 2

Experimental settings

2.1 Software Tools

To carry out all the experiments and simulations, we used the programming language R and some of its packages :

- *dplyr* is a package to help with data manipulation such as aggregation on data frames[32]
- *glmnet* implements Ridge regression, Lasso Regression and elastic net[8]
- *liblinear* implements regularized linear SVM regression[15]
- *randomForest* implements regression using Random Forests[24]

2.2 Data set : psychological questionnaire and reported pain

The data came from 3 studies carried out by Tools4Patient using a psychological questionnaire (called MPSQ). The first two were carried out at baseline[1] :

- T1001-01 : The first study, on PNP (Peripheral Neuropathic Pain) patients
- T1001-02 : The Second study, on PNP and OA (Osteoarthritic) patients
- T1001-03 : The follow up study[2] with patients from the first 2 studies

[1] During the week before taking the placebo
[2] Study made a while after the first ones in order to verify that the answers are stable

The first two studies are very similar. The main difference is the diseases considered and the version of the questionnaire (MPSQ_v1 for study T1001-01 and MPSQ_v2 for study T1001-02). The second version was also used for study T1001-03. The first questionnaire was modified by removing useless or redundant questions and adding new ones that are more relevant.

The data consists of 6 R matrices :

- *mpsqData*, *mpsqData2*, *mpsqData3* : The first 3 matrices consist of the answers of patients (rows) to a set of 147 psychological questions (columns, also called items) from MPSQ_v2 such as "I enjoy meeting new people" or "I'm easily disappointed". The answers are ordinal and range from 1 to 5, 1 meaning "I totally disagree" and 5 meaning "I fully agree".
  - *mpsqData* : Answers from 151 patients from the first and second studies at the baseline. Since only questions from the second version of the questionnaire are reported, there are some missing values for patients of the first study.
  - *mpsqData2* : Answers from 110 patients from the end of the second study.
  - *mpsqData3* : Answers from the follow-up study, some patients from the first and second study came back and took the questionnaire again. The goal of this follow-up study is to assess the long term stability of the questionnaire (1-3 years).

  Tools4Patient provided some of the questions of the psychological questionnaire but not all of them. The facets were kept confidential.
- *patientInfo* contains additional information about all the patients (one row per patient) :
  - Age : age of the patient at his baseline
  - Sex : 0 for female and 1 for male
  - Disease : 0 for the OA and 1 for the PNP patients
  - MPSQ_version : 1 or 2 for MPSQ_v1 or MPSQ_v2
- *mpsqInfo* contains information about the questions in the questionnaire (one row per question) :
  - Facet : facet number of the question (a facet defines a group of questions around an aspect of the personality, e.g. kindness, optimism, etc)

- Reverse : Some questions are asked in a positive way and some are asked in a negative way (compared to the facet they belong to). Reverse indicates which questions must be reversed, i.e. the ones that were asked in a negative way, so that they fit the aspect of the personality correctly and not oppositely. 0 means the question is not to be reversed and 1 means the question is to be reversed. To reverse a question, simply replace the answers by $6 -$ answers (example : for the facet "kindness", the question "I enjoy hurting people" is to be reversed because it fits oppositely the personality trait. If a patient answers 5 to this question, the reversed answer is $6 - 5 = 1$).
- MPSQ_v1 : Questions are asked in a specific order. MPSQ_v1 defines the position of the question in the first version of the questionnaire. This could be used to try to see if people tend to answer in a bad way when they have already answered a lot of questions (because they are bored or tired for example).
- MPSQ_v2 : Same but for the second version of the questionnaire

- *painData* contains baseline pain measurements from the first 2 studies (one row per patient) :
  - APS_Baseline : Daily Average Pain score during the baseline week (between 0-10)
  - WPS_Baseline : Daily Worst Pain score during the baseline week (between 0-10)
  - BPI-Severity_V2 (BPI : Brief Pain Inventory[3]) : Mean of the maximum pain, minimum pain, average pain and current pain during the last day of the baseline week (between 0-10)
  - BPI-Interference_V2 : Rates how much the pain interferes with daily life
  - BPI-Effet_V2 : How much the treatment helps (between 0-100)
  - IGAC_v1 : Investigator Global Assessment of Change (between 0-10)
  - PGAC_v1 : Patient Global Assessment of Change (between 0-10)

[3]It measures both the intensity of pain and its interference with daily life. It also takes into account the patient's perception of pain, its cause and its relief.[4]

2.3 Statistical hypothesis testing, p-value and confidence intervals

A statistical hypothesis is an assumption about a parameter of a set of random variables (a population). It is a hypothesis that can be tested using a statistical hypothesis test. It will be used widely in our work to assess results of experiments. To perform such a test, several elements are needed [30, chapter 10] :

- A null hypothesis H_0 : a hypothesis to be tested.
- An alternative hypothesis H_a : the hypothesis to accept if H_0 is rejected.
- A test statistic Z : a function of the sample measurements.
- A significance level α : a probability threshold below which the null hypothesis will be rejected, a value of 1% or 5% is usually used. It defines the rejection region (values of the test statistic for which the null hypothesis is to be rejected). It is also the probability of type 1 error (rejecting the null hypothesis while it is true).

Assumptions about the distributions of the observations are also to be made so that the distribution of the test statistic can be derived. Once everything is set, the test can be carried out. The observed value of the test statistic z_{obs} is computed from the observations and the decision about the null hypothesis is made.

There is an alternative way to test the hypotheses. Once z_{obs} is computed, the p-value associated is calculated. It is defined as the smallest value of α for which the null hypothesis can be rejected. Another definition for the p-value is : the probability under the null hypothesis to observe a test statistic at least as extreme as the one that was observed. If and only if the p-value is less than the significance level α, then the null hypothesis is rejected in favor of the alternative hypothesis. This p-value is often reported because it gives more information than simply reporting the rejection (or not) of the null hypothesis. To compute a p-value, the distribution of the test statistic must be known or assumed. Either use probability tables (depending on the distribution) and interpolate if the table doesn't contain the p-value associated to the observed statistic, or use the cumulative distribution function of the test statistic.

From such tests, confidence intervals can be obtained. A confidence interval is a range of potential values of the unknown parameter to be estimated. However it does not always include the true value of the parameter. A confidence level is associated to this interval, it quantifies the level of confidence that the true

value belongs to the interval (often 95%). 95% of confidence means that a future confidence interval will encompass the true value of the parameter with a probability of 95%. It does not mean that the true value of the parameter is included in the range with a 95% probability.
They are computed using the distribution of the test statistic and the level of significance.

2.4 The χ^2 test of goodness of fit to test the uniformity of a distribution

In the **next chapter**, the performance assessment of predictive regression models is presented. That chapter aims to choose an assessment scheme. Several criteria are to be met when making such a choice. The performance must be well computed, i.e. the bias on the performance has to be as low as possible. The variance of the performance must also be as low as possible. And finally, the p-values associated to the statistical test carried out on the performance must be computed correctly. This latter criterion can be verified quite easily.

The p-values are correctly computed if, under the null hypothesis, their distribution is uniform[20]. Indeed it comes from the definition of the p-value and the confidence level α (also defined as the probability of type 1 error, i.e. the null hypothesis is true but is rejected). The framework is that the null hypothesis is rejected when the p-value is lower than α because it means that if the null hypothesis is indeed true, the probability to reject it is inferior to α and so it is deemed unlikely. For the p-value to have this meaning, its distribution must effectively be uniform between 0 and 1 under the null hypothesis because it implies that if a threshold α is chosen to reject it, there is effectively a probability α to make a mistake as the probability of having the p-value inferior to α under the null hypothesis is equal to α.

The χ^2 test of goodness of fit can be used to test several hypotheses[5]. One of them is the uniformity of a distribution[9]. Let J be the number of p-values computed. These p-values are grouped into l bins $[b_1, b_2, ..., b_l]$, i.e. ranges of values (of same width). The number of bins l is chosen by the *hist* function from the R language. It is computed depending on the number of p-values, the more p-values are given, the more bins are used. Each bin has thus an associated frequency $[f_1, f_2, ..., f_l]$ being the number of p-values fitting in the bin. Let p_i be the probability for a p-value to belong to bin i. The four elements needed for the statistical test are :

- The null hypothesis H_0 : $p_i = \frac{1}{l}$ $\forall i \in [1..l]$ (the p-values are uniformly distributed)
- The alternative hypothesis H_a : $\exists i \in [1..l]$ s.t. $p_i \neq \frac{1}{l}$ (the p-values aren't uniformly distributed)
- The test statistic (denoted χ^2 to follow the literature and because the distribution of such a test statistic is χ^2 and not normal) : $\chi^2 = \sum_{i=1}^{l} \frac{(f_i - \frac{J}{l})^2}{\frac{J}{l}}$
- The significance level $\alpha = 0.05$

Once the test statistic is computed, a p-value can be derived to see if the null hypothesis has to be rejected or not. It can be found in a χ^2 table or computed via the cumulative probability density function of the χ^2 distribution. It depends on the number of degree of freedom of the test statistic which is equal to the number of bins minus 1. If the p-value is greater than α, the null hypothesis cannot be rejected meaning that it cannot be affirmed that the distribution of the p-values is not uniform. In R, the function used to compute the p-value is *pchisq*.

2.5 Pearson correlation coefficient

In this document, the Pearson correlation coefficient is used for multiple purposes. This coefficient is a well known measure of the existence of a linear relationship between two random variables. The coefficient values are bounded between -1 and +1, -1 meaning that there is a perfect negative linear relationship between the variables and 1 meaning that the relationship between the variables is perfectly linear and positive. In R, the function to compute the sample correlation between two vectors is called *cor*.

Given two random variables X and Y, the correlation between them is computed as such :

$$\begin{aligned} r &= \frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y} \\ &= \frac{\mathrm{E}[(X-\bar{X})(Y-\bar{Y})]}{\sqrt{\mathrm{E}[(X-\bar{X})^2]\mathrm{E}[(Y-\bar{Y})^2]}} \\ &= \frac{\mathrm{E}[XY] - \mathrm{E}[X]\mathrm{E}[Y]}{\sqrt{\mathrm{E}[X^2] - \mathrm{E}[X]^2}\sqrt{\mathrm{E}[Y^2] - \mathrm{E}[Y]^2}} \end{aligned}$$

When applied to a sample (n values for each variable), a new formula is obtained by substituting the estimates of the covariance and variances :

$$\hat{r} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

This coefficient is sensitive to outliers in the data, and so it is always a good idea to visualize the data that generated the correlation measurement.

On the left of figure 2.1, there is a plot of samples of one variable versus another. The samples were generated such that $Y = 0.2 * X + \mathrm{U}(-0.01, 0.01)$ so that the variables are correlated. The straight line that is plotted is a simple linear regression between X and Y and fits the data well. On the right, an outlier was added to show that one point can drastically change the results. With the outlier, the correlation decreases from 0.943 to -0.103 and the regression doesn't fit the data well anymore.

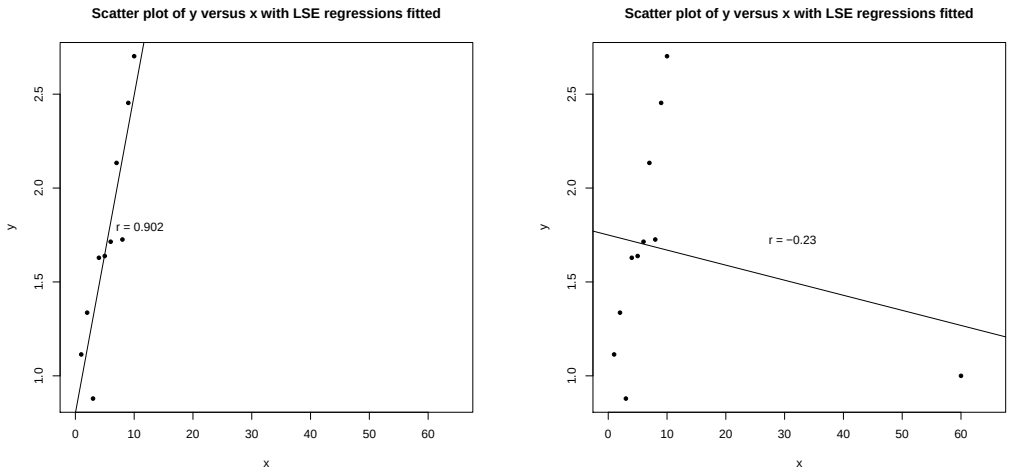


Figure 2.1: Influence of an outlier when computing the correlation between two correlated samples. The outlier makes the correlation low even though the rest of the data is correlated

This next figure (2.2) is similar to the previous one, except that the effect is reversed. On the left plot, the variables are uncorrelated with each other ($X, Y = \mathrm{U}(0, 5)$). On the right plot, the outlier increases the correlation from -0.175 to 0.734.

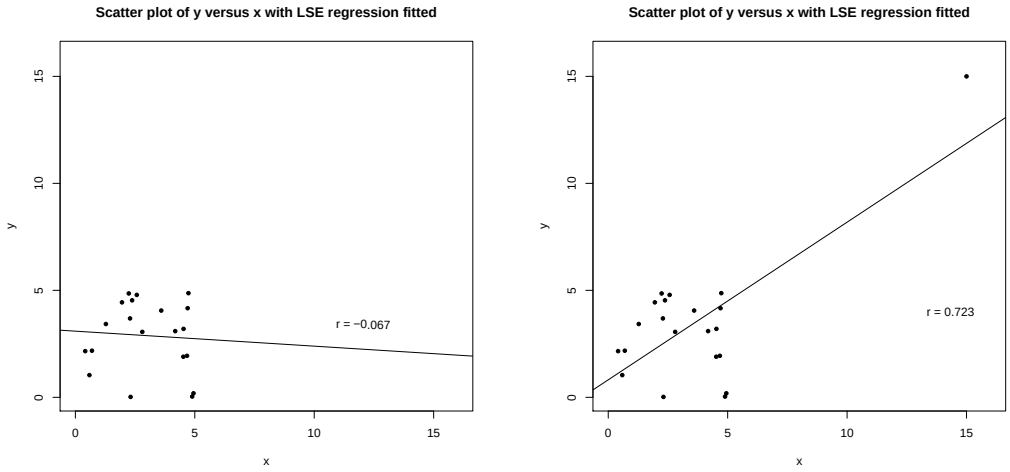


Figure 2.2: Influence of an outlier when computing the correlation between two uncorrelated samples. The outlier makes the correlation high even though the rest of the data is uncorrelated.

2.5.1 Fisher's z-transformation

Fisher's z-transformation (hyperbolic arctangent function) is a function that transforms an estimated correlation coefficient, which is a skewed student distributed random variable, into an approximately normally distributed random variable. In R, the function is called *atanh* and its inverse *tanh*. The expression of this transformation is :

$$\mathrm{F}(r) = \frac{1}{2}\ln\left(\frac{1+r}{1-r}\right) = \mathrm{atanh}(r) = z$$

The standard deviation of this new normal variable is :

$$\sigma = \frac{1}{\sqrt{n-3}}$$

with n being the number of samples used to compute the estimated correlation between the two random variables.[1]
Its inverse is :

$$\mathrm{F}^{-1}(z) = \frac{\exp(2z)-1}{\exp(2z)+1} = \tanh(z) = r$$

2.5.2 Statistical hypothesis tests for Pearson's correlation using Fisher's z-transformation

To carry out hypothesis tests and compute confidence intervals for Pearson's correlation, Fisher's z-transformation is used. That way the performed test is a z-test,

i.e. a test on a normally distributed variable.

Let $\hat{r}$ be a correlation computed with two sample sets of n values each (the observations) and let r be the real (unknown) correlation. $z = \mathrm{F}(\hat{r})$ is the Fisher-transformed correlation. The four elements needed for the test are :

- The null hypothesis $H_0 : r = 0$
- The alternative hypothesis $H_a : r \neq 0$
- The test statistic (a z-score) : $Z = \frac{z - \mathrm{F}(0)}{\sigma} = \frac{z}{\sigma}$
- The significance level : $\alpha = 0.05$

The standard deviation of z is known :

$$\sigma = \frac{1}{\sqrt{n-3}}$$

From it, a 95% confidence interval in the Fisher domain can be built :

$$\text{CI in Fisher domain :} \quad [z - z_{\alpha/2}\ \sigma,\ z + z_{\alpha/2}\ \sigma]$$

with $z_{\alpha/2} = z_{0.025} = 1.96$ being the threshold of the rejection region (if $|z| > z_{\alpha/2}$, then the null hypothesis is rejected in favor of the alternative hypothesis). This confidence interval is then transformed back using the inverse Fisher function :

$$\text{CI :} \quad [\mathrm{F}^{-1}(z - z_{\alpha/2}\ \sigma),\ \mathrm{F}^{-1}(z + z_{\alpha/2}\ \sigma)]$$

The associated p-value can also be computed using the observed value of the test statistic and the normal cumulative density distribution function Φ (in R, Φ is the function *pnorm*) :

$$z_{\text{obs}} = \frac{z - \mathrm{F}(0)}{\sigma} = z\sqrt{n-3}$$

$$\text{p-value} = 2\Phi(-|z_{\text{obs}}|)$$

In later sections of this work, the reported values for predictive performance are always the observed value of the correlation, the p-value associated and the 95% confidence interval when performing a statistical test.

2.6 Handling missing answers in the psychological questionnaire

As said in the previous section, two versions of the psychological survey exist. Only answers to the second version of the survey were given by Tools4Patient, meaning that the answers of some questions for patients that took the first survey are missing. Given the number of patients available in the dataset (151), simply deleting the 42 patients (about one third of the samples) with missing answers would drop a significant part of the data.

The process used to replace missing data is called imputation. The idea is to guess reasonable values to fill in the gaps. Many techniques exist and one of the simplest way to do it is to replace the missing values by the mean of the "not-missing" values independently for each of the features. Nonetheless, the dataset contains the answers to the follow-up study. Some of the patients took the second version of the survey even though they previously took the first version. If the survey is assumed stable (meaning that the answers of the patients don't change a lot between the times they take it), the answers to the follow-up study (in *mpsqData3*) can be used to reasonably replace the missing values of *mpsqData1*. To evaluate the stability of the survey, the correlation between answers for the beginning of the studies and the follow-up studies is computed. An histogram of the correlations is shown on figure 2.3. A table with the correlations is available in the Appendix.

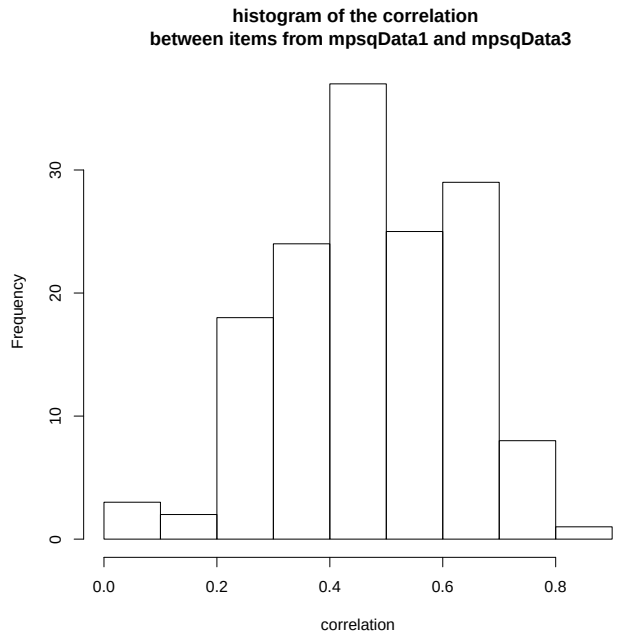


Figure 2.3: Histogram of the correlation for each item between mpsqData1 and mpsqData3. They are always positively correlated.

It is interesting to see on figure 2.3 that all the coefficients are positive. Moreover, a lot of the coefficients are high meaning it is reasonable to replace missing values in mpsqData with values from mpsqData3.

Chapter 3

Performance assessment of regression models

3.1 Optimistic bias

When attempting to build predictive models, performance assessment should be one of the tasks that come to mind. The performance of a model should describe how well a model does on unseen data (but evaluating the model on unseen data is impossible). A common pitfall in supervised machine learning is called the optimistic bias. This bias should usually be avoided. It arises when assessing the performance of a model on samples that were used to build it. The performance on already seen samples is very likely to be better than the performance on unseen ones. It is then absolutely necessary to avoid using the same samples to train (i.e. build) and to test the model. The choice of hyper-parameters[1] and dimensionality reduction must also be done separately as if you tune them using all the data, you effectively used all the data to build the model.

The simplest way to achieve this is to split the input and response data into two sets called the **train** set and the **test** set. If $x \in \mathbb{R}^{n \times p}$ and $y \in \mathbb{R}^n$ are respectively the input data and the response data, a possible split into train and test set could be $[x_{\text{train}} \in \mathbb{R}^{k \times p}; y_{\text{train}} \in \mathbb{R}^k]$, $[x_{\text{test}} \in \mathbb{R}^{(n-k) \times p}; y_{\text{test}} \in \mathbb{R}^{(n-k)}]$ with $0 < k < n$. The names are quite revealing, the train set is used to build the model and the test set is used to assess its performance. The performance is computed by comparing (with a performance metric) the actual response values of the test set and the response values predicted by the model. Such a scheme is already better than simply using the training samples to test the performance. However this scheme is

[1] Each model has hyper-parameters, i.e. parameters tuning the learning algorithm. The learned model is influenced a lot by these hyper-parameters.

still deficient in the sense that only evaluating one split of the data means that the variance of the assessed performance is very high.

To alleviate this problem, a common solution is to evaluate the performance for multiple train/test splits and to aggregate those results in order to have an estimation of the performance with less variance. The standard way to do so is cross validation.

3.2 Cross validation

To evaluate the performance of our models, a cross-validation scheme is needed to avoid having an optimistic bias on the measured performance. There are several kinds of cross validation, the two of them considered in this work being k-fold cross validation and Monte Carlo cross validation.

3.2.1 K-fold cross validation

Let $x \in \mathbb{R}^{n \times p}$ be the input data (n samples of p features) and $y \in \mathbb{R}^{n}$ the corresponding response data (for the n input samples). From this data, it is possible to create folds, i.e. different splits of the data. At the start of a k-fold cross validation, k folds of equal size are constructed (x and y are split, keeping the corresponding values between the two matrices). Each fold is used once as the test set during the validation. They each consist of $\frac{100}{k}\%$ of the data (10% if $k = 10$, 20% if $k = 5$, etc) and don't overlap, meaning that a sample will only be in one fold.

Here is an example of folds for $k = 3$, $n = 9$ and $p = 1$. Let

$$x = \begin{pmatrix} x_{1*} \\ x_{2*} \\ \vdots \\ x_{9*} \end{pmatrix} \text{ and } y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_9 \end{pmatrix}$$

From this data, a split into 3 folds could be :

$$S_1 = \begin{pmatrix} x_{1*} \\ x_{5*} \\ x_{9*} \end{pmatrix} ; \begin{pmatrix} y_1 \\ y_5 \\ y_9 \end{pmatrix}, \; S_2 = \begin{pmatrix} x_{2*} \\ x_{4*} \\ x_{6*} \end{pmatrix} ; \begin{pmatrix} y_2 \\ y_4 \\ y_6 \end{pmatrix} \text{ and } S_3 = \begin{pmatrix} x_{3*} \\ x_{7*} \\ x_{8*} \end{pmatrix} ; \begin{pmatrix} y_3 \\ y_7 \\ y_8 \end{pmatrix}$$

The correspondence between the input data and the response data is kept and each fold consists of $\frac{100}{k} = 33\%$ of the data.

Here is how a k-fold cross validation is done in practice :

1. Randomly split the data into k folds of equal size : $S = (S_1, S_2, ..., S_k)$; Initialize i to 1;
2. Select the i^{th} fold S_i and pick it as a test set. The rest of the folds ($S \backslash \{S_i\}$) form the train set;
3. Train a model using the train set;
4. Predict responses on S_i (the test set) using the learned model;
5. Report the performance using a performance metric on the actual response values (from S_i) and the predictions;
6. Increment i and repeat from step 2 (stop if $i > k$);

The performance is then computed using the reported values from step 5 and an aggregation method. Three aggregation methods are considered in this work : a simple mean of the reported performances, the mean of the reported performances in the Fisher domain as well as a method called pooling (those last two are explained later).

Once the performance is computed, a hypothesis test is carried out to compute a confidence interval and a p-value.

3.2.2 Monte Carlo cross validation

The Monte Carlo cross validation is very similar to the k-fold cross validation in the sense that it also uses folds. The main difference is that the folds can overlap themselves and their number can, in theory, be infinite. The goal is to have many more splits (i.e. models built) than in a k-fold cross validation. For a Monte Carlo cross validation, instead of choosing the number of folds k, simply choose the number of iterations (i.e. folds) m and the fraction of data (typically 10%) that each fold must consist of.

Here is how it works :

1. Initialize i to 1;
2. Randomly select without replacement 10% of the data set to be the test set. The remaining data points form the train set;
3. Train a model using the train set;
4. Predict responses using the learned model on the test set;

5. Report the performance using a performance metric on the actual response values (from the test set) and the predictions;

6. Increment i and repeat from step 2 (until $i > m$);

The idea behind the usage of this cross-validation is that when performing a k-fold cross validation, only k possible splits of the data are seen. Monte Carlo cross validation tries to be more exhaustive by attempting a higher number of splits.

3.3 Performance metric used in the cross validation

At step 5 of either the k-fold cross validation or the Monte Carlo cross validation, when the performance of the fold is computed, a choice has to be made : the performance metric. The usual choice in case of regression is the Mean Squared Error (MSE) between the predictions and the actual values (or its square root, RMSE). For n predictions $\hat{y}$ and their corresponding actual values y, the MSE is :

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

But in the case of this master thesis, **Pearson's correlation coefficient** (also between the predictions and the actual response values) was chosen. It is much more significant for Tools4Patient because studies tend to change : the disease studied don't inflict the same pain and the way the pain data is collected changes as well. The goal is to have a performance metric that is coherent between different studies. Pearson's correlation coefficient fits this purpose very well while MSE will differ for each study. Also, it is not expected to have nearly perfect predictions for model built on our data as the number of samples is low and the data inherently noisy (the answers were produced by humans that are influenced by a lot more variables than the one in our data), so using correlation enables us to have a measure bounded between -1 and 1 as well as having a statistical test to assess whether the results are indeed significant.

3.4 Aggregation methods for correlation coefficients

At the end of a cross validation, an overall performance is expected. To compute it, results from every iteration (i.e. fold) are aggregated. Three aggregation methods are considered : the simple average of the correlation coefficients, the average of the

correlation coefficients in the Fisher domain and the pooling method. The methods are theoretically compared and a simulation is used to validate the predicted outcome for each method and to help choose the one used in the experimental results.

3.4.1 Estimating correlation using the average of multiple correlation coefficients

Averaging correlation coefficients is as simple as it seems. From several coefficients, simply average them and an overall performance for the cross validation is obtained. Let c be a vector of correlation coefficients computed during a cross validation of n iterations :

$$c = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix}$$

The average is simply :

$$\hat{r} = \frac{1}{n} \sum_{i=1}^{n} c_i$$

However, it was found in [26] that there is a better way to average correlations. Indeed by averaging correlations right away, a negative bias is introduced (leading to underestimation) because the distribution of the correlation coefficients is skewed. The authors confirmed this after running a simulation. First, they generate a vector y of pseudo-random data points. From these points, they create another vector $\hat{y}$ so that they are correlated with each other (for different values of correlation going from 0 to 0.9). Monte Carlo simulations for several sample sizes (from 10 to 30) are then run on these points, providing correlation coefficients. The first vector y represents values measured from the real world and $\hat{y}$ represents their predictions. The coefficients are then averaged and compared to the expected correlation. For each experiment, a negative bias was observed.

To avoid this, it is recommended to average the correlation coefficients in the Fisher domain. It means transforming them using the *atanh* function, average them and back-transform the result with *tanh*, giving this formula :

$$\hat{r} = \tanh\left(\frac{1}{n} \sum_{i=1}^{n} \operatorname{atanh}(c_i)\right)$$

Nevertheless the bias with this method is not null either, only better. It is observed using the same simulation that the back-transformed correlation has a positive bias. This bias is lower in absolute value than the bias observed when using the simple average and it is thus preferable to use a Fisher transformation when performing an average of correlation coefficients.

Here is an example to show the difference between the two methods. Let's consider a cross validation of 5 iterations. For each of them, 5 predictions are made (of range [1,10]) :

Iteration	1	2	3	4	5
Prediction	3 4 4 2 1	8 2 9 6 5	8 7 9 4 8	5 4 2 7 8	7 3 8 6 3
Actual	3 5 3 3 1	7 3 9 5 6	6 7 7 7 9	6 3 2 9 6	9 5 7 5 6
Correlation	0.81	0.94	0.095	0.82	0.6

Table 3.1: Example of predictions output from a cross validation

From table 3.1, a correlation is computed by using the simple average of correlation coefficients from each iteration :

$$\hat{r} = \frac{0.81 + 0.94 + 0.095 + 0.82 + 0.6}{5} = 0.65$$

or by using the average in the Fisher domain :

$$\tanh\left(\frac{\text{atanh}(0.81) + \text{atanh}(0.94) + \text{atanh}(0.095) + \text{atanh}(0.82) + \text{atanh}(0.6)}{5}\right) = 0.74$$

As expected, the computed correlations differ a lot and the one computed using the average in the Fisher domain is higher than the one computed with the simple average. The problem here is that the expected correlation isn't known as it is an arbitrary example, meaning that the best method cannot be determined with such an example. Moreover the example only contains a few predictions, leading into a poorly estimated correlation. The point is just to illustrate the difference between the two methods. In a **later section**, a simulation was put in place to determine the best way to aggregate results from a cross validation.

3.4.2 Estimating correlation using pooling

Another way of aggregating results from a cross validation is called *pooling*. Instead of recording performance values for each iteration, the predictions and the paired actual values are kept until the end of the cross validation. The performance metric

is used only once, at the end, on all the recorded predictions.
For example, using results from table 3.1 and the pooling method, the overall correlation coefficient is computed between the two vectors in table 3.2.

Prediction	3	4	4	2	1	8	2	9	6	5	8	7	9	4	8	5	4	2	7	8	7	3	8	6	3
Actual	3	5	3	3	1	7	3	9	5	6	6	7	7	7	9	6	3	2	9	6	9	5	7	5	6

Table 3.2: Results of a cross validation of 5 iterations and 5 predictions per iteration when using pooling as an aggregation method

The computation gives a correlation of 0.81. This is yet another value that is quite far from the other two. The pooling method is also assessed in the next section.

3.4.3 Comparison of the three methods of aggregation using simulation

This section concerns the comparison of the considered aggregation methods (in order to compute an overall correlation coefficient) of results from a cross validation. First, the average in the Fisher domain and the pooling method will be compared theoretically from a variance standpoint. The simple average and the average in the Fisher domain are also compared to one another but from a bias standpoint. Then an experimental comparison between the three methods is presented. For the sake of simplicity, and as the statistical test for the pooling and the mean in the Fisher domain is well defined in this case, the theoretical analysis and the simulation are done in the context of a k-fold cross validation. Similar results are expected with a Monte Carlo cross validation, but the statistical testing differs a little since the samples may be used several times during the validation.

In theory

This theoretical reasoning shows that the variance (of the correlation coefficient) when using the average in the Fisher domain is greater (resulting in a wider confidence interval) than the one when using the pooling method.

Let $a \in \mathbb{R}^{\frac{n}{k} \times k}$ be k sets of predictions of $\frac{n}{k}$ values from a k-fold cross validation. The columns of a are the predictions of one iteration. Let $b \in \mathbb{R}^{\frac{n}{k} \times k}$ be the actual response values (the columns of a correspond to the columns of b).
First case : the pooling method. The correlation is directly computed using the n predictions and n actual values as illustrated in table 3.3.

Prediction	$a_{11}\ a_{21}\ ...\ a_{\frac{n}{k}1}\ a_{12}\ ...\ ...\ a_{\frac{n}{k}k}$
Actual	$b_{11}\ b_{21}\ ...\ b_{\frac{n}{k}1}\ b_{12}\ ...\ ...\ b_{\frac{n}{k}k}$

Table 3.3: Data used to compute the overall correlation coefficient with pooling as aggregation method in a k-fold cross validation

The correlation is then transformed using Fisher's z-transformation and the confidence interval and p-value can be computed. The z-score is obtained using the variance of the Fisher transformed variable which depends on the number of samples used to compute the correlation (as seen in **a previous section**). The formula of the variance is thus :

$$\sigma^2 = \frac{1}{n-3}$$

Second case : averaging in the Fisher domain. k correlations are computed from each column of a and b as illustrated in table 3.4.

Correlation coefficients	$\text{Cor}(a_{*1}, b_{*1})\ \text{Cor}(a_{*2}, b_{*2})\ ...\ \text{Cor}(a_{*k}, b_{*k})$

Table 3.4: Data used to compute the overall correlation coefficient with mean in the Fisher domain as an aggregation method in a k-fold cross validation

Then they are all transformed using the *atanh* function : $[c_1, c_2, ..., c_k]$. The average is performed in the Fisher domain and the confidence interval and p-value can be computed. The problem is that the formula for the variance has changed since the result comes from an average. The variance of a linear combination of k variables is computed like this :

$$\text{Var}\left(\sum_{i=1}^{k} v_i X_i\right) = \sum_{i=1}^{k} v_i^2 \text{Var}(X_i) + 2 \sum_{1 \le i < j \le k} v_i v_j \text{Cov}(X_i, X_j)$$

In this case the variance of the average of transformed correlation coefficients c_i is at least :

$$\begin{aligned}
\sigma^2 &= \text{Var}\left(\sum_{i=1}^{k} \frac{1}{k} c_i\right) \\
&= \sum_{i=1}^{k} \frac{1}{k^2} \text{Var}(c_i) + 2 \sum_{1 \le i < j \le k} \frac{1}{k}\frac{1}{k} \text{Cov}(c_i, c_j) \\
&\ge \sum_{i=1}^{k} \frac{1}{k^2} \frac{1}{\frac{n}{k} - 3} = \frac{k}{k^2(\frac{n}{k} - 3)} = \frac{1}{k(\frac{n}{k} - 3)} \qquad \text{As it is supposed } \sum_{1 \le i < j \le k} \text{Cov}(c_i, c_j) \ge 0
\end{aligned}$$

The value of the variance is uncertain because the covariances between all transformed correlations are unknown. That is why we made the assumption that the sum of these covariances is positive. It is a reasonable assumption because the correlations all come from predictions made with the same model. One could expect the correlations to be positively correlated, meaning that the term

$$2 \sum_{1 \leq i < j \leq k} \frac{1}{k} \frac{1}{k} \mathrm{Cov}(c_i, c_j)$$

is expected to be positive.

Third case : the simple average. Since the Fisher transformation is approximately linear when the input isn't too close to -1 or 1 (especially near 0, see figure 3.1), the correlation should not be different from the simple average when using the average in the Fisher domain for correlations close to 0.

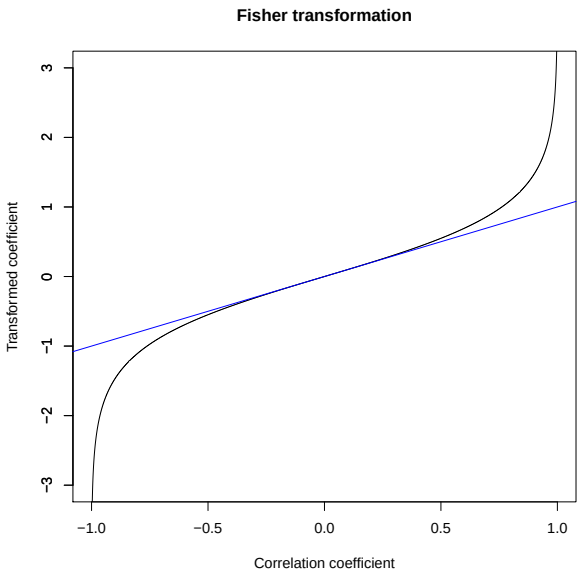


Figure 3.1: Fisher transformation. The black line is the actual transformation, the blue one is its linear approximation computed at the origin.

Indeed, if the Fisher transformation is linear and is of the form $F(x) = ux + v$, then the average of transformed correlation is equal to the transformation of the

average of correlations :

$$\begin{aligned}\text{Average in Fisher domain} &= \frac{1}{k}\sum_{i=1}^{k} F\left(\text{Cor}(a_i, b_i)\right) = \frac{1}{k}\sum_{i=1}^{k}\left(u\text{Cor}(a_i, b_i) + v\right) \\ &= v + u\left(\frac{1}{k}\sum_{i=1}^{k}\text{Cor}(a_i, b_i)\right) \\ \text{Simple average transformed} &= F\left(\frac{1}{k}\sum_{i=1}^{k}\text{Cor}(a_i, b_i)\right) = u\left(\frac{1}{k}\sum_{i=1}^{k}\text{Cor}(a_i, b_i)\right) + v\end{aligned}$$

If the linearity assumption holds (i.e. the correlation is near 0), they are the same and compute the same correlation coefficient. The correlation coefficients will differ from one another when the expected correlation gets closer to 1 or -1. The probability density distribution of a Fisher transformed average of correlation coefficients is unknown to us, meaning that no assumption about the variance when using the simple average can be made.

If $n = 150$ and $k = 10$, the variance (in the Fisher domain) when using the pooling method should be 18% lower than when using the average in the Fisher domain. If k the number of folds of the k-fold cross validation increases, the variance observed when using the average in the Fisher domain increases too. From a variance standpoint, the pooling method is better than the mean in the Fisher domain method as the variance of the latter method is greater.

In practice

To confirm our theory and the choice between pooling and averaging (simple or in Fisher domain), we created a simulation of our own. It is divided in two parts. First the behavior of each method when an expected correlation of 0.7 is assessed to see what happens when there is correlation. Then it is repeated for an expected correlation of 0 to confirm that the p-values are well computed.

Expected correlation of 0.7 A vector $x \in \mathbb{R}^{150}$ of 150 random data points (normally distributed) is generated. The following loop is then repeated $J = 10000$ times :

1. Generate a vector $\hat{y}$ from $x \in \mathbb{R}^{150}$ so that it has on average a correlation of $r = 0.7$ using this formula : $\hat{y} = rx + \sqrt{1 - r^2}x'$ with x' being a random normal vector uncorrelated with x and with same mean and variance.

2. Imitate a 10-fold cross validation with x being the actual values and $\hat{y}$ representing the predictions

3. Compute the correlation coefficient using the simple average, the average in Fisher domain and the pooling method

From the correlation coefficients computed with the different methods, the mean and the variance are computed. They can be observed in figures 3.2, 3.3 and 3.4. On each graph, the red line is the expected correlation (0.7) and the blue line is the mean of the correlation coefficients obtained during the simulation. The estimated distribution of these coefficients is displayed.

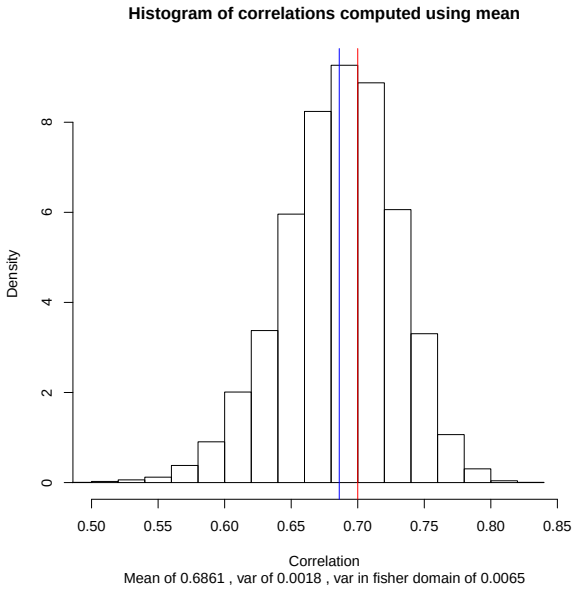


Figure 3.2: Correlation coefficients computed using the simple average (expected correlation is 0.7). The expected negative bias is present.

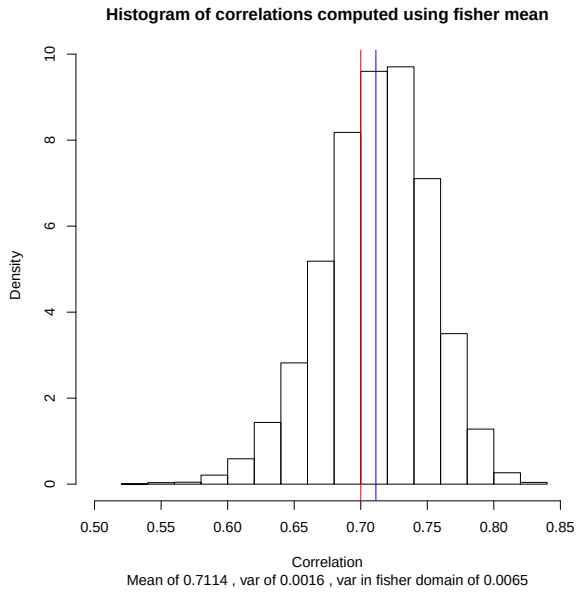


Figure 3.3: Correlation coefficients computed using the average in Fisher domain (expected correlation is 0.7). The expected positive bias is present and is less in absolute value than the one using the simple average.

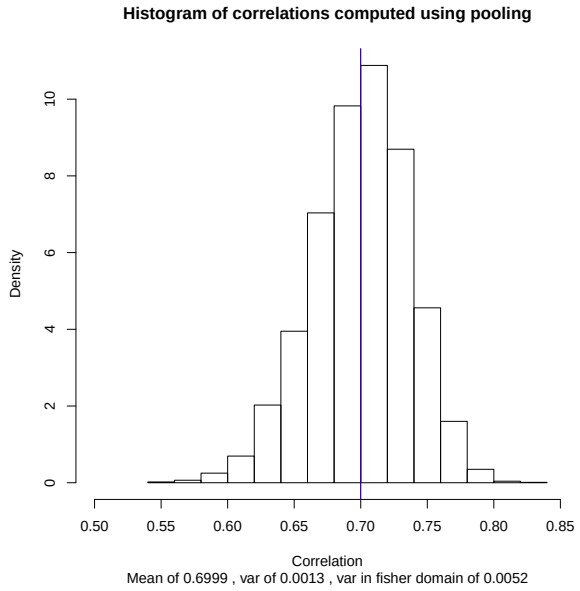


Figure 3.4: Correlation coefficients computed using the pooling method (expected correlation is 0.7). The bias is almost null and the variance is 20% less than with the two other methods.

The expected biases for the simple average and the average in the Fisher domain are observed. As for the pooling method, the bias is near 0. This makes sense because the pooling method uses all the samples gathered during the k-fold cross validation at once to compute the correlation and so no bias is introduced by the aggregation method. Since on average $\hat{y}$ has a correlation of 0.7 with x and each correlation computed is the real correlation between $\hat{y}$ and x, the bias is null.

Around 0.7, the Fisher transformation cannot be assumed linear. The correlations computed with the simple average and the average in the Fisher domain are in consequence different. The pooling method leads to a variance that is 20% lower than with the two other methods. This result is close to the theoretical value of 18% and can be explained because of the approximation on the covariances.

Expected correlation of 0 The simulation in this case is exactly the same as before, but with $r = 0$. Figures 3.5, 3.6 and 3.7 are similar to the ones of the first part of the simulation. The distribution of the correlation coefficients are shown as well as the expected correlation (red) and the observed mean correlation (blue).

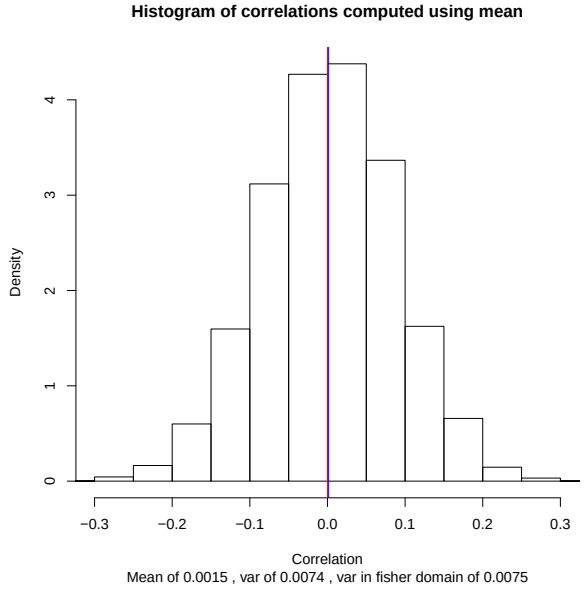


Figure 3.5: Correlation coefficients computed using the average (expected correlation is 0). The mean correlation is 0.0015 and the variance (in the Fisher domain) 0.0075.

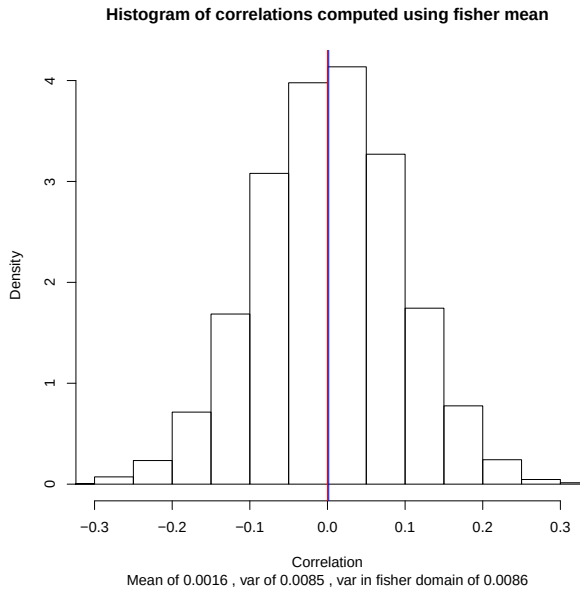


Figure 3.6: Correlation coefficients computed using average in Fisher domain (expected correlation is 0). The mean correlation is 0.0016 and the variance 0.0086.

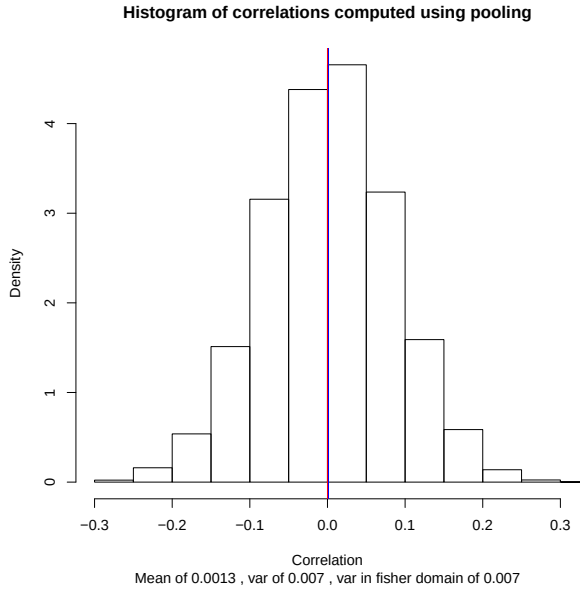


Figure 3.7: Correlation coefficients computed using the pooling method (expected correlation is 0). The mean correlation is 0.0013 and the variance is 0.007.

Here for an expected correlation of 0, the variance is 18% lower than with

the average in the Fisher domain when using the pooling method. The theory holds. The biases for each method aren't significantly different from one another, especially between the simple average and the average in the Fisher domain (the assumption about the Fisher transformation seems to be good). The same result was observed in the simulation of [26].

Under the null hypothesis (i.e. when the null hypothesis is true : $H_0 : r = 0$), another property must be observed : the distribution of the p-values should be uniformly distributed (as stated in **this section**).

As expected, for both the average in the Fisher domain and the pooling method, the estimated distribution is approximately uniform (figures 3.8 and 3.9). The p-values for the simple average cannot be computed because of the unknown distribution of the average of correlation coefficients.

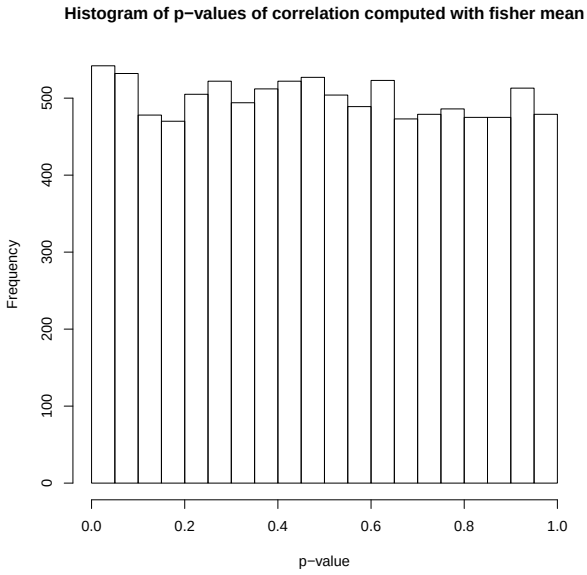


Figure 3.8: p-values computed when using average in the Fisher domain to compute correlation coefficients (expected correlation is 0). As expected, the estimated distribution is approximately uniform.

Using the uniformity test defined in **this section**, the hypothesis that the distribution of the p-values computed in the case of the average in the Fisher domain is uniform is not rejected. The χ^2 test statistic has a value of 19.932 and there are 19 degrees of freedom. The p-value is 0.399 which is well above the value of 0.05 needed to reject the uniformity hypothesis.

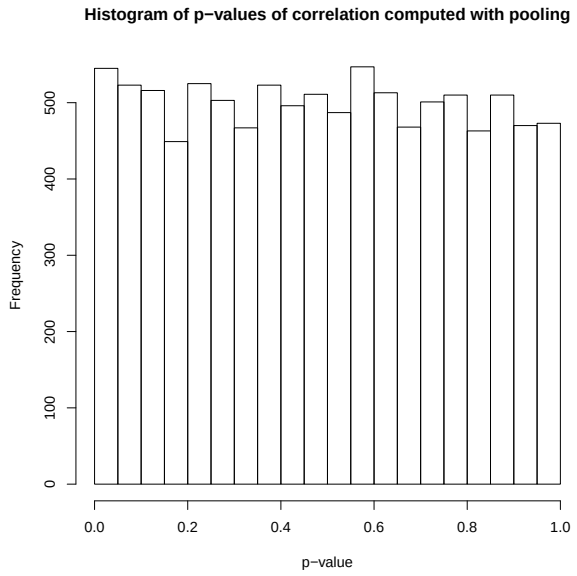


Figure 3.9: p-values computed when using the pooling method to compute correlation coefficients (expected correlation is 0). As expected, the estimated distribution is approximately uniform.

As for the pooling method, the χ^2 test statistic has a value of 29.14 and there are also 19 degrees of freedom. The p-value is 0.064 which is a little above the value of 0.05 needed to reject the uniformity hypothesis.

Since the mean correlation obtained using the pooling method is a lot less biased than using the average in the Fisher domain, and since the variance is lower, the performance assessment is done with the pooling method.

3.5 Hypothesis testing when Monte Carlo cross validation and pooling are used

3.5.1 Modification to the test statistic of the hypothesis test

The choice between a k-fold cross validation and a Monte Carlo cross validation is somewhat arbitrary, we chose to use it for our experiments (on the questionnaire and pain data) as it is more exhaustive (more number of splits are assessed). To make sure that the chosen validation scheme doesn't give different results from the k-fold cross validation, the simulation is done again but with an imitation of a

Monte Carlo cross validation. The results are displayed on figures 6.1, 6.2, 6.3, 6.4, 6.5, 6.6, 6.7 and 6.8 in the appendix.

The results are similar to the ones found in the previous simulation, at least for the computation of the correlation coefficients. The same observations about the biases can be done. However, for the computation of the p-values, things change a little. It is said in the previous section that hypothesis testing in the case of a Monte Carlo cross validation is different. Indeed, the test statistic must be adapted because samples can be used more than once. Only the pooling method and the average in the Fisher domain are considered for the same reason as before : the distribution of a Fisher transformed average of correlation coefficients is unknown so p-values when simply averaging the correlation coefficients can't be computed.

For the pooling method, a quite simple correction to the test can be made. As stated in **this section**, the standard deviation of the transformed correlation z is

$$\sigma = \frac{1}{\sqrt{n-3}}$$

with n values used to compute the correlation. In a k-fold cross validation with pooling as aggregation method, the correlation is computed with all the response values (n is equal to the length of the response data set). Even if k changes, the number of samples used does not change. However in a Monte Carlo cross validation, the correlation is computed with a number of values depending on the number of iterations. It is actually equal to the number of iterations times the chosen size of the test set. It is usually more than the actual number of samples as samples can be seen multiple times. The resulting standard deviation would be underestimated if this number was used. Moreover, any model could become significant (the paired statistical test could report a correlation significantly different from 0) if the number of iterations becomes high enough. Instead, n should be equal to the length of the response data set. This correction is the only modification brought to the statistical test. The next is section a theoretical reasoning dedicated to this correction.

As for the average in the Fisher domain, the variance must also be modified. Instead of estimating it as previously done in the case of a k-fold cross validation, the observed variance is adapted and used as [21] proposes. Let $\frac{|\text{test}|}{|\text{train}|}$ be the ratio between the size of the train set and the test set for the Monte Carlo simulation (typically 0.1). Let m be the number of iterations of the Monte Carlo cross validation. Let $z_i \; \forall i \in [1..m]$ be the m Fisher transformed correlation coefficients obtained during the validation and let $\bar{z}$ be their average. The variance used is

then simply

$$\sigma^2 = \left(\frac{1}{m} + \frac{|\text{test}|}{|\text{train}|}\right)\left(\frac{1}{m-1}\sum_{i=1}^{m}(z_i - \bar{z})^2\right)$$

However, this correction is said to be gross by the authors of [21] but better than simply using the observed variance of the performance between the folds of the monte carlo cross validation.

Only the p-values computed using the pooling method under the null hypothesis (expected correlation of 0) are uniformly distributed. The χ^2 test for uniformity yields a test statistic of 15.5. The number of degrees of freedom is 19, giving a p-value of 0.69 which is a lot more than 0.05. The uniformity hypothesis cannot be rejected. As for the average in the Fisher domain, the χ^2 test statistic is 94.98 and the number of degrees of freedom is the same. The p-value is 4.29 10^{-12}, well under the 0.05. The alternative hypothesis is accepted : the p-values aren't uniformly distributed.

The pooling method is the only one leading to well computed p-values. Its variance and bias are also lower. For our later experiments, Monte Carlo cross validation with pooling as aggregation method is used to assess performance.

3.5.2 Theoretical reasoning behind the correction of variance when using pooling

The following reasoning supporting this statement is the outcome of a discussion with our supervisor Samuel Branders. It shows that this approximate correction is reasonable to compute the performance of a model if the performance metric is the correlation between the prediction values and the response values.

First, some notations :

- x : data set of size n, answers to a psychological questionnaire from n patients
- y : set of n response values, reported pain of the patients
- $c(y, y)$: concatenation of y and y
- $\hat{y}$: set of n predictions of response values
- p_i : prediction for a patient i
- f : a model
- $f(x_{i*})$: the prediction for the patient x_{i*}

- $f(x)$: the predictions for all the patients of x
- m : number of iteration in a Monte Carlo cross validation

The reasoning starts with a property of the Monte Carlo cross validation (1), then the behavior of the variance under the null hypothesis is analyzed (2) and finally a justification of why the approximate correction is conservative is presented (3).

(1) Monte Carlo cross validation with pooling is asymptotically (i.e. when the number of iterations tends to infinity) unbiased if the model is stable.

Let f be a stable model, always predicting p_i for a patient x_{i*}. Its performance is assessed through a Monte Carlo cross validation with pooling (on x, the input data and y, the response data). When the number of iterations m tends to infinity, the estimated correlation tends to $\text{Cor}(y, \hat{y})$, with $\hat{y} = [p_1, p_2, ..., p_n]$. Indeed, when m tends to infinity, the proportion of each patient in the pool of predictions converges to $\frac{1}{n}$. Also, $\text{Cor}(y, \hat{y}) = \text{Cor}(c(y, y, ...), c(\hat{y}, \hat{y}, ...))$.

(2) Let $\widetilde{D} = (\widetilde{x}, \widetilde{y})$ be a population of patients. The available data set $D = (x, y)$ is a random sample of $\widetilde{D}$. If the total number of samples seen during the validation of a model f is used (m times the size of the test set), the performance estimated is the one for D. The more iterations are done, the lower the variance is, which isn't really interesting. What is really interesting are the bounds on the performance on the population $\widetilde{D}$.

Under the null hypothesis (i.e. there is no correlation between the predictions and the actual values, meaning that the prediction are random), a model f_r is assumed to be a random model. It is tested on population D with a Monte Carlo cross validation. The worst case for the variance is when f_r is stable ($f_r(x_{i*})$ is always p_i). If it isn't the case, f_r changes between each iterations and the correlation coefficients that are computed by several Monte Carlo cross validations are distributed around 0 and converge towards 0 when the number of iterations m tends to infinity. If the model is stable (with (1)), the correlation coefficients obtained by the validations are always $\text{Cor}(f_r(x), y)$. It is like setting the correlation before the cross validation. The resulting variance is greater.
So for the worst case, thanks to (1) the correlation can be estimated through a Monte Carlo cross validation and tend toward $\text{Cor}(f_r(x), y)$. It is the standard case of the correlation. The Fisher transformed correlation approximately follows (under the null hypothesis) a normal distribution $\mathcal{N}\left(0, \frac{1}{n-3}\right)$ and n is the size of the data set.

(3) If f is not random :

(a) Let f be a model that wasn't learned on D. In this case Monte Carlo cross validation isn't needed, the correlation can directly be computed as $\mathrm{Cor}(f(x), y)$ with its confidence interval and p-value. If the Monte Carlo cross validation is used, the computed value converges on the same result. Since the model isn't build on D, it doesn't change between the iterations of the cross validation. With (1), if enough iterations are done, the correlation converges to $\mathrm{Cor}(f(x), y)$. If n is considered for the test statistic instead of m times the size of the test set, the same confidence interval and p-value are obtained as when directly computing the correlation. The correlation obtained by Monte Carlo cross validation is asymptotically equivalent to $\mathrm{Cor}(f(x), y)$.

(b) Let f_D be a model learned on the whole data set D. The goal is to assess the performance on $\widetilde{D}$. It is reasonable to assume that this model f_D learned on D is the best model learned on a subset of the data D when tested on D.

In the ideal case during the cross validation, model f_D is learned at each iteration. This is the same case as (3a).

During a Monte Carlo cross validation, models are learned on (typically) 90% of the data D. In average, one can assume that the performances of such models are lower than the one of f_D. The method is thus conservative and is a reasonable approximation to assess the performance of a model.

Chapter 4

Dimensionality reduction techniques

To reduce the dimension of some data, either a subset of features is selected as is or the features are combined into a few new features.

4.1 Feature selection

Feature selection is the task of choosing a relevant subset of features among the ones available. Doing so is desirable for multiple reasons :

- To give a good insight about the importance of the different features.
- If a limited number of features is selected (2-3), the data can be visualized and it might be easier to understand them.
- To mitigate the impact of the curse of dimensionality that hinders some learning algorithms.
- To avoid overfitting models.
- Models learned on a subset of features might be easier to understand and interpret.
- Reducing the number of features may be necessary from a computational standpoint as building a model on a excessive number of features may take too much time or memory.

In the next sections, three feature selection frameworks are discussed : wrappers, filters and embedded methods.[10] The study of the stability of the selected features is also considered.

4.1.1 Wrappers

The principle of the wrapper method for feature selection is quite simple. A subset is said to be better than other ones if the performance of a model built with this subset is higher than for other subsets. This method is interesting because it chooses features that are helpful for the model considered (for example if a linear model is used, it should not select features that are in a non-linear relationship with the response). Figure 4.1 illustrates how wrappers works.

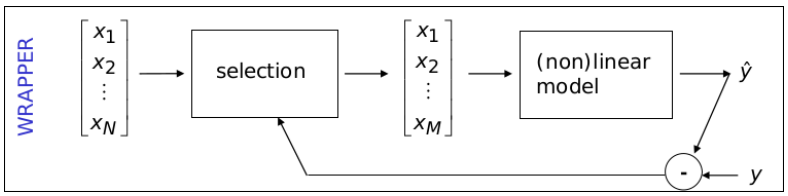


Figure 4.1: Illustration of wrapper feature selection [17]

The biggest downside of the wrapper method is that a subset exploration strategy needs to be setup, as exploring all possible subsets is intractable : indeed for input data with D features, there are 2^D possible subsets of features.

The most often used methods are either greedy or randomized methods :

- Forward search (greedy) : start with one feature, iteratively add the feature increasing the most the performance of the learned model until enough features are selected or until the performance doesn't increase anymore.
- Backward search (greedy) : start with the whole set of features and iteratively remove the least promising feature (the one that either increases the performance or decreases it the least) until enough features are selected or until the performance drop is too high.
- Forward-backward search (greedy) : Combination of both forward and backward search.
- Genetic algorithms (randomized)[11]

This framework is not used in this work for two reasons :

1. There are 150 available features in the data set. Wrapper methods are computationally intensive as it needs to train and assess the performance of models built on a lot of subsets of the features of the data.
2. Genetic algorithms are randomized and raise questions about the stability and the interpretability of the selected features which is crucial in this work.

4.1.2 Filters

In the filter framework, the criterion for choosing a subset is a statistical measure of the existence of a relationship between the input data and the response. Compared to wrappers, they are generally faster and provide a generic subset of features instead of a tuned one for a specific model. Also, as an information criterion is used to choose features, the resulting subset is more explainable. Figure 4.2 illustrates how filters work.

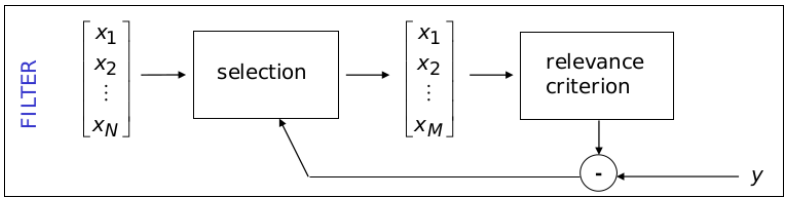


Figure 4.2: Illustration of filter feature selection [17]

There are two types of filters :

- Univariate filters : The statistical measure is done between only one feature and the response. With such methods, there is no need to search inside the subset space of the features. They only assess whether a feature should be selected or not.

- Multivariate filters : The statistical measure is done between a subset of features and the response. Some features might be useless on their own but with others, they might bring a lot of information. When using multivariate filters, redundancy is minimized and relevance is maximized. It means that the best subset gives a lot of information and its features give as little redundant information as possible. The same problem as the wrapper framework concerning the number of possible subsets also arises. The same search strategies can be used to overcome this problem.

The downsides of filters, is that they are not aware in any way of the model that is used for prediction. For a linear model it could select features non-linearly related with the response. This method selects features that are deemed relevant by the filter criterion but not the features that optimize the performance criterion of the model used for predictions. There are other downsides for univariate information criteria : they can fail to select important variables when multiple features are needed to predict the response. Also, univariate methods may select redundant features which is not desirable when a minimal number of features is wanted.

Pearson's correlation coefficient as filter criterion

As stated previously in this document, Pearson's correlation coefficient measures the linear dependence between two variables. It is thus suitable for feature selection. The correlation between each feature and the response is computed. The features are then sorted decreasingly according to the correlation with the response. Selecting features comes down to simply selecting the first features from the sorted ones.

The issue with this criterion is that if the selected features are used with a non-linear model, it may discard very useful features that are in a non-linear relation with the response. It is also an univariate criterion so it suffers from the common downsides of univariate information criteria. Nonetheless, it is a fast and interpretable way to select features. Moreover, the purpose of feature selection isn't only prediction, it is also useful to analyze data.

Pearson's correlation coefficient paired with prior knowledge

It is also possible to use prior knowledge when selecting features. For example, the questionnaire data provided by Tools4Patient contains answers to psychological questions. Each question belongs to one facet (each facet assesses one trait of personality). Intuitively, choosing to pick at least one question per facet can be a good choice of feature selection method if the assumption that the facets are well built is true. To choose the "best" question for each facet, Pearson's correlation can be used as described previously.

4.1.3 Embedded methods

Embedded methods, as their name states, embed the features selection inside a learning algorithm. The idea behind those methods is that letting the model decide the features that should be used may lead to better results than separating the selection and the learning of a model. Such methods include Lasso and Elastic net.

Lasso

As seen in a **previous section**, the Lasso regression model, as well as adding regularization compared to the LSE regression model, has the property of producing sparse models : models where a lot of features are unused. The number of unused features depends on the value of the regularization parameter λ (the higher it is, the less features are selected). This means that even if Lasso is a regression model, it can be used to perform feature selection. Several models are learned for different

values of λ and they each have different number of weights set to zero. To choose how many features the technique should select, simply choose a value for λ.

Elastic Net

The Elastic Net, also seen in a **previous section** is a combination of the Lasso and Ridge regression that usually performs better for feature selection because it tends to select groups of correlated features together whereas Lasso only selects one of the features randomly. It also has the parameter λ to tune the compromise between the number of selected features and the MSE. The number of selected features is set the same way as with Lasso.

4.1.4 Stability of the feature selection

When performing feature selection, one should evaluate the stability of the selection. To do so, a Monte Carlo cross validation scheme is used. Of course, instead of assessing the performance of a model, the stability of the feature selection technique is computed. The cross validation is modified as followed :

1. Initialize i to 1;
2. Randomly select without replacement 90% of the data set to be the train set on which the feature selection will be done. The remaining data points are unused;
3. Perform feature selection on the train set;
4. Report the subset of selected features;
5. Increment i and repeat from step 2 (until $i > m$, m being the number of iterations of the cross validation);

A stability index is then computed from all the subsets reported during the cross validation. It is interesting to see whether the selected features are random or if there is actually a trend. In this document, the use of a good and stable feature selection method is even more important as it brings insight about the relationships between the features and the pain of a patient.

As stability index of a feature selection method, Kuncheva's index of stability is a natural choice[12]. It is the topic of the next sections.

Consistency between two subsets : Kuncheva's index of stability

Let $X = \{x_{*1}, ..., x_{*p}\}$ be the whole set of p features. Let k be the size of the subset of selected features.

An index of consistency between two subsets is based on the size of the intersection of the two said subsets. It is not surprising as a high size for the intersection means that the two subsets have a lot of features in common. If A and B are two subset of X, let $\kappa = |A \cap B|$ be the cardinality of the intersection.

Kuncheva's index is thus :

$$\frac{\text{Observed } \kappa - \text{Expected } \kappa}{\text{Maximum } \kappa - \text{Expected } \kappa}$$

with Observed κ being the observed cardinality of the intersection of A and B, Maximum κ the maximum value that κ can take for two subsets of size k and Expected κ the expected value of κ when drawing features randomly in X.

In this case, Maximum κ is simply k (when A and B are equal). The expected value for κ is obtained by considering it as an element drawing problem[12]. Suppose that the k elements of A are fixed and are colored black. The elements of X that are in A are black and the others are white. The number of black element drawn from X is a hypergeometric distribution with probability density function :

$$P(Y = \kappa) = \frac{\binom{k}{\kappa}\binom{p-k}{k-\kappa}}{\binom{p}{k}}$$

The expected value for κ is thus $\frac{k^2}{p}$. The final formulation of the index is consequently :

$$\begin{aligned} I(n, A, B) &= \frac{\kappa - \frac{k^2}{p}}{k - \frac{k^2}{p}} \\ &= \frac{\kappa p - k^2}{kp - k^2} \\ &= \frac{\kappa p - k^2}{k(p-k)} \end{aligned}$$

This metric is interesting because it is monotonic, bounded with a maximum value of 1 (because $\max(0, 2k-p) \leq \kappa \leq k$) and corrected for chance (the expected size of the intersection between the sets is taken into account).

Consistency between more than two subsets

As mentioned earlier, the stability of the feature selection is assessed through a Monte Carlo cross validation. It means that the consistency must be measured on more than two subsets. To do so, a stability index is computed for each pair of subsets obtained during the validation. The final index of stability is simply their average.

Finally, it is good to know that a value for the index above 0.5 is considered stable.[12]

4.2 Principal component analysis (PCA)

Principal component analysis (PCA) is probably the most well known unsupervised dimensionality reduction technique. In short, PCA is a technique that applies a linear transformation to the data in order to obtain uncorrelated variables (or even white variables, meaning uncorrelated, zero-mean, unit-variance variables)[17].

Let $x \in \mathbb{R}^{n \times p}$ be the data set with n rows (samples) and p columns (features).

Covariance matrix

Lets define $\Sigma_x = [Cov[x_{*i}, x_{*j}]] \ \forall i, j \in \{1..p\}$, the covariance matrix of x. The covariance matrix can also be expressed as :

$$\hat{\Sigma}_x = \frac{1}{n}(cx)^T(cx) = \frac{1}{n}x^T ccx$$

with $c = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$ being the centering matrix ($\mathbf{1}$ is a column vector of ones).
An interesting property is that the covariance matrix is Positive Semi-Definite (PSD)[1]. This is proved because a matrix such as : $A = B^T B$ is always PSD. Indeed :

$$x^T Ax = (Bx)^T(Bx) = \|Bx\|_2^2 \geq 0$$

By design, Σ_x fits this property.

Orthogonal Matrix

A matrix U is orthogonal if and only if $U^T U = I = UU^T$, i.e. $U^{-1} = U^T$. It means that the columns of U are perpendicular to each other with unit length. In other words U defines a transformation of matrices to a new space.

[1] A square symmetric matrix A is PSD if and only if $b^T Ab \geq 0$ for every nonzero column vector b.

Eigenvalue decomposition

Another concept needed is the Eigenvalue decomposition. If A is square, symmetric, real-valued it can be decomposed as :

$$A = \vee \wedge \vee^T$$

with $\vee$ being an orthogonal matrix. Its columns are called the eigen vectors. $\wedge$ is a diagonal matrix, its diagonal elements λ_i's are called the eigen values.
The order of the vector and value is arbitrary, so the usual convention is to order the eigen values in decreasing order.

Solution

The goal of PCA is to find a linear transformation $z = xw$ of x such that Σ_z is a diagonal matrix, meaning that there is no covariance between the different variables.

First, the data is considered to be centered ($x = cx$). The expression of the covariance matrix becomes :

$$\hat{\Sigma}_x = \frac{1}{n} x^T x$$

As it is PSD, EVD can be used on it :

$$\hat{\Sigma}_x = \vee \wedge \vee^T$$

This is interesting because one solution for w to make z uncorrelated is $w = \vee$. Here is a proof of this :

$$\begin{aligned} \hat{\Sigma}_z &= \frac{1}{n} z^T z = \frac{1}{n} (xw)^T (xw) \\ &= \frac{1}{n} (x\vee)^T (x\vee) = \frac{1}{n} \vee^T x^T x \vee \end{aligned}$$

and

$$\hat{\Sigma}_x = \frac{1}{n} x^T x$$

So :

$$\begin{aligned} \hat{\Sigma}_z &= \vee^T \hat{\Sigma}_x \vee \\ &= \vee^T \vee \wedge \vee^T \vee \end{aligned}$$

As $\vee$ is orthogonal, $\vee\vee^T = I = \vee^T\vee$ and so :

$$\hat{\Sigma}_z = \wedge$$

And as the eigen value matrix is diagonal, the features of z are effectively uncorrelated. Moreover, because of the orthogonality of w, x can be retrieved from z :

$$\begin{aligned} z &= xw \\ zw^T &= xww^T \\ zw^T &= x \end{aligned}$$

Dimensionality reduction

PCA is often used to reduce the number of variables of the data. By making the data uncorrelated, most of the information contained in the data is contained in the first features of the uncorrelated data.

Let $w_{1..d}$ contain the first P columns of w, $z = xw_{1..d}$ effectively project x from $\mathbb{R}^{\mathrm{n}\times\mathrm{p}}$ to $\mathbb{R}^{\mathrm{n}\times\mathrm{d}}$. Taking the d leftmost columns of w effectively captures the maximum percentage of variance of x using d variables. The percentage is : $\frac{\sum_{i=1}^{d}\lambda_i}{\sum_{i=1}^{p}\lambda_i}$

An evident downside of this technique is that $w_{1..d}$ is not orthogonal so there is an information loss when trying to retrieve x from z. $\hat{x} = zw_{1..d}w_{1..d}^T$

Chapter 5

Experimental results

In this chapter, the task of finding relevant psychological questions when it comes to pain and the task of building models to predict patient pain from personality traits are addressed. As stated earlier in this document, the patients either suffer from osteoarthritis or peripheral neuropathic pain. When referring to the questionnaire data, the features *Age*, *Disease* and *Sex* from the matrix *patientInfo* are included. After imputing the missing values using answers from mpsqData3 and removing patients for which there is still missing values, the remaining data consists of 150 features for 141 patients.

5.1 Assessment method and model parameters

In the next sections, the results of our experiments are presented and interpreted. They were obtained via a Monte Carlo cross validation with 1000 iterations. The percentage of data used as test set for each iteration is 10%. Every model presented in chapter 1 was tried, except the ordinary least square regression because the questionnaire data is ill-configured.

For each experiment, the hyper-parameters[1] of the models are set and don't change between the experiments. They are arbitrary and not necessarily the best as the goal is to find promising methods and not trying all the possible hyper-parameters in order to maximize the correlation. Here are the most important hyper-parameters for each model :

- Ridge regression : Response type = Gaussian (distribution of the response), nlambda = 100 (number of different values for λ, they are chosen in an efficient way by the *glmnet* function) and the rest is default.

[1] parameters used to tune the learning of the model

- Lasso regression : Same as Ridge
- Elastic Net : Same as Ridge and Lasso, α is set to 0.5.
- SVR L1-loss : Cost (C) = estimated by a function called *heuristicC* from the *LiblineaR* package, svr_eps (ϵ) = 0.05 (size of the margin), the rest is default.
- SVR L2-loss : Same as SVR L1-loss
- Random Forest : ntree = 250 (the number of trees in the forest), mtry = $p/3$ (the number of features considered among a total of p when choosing the best split possible), replace = TRUE (when sampling to build trees, replacements[2] are allowed), nodesize = 5 (minimum size of terminal nodes), sampsize = size of the input (number of samples drawn before building a tree, since replacements are allowed each sampling will probably be different with a maximum of data), the rest is default.

5.2 Approach followed for the tasks at hand

To carry out the two tasks at hand, the data is adapted in order to be able to make decent predictions. First, the response data is transformed using PCA, then a normalization scheme for the questionnaire data is defined.

The next sections then present the attained correlations for the different models using these adaptations of the data. Finally, the attained performance when building models with a limited number of features as well as the most often selected features are reported.

PCA of the pain data

The pain data contains several measurements (features), each of them trying to measure pain. The first three features (APS_Baseline : the average pain felt during the baseline week, WPS_Baseline : the worst pain felt during the baseline week and BPI-Severity : mean of the maximum pain, minimum pain, average pain and current pain during the last day of the baseline week) are judged as the most relevant ones by our supervisor *Samuel Branders*. Moreover, the others contain a lot of missing values, making them harder to use.

[2]drawing a sample more than once

Before trying to predict pain, it may be a good idea to have an accurate measurement of the real pain felt by a patient, in a single feature. The PCA of the first three features can be used as an attempt to create it : the first component of the analysis could be used as the response data. Predicting the first component of this PCA is interesting because the intrinsic value behind all the measurements is supposed to be the same : the actual pain of the patient. The three features are actually assumed to be a scaled and shifted, noisy version of the pain. Ideally, if the noise components of each variable are independent, the first component of the PCA will be highly correlated with the actual pain of the patient. Figure 5.1 shows the percentage of the variance captured by each of the components. The first component captures almost 80% of the variance. This means that the three features contain a lot of redudant information and probably contain a measurement of the same thing (the pain).

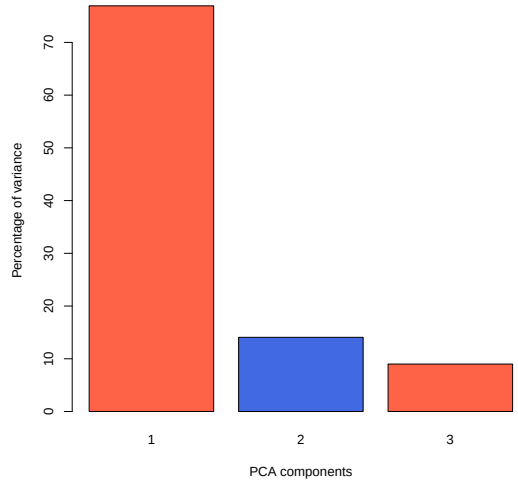


Figure 5.1: Percentage of variance captured by each component of the PCA of the first three features of pain

Usually, when using PCA in the context of predictive models, it is advisable to learn the PCA coefficients on the training set and then apply them on the test set in order to avoid a possible optimistic bias. Nonetheless, in this work, PCA was directly applied on the whole set of response data. It is justified as our goal of retrieving the actual pain measure could have been achieved by simply averaging the three features but PCA has the advantage of taking into account the scale of each response features.

The PCA leads to three coefficients to compute the first component :

$$\text{First component} = 0.54\ \text{APS\_Baseline} + 0.58\ \text{WPS\_Baseline} + 0.61\ \text{BPI-Severity}$$

Normalization of the questionnaire data

To make predictions, the input data isn't usually used as is. There is first a preprocessing step : normalization. Normalization is a common concern when working with data. Normalizing the data can indeed lead to better performance for predictive tasks and may help interpreting the data manually as it tends to put all the features on a same scale. A common type of normalization is the standard scaling which transforms the features so that the observed mean for each feature is null and the observed variance is 1. If you have a feature x_{*i}, the scaled feature is obtained like this : $\frac{x_{*i} - \bar{x}_{*i}}{\sigma_{x_{*i}}}$.

Nonetheless, by knowing how the data is generated, data specific normalization can be attempted. In this document, another normalization scheme is used before the standard scaling. We called it "patient scaling". In this scheme, the patients are scaled independently so that the answers of each patient have a mean of zero and a variance of 1. The reasoning behind this scaling is that it removes the natural bias of each patient as some patients consistently answer with high/low values[3]. This is in fact the same as the standard scaling but along the samples rather than along the features. The answers for a question are then reversed if needed and the whole questionnaire is then scaled in a standard way. The other features not coming from the psychological survey (Age, Sex, Disease) are simply scaled in a standard way. To avoid the optimistic bias, the parameters for the final standard scaling (the mean and variance of each feature) should be learned on the training set and then applied on the test set to scale it. This is effectively done here, at each iteration of the Monte Carlo cross validation.

Baseline predictions

The baseline predictions are the ones made using the normalized questionnaire data. The response data is the first component of the PCA of the pain data as stated earlier. To make sure that the choice of using the first component of PCA as response data and the choice of normalizing the questionnaire data was reasonable, models are also built on the raw data. The input data is simply the questionnaire data unchanged and the response data is APS_Baseline. The performance of the two approaches are compared.

[3]Each patient has a positive or negative bias in their answers, the goal is to remove it.

Dimensionality reduction and predictions when reducing the dimension of the questionnaire data

Once the performance of models that use all the questionnaire data is assessed, the performance of each model when selecting (or reducing the dimension to) 25 features is evaluated for each model. The choice to select (or reduce the dimension to) 25 features is explained later. Of course, to avoid an optimistic bias, the dimensionality reduction methods select or create features using only the train set. The test set is then modified accordingly.

The different dimensionality reduction methods considered are :

- Correlation as a filter criterion : the 25 features that are the most correlated with the response are selected
- Facet and correlation : one feature per facet is selected using correlation as a filter criterion
- Lasso regression : by modulating the parameter λ, the number of non-zero weights in a model built using Lasso can be chosen. The model is build on the normalized training set. The features which have non-zero weights are the selected ones. λ is chosen so that at least 25 features have non-zero weights.
- Elastic net : it is used in the same way as the Lasso regression, but the quality of the selection and the stability of the selection could be better for reasons explained in **an earlier section**

The questions that are the most often chosen by the different methods are also reported in order to interpret the possible relation witsh the pain.

5.3 Baseline predictions

This section presents results obtained using "baseline predictors", i.e. predictors which are built with the normalized questionnaire data to predict the first component of the PCA of the pain data. The results are gathered in table 5.1. Each correlation coefficient obtained is significantly different from 0 except for the Random Forest model. For the Ridge, Lasso and Elastic Net models, the predictions are made using the smallest value of λ computed during training (the regularization of the weights is minimal but still present and doesn't set weights to 0 in the case of Lasso and Elastic Net).

[4]Confidence Interval

Model	Correlation	p-value	Lower bound of CI[4]	Upper bound of CI
Ridge regression	0.3	2.7 10^{-4}	0.14	0.44
Lasso regression	0.29	5 10^{-4}	0.13	0.43
Elastic net	0.29	3.8 10^{-4}	0.13	0.44
SVR L1-Loss	0.29	4.3 10^{-4}	0.13	0.44
SVR L2-Loss	0.29	5.5 10^{-6}	0.13	0.43
Random Forest	0.13	0.11	-3.1 10^{-2}	0.29

Table 5.1: Correlation obtained when using predictors learned on the normalized questionnaire data to predict the first component of the PCA of the pain data

To make sure that normalizing the questionnaire data and modifying the response data was a right choice, the performance of the models are also assessed when they are built using the raw questionnaire data to predict *APS_Baseline*. The results are in table 5.2. Here, almost every correlation coefficient obtained cannot be said to be significantly different from 0. This seems to indicate that the normalization method is reasonable.

Model	Correlation	p-value	Lower bound of CI	Upper bound of CI
Ridge regression	0.15	0.08	-0.02	0.30
Lasso regression	0.08	0.34	-0.09	0.24
Elastic net	0.09	0.31	-0.08	0.25
SVR L1-Loss	0.20	0.02	0.04	0.35
SVR L2-Loss	0.16	0.05	-1.2 10^{-3}	0.32
Random Forest	0.15	0.07	-1.5 $10^{-}2$	0.31

Table 5.2: Correlation obtained when using predictors built on the raw questionnaire data to predict *APS_Baseline*

From now on, when referring to "pain data", it is actually the first component of the Principal Component Analysis of the first three features (*APS_Baseline, WPS_Baseline and BPI_Severity_V2*) available in the pain matrix as it was explained in **the previous section**.

5.4 Dimensionality reduction and its consequence on the predictive performance of models

In this section, the dimensionality reduction techniques presented earlier are assessed through the same Monte Carlo cross validation used for predictions. At each iteration (in independent Monte Carlo cross validations for each method), each method is trained on the normalized train set and applied on the normalized test set to avoid an optimistic bias. The performance of each considered model is reported when the dimension is reduced to 25 features using every considered dimensionality reduction method.

For feature selection methods only, the selected questions at each iteration are recorded. That way, the stability of the feature selection techniques can be computed and a ranking of the most frequent questions can also be made. The 25 features that appeared the most in the 1000 iterations are reported with their frequency of occurrence and the correlation of the feature with the pain data (for the first two methods) or the average weight associated to the question learned by the model (for the last two : Lasso and Elastic Net). It is also indicated if the question is to be reversed or not to help the interpretation. If the question is to be reversed, the opposite of the correlation or the weight reported is best suited for interpretation. The value of Kuncheva's index is reported for each method, with an histogram of the pairwise Kuncheva's index between the 1000 subset of features (in appendix). The facets associated with the questions are reported with the frequency of occurrence as well as the relative frequency of occurrence (frequency divided by the number of questions in the facet) in order to account that facets comprising a lot of questions are more likely to appear. They are also in the appendix as no interpretation can be done without their name.

5.4.1 Choosing the number of features to select/create

To choose the number of features to select or create when using dimensionality reduction techniques, we chose to analyze in parallel the stability of the feature selection and performance of the Elastic Net model when the number of features selected increases.

Usually, the stability grows until it reaches a peak and then decreases. This makes sense because when selecting a number of features that is lower than the right one (i.e. the one to get most of the features really needed for prediction), features will probably be "randomly" chosen among the most relevant ones. While when selecting more features than needed, the most important ones are usually

going to be selected but the less important features are selected randomly as they don't help much to make predictions.

The performance should increase until reaching a certain limit. It also makes sense because the more features are selected, the more information is available for the model. But at a certain point, the added features are mostly useless.

A good number of features to select would be a number that belongs to the range of the stability peak and that (almost) attains the limit of the performance. Figure 5.2 displays these two graphs and it is observed that at 25 features, the performance begins to reach its limit while being the number of features leading to the best stability. It is remarkable that this number is close to the number of facets (24). This means that prior knowledge could probably also have been used to choose the number of features to select or create.

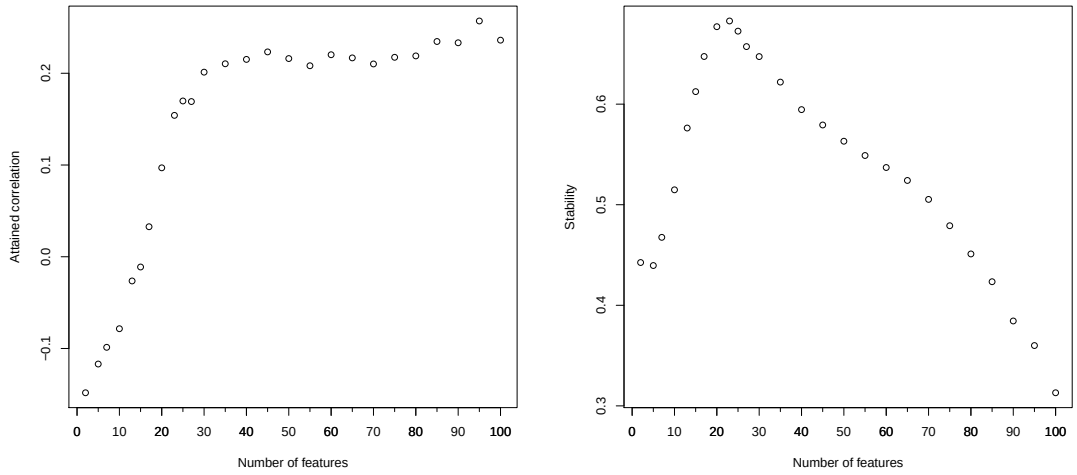


Figure 5.2: Performance (correlation) and stability of feature selection in function of the number of features selected when using the Elastic Net model. The features are selected by adjusting the parameter λ when predicting the pain of patients.

This choice obviously introduces an optimistic bias because it is made using all the data. It is reasonable as only the number of features is chosen and not the actual chosen features. This fact should still be kept in mind for further analysis.

5.4.2 Selection with correlation

The first feature selection technique considered is using Pearson's correlation coefficient as a filter. It keeps the 25 features with the highest absolute correlation with the pain data. In figure 6.9, it is observed that selecting features this way is quite stable (index of stability : 0.67).

The performances of the models when using this feature selection technique are in table 5.3. Every model leads to a correlation significantly different from 0 except the Random Forest model.

Model	Correlation	p-value	Lower bound of CI	Upper bound of CI
Ridge regression	0.18	0.03	$1.6\ 10^{-2}$	0.34
Lasso regression	0.18	0.03	$1.6\ 10^{-2}$	0.34
Elastic net	0.18	0.03	$1.6\ 10^{-2}$	0.34
SVR L1-Loss	0.18	$3.5\ 10^{-2}$	$1.2\ 10^{-2}$	0.33
SVR L2-Loss	0.19	$2.3\ 10^{-2}$	$2.6\ 10^{-2}$	0.35
Random forest	0.15	$8.3\ 10^{-2}$	$-2\ 10^{-2}$	0.30

Table 5.3: Correlation obtained when using predictors learned on the 25 features with highest absolute correlation of the normalized questionnaire data to predict the first component of the PCA of the pain data

The result table 5.4 shows that some items are consistently selected more than others.

Item	Corresponding psychological question	Frequency	r	Reversed
Age	/	1000.00	-0.22	/
item_143	My pain disappears when I take painkillers.	1000.00	-0.26	F
item_070		985.00	-0.21	F
Disease	/	979.00	0.20	/
item_138	My relatives agree that painkillers is a good treatment for what I suffer.	976.00	-0.21	F
item_050	I worry when I think that I've not done something correctly.	941.00	-0.19	F
item_090	Usually, I'm efficient when in situation of emergency.	936.00	-0.19	F

item_109	I'm often nervous and tense.	919.00	0.19	F
item_124	I've already felt so ashamed that I wanted to disappear.	917.00	0.19	F
item_122	I'm always afraid of doing something bad when it comes to other people.	878.00	0.18	F
item_097	/	857.00	-0.17	F
item_113	/	850.00	0.18	F
item_047	/	840.00	-0.17	F
item_017	/	785.00	-0.16	F
item_130	I rarely overdo in anything.	758.00	-0.16	T
item_006	/	684.00	-0.16	T
item_139	Painkillers correct changes in my brain connected to stress or issues.	684.00	-0.16	F
item_103	I have the feeling that if there's a risk that something is going to go wrong for me, it will.	671.00	-0.16	T
item_005	/	665.00	-0.16	F
item_105	I don't expect a bright future for myself.	665.00	-0.16	T
item_026	I think I'm a shy person.	661.00	0.16	T
item_008	I think I am sometimes rude to people.	657.00	-0.16	T
item_146	/	629.00	-0.16	F
item_059	I often act impulsively.	626.00	-0.16	F
item_009	/	491.00	-0.15	F

Table 5.4: Frequency and correlation (r) with the pain data of the most relevant features selected with correlation. The items are sorted decreasingly by frequency. Note that Age and Disease are often selected.

5.4.3 Facet and correlation

One feature for each facet is selected with correlation (in the same fashion as in the previous section) in order to use the prior knowledge about the features. This kind of selection is perfectly stable (index of stability of 1) which means that one question of each facet is clearly more correlated with the pain than the others. The performances of the models when using this feature selection method are in table

5.5. Only Random Forest leads to a correlation significantly different from 0. It makes sense because looking at the results in table 5.6, it is observed that for quite a lot of facets, the best feature according to its correlation with the pain data isn't highly correlated with the pain data. Random Forest being a non linear model, it may use non-linear relationships between the selected features and the pain to make decent predictions.

Model	Correlation	p-value	Lower bound of CI	Upper bound of CI
Ridge regression	0.12	0.14	-3.9 10^{-2}	0.29
Lasso regression	0.12	0.14	-4.2 10^{-2}	0.28
Elastic net	0.12	0.14	-4.2 10^{-2}	0.28
SVR L1-Loss	2.8 10^{-2}	0.74	-0.14	0.19
SVR L2-Loss	0.14	0.1	-2.8 10^{-2}	0.30
Random forest	0.2	1.5 10^{-2}	4 10^{-2}	0.36

Table 5.5: Correlation obtained when using predictors learned on one feature per facet (selected as the feature with highest absolute correlation for each facet) of the normalized questionnaire data to predict the first component of the PCA of the pain data

Item	Corresponding psychological question	Frequency	r	Reversed
item_001	I tend to criticize others.	1000.00	0.13	T
item_011	/	1000.00	-0.03	F
item_020	/	1000.00	0.008	F
item_028	/	1000.00	0.005	F
item_036	/	1000.00	0.06	F
item_046	/	1000.00	0.01	T
item_053	/	1000.00	-0.05	F
item_057	I'm always ready to do something new, especially if I think it will be fun.	1000.00	0.13	F
item_061	I enjoy doing task which I am good at.	1000.00	0.02	F
item_066	/	1000.00	0.06	F
item_069	/	1000.00	-0.1	F

item_072	I feel sympathy for people that are unlucky.	1000.00	0.10	F
item_079	/	1000.00	0.12	F
item_086	/	1000.00	-0.05	F
item_093	/	1000.00	-0.04	T
item_100	/	1000.00	0.006	F
item_103	I have the feeling that if there's a risk that something is going to go wrong for me, it will.	1000.00	-0.16	T
item_106	/	1000.00	0.10	T
item_114	/	1000.00	0.09	F
item_122	I'm always afraid of doing something bad when it comes to other people.	1000.00	0.18	F
item_130	I rarely overdo in anything.	1000.00	-0.16	T
item_138	My relatives agree that painkillers is a good treatment for what I suffer.	1000.00	-0.21	F
item_141	/	1000.00	0.03	F
item_143	My pain disappears when I take painkillers.	1000.00	-0.26	F

Table 5.6: Frequency and correlation (r) with the pain data of the most relevant features selected with the correlation and imposing one feature per facet. The items are sorted decreasingly by frequency.

5.4.4 Lasso regression and Elastic Net

The variables are selected by building a model with Lasso/Elastic Net. By carefully choosing the parameter λ, the number of chosen variables is selected. This method of feature selection is also considered stable (stability index of 0.655 for Lasso and 0.671 for Elastic Net). A lot of selected questions overlap the ones selected in the previous sections and the weights usually have the same sign as the correlation, which is expected. The most selected features are listed in tables 5.9 and 5.10. The performances are gathered in tables 5.7 and 5.8. Here, every model performs with a correlation significantly different from 0.

Model	Correlation	p-value	Lower bound of CI	Upper bound of CI
Ridge regression	0.28	$8.8\ 10^{-4}$	0.12	0.42
Lasso regression	0.28	$9\ 10^{-4}$	0.12	0.42

Elastic net	0.28	$9\ 10^{-4}$	1.2	0.42
SVR L1-Loss	0.26	$2.1\ 10^{-3}$	$9.3\ 10^{-2}$	0.4
SVR L2-Loss	0.27	$1.3\ 10^{-3}$	0.11	0.41
Random forest	0.21	$1.2\ 10^{-2}$	$4.7\ 10^{-2}$	0.36

Table 5.7: Correlation obtained when using predictors learned on 25 features selected by Lasso of the normalized questionnaire data to predict the first component of the PCA of the pain data

Model	Correlation	p-value	Lower bound of CI	Upper bound of CI
Ridge regression	0.29	$4.5\ 10^{-4}$	0.13	0.43
Lasso regression	0.29	$4.5\ 10^{-4}$	0.13	0.43
Elastic net	0.29	$4.5\ 10^{-4}$	0.13	0.43
SVR L1-Loss	0.28	$7\ 10^{-4}$	0.12	0.43
SVR L2-Loss	0.28	$8.1\ 10^{-4}$	0.12	0.42
Random Forest	0.19	$2.5\ 10^{-2}$	$2.4\ 10^{-2}$	0.34

Table 5.8: Correlation obtained when using predictors learned on 25 features selected with Elastic Net of the normalized questionnaire data to predict the first component of the PCA of the pain data

Item	Corresponding psychological question	Frequency	Weight	Reversed
item_143	My pain disappears when I take painkillers.	1000.00	-0.28	F
item_026	I think I'm a shy person.	998.00	0.21	T
Age	/	995.00	-0.16	/
item_033	I'm sometimes moody.	988.00	-0.17	F
item_097	/	987.00	-0.11	F
item_050	I worry when I think that I've not done something correctly.	972.00	-0.12	F
item_130	I rarely overdo in anything.	963.00	-0.11	T
item_070	/	962.00	-0.11	F
item_122	I'm always afraid of doing something bad when it comes to other people.	950.00	0.13	F

item_090	Usually, I'm efficient when in situation of emergency.	934.00	-0.10	T
Disease	/	924.00	0.09	/
item_138	My relatives agree that painkillers is a good treatment for what I suffer.	923.00	-0.11	F
item_059	I often act impulsively.	920.00	-0.11	F
item_017	/	859.00	-0.07	F
item_005	/	804.00	-0.07	F
item_047	I'm affected by reproaches and negative remarks.	717.00	-0.06	F
item_113	/	659.00	0.06	F
item_109	I'm often nervous and tense.	642.00	0.07	F
item_009	/	604.00	-0.05	F
item_006	/	595.00	-0.05	T
item_001	I tend to criticize others.	593.00	0.05	T
item_103	I have the feeling that if there's a risk that something is going to go wrong for me, it will.	525.00	-0.05	T
item_008	I think I am sometimes rude to people.	507.00	-0.04	T
item_139	Painkillers correct changes in my brain connected to stress or issues.	504.00	-0.05	F
item_105	I don't expect a bright future for myself.	494.00	-0.06	T

Table 5.9: Frequency and weight (w) of the most relevant features selected with Lasso. The items are sorted decreasingly by frequency. Note that Age and Disease are often selected

Item	Corresponding psychological question	Frequency	Weight	Reversed
item_143	My pain disappears when I take painkillers.	1000.00	-0.23	F
item_026	I think I'm a shy person.	997.00	0.16	T
Age	/	996.00	-0.14	/

item_033	I'm sometimes moody.	986.00	-0.13	F
item_097	/	983.00	-0.10	F
item_070	/	969.00	-0.10	F
item_050	I worry when I think that I've not done something correctly.	968.00	-0.11	F
item_130	I rarely overdo in anything.	967.00	-0.09	T
item_090	Usually, I'm efficient when in situation of emergency.	956.00	-0.09	T
item_122	I'm always afraid of doing something bad when it comes to other people.	945.00	0.10	F
Disease	/	936.00	0.09	/
item_138	My relatives agree that painkillers is a good treatment for what I suffer.	931.00	-0.10	F
item_059	I often act impulsively.	918.00	-0.09	F
item_017	/	866.00	-0.06	F
item_005	/	780.00	-0.0-	F
item_047	I'm affected by reproaches and negative remarks.	753.00	-0.06	F
item_109	I'm often nervous and tense.	686.00	0.06	F
item_113	/	675.00	0.06	F
item_001	I tend to criticize others.	618.00	0.04	T
item_006	/	594.00	-0.04	T
item_009	/	593.00	-0.05	F
item_103	I have the feeling that if there's a risk that something is going to go wrong for me, it will.	556.00	-0.04	T
item_124	I've already felt so ashamed that I wanted to disappear.	550.00	0.05	F
item_139	Painkillers correct changes in my brain connected to stress or issues.	536.00	-0.05	F
item_008	I think I am sometimes rude to people.	518.00	-0.04	T

Table 5.10: Frequency and weight (w) of the most relevant features selected with Elastic Net. The items are sorted decreasingly by frequency. Note that Age and Disease are also often selected.

5.4.5 Interpretation

Here are some interpretations of different questions selected with the first and the last two methods :

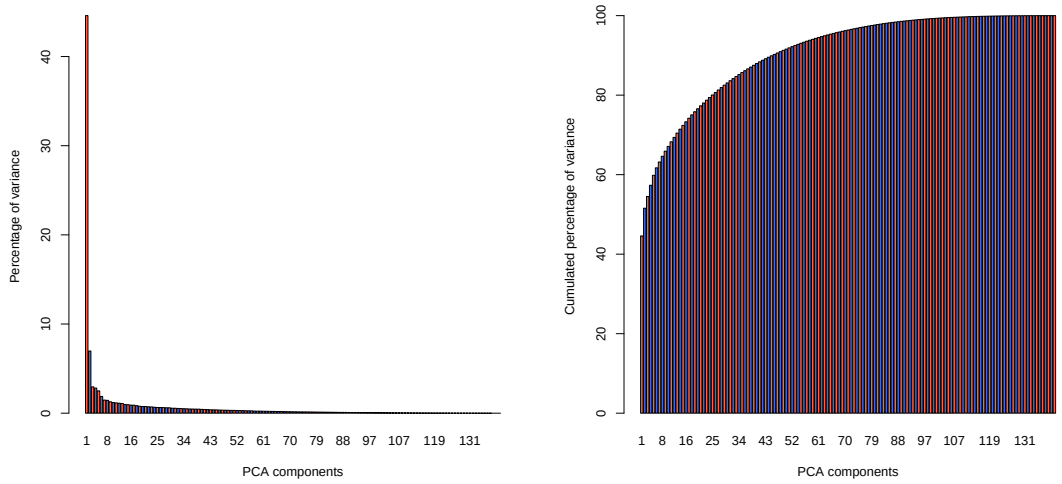
- The question "My pain disappears when I take painkillers" is negatively correlated with the pain. The weights in Lasso and Elastic Net are also negative. It makes sense because if the pain disappears with painkillers, it probably means that it wasn't big in the first place.
- The question "I'm often nervous and tense" is positively correlated with the pain and the weights are positive. Again, this makes sense because a nervous and tense person is focused on the negative things such as pain. If he is focused on it, maybe he feels it more than another person.
- The question "I think I'm a shy person" is a question that is to be reversed. It means that the weights are actually negative. A shy person might not dare to express his pain or even feel it.
- The question "I often act impulsively" is negatively correlated and its weights are negative too. When people act impulsively, they often don't care about what happens next. That might mean that in situation supposedly leading to pain, they actually won't feel as much as others might.

Age and Disease appear when using the first and last two methods. It is not surprising as it is very likely that one of the diseases is more painful than the other. Here it seems to indicate that PNP is more painful that OA. As for the age, one might think that when getting older, you get used to pain. The correlation and weights are negative, which seems to support that statement.

5.4.6 PCA of the questionnaire data

As explained before, PCA can be used as a dimensionality reduction technique. Figure 5.3 shows the percentage of variance captured by every component of the PCA of the questionnaire data as well as the cumulative percentage of variance captured. Most of the information is contained in the first components. In fact, with 25 components, approximately 80% of the variance is already captured. It could be a good idea to use them to lower the dimension of the input data. This

agrees with the previous choice to select 25 features when performing feature selection.



(a) Percentage of variance captured by PCA components

(b) Cumulative percentage of variance captured by PCA components

Figure 5.3: PCA of the questionnaire data. The left plots shows that the first components are the most important ones. The right plot shows that when selecting only the 25 first components, approximately 80% of the variance is captured.

As for the feature selection and the normalization, the PCA of the questionnaire data must be done at each iteration on the train set and applied to the test set to avoid the optimistic bias. The resulting correlations are gathered in table 5.11. There is no real improvement in the performance of the models. It isn't surprising because the goal of dimensionality reduction is to avoid problems due to high dimensionality for some models while keeping the performance as high as possible. It usually decreases it even though in some cases it could increase it by removing useless information.

Model	Correlation	p-value	Lower bound of CI	Upper bound of CI
Ridge regression	0.23	$6.2\ 10^{-3}$	$6.6\ 10^{-2}$	0.38
Lasso regression	0.23	$6.2\ 10^{-3}$	$6.6\ 10^{-2}$	0.38
Elastic net	0.23	$6.3\ 10^{-3}$	$6.6\ 10^{-2}$	0.38
SVR L1-Loss	0.23	$5.2\ 10^{-3}$	0.07	0.38
SVR L2-Loss	0.24	$4.7\ 10^{-3}$	$7.3\ 10^{-2}$	0.39

Random Forest	0.16	$6.3\ 10^{-2}$	$-8.3\ 10^{-3}$	0.31

Table 5.11: Correlation obtained when using predictors learned on the 25 first components of the PCA of the normalized questionnaire data to predict the first component of the PCA of the pain data

Conclusion and perspectives

This thesis ends with promising results about the existence of a relationship between the personality of a patient and his baseline pain. Achieving correlation of the order of 0.3 with a p-value of 2.7 10^{-4} with Ridge regression learned on the normalized questionnaire data to predict the first component of the PCA of the first three features of pain, and a correlation of 0.29 with a p-value of 4.5 10^{-4} with Ridge, Lasso and Elastic Net learned on 25 features selected with Elastic Net from the normalized questionnaire data (also to predict the first component of the PCA of the first three features of pain).

However these correlation coefficients aren't high enough to claim that the baseline pain of a patient can be predicted correctly (as the individual predictions are pretty poor). But it is high enough to warrant the possibility to predict well the pain of a patient using more advanced techniques and models learned on a more extensive set of data, as well as warrant the existence of an underlying relationship between the personality of the patient and its pain.

From a feature selection standpoint, some questions have been identified reliably with our different techniques which makes us confident about their importance for pain prediction.

As a perspective for our work, multiple ideas that have a chance to make the predictions better or bring insights about the data have been identified :

- The survey is nearly 150 questions long and it would be foolish to hope that patients that have taken it would answer the last questions with the same thoroughness as for the first ones. Modifying the data to take into account the order of the questions would maybe lead to better predictions.
- Find a way to remove some outliers from the data. Correlation is very sensitive to them and so are some learning algorithms.
- In the case of this master thesis, using PCA on the questionnaire isn't very interesting because the new features make further interpretation difficult.

However, feature selection is highly desirable for interpretation as it isolates a subset of interesting features. Using the prior knowledge about the questionnaire (the facets) to select features may yields interesting results. Such a scheme was attempted with poor results but it is probably possible to design a more complex scheme taking advantage of the prior knowledge.

- Some time could have been taken to identify if some questions were too similar to another one. This information would have been helpful for Tools4Patient as the smaller the survey, the easier it is for the patient, the less time is required to ask the questions and the less expensive it is.
- Considering more non-linear models. In this thesis, almost every model used is linear. Knowing that the data isn't a lot linearly related (given the correlations between the features and the pain), it might be interesting to use other kinds of models.

Bibliography

[1] AG Asuero, A Sayago, and AG Gonzalez. "The correlation coefficient: An overview". In: *Critical reviews in analytical chemistry* 36.1 (2006), pp. 41–59.

[2] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[3] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[4] CS Cleeland and KM Ryan. "Pain assessment: global use of the Brief Pain Inventory." In: *Annals, Academy of Medicine, Singapore* (1994).

[5] William G Cochran. "The $\chi 2$ test of goodness of fit". In: *The Annals of Mathematical Statistics* (1952), pp. 315–345.

[6] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.

[7] Pierre Dupont. *Lecture slides in machine learning course, LINGI2262*.

[8] *glmnet.pdf*. `https://cran.r-project.org/web/packages/glmnet/glmnet.pdf`. (Accessed on 04/21/2019).

[9] Priscilla E Greenwood and Michael S Nikulin. *A guide to chi-squared testing*. Vol. 280. John Wiley & Sons, 1996.

[10] Isabelle Guyon and André Elisseeff. "An introduction to variable and feature selection". In: *Journal of machine learning research* 3.Mar (2003), pp. 1157–1182.

[11] Ron Kohavi and George H John. "Wrappers for feature subset selection". In: *Artificial intelligence* 97.1-2 (1997), pp. 273–324.

[12] Ludmila I Kuncheva. "A stability index for feature selection." In: *Artificial intelligence and applications*. 2007, pp. 421–427.

[13] Roger J Lewis. "An introduction to classification and regression tree (CART) analysis". In: *Annual meeting of the society for academic emergency medicine in San Francisco, California*. Vol. 14. 2000.

[14] *liblinear.pdf*. https://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf. (Accessed on 05/08/2019).

[15] *LiblineaR.pdf - R package*. https://cran.r-project.org/web/packages/LiblineaR/LiblineaR.pdf. (Accessed on 04/21/2019).

[16] Donald W Marquardt and Ronald D Snee. "Ridge regression in practice". In: *The American Statistician* 29.1 (1975), pp. 3–20.

[17] John Lee Michel Verleysen. *Lecture slides in machine learning course, LELEC2870*.

[18] Tom M Mitchell and Machine Learning. "Mcgraw-hill science". In: *Engineering/Math* 1 (1997), p. 27.

[19] Andrew Moore et al. "Expect analgesic failure; pursue analgesic success". In: *BMJ: British Medical Journal (Online)* 346 (2013).

[20] Duncan J Murdoch, Yu-Ling Tsai, and James Adcock. "P-values are random variables". In: *The American Statistician* 62.3 (2008), pp. 242–245.

[21] Claude Nadeau and Yoshua Bengio. "Inference for the generalization error". In: *Advances in neural information processing systems*. 2000, pp. 307–313.

[22] *Overview of Clinical Trials | CenterWatch*. https://www.centerwatch.com/clinical-trials/overview.aspx/. (Accessed on 05/05/2019).

[23] *Peripheral neuropathy - NHS*. https://www.nhs.uk/conditions/peripheral-neuropathy/. (Accessed on 02/11/2019).

[24] *randomForest.pdf*. https://cran.r-project.org/web/packages/randomForest/randomForest.pdf. (Accessed on 04/21/2019).

[25] *Ridge regression*. http://www.few.vu.nl/~wvanwie/Courses/HighdimensionalDataAnalysis/WNvanWieringen_HDDA_Lecture234_RidgeRegression_20182019.pdf. (Accessed on 05/09/2019).

[26] N Clayton Silver and William P Dunlap. "Averaging correlation coefficients: Should Fisher's z transformation be used?" In: *Journal of Applied Psychology* 72.1 (1987), p. 146.

[27] *The Placebo Effect: What Is It?* https://www.webmd.com/pain-management/what-is-the-placebo-effect. (Accessed on 02/13/2019).

[28] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), pp. 267–288.

[29] Alexander H Tuttle et al. "Increasing placebo responses over time in US clinical trials of neuropathic pain". In: *Pain* 156.12 (2015), pp. 2616–2626.

[30] Dennis Wackerly, William Mendenhall, and Richard L Scheaffer. *Mathematical statistics with applications*. Cengage Learning, 2014.

[31] *What is Osteoarthritis?* https://www.arthritis.org/about-arthritis/types/osteoarthritis/what-is-osteoarthritis.php. (Accessed on 02/11/2019).

[32] Hadley Wickham. *dplyr v0.7.8*. https://www.rdocumentation.org/packages/dplyr/versions/0.7.8. (Accessed on 05/01/2019).

[33] Hui Zou and Trevor Hastie. "Regularization and variable selection via the elastic net". In: *Journal of the royal statistical society: series B (statistical methodology)* 67.2 (2005), pp. 301–320.

Appendix

Correlation between corresponding items of MPSQDATA_1 and MPSQDATA_3

item	correlation
item_001	0.72
item_002	0.45
item_003	0.46
item_004	0.53
item_005	0.43
item_006	0.65
item_007	0.29
item_008	0.43
item_009	0.22
item_010	0.28
item_011	0.33
item_012	0.52
item_013	0.43
item_014	0.73
item_015	0.80
item_016	0.29
item_017	0.43
item_018	0.50
item_019	0.63
item_020	0.76
item_021	0.79

item_022	0.55
item_023	0.46
item_024	0.60
item_025	0.56
item_026	0.68
item_027	0.64
item_028	0.50
item_029	0.66
item_030	0.76
item_031	0.67
item_032	0.57
item_033	0.63
item_034	0.42
item_035	0.75
item_036	0.58
item_037	0.20
item_038	0.70
item_039	0.71
item_040	0.60
item_041	0.71
item_042	0.50
item_043	0.28
item_044	0.45
item_045	0.69
item_046	0.53
item_047	0.61
item_048	0.56
item_049	0.61
item_050	0.38
item_051	0.42
item_052	0.50
item_053	0.56
item_054	0.46
item_055	0.28

item_056	0.60
item_057	0.54
item_058	0.35
item_059	0.69
item_060	0.42
item_061	0.22
item_062	0.34
item_063	0.38
item_064	0.38
item_065	0.51
item_066	0.48
item_067	0.37
item_068	0.32
item_069	0.38
item_070	0.39
item_071	0.42
item_072	0.04
item_073	0.24
item_074	0.27
item_075	0.49
item_076	0.49
item_077	0.46
item_078	0.67
item_079	0.23
item_080	0.55
item_081	0.39
item_082	0.53
item_083	0.49
item_084	0.63
item_085	0.42
item_086	0.65
item_087	0.42
item_088	0.64
item_089	0.68

item_090	0.37
item_091	0.61
item_092	0.63
item_093	0.39
item_094	0.34
item_095	0.24
item_096	0.50
item_097	0.30
item_098	0.62
item_099	0.22
item_100	0.36
item_101	0.57
item_102	0.28
item_103	0.49
item_104	0.58
item_105	0.53
item_106	0.27
item_107	0.46
item_108	0.45
item_109	0.64
item_110	0.55
item_111	0.51
item_112	0.50
item_113	0.40
item_114	0.39
item_115	0.65
item_116	0.68
item_117	0.02
item_118	0.40
item_119	0.54
item_120	0.56
item_121	0.49
item_122	0.63
item_123	0.28

item_124	0.47
item_125	0.32
item_126	0.49
item_127	0.63
item_128	0.47
item_129	0.28
item_130	0.37
item_131	0.45
item_132	0.29
item_133	0.67
item_134	0.35
item_135	0.56
item_136	0.39
item_137	0.43
item_138	0.09
item_139	0.12
item_140	0.34
item_141	0.62
item_142	0.47
item_143	0.47
item_144	0.36
item_145	0.42
item_146	0.32
item_147	0.46

Table 6.1: Table listing the correlation of each item between mpsqData1 and mpsqData3.

Monte Carlo simulation with pooling as aggregation method

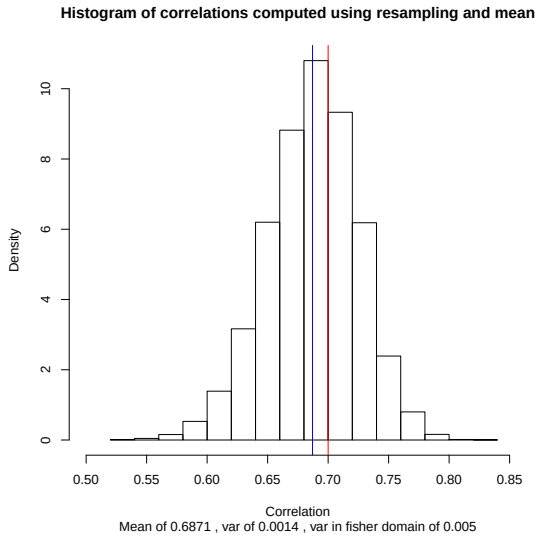


Figure 6.1: Correlation coefficients computed using the average in the Fisher domain in the case of a Monte Carlo cross validation. The expected correlation is 0.7. The observed mean correlation is 0.6871 and the variance in the Fisher domain is 0.005

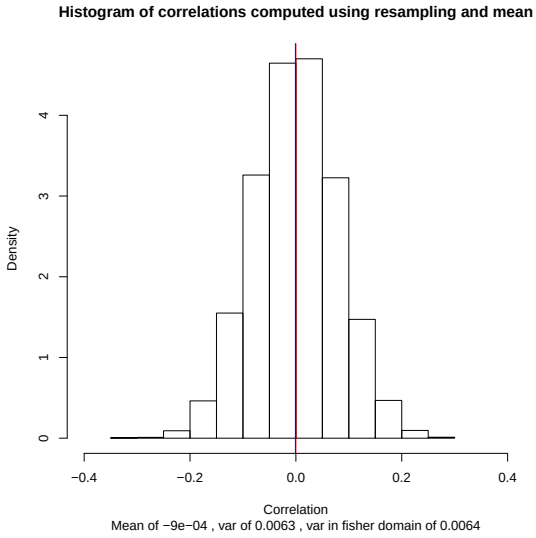


Figure 6.2: Correlation coefficients computed using the pooling method in the case of a Monte Carlo cross validation. The expected correlation is 0. The observed mean correlation is $-9\ 10^{-4}$ and the variance in the Fisher domain is 0.0064.

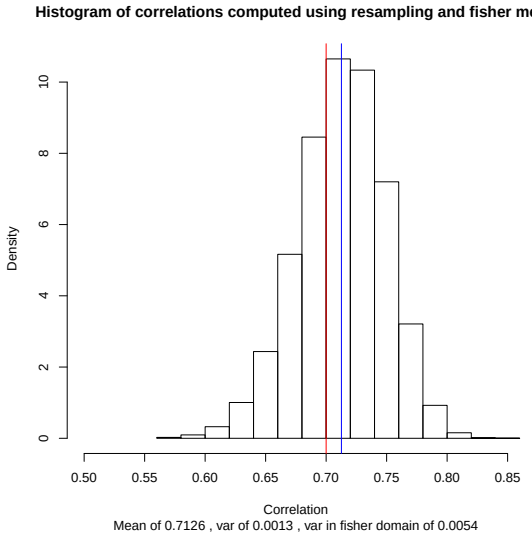


Figure 6.3: Correlation coefficients computed using the average in the Fisher domain in the case of a Monte Carlo cross validation. The expected correlation is 0.7. The observed mean correlation is 0.7126 and the variance in the Fisher domain is 0.0054

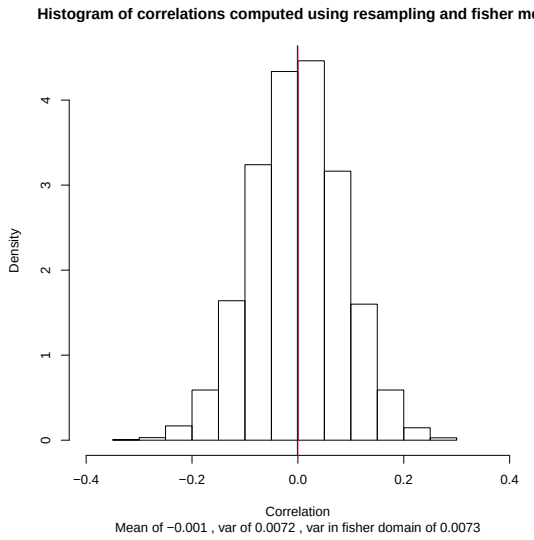


Figure 6.4: Correlation coefficients computed using the pooling method in the case of a Monte Carlo cross validation. The expected correlation is 0. The observed mean correlation is -0.001 and the variance in the Fisher domain is 0.0073.

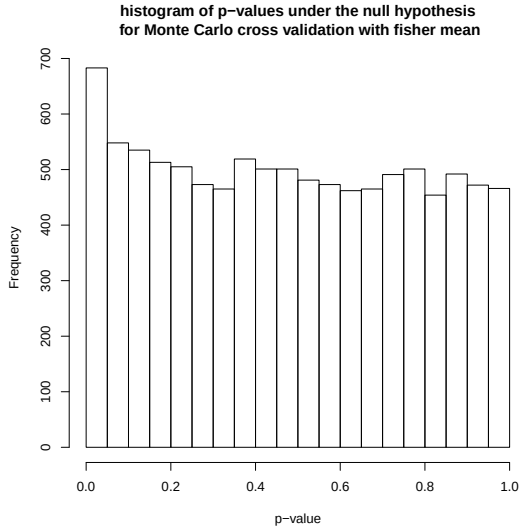


Figure 6.5: p-values computed when using the average in the Fisher domain to compute the correlation coefficients in the case of a Monte Carlo cross validation (expected correlation is 0)

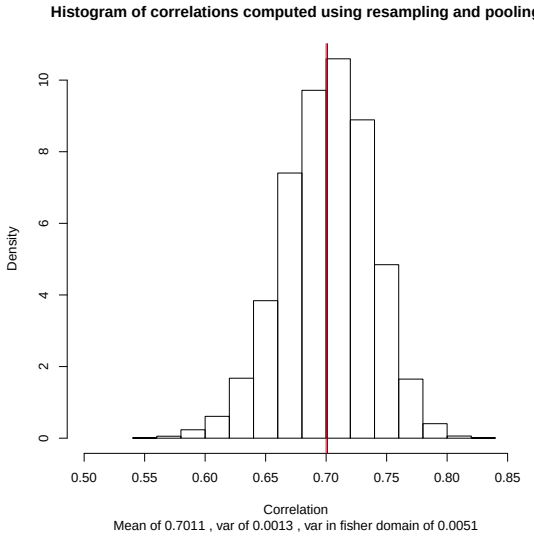


Figure 6.6: Correlation coefficients computed using the pooling method in the case of a Monte Carlo cross validation. The expected correlation is 0.7. The observed mean correlation is 0.7011 and the variance in the Fisher domain is 0.0051

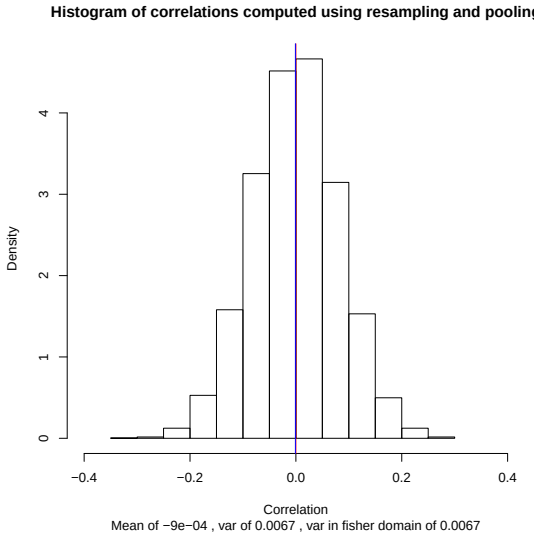


Figure 6.7: Correlation coefficients computed using the pooling method in the case of a Monte Carlo cross validation. The expected correlation is 0. The observed mean correlation is $-9\ 10^{-4}$ and the variance in the Fisher domain is 0.0067.

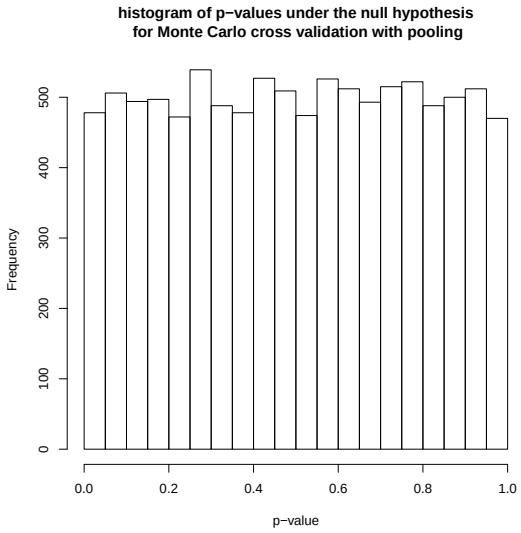


Figure 6.8: p-values computed when using the pooling method to compute the correlation coefficients in the case of a Monte Carlo cross validation (expected correlation is 0)

Stability of feature selection

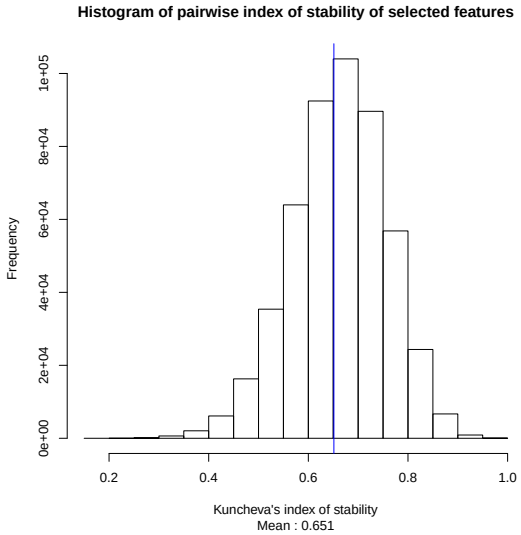


Figure 6.9: Distribution of pairwise Kuncheva's index for 25 features selected using correlation

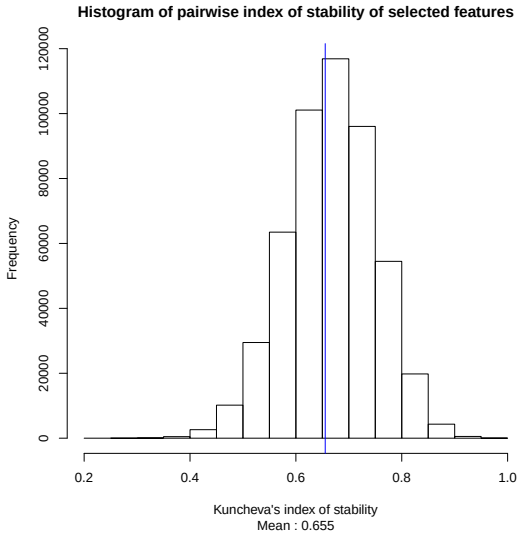


Figure 6.10: Distribution of pairwise Kuncheva's index for 25 features selected using lasso regression

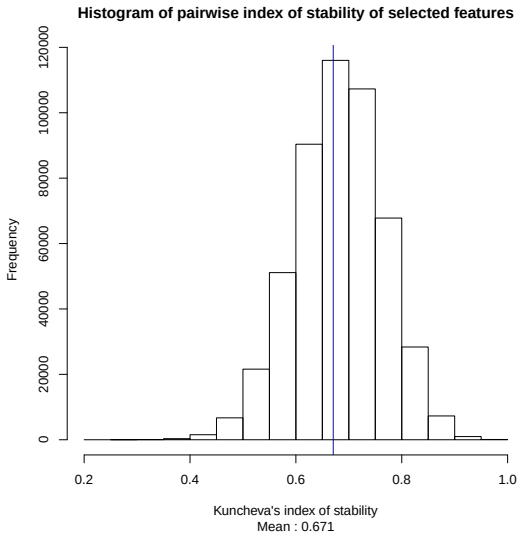


Figure 6.11: Distribution of pairwise Kuncheva's index for 25 features selected using Elastic net.

Facets found in feature selection

Facet	Frequency	Relative frequency
Facet_022	1660.00	553.33
Facet_017	1336.00	445.33
Facet_011	985.00	328.33
Facet_024	1629.00	325.80
Facet_006	1781.00	254.43
Facet_001	2497.00	249.70
Facet_020	1795.00	224.38
Facet_018	1769.00	221.12
Facet_008	626.00	156.50
Facet_014	936.00	133.71
Facet_015	857.00	122.43
Facet_021	758.00	94.75
Facet_002	785.00	87.22
Facet_003	661.00	82.62

Table 6.2: Frequency and relative frequency of the most selected (with correlation) facets. A facet is selected if one of its feature is selected. They are sorted decreasingly by relative frequency. The relative frequency is the sum of the frequencies of its selected features divided by the total number of features belonging to the facet.

Facet	Frequency	Relative frequency
Facet_022	1427.00	475.67
Facet_017	1019.00	339.67
Facet_011	962.00	320.67
Facet_001	3103.00	310.30
Facet_006	1689.00	241.29
Facet_008	920.00	230.00
Facet_024	1000.00	200.00
Facet_018	1301.00	162.62
Facet_015	987.00	141.00
Facet_014	934.00	133.43
Facet_003	998.00	124.75
Facet_004	988.00	123.50
Facet_021	963.00	120.38
Facet_020	950.00	118.75
Facet_002	859.00	95.44

Table 6.3: Frequency and relative frequency of the most selected (with correlation) facets. A facet is selected if one of its feature is selected. They are sorted decreasingly by relative frequency. The relative frequency is the sum of the frequencies of its selected features divided by the total number of features belonging to the facet.

Facet	Frequency	Relative frequency
Facet_022	1467.00	489.00
Facet_011	969.00	323.00
Facet_001	3103.00	310.30
Facet_006	1721.00	245.86
Facet_008	918.00	229.50
Facet_024	1000.00	200.00
Facet_020	1495.00	186.88
Facet_017	556.00	185.33
Facet_018	1361.00	170.12
Facet_015	983.00	140.43
Facet_014	956.00	136.57
Facet_003	997.00	124.62
Facet_004	986.00	123.25
Facet_021	967.00	120.88
Facet_002	866.00	96.22

Table 6.4: Frequency and relative frequency of the most selected (with correlation) facets. A facet is selected if one of its feature is selected. They are sorted decreasingly by relative frequency. The relative frequency is the sum of the frequencies of its selected features divided by the total number of features belonging to the facet.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**