Journal of Statistics and Computer Science
Vol. 1, No. 1, 2022, pp. 55-76

URL : www.arfjournals.com
https://DOI: 10.47509 /JSCS.2022.v01i01.05

Cross-Validation for Supervised Learning with Tuning Parameters

Lyron Winderbaum[1,2] and Inge Koch[1]
[1]*The University of Western Australia*
[2]*The University of South Australia, Australia*

Received April 25, 2022 • Revised June 7, 2022 • Accepted June 25, 2022 • Published July 15, 2022

Abstract: Recent advances in machine learning and data science have led to widespread adoption of complex predictive modelling. Increasing awareness of the 'reproducibility crisis' has led to calls for improved transparency and accountability in scientic reporting. One important aspect of veridical data science is the robust estimation of prediction error. Availability of computational resources has led to cross-validation (CV) as a main tool for such estimation. We consider CV estimation in supervised learning for high-dimensional data, and focus on linear regression and discriminant analysis approaches based on variable selection with direct dimension reduction as well as lasso-type sparsity criteria. We highlight how the same description of a method could in fact apply to any one of several different cross-validation implementations. We outline key principles underpinning good cross-validation practice, several 'pitfall' implementations which subtly violate these principles in different ways as well as a more complex and computationally intensive implementation which does not. We demonstrate the differences in the estimated error resulting from these different implementations with real data relating to endometrial cancer, in the context of high-stakes decision making where accurate and robust estimation of prediction error is critical. We use simulated data to illustrate how these different implementations result in estimators for prediction error with very different properties and relationships to the true prediction error. We call for increased detail in method-reporting, present principles for good practice in the implementation of cross-validation, and make recommendations to guide cross-validation implementation.

Keywords: Cross-Validation, Prediction, Proteomics, Reproducibility.

To cite this article

Lyron Winderbaum & Inge Koch (2022). Cross-Validation for Supervised Learning with Tuning Parameters. *Journal of Statistics and Computer Science*. Vol. 1, No. 1, pp. 89-106. *https://DOI: 10.47509 /JSCS.2022.v01i01.05*

1. Introduction

Machine learning, and in particular supervised learning, has been growing rapidly in popularity not only in mathematical and statistical research but more broadly in application to scientific research. Meanwhile veridical or principled data science has gained in relevance, particularly in the light of the 'reproducibility crisis', see Yu and Kumbier (2020), Baker (2016), Munafo *et al.* (2017). One of the key components of veridical data science is the estimation of predictability, the ability to make accurate predictions for new data. The

empirical rigour with which predictability is quantified in machine learning and specifically in supervised learning applications is largely made possible by cross-validation (CV), see Yu and Kumbier (2020). Improper or inaccurate use of CV can alter the resulting predictability estimate, and various such uses of CV are common in the published scientific literature, particularly when cross-validation is combined with dimension reduction methods, see Molinaro *et al.* (2005), Varma and Simon (2006), Rao *et al.* (2008). These dangers are well understood in some circles, yet persist in others, see Rauschenberger and Baeza-Yates (2020), Guo *et al.* (2017). Classically cross-validation was performed for data with a moderate number of variables to estimate the prediction error by dividing the available data into training and testing and did not include tuning parameters. This process is generally well understood. We will come back to it at the beginning of Section 3. However, as the number of variables increases or exceeds the sample size, some form of variable or feature selection, dimension reduction or sparsity has to be introduced into the supervised learning process in order to be able to arrive at a solution. If a statistical analysis involves a form of variable selection as well as estimation of errors and accurate prediction, then care needs to be taken in how the different components are combined. Various implementations or combinations are feasible and commonly in use, and are often described by statements like 'dimension reduction followed by CV was used'. This lack of precision in the description of the approach affects reproducibility of the results directly and may adversely impact on the analysis and results.

The aim of this paper is to enhance reproducible and veridical research related to cross-validation for supervised learning and to improve communication of such research by clarifying principles underlying CV, by examining the adherence of different approaches to variable selection combined with CV to these principles, and by making recommendations. Our research is motivated by the need for accurate prediction of the occurrence of metastasis in patients diagnosed with endometrial cancer from proteomics mass spectrometry data. For these high-dimension low sample size (HDLSS) proteomics mass spectrometry data, early and accurate prediction of the presence or absence of lymph node metastasis (LNM) is crucial to survival, see Mittal *et al.* (2016). We provide more detail regarding these data later in this section and in Section 4.1.

In the supervised learning context we consider four broad scenarios

1. discriminant analysis (DA) with dimension reduction,
2. linear regression (LR) with dimension reduction,
3. linear regression based on the LASSO, and
4. discriminant analysis with LASSO constraints.

Crucial to all these approaches is the selection of the tuning parameter. In the first two approaches the tuning parameter will refer to the reduced dimension of the data, in the last two it reflects the degree of sparsity. In this paper we use the term variable selection to cover explicit dimension reduction, as achieved with principal component analysis and

other component methods, as well as sparse methods based on the lasso, the ridge idea, or the elastic net. In many of these variable sections approaches the selected variables are weighted combinations of the original variables. These are sometimes also called features.

The first two scenarios are commonly used in statistics and its applications, the third one has also become well established, see Tibshirani (1996), Hastie *et al.* (2009). For the possibly lesser known last scenario see Cai and Liu (2011). Our rationale for focussing on specific, but broad, supervised learning approaches is that we want to facilitate better communication of results and hence want to explain explicitly how each of these incorporates the slightly varying implementations of CV which we examine in this paper. The restriction to the four scenarios listed above does not imply that CV only works for these. Indeed its range of applicability goes far beyond this list.

In Section 2 we start with a description of the principles underpinning CV which relate to 'out-of-sample' testing and an 'internal coherence' in a training/testing and prediction context. With reference to the four scenarios listed above, we describe, in Section 3, different implementations of CV with variable selection, all of which are used in practice. We explain how each of these implementations aligns with the CV principles and discuss the shortcomings as well as decisions and reasoning that would lead to each of these implementations. In Section 4 we illustrate the impact these decisions have on resulting predictability estimates with a real example – our motivating proteomic mass spectrometry imaging data. We complement the real data analysis with simulations based on these data, which comprise training, testing as well as prediction, and we showcase the performance as well as the drawbacks of the different implementations. Our final section includes a discussion of the results and recommendations. Details of the exact implementation of these methods can be found in the Supplementary Material.

The HDLSS proteomics mass data are from patients diagnosed with endometrial cancer, see Mittal *et al.* (2016). They have been measured at the University of South Australia, and we previously worked with their lab and these specific data, see Winderbaum *et al.* (2016). Endometrial cancer is the most common gynaecological malignancy in Australia. The presence of lymph node metastasis (LNM) is a crucial factor in determining treatment and prognosis for endometrial cancer patients, however it is difficult to assess LNM status accurately prior to surgery, see Bendifallah *et al.* (2012). Current standard practice is to remove the lymph nodes along with the primary tumour during surgery. For the approximately 15% of patients with LNM this practice will typically be life-saving, but for the majority of cases, namely those without LNM, this results in unnecessarily invasive surgery which carries additional risks. If the LNM status of a patient could be predicted with suciently low error, prognoses would dramatically improve for patients without LNM as their lymph nodes would not need to be removed. This is a high-stakes example, and so a good demonstration of the point that accurate estimation of the prediction error can be crucial. The data are also high-dimensional, which means that variable selection has to be carried out as part of the statistical analysis. We provide more detailed information on the data in Section 4.1.

We conclude this section with notation used in this paper.

Notation

Let X_i denote random vectors in $\mathbb{R}^d$ with $i = 1, 2, ..., n$;

$\mathbb{X} = [X_1, X_2, ..., X_n]$ the set of X_i or the $d \times n$ data matrix with columns X_k;

$Y = [Y_1, Y_2, ..., Y_n]$ the (row) vector of responses which are class labels in *DA* and continuous response variables in *LR*;

$\mathbb{X}_{[j]}$ with $j = 1, 2, ..., m$ a partition of $\mathbb{X}$ into m disjoint sets;

$\mathbb{X}_{[j,i]}$ with fixed j and $i = 1, 2, ..., w$ a partition of $\mathbb{X}_{[j]}$ into w disjoint sets for fixed j;

$\mathbb{X}_{(0,j)} = \mathbb{X} \setminus \mathbb{X}_{[j]}$ the complement of $\mathbb{X}_{[j]}$ for each j,

$\mathbb{X}_{(0,j,i)} = \mathbb{X} \setminus (\mathbb{X}_{[j]} \cup \mathbb{X}_{[j,i]})$ the complement of $\mathbb{X}_{[j]} \cup \mathbb{X}_{[j,i]}$;

$Y_{[j]}$, $Y_{(0,j)}$, $Y_{[j,i]}$ subvectors of Y corresponding to $\mathbb{X}_{[j]}$, $\mathbb{X}_{(0,j)}$, $\mathbb{X}_{[j,i]}$ respectively;

π_ν variable selection transform applied to the X_i, with

$\nu = k$ for dimension reduction, k the dimension of the selected variables;

$\nu = \lambda$ for the lasso, λ controlling the sparsity;

τ a classication or linear regression rule assigning (transformed) random vectors to classes in *DA* and to a continuous response in *LR*.

$\tau|\mathbb{X}^0$ the classication rule trained on some data $\mathbb{X}^0$;

$||X||_\gamma$ the l_γ norm of a vector X with $\gamma = 1, 2$ and ∞.

Unless a distinction is necessary we will typically refer to the transform π_ν and only refer to the two cases when we want to detail how they differ.

2. Principles of Cross-Validation

In the following, we use the term ‘rule’ to describe a function with parameters determined by the dataset it is constructed from, and using these parameters, the function is applied to the same or other data to obtain ‘responses’. In machine learning language the steps of supervised learning can be categorised as:

- **training/learning:** constructing a rule by using data to determine its parameters,
- **testing:** applying a rule to data with known responses, and
- **prediction:** applying a rule to data with unknown responses.

If we want to distinguish between rules whose parameters are determined from different subsets of the data, we may do so by referring explicitly to the subset that is used, so a CV-rule will refer to the rule and the (sub) set from which the parameters are determined. ‘Full-rule’ will refer to the rule with parameters obtained from all data. Some authors use the term ‘prediction’ to describe a testing step, but we will exclusively use ‘prediction’ for the case when the response is not known. So in the proteomics example this means making a prediction of LNM status for a patient when their true LNM status is not known. In the construction of new rules, the prediction step is neglected and often training and testing steps are alternated to optimise performance. Ultimately from the

application point of view the goal of this optimisation is to arrive at a rule which will perform well in prediction.

The full-rule is typically used for prediction. For a given dataset the prediction error E^{pred} in this paper is based on the distribution of the data and on the full-rule trained on the data it refers to. That is, the parameters of the rule used in E^{pred} are those obtained on the specific dataset. So E^{pred} is the theoretical error when applying the full-rule to observations with unknown responses. If two datasets differ, then their corresponding E^{pred} will also differ even if the data come from the same distribution. Calculation of E^{pred} does require knowledge of the population distribution, which is typically not the case in practice. The CV error E^{cv} is calculated directly from the data without requiring knowledge of the distribution of the data. It is commonly used as an estimator of E^{pred}. In addition, E^{cv} may be employed to determine tuning parameters, and care needs to be taken when E^{cv} is used in this latter way. We provide explicit expressions for E^{cv} in the following sections. Note that although we will compare different estimators for E^{pred}, doing so on the basis of asymptotic behaviour is beyond the scope of this work. There has been some work in this direction, see Kearns and Ron (1999), Zhang (2003), Rao *et al.* (2008), but it remains a space for further development.

Although E^{cv} is an estimator of the unknown population quantity E^{pred}, the parameters of each of the rules contributing to determine E^{cv}, the CV-rules, and those of the full-rule used for prediction may differ. The mechanism by which E^{cv} estimates E^{pred} is therefore not through a specific rule, but rather through the process or algorithm a rule's parameters are determined from data. The first principle of CV is that the process used to train the CV-rules in the calculation of E^{cv} should be the same as the process used to train the full-rule for prediction. We refer to this idea as 'internal coherence'. Although this principle is used in the literature, there does not seem to be a specific terminology for this concept. In the context of variable selection, this principal implies that the value of the tuning parameter is part of the training of model-tting process. We will return to this point in more detail.

If the full-rule is used to make predictions for the same data that was used to train it, this results in the classication error, E^{class}. E^{class} is a commonly reported and widely used quantity, and is a useful measure of model fit. However E^{class} is typically an underestimate of the prediction error E^{pred}, particularly with increasing dimensionality of the data. The second principle of CV, the 'out-of-sample' idea, is motivated by correcting this underestimation. It is a classical idea, used far more broadly than just in CV, and states that different data should be used to train and test a rule (Lachenbruch and Mickey, 1968; Stone, 1974, 1978, Efron, 1983). This principle is violated when information about the test data is (often inadvertently) "leaked" into the training/ model-fitting process. Two common ways this can happen is through some preprocessing of the data (such as dimension reduction) prior to CV, or through the criteria used to choose a tuning parameter.

Before we detail CV implementations, we consider the steps in our four scenarios in light of the different error calculations referred to this section. For scenarios 1 and 2 with

explicit dimension reduction transformation π_k and a chosen dimension $k \leq d$ the estimation of the rule includes a separate step which is carried out before the actual rule is applied. Consider the steps

$$\mathbb{X} \to \pi_{\mathbf{k}}(\mathbb{X}) \to \mathfrak{r}_k(\pi_k(\mathbb{X})), \text{ with}$$
$$k^* = \text{argmin } E_k^{class} \text{ or argmin } E_k^{lreg} \text{ respectively,} \quad (1)$$

where $\mathfrak{r}_k$ is the DA or LR rule induces from $\mathfrak{r}$ and applied to the k-dimension reduced data $\pi_k(\mathbb{X})$. For scenarios 3 and 4 based on sparsity with the lasso constraint, $\lambda > 0$, the two steps shown in (1) are combined into a single step when the parameters of the rule are calculated for a given value of λ and no prior reduction of the variables is carried out, as the shrinkage is a consequence of the estimation of the rule's parameters for a given constraint λ. For these scenarios we consider the step

$$\mathbb{X} \to \mathfrak{r}_\lambda(\mathbb{X}), \text{ with}$$
$$\lambda^* = \text{argmin } E_\lambda^{class} \text{ or argmin } E_\lambda^{lreg} \text{ respectively.} \quad (2)$$

Here the notation $\mathfrak{r}_\lambda$ is used as a placeholder for $\mathfrak{r} \circ \pi_\lambda$ to indicate that a positive value λ is included directly into the formulation of the rule. The 'how' is made explicit in the error calculations of the table below. Unlike the sequence in (1), the rule (2) adds the sparsity parameter λ as a constraint to the rule $\mathfrak{r}$. The different error calculations in *DA* and *LR* are listed in Table 1 which refers to the coefficient vector β in the regression setting, to W, the sum of the class covariance matrices, and ξ, a sparse solution in *DA* which approximates $W^{-1}\left(\bar{X}_1 - \bar{X}_2\right)$. (Note that W^{-1} may not exist.) The expression for E_λ^{class} in Table 1 is a reformulation of (3) in Cai and Liu (2011).

Table 1: Regression and Classication Errors for Data with Variable Selection

	LR	*DA*
data	$E^{lreg} = \|\|Y - \beta^T\mathbb{X}\|\|_2^2$	$E^{class} = \|\|Y - \mathfrak{r}(\mathbb{X})\|\|_1$
dim red	$E^{lreg} = \|\|Y - \beta^T\pi_k(\mathbb{X})\|\|_2^2$	$E_k^{class} = \|\|Y - \mathfrak{r}(\pi_k(\mathbb{X}))\|\|_1$
sparse	$E_\lambda^{lreg} = \|\|Y - \beta^T\mathbb{X}\|\|_2^2 + \lambda\|\|\beta\|\|_2$	$E_\lambda^{class} = \|\|Y - \mathfrak{r}(\pi_\lambda(\mathbb{X}))\|\|_\infty + \lambda\|\|\xi\|\|_1$ $\xi \approx W^{-1}\left(\bar{X}_1 - \bar{X}_2\right)$

The process described above has focussed on variable selection and error calculation for a given tuning parameter, but in the presence of tuning parameters, we require another step, namely

- the optimisation step which determines the value ν^* of the tuning parameter.

For predictions, the rule involving variable selection $\mathfrak{r}_\nu\pi_\nu$ is applied to $\mathbb{X}$ and one optimises the tuning parameter over $\mathbb{X}$ which is regarded as training and testing set. As the resulting error is an underestimate of the true prediction error, we turn to ways of improving on this estimate using cross-validation.

3. CV Implementations

Traditionally, that is when there are no tuning parameters to be chosen, *m*-fold CV is performed in the following way:

1. choose a partitioning of the dataset $\mathbb{X}$ into m disjoint sets $\mathbb{X}_{[j]}$ ($j = 1, 2, ..., m$),
2. train a rule, $\mathfrak{r}_j = \mathfrak{r}|\mathbb{X}_{(0,j)}$, on each of the *m* subsets $\mathbb{X}_{(0,j)} = \mathbb{X} \setminus \mathbb{X}_{[j]}$;
3. test the rule $\mathfrak{r}_j$ on $\mathbb{X}_{[j]}$: evaluate $\mathfrak{r}_j(\mathbb{X}_{[j]})$ for each $j = 1, 2, ..., m$,
4. combine these responses $\mathfrak{r}_j(\mathbb{X}_{[j]})$ and calculate E^{cv} for the $\mathbb{X}_{[j]}$ using the *DA* and *LR* error criterion respectively.

The CV error E^{cv} depends on the size and choice of the partitioning subsets $\mathbb{X}_{[j]}$ that are used in the testing step. In *m*-fold CV the m classication rules $\mathfrak{r}|\mathbb{X}_{(0,j)}$ will differ since they are constructed on overlapping but different subsets $\mathbb{X}_{(0,j)}$ of $\mathbb{X}$. With increasing *m* the computational effort increases, but it is generally assumed that the rules $\mathfrak{r}|\mathbb{X}_{(0,j)}$ become more similar to $\mathfrak{r}|\mathbb{X}$ as $\mathbb{X}_{(0,j)}$ gets closer to $\mathbb{X}$. *LOO* or *n*-fold CV refers to the special case for which $m = n$ and $\mathbb{X}_{[j]} = \{X_j\}$ refers to the j^{th} single observation. If computational efficiency is not a concern, *LOO*-CV is a natural choice for classication and regression problems because of the intuition that the rules $\mathfrak{r}|\mathbb{X}_{(0,j)}$ will be closer to the rule used for prediction $\mathfrak{r}|\mathbb{X}$. In addition, the partitioning of the data in *LOO*-CV is unique and so is simpler to reproduce. We will exclusively use *LOO*-CV in our analyses, while keeping our notation general.

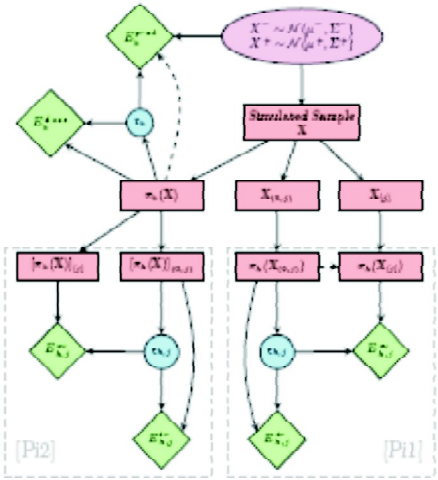

Figure 1: Flowchart illustrating, in the dimension reduction setting with π_k, the calculation of key quantities represented by green diamonds. In particular, the difference between [Pi1] and [Pi2] is illustrated in the red rectangles. Notice that $E^{cv}_{k,j}$ and $E^{tr}_{k,j}$ are represented twice, and that these two different versions of each correspond to that calculated using [Pi1] and [Pi2] respectively. The magenta ellipse represents the population distributions (which are known when dealing with simulated data). Blue circles represent trained discriminant rules. Arrows $A \rightarrow B$ can be interpreted as '*B* is calculated from *A*'. Dashed arrows such as in particular the small dashed arrow from the training data to the testing data in [Pi1] represents that the dimension reduction transform π_k in $\pi_k(\mathbb{X}_{[j]})$ is inherited from $\pi_k(\mathbb{X}_{(0,j)})$. Note that in [Pi2] there is no such inheritance of π_k as dimension reduction is done prior to splitting the data into subsets, and this is the essence of the dierence between [Pi1] and [Pi2].

3.1 CV implementations and Pitfalls

In this section we initially focus on the dimension reduction transform π_k rather than on the more general variable selection. The two choices we put forward also apply to the lasso though more explicitly at the level at which the rule is calculated, which has direct implications for the CV error calculations in (8) and (7).

High-dimension low sample (HDLSS) size data typically require dimension reduction or feature extraction as an integral part of any analysis. The last decade saw rapid advances in the development of such integrated methods in response to the increased prevalence of HDLSS across many areas of data science see Filzmoser *et al.* (2012); Fan and Lv (2010); Jimenez-Rodriguez *et al.* (2007). Typically analysis of HDLSS data will involve first reducing the dimensionality of the data, either by working with a range of dimensions, or by finding some 'optimal' value of the tuning parameter k, and then performing more standard analysis on the dimension-reduced data. The choice of a single k^*, is in fact the selection of a tuning parameter and should be treated in this way.

We describe four implementations of CV by two decisions annotated [Pi1]/[Pi2] and [CV1]/[CV2] respectively which essentially offer two options for how the transform π_ν is integrated into the CV, and for how ν^* is chosen respectively.

The [Pi1]/[Pi2] choice is outlined schematically below, highlighting that the difference between the two is the order in which the partitioning into training sets $\mathbb{X}_{(0,\,j)}$ and the dimension reduction to k dimensions are accomplished:

[Pi1]

$$\mathbb{X} \to \mathbb{X}_{(0,\,j)} \to \pi_\nu(\mathbb{X}_{(0,\,j)}); \tag{3}$$

[Pi2]

$$\mathbb{X} \to \pi_\nu(\mathbb{X}) \to [\pi_\nu(\mathbb{X})]_{(0,\,j)}. \tag{4}$$

For π_k and [Pi1] the data are split into training sets $\mathbb{X}_{(0,\,j)}$ first and then the transformation π_k is applied to each $\mathbb{X}_{(0,\,j)}$. In [Pi2], the transformation π_k is applied to the full data $\mathbb{X}$, and the resulting k-dimensional data $\pi_k(\mathbb{X})$ are split into training sets as indicated above by the subscript $(0, j)$ of $[\pi_k(\mathbb{X})]_{(0,\,j)}$. The two sequences typically produce different training sets, so in general $\pi_k(\mathbb{X}_{(0,\,j)}) \neq [\pi_k(\mathbb{X})]_{(0,\,j)}$.

This process is illustrated in Figure 1, specifically by the red rectangles therein. We first introduce general notation for these training $\mathbb{X}^{tr}_{k,\,j}$ and testing $\mathbb{X}^{test}_{k,j}$ sets, and define these when either [Pi1] or [Pi2] are chosen:

$$\mathbb{X}^{tr}_{kj} = \begin{cases} \pi_k(\mathbb{X}_{(0,\,j)}) & \text{if [Pi1] chosen} \\ [\pi_k(\mathbb{X})]_{(0,\,j)} & \text{if [Pi1] chosen} \end{cases} \tag{5}$$

$$\mathbb{X}^{test}_{k,j} = \begin{cases} \pi_k(\mathbb{X}_{[\,j]}) & \text{if [Pi2] chosen} \\ [\pi_k(\mathbb{X})]_{[\,j]} & \text{if [Pi2] chosen} \end{cases}. \tag{6}$$

When the transform π_k is constructed and applied to the same data, *i.e.* $\pi_k(\mathbb{X})$, this notation is unambiguous. However for [Pi1], the transform π_k that is used in $\pi_k(\mathbb{X}_{[j]})$ of (6) is the same as in (5). This inheritance of π_k is represented in Figure 1 by the small dashed arrow between these. Note that there is a similar inheritance of π_k in the population case when the dimension reduction constructed on the sample is applied to the population distribution in order to calculate the prediction error.

Since [Pi2] with π_k uses the full data in the dimension reduction step, it allows for information about the testing data $\mathbb{X}_{[j]}$ to 'leak' into the training data $\mathbb{X}^{tr}_{v,j}$, thereby contaminating the 'out-of-sample' principle. This is probably the most common pitfall when implementing CV with dimension reduction.

Figure 1 schematically outlines how both $E^{tr}_{v,j}$ and $E^{cv}_{v,j}$ can be calculated using either [Pi1] or [Pi2]. More precisely these are calculated as:

$$E^{tr}_{k,j} = ||Y_{(0,\,j)} - \mathfrak{r}_{k,\,j}(\mathbb{X}^{tr}_{k,j})||, \tag{7}$$

$$E^{cv}_{k,j} = ||Y_{[\,j]} - \mathfrak{r}_{k,j}(\mathbb{X}^{test}_{k,j})||, \tag{8}$$

where $\gamma = 1$ for DA and $\gamma = 2$ for LR, $r_{k,j} = r|\mathbb{X}^{tr}_{k,j}$, Y is the vector of known responses, and $Y_{[j]}$, $Y_{(0,j)}$ are subvectors of Y corresponding to $\mathbb{X}_{[j]}$, $\mathbb{X}_{(0,j)}$ respectively for each j.

For variable selection based on the lasso, the [Pi1]/[Pi2] split is just as important as in dimension reduction, however this split becomes apparent or 'visible' only in the equivalent expressions of (7) and (8). In the case of [Pi1],

[Pi1] and LR:

$$\begin{aligned} E^{tr}_{\lambda,j} &= ||Y_{(0,j)} - \beta^T_{(0,j)}\mathbb{X}_{(0,j)}||^2_2 + \lambda||\beta_{(0,j)}||_1 \\ E^{cv}_{\lambda,j} &= ||Y_{[j]} - \beta^T_{(0,j)}\mathbb{X}_{[j]}||^2_2 + \lambda||\beta_{(0,j)}||_1 \end{aligned} \tag{9}$$

DA:

$$\begin{aligned} E^{tr}_{\lambda,j} &= ||Y_{(0,j)} - \mathfrak{r}(\pi_\lambda(\mathbb{X}_{(0,j)}))||_\infty + \lambda||\xi_{(0,j)}||_1 \\ E^{cv}_{\lambda,j} &= ||Y_{[j]} - \mathfrak{r}(\pi_\lambda(\mathbb{X}_{[\,j]}))||_\infty + \lambda||\xi_{(0,j)}||_1. \end{aligned} \tag{10}$$

The subscript $(0, j)$ used with β and ξ 'indicates that these values have been calculated from $\mathbb{X}_{(0,j)}$. Further the DA vector $\xi_{(0,j)}$ is calculated as in the Table 1, but based on the relevant quantities of $\mathbb{X}_{(0,j)}$.

In the case of [Pi2], the vectors β in LR and ξ in DA are obtained from $\mathbb{X}$, so as in Table 1, and are then applied to $Y_{[j]}$ in the calculation of $E^{cv}_{\lambda,j}$ resulting in

[Pi2] and LR:

$$\begin{aligned} E^{tr}_{\lambda,j} &= ||Y_{(0,j)} - \beta^T\mathbb{X}_{(0,j)}||^2_2 + \lambda||\beta||_1 \\ E^{cv}_{\lambda,j} &= ||Y_{[\,j]} - \beta^T\mathbb{X}_{[\,j]}||^2_2 + \lambda||\beta||_1 \end{aligned} \tag{11}$$

DA:

$$\begin{aligned} E^{tr}_{\lambda,j} &= ||Y_{(0,j)} - \mathfrak{r}(\pi_\lambda(\mathbb{X}_{(0,j)}))||_\infty + \lambda||\xi||_1 \\ E^{cv}_{\lambda,j} &= ||Y_{[\,j]} - \mathfrak{r}(\pi_\lambda(\mathbb{X}_{[\,j]}))||_\infty + \lambda||\xi||_1 \end{aligned} \tag{12}$$

We define [CV1] and [CV2] in the same way for all four scenarios by:
[CV1]

$$\nu^* = \min\left[\arg\min_{\nu} \sum_j E^{cv}_{\nu,j}\right] \text{ and} \tag{13}$$

$$E^{cv} = \frac{1}{n}\sum_j E^{cv}_{\nu^*,j} \text{ where } \nu^* \text{ is as in (13),} \tag{14}$$

[CV2]

$$\nu_j^* = \min\left[\arg\min_{\nu} \sum_j E^{tr}_{\nu,j}\right] \text{ and} \tag{15}$$

$$E^{cv} = \frac{1}{n}\sum_j E^{cv}_{\nu^*,j} \text{ where } \nu_j^* \text{ is as in (15).} \tag{16}$$

In [CV1] we obtain a single ν^* based on $\Sigma_j E^{cv}_{k,j}$ by (13). The rule $\mathfrak{r}_{\nu,j}$ in (8) – (12) contains information from the test data $\mathbb{X}^{test}_{\nu,j}$ if [Pi2] is used. With [Pi1], each $E^{cv}_{\nu,j}$ satises the 'out-of-sample' principle as information about the test data is not used to train the rule. However, $E^{cv}_{\nu,j}$ does contain information about the test data, as per (8) – (12). Since selection of ν^* is part of training, E^{cv} (which uses ν^*) does not satisfy the 'out-of-sample' principle due to this use of test data in the selection of ν^*.

In [CV2] a different ν_j^* is chosen for each of the m sets $\mathbb{X}_{[j]}$ using $E^{tr}_{\nu,j}$. As long as [Pi1] is used, $E^{tr}_{\nu,j}$ does not contain any information about the test data $X^{test}_{\nu,j}$ but, unlike [CV1], the chosen number of dimensions ν_j^* does not contain any information about the test data. However [CV2] violates the 'out-of-sample' principle for a different reason: $E^{tr}_{\nu,j}$ is calculated by testing on the same data used to train the rule in (8) – (12) which violates the idea of testing on data that is 'out-of-sample' *i.e.* data other than the training data.

3.2 Prediction

Ultimately we are interested in prediction, for which we need a single ν^*. For [CV1], we use the same ν^* defined in (13). However since [CV2] calculates m values ν_j^*, we use, as appropriate, the four classication or regression errors which are displayed in the last two rows of Table 1. We now choose a single value for prediction:

$$\nu^* = \min\left[\arg\min_{\nu} \mathbb{E}^{lreg/class}_{\nu}\right] \text{ where } \mathbb{E}^{lreg/class}_{\nu} \text{ is as in Table 1,} \tag{17}$$

where $\mathbb{E}^{lreg/class}$ is either $\mathbb{E}^{lreg}$ or $\mathbb{E}^{class}$ as appropriate. Note that (17) is independent of [Pi1]/ [Pi2] since the dimension reduction $\pi_\nu(\mathbb{X})$ uses the full data $\mathbb{X}$, while (13) will produce different results for [Pi1] and [Pi2].

The principle of 'internal coherence' specifies that the process used to construct the full-rule from $\mathbb{X}$ should be the same as the process used to construct a CV-rule from a subset $\mathbb{X}_{(0,j)}$. In [CV1] the full-rule for prediction is calculated using CV on the full data $\mathbb{X}$.

If [CV1] were to satisfy the 'internal coherence' principle, a CV-rule would be calculated using CV on a subset $\mathbb{X}_{(0,j)}$. The fact that [CV1] does not do this, and, instead, uses $\mathbb{X}_{[j]}$ in the choice of v^* for the CV-rules not only violates the 'out-of-sample' principle, but also violates the 'internal coherence' principle.

At first glance [CV2] seems to satisfy the 'internal coherence' principle. However, since v^* is calculated from $\mathbb{X}$ in (17) using the dimension reduction $\pi_v(\mathbb{X})$, in order to satisfy the 'internal coherence' principle, v_j^* would need to be calculated from $\mathbb{X}_{(0,j)}$ in (15) using the dimension reduction $\pi_v(\mathbb{X}_{(0,j)})$. When [Pi1] is used, this is the case and the 'internal coherence' principle is satised, however when [Pi2] is used the dimension reduction $[\pi_v(\mathbb{X})]_{(0,j)}$ is used instead and so [Pi2]-[CV2] does not satisfy the 'internal coherence' principle.

In summary: In the choice between [Pi1][/Pi2], the choice [Pi1] is the preferred option, in particular for the lasso-based solutions. And of the four implementations discussed so far [Pi1]-[CV2] is the only one that satises the 'internal coherence' principle, and although each rule trained and tested in the process of calculating E^{cv} with [Pi1]-[CV2] preserves the 'out-of-sample' principle, the criterion [CV2] used to choose v^* does not, as it involves training and testing on the same data. In the next section we address this remaining issue.

3.3 Double CV

Filzmoser *et al*. (2009) proposed a 'repeated double CV' based on Stone (1974) for a partial least square setting with applications to chemometrics. We extend their ideas to the four scenarios in classication and regression listed in the introduction and complement their empirical workflow with the more formal setting based on the ideas of reproducibility and adherence to CV principles. As mentioned at the end of the previous section, [Pi1] is the more appropriate of the two versions. It remains to adapt the CV-version, which we do below and we refer to this Double CV (D-CV) approach by [Pi1]-[CV3].

D-CV extends the [Pi1] implementations by nesting a second CV-partitioning within the first as follows.

$$\mathbb{X} \xrightarrow[\text{loop}]{\text{Outer}} \mathbb{X}_{(0,j)} \xrightarrow[\text{loop}]{\text{Inner}} \mathbb{X}_{(0,j,i)}.$$

The outer first partitioning remains the same as previously discussed: the $\mathbb{X}_{[j]}$ partition $\mathbb{X}$ into m sets. The inner second partitioning of each $\mathbb{X}_{(0,j)}$ we denote by $\mathbb{X}_{[j,i]}$. It partitions $\mathbb{X}_{(0,j)}$ into m sets, unless LOO is used both times. We let $\mathbb{X}_{(0,j,i)} = \mathbb{X} \setminus (\mathbb{X}_{[j]} \cup \mathbb{X}_{[j,i]})$ be the complement of $\mathbb{X}_{[j,i]}$ within $\mathbb{X}_{(0,j)}$.

Analogous to the calculations in Section 3.1, we calculate CV errors $E^{cv}_{\nu,j,i}$ for the responses $Y_{[j,i]}$ corresponding to the partition $\mathbb{X}_{[j,i]}$ in the inner loops by

$$E^{cv}_{\nu,j,i} = \begin{cases} \left\| Y_{[j,i]} - \mathfrak{r}_{\nu,j,i}(\pi_\nu(\mathrm{X}_{[j,i]})) \right\|_\gamma & \text{for dimension reduction} \\ \left\| Y_{[j,i]} - \beta^T_{\nu,j,i}\mathrm{X}_{[j,i]} \right\|_2^2 + \nu \left\| \beta_{(0,j,i)} \right\|_1 & \text{for LR and lasso, } \nu = \lambda \\ \left\| Y_{[j,i]} - \mathfrak{r}(\pi_\nu(\mathrm{X}_{[j,i]})) \right\|_\infty + \nu \left\| \xi_{(0,j,i)} \right\|_1 & \text{for DA and lasso, } \nu = \lambda \end{cases}, \tag{18}$$

where $\gamma = 1$ or 2 corresponding to DA and LR, and with the adjusted subscript notation for the vectors β and ξ.

Next we define the new CV implementation based on (18) by:

[CV3]

$$\nu_j^* = \min\left[\arg\min_{\nu} \sum_i \mathbb{E}^{cv}_{\nu,j,i}\right] \text{ and} \tag{19}$$

$$E^{cv} = \frac{1}{n}\sum_j \mathbb{E}^{cv}_{\nu_j^*,j} \text{ , where } \nu_j^* \text{ is as in (19).} \tag{20}$$

The inner second partitioning $\mathbb{X}_{[j,i]}$ is used in (19) to obtain a value ν_j^* for each $\mathbb{X}_{(0,j)}$. Equipped with these ν_j^* we return to the outer first partition and calculate $\mathbb{E}^{cv}_{\nu_j,j}$ separately for the three cases as done in (8) to (10), at the corresponding value $\nu_j = \nu_j^*$. Using the expressions $\mathbb{E}^{cv}_{\nu_j^*,j}$ yields (20).

The E^{cv} in (20) is similar to (16) but chooses ν_j^* in a different way, namely as in (19). By this choice, E^{cv} does not test and train on the same data at any stage. By using a second, nested or inner, partitioning to calculate $E^{cv}_{\nu,j,i}$, we ensure that each ν_j^*, when chosen as in (19), does not contain any information about $\mathbb{X}_{[j]}$, and so E^{cv} of (20) does satisfy the 'out-of-sample' principle, unlike the E^{cv} of (14).

For prediction, [CV3] uses only the first partitioning and not the second to calculate ν^*as in (13) *i.e.* [Pi1]-[CV1]. Unlike [CV1], in [CV3] the number of dimensions chosen for prediction is different to that used in the calculation of E^{cv}. This difference allows for [CV3] to satisfy the 'internal coherence' principle (which [CV1] did not satisfy). Specifically for prediction [CV3] uses ν^* which is calculated from $\mathbb{X}_{(0,j)}$ as per (15), while in the calculation of E^{cv}, ν_j^* is calculated in the same way, but from $\mathbb{X}_{(0,j,i)}$ (as per (19)) instead of $\mathbb{X}_{(0,j)}$. So [Pi1]-[CV3] or D-CV is the implementation which satises both principles of CV.

4. Applications to Data

We focus on one specific application – our motivating proteomics mass spectrometry imaging (MSI) data from subjects with endometrial cancer. These HDLSS data arise from two classes. We consider dimension reduction transforms for these data, so focus on the first scenario listed in Section 1. Within this scenario we do not restrict the dimension reduction transform to the common principal component analysis (PCA), but also include a canonical correlation analysis (CCA) based dimension reduction. This is done partly to familiarise readers with this approach which is often more powerful than the PCA-based reduction and partly to provide comparisons.

4.1 Proteomics Data

We begin with a brief description of the proteomics MSI data which is based on Winderbaum *et al.* (2015, 2016) and references therein. A recent development, matrix assisted laser

desorption ionization (MALDI), which is applicable to the characterisation of tissue sections and enables the quantication and spatial expression profiling of thousands of peptides, the building stones of proteins, within and between tissues, was used to acquire mass spectrometry profiles from tissue sections of patients. MALDI-MSI became part of the routine research practice in proteomics labs in the early years of this century, however, it took some time before its potential in the discovery of new drugs and cancer biomarkers was proved, and good and efficient computational methods for MALDI-MSI are still being researched and are currently still lagging behind the success of the method in the laboratory.

For the data we consider in this analysis, a pathologist annotated the tumour regions in two tissue samples of patients with endometrial cancer. MALDI-MSI data were acquired in the annotated regions in the mass range of 800-4000 Da in the lab at the University of South Australia where the first author was a research associate at the time of the data collection. Since mass is a continuous variable, we are dealing with functional data. In practice, however, the spectra or profiles are observed at discrete mass values which may vary from patient to patient. We applied preprocessing steps including baseline subtraction and peak-picking to the data. In the next step we binned the spectra into bins of width 0.25 Da and only included those bins in the analysis that contained at least one peak. This step created the same bins for all patients and further reduced the number of masses (about 170K) of the original data dramatically – here to 4582 variables. In this analysis, we regard the mass bins as the variables. For each patient we averaged the spectra leading to (proteomics) MSI data that consist of 4582 mass variables, the dimension of the input data, for each of $n = 43$ patients with endometrial cancer, 16 of these patients are LNM positive, and the remaining 27 are LNM negative.

The aim in the classication of the LNM status of patients with endometrial cancer is to determine the mass variables that are most strongly related to discriminating between the LNM positive and LNM negative patients. We typically also want to rank these variables by their ability to discriminate or effectiveness in achieving good discrimination between the two groups, and finally one wants to relate the masses to biomarkers that are responsible for the onset of LNM.

The primary aim of the analysis presented in this paper is to compare different approaches, including their performance in determining a list of discriminating masses. A closer inspection of such a list and the link of the masses to potential biomarkers is beyond the scope of this research, but has been considered in Mittal *et al.* (2016).

Below we describe and interpret the performance of the different CV implementations we presented in Section 3 using the notation we introduced there.

Table 2 shows the CV errors E^{cv} and the optimal number of dimensions k^* for each of the five different CV implementations resulting from choosing [Pi1] or [Pi2] and [CV1] or [CV2]. These calculations are repeated using PCA or CCA for dimension reduction. References providing precise details for these dimension reduction methods are provided in the supporting information.

Table 2: CV error E^{cv} and associated 'optimal' number of dimensions k^* using each of the five different CV implementations, with both PCA and CCA, on the proteomics data.

		[CV2]		[CV1]		[CV3]
		[Pi1]	[Pi2]	[Pi1]	[Pi2]	[Pi1]
E^{cv}	CCA	0.47	0.16	0.26	0.14	0.30
	PCA	0.44	0.44	0.33	0.21	0.35
k^*	CCA	24-38	12-18	15	16	3-38
	PCA	22-35	23-33	4	35	4-35

Note that in all cases, values of k considered were restricted to the range 2-38. For [CV2] and [CV3] multiple k^* (*i.e.* k_j^*) values are calculated for each value of E^{cv}, while for [CV1] a single k^* value is calculated. In Table 2, the range of k_j^* values is reported (min-max) for [CV2] and [CV3] while for [CV1] the k^* values themselves are reported.

In terms of effectiveness of the performance of the different CV implementations, we observe that, in 4 of the 5 cases, dimension reduction with CCA works better than dimension reduction based on PCA. More parsimonious models, which are characterised by smaller k^* values, however, do not always result in implementations with small CV errors. In the Discussion Section we focus more explicitly on an interpretation of the different implementations with respect to adherence to the internal coherence and out-of-sample principles of these implementations.

Simulated Data

We simulate datasets from two multivariate normal distributions with class means and covariance matrices equal to the sample means and sample covariance matrices of the two LNM classes of the endometrial cancer data. We sample a number of observations from each distribution equal to the number of observations in each LNM class in the real data: 27 for the LNM negative class and 16 for the LNM positive class. We simulate 200 such datasets.

For each simulated dataset, we repeat the calculation of E^{cv} as above for Table 2. Additionally, knowing the exact distributions allows the prediction errors E^{pred} to be calculated. A more detailed description of the calculation of E^{pred} is presented in the supporting information appendix. The simulations allow us to directly compare our estimated predictability to the true predictability, see Figure 2.

The calculations and graphs in Figure 2 show that E^{cv} is smaller for PCA than for CCA when [CV1] and [CV3] are used (rows 1, 2 and 5), and it agrees better with E^{pred} in each of these cases. In contrast, both E^{cv} and E^{pred} are smaller in the presence of [CV2] (rows 3 and 4) . We will examine these observations further in the discussion, and suggest reasons for this behaviour. We will also look at the performance measures and relate it to the implementation's adherence to the 'out-of-sample' and 'internal coherence' principles.

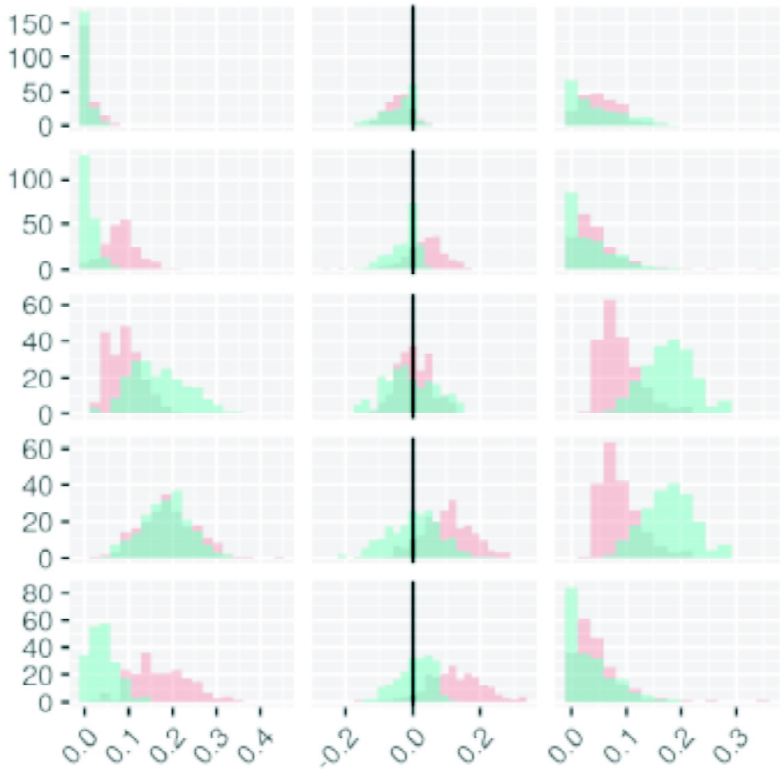

Figure 2: Histograms for 200 simulated samples: CV errors E^{cv} (left column), prediction errors E^{pred} (right column), and paired differences $E^{cv} - E^{pred}$ (centre column) for each of the four combinations of [Pi1] (rows 2 and 4) or [Pi2] (rows 1 and 3) with [CV1] (rows 1 and 2) or [CV2] (rows 3 and 4), as well as for [CV3] (row 5) for both PCA (blue) and CCA (red).

Discussion

[Pi2] allows information about the test data to leak into the training data through the variable selection step, contaminating the 'out-of-sample' principle, while [Pi1] preserves the 'out-of-sample' principle. This contamination arising from [Pi2] is reflected in both real and simulated results by smaller values of E^{cv}. As expected this effect is substantially more pronounced when CCA is used as CCA incorporates information about the predictor as well as the response variables into the dimension reduction. Although much smaller than that for CCA, a similar effect is observable when using PCA. Since PCA does not use information about the response variable this effect may seem surprising, but it has been shown and discussed previously Smialowski *et al.* (2010).

[CV1] violates the 'internal coherence' principle, while [CV2] satises it, at least when applied in combination with [Pi1]. This indicates that [Pi1]-[CV2] should be a good candidate for satisfying both the 'out-of-sample' and 'internal coherence' principles and although it

satisfies the latter, [CV2] violates the 'out-of-sample' principle. More specifically, [CV2] minimises criteria that test and train on the same data (E^{class} and E^{tr}) in order to choose the tuning parameter ν^*. Such criteria tend to act as optimistic estimators of E^{pred}. As a consequence, if the objective in choosing ν^* is to minimise E^{pred}, this choice may not be optimal. In contrast, [CV1] and [CV3] minimise criteria based on 'out-of-sample' testing (using E_j^{cv}) in order to choose ν^*, which tend to be better estimators of E^{pred}. Consequently [CV1] and [CV3] will tend to achieve better E^{pred}, since their approach to selecting ν^* is based on optimising better estimates of E^{pred}. The results of the simulations strongly support this conclusion, with [CV1] and [CV3] achieving substantially better E^{pred} than [CV2].

[CV3] preserves both the 'out-of-sample' and 'internal coherence' principles, while optimising E^{cv} when choosing ν^* thereby staying consistent with the 'out-of-sample' principle and also when selecting ν^*. In the simulated data, this results in E^{pred} comparable to that of [CV2], but with more accurate (or at least more conservative, in the case of CCA) estimation of predictability by E^{cv}.

Conclusions and Recommendations

The principles of CV are well understood across scientific communities. This understanding is a possible reason why these approaches are often taken for granted, and the required care is not taken when considering how to apply these principles to increasingly complex problems, particularly those involving preprocessing and variable selection or dimension reduction steps that require the choice of tuning parameters. We have presented five implementations of CV – four are commonly used implementations of CV to these kinds of problems. These four have been recognised to be problematic in the statistics community, while the fifth implementation, which is discussed in the regression context in Filzmoser *et al.* (2009), addresses these problems through a more complex double CV setting. Analyses of real high-dimension low sample size data from proteomics mass spectrometry imaging of patients with endometrial cancer and of derived simulated data show that these five CV implementations result in widely different errors for CV and prediction.

We suggest that users be guided not only by the size of the error, as expressed by E^{cv} and, in simulations, by E^{pred} as well, but to consider the performance measure of a CV implementation in connection with its adherence to the principles that are satisfied. In particular, a small E^{cv} should not be the only criterion for deciding which approach to use.

We make two recommendations relating to CV implementations and performance. First, since all five CV implementations which we presented would generally all be described by the same phrase in the methods section of a scientific paper (something to the effect of 'CV was used to estimate the prediction error of a LDA classifier trained on data that has been dimension-reduced by PCA/CCA' and analogously for regression), we emphasise the importance of reproducible reporting of methods used – by detailed description of the implementation used, and the inclusion of code reproducing the analysis.

This more detailed reporting is critical for results to be compared to one another in a valid way. Second, we recommend readers consider how to implement CV at each stage: at the preprocessing stage of data, and each time a tuning parameter is chosen. We have included and recommend use of a double CV approach, but as the results show no approach is perfect and it is important to be aware of the limitations of the chosen approach when interpreting results.

Acknowledgements

This work began while the first author was employed at the mass spectrometry lab in the Future Industries Institute at the University of South Australia. We thank the members of the mass spectrometry lab for helpful discussions. The authors thank the editor and a reviewer for their helpful comments.

References

Baker, M. (2016). Reproducibility crisis. *nature* 533 (26), 353-66.

Bendifallah, S., A. S. Genin, I. Naoura, N. C. Buet, F. C. Chapelon, B. Haddad, D. Luton, E. Darai, R. Rouzier, and M. Koskas (2012). A nomogram for predicting lymph node metastasis of presumed stage I and II endometrial cancer. *American journal of obstetrics and gynecology* 207 (3), 197.e1-197.e8.

Cai, T. and W. Liu (2011). A direct estimation approach to sparse linear discriminant analysis. *Journal of the American statistical association* 106 (496), 1566-1577.

Efron, B. (1983). Estimating the error rate of a prediction rule: improvement on cross-validation. *Journal of the American statistical association* 78 (382), 316-331.

Fan, J. and J. Lv (2010). A selective overview of variable selection in high dimensional feature space. *Statistica Sinica* 20 (1), 101.

Filzmoser, P., M. Gschwandtner, and V. Todorov (2012). Review of sparse methods in regression and classication with application to chemometrics. *Journal of Chemometrics* 26 (3-4), 42-51.

Filzmoser, P., B. Liebmann, and K. Varmuza (2009). Repeated double cross validation. *Journal of Chemometrics: A Journal of the Chemometrics Society* 23 (4), 160-171.

Guo, S., T. Bocklitz, U. Neugebauer, and J. Popp (2017). Common mistakes in cross-validating classification models. *Analytical Methods* 9 (30), 4410-4417.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction.* Springer Science & Business Media.

Jimenez-Rodriguez, L. O., E. Arzuaga-Cruz, and M. V–elez-Reyes (2007). Unsupervised linear feature-extraction methods and their effects in the classification of high-dimensional data. *IEEE Transactions on geoscience and remote sensing* 45 (2), 469-483.

Kearns, M. and D. Ron (1999). Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural computation* 11 (6), 1427-1453.

Lachenbruch, P. A. and M. R. Mickey (1968). Estimation of error rates in discriminant analysis. *Technometrics* 10 (1), 1-11.

Mittal, P., M. Klingler-Homann, G. Arentz, L. Winderbaum, N. A. Lokman, C. Zhang, L. Anderson, J. Scurry, Y. Leung, C. J. Stewart, *et al.* (2016). Lymph node metastasis of primary endometrial cancers: Associated proteins revealed by MALDI imaging. *Proteomics* 16 (11-12), 1793-1801.

Molinaro, A. M., R. Simon, and R. M. Pfeier (2005). Prediction error estimation: a comparison of resampling methods. *Bioinformatics* 21 (15), 3301-3307.

Munafo, M. R., B. A. Nosek, D. V. Bishop, K. S. Button, C. D. Chambers, N. P. Du Sert, U. Simonsohn, E.-J. Wagenmakers, J. J. Ware, and J. P. Ioannidis (2017). A manifesto for reproducible science. *Nature human behaviour* 1 (1), 1-9.

Rao, R. B., G. Fung, and R. Rosales (2008). On the dangers of cross-validation. an experimental evaluation. In *Proceedings of the 2008 SIAM international conference on data mining*, pp. 588-596. SIAM.

Rauschenberger, M. and R. Baeza-Yates (2020). Recommendations to handle health-related small imbalanced data in machine learning. *Mensch und Computer 2020-Workshopband* -(-).

Smialowski, P., D. Frishman, and S. Kramer (2010). Pitfalls of supervised feature selection. *Bioinformatics* 26 (3), 440-443.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Methodological)* 36 (2), 111-133.

Stone, M. (1978). Cross-validation: A review. *Statistics: A Journal of Theoretical and Applied Statistics* 9 (1), 127-139.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B (JRSS-B)* 58, 267-288.

Varma, S. and R. Simon (2006). Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics* 7 (1), 91.

Winderbaum, L., I. Koch, P. Mittal, and P. Homann (2016). Classification of MALDI-MS imaging data of tissue microarrays using canonical correlation analysis-based variable selection. *Proteomics* 16 (11-12), 1731-1735.

Winderbaum, L. J., I. Koch, O. J. Gustafsson, S. Meding, P. Homann, *et al.* (2015). Feature extraction for proteomics imaging mass spectrometry data. *The Annals of Applied Statistics* 9 (4), 1973-1996.

Yu, B. and K. Kumbier (2020). Veridical data science. *Proceedings of the National Academy of Sciences* 117 (8), 3920-3929.

Zhang, T. (2003). Leave-one-out bounds for kernel methods. *Neural computation* 15 (6), 1397-1437.

Supplementary Material Cross-validation for Supervised Learning with Tuning Parameters

Lyron Winderbaum and Inge Koch

Computational Details

The results in this paper were obtained on x86 64-pc-linux-gnu (64-bit) with Ubuntu 18.04.3 LTS and R 3.6.2 as well as the following R packages: base and stats, R, MASS, Ripley 2002, geigen geigen, and plyr. Additionally, ggplot2 was used to produce the figures. All packages used are available from the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/. All data and code required to reproduce the results in this paper are available at https://zenodo.org/badge/latestdoi/235052456 in the form of R scripts.

Methodological Details

In this section we provide more detailed and explicit descriptions of the methods used in the various calculations presented in the main text.

LDA

We originally considered using the standard R function lda() from the standard MASS package but its non-deterministic behaviour when making predictions (specifically in MASS:::predict.lda()) for new data near the decision boundary is not ideal for reproducibility purposes, even though it can be made to be reproducible using set.seed(). So we use our own deterministic implementation train_lda() instead. The code for this function can be found in the housekeeping_functions. R script at https://zenodo.org/badge/latestdoi/235052456. What follows is a detailed description of that implementation. We split the data set $\mathbb{X}$ into the n_1 observations in class 1, which we denote $\mathbb{X}_1$, and those n_2 observations in class 2 (so $n_1 + n_2 = n$), which we denote $\mathbb{X}_2$, with respective sample means $\bar{X}_1$ and $\bar{X}_2$. We will train a classication rule by calculating $\eta \in \mathbb{R}^d$ and $m \in \mathbb{R}$. We treat $\mathbb{X}_1$ and $\mathbb{X}_2$ as $d \times n_1$ and $d \times n_2$ matrices respectively, and hence $\bar{X}_1$ and $\bar{X}_2$ as column vectors. We denote the $d \times 2$ matrix with $\bar{X}_1$ and $\bar{X}_2$ as columns $\bar{\mathbb{X}}$. Note that when we write addition or subtraction between a matrix and vector such as $\mathbb{X}_1 - \bar{X}_1$ we mean the obvious thing: to subtract $\bar{X}_1$ from each observation in $\mathbb{X}_1$. First,

we calculate

$$B = (\bar{\mathbb{X}} - \bar{X}^*)^T (\bar{\mathbb{X}} - \bar{X}^*) \text{ and} \tag{1}$$

$$W = \frac{1}{n_1 - 1}(\mathbb{X}_1 - \bar{X}_1)^T (\mathbb{X}_1 - \bar{X}_1) + \frac{1}{n_2 - 1}(\mathbb{X}_2 - \bar{X}_2)^T (\mathbb{X}_2 - \bar{X}_2), \tag{2}$$

where $\bar{X}^* = \frac{1}{2}(\bar{X}_1 + \bar{X}_2)$. Then, we estimate the eigendecomposition of the symmetric matrix W

$$W = \Gamma^T \Lambda \Gamma \tag{3}$$

where Λ is the $d \times d$ diagonal matrix of eigenvalues $\lambda_1 \geq \lambda_2 \ \ldots \geq \lambda_d$ and Γ is the $d \times d$ matrix with the corresponding eigenvectors $\gamma_1, \gamma_2, \ldots, \gamma_d$ as columns and both Λ and Γ are numerically estimated using the function eigen() from the standard base package Core Team (2019). We estimate the rank r of W as the number of $\Lambda_i \geq 10^{-14}$ as below this is close to double precision. Then, if W has full rank $r = d$, we solve the generalised eigenvalue problem

$$B\gamma = \lambda W\gamma \tag{4}$$

directly using the geigen() function from the geigen package Hasselman and authors (2019) and take η to be the solution for γ corresponding to the largest eigenvalue λ. Otherwise, if we estimate W to have rank $r < d$, we calculate

$$W = \Gamma_r^T \Lambda_r^{-1} \Gamma_r \tag{5}$$

where Λ_r is the $r \times r$ diagonal matrix of eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r$ and Γ_r is the $d \times r$ matrix with the corresponding eigenvectors $\gamma_1, \gamma_2, \ldots, \gamma_r$. We then take η to be the (right) eigenvector corresponding to the largest eigenvalue in the eigendecomposition of the matrix W^*B calculated using eigen, *i.e.* solution to the standard eigenvalue problem

$$W^*B\gamma = \lambda\gamma. \tag{6}$$

Either way, we now have a vector η although it is only specfiied up to sign, and as such we multiply it by –1 as necessary so that $\eta^T \bar{X}_1 > \eta^T \bar{X}_2$. Then we calculate

$$m = \eta^T (\bar{X}^*) = \frac{1}{2}\eta^T (\bar{X}_1 + \bar{X}_2), \tag{7}$$

and we are done. Training an LDA classication rule essentially constitutes calculating a pair (η, m). The resulting classication rule $\mathfrak{r}$ (with parameters η and m) assigns a class to a new observation x as follows:

$$\mathfrak{r}(x) = \begin{cases} 1 & \text{if } \eta^T x > m \\ 2 & \text{if } \eta^T x < m. \end{cases} \tag{8}$$

PCA and CCA

We use two contrasting dimension reduction transforms for illustration and comparison. Probably the most common dimension reduction transform is PCA – see Chapter 2 of Koch (2013), which uses the sample covariance matrix and finds the directions which

maximise variability. PCA does not take into account information regarding the class membership of the observations, and as a consequence PCA directions may not include directions which separate the classes best. In the data analysis we use the R function prcomp in the stats package Core Team (2019) to carry out PCA without scaling.

The other dimension reduction transform we use is based on CCA and ranks the variables of the data by exploiting the relationship between the data and the vector of class labels, *Y*. Define a ranking of the original variables of X by the order of absolute values of entries in a ranking vector starting with the largest value. The CCA-based dimension reduction transform π_k selects the first k of the original variables in this ranking. The version we use is a modification of the approach of Tamatani *et al.* (2012) – for more details see (Koch, 2013, Sections 3.2, 4.8 and 13.3), and for a specific derivation of the 'ranking vector' we use see Winderbaum *et al.* (2016).

Calculation of E^{pred}

For an LDA classication rule $\mathfrak{r}$ with η, *m* as described above in this document, we calculate the true prediction error E^{pred} as follows. Our simulations come from a population that consists of a mixture of two multivariate normal distributions, $X^+ \sim \mathcal{N}(\mu^+, \Sigma^+)$ and $X^- \sim \mathcal{N}(\mu^-, \Sigma^-)$ with a probability of being from X^+ of $\frac{16}{43}$ and a probability of being from X^- of $\frac{27}{43}$. That is, if we let *Y* denote a random new observation drawn from this distribution,

$$P(Y \equiv X^+) = \frac{16}{43} \text{ and}$$

$$P(Y \equiv X^-) = \frac{27}{43}.$$

Note that in reality each sample of data is fixed to this exact ratio and not actually sampled from a binomial class distribution, but we claim this would be the distribution from which a new observation would be drawn – *i.e.* for prediction. We calculate

$$\begin{aligned} E^{pred} &= P((\mathfrak{r}(Y) = 1) \cap (Y \equiv X^-)) + P((\mathfrak{r}(Y) = 2) \cap (Y \equiv X^+)) \\ &= P(Y \equiv X^-)P(\mathfrak{r}(Y) = 1 | Y \equiv X^-) + P(Y \equiv X^+)P(\mathfrak{r}(Y) = 2 | Y \equiv X^+) \\ &= \frac{27}{43} P(\mathfrak{r}(Y) = 1 | Y \equiv X^-) + \frac{16}{43} P(\mathfrak{r}(Y) = 2 | Y \equiv X^+) \end{aligned}$$

in practice, this is calculated in terms of univariate normal distributions as

$$E^{pred} = \frac{27}{43} P(\eta^T Y > m | \eta^T Y \sim \mathcal{N}(\eta^T \mu^-, \eta^T \Sigma^- \eta)) + \frac{16}{43} P(\eta^T Y < m | \eta^T Y \sim \mathcal{N}(\eta^T \mu^+, \eta^T \Sigma^- \eta)).$$

More CV Alternatives

We discussed [CV1], [CV2], and [CV3] as different options for choosing the 'optimal' number of dimensions. These are by no means the only options, they are simply intuitive, commonly used, and made for good discussion. Another method is one that essentially combines ideas from [CV1] and [CV2]: choosing a different number of dimensions for each j as in [CV2] but the preserving the 'out-of-sample' testing idea from [CV1] and instead choosing:

$$k_j^* = \min[\text{argmin } E_{k,j}^{cv} \text{ for each } j = 1, 2, ..., m, \tag{9}$$

where the argmin is taken over k. This is reasonable if $m \ll n$, but in the LOO case this bases each decision on a single test observation, which could be noisy. Another approach would be to choose a k^* apriori. This is perhaps the most commonly used argument, as it is simple and easy to understand, and treats k^* as a constant rather than as a constant parameter.

References

Core Team, R. (2019). R: *A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Hasselman, B. and L. authors (2019). *geigen: Calculate Generalized Eigenvalues, the Generalized Schur Decomposition and the Generalized Singular Value Decomposition of a Matrix Pair with Lapack*. R package version 2.3.

Koch, I. (2013). *Analysis of Multivariate and High-Dimensional Data, Volume 32 of Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press.

Tamatani, M., I. Koch, and K. Naito (2012). Pattern recognition based on canonical correlations in a high dimension low sample size context. *Journal of Multivariate Analysis* 111, 350-367.

Winderbaum, L., I. Koch, P. Mittal, and P. Homann (2016). Classification of MALDI-MS imaging data of tissue microarrays using canonical correlation analysis-based variable selection. *Proteomics* 16 (11-12).