2020

Deriving statistical inference from the application of artificial neural networks to clinical metabolomics data

Kevin M. Mendez
Edith Cowan University

Edith Cowan University

Copyright Warning

Deriving Statistical Inference from the Application of Artificial Neural Networks to Clinical Metabolomics Data

This thesis is presented for the degree of

Master of Science (Chemistry)

Kevin Milton Mendez

Edith Cowan University

School of Science

2020

Declaration

I certify that this thesis does not, to the best of my knowledge and belief:

i) incorporate without acknowledgement any material previously submitted for a degree or diploma in any institution of higher education;

ii) contain any material previously published or written by another person except where due reference is made in the text; or

iii) contain any defamatory material.

iv) I also grant permission for the Library at Edith Cowan University to make duplicate copies of my thesis as required.

Kevin Milton Mendez

Abstract

Metabolomics data are complex with a high degree of multicollinearity. As such, multivariate linear projection methods, such as partial least squares discriminant analysis (PLS-DA) have become standard. Non-linear projections methods, typified by Artificial Neural Networks (ANNs) may be more appropriate to model potential nonlinear latent covariance; however, they are not widely used due to difficulty in deriving statistical inference, and thus biological interpretation. Herein, we illustrate the utility of ANNs for clinical metabolomics using publicly available data sets and develop an open framework for deriving and visualising statistical inference from ANNs equivalent to standard PLS-DA methods.

Table of Contents

Acknowledgements

I would first like to express my immense gratitude to my supervisors Professor David Broadhurst and Dr Stacey Reinke. David, for all the knowledge in classical statistics and machine learning that you have imparted. Stacey, for all the support and guidance in developing as a researcher. I will never forget all the opportunities you both have given me.

I would like to thank Associate Professor Mary Boyce who continuously kept me on track to complete this MSc thesis on time. These last two years have truly been enjoyable, especially in the post-doc room with Dr Brett Chapman, Dr Jessica Pandohee, Dr Nathan Lawler, and Dr Thao Thi Thu Le.

I would also like to thank Dr Leighton Pritchard who imparted his knowledge around open data science. It was a pleasure collaborating with you.

This project was only possible with the support of the Centre for Integrative Metabolomics and Systems Biology (CIMCB) and the School of Science at Edith Cowan University (ECU). I'm grateful for all the support I received from ECU staff members, especially Mrs Yvonne Garwood.

To all my friends and family, thank you for your patience and understanding.

List of Publications

Mendez, K.M, Broadhurst, D.I., Reinke, S.N. (2019) The Application of Artificial Neural Networks in Metabolomics: A Historical Perspective. *Metabolomics,* **15,** 142.

Mendez, K.M., Pritchard, L., Reinke, S.N., Broadhurst, D.I. (2019) Toward Collaborative Open Data Science in Metabolomics using Jupyter Notebooks and Cloud Computing. *Metabolomics,* **15,** 125.

Mendez, K.M, Reinke, S.N., Broadhurst, D.I. (2019) A Comparative Evaluation of the Generalised Predictive Ability of Eight Machine Learning Algorithms across Ten Clinical Metabolomics Data Sets for Binary Classification. *Metabolomics,* **15,** 150

Mendez, K.M, Broadhurst, D.I., Reinke, S.N. (2019) Migrating from Partial Least Squares Discriminant Analysis to Artificial Neural Networks: A Comparison of Functionally Equivalent Visualisation and Feature Contribution Tools using Jupyter Notebooks. *Metabolomics,* **16,** 17

List of Conference Proceedings

Mendez, K.M., Reinke, S.N., Broadhurst, D.I. (2019) The Application of Artificial Neural Networks in Metabolomics. *The 15th Annual Conference of the Metabolomics Society*, 23rd - 27th June. The Hague, Netherlands.

List of Tables

List of Figures

List of Abbreviations

ANN	Artificial neural network
API	Application programming interface
AUC	Area under the receiver operating characteristic curve
CLI	Command line interface
CV	Cross-validation
CWA	Connection weight algorithm
DNN	Deep neural network
FAIR	Findable, accessible, interoperable, and reusable
GA	Garson's algorithm
GC-MS	Gas chromatography-mass spectrometry
GUI	Graphical user interface
IB	In-bag
IDE	Integrated development environment
JSON	JavaScript object notation
LC-MS	Liquid chromatography-mass spectrometry
ML	Machine learning
NMR	Nuclear magnetic resonance
OOB	Out-of-bag
PCA	Principal component analysis
PCLR	Principal component logistic regression
PCR	Principal component regression
PLS	Partial least squares regression
PLS-DA	Partial least squares discriminant analysis
RBF	Radial basis function
RF	Random forest
ROC	Receiver operating characteristic
SVM	Support vector machine
VIP	Variable influence on projection

Statement of Contribution

I, Kevin Milton Mendez, certify that the following author contributions are accurate. This is endorsed by all co-authors (Appendix 1-4).

Kevin Milton Mendez

Publication (1): Mendez, K.M., Broadhurst, D.I., Reinke, S.N. (2019) The Application of Artificial Neural Networks in Metabolomics: A Historical Perspective. *Metabolomics* **15,** 142.

Author Contributions (1): All authors conceived of the idea. KMM wrote the manuscript. DIB and SNR edited the manuscript.

Publication (2): Mendez, K.M., Pritchard, L., Reinke, S.N., Broadhurst, D.I. (2019) Toward Collaborative Open Data Science in Metabolomics using Jupyter Notebooks and Cloud Computing. *Metabolomics,* **15,** 125.

Author Contributions (2): All authors contributed equally to this work. KMM developed the code for the Python cimcb-lite package. KMM & DIB developed the tutorial Jupyter notebook code. LP and SNR edited the learning tutorials to align to current pedagogical best practices. KMM wrote the initial draft of the review section. All authors edited the manuscript.

Publication (3): Mendez, K.M, Reinke, S.N., Broadhurst, D.I. (2019) A Comparative Evaluation of the Generalised Predictive Ability of Eight Machine Learning Algorithms across Ten Clinical Metabolomics Data Sets for Binary Classification. *Metabolomics,* **15,** 150

Author Contributions (3): All authors conceived of the idea. KMM developed the Jupyter notebooks. KMM developed the Python packages. KMM performed the optimisation and evaluation of all models. DIB & SNR independently checked each model. KMM wrote the manuscript. DIB and SNR edited the manuscript.

Publication (4): Mendez, K.M, Broadhurst, D.I., Reinke, S.N. (2019) Migrating from Partial Least Squares Discriminant Analysis to Artificial Neural Networks: A Comparison of Functionally Equivalent Visualisation and Feature Contribution Tools using Jupyter Notebooks. *Metabolomics,* **16,** 17

Author Contributions (4): All authors conceived of the idea. KMM and DIB developed the software. KMM wrote the manuscript. DIB and SNR edited the manuscript.

Copyright Statement

I, Kevin Milton Mendez, warrant that I have obtained, where necessary, permission from the copyright owners to use any third-party copyright material reproduced in the thesis, or to use any of my own published work in which the copyright is held by another party.

Kevin Milton Mendez

Chapter One: General Introduction

1.1. Introduction

Data is a critical scientific asset; however, data's intrinsic value is dependent on the ability to extract critical insights. The multidisciplinary field of *data science* is concerned with extracting insights from data using a diverse set of computational methodologies, theories, and technologies (Cao, 2017; Maneth and Poulovassilis, 2017). In data science, there are two fundamental scientific philosophies: *classical statistics* and *machine learning* (Breiman, 2001). Classical statistics aims to formalise the relationships within the data in the form of mathematical equations, based on a clearly defined set of assumptions about the data structure; whereas, machine learning uses ad-hoc computational algorithms that iteratively optimise (or 'learn') from data without necessarily relying on rules-based programming, of any formal statistical assumptions. In the era of 'big' data, datasets are increasingly becoming too large and complex for classical statistics to cope with, and machine learning approaches are becoming essential.

One application for machine learning is in *metabolomics*. Metabolomics, a core component of systems biology, is the systematic study of low molecular weight biochemicals (metabolites) in biological systems (Dunn *et al.*, 2011b). As metabolites are the final down-stream product of gene expression and are highly sensitive to changes in the environment, they are in close biological proximity to the phenotype of the biological systems. In clinical applications, metabolomics has great potential to improve diagnosis, prognosis, and treatment. With proper statistical inference, metabolomics can be used to identify biomarkers, stratify patients by disease sub-types, and understand underlying mechanisms of disease (Beger *et al.*, 2016;

Johnson *et al.*, 2016; Zhang *et al.*, 2015). Classical statistical methods are currently the preferred method of data analysis as these models formalise simple relationships between metabolite abundance and a biological question in a very understandable way. That said, metabolomics data is highly complex, with inherent covariance structure which is beyond the scope of most classical statistical methods. As such, multivariate hybrid machine learning / classical statistics methods have been very popular, such as partial least squares (PLS) regression (note; PLS is generally referred to as PLS *discriminatory analysis* (PLS-DA) when used for classification). Increasingly, more data driven machine learning methods have become popular to model this complex covariance data structure including random forest (RF), support vector machines (SVMs), and artificial neural networks (ANNs). Due to the highly flexible architecture of ANNs, there are potential applications of ANNs beyond a typical clinical metabolomic study (case/control with one platform) including the integration of multiple platforms and 'omic technologies (i.e. integrative systems biology).

ANNs are a network of interconnected neurons at an input layer, hidden layer(s), and output layer, where a neuron represents a mathematical transfer function, and a connection represents a multiplication factor (or "weight") from the output of one neuron to the input of the next (Fig 1.1). The interaction from one layer to the next can be considered as a weighted sum followed by a linear or non-linear transformation. With the combination of multiple (hidden) layers (commonly referred to as "deep learning"), complex functions can be learned. This high-level abstraction results in the high predictability of models at the expense of interpretability (Lecun *et al.*, 2015). Thus, deriving statistical inference of ANNs is a challenge and an area of ongoing research (Samek *et al.*, 2017; Zhang and Zhu, 2018; Zhang *et al.*, 2018). This difficulty in deriving statistical inference needs to be resolved for ANNs to be viable in metabolomics.

Therefore, the overarching aim of this project is to evaluate the utility of non-linear ANNs in deriving statistical inference from clinical metabolomic data sets.

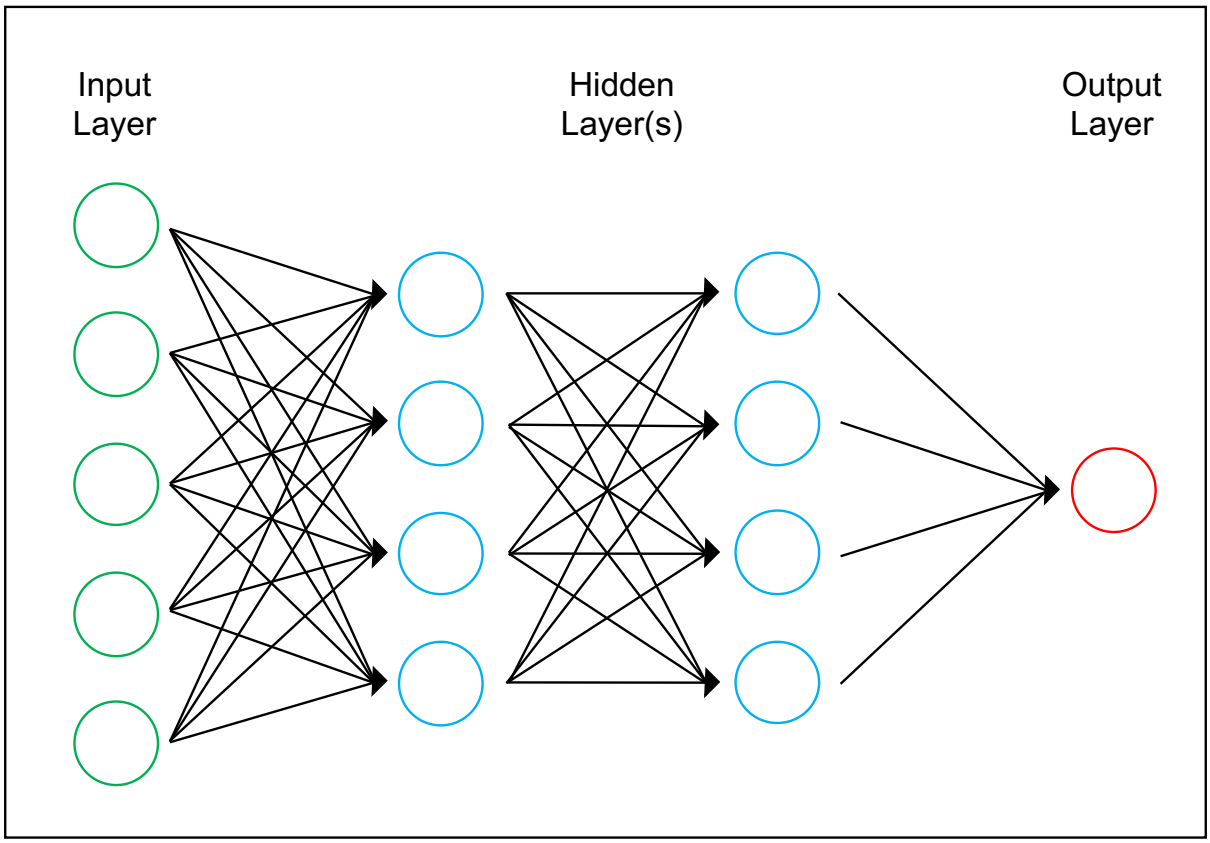


Figure 1.1: An example of an artificial neural network. Circles represent neurons in each layer. Arrows represent the connect of the output of a neuron to the input of another neuron.

1.2. Metabolomics and Systems Biology

Metabolomics is the systematic study of metabolites in a biological system (organelles, cells, tissues, organs and organisms) (Dunn and Ellis, 2005). The complete set of metabolites (exogenous and endogenous) in a biological system is referred to as the metabolome. The metabolome can be considered the final down-stream product of gene expression that is highly sensitive to changes in the environment, meaning metabolomics is in closer biological

proximity to the phenotype of the biological systems when compared to other omics' (Fiehn, 2002).

Biological systems can be viewed as a construction of a complex and interconnected network of genes, transcripts, proteins, and metabolites (Dunn *et al.*, 2010). It is the complex interaction of these components that give rise to the function and behaviour of the biological system. This perspective known as systems biology uses an integrative approach (as opposed to a reductionist approach) to understand biological function or phenotype (Karahalil, 2016). Given the importance of the metabolome in biological systems, there is high demand in data science for the integration of metabolomics within systems biology frameworks (Weckwerth, 2010). Current limitations arise due to the complex and high-dense nature of metabolomics datasets.

Metabolites comprise of many classes and can vary in their specific physicochemical properties. The metabolome is complex with over 100,000 metabolites entries on the human metabolome database (Wishart *et al.*, 2018). No single analytical platform that can be used to analyse all metabolites in a given biological sample. Nuclear magnetic resonance (NMR) spectroscopy and mass spectrometry (MS) are the two most widely used analytical methods of detection in metabolomics (Dunn *et al.*, 2011a). Mass spectrometers are typically used in combination with chromatography: gas chromatography (GC) or liquid chromatography (LC). The raw data acquired by analytical instruments is highly variable, due to differences in instruments and associated instrumental methods used. The visualisation of variable and complex datasets requires the aid of robust statistical analysis techniques and strategies.

1.3. Machine Learning

Machine learning is a field of computer science that uses algorithmic approaches to provide computer systems with the ability to learn without explicit programming (Samuel, 1959). Machine learning is data-driven, with iterative learning (on repeated data) based on a mathematical optimisation (learning of parameters), where the goal is to optimise (minimise or maximise) a value based on an initially defined performance metric (Mitchell, 1997). Classical statistics is driven by formal hypothesis testing, with clear and testable assumptions, with the focus on the understanding of model parameters (Breiman, 2001). The challenge for classical statistics with large and complex ('big') data is the inability to adherence to the strict assumption of the model (Fan *et al.*, 2013). Machine learning is data-driven and requires no assumptions making it the preferable approach with large and complex data, particularly within the last decade with advancements in computational power and modelling techniques (Kocheturov *et al.*, 2018).

There are multiple machine learning techniques commonly used including support vector machines (SVM), random forest (RF), partial least squared regression (PLS) and artificial neural networks (ANN). Table 1.1 includes a brief description of SVM, RF, PLS, and ANN along with the advantages and caveats of each machine learning technique. PLS differs from SVM, RF, and ANN as it enables simple visualisation of internal data structures (latent variable scores plot and latent variable weights plot) and easy interpretation of model parameters using a coefficients plot and variable important in projection (VIP) plot. PLS is an interesting member of the machine learning family as it can be considered both a classical statistical model (due to clear parametric assumptions about underlying data structure) and a machine learning algorithm (due to the iterative parameter optimisation). There is a clear similarity between PLS

and ANNs as both are projection-based methods as they project data into a hidden layer and the number of nodes (latent variables) is determined by data-driven methods. However, due to the manner in which ANNs optimise parameter values, they are a much more flexible class of model, allowing multiple hidden layers and non-linear transfer functions.

Table 1.1: Brief summary and comparison of common machine learning algorithms (PLS, SVM, RF and ANN). The comparison includes generally recognised advantages and caveats.

Algorithms	Advantages	Caveats
Partial Least Squares Regression A method that uses a *projection to latent space* approach to model the linear covariance between two data matrices	- Produces estimates of variable importance -Visualisation (scores, weights, coefficients, and VIP plot) - Computationally inexpensive	- Linear mapping only
Support Vector Machine A classifier that separates the classes via an optimised hyperplane	- Linear and non-linear mapping (multiple kernels)	- Lack of standard variable importance tools (for non-linear kernels) - Lack of visualisation tools - Computationally expensive
Random Forest An ensemble learning method that combines multiple decision trees and outputs a mean/mode prediction	- Produces estimates of variable importance - Linear and non-linear mapping	- Complex visualisation - Computationally expensive
Artificial Neural Network A network of interconnected neurons at an input layer, hidden layer(s) and output layer	- Linear and non-linear mapping - Highly modular/flexible (multiple variants)	- Lack of standard feature (metabolite) importance tools - Lack of visualisation tools - Computationally expensive

1.4. Machine Learning and Metabolomics

Machine learning algorithms are important in the analysis of metabolites in a biological system. In its most simplistic form, a metabolomics study can be considered as the comparison of a system's metabolome after a single factor perturbation (e.g. a two population "case vs control" study) (Broadhurst and Kell, 2006), with the aim of discovering how the metabolic profile differs significantly. Metabolic profiling (investigation of the metabolome) is with minimal *a priori* biological knowledge, making it a powerful hypothesis-generating tool for biological questions (e.g. mechanisms behind disease). As this approach is ostensibly data-driven with minimal *a priori* biological knowledge, it a powerful hypothesis-generating tool for biomarker discovery and understanding biological mechanisms.

Machine learning techniques used in metabolomics must reflect the goal of discovering important metabolites in relation to the biological question. The need for the simple interpretation of metabolites is key to this aim. Hence why PLS with its simple model structure and visualisation is widely used (Gromski *et al.*, 2015). PLS projects linear combinations of metabolites into low dimensional space (latent variables – LV) which reflect metabolite covariance. This covariance is functionally interpreted using the LV score plots, with the corresponding LV weights plot representing the importance of each feature (metabolite). Additionally, coefficients and VIP scores can be used to estimate the contribution of each feature (metabolite) in the overall model. In combination, these plots allow for the investigation and visualisation of metabolite contribution in relation to biological questions in clinical metabolomic studies.

Historically, ANNs have also been applied to metabolomics data (Goodacre *et al.*, 1992); however, they have fallen out of favour compared to PLS machine learning methods. Goodacre (2003), summarised the need for simple results in metabolomics as ANNs were perceived by the metabolomics community as a 'black box', and considered too difficult to interpret. The success of deep learning in the analysis of 'big' data (Chen and Lin, 2014; Najafabadi *et al.*, 2015), has heralded the potential return of ANNs to metabolomics, but the historical challenges of poor interpretability remain. One possible solution may be to migrate tools for interpretation from PLS to ANNs, as both are projection-based methods. In fact, a linear ANN with a single hidden layer has *structural equivalence* with PLS, only differing in the manner in which model parameters are optimised. We hypothesise that *functionally equivalent* visualisation and feature contribution tools for PLS can be migrated to single hidden layer ANNs. Provided this approach to statistical inference is successful, it opens the door to more complex ANN architectures in applications such as integrative systems biology.

1.5. Machine Learning and Integrative Systems Biology

Integrative systems biology is an interdisciplinary approach to understand biology in a holistic way. The combination of multiple 'omics provides a comprehensive global snapshot of biological function. Due to rapid development in high-throughput technologies, integrative systems-biology is becoming a central part of 'omics research. There are four cardinal approaches to integrate multiple 'omics: correlation-, concatenation-, multivariate-, and meta-analysis- based data integration (Cavill *et al.*, 2016). Multivariate-based data integration is useful in integrative systems-biology as it models the covariance data structures within and between 'omics data blocks.

Generally, multivariate-based data integration has been focused on hierarchical applications of linear projection-based models such as OnPLS (Löfstedt and Trygg, 2011; Reinke *et al.*, 2018). This complex hierarchy makes it difficult to derive and visualise statistical inference. Rather than hierarchical optimisation of multiple sub-models, the inherent flexibility of ANNs allows for a single model architecture. This flexibility of architecture combined with the non-linear nature of modelling makes ANNs potentially suitable to model highly complex multi-block data. Two examples of ANN architectures for multi-block data integration are shown in Fig. 1.2. If the lack of statistical inference of ANNs is resolved for simple ANNs and the solution extended to more complex architectures, ANNs could become a core method for integrative systems biology.

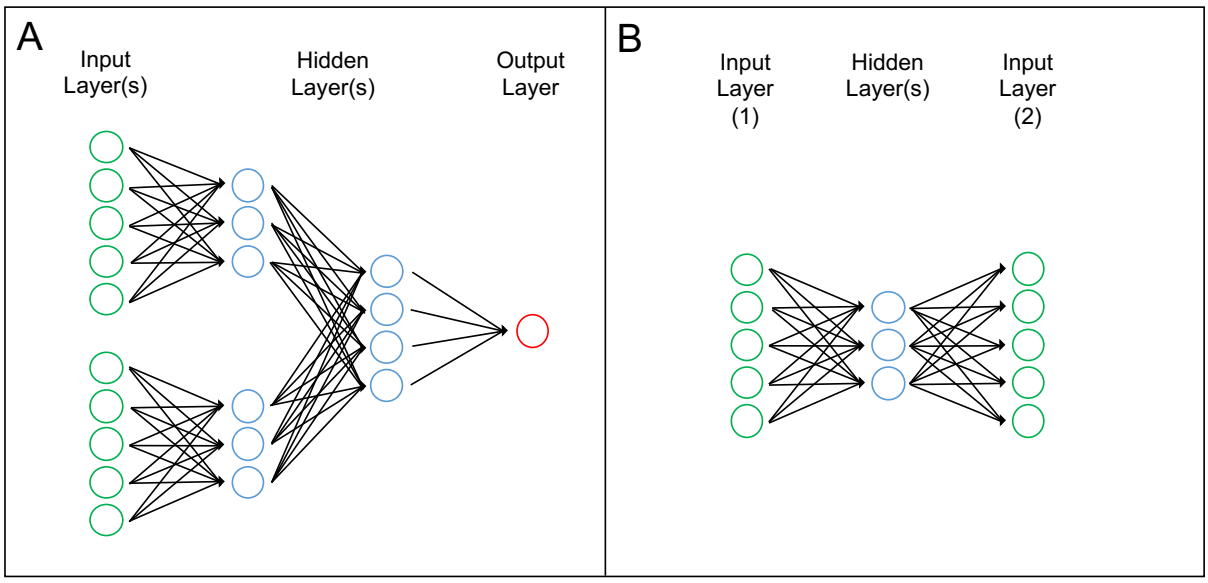


Figure 1.2: Examples of multi-block ANN architectures. **a** Multiple input layers that feed to an output layer. **b** Input layer (1) that feeds into input layer (2) or vice-versa.

1.6. Summary and Aims

PLS has become the standard approach in clinical metabolomics due to the ability to formalise the relationship between metabolite abundance and potential pathophysiology or mechanism

behind disease. However, metabolic profiling data is often highly complex, resulting in the inability of PLS to provide adequate modelling. There is a high demand for ANNs with recent advances in computational power, modelling improvements, community acceptance, and the growing interest in integrative systems-biology. Therefore, the overarching aim of this proposal is to evaluate the utility of non-linear ANNs in deriving statistical inference from clinical metabolomic data sets.

The specific aims of this thesis were:

1. To develop an open source computational framework to allow for interactive modelling and assessment of common machine learning methods specifically tailored for metabolomics studies.
2. To compare the predictive accuracy of non-linear artificial neural networks with current linear projection methods, and alternative non-linear machine learning algorithms, specifically applied to clinical metabolomic studies.
3. To investigate visualisation and variable importance techniques for artificial neural networks that are comparable to current linear projection methods
4. Critically compare any novel artificial neural network visualisation and variable importance techniques against existing linear projection workflows using publicly available metabolomic data sets.

1.7. Description of Publications

This thesis includes 4 peer-reviewed publications. Full author contributions can be found in the author contribution section of the corresponding chapter, or in Appendix 1 - 4. A description of each chapter, along with its importance to this thesis are provided below:

Chapter Two is a literature review published in the journal *Metabolomics* (springer.com) titled, "The application of artificial neural networks in metabolomics: a historical perspective" (Mendez *et al.*, 2019a). This paper provides a detailed description of ANNs, and the historical, current, and potential future use of ANNs in metabolomics. Additionally, there is an explanation of the *structural equivalence* between ANNs and PLS, which is key to chapter five in developing *functionally equivalent* visualisation and feature contribution tools for ANNs.

Chapter Three addresses Aim 1 of the thesis and is presented in the form of a tutorial review paper, published in the journal *Metabolomics* (springer.com) titled "Toward collaborative open data science in metabolomics using Jupyter Notebooks and cloud computing" (Mendez *et al.*, 2019b). To align with FAIR (Findable, Accessible, Interoperable, and Reusable) data principles (Wilkinson *et al.*, 2016), there was a need to develop an open framework for the data analysis. The approach used in this thesis is based around Jupyter Notebook, in combination with the GitHub and Binder. Firstly, this paper provides a background on the 'reproducibility crisis', describes the need to align to FAIR principles, details the measures the community has taken, and presents a solution for open data science in metabolomics. Secondly, to encourage this solution, the paper includes 4 tutorials on creating (and using) open data analysis workflows with Jupyter Notebooks, GitHub, and Binder. This method of creating open data analysis workflows is used in the following publications which directly address Aims 2, 3, and

4. These learning tutorials provide unfamiliar readers with an important understanding of how to use and interact with the provided open data analysis workflows.

Chapter Four addresses Aim 2 of the thesis and is presented in the form of a research paper published in the journal *Metabolomics* (springer.com) titled, *"A comparative evaluation of the generalised predictive ability of eight machine learning algorithms across ten clinical metabolomics data sets for binary classification"* (Mendez *et al.*, 2019c). This paper compared the general predictive performance of eight machine learning algorithms across ten publicly available clinical metabolomics data sets. The datasets represented a cross-section of published clinical metabolomic data representing typical sample size, complexity of the biological question, analytical instrument, and biofluid. The machine learning algorithms compared were: partial least squares discriminant analysis (PLS-DA), artificial neural networks (ANNs), principal component regression (PCR), principal component logistic regression (PCLR), support vector machines (SVM), and random forest (RF). In total, 80 open data analysis workflows (in the form of Jupyter notebooks) were created using the method described in chapter three. All machine learning algorithms were implemented in the Python programming language. All code and results have been made publicly available as Jupyter notebooks.

Chapter Five addresses Aims 3 and 4 of the thesis and is presented in the form of a research paper published in the journal *Metabolomics* (springer.com) titled, *"Migrating from Partial Least Squares Discriminant Analysis to Artificial Neural Networks: A Comparison of Functionally Equivalent Visualisation and Feature Contribution Tools using Jupyter Notebooks"* (Mendez *et al.*, 2019d). This paper compared a standard PLS-DA optimisation, visualisation, evaluation and statistical inference workflow against a set of novel equivalent tools specifically designed to allow ANNs to be applied metabolomics data sets and interpreted

in a similar way. Both workflows were implemented in the Python programming language, and workflows publicly available as Jupyter Notebooks.

References

Beger, R.D., Dunn, W., Schmidt, M.A., Gross, S.S., Kirwan, J.A., Cascante, M., Brennan, L., Wishart, D.S., Oresic, M., Hankemeier, T., Broadhurst, D.I., Lane, A.N., Suhre, K., Kastenmüller, G., Sumner, S.J., Thiele, I., Fiehn, O., Kaddurah-Daouk, R., for "Precision, M. and Pharmacometabolomics Task Group"-Metabolomics Society, I. (2016) Metabolomics enables precision medicine: "A White Paper, Community Perspective". *Metabolomics* **12,** 149.

Breiman, L. (2001) Statistical modeling: the two cultures. *Statistical Science* **16,** 199-215.

Broadhurst, D.I. and Kell, D.B. (2006) Statistical strategies for avoiding false discoveries in metabolomics and related experiments. *Metabolomics* **2,** 171-196.

Cao, L. (2017) Data science: A comprehensive overview. *ACM Computing Surveys (CSUR)* **50,** 1-42.

Cavill, R., Jennen, D., Kleinjans, J. and Briede, J.J. (2016) Transcriptomic and metabolomic data integration. *Briefings in Bioinformatics* **17,** 891-901.

Chen, X.-W. and Lin, X. (2014) Big data deep learning: challenges and perspectives. *IEEE access* **2,** 514-525.

Dunn, W.B., Broadhurst, D., Begley, P., Zelena, E., Francis-Mcintyre, S., Anderson, N., Brown, M., Knowles, J.D., Halsall, A., Haselden, J.N., Nicholls, A.W., Wilson, I.D., Kell, D.B., Goodacre, R., Human Serum Metabolome, H.C. and Human Serum Metabolome, C. (2011a) Procedures for large-scale metabolic profiling of serum and plasma using gas chromatography and liquid chromatography coupled to mass spectrometry. *Nature Protocols* **6,** 1060-1083.

Dunn, W.B., Broadhurst, D.I., Atherton, H.J., Goodacre, R. and Griffin, J.L. (2010) Systems level studies of mammalian metabolomes: the roles of mass spectrometry and nuclear magnetic resonance spectroscopy. *Chemical Society Reviews* **4,** 387-426.

Dunn, W.B., Broadhurst, D.I., Atherton, H.J., Goodacre, R. and Griffin, J.L. (2011b) Systems level studies of mammalian metabolomes: the roles of mass spectrometry and nuclear magnetic resonance spectroscopy. *Chemical Society Reviews* **40,** 387-426.

Dunn, W.B. and Ellis, D.I. (2005) Metabolomics: Current analytical platforms and methodologies. *Trends in Analytical Chemistry* **24,** 285-294.

Fan, J., Han, F. and Liu, H. (2013) Challenges of big data analysis. *National Science Review*. **1**, 293-314.

Fiehn, O. (2002) Metabolomics – the link between genotypes and phenotypes. *Plant Molecular Biology* **48,** 155-171.

Goodacre, R. (2003) Explanatory analysis of spectroscopic data using machine learning of simple, interpretable rules. *Vibrational Spectroscopy* **32,** 33-45.

Goodacre, R., Kell, D.B. and Bianchi, G. (1992) Neural networks and olive oil. *Nature* **359,** 594-594.

Gromski, P.S., Muhamadali, H., Ellis, D.I., Xu, Y., Correa, E., Turner, M.L. and Goodacre, R. (2015) A tutorial review: Metabolomics and partial least squares-discriminant analysis--a marriage of convenience or a shotgun wedding. *Analytica Chimica Acta* **879,** 10-23.

Johnson, C.H., Ivanisevic, J. and Siuzdak, G. (2016) Metabolomics: beyond biomarkers and towards mechanisms. *Nature Reviews Molecular Cell Biology* **17,** 451-459.

Karahalil, B. (2016) Overview of systems biology and omics technologies. *Current Medicinal Chemistry* **23,** 4221-4230.

Kocheturov, A., Pardalos, P.M. and Karakitsiou, A. (2018) Massive datasets and machine learning for computational biomedicine: trends and challenges. *Annals of Operations Research***,** 1-30.

Lecun, Y., Bengio, Y. and Hinton, G. (2015) Deep learning. *Nature* **521,** 436-444.

Löfstedt, T. and Trygg, J. (2011) OnPLS—a novel multiblock method for the modelling of predictive and orthogonal variation. *Journal of Chemometrics* **25,** 441-455.

Maneth, S. and Poulovassilis, A. (2017) Data science. *The Computer Journal* **60,** 285-286.

Mendez, K.M., Broadhurst, D.I. and Reinke, S.N. (2019a) The application of artificial neural networks in metabolomics: a historical perspective. *Metabolomics* **15,** 142.

Mendez, K.M., Pritchard, L., Reinke, S.N. and Broadhurst, D.I. (2019b) Toward collaborative open data science in metabolomics using Jupyter Notebooks and cloud computing. *Metabolomics* **15,** 125.

Mendez, K.M., Reinke, S.N. and Broadhurst, D.I. (2019c) A comparative evaluation of the generalised predictive ability of eight machine learning algorithms across ten clinical metabolomics data sets for binary classification. *Metabolomics* **15,** 150.

Mendez, K.M., Broadhurst, D.I. and Reinke, S.N. (2019d) Migrating from partial least squares discriminant analysis to artificial neural networks: a comparison of functionally equivalent visualisation and feature contribution tools using jupyter notebooks. *Metabolomics* **16,** 17.

Mitchell, T.M. (1997) *Machine Learning*. McGraw-Hill, New York.

Najafabadi, M.M., Villanustre, F., Khoshgoftaar, T.M., Seliya, N., Wald, R. and Muharemagic, E. (2015) Deep learning applications and challenges in big data analytics. *Journal of Big Data* **2,** 1.

Reinke, S.N., Galindo-Prieto, B., Skotare, T., Broadhurst, D.I., Singhania, A., Horowitz, D., Djukanović, R., Hinks, T.S.C., Geladi, P., Trygg, J. and Wheelock, C.E. (2018) OnPLS-

based multi-block data integration: A multivariate approach to interrogating biological interactions in asthma. *Analytical Chemistry* **90,** 13400-13408.

Samek, W., Wiegand, T. and Müller, K.-R. (2017) Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv:1708.08296*.

Samuel, A.L. (1959) Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development* **3,** 210-229.

Weckwerth, W. (2010) Metabolomics: An integral technique in systems biology. *Bioanalysis* **2,** 829-836.

Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L.B., Bourne, P.E., Bouwman, J., Brookes, A.J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C.T., Finkers, R., Gonzalez-Beltran, A., Gray, A.J.G., Groth, P., Goble, C., Grethe, J.S., Heringa, J., 't Hoen, P.A.C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S.J., Martone, M.E., Mons, A., Packer, A.L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M.A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J. and Mons, B. (2016) The FAIR guiding principles for scientific data management and stewardship. *Scientific Data* **3,** 160018.

Wishart, D.S., Feunang, Y.D., Marcu, A., Guo, A.C., Liang, K., Vazquez-Fresno, R., Sajed, T., Johnson, D., Li, C.R., Karu, N., Sayeeda, Z., Lo, E., Assempour, N., Berjanskii, M., Singhal, S., Arndt, D., Liang, Y.J., Badran, H., Grant, J., Serra-Cayuela, A., Liu, Y.F., Mandal, R., Neveu, V., Pon, A., Knox, C., Wilson, M., Manach, C. and Scalbert, A. (2018) HMDB 4.0: the human metabolome database for 2018. *Nucleaic Acids Research* **46,** D608-D617.

Zhang, A., Sun, H., Yan, G., Wang, P. and Wang, X. (2015) Metabolomics for biomarker discovery: moving to the clinic. *BioMed Research International* **2015,** 354671-354671.

Zhang, Q.-s. and Zhu, S.-c. (2018) Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering* **19,** 27-39.

Zhang, Z., Beck, M.W., Winkler, D.A., Huang, B., Sibanda, W., Goyal, H. and written on behalf of, A.M.E.B.-D.C.T.C.G. (2018) Opening the black box of neural networks: methods for interpreting neural network models in clinical applications. *Annals of Translational Medicine* **6,** 216-216.

Chapter Two: The Application of Artificial Neural Networks in Metabolomics: A Historical Perspective

Authors

Kevin M Mendez[1], David I Broadhurst[1*], Stacey N Reinke[1*]

[1]Centre for Integrative Metabolomics & Computational Biology, School of Science, Edith Cowan University, Joondalup, 6027 Australia

*Corresponding authors:

email: d.broadhurst@ecu.edu.au, stacey.n.reinke@ecu.edu.au

phone: +61 (0)8-6304-2705

This is a post-peer-review, pre-copyedit version of an article published in *Metabolomics*. The final authenticated version is available online at: https://doi.org/10.1007/s11306-019-1588-0.

Abstract

Background: Metabolomics data, with its complex covariance structure, is typically modelled by projection-based machine learning (ML) methods such as partial least squares (PLS) regression, which project data into a latent structure. Biological data are often non-linear, so it is reasonable to hypothesize that metabolomics data may also have a non-linear latent structure, which in turn would be best modelled using non-linear equations. A non-linear ML method with a similar projection equation structure to PLS is artificial neural networks (ANNs). While ANNs were first applied to metabolic profiling data in the 1990s, the lack of community acceptance combined with limitations in computational capacity and the lack of volume of data for robust non-linear model optimisation inhibited their widespread use. Due to recent advances in computational power, modelling improvements, community acceptance, and the more demanding needs for data science, ANNs have made a recent resurgence in interest across research communities, including a small yet growing usage in metabolomics. As metabolomics experiments become more complex and start to be integrated with other omics data, there is potential for ANNs to become a viable alternative to linear projection methods.

Aim of review: We aim to first describe ANNs and their structural equivalence to linear projection-based methods, including PLS regression. We then review the historical, current, and future uses of ANNs in the field of metabolomics.

Key scientific concept of review: Is metabolomics ready for the return of artificial neural networks?

Key Words: Artificial Neural Network, Machine Learning, Deep Learning, Partial Least Squares, Metabolomics

2.1. Introduction

Metabolomics data are complex, due to their high dimensionality and a high degree of multicollinearity between variables (Worley and Powers, 2013). These complex covariant data structures are traditionally modelled using projection methods that reduce dimensionality by projecting input variables into a latent structure (Seber, 2004). Conceptually, latent structures are hidden (or unobserved) variables that are derived from a transformation of the input variables. The accepted standard unsupervised linear projection method is principal component analysis (PCA), where the first n latent variables (referred to as principal components) describe the dominant variance in the input data matrix (Hotelling, 1933; Wold *et al.*, 1987).

PCA can be extended into a classification tool called principal component regression (PCR), by projecting the principal components into a multiple linear regression (Jolliffe, 1982; Kendall, 1957). A more familiar alternative to PCR is partial least squares (PLS) regression, or alternatively termed projection to latent structure, where the latent variables describe the covariance between the input data (i.e. metabolite data) and the output data (i.e. sample classification) as opposed to the variance within the input data (Geladi and Kowalski, 1986; Wold, 1975). Linear projection methods of this type can be visually represented as a network, which can be deconvolved into a single matrix equation (for example, PLS regression in Fig. 2.1b). Through this network representation, the latent structure is represented as a set of matrices. Individual vectors within these matrices can be easily plotted and therefore visualised, thus enabling clear interpretation of the model. It is primarily due to this ease of

interpretation of the latent structure that PLS has become a commonly used method in metabolomics (Gromski *et al.*, 2015).

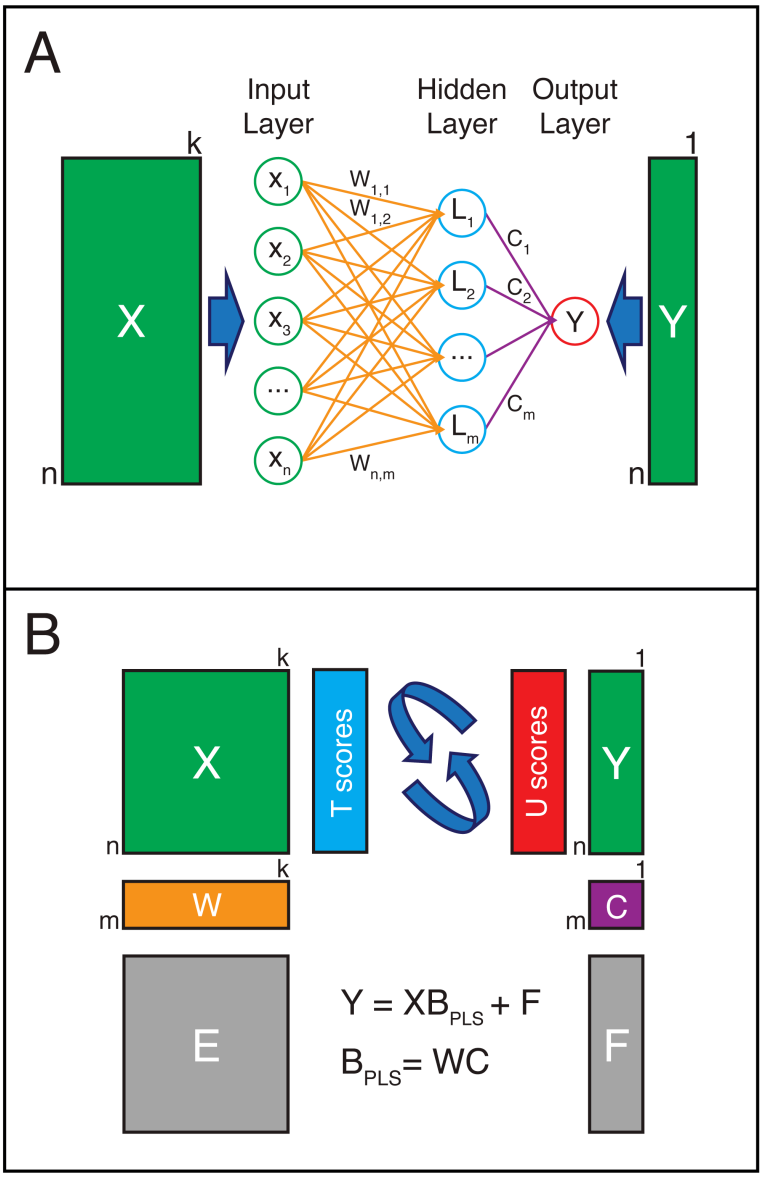


Figure 2.1: Illustration of the structural equivalence of an ANN and PLSR= model. **a** The network representation of a PLS model as a single layer feed-forward neural network. **b** The matrix relationships in PLS regression (adapted from (Eriksson *et al.*, 2013)). The PLS scores comprise of **T** and **U**. The **X** weights, **W**, are equivalent to the complete set of synaptic weights, $\mathbf{w_{i,j}}$, linking the input layer to the hidden layer. The **Y** weights, **C**, are equivalent to the synaptic weights, $\mathbf{c_{i,j}}$, linking the hidden layer to the output layer. The variance left out of the model for the residual matrices **E** and **F**.

While easily interpretable, PCA and PLS are linear methods. As biological data are often non-linear (Mosconi *et al.*, 2008), it is possible that metabolomics data may also have a non-linear latent structure. As such, more complex non-linear machine learning (ML) methods such as random forest (RF), support vector machine (SVM) with a non-linear kernel, and artificial neural networks (ANNs) may be applicable for analysing metabolomics data. As the name suggests, an ANN can be visualised as a network of interconnected neurons, where each interconnection has a 'weight', and each neuron acts as a summation of weighted inputs passed through a linear or nonlinear activation function (see Fig. 2.1a). As such, a complex equation can be quickly built up by stacking layers of interconnected nodes. It is important to note that a three-layer ANN with linear activation functions is structurally equivalent to a PLS model, such that it can be algebraically simplified to a linear equation of the form: $y = XB$, where X is the input (metabolite) matrix and B is a coefficients vector indicating the importance of each variable (metabolite) on the predicted value, y (Fig. 2.1b). In this form the two methods differ only in the manner in which the model's weights are optimised. Thus, ANNs can also be considered as a method for projection into a smaller dimensional latent space. However, PLS models can only be a single projection, whereas the more layers of neurons added to the ANN model the more complex and refined the data abstraction (Chollet, 2018). Due to this structural equivalence, visualisation and interrogation techniques standardised for PLS may be directly adapted to shallow (2 or 3 layer) ANN architectures, as will be discussed in detail later. However, extending visualisation and interrogation techniques to more complex, or deep, ANN architectures (with multiple hidden layers and non-linear activation functions) is a challenge, and an area of ongoing research within the ML community (Samek *et al.*, 2017; Zhang and Zhu, 2018; Zhang *et al.*, 2018). Provided suitable methods of visualisation and interpretability can be developed, ANNs may now be highly applicable to metabolomics data. This review will introduce the reader to the fundamental computational concepts of ANNs, provide a brief

historical context of their use in metabolomics, discuss the advantages and disadvantages of this method when applied to ‘omic data, and finally look toward the future application and challenges.

2.2. What is an Artificial Neural Network (ANN)?

As the name suggests, ANNs were inspired by the biological interconnections of neurons in a brain (Jolliffe, 2002; McCulloch and Pitts, 1943). The original aim of ANNs was to understand brain and nervous system function by mathematically modelling signal transfer between neurons (Fukushima, 1980; Robinson, 1992). This research ultimately led to the simplified mathematical models of an artificial neuron. ANN usage quickly extended beyond attempts to understand neurophysiology as researchers built computational networks of artificial neurons; this was done with the abstract aim of mimicking the human ability to “learn” from example (training) data in order to create generalised associations to correctly classify unknown (test) data (Zurada, 1992). The resulting neural networks were only loosely comparable to the biological equivalent and thus the term was prefixed with the word “artificial”.

The most common ANN is the feed-forward neural network. Here the network is represented by multiple layers of neurons in the typical projection-based framework (Fig. 2.2). These neuron layers include an input layer, one or more hidden (i.e. latent) layers, and an output layer, with neurons being connected by synaptic weights. Each neuron represents a transfer function that consists of two sequential steps (Basheer and Hajmeer, 2000; Gardner and Dorling, 1998; Schalkoff, 1997). First, a summation function calculates the weighted sum of the input neurons offset by a constant value (bias). Second, an activation function transforms the sums using either a linear or non-linear (such as hyperbolic tangent and sigmoidal) transformation. During

ANN model training, the weights between each layer of neurons are iteratively optimised in a two-phase cycle: forward propagation through the network, after which an error term is calculated based on the difference between the target and actual outputs, and backpropagation to adjust the weights. This unique method of model parameter optimisation used for ANNs is called backpropagation. It is this conceptually simple model optimisation method that allows for the flexibility of ANN structures, from one to multiple hidden layers, the latter being referred to as deep neural networks or deep learning.

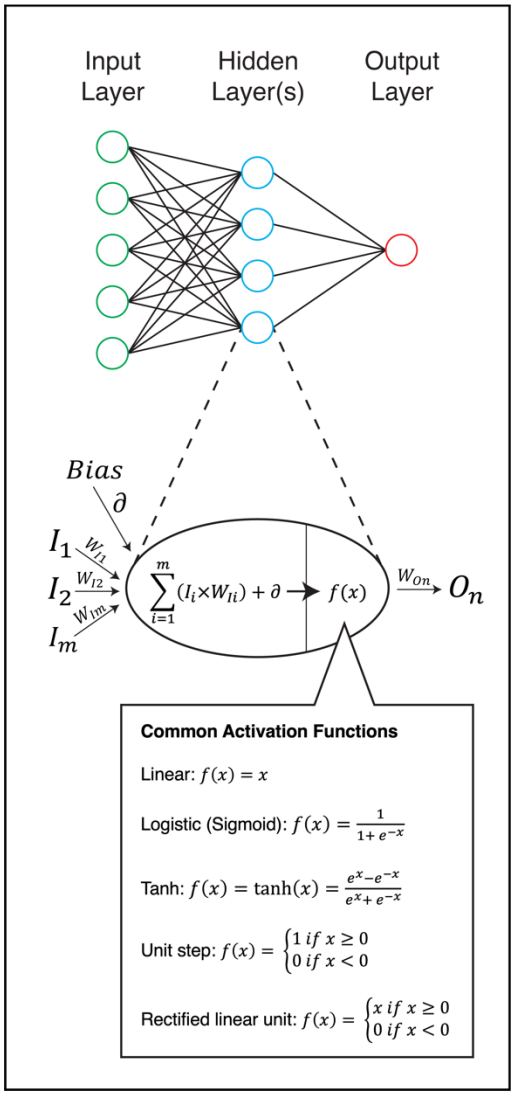


Figure 2.2: Graphical representation of an ANN, where each circle represents a neuron in each layer, and the arrows represent the synaptic weight between the output of a neuron to the input of another neuron. Each neuron represents a simple transfer function split into two stages; the summation of the inputs followed by an activation function.

In the context of matrix algebra, an ANN with one hidden layer and linear activation functions is structurally equivalent to linear projection models (such as PLS). As shown in Fig. 2.1a, each neuron in the input, hidden, and output layer corresponds to the X matrix, X score, and Y matrix (outcome) respectively. Weights connecting the input and hidden layer, and the hidden and output layers are therefore equivalent to the X loadings (W) and Y loadings (C) matrices in PLS, respectively. While there are a number of variants of PLS with alternative algorithms, these variants differ in their calculation/optimisation of the values in the scores and loadings matrices as opposed to changing the structure of the model. The final structure of a PLS model is always linear; however, ANNs can be either linear or non-linear, dependent on the transfer functions used (such as linear or sigmoidal) for each neuron. Due to the similarity of ANNs and PLS, visualisation and interrogation techniques standardised for PLS (Seber, 2004) may be readily adapted to three-layer ANN models. For example, with a three-layer ANN model, it is possible to plot data projected into the latent neuron space to visualise latent structure in the data (equivalent to the PLS latent variable projection plots). It is also possible to present loadings plots showing the importance of input variables to each hidden neuron (equivalent to the PLS loadings plots), and it is also possible to generate a variable importance score for ANN equivalent to the PLS variable in projection score (VIP score) for determining the importance of each input variable to the overall model (Olden *et al.*, 2004). While these visualisations draw on parallels between PLS and ANN model structure, as ANN architectures get deeper (more layers), and the data abstraction more complex, such visualisation methods rapidly become cumbersome and of little value.

2.3. Deep Learning

The term deep learning refers to an extension of feedforward ANNs focussing on the concept of learning consecutive layers of increasingly meaningful data representations, with each neural layer realising a specific data abstraction. By stacking multiple layers (often tens or hundreds of layers) a complex noisy data manifold can be unpacked and transformed into an accurate high-level abstract classification (Goodfellow *et al.*, 2016).

A factor fundamental to the increased popularity of deep learning across multiple applications is their ability to integrate, and automate, a critical step in ML known as feature engineering (Chollet, 2018). Feature engineering is the process of transforming raw data into a form that can be manageably processed by traditional machine learning and statistical methods. In the context of metabolomics this would be the process of spectral deconvolution. In its simplest form (often used in spectroscopic methods such as nuclear magnetic resonance spectroscopy) this process is simply data binning, where the area under the curve of selected/predefined segments (bins) in the full spectrum is computed for each sample. For three-dimensional metabolomics spectra (e.g. liquid chromatography mass spectrometry) more complex semi-automated algorithms are needed to identify unique metabolite peaks across multiple samples [e.g. XCMS (Tautenhahn *et al.*, 2012)]. Additionally, it is often common practice to remove high variant features, and perform a number of normalization/transformation steps, before modelling. Typically, the complete feature engineering process is time consuming and requires a level of domain expertise. Only after this has been performed can the data be applied to projection methods such as PLS or a shallow ANN. The promise of deep learning is the ability to combine the mapping of data through spectral deconvolution with projection-based classification into a single step (see Fig. 2.3), thus removing the multiple steps of

deconvolution, data scaling/normalisation and feature selection, commonly used in the metabolomics workflow. This could potentially improve performance, but most importantly improving deployment efficiency and avoiding domain expertise. This has not yet been done; however, a recent publication by Risum and Bro (2019) successfully implemented a deep learning algorithm to perform automated spectral deconvolution. It is reasonable to speculate that we are now in reach of a single deep learning algorithm for accurately classifying raw spectra straight from the instrument. However, the limiting factor for success will likely be obtaining sufficiently large data sets required to train such computational "greedy" algorithms (Goodfellow *et al.*, 2016). This will be discussed later.

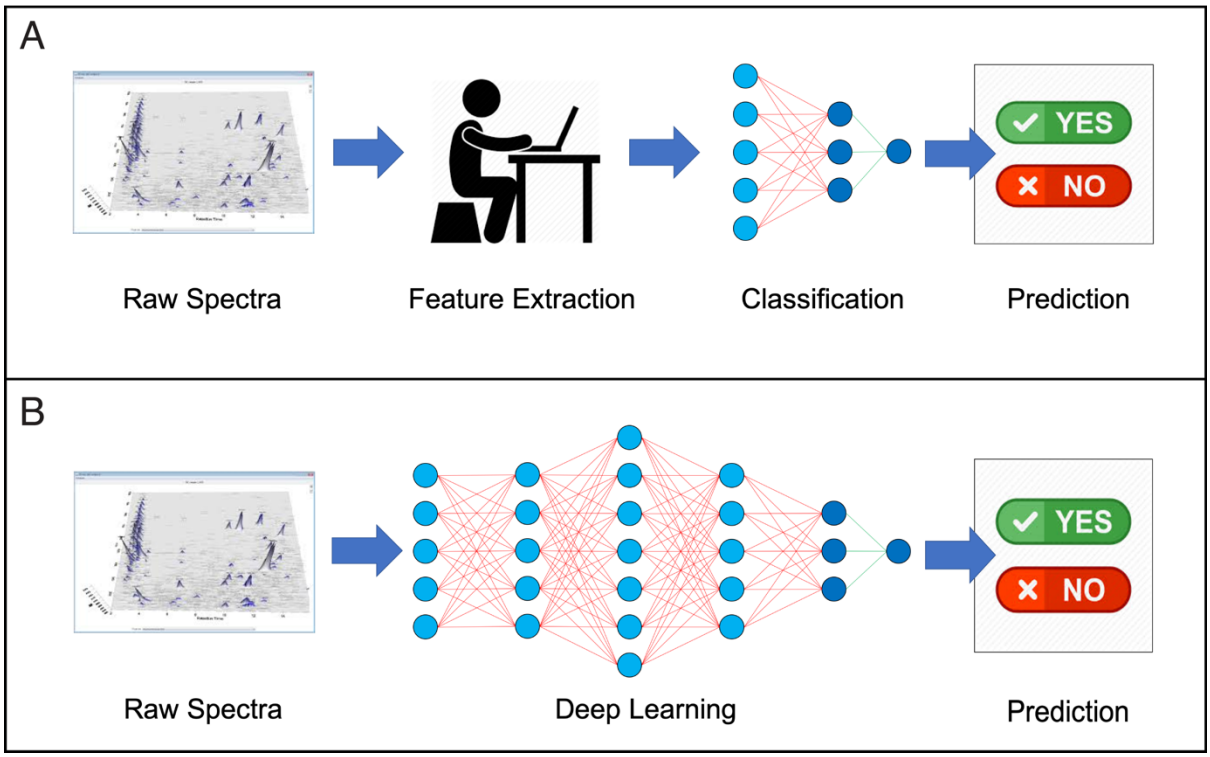


Figure 2.3: Illustration of the differences in complexity and implementation between traditional machine learning (**a**) and deep learning (convolutional neural networks) (**b**). Traditionally, the first step toward building a predictive model is to convert raw data into a form that can be manageably processed. For metabolomics this would be a matrix of metabolite concentrations. This step is known as 'feature selection' or 'feature engineering'. The semi-manually extracted data will then be modelled using a relatively simple predictive algorithm (ANN, SVM, PLS). For deep learning the adopted philosophy is to incorporate both these steps into a single algorithm. This requires multiple layers of neurons stacked to sequentially deconvolve data from its raw state, to abstract latent structure, to effective prediction.

Alternatively, feature importance can be determined by systematically removing unimportant features (network pruning) using L1 and L2 regularisation, as implemented by Keras (https://keras.io), a popular open-source neural network library (Chollet, 2015). A review of the many proposed approaches to visualising and interpreting deep learning algorithms is beyond the scope of this publication. For further information the authors point the reader toward Samek *et al.* (2019). Based on current evidence, the interpretability of deep learning models will be perpetually a challenge, particularly as compute power increases and more complex networks are developed. This leads us to question the necessity of interpretability. Interpretability is not required for verifiability. If suitable validation methods are employed to verify that the model is working correctly, the need to interpret and understand the inner workings of the model may not be needed for use in critical applications. For studies where classification is vital (e.g. success of a treatment or therapy), the high predictability of a non-interpretable model may be more appropriate, and ethical, than the low predictability of a highly interpretable model. The ethical use of artificial intelligence is now being regularly discussed (Adadi and Berrada, 2018; Bostrom and Yudkowsky, 2014; Holzinger *et al.*, 2017; Russell *et al.*, 2015).

2.4. Historical Perspectives

Artificial neural networks were first applied to metabolomic profiling ca. 1992 by Goodacre *et al.* (1992) to discriminate between pure and adulterated extra-virgin olive oil using pyrolysis mass spectrometry. This research group also used novel ANN approaches to correct drift in spectra collected over time (Goodacre and Kell, 1996) and on different instruments (Goodacre *et al.*, 1997). Additionally, they also used a variant of ANNs (now referred to as autoencoders) as an alternative method to PCA (Goodacre *et al.*, 1996a). Other research groups applied ANNs

for metabolite quantification as an alternative to approach to the conventional line-shape fitting approach (AlaKorpela *et al.*, 1997; Kaartinen *et al.*, 1998). More commonly, ANNs were used for regression, and binary and multi-class classification in a variety of contexts (AlaKorpela *et al.*, 1997; Anthony *et al.*, 1995; Goodacre and Kell, 1993; Goodacre *et al.*, 1992, 1993; Goodacre *et al.*, 1998; Goodacre *et al.*, 1996b; Goodacre *et al.*, 1994; Kaartinen *et al.*, 1998; Usenius *et al.*, 1996). In this time period, alternative ML methods for binary classification were also applied to metabolomics such as logistic regression (LR) (Lang *et al.*, 1994; Moen *et al.*, 1996; Peters *et al.*, 1991; Wolff *et al.*, 1993) and PLS (Fayolle *et al.*, 1997; Frisvad, 1992; Goodacre *et al.*, 1994; Harthun *et al.*, 1998; Sjogren *et al.*, 1996). By 1999, the number of publications per year in metabolomics using LR or PLS noticeably deviated from ANN (which stagnated), and from 2006 onwards, PLS was the most applied ML method per year (Fig. 2.4a).

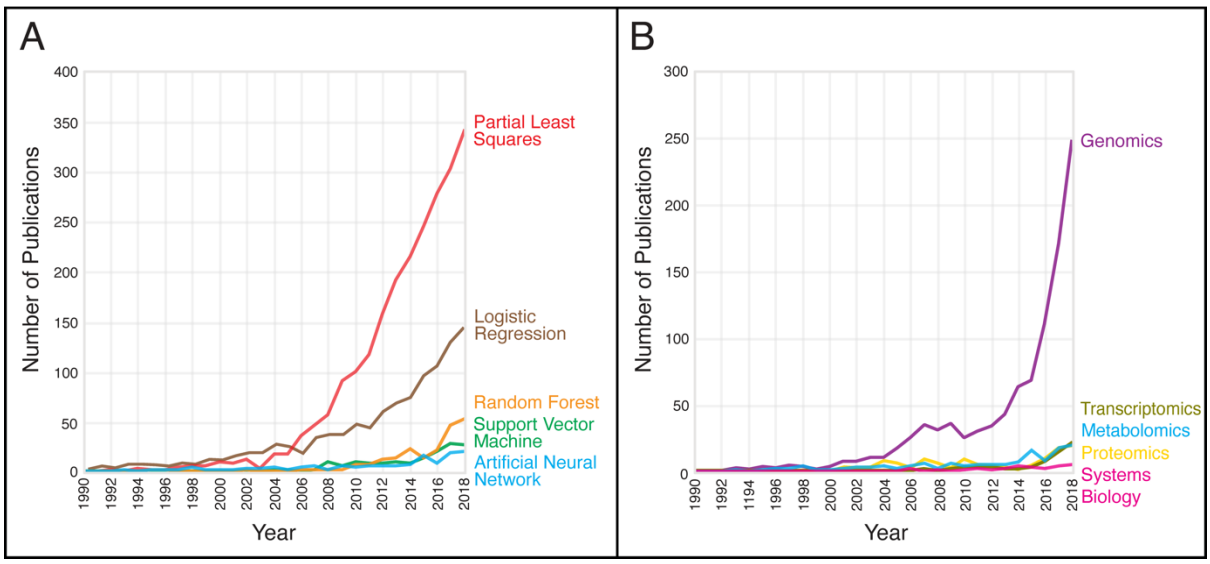


Figure 2.4: Number of publications per year (from Web of Science). **a** Publications that include the key term metabolite*, metabolom* or metabonom* with the key term logistic regression (brown), partial least squares or projection to latent structure (red), random forest (orange), support vector machine (green), or artificial neural network or deep learning (blue). **b** Publications that include the key term artificial neural network or deep learning with the key term gene, genes or genom* (purple), metabolite*, metabolom* or metabonom* (blue), proteom* (yellow), or transcriptome* (olive), or systems biology (pink). * Denotes a wild card (for example metabolom* can include the key terms metabolome and metabolomics).

In the early days of metabolomics, spectral fingerprinting (Dunn *et al.*, 2011) was common practice. Here, non-specific snapshots of the metabolome were acquired, typically with holistic and rapid acquisition analytical platforms. Subsequent classification was based on overall spectral pattern, and identification of the specific components causing the differences was not of primary interest. As such, ANNs were useful due to their strength as a classification method. However, as technology advanced, the ability and desire to identify component metabolites important to a given classification came to be of primary interest. Researchers using ANNs were not able to readily infer variable contributions to the model, and therefore could not provide information about the underlying biology. At this time, the PLS algorithm was becoming more readily available to researchers through companies such as Umetrics (Umeå, Sweden) and Eigenvector Research (Washington, USA). PLS was more easily interpretable, particularly with the introduction of variable importance in projection (VIP) that directly linked the metabolite abundance to the outcome, therefore informing biology (Wold *et al.*, 1993). By the early 2000s, the use of VIP plots, and the interrogation of the latent structure with X scores and loadings plots became standard for the interpretation of PLS models in metabolomics (Azmi *et al.*, 2002; Pérez-Enciso and Tenenhaus, 2003).

With the shift towards interpretability, PLS became the standard supervised multivariate method used by the metabolomics community. ANNs (and other non-linear ML methods) were relegated to being a 'black box' approach useful for classification but not for meaningful interpretation of underlying biology (Goodacre, 2003). Combined with community resistance, ANNs were also not practical for the time. In the early 2000s metabolomics studies were generally small compared to now. Typically, metabolomics data sets consisted of only 10's of samples. It was not until around 2010 that metabolite data sets consisting of over 1,000 samples in a single study started to regularly appear (e.g. HUSERMET (Dunn *et al.*, 2015)), despite a

couple of early attempts to compare large sample numbers from multiple sites using NMR spectroscopy (COMET (Lindon *et al.*, 2003) and INTERMAP (Holmes *et al.*, 2008)). The lack of data meant that non-linear models, with far greater degrees of freedom compared to linear models, were not stable and prone to extreme overfitting. As members of the metabolomics research community started to investigate the misuse of statistical methods and suggest ways to avoid false discovery (Broadhurst and Kell, 2006), was clear that complex models built with small data sets often could not stand up to peer review. ANNs were also held back by computer technology. ANNs require many thousands of calculations (training iterations) to fully optimise. At the time of their demise it would often take several hours to train a simple ANN model, compared to several seconds for PLS. Also, there were no commercial or open source ANN software available for casual users to investigate, so software availability also played a crucial role in the stunted growth.

The combination of lack of transparency, need for larger datasets due to improved understanding of post hoc performance statistics, lack of compute power, and poor software availability made the demise of ANNs in the early 2000s inevitable. The application of ANNs in metabolomics diminished to an intellectual curiosity with no practical application.

2.5. The Renaissance of ANNs

While ANNs usage diminished within the metabolomics community, it gained traction in areas such as image processing (Egmont-Petersen *et al.*, 2002; Rawat and Wang, 2017; Simard *et al.*, 2003), and natural language processing (Bengio *et al.*, 2003; Cambria and White, 2014; Morin and Bengio, 2005). In this context, the need for high accuracy of classification overshadowed the need for the interpretability of the models. A major breakthrough for ANNs

was the implementation of graphical processing units, which greatly accelerated training (Oh and Jung, 2004). This allowed for more complex ANN architectures with multiple hidden layers, which are now referred to as deep neural networks (DNN) or simply "deep learning". By 2012, ANNs started to dominate various pattern recognition contests (Cireşan *et al.*, 2012a, 2013; Cireşan *et al.*, 2012b; Schmidhuber, 2012); however, the winning of the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) was particularly instrumental to the influx of interest in ANNs and deep learning (Alom *et al.*, 2018). In this challenge, with approximately 1.2 million images and 1000 categories, a class of DNNs called a convolution neural network achieved a top-5 test error rate of 15.3%, significantly lower than the second-best entry with 26.2% (Krizhevsky *et al.*, 2012). Due to this leap in classification performance gained, ANNs (in particular DNNs) are a standard method for research in image processing regardless of being a 'black box' approach. This in turn has led to greater acceptance by research communities in other fields and society at large. Through the efforts of companies such as Google, IBM, and Microsoft, where high-profile achievements using deep learning such as IBM's Watson beating two champions in Jeopardy!™ (Ferrucci, 2012), and Google's AlphaGo beating a grandmaster in Go (Chen, 2016; Wang *et al.*, 2016), has placed the concept of artificial intelligence and deep learning in the public zeitgeist.

Fuelled by industry seeking to gain competitive leverage from large in-house databases, deep learning is particularly now seeing widespread adoption. In this era of 'big data', DNNs have become efficient ways to handle the volume of large-scale data mining, with the size of the digital universe estimated at 4.4 zettabytes in 2013 (and an exponential trajectory towards 44 zettabytes by 2020 (Erevelles *et al.*, 2016). The use of ANNs is already widespread, ANNs are deeply integrated into a wide range of data analytics, such as in spam detection (Wu *et al.*, 2017), news aggregation (Zheng *et al.*, 2018), imaging tagging (Guo *et al.*, 2016; Shen *et al.*,

2017), robotics (Lenz *et al.*, 2015; Levine *et al.*, 2018), autonomous vehicles technologies (Falcini *et al.*, 2017; Luckow *et al.*, 2016), and natural language processing (Manning *et al.*, 2014; Young *et al.*, 2018). For example, ANNs are used for natural language processing in all common virtual assistants such as Siri, Amazon Alexa, Google Assistant and Cortana by Apple Inc., Amazon.com Inc., Google LLC, and Microsoft Corporation respectively. The massive research and development from these large-scale technology companies has also resulted in both software and hardware available to academia making ANNs increasingly applicable in research.

In recent years, ANNs have become much more accessible for academic research due to continued advancements in affordable computational power and open-source software. Considering Moore's law that states the number of transistors in an integrated circuit doubles approximately every 2 years (Moore, 1975), computational power has increased by 16384-fold from 1990 to 2018. Additionally, with more computational power over time, there has been a shift towards cloud computing through infrastructure providers such as Amazon Web Services, Google Cloud, or Microsoft Azure, allowing for on-demand availability of computer resources without the upfront cost of hardware. With this powerful hardware, there has been a recent release of ANN software libraries to the public such as TensorFlow (www.tensorflow.org/), and the Microsoft Cognitive Toolkit (https://cntk.ai/), by Google LLC in 2015 and the Microsoft Corporation in 2016. These powerful and easy to use open-source libraries has allowed for the quick prototyping of various ANN architecture and applications, where previously it would require the massive development of core code to build even simple ANNs. Combined with accessible high-power computing, ANN research and application across research fields is much more accessible than previously before.

While the use of ANNs within metabolomics has shown only recent growth, in genomics the use of ANNs (particularly deep learning) has grown exponentially and is now well established (Fig. 2.4b). It has been suggested that this growth was due to breakthroughs in image processing and natural language processing combined with seminal publications in 2015 that demonstrated the applicability to DNA sequence data (Eraslan *et al.*, 2019). Since then, there has been a wide application of ANNs within genomics including 3D organisation, DNA accessibility and chromatin, DNA methylation, tumour genomes, base calling, pathogenic variants, transcription, and RNA analysis (for further details, refer to the following reviews: (Yue and Wang, 2018; Zou *et al.*, 2019)). An excellent review of the application of convolutional neural networks in population genetics has also been recently published (Flagel *et al.*, 2018).

In metabolomics four distinct areas of ANN research have emerged over the last 5 years (Table 2.1). Firstly, ANNs have been applied to feature extraction, specifically, spectral deconvolution for LC- and GC–MS to improve chemical identification (Allen *et al.*, 2016; Hall *et al.*, 2018; Hall *et al.*, 2015; Ou *et al.*, 2015; Risum and Bro, 2019; Samaraweera *et al.*, 2018; Woldegebriel and Derks, 2017). Secondly, ANNs are being applied (as deep unsupervised neural network) as an imaging tool for 3-D DESI imaging data to cluster tumour tissue into sub-regions (Inglese *et al.*, 2017). Thirdly, ANNs were used to model drug interactions with the prediction of the down-stream metabolites yield from proposed drug toxicology (Barnette *et al.*, 2018; Hughes and Swamidass, 2017). Lastly, ANNs, in their traditional backpropagation architecture, have returned to use as a supervised classification tool in a number of contexts (Alakwaa *et al.*, 2018; Asakura *et al.*, 2018; Chagas-Paula *et al.*, 2015; Date and Kikuchi, 2018; Trainor *et al.*, 2017). With this high flexibility of ANNs, in part due to the ease of use of

software libraries, the prototyping of various ANNs architecture for novel applications is now practical.

Table 2.1: Artificial neural network publication in metabolomics since 2015. (* deep learning)

Application Area	Publication
Feature Extraction	Allen *et al.*, 2016; Hall *et al.*, 2018; Hall *et al.*, 2015; Ou *et al.*, 2015; Risum and Bro, 2019*; Samaraweera *et al.*, 2018; Woldegebriel and Derks, 2017
Imaging	Inglese *et al.*, 2017*
Drug Interaction	Barnette *et al.*, 2018*; Hughes and Swamidass, 2017*
Classification	Alakwaa *et al.*, 2018*; Aliakbarzadeh *et al.*, 2016; Asakura *et al.*, 2018*; Banerjee *et al.*, 2017; Chagas-Paula *et al.*, 2015; Cortina *et al.*, 2018; Date and Kikuchi, 2018*; Dong *et al.*, 2017; Guo *et al.*, 2015; Hettinga *et al.*, 2015; Long *et al.*, 2017*; Pecnik *et al.*, 2018; Qiu *et al.*, 2016; Wang *et al.*, 2017; Zhang *et al.*, 2017
Review	Fatemi *et al.*, 2015; Grapov *et al.*, 2018; Huang *et al.*, 2015; Maudsley *et al.*, 2018; Trainor *et al.*, 2017; Trivedi *et al.*, 2017; Zhao *et al.*, 2018

The recent reintroduction of shallow ANNs as a metabolomics classifier warrants further discussion. The computational and software limitations of 20 years ago have almost disappeared meaning that multiple ML methods can be applied to a data set with little cost. However, peer reviewed studies comparing multiple linear and non-linear machine learning methods to metabolomics data are currently limited, with no clear conclusion. One study based on simulated data (with 40 samples and 25 peaks) ranked ML methods (from least to greatest error) as SVM, RF, Naïve Bayes, sparse PLS, ANN, PLS, and k-Nearest Neighbours (Trainor

et al., 2017). Another study (with 271 samples and 162 peaks) indicated that the deep learning model had the highest predictive accuracy compared to the other 6 ML methods tested: RF, SVM, recursive partitioning and regression trees, linear discriminant analysis, and generalised boosted models (Alakwaa *et al.*, 2018). At the 2019 Metabolomics Society meeting in The Hague, preliminary data was presented that suggested for a simple binary discrimination most of the popular ML methods (PLS, RF, SVM and ANN) produced very similar performance across ten randomly selected data sets of differing sizes and differing predictive power. All methods overtrained, but the non-linear models were the least robust in terms of repeatability (Broadhurst, 2019).

Limited training data remains a concern for all ML methods when applied to high dimensional data. As pointed out by Breiman (2001) the curse of dimensionality dictates that the expected generalization error is proportional to the complexity of the model and inversely proportional to the number of samples used to build the model. Thus, for high dimensional data (hundreds of metabolite peaks) a complex model trained on a small data set will tend to have poor generalised performance as a classifier. There is no "magic" equation for the number of samples needed for robust metabolomics ML model; it is dependent on many factors, including: the dimensionality of the data, the strength of effect size, the degree of covariance (strength of latent structure), the heterogeneity of the sample population, the repeatability of the measurement. However, what is clear from the general literature, is that the deeper the ANN (the more layers you stack) the number of samples required to effectively train an ANN grows at an alarming rate. It is typical to require many thousands of samples. For example, the ILSVRC2012 ImageNet dataset had over 14 million images (Russakovsky *et al.*, 2015).

In genomics sample numbers has been less of an issue as standardised analytical platforms have allowed the concatenation of data sets from multiple laboratories (Hamid *et al.*, 2009), such that datasets of over 100,000 samples can be collated (Roundtable on Translating Genomic-Based Research for Health *et al.*, 2016). Indeed, in 2019 twenty-one European countries signed a declaration to transnationally share data on at least 1 million human genomes by 2022 (Saunders *et al.*, 2019). For untargeted metabolomics the ability to directly concatenate deconvoluted data sets from multiple labs is currently intractable due to differences in instrument metabolite coverage and sensitivity from lab to lab (Beger *et al.*, 2019; Broadhurst *et al.*, 2018). However, data availability (and concatenation) is of less concern in the area of metabolite identification, and thus more amenable to ANN and deep learning. The most comprehensive, and open access, example of this application is provided by Risum and Bro (2019). Here they describe the implementation of a deep learning convolutional neural network used to automatically evaluate whether chromatographic components of raw gas chromatography mass spectrometry (GC–MS) data reflect chemical information or baseline. In effect, this determines whether a peak is real or noise. Probably the two most important messages from this paper are (i) the training data consisted of 70,000 elution profile samples, illustrating how "greedy" deep learning can be, and (ii) each of these profiles had to be (semi)manually labelled by an expert (in this instance using the PARAFAC2 algorithm). This may seem an obvious statement, but the need for correctly classified data train an ANN is vital. Deep learning ANNs will only learn by example. If samples are mislabelled, then the ANN may blindly learn the wrong association. Also, it is important to carefully select data representative of the correct classification domain. It is worth reminding the reader of the apocryphal story of an early application where an ANN was built to classify images of military tanks into either of Russian or American origin. Unfortunately, all the photos of Russian tanks were taken on overcast days. The result was a very good predictor of weather but not tanks.

2.6. Future Perspectives and Challenges

With the extensive adoption of ANNs in genomic research (Fig. 2.4b) we are likely to see a continued growth in popularity within the other 'omic sciences, including metabolomics. As deep learning is most amenable to the process of sequentially deconvolving data, through multiple neural layers, into meaningful abstraction it is probable that deep convolutional neural networks will be applied to raw metabolomics data such that both feature extraction and multivariate classification will be integrated into a single deep learning model (as discussed in Sect. 2.3). To this end, data sets of the requisite size are starting to appear (for example, Deelen *et al.* (2019) used the Nightingale NMR platform to create a data set of 44,168 individuals). However, even if deep learning becomes computationally tractable, there is a continued danger that a lack of explanatory mechanisms will once again limit widespread adoption, despite improvements in computational understanding and software availability.

In order to maintain transparency, it may be necessary to limit the depth of networks. Unfortunately, based on current evidence (Alakwaa *et al.*, 2018; Trainor *et al.*, 2017), simple fully connected three-layer ANNs perform no better than other much quicker ML methods, and are equally constrained by the same bias vs. variance limitations (Broadhurst and Kell, 2006); increased samples numbers do not seem to improve ANNs position (Broadhurst, 2019).

That said, there is one aspect of multilayer ANNs that has not yet been fully exploited. ANNs have the potential for extraordinary flexibility of network architecture. It is possible to impose sparse network architectures that focus on specific predefined covariance structures in the available data. This is particularly interesting for metabolomics and multi- omic studies, where data for a sample population is collected in blocks. These blocks could be multiple analytical

platforms (e.g. multiple modes of liquid chromatography mass spectrometry), or multiple biofluids (urine, plasma, faeces), or multiple 'omic platforms (genomic, proteomic, metabolomic etc.), or a mixture of all three. To fully understand the underlying biochemical mechanisms of a system it may be more productive to focus on determining, and visualising, covariance within and between blocks rather than predicting an outcome. Thus, the focus of any subsequent ANN algorithm will be to determine those features (e.g. metabolites, proteins, genes) that jointly explain the biological outcomes in an observed study.

There are four cardinal approaches for integration between two or more blocks of data; correlation-, concatenation-, multivariate-, and meta-analysis- based integration (Cavill *et al.*, 2016). The many computational algorithms used to integrate multi-omic data have recently been reviewed in-depth (Pinu *et al.*, 2019). Of particular interest here is the concerted efforts of several research groups to implement machine learning based methods. Thus far, such approaches have focussed on hierarchical application of existing linear projection models. For example, OnPLS (Löfstedt and Trygg, 2011; Reinke *et al.*, 2018) is a combinatorial amalgamation of multiple PLS models, and Mixomics (Rohart *et al.*, 2017) is a stepwise integration of canonical correlation analysis and sparse PLS. The underlying limitations of linear matrix algebra constrain the ability for such models to cleanly model complex interactions between data blocks, with a constant danger of overfitting. Also, the complex hierarchy of model loadings and coefficients ultimately makes interpretation extremely difficult (Reinke *et al.*, 2018).

The inherent flexibility of ANN architecture allows complex relationships to be combined into a single model. This means that rather than relying on hierarchical optimisation of multiple sub-models (e.g. hierarchical PLS), it may be possible to construct an ANN architecture to

combine multiple data blocks into a single model without resorting to over-simplified data concatenation. Various architectures which are readily implemented as an ANN are illustrated in Fig. 2.5. Already, a simple multi-block ANN has been applied to multi-platform genomic data by Sharifi-Noghabi *et al.* (2019). Here 5593 samples were used to predict chemotherapy drug response resulting in high predictive power, but no methods of interpretability were provided. There have been several other rudimentary attempts at multi-omics analysis using ANNs. These include transcriptomics integrated with epigenetics (Bica *et al.*, 2018), proteomics with metabolomics (Chung *et al.*, 2019), RNA-Seq with Microarray Gene Expression (Francescatto *et al.*, 2018), and mRNA-seq with miRNA-seq (Chaudhary *et al.*, 2018; Huang *et al.*, 2019). Most of these methods used an ANN architecture similar, in principle, to that shown in Fig. 2.5b, with the exception of the proteomics/metabolomics model which used an autoencoder ANN (similar architecture to Fig. 2.5c), and were successfully trained with a number of samples ranging from 200 to 600.

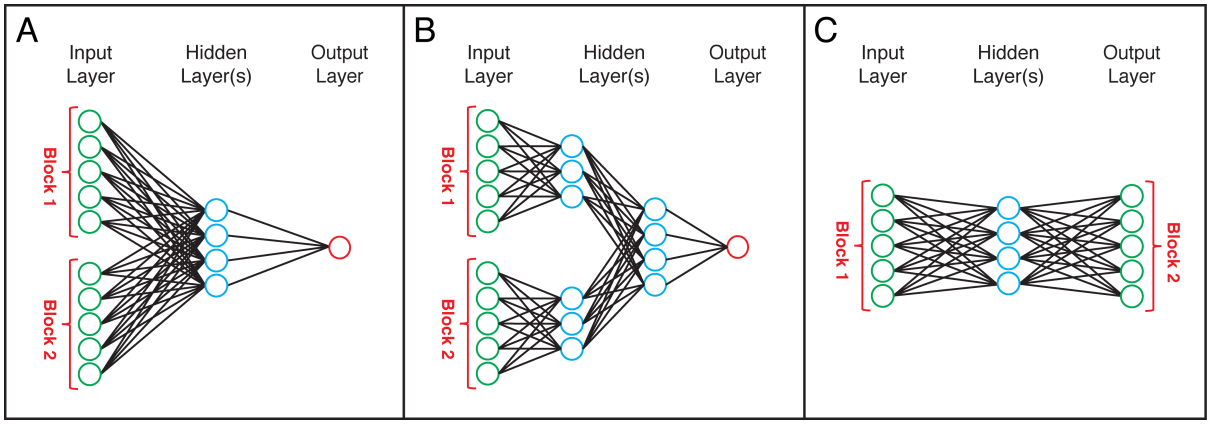


Figure 2.5: Examples of multi-block ANN architectures. **a** The concatenation of two data blocks to an output layer. **b** Multiple input layers that feed to an output layer. **c** Input layer for one data block that feeds into input layer for a second data block or vice versa.

2.7. Conclusion

As ANNs grow in popularity across systems biology, it is important to temper excessive optimism with a level of caution. Non-linear ML models may not be appropriate for every research question and the issue of interpretability remains a major challenge. In studies where classification is vital and direct interpretation less important than deep learning seems a viable option if suitable size data is available. Deep learning may play a role in improving in silco metabolite identification, but there is currently no reported evidence of this.

While relatively shallow ANNs may be useful for multi-omic data integration, it may remain impractical to use the type of deep learning that has had such an incredible impact on image processing. Although technological advancements have led to higher throughput and high-quality data at a lower cost, it is still expensive compared to the typical data used by deep learning algorithms. Even if cost of analysis is driven down to ~ $100 per sample, if a deep ANN application requires 5000 training examples then that equates to $0.5 million for data acquisition alone. The logistics of collecting and biobanking that number of samples (e.g. serum for a clinical application) could easily triple that cost. In comparison, equivalent size data from social media, video feed, or photography may cost only pennies. The successful multi-platform genomic study with 5593 samples Sharifi-Noghabi *et al.* (2019) shows the potential, but a clear breakthrough application is needed to fully justify funding, together with a massive decrease in cost of data acquisition.

Authors Contributions

All authors conceived of the idea. KMM wrote the manuscript. DIB and SNR edited the manuscript.

Conflict of interest

The authors have no disclosures of potential conflicts of interest related to the presented work.

Research Involving Human or Animal Rights

No research involving human or animal participants was performed in the construction of this manuscript.

References

Adadi, A. and Berrada, M. (2018) Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access* **6,** 52138-52160.

AlaKorpela, M., Changani, K.K., Hiltunen, Y., Bell, J.D., Fuller, B.J., Bryant, D.J., TaylorRobinson, S.D. and Davidson, B.R. (1997) Assessment of quantitative artificial neural network analysis in a metabolically dynamic ex vivo P-31 NMR pig liver study. *Magnetic Resonance in Medicine* **38,** 840-844.

Alakwaa, F.M., Chaudhary, K. and Garmire, L.X. (2018) Deep learning accurately predicts estrogen receptor status in breast cancer metabolomics data. *Journal of Proteome Research* **17,** 337-347.

Aliakbarzadeh, G., Sereshti, H. and Parastar, H. (2016) Pattern recognition analysis of chromatographic fingerprints of Crocus sativus L. secondary metabolites towards source identification and quality control. *Analytical and Bioanalytical Chemistry* **408,** 3295-3307.

Allen, F., Pon, A., Greiner, R. and Wishart, D. (2016) Computational prediction of electron ionization mass spectra to assist in GC/MS compound identification. *Analytical Chemistry* **88,** 7689-7697.

Alom, M.Z., Taha, T.M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M.S., Van Esesn, B.C., Awwal, A.A.S. and Asari, V.K. (2018) The history began from AlexNet: a comprehensive survey on deep learning approaches. *arXiv:1803.01164*.

Anthony, M.L., Rose, V.S., Nicholson, J.K. and Lindon, J.C. (1995) Classification of toxin-induced changes in 1H NMR spectra of urine using an artificial neural network. *Journal of Pharmaceutical and Biomedical Analysis* **13,** 205-11.

Asakura, T., Date, Y. and Kikuchi, J. (2018) Application of ensemble deep neural network to metabolomics studies. *Analytica Chimica Acta* **1037,** 230-236.

Azmi, J., Griffin, J.L., Antti, H., Shore, R.F., Johansson, E., Nicholson, J.K. and Holmes, E. (2002) Metabolic trajectory characterisation of xenobiotic-induced hepatotoxic lesions using statistical batch processing of NMR data. *Analyst* **127,** 271-276.

Banerjee, P., Barman, S.R., Sikdar, D., Roy, U., Mukhopadhayay, A. and Das, P. (2017) Enhanced degradation of ternary dye effluent by developed bacterial consortium with RSM optimization, ANN modeling and toxicity evaluation. *Desalination and Water Treatment* **72,** 249-265.

Barnette, D.A., Davis, M.A., Dang, N.L., Pidugu, A.S., Hughes, T., Swamidass, S.J., Boysen, G. and Miller, G.P. (2018) Lamisil (terbinafine) toxicity: Determining pathways to bioactivation through computational and experimental approaches. *Biochemical Pharmacology* **156,** 10-21.

Basheer, I.A. and Hajmeer, M. (2000) Artificial neural networks: fundamentals, computing, design, and application. *Journal of Microbiological Methods* **43,** 3-31.

Beger, R.D., Dunn, W.B., Bandukwala, A., Bethan, B., Broadhurst, D., Clish, C.B., Dasari, S., Derr, L., Evans, A., Fischer, S., Flynn, T., Hartung, T., Herrington, D., Higashi, R., Hsu, P.-C., Jones, C., Kachman, M., Karuso, H., Kruppa, G., Lippa, K., Maruvada, P., Mosley, J., Ntai, I., O'Donovan, C., Playdon, M., Raftery, D., Shaughnessy, D., Souza, A., Spaeder, T., Spalholz, B., Tayyari, F., Ubhi, B., Verma, M., Walk, T., Wilson, I., Witkin, K., Bearden, D.W. and Zanetti, K.A. (2019) Towards quality assurance and quality control in untargeted metabolomics studies. *Metabolomics* **15,** 4.

Bengio, Y., Ducharme, R., Vincent, P. and Jauvin, C. (2003) A neural probabilistic language model. *Journal of Machine Learning Research* **3,** 1137-1155.

Bica, I., Velickovic, P., Xiao, H. and Li, P. (2018). Multi-omics data integration using cross-modal neural networks. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.*, pp. 385-390.

Bostrom, N. and Yudkowsky, E. (2014) Chapter 15 - The ethics of artificial intelligence, *The Cambridge Handbook of Artificial Intelligence*, Cambridge University Press, Cambridge, United Kingdom.

Breiman, L. (2001) Statistical modeling: The two cultures. *Statistical Science* **16,** 199-231.

Broadhurst, D. (2019). Is metabolomics ready for the return of artificial neural networks? Retrieved August 25, 2019, from https://doi.org/10.6084/m9.figshare.8326529.v1

Broadhurst, D., Goodacre, R., Reinke, S.N., Kuligowski, J., Wilson, I.D., Lewis, M.R. and Dunn, W.B. (2018) Guidelines and considerations for the use of system suitability and quality control samples in mass spectrometry assays applied in untargeted clinical metabolomic studies. *Metabolomics* **14,** 72.

Broadhurst, D.I. and Kell, D.B. (2006) Statistical strategies for avoiding false discoveries in metabolomics and related experiments. *Metabolomics* **2,** 171-196.

Cambria, E. and White, B. (2014) Jumping NLP curves: A review of natural language processing research. *IEEE Computational Intelligence Magazine* **9,** 48-57.

Cavill, R., Jennen, D., Kleinjans, J. and Briede, J.J. (2016) Transcriptomic and metabolomic data integration. *Briefings in Bioinformatics* **17,** 891-901.

Chagas-Paula, D.A., Oliveira, T.B., Zhang, T., Edrada-Ebel, R. and Da Costa, F.B. (2015) Prediction of anti-inflammatory plants and discovery of their biomarkers by machine learning algorithms and metabolomic Studies. *Planta Medica* **81,** 450-458.

Chaudhary, K., Poirion, O.B., Lu, L. and Garmire, L.X. (2018) Deep learning-based multi-omics integration robustly predicts survival in liver cancer. *Clinical Cancer Research* **24,** 1248-1259.

Chen, J.X. (2016) The evolution of computing: AlphaGo. *Computing in Science & Engineering* **18,** 4.

Chollet, F. (2015). Keras. https://keras.io/

Chollet, F. (2018) Chapter 2: Before we begin: the mathematical building blocks of neural networks, *Deep learning with Python*, Manning Publications Co., New York, United States of America.

Chung, N.C., Mirza, B., Choi, H., Wang, J., Wang, D., Ping, P. and Wang, W. (2019) Unsupervised classification of multi-omics data during cardiac remodeling using deep learning. *Methods* **166,** 66-73.

Cireşan, D.C., Giusti, A., Gambardella, L.M. and Schmidhuber, J. (2012a) Deep neural networks segment neuronal membranes in electron microscopy images. *Proceedings of the 25th International Conference on Neural Information Processing Systems* **1,** 2843-2851.

Cireşan, D.C., Giusti, A., Gambardella, L.M. and Schmidhuber, J. (2013) Mitosis detection in breast cancer histology images with deep neural networks. *Medical Image Computing and Computer-Assisted Intervention 2013* **1,** 411-418.

Cireşan, D.C., Meier, U., Masci, J. and Schmidhuber, J. (2012b) Multi-column deep neural network for traffic sign classification. *Neural Networks* **32,** 333-338.

Cortina, P.R., Santiago, A.N., Sance, M.M., Peralta, I.E., Carrari, F. and Asis, R. (2018) Neuronal network analyses reveal novel associations between volatile organic compounds and sensory properties of tomato fruits. *Metabolomics* **14,** 15.

Date, Y. and Kikuchi, J. (2018) Application of a deep neural network to metabolomics studies and its performance in determining important variables. *Analytical Chemistry* **90,** 1805-1810.

Deelen, J., Kettunen, J., Fischer, K., van der Spek, A., Trompet, S., Kastenmüller, G., Boyd, A., Zierer, J., van den Akker, E.B., Ala-Korpela, M., Amin, N., Demirkan, A., Ghanbari, M., van Heemst, D., Ikram, M.A., van Klinken, J.B., Mooijaart, S.P., Peters, A., Salomaa, V., Sattar, N., Spector, T.D., Tiemeier, H., Verhoeven, A., Waldenberger, M., Würtz, P., Davey Smith, G., Metspalu, A., Perola, M., Menni, C., Geleijnse, J.M., Drenos, F., Beekman, M., Jukema, J.W., van Duijn, C.M. and Slagboom, P.E. (2019) A metabolic profile of all-cause mortality risk identified in an observational study of 44,168 individuals. *Nature Communications* **10,** 3346.

Dong, W.J., Zhao, J.P., Hu, R.S., Dong, Y.P. and Tan, L.H. (2017) Differentiation of Chinese robusta coffees according to species, using a combined electronic nose and tongue, with the aid of chemometrics. *Food Chemistry* **229,** 743-751.

Dunn, W.B., Broadhurst, D.I., Atherton, H.J., Goodacre, R. and Griffin, J.L. (2011) Systems level studies of mammalian metabolomes: the roles of mass spectrometry and nuclear magnetic resonance spectroscopy. *Chemical Society Reviews* **40,** 387-426.

Dunn, W.B., Lin, W., Broadhurst, D., Begley, P., Brown, M., Zelena, E., Vaughan, A.A., Halsall, A., Harding, N., Knowles, J.D., Francis-McIntyre, S., Tseng, A., Ellis, D.I., O'Hagan, S., Aarons, G., Benjamin, B., Chew-Graham, S., Moseley, C., Potter, P., Winder, C.L., Potts, C., Thornton, P., McWhirter, C., Zubair, M., Pan, M., Burns, A., Cruickshank, J.K., Jayson, G.C., Purandare, N., Wu, F.C.W., Finn, J.D., Haselden, J.N., Nicholls, A.W., Wilson, I.D., Goodacre, R. and Kell, D.B. (2015) Molecular phenotyping of a UK population: defining the human serum metabolome. *Metabolomics* **11,** 9-26.

Egmont-Petersen, M., de Ridder, D. and Handels, H. (2002) Image processing with neural networks—a review. *Pattern Recognition* **35,** 2279-2301.

Eraslan, G., Avsec, Ž., Gagneur, J. and Theis, F.J. (2019) Deep learning: new computational modelling techniques for genomics. *Nature Reviews Genetics* **20,** 389-403.

Erevelles, S., Fukawa, N. and Swayne, L. (2016) Big data consumer analytics and the transformation of marketing. *Journal of Business Research* **69,** 897-904.

Eriksson, L., Byrne, T., Johansson, E., Trygg, J. and Vikström, C. (2013) *Multi- and megavariate data analysis: basic principles and applications*, 3rd edn. Umetrics Academy, Malmö, Sweden.

Falcini, F., Lami, G. and Costanza, A.M. (2017) Deep learning in automotive software. *IEEE Software* **34,** 56-63.

Fatemi, M.H., Shahroudi, E.M. and Amini, Z. (2015) Development of quantitative interspecies toxicity relationship modeling of chemicals to fish. *Journal of Theoretical Biology* **380,** 16-23.

Fayolle, P., Picque, D. and Corrieu, G. (1997) Monitoring of fermentation processes producing lactic acid bacteria by mid-infrared spectroscopy. *Vibrational Spectroscopy* **14,** 247-252.

Ferrucci, D.A. (2012) Introduction to "This Is Watson". *IBM Journal of Research and Development* **56,** 235-249

Flagel, L., Brandvain, Y. and Schrider, D.R. (2018) The unreasonable effectiveness of convolutional neural networks in population genetic inference. *Molecular Biology and Evolution* **36,** 220-238.

Francescatto, M., Chierici, M., Rezvan Dezfooli, S., Zandonà, A., Jurman, G. and Furlanello, C. (2018) Multi-omics integration for neuroblastoma clinical endpoint prediction. *Biology Direct* **13,** 5.

Frisvad, J.C. (1992) Chemometrics and chemotaxonomy: A comparison of multivariate statistical methods for the evaluation of binary fungal secondary metabolite data. *Chemometrics and Intelligent Laboratory Systems* **14,** 253-269.

Fukushima, K. (1980) Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics* **36,** 193-202.

Gardner, M.W. and Dorling, S.R. (1998) Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric Environment* **32,** 2627-2636.

Geladi, P. and Kowalski, B.R. (1986) Partial least-squares regression: a tutorial. *Analytica Chimica Acta* **185,** 1-17.

Goodacre, R. (2003) Explanatory analysis of spectroscopic data using machine learning of simple, interpretable rules. *Vibrational Spectroscopy* **32,** 33-45.

Goodacre, R. and Kell, D.B. (1993) Rapid and quantitative analysis and bioprocesses using pyrolysis mass spectrometry and neural networks: application to indole production. *Analytica Chimica Acta* **279,** 17-26.

Goodacre, R. and Kell, D.B. (1996) Correction of mass spectral drift using artificial neural networks. *Analytical Chemistry* **68,** 271-280.

Goodacre, R., Kell, D.B. and Bianchi, G. (1992) Neural networks and olive oil. *Nature* **359,** 594-594.

Goodacre, R., Kell, D.B. and Bianchi, G. (1993) Rapid assessment of the adulteration of virgin olive oils by other seed oils using pyrolysis mass spectrometry and artificial neural networks. *Journal of the Science of Food and Agriculture* **63,** 297-307.

Goodacre, R., Rischert, D.J., Evans, P.M. and Kell, D.B. (1996a) Rapid authentication of animal cell lines using pyrolysis mass spectrometry and auto-associative artificial neural networks. *Cytotechnology* **21,** 231-41.

Goodacre, R., Rooney, P.J. and Kell, D.B. (1998) Discrimination between methicillin-resistant and methicillin-susceptible Staphylococcus aureus using pyrolysis mass spectrometry and artificial neural networks. *Journal of Antimicrobial Chemotherapy* **41,** 27-34.

Goodacre, R., Timmins, É.M., Jones, A., Kell, D.B., Maddock, J., Heginbothom, M.L. and Magee, J.T. (1997) On mass spectrometer instrument standardization and interlaboratory calibration transfer using neural networks. *Analytica Chimica Acta* **348,** 511-532.

Goodacre, R., Timmins, E.M., Rooney, P.J., Rowland, J.J. and Kell, D.B. (1996b) Rapid identification of Streptococcus and Enterococcus species using diffuse reflectance-absorbance Fourier transform infrared spectroscopy and artificial neural networks. *FEMS Microbiology Letters* **140,** 233-239.

Goodacre, R., Trew, S., Wrigleyjones, C., Neal, M.J., Maddock, J., Ottley, T.W., Porter, N. and Kell, D.B. (1994) Rapid screening for metabolite overproduction in fermentor broths, using pyrolysis mass-spectrometry with multivariate calibration and artificial neural networks. *Biotechnology and Bioengineering* **44,** 1205-1216.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep learning*. MIT press, Massachusetts, United States of America.

Grapov, D., Fahrmann, J., Wanichthanarak, K. and Khoomrung, S. (2018) Rise of deep learning for genomic, proteomic, and metabolomic data integration in precision medicine. *Omics-a Journal of Integrative Biology* **22,** 630-636.

Gromski, P.S., Muhamadali, H., Ellis, D.I., Xu, Y., Correa, E., Turner, M.L. and Goodacre, R. (2015) A tutorial review: Metabolomics and partial least squares-discriminant analysis--a marriage of convenience or a shotgun wedding. *Analytica Chimica Acta* **879,** 10-23.

Guo, J.R., Chen, Q.Q., Lam, C.W.K., Wang, C.Y., Wong, V.K.W., Xu, F.G., Jiang, Z.H. and Zhang, W. (2015) Application of artificial neural network to investigate the effects of 5-fluorouracil on ribonucleotides and deoxyribonucleotides in HepG2 cells. *Scientific Reports* **5,** 14.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S. and Lew, M.S. (2016) Deep learning for visual understanding: A review. *Neurocomputing* **187,** 27-48.

Hall, L.M., Hill, D.W., Bugden, K., Cawley, S., Hall, L.H., Chen, M.H. and Grant, D.F. (2018) Development of a reverse phase HPLC retention index model for nontargeted metabolomics using synthetic compounds. *Journal of Chemical Information and Modeling* **58,** 591-604.

Hall, L.M., Hill, D.W., Menikarachchi, L.C., Chen, M.H., Hall, L.H. and Grant, D.F. (2015) Optimizing artificial neural network models for metabolomics and systems biology: an example using HPLC retention index data. *Bioanalysis* **7,** 939-55.

Hamid, J.S., Hu, P., Roslin, N.M., Ling, V., Greenwood, C.M.T. and Beyene, J. (2009) Data integration in genetics and genomics: methods and challenges. *Human Genomics and Proteomics* **2009,** 869093.

Harthun, S., Matischak, K. and Friedl, P. (1998) Simultaneous prediction of human antithrombin III and main metabolites in animal cell culture processes by near-infrared spectroscopy. *Biotechnology Techniques* **12,** 393-397.

Hettinga, K.A., de Bok, F.A.M. and Lam, T. (2015) Short communication: Practical issues in implementing volatile metabolite analysis for identifying mastitis pathogens. *Journal of Dairy Science* **98,** 7906-7910.

Holmes, E., Loo, R.L., Stamler, J., Bictash, M., Yap, I.K., Chan, Q., Ebbels, T., De Iorio, M., Brown, I.J., Veselkov, K.A., Daviglus, M.L., Kesteloot, H., Ueshima, H., Zhao, L., Nicholson, J.K. and Elliott, P. (2008) Human metabolic phenotype diversity and its association with diet and blood pressure. *Nature* **453,** 396-400.

Holzinger, A., Biemann, C., Pattichis, C.S. and Kell, D.B. (2017) What do we need to build explainable AI systems for the medical domain? *arXiv:1712.09923*.

Hotelling, H. (1933) Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology* **24,** 417-441.

Huang, T., Lan, L., Fang, X.X., An, P., Min, J.X. and Wang, F.D. (2015) Promises and challenges of big data computing in health sciences. *Big Data Research* **2,** 2-11.

Huang, Z., Zhan, X., Xiang, S., Johnson, T.S., Helm, B., Yu, C.Y., Zhang, J., Salama, P., Rizkalla, M., Han, Z. and Huang, K. (2019) SALMON: Survival analysis learning with multi-omics neural networks on breast cancer. *Frontiers in Genetics* **10,** 166.

Hughes, T.B. and Swamidass, S.J. (2017) Deep learning to predict the formation of quinone species in drug metabolism. *Chemical Research in Toxicology* **30,** 642-656.

Inglese, P., McKenzie, J.S., Mroz, A., Kinross, J., Veselkov, K., Holmes, E., Takats, Z., Nicholson, J.K. and Glen, R.C. (2017) Deep learning and 3D-DESI imaging reveal the hidden metabolic heterogeneity of cancer. *Chemical science* **8,** 3500-3511.

Jolliffe, I.T. (1982) A note on the use of principal components in regression. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **31,** 300-303.

Jolliffe, I.T. (2002) *Principal Component Analysis*, 2nd edn. Springer, New York, United States of America.

Kaartinen, J., Mierisova, S., Oja, J.M.E., Usenius, J.P., Kauppinen, R.A. and Hiltunen, Y. (1998) Automated quantification of human brain metabolites by artificial neural network analysis from in vivo single-voxel H-1 NMR spectra. *Journal of Magnetic Resonance* **134,** 176-179.

Kendall, M.G. (1957) *A course in multivariate analysis*. Hafner Publishing Company, New York, United States of America.

Krizhevsky, A., Sutskever, I. and E. Hinton, G. (2012) ImageNet classification with deep convolutional neural networks. *Proceedings of the 25th International Conference on Neural Information Processing Systems* **1,** 1097-1105

Lang, N.P., Butler, M.A., Massengill, J., Lawson, M., Stotts, R.C., Hauerjensen, M. and Kadlubar, F.F. (1994) Rapid metabolic phenotypes for acetyltransferase and cytochrome P4501A2 and putative exposure to food-borne heterocyclic amines increase the risk for colorectal cancer or polyps. *Cancer Epidemiology Biomarkers & Prevention* **3,** 675-682.

Lenz, I., Lee, H. and Saxena, A. (2015) Deep learning for detecting robotic grasps. *The International Journal of Robotics Research* **34,** 705-724.

Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J. and Quillen, D. (2018) Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research* **37,** 421-436.

Lindon, J.C., Nicholson, J.K., Holmes, E., Antti, H., Bollard, M.E., Keun, H., Beckonert, O., Ebbels, T.M., Reily, M.D., Robertson, D., Stevens, G.J., Luke, P., Breau, A.P., Cantor,

G.H., Bible, R.H., Niederhauser, U., Senn, H., Schlotterbeck, G., Sidelmann, U.G., Laursen, S.M., Tymiak, A., Car, B.D., Lehman-McKeeman, L., Colet, J.M., Loukaci, A. and Thomas, C. (2003) Contemporary issues in toxicology the role of metabonomics in toxicology and its evaluation by the COMET project. *Toxicol Appl Pharmacol* **187,** 137-46.

Löfstedt, T. and Trygg, J. (2011) OnPLS—a novel multiblock method for the modelling of predictive and orthogonal variation. *Journal of Chemometrics* **25,** 441-455.

Long, N.P., Lim, D.K., Mo, C., Kim, G. and Kwon, S.W. (2017) Development and assessment of a lysophospholipid-based deep learning model to discriminate geographical origins of white rice. *Scientific Reports* **7,** 10.

Luckow, A., Cook, M., Ashcraft, N., Weill, E., Djerekarov, E. and Vorster, B. (2016) Deep learning in the automotive industry: Applications and tools. *IEEE International Conference on Big Data* **1,** 3759-3768.

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. and McClosky, D. (2014) The Stanford CoreNLP natural language processing toolkit. *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations* **1,** 55-60.

Maudsley, S., Devanarayan, V., Martin, B., Geerts, H. and Brain Hlth Modeling, I. (2018) Intelligent and effective informatic deconvolution of "Big Data" and its future impact on the quantitative nature of neurodegenerative disease therapy. *Alzheimers & Dementia* **14,** 961-975.

McCulloch, W.S. and Pitts, W. (1943) A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics* **5,** 115-133.

Moen, B.E., Nilsson, R., Nordlinder, R., Ovrebo, S., Bleie, K., Skorve, A.H. and Hollund, B.E. (1996) Assessment of exposure to polycyclic aromatic hydrocarbons in engine rooms by measurement of urinary 1-hydroxypyrene. *Occupational and Environmental Medicine* **53,** 692-696.

Moore, G.E. (1975) Progress in digital integrated electronics. *Electron Devices Meeting* **21,** 11-13.

Morin, F. and Bengio, Y. (2005) Hierarchical probabilistic neural network language model. *International Conference on Artificial Intelligence and Statistics* **5,** 246-252.

Mosconi, F., Julou, T., Desprat, N., Sinha, D.K., Allemand, J.-F., Croquette, V. and Bensimon, D. (2008) Some nonlinear challenges in biology. *Nonlinearity* **21,** 131-147.

Oh, K.-S. and Jung, K. (2004) GPU implementation of neural networks. *Pattern Recognition* **37,** 1311-1314.

Olden, J.D., Joy, M.K. and Death, R.G. (2004) An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. *Ecological Modelling* **178,** 389-397.

Ou, X.Q., Li, H., Yang, X.M., Tan, M.L., Ao, H. and Wang, J. (2015) Artificial neural network analysis of xinhui pericarpium citri reticulatae using gas chromatography - mass spectrometer - automated mass spectral deconvolution and identification system. *Tropical Journal of Pharmaceutical Research* **14,** 2071-2075.

Pecnik, K., Todorovic, V., Bosnjak, M., Cemazar, M., Kononenko, I., Sersa, G. and Plavec, J. (2018) The general explanation method with NMR spectroscopy enables the identification of metabolite profiles specific for normal and tumor cell lines. *ChemBioChem* **19,** 2066-2071.

Pérez-Enciso, M. and Tenenhaus, M. (2003) Prediction of clinical outcome with microarray data: a partial least squares discriminant analysis (PLS-DA) approach. *Human Genetics* **112,** 581-592.

Peters, W., Gang, E.S., Okazaki, H., Solingen, S., Kobayashi, Y., Karagueuzian, H.S. and Mandel, W.J. (1991) Acute effects of intravenous propafenone on the internal ventricular defibrillation threshold in the anesthetized dog. *American Heart Journal* **122,** 1355-1360.

Pinu, R.F., Beale, J.D., Paten, M.A., Kouremenos, K., Swarup, S., Schirra, J.H. and Wishart, D. (2019) Systems biology and multi-omics integration: viewpoints from the metabolomics research community. *Metabolites* **9,** 76.

Qiu, S., Yang, W.Z., Yao, C.L., Qiu, Z.D., Shi, X.J., Zhang, J.X., Hou, J.J., Wang, Q.R., Wu, W.Y. and Guo, D.A. (2016) Nontargeted metabolomic analysis and "commercial-homophyletic" comparison-induced biomarkers verification for the systematic chemical differentiation of five different parts of Panax ginseng. *Journal of Chromatography A* **1453,** 78-87.

Rawat, W. and Wang, Z. (2017) Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation* **29,** 2352-2449.

Reinke, S.N., Galindo-Prieto, B., Skotare, T., Broadhurst, D.I., Singhania, A., Horowitz, D., Djukanović, R., Hinks, T.S.C., Geladi, P., Trygg, J. and Wheelock, C.E. (2018) OnPLS-based multi-block data integration: a multivariate approach to interrogating biological interactions in asthma. *Analytical Chemistry* **90,** 13400-13408.

Risum, A.B. and Bro, R. (2019) Using deep learning to evaluate peaks in chromatographic data. *Talanta* **204,** 255-260.

Robinson, D.A. (1992) Implications of neural networks for how we think about brain function. *Behavioral and Brain Sciences* **15,** 644-655.

Rohart, F., Gautier, B., Singh, A. and Lê Cao, K.-A. (2017) mixOmics: An R package for 'omics feature selection and multiple data integration. *PLOS Computational Biology* **13,** e1005752.

Roundtable on Translating Genomic-Based Research for Health, Board on Health Sciences Policy, Health and Medicine Division and National Academies of Sciences, E., and

Medicine, (2016) F, Large Genetic Cohort Studies: A Background in Siobhan, A., Steve, O. and Sarah, H.B. (Eds), *Applying an Implementation Science Approach to Genomic Medicine: Workshop Summary*, The National Academies Press, Washington, D.C., United States of America.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C. and Fei-Fei, L. (2015) ImageNet large scale visual recognition challenge. *International Journal of Computer Vision* **115,** 211-252.

Russell, S., Hauert, S., Altman, R. and Veloso, M. (2015) Ethics of artificial intelligence. *Nature* **521,** 415-416.

Samaraweera, M.A., Hall, L.M., Hill, D.W. and Grant, D.F. (2018) Evaluation of an artificial neural network retention index model for chemical structure identification in nontargeted metabolomics. *Analytical Chemistry* **90,** 12752-12760.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K. and Muller, K.-R. (Eds) (2019) *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, Springer International Publishing, Basel, Switzerland.

Samek, W., Wiegand, T. and Müller, K.-R. (2017) Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv:1708.08296*.

Saunders, G., Baudis, M., Becker, R., Beltran, S., Béroud, C., Birney, E., Brooksbank, C., Brunak, S., Van den Bulcke, M., Drysdale, R., Capella-Gutierrez, S., Flicek, P., Florindi, F., Goodhand, P., Gut, I., Heringa, J., Holub, P., Hooyberghs, J., Juty, N., Keane, T.M., Korbel, J.O., Lappalainen, I., Leskosek, B., Matthijs, G., Mayrhofer, M.T., Metspalu, A., Navarro, A., Newhouse, S., Nyrönen, T., Page, A., Persson, B., Palotie, A., Parkinson, H., Rambla, J., Salgado, D., Steinfelder, E., Swertz, M.A., Valencia, A., Varma, S., Blomberg, N. and Scollen, S. (2019) Leveraging European infrastructures to access 1 million human genomes by 2022. *Nature Reviews Genetics* **20,** 693-701.

Schalkoff, R.J. (1997) *Artificial Neural Networks*, International ed edn. McGraw-Hill, London, United Kingdom.

Schmidhuber, J. (2012) Multi-column deep neural networks for image classification. *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)***,** 3642-3649.

Seber, G.A.F. (2004) *Multivariate Observations*, 2nd edn. Wiley, New Jersey, United States of America.

Sharifi-Noghabi, H., Zolotareva, O., Collins, C.C. and Ester, M. (2019) MOLI: Multi-Omics Late Integration with deep neural networks for drug response prediction. *Bioinformatics* **35,** i501–i509.

Shen, D., Wu, G. and Suk, H.-I. (2017) Deep learning in medical image analysis. *Annual Review of Biomedical Engineering* **19,** 221-248.

Simard, P.Y., Steinkraus, D. and Platt, J.C. (2003) Best practices for convolutional neural networks applied to visual document analysis. *International Conference on Document Analysis and Recognition* **3,** 1-6.

Sjogren, M., Ehrenberg, L. and Rannug, U. (1996) Relevance of different biological assays in assessing initiating and promoting properties of polycyclic aromatic hydrocarbons with respect to carcinogenic potency. *Mutation Research-Fundamental and Molecular Mechanisms of Mutagenesis* **358,** 97-112.

Tautenhahn, R., Patti, G.J., Rinehart, D. and Siuzdak, G. (2012) XCMS Online: a web-based platform to process untargeted metabolomic data. *Analytical Chemistry* **84,** 5035-5039.

Trainor, P.J., DeFilippis, A.P. and Rai, S.N. (2017) Evaluation of classifier performance for multiclass phenotype discrimination in untargeted metabolomics. *Metabolites* **7,** 20.

Trivedi, D.K., Hollywood, K.A. and Goodacre, R. (2017) Metabolomics for the masses: The future of metabolomics in a personalized world. *New Horizons in Translational Medicine* **3,** 294-305.

Usenius, J.P., Tuohimetsa, S., Vainio, P., AlaKorpela, M., Hiltunen, Y. and Kauppinen, R.A. (1996) Automated classification of human brain tumours by neural network analysis using in vivo H-1 magnetic resonance spectroscopic metabolite phenotypes. *Neuroreport* **7,** 1597-1600.

Wang, F., Wang, B., Wang, L., Xiong, Z.Y., Gao, W., Li, P. and Li, H.J. (2017) Discovery of discriminatory quality control markers for Chinese herbal medicines and related processed products by combination of chromatographic analysis and chemometrics methods: Radix Scutellariae as a case study. *Journal of Pharmaceutical and Biomedical Analysis* **138,** 70-79.

Wang, F.-Y., Zhang, J.J., Zheng, X., Wang, X., Yuan, Y., Dai, X., Zhang, J. and Yang, L. (2016) Where does AlphaGo go: From church-turing thesis to AlphaGo thesis and beyond. *IEEE/CAA Journal of Automatica Sinica* **3,** 113-120.

Wold, H. (1975) Path models with latent variables: The NIPALS approach, *Quantitative sociology*, Elsevier. 307-357.

Wold, S., Esbensen, K. and Geladi, P. (1987) Principal component analysis. *Chemometrics and Intelligent Laboratory Systems* **2,** 37-52.

Wold, S., Johansson, E. and Cocchi, M. (1993) PLS: Partial Least Squares Projections to Latent Structures, *3D QSAR in Drug Design: Theory, Methods and Applications.*, Kluwer/Escom, Dordrecht, The Netherlands.

Woldegebriel, M. and Derks, E. (2017) Artificial neural network for probabilistic feature recognition in liquid chromatography coupled to high-resolution mass spectrometry. *Analytical Chemistry* **89,** 1212-1221.

Wolff, M.S., Toniolo, P.G., Lee, E.W., Rivera, M. and Dubin, N. (1993) Blood levels of organochlorine residues and risk of breast cancer. *Journal of the National Cancer Institute* **85,** 648-652.

Worley, B. and Powers, R. (2013) Multivariate analysis in metabolomics. *Current Metabolomics* **1,** 92-107.

Wu, T., Liu, S., Zhang, J. and Xiang, Y. (2017) Twitter spam detection based on deep learning. *Proceedings of the Australasian Computer Science Week Multiconference* **1,** 3.

Young, T., Hazarika, D., Poria, S. and Cambria, E. (2018) Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine* **13,** 55-75.

Yue, T. and Wang, H. (2018) Deep learning for genomics: A Concise Overview. *arXiv:1802.00810* **1,** 1-40.

Zhang, Q.-s. and Zhu, S.-c. (2018) Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering* **19,** 27-39.

Zhang, X.X., Li, Y.Z., Liang, Y., Sun, P.T., Wu, X., Song, J.H., Sun, X.Y., Hong, M., Gao, P. and Deng, D.F. (2017) Distinguishing intracerebral hemorrhage from acute cerebral infarction through metabolomics. *Revista de Investigación Clínica - Clinical and Translational Investigation* **69,** 319-328.

Zhang, Z., Beck, M.W., Winkler, D.A., Huang, B., Sibanda, W., Goyal, H. and written on behalf of, A.M.E.B.-D.C.T.C.G. (2018) Opening the black box of neural networks: methods for interpreting neural network models in clinical applications. *Annals of translational medicine* **6,** 216-216.

Zhao, X.Y., Qin, W.J. and Qian, X.H. (2018) Application of deep learning in biological mass spectrometry and proteomics. *Progress in Biochemistry and Biophysics* **45,** 1214-1223.

Zheng, G., Zhang, F., Zheng, Z., Xiang, Y., Yuan, N.J., Xie, X. and Li, Z. (2018) DRN: A deep reinforcement learning framework for news recommendation. *Proceedings of the 2018 World Wide Web Conference on World Wide Web* **1,** 167-176.

Zou, J., Huss, M., Abid, A., Mohammadi, P., Torkamani, A. and Telenti, A. (2019) A primer on deep learning in genomics. *Nature Genetics* **51,** 12-18.

Zurada, J.M. (1992) *Introduction to artificial neural systems*. West Publishing Company Minnesota, United States of America.

Chapter Three: Toward Collaborative Open Data Science in Metabolomics using Jupyter Notebooks and Cloud Computing

Authors

Kevin M Mendez[1,†], Leighton Pritchard[2,†], Stacey N Reinke[1,*], David I Broadhurst[1,*]

[1]Centre for Metabolomics & Computational Biology, School of Science, Edith Cowan University, Joondalup, 6027 Australia

[2]Information and Computational Sciences, James Hutton Institute, Invergowrie, Dundee DD2 5DA, Scotland

† These authors contributed equally to this article.

*Corresponding authors:

email: d.broadhurst@ecu.edu.au, stacey.n.reinke@ecu.edu.au

phone: +61 (0)8-6304-2705

This is a post-peer-review, pre-copyedit version of an article published in *Metabolomics*. The final authenticated version is available online at: https://doi.org/10.1007/s11306-019-1588-0.

Abstract

Background: A lack of transparency and reporting standards in the scientific community has led to increasing and widespread concerns relating to reproduction and integrity of results. As an omics science, which generates vast amounts of data and relies heavily on data science for deriving biological meaning, metabolomics is highly vulnerable to irreproducibility. The metabolomics community has made substantial efforts to align with FAIR data standards by promoting open data formats, data repositories, online spectral libraries, and metabolite databases. Open data analysis platforms also exist; however, they tend to be inflexible and rely on the user to adequately report their methods and results. To enable FAIR data science in metabolomics, methods and results need to be transparently disseminated in a manner that is rapid, reusable, and fully integrated with the published work. To ensure broad use within the community such a framework also needs to be inclusive and intuitive for both computational novices and experts alike.

Aim of Review: To encourage metabolomics researchers from all backgrounds to take control of their own data science, mould it to their personal requirements, and enthusiastically share resources through open science.

Key Scientific Concepts of Review: This tutorial introduces the concept of interactive web-based computational laboratory notebooks. The reader is guided through a set of experiential tutorials specifically targeted at metabolomics researchers, based around the Jupyter Notebook web application, GitHub data repository, and Binder cloud computing platform.

Key Words: Artificial Neural Network, Machine Learning, Deep Learning, Partial Least Squares, Metabolomics

3.1. Introduction

Historically, journal articles have been the primary medium for sharing new scientific research. The intent of article content, and the corresponding review process, is to ensure adequate evidence of reproducibility; however, a recent report highlights increasing and widespread concerns relating to reproduction and integrity of results, with 52% of responding scientists agreeing there is a significant 'crisis' of reproducibility (Baker, 2016). We and many others in the metabolomics community hold the view that a lack of transparency and incomplete reporting has led to significant misinterpretation of data and a lack of trust in reported results (Broadhurst and Kell, 2006; Considine *et al.*, 2017; Goodacre *et al.*, 2007; Spicer *et al.*, 2017; Xia *et al.*, 2013). A mechanism that may address these concerns is for the scientific community to take advantage of new online publishing media and associated data services, encouraging open science that recognises and aligns with the FAIR (Findable, Accessible, Interoperable, and Reusable) data principles (Wilkinson *et al.*, 2016).

This concern in metabolomics and other post-genomic platforms is a consequence of their success. The unprecedented rate at which new mathematical algorithms and computational tools are developed and adopted means that published findings are increasingly the sole result of computationally intensive data processing (Teschendorff, 2019). Advances in measurement technologies continue to generate ever increasing volumes of high-throughput data, which in turn require multidisciplinary expertise as ever more complex and elaborate statistical methods are used to infer generalisable biological associations (Pinu *et al.*, 2019) and sophisticated

visualization tools are used to make large datasets more understandable (Gehlenborg *et al.*, 2010; Holten, 2006). Indeed, recent advances in machine learning algorithms combined with improved visualisation strategies now allow researchers to integrate and interrogate multi-omic data sets of a size that was unmanageable 5 years ago (for example, Lee *et al.*, 2019; Reinke *et al.*, 2018; Rohart *et al.*, 2017). For science to continue to move forward under this deluge of data while avoiding technical debt and maintaining reproducibility, the processes surrounding research data management, storage, analysis, and presentation need to be agile, transparent, reusable, and recoverably attached to published work. A further challenge is that, to gain broad adoption and be used by busy practising researchers, frameworks conforming to these requirements must also be intuitive and accessible to users who may have limited computational expertise.

Over the last several years, the metabolomics community has made bold strides towards adopting FAIR standards for data including: development of vendor-independent raw data formats (such as mzXML) (Pedrioli *et al.*, 2004); open access data repositories such as MetaboLights (Haug *et al.*, 2012), and Metabolomics Workbench (Sud *et al.*, 2016); open access online spectral reference libraries such as METLIN (Smith *et al.*, 2005), mzCloud (https://www.mzcloud.org/), and MassBank (Horai *et al.*, 2010); and online databases for metabolite identification and biochemical association such as HMDB (Wishart *et al.*, 2018). These resources and others like them are fundamental to the future integrity of metabolomics as a science. It is well-recognised that open, interoperable datasets are essential for progress, and the computational tools and methods that convert, step-by-step, metabolite data to biochemical meaning also need to be FAIR (Wilkinson *et al.*, 2016).

Numerous groups within the metabolomics community actively work to standardise computational workflows and provide online tools for statistical analysis (some recent advances are discussed later in this paper). However, a common characteristic of many computational frameworks encountered by researchers is a tendency to be prescriptive, in that they provide a restricted set of well-curated "plug and play" computational stepping stones that enable only limited choices within the workflow framework. These constraints limit the ability of a user to fully exploit the provided methodologies, or to explore and develop new analytical approaches. Presentation of analysis steps as pluggable "black box" approaches is convenient but diminishes opportunities for education and understanding of the analysis methods being used. To fully embrace the concept of 'open data science' the metabolomics community needs an open and easily accessible computational environment for rapid collaboration and experimentation.

The subject of this tutorial review is a practical open-science solution to this problem that balances ease-of-use and flexibility, specifically targeted to novice metabolomic data scientists. This solution takes the form of 'computational lab books', such as Jupyter Notebooks (Kluyver *et al.*, 2016), that have a diverse range of overlapping potential applications in the post-genomic research community (Fig. 3.1). Firstly, they enable open collaboration by providing a central platform for researchers to cooperatively develop methodology and perform data analysis. Secondly, they provide a means for transparent dissemination of a finished study or product. In a formal context computational lab books can comprise supplemental material extending the reach of a publication that enables readers to rapidly recreate data analyses and figures for themselves. In an informal context, they can provide a polished "showcase" that allows users to interact with and understand the functionality of underlying algorithms. Finally, the inherent promotion of direct user interaction enables experiential learning opportunities, where the user

develops their understanding and skills through active experimentation, reflective observation, and abstract conceptualisation (Kolb, 1984).

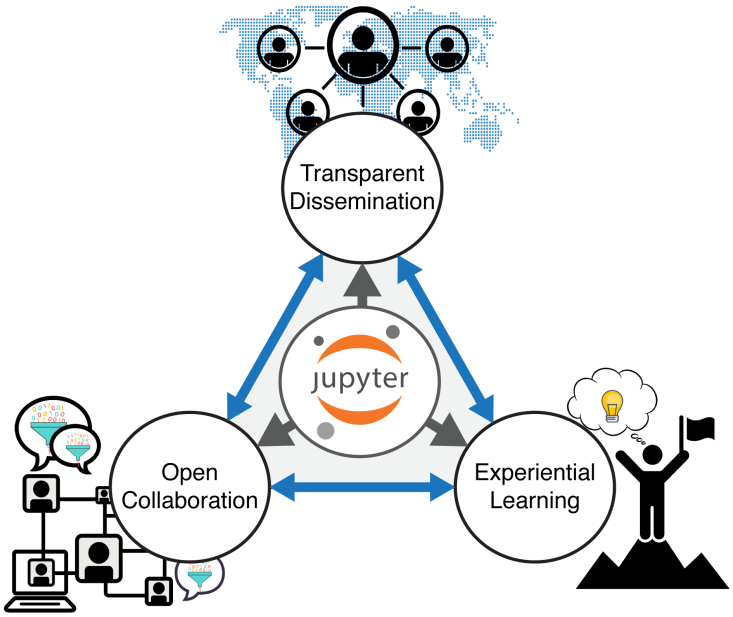


Figure 3.1: Applications for Jupyter Notebooks in the postgenomic community. Open virtual notebooks have three main, non-mutually exclusive, applications. First, they provide an efficient means for transparent dissemination of methods and results, thereby enabling alignment with FAIR data principles. Second, they provide a central and interactive platform that facilitates open collaboration to develop methodology and perform data analysis. Finally, their interactive and easily deployable framework can drive experiential learning opportunities for computational novices to develop their own skills and better understand metabolomics data analysis.

In this review, we provide a brief overview of current data science frameworks relevant to the metabolomics community, corresponding barriers to achieving open science, and finally a practical solution in the form of the computational lab notebook, where code, prose and figures are combined into an interactive notebook that can be published online and accessed in a modern web browser through cloud computing. We present a set of experiential learning tutorials introducing the Jupyter Notebook framework, specifically tailored to the needs of a metabolomics researcher. The tutorials are designed in a hierarchy of complexity following

Bloom's taxonomy of educational learning objectives (Anderson *et al.*, 2001). Tutorial one introduces the basic concepts of Jupyter Notebooks. Tutorial two encourages interactive learning using an existing metabolomics data science Jupyter notebook. Tutorial three establishes the framework in which the user can create a Jupyter notebook on a local computer. Tutorial four teaches the user how to create a simple notebook for their own data. Tutorial five explains how to publish and share a new Jupyter notebook in the cloud. The overarching aim of this document is to encourage metabolomics researchers from all backgrounds, possibly with little or no computational expertise, to seize the opportunity to take control of their own data science, mould it to their personal requirements, and enthusiastically share resources through open science.

3.2. Background

A glossary of terms has been provided in Table 3.1 to help clarify technical terms used in this tutorial.

3.2.1. Software Tools and Barriers to Open Science

Many statistical and data science software tools are available for use in metabolomics data analysis and visualisation. They can be classified as commercial (requiring a paid licence) or "free" (as in zero-cost) and, in either case, may be open-source (the underlying computer code is available for inspection) or proprietary (closed-source, code unavailable for inspection). The primary mode of interaction with the user may be via scripting, a command line (CLI), or a graphical (GUI) user interface.

Table 3.1. Glossary of terms.

Paper Section	Term	Definition
1	Data repository	A platform (such as Metabolights or Metabolomics Workbench) used to store metadata and experimental data.
2.1	Command Line Interface (CLI)	A user interface that is used to execute operating system functions using text.
2.1	Graphical User Interface (GUI)	A user interface that is used to execute operating system functions using graphical icons or other visual indicators.
2.1	Integrated Development Environment (IDE)	A software application that provides an interface to write and test code (such as RStudio, PyCharm and Visual Studio Code). It typically includes basic tools such as a code editor, compiler, and a debugger.
2.1	Containers	Self-contained units of software that packages code, dependencies, system tools and system libraries. The purpose is to be reliably transferred between, and deployed on, various operating systems and infrastructures.
2.1	JavaScript Object Notation (JSON) format	A lightweight data-interchange format commonly used for communication between a browser and server. Internally, Jupyter Notebooks are JSON files with the .ipynb extension.
2.1	Packages	Units of shareable code that can be imported and used to provide additional functionality (such as matplotlib and scikit-learn).
2.1	Application Programming Interface (API)	A set of defined functions and protocols for interacting with the software or package.
2.1	Kernel	The "computational engine" that runs and introspects the code contained in a notebook document. Jupyter supports a kernel for Python, as well as kernels for many other languages (such as R, Julia, Kotlin, etc.).
2.2	Version Control	A documented history of changes made to a file, enabling step-by-step reproduction and reconstruction of its development
2.2	Code repository	A hosted archive (such as those at GitHub and BitBucket) of source code and supporting files.
3	Virtual Environment	An isolated environment that contains a specific version of Python and dependencies.
3.1.1	Distribution (Software)	A collection of software bundled together.
3.1.1	Markdown	A lightweight markup language used to add and format plain text. It is used in Jupyter Notebooks within "Markdown" cells.
3.1.3	Configuration file	A file used to set the initial settings and parameters for computer applications. It is used in Binder to build the virtual environment with specific dependencies.
3.2.1	Text cell (Markdown cell)	A cell in the Jupyter Notebook used to write text (using the Markdown language).
3.2.1	Code cell	A cell in the Jupyter Notebook used to run code (such as Python code).
3.2.3	Sandbox (Software development)	A software environment typically used to run or test experimental code in isolation from the rest of the system.
3.2.5	Dependencies	The packages (and versions) that are required to be installed to use the software. For Python, these are the packages that need to be imported at the start of the file.
3.2.5	Channels (Specific to Anaconda)	The location where packages that are installed using conda are stored (such as conda-forge and bioconda).
3.2.5	README	A file (commonly markdown or text) used to communicate information to visitors about the repository (such as purpose, usage, and contributors).
3.2.5	Root directory	The directory (or folder) that is the highest level in a hierarchy.

Command-line or script-based proprietary software packages such as MATLAB, SAS, and Stata overcome some of the limitations imposed by graphical interfaces and closed-source code by allowing third party code to be embedded, and implementation of alternative algorithms and arbitrary workflows by the researcher. In the case of MATLAB the source code of some or all of the proprietary tools is readable, which improves transparency of methods, and it is possible for the programmer to develop open custom graphical interfaces. However, even then open-source commercial packages can carry a significant financial cost limiting the ability of researchers, especially those in developing nations or on smaller budgets, to replicate results, adapt methods, or collaborate to develop better workflows. We consider that open-source "free" tools and applications will form the future basis of shareable research, as they enable the greatest possible degree of transparency and reproducibility.

Open-source GUI workflows providing simplified or user-friendly access to underlying programs and analytical tools have been developed to improve usability for scientists who have not yet acquired the programming skills necessary to write their own pipelines and workflows. Within the metabolomics community popular applications include: MetaboAnalyst (Xia and Wishart, 2011), Galaxy-M (Davidson *et al.*, 2016), and Workflow4Metabolomics (Giacomoni *et al.*, 2015). Galaxy workflows provide a unified data visualisation and analysis environment that allows seamless (to the user) integration of multiple open-source software packages, and tools written in multiple programming languages (Afgan *et al.*, 2018). These tools allow rapid construction, implementation, and sharing of standardised workflows, including integration with remote and local databases, without the need for programming skills. This provides a mechanism to ensure methodological consistency and precise reporting standards. Resources such as Galaxy simplify the user experience and enable flexible use of a wide range of open source tools.

Despite the many strengths of open-source GUI workflows such as Galaxy, they do not always provide users with a free choice of available data analysis methods. For example, unless the user has administrative rights on the server, the browser interface of Galaxy does not permit direct access to software package management. This restricts extension, modification, and development of workflows by the user. Although an arbitrary set of tools can in principle be "wrapped" by a researcher for use with Galaxy, there may be in practice only limited support for requests to implement a tool, especially when working on public servers. It is possible to implement arbitrary tools and processes in a locally-managed Galaxy instance with administrative control of the workflow service, but this requires investment of time, technical expertise, and local computational capacity, as well as carrying implications for long-term systems support and maintenance.

Even with a free choice of tools and algorithms, workflows implemented in GUI-based tools like Galaxy are "linear" in the sense that the browser interface imposes a process in which data passes through a sequential chain of operations. These interfaces are not well-suited to representing complex workflow logic, such as branches and loops that explore alternative approaches or parameter choices as part of the same analysis. This can inadvertently encourage a "black box" one-size-fits-all approach to analysis that may be of concern when the dataset is non-standard or, for example, when a statistical analysis requires customisation due to assumptions made by the model regarding the distribution of the input data. Incurious application of standardised GUI workflows with limited opportunity for experimentation can lead to inappropriate analytical strategies and unintentional misreporting of results. The linearity constraint is recognised by the Galaxy developers, who provide a programmatic Application Programming Interface (API) enabling automation of complex workflow logic, but this requires programming ability to use.

Another limitation of GUI workflow-based applications can be a lack of contextual annotation. With most interfaces the user must document separately why computational methods and parameter settings were chosen in a specific workflow. It is not typically possible through GUI workflow interfaces to embed the experimental context, explanation of methods, code, and figures into a single live (interactive) document. The formal reporting may then be reduced to a terse listing of steps and parameter values for generating data, tables and figures, rather than a more readable "literate programming" account of the analysis. This retrospective approach is sufficient and appropriate for standardised, repeated workflows that vary little from experiment to experiment, such as a mass spectrometry deconvolution workflow that converts a set of raw instrument files into an annotated table (e.g. XCMS → CAMERA → MetFrag). However, when a metabolomics scientist moves on to statistical analysis, multivariate machine learning, and data visualisation to extract and present a biologically-informative interpretation of the data, it is desirable to have an integrated, flexible data analysis environment that includes detailed annotation of analysis choices.

The most flexible data science solution is to conduct analyses in one or more high-level open-source programming languages such as C, Fortran, Java, Julia, Perl, Python, Octave, R, or Scala, that also support sophisticated statistical tools. Python and R have become especially popular languages in data science due to the availability of comprehensive, robust, and well-documented code libraries (modules/packages). Many statistical and machine learning packages are available for these languages (including bindings to Galaxy, which overcomes some of the GUI-based limitations of that platform), with strong data science community support (Lantz, 2013; Müller and Guido, 2017). However, these general-purpose languages may present novice (or non) data scientists with a forbiddingly steep learning curve, especially in comparison with GUI tools. To be most effective in these languages a researcher requires a

basic understanding of computer programming to use the available code libraries in their specific field. There is an initial learning curve, but knowledge of a programming language is more generally useful and broadly applicable than familiarity with a specific software tool's interface and can impact positively on many areas of research. Programming is increasingly recognised as a foundational skill for research and promoted at all levels from primary to postgraduate education (Passey, 2017). The broad impact of this skillset throughout academic research, including arts and humanities, is recognised in the growing influence of training foundations such as The Carpentries (https://carpentries.org/) that aim to "[teach] researchers the computing skills they need to get more done in less time and with less pain."

Several freely-available software tools bridge the gap between GUI interfaces and high-level languages by providing a user interface for researchers to develop their own code. For Python and R, integrated development environments (IDEs) such as PyCharm (Python), RStudio (R), and more general multi-language IDEs (e.g. Visual Studio Code, Komodo and Eclipse), provide additional tools for automating, testing and visualizing the process of writing scripts, programs and analysis workflows. These IDEs can simplify the learning and programming experience but are primarily designed for larger program and application development, rather than composing and sharing data analysis workflows. However, IDEs in general are extremely useful even to the novice programmer, and some prominent examples are specifically targeted towards data analysis, such as RStudio and JupyterLab.

Recently, several independent strands of general-purpose data science software development have been woven into practical solutions to the various limitations of the above frameworks. Firstly, RStudio established itself as the 'go to' data science IDE for R programming and was extended to allow integration of R code, narrative text, and figures into a single notebook

interface using “RMarkdown” (Baumer *et al.*, 2014). The software companies Enthought Inc. and Anaconda (formerly Continuum Analytics) independently developed distributions of the Python programming language to include core scientific computing packages. Anaconda later extended their distribution to include R. In 2015, the non-profit Project Jupyter was established (Kluyver *et al.*, 2016) to “develop open-source software, open-standards, and services for interactive computing across dozens of programming languages” (Project Jupyter, 2019). Their main product is Jupyter Notebook, a browser-based interactive data science notebook environment. Jupyter Notebook allows seamless integration of code, narrative text, and figures into a single live executable and editable document, recorded in the open-standard and language-independent JavaScript Object Notation (JSON) format. Notebooks may be written in a single programming language, or a combination of multiple languages. Jupyter Notebooks can use kernels for new or more specialised languages (such as Kotlin, GAP, Haskell, etc.), which gives them an advantage of being agnostic to programming language. Finally, integration of Jupyter Notebooks with the Docker (www.docker.com) virtualization platform enables operating system level working environments to be packaged into virtual “containers”, which allows collections of notebooks and the supporting third-party tools and software to be deployed as public, self-contained, reproducible interactive services using cloud computing.

3.2.2. Collaboration through Cloud Computing

Open and dynamic collaboration on projects is critical to effective working but remains a significant challenge for researchers. There is a real and present need for efficient sharing and management of files that allows easy access, use, and version control (a documented history of the changes made to a file, enabling step-by-step reproduction and reconstruction of its development) for all collaborators. Widely-used collaboration mechanisms such as sharing

code via email or online blogs are cumbersome, frequently leading to conflicts between the work of different researchers as they work on the same files at the same time in different locations. Cloud services including Box, Google Drive, and Dropbox have become essential tools for scientists by providing shared online data and document storage. Tools such as Microsoft Office Online and Google Suite provide real-time collaboration tools enabling true simultaneous editing of a single document by multiple authors, and services like Dropbox are able to track edits and prompt users to keep local copies of files up to date. Both approaches allow users to step back through document history as a rudimentary form of version control. They reduce practical barriers to collaborative working and reduce frustration and conflicts resulting from two or more people editing different copies of the same file at the same time. Collaborative working on metabolomic data analysis workflows would benefit from adoption of similar approaches.

The source code hosting facilities Bitbucket, GitHub and SourceForge are currently the dominant platforms for sharing and collaborating on (particularly open-source) software. GitHub has become the largest source code hosting facility in the world, with over 36 million users and 100 million repositories (GitHub, 2019). These facilities offer many benefits including: free public (and private) source code repositories; enforced best practice through version control; and additional administrative and project management services that foster collaboration, including project webpages and wikis, issue tracking, code reviews, and task management. This makes GitHub and similar services a practical option for development, publication and distribution of Jupyter Notebooks, together with their associated source code and test data.

Services such as GitHub and BitBucket allow collaborators to view and edit static code and view static notebooks, but code cannot be executed directly on their servers. To run Jupyter Notebooks and associated source code, the user must either download and run a local copy of the files, or upload and run the notebook "in the cloud" using a cloud infrastructure provider such as Amazon Web Services, Google Colab, Openstack, or Microsoft Azure. The process of enabling the practical use of this shared resource can therefore require a level of computational expertise that may be a deterrent to casual users and restrict uptake by non-expert data-curious scientists.

The PhenoMeNal portal (http://phenomenal-h2020.eu) is an elegant solution to this problem for the metabolomics community. PhenoMeNal (Peters *et al.*, 2019) is an easy-to-use, cloud-based metabolomics research environment led by EMBL's European Bioinformatics Institute. The PhenoMeNal App Library includes over 50 widely used metabolomics data analysis tools that can be accessed either through Jupyter or Galaxy and deployed using a cloud infrastructure provider. This curated software library allows the community to maintain consistency across workflows but, in common with other GUI tools and centrally-managed workflow approaches, it can be restrictive.

A comparable but completely general public service is provided by the Binder team at mybinder.org (Project Jupyter *et al.*, 2018). Binder is an open-source web service that allows users to share notebooks by creating a temporary cloud-based copy of the GitHub repository that contains them. This enables reproducible sharing of interactive and editable Jupyter or Rstudio notebooks as a virtual machine running in the cloud. The user can start and access a new virtual machine running live notebooks by following a single web link. In use, the notebooks appear to the user as if they were any other Jupyter notebook running on their own

computer, with all the necessary dependencies, supplementary code and data pre-installed. Using the Binder framework gives researchers the power to reproduce and thoroughly test published results, or apply the analyses to their own data by running the source code interactively in their browser. In this tutorial review we take the reader through a process of using, writing, and deploying Jupyter Notebooks on Binder to help them take control of their own data science, and share their work through open science approaches.

3.3. Experiential Learning Tutorials

The remainder of this review provides readers with an experiential learning opportunity (Kolb 1984) using an example interactive metabolomics data analysis workflow deployed using a combination of Python, Jupyter Notebooks, and Binder. We assume that the initial stage of data-processing for the computational workflow (converting raw instrument files into an annotated data table) has already been completed, and that a deconvolved, but not necessarily annotated, data table has been created and checked for errors. These assumptions are made to make the learning objectives presented manageable, not as a directive for obfuscating the complete metabolomics workflow. It is possible, and encouraged, to include all data processing steps in interactive notebooks. The tutorial takes the reader through the process of using interactive notebooks to produce a shareable, reproducible data analysis workflow that connects the study design to reported biological conclusions in an interactive document, using data from two previously published metabolomics studies. This workflow includes a discrete set of interactive and interlinked procedures: data cleaning, univariate statistics, multivariate machine learning, feature selection, and data visualisation (Fig. 3.2).

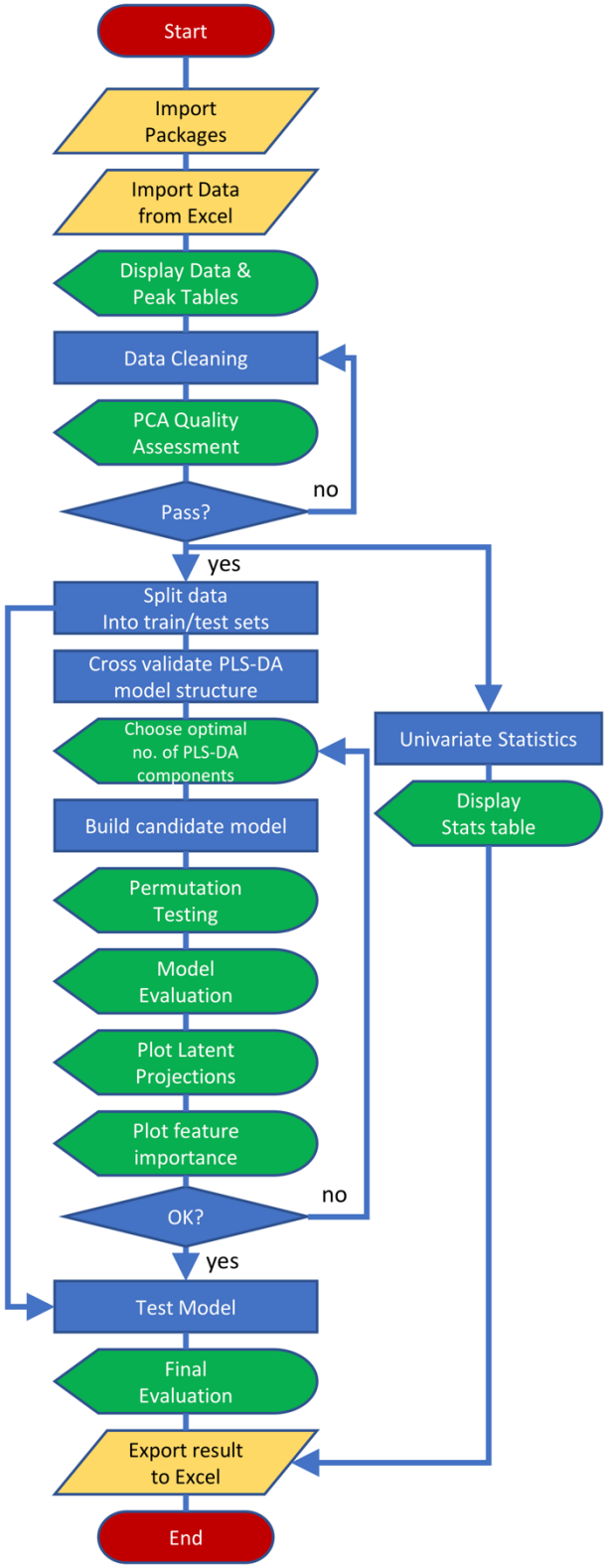


Figure 3.2: Metabolomics data analysis workflow. The workflow implemented in Tutorials 1 and 2 represents a typical metabolomics data science workflow for a binary classification outcome. The following steps are included: data import, data cleaning based on pooled QC relative standard deviation, PCA to visually inspect data reproducibility, univariate statistics, multivariate machine learning (PLS-DA including cross validation, feature selection, and permutation testing). The flow diagram is coloured by primary operation type (yellow=data import/export; green=data visualisation; blue=data processing).

The following five tutorials have been pedagogically designed to lead the reader through increasing levels of cognitive complexity, according to Bloom's revised taxonomy (Anderson *et al.*, 2001):

1. Launch and walk through a published Jupyter notebook using Binder in the cloud to *duplicate* a set of results.
2. Interact with and edit the content of a published Jupyter notebook using Binder in the cloud to *understand* workflow methods.
3. Install Python and use published Jupyter Notebooks on the researcher's computer to *apply* and *experiment* with workflow methods locally.
4. *Create* a metabolomics Jupyter notebook on a local computer.
5. Deploy the Jupyter notebook from Tutorial 4 on Binder in the cloud *via* GitHub.

3.3.1. Overview of Jupyter/GitHub/Binders

Before beginning the tutorial, we review some fundamental concepts behind Jupyter Notebooks, GitHub, and Binder, as understanding these can aid successful independent execution of this open-science approach (Fig. 3.3). All code embedded in each of the example notebooks is written in the Python programming language and is based upon extensions of popular open source packages with high levels of community uptake and support. These include: Numpy for matrix-based calculations (van der Walt *et al.*, 2011); Pandas for high level data table manipulation (McKinney, 2017); Scikit-learn for machine learning (Pedregosa *et al.*, 2011); and Matplotlib (Hunter, 2007), Bokeh (Bokeh Development Team, 2018), Seaborn (Waskom *et al.*, 2018), and BeakerX (Beaker X Development Team, 2018) for data visualisation. Additionally, we deploy a simple package called 'cimcb-lite', developed by the

authors for this publication, that integrates the functionality of the above packages into a set of basic methods specific to metabolomics. A tutorial on the Python programming language itself is beyond the scope of this publication, but we hope that the code presented is sufficiently well-documented in each notebook to be understood. Many excellent publications can be consulted for an in-depth introduction to using Python for data science (Jones, 2013; Ramalho, 2015; The Carpentries, 2019; VanderPlas, 2016).

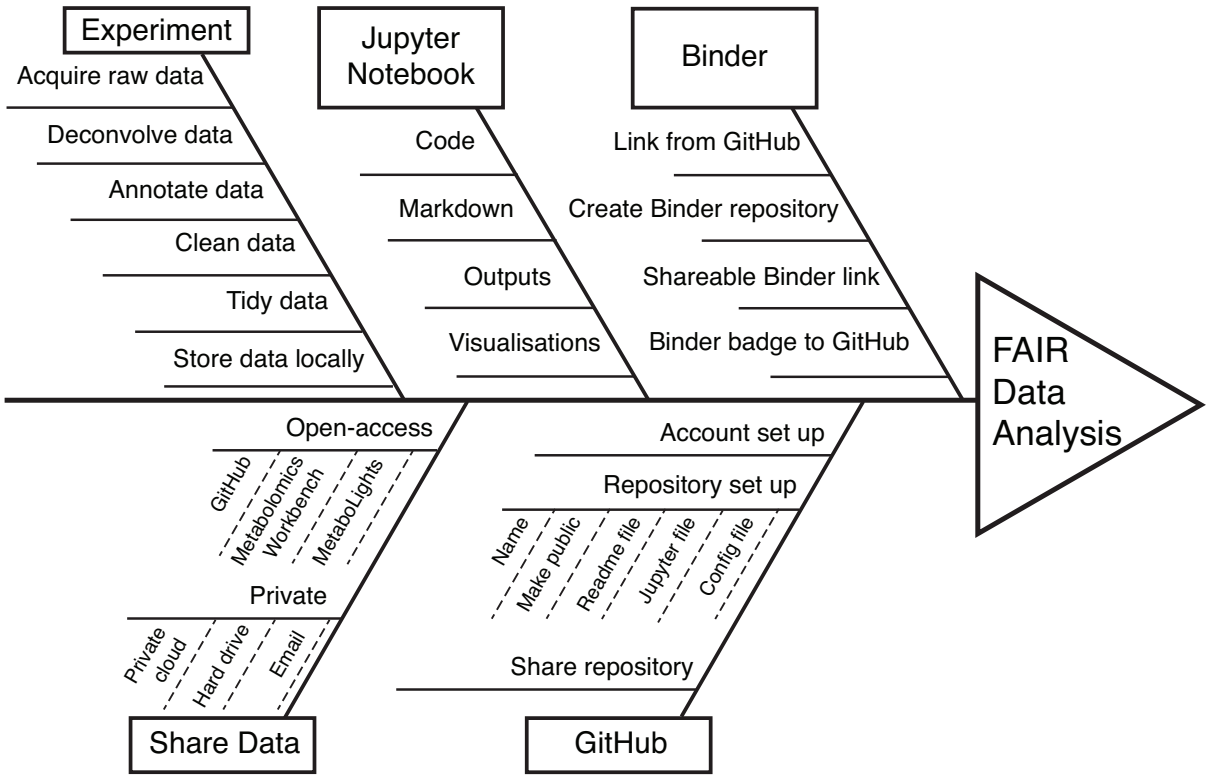


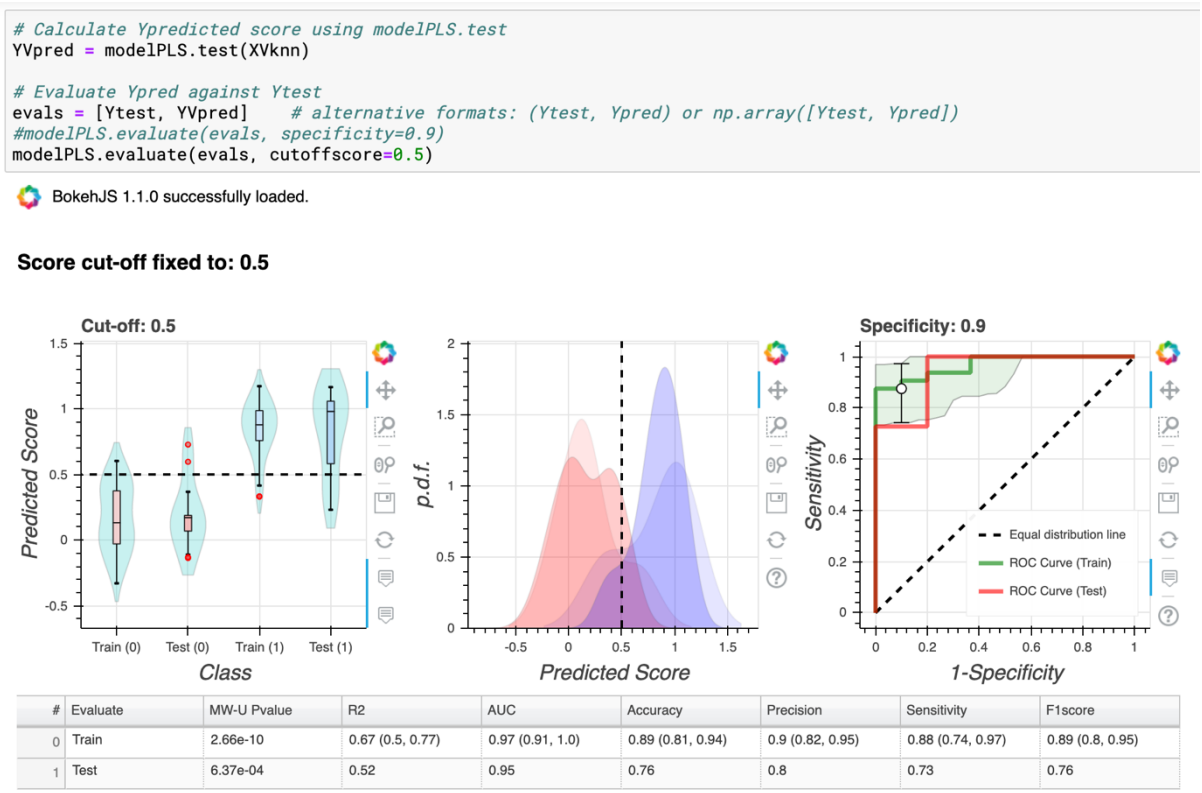
Figure 3.3: Key elements required for FAIR data analysis, using Jupyter Notebooks and Binder deployment. A fishbone diagram describing the detailed requirements for FAIR data analysis in metabolomics. Experimental data are derived from typical metabolomics workflows and formatted appropriately for analysis. Data need to be shared, either privately (for pre-publication collaboration) or publicly (for open dissemination). The Jupyter Notebook contains all code, markdown comments, outputs, and visualisations corresponding to the study. The Jupyter Notebook and other required files (such as Readme and configuration files) are compiled into a public GitHub repository. Finally, Binder is used to easily deploy and share the Jupyter Notebook.

Digital object identifiers (DOI) are widely used to identify academic and government information in the form of journal articles, research reports and data sets. It is also possible to assign a DOI to open access software. Specifically, researchers are able to make the work shared on GitHub citable by archiving with a data archiving tool such as Zenodo (www.zenodo.org) (Sicilia *et al.*, 2017). A detailed tutorial is available (Open Science MOOC, 2018). This archiving tool will 'fix' in time a given repository (e.g. Jupyter notebook and meta data), so that it can be associated with a particular static publication, while allowing the programmer to further develop the notebook on GitHub. The tutorials in this paper are archived with the handle https://doi.org/10.5281/zenodo.3362624 (https://doi.org/10.5281/zenodo.3362624).

3.3.1.1. Jupyter Notebook

Jupyter Notebook (jupyter.org) is a powerful, open-source, browser-based tool for interactive development and presentation of data science projects. Each notebook consists of a collection of executable cells, and each cell contains either text formatted using the Markdown language (Gruber, 2004) or executable code (usually Python or R). When a 'code cell' is executed any graphical or text output (numerical results, figures or tables) is presented within the document immediately below the cell. Figure 3.4 shows an example of a notebook after execution. A popular way to get started with Jupyter Notebooks is to install the Anaconda distribution (anaconda.com), for which graphical installers are available on Windows, macOS and Linux operating systems (anaconda.com/distribution/). After installation a local Jupyter server can be launched using the Anaconda-Navigator application. To run a specific local Jupyter notebook with Anaconda-Navigator the user can navigate to the appropriate local folder using the

browser-based interface, and click on the desired notebook file (which can be identified by the .ipynb suffix).



#	Evaluate	MW-U Pvalue	R2	AUC	Accuracy	Precision	Sensitivity	F1score
0	Train	2.66e-10	0.67 (0.5, 0.77)	0.97 (0.91, 1.0)	0.89 (0.81, 0.94)	0.9 (0.82, 0.95)	0.88 (0.74, 0.97)	0.89 (0.8, 0.95)
1	Test	6.37e-04	0.52	0.95	0.76	0.8	0.73	0.76

Figure 3.4: Example Jupyter Notebook Screenshot. At the top of the page, there is the Jupyter menu bar and ribbon of action buttons. The main body of the notebook then displays text and code cells, and any outputs from code execution. This screenshot taken near the end of Tutorial 1 when the partial least squares discriminant analysis model is being evaluated. Three plots are generated, showing comparisons of the performance of the model on training and holdout test datasets: a violin plot showing the distribution of known positive and negative in both training and test sets, and the class cut-off (dotted line); probability density functions for positive and negative classes in the training and test sets (the training set datapoints are rendered as more opaque); ROC curves of model performance on training (with 95% CI) and test set.

3.3.1.2. GitHub

GitHub (github.com) is a cloud-based web service that helps programmers store, manage, and share their code (and associated data files), as well as track and control changes to their code

(version control). It is free to sign up and host a public code repository, which makes GitHub especially popular with open-source projects and a good choice for distributing Jupyter Notebooks, project-specific code and documentation. Jupyter Notebooks stored publicly on GitHub can be downloaded and run on a local machine using Anaconda or linked to a cloud-based platform. To complete all the steps of this tutorial a (free) GitHub account is required. An account at GitHub may be created by clicking "sign up" on the GitHub home page (github.com) and following the instructions.

3.3.1.3. Binder

Binder (mybinder.org) is an open source web service that allows users to deploy a GitHub repository comprising a collection of Jupyter Notebooks (with configuration files that describe the required computing environment) as a temporary cloud-based virtual machine. The Binder deployment is accessible by web browser and includes the programming language and all necessary packages and data. As with all publicly-accessible cloud storage care must be taken if data are sensitive or private. Researchers can launch the virtual machine in their browser but, because the user environment is temporary, once the session is closed all new results are lost. If changes are made, the user must download any changed files or output they wish to keep.

3.3.2. Tutorials

3.3.2.1. Tutorial 1: Launching and Using a Jupyter Notebook on Binder

This tutorial demonstrates the use of computational notebooks for transparent dissemination of data analysis workflows and results. The tutorial steps though a metabolomics computational

workflow implemented as a Jupyter Notebook and deployed on Binder. The workflow is designed to analyse a deconvolved and annotated metabolomics data set (provided in an Excel workbook) and is an example of the standard data science axiom: *Import, Tidy, Model,* and *Visualise*.

The Jupyter notebook for this tutorial is named *Tutorial1.ipynb* and is available at GitHub in the repository https://github.com/cimcb/MetabWorkflowTutorial. This repository can be downloaded (cloned) to the researcher's own computer, or run on the Binder service. In the text we assume that the tutorial is being run using the Binder service. To open the notebook on Binder, go to the tutorial homepage: https://cimcb.github.io/MetabWorkflowTutorial and click on the topmost "Launch Binder" icon to "launch the tutorial environment in the cloud". It will take a short while for Binder to build and deploy a new temporary virtual machine. Once this is ready the Jupyter notebook landing page will show the files present in this copy of the GitHub repository (Supplementary Fig. 3.1).

The tutorial workflow analysis interrogates a published dataset used to discriminate between samples from gastric cancer and healthy patients (Chan *et al.*, 2016). The dataset is available in the Metabolomics Workbench database (http:// www.metabolomicsworkbench.org, Project ID PR000699). For this tutorial, the data are stored in the Excel workbook *GastricCancer_NMR.xlsx* using the *Tidy Data* framework (Wickham, 2014): each variable is a column, each observation is a row, and each type of observational unit is a table. The data are split into two linked tables. The first, named 'Data', contains data values related to each observation. i.e. metabolite concentrations M_1 ... M_n, together with meta-data such as: 'sample type', 'sample identifier' and 'outcome class'. The second, named 'Peak', contains data that links each metabolite identifier (M_i) to a specific annotation and optional metadata (e.g. mass,

retention time, MSI identification level, number of missing values, quality control measures, etc.). The Excel file can also be downloaded from the Binder virtual machine for inspection on your own machine by selecting the checkbox next to the filename and clicking on the Download button in the top menu (Supplementary Fig. 3.1).

To begin the tutorial, click on the *Tutorial1.ipynb* filename (Supplementary Fig. 3.1). This will open a new tab in your browser presenting the Jupyter notebook (Supplementary Fig. 3.2). At the top of the page there is a menu bar and ribbon of action buttons similar to those found in other GUI-based software, such as Microsoft Word. The interface is powerful, and it is worth taking time to become familiar with it, but for this tutorial only the "Run" button and the "Cell" and "Kernel" drop down menus are required.

The rest of the page is divided into "code cells" and "text cells". The "text cells" briefly outline the context and computation of the "code cells" beneath them. Code and text cells can be distinguished by their background colour (code cells are slightly grey, text cells are slightly red), by the text formatting (code cells have a fixed-width font, text cells have word processor-like formatting), and the "In []:" marker text is present next to each code cell.

To run a single code cell, first select it by clicking anywhere within the cell, which will then be outlined by a green box (if you select a text cell, this box is blue—Supplementary Fig. 3.3). Once a cell is selected, the code in the cell can be executed by clicking on the "Run" button in the top menu. Multiple cells can also be run in sequence by choosing options from the dropdown list in the "Cell" menu item. The options include "Run All" (runs all the cells in the notebook, from top to bottom), and "Run all below" (run all cells below the current selection).

These can be used after changing the code or values in one cell to recalculate the contents of subsequent cells in the notebook.

The "computational engine" that executes the code contained in a notebook document is called the *kernel*, and it runs continually in the background while that notebook is active. When you run a code cell, that code is executed by the kernel and any output is returned back to the notebook to be displayed beneath the cell. The kernel stores the contents of variables, updating them as each cell is run. It is always possible to return to a "clean" state by choosing one of the "Restart Kernel" options from the "Kernel" menu item's dropdown list. Selecting "Restart & Run All" from the "Kernel" dropdown menu will restart the kernel and run all cells in order from the start to the end of the notebook.

Beginning from a freshly-loaded *Tutorial1.ipynb* notebook in the Binder, clicking on "Cell->Run All" or "Kernel->Restart & Run All" will produce a fully executed notebook that matches the output in the static supplementary html file Tutorial1.html (cimcb.github.io/MetabWorkflowTutorial/Tutorial1.html). Choosing "Restart and Clear Outputs" from the "Kernel" dropdown menu, will reset the notebook and clear all data from memory and remove any outputs, restoring its original state.

The tutorial can be completed by reading the text cells in the notebook and inspecting, then running, the code in the corresponding code cells. This is an example of "Literate Programming" that weaves traditional computing source code together with a human-readable, natural language description of the program logic (Knuth, 1984). The notebook interface makes notable advances on the original proposition for literate programming that are used in this tutorial, the most significant of which is that the output of running the code is also incorporated

into the document. The browser interface allows for further enhancements, such as hyperlinks to external webpages for explanations and further reading about technical terms, embedded interactive spreadsheet-like representation of large datasets (e.g. section 2. Load Data and Peak Sheet), and embedded interactive graphical output (e.g. section 4. PCA Quality Assessment).

3.3.2.2. Tutorial 2: Interacting with and Editing a Jupyter Notebook on Binder

The second tutorial is interactive and showcases the utility of computational notebooks for both open collaboration and experiential education in metabolomics data science. Tutorial 2 is accessed on GitHub through the same process as described for Tutorial 1. To open the notebook on Binder, go to the tutorial homepage: https://cimcb.github.io/MetabWorkflowTutorial and click on the topmost "Launch Binder" icon to "launch the tutorial environment in the cloud", then click the *Tutorial2.ipynb* link on the Jupyter landing page. This will present a new tab in your browser containing the second tutorial notebook. The functionality of this notebook is identical to Tutorial 1, but now the text cells have been expanded into a comprehensive interactive tutorial. Text cells, with a yellow background, provide the metabolomics context and describe the purpose of the code in the following code cell. Additional coloured text boxes are placed throughout the workflow to help novice users navigate and understand the interactive principles of a Jupyter Notebook:

Action (red background labelled with 'gears' icon)

Red boxes provide suggestions for changing the behaviour of the subsequent code cell by editing (or substituting) a line of code. For example, the first red cell describes how to change the input dataset by changing the path to the source Excel file.

Interaction (green background with 'mouse' icon)

Green boxes provide suggestions for interacting with the visual results generated by a code cell. For example, the first green box in the notebook describes how to sort and colour data in the embedded data tables.

Notes (blue background with 'lightbulb' icon)

Blue boxes provide further information about the theoretical reasoning behind the block of code or a given visualisation. This information is not essential to understand Jupyter Notebooks but may be of general educational utility and interest to new metabolomics data scientists.

To complete the tutorial, first execute the notebook by selecting the "Restart & Run All" option in the "Kernel" dropdown menu. Move through the notebook one cell at a time reading the text and executing the code cells. When prompted, complete one (or multiple) modifications suggested in each 'action' box, and the click "Run all below" from the "Cell" dropdown menu, observing the changes in cell output for all the subsequent cells. Further guidance is included in the notebook itself.

It is possible to save the edited notebook to the Binder environment, but any changes made to the notebook during the tutorial are lost when the Binder session ends. To keep changes made to the tutorial notebook or its output, modified files must be downloaded to your local computer before you end the session. Modified files can also be downloaded from the Jupyter landing page. To download files, click the checkbox next to each file you wish to download, and then click the 'Download' button from the top menu.

3.3.2.3. Tutorial 3: Downloading and Installing a Jupyter Notebook on a Local Machine

Jupyter Notebooks can be run on a standard laptop or desktop computer in a number of different ways, depending on the operating system. The Anaconda distribution provides a unified, platform-independent framework for running notebooks and managing Conda virtual environments that is consistent across multiple operating systems, so for convenience we will use the Anaconda interface in these tutorials.

To install the Anaconda distribution, first download the *Python 3.x Graphical Installer* package from the Anaconda webpage (https://www.anaconda.com/distribution/) then open the installer and follow the instructions to compete the installation (https://docs.anaconda.com/anaconda/install/). Be sure to download the installer package specific to your computer's operating system (e.g. macOS, Microsoft Windows or Linux). When the process is completed, the "Anaconda Navigator" application will be installed in your applications folder.

To start Jupyter on your machine first launch the Anaconda Navigator application. This will display a home screen with a sidebar menu on the left-hand side and the main area showing a panel of application icons, with short descriptions. Locate the Jupyter Notebook application and icon in this panel and click the "launch" button under the icon. This will start a Jupyter web server and open the Jupyter landing page in your default web browser. To run an existing Jupyter notebook, navigate to the appropriate folder on your computer's filesystem in the Jupyter landing page, and click on the notebook (.ipynb) file you wish to open. To end a Jupyter

session, click on the “quit” button in the top right-hand corner of the Jupyter landing page. Quit now if you have been working along.

To run the Tutorial notebooks, we need to download the tutorial repository containing those notebooks from GitHub and set up a local “virtual environment” that contains the programming libraries and software tools necessary to run the code cells in the notebooks.

To download the notebook and associated files from the Github repository page (https://github.com/cimcb/MetabWorkflowTutorial), click on the green button labelled “clone or download” and choose the option to “Download ZIP”. Save the zip file (MetabWorkflowTutorial-master. zip) in a convenient location. Extract the zip file to create a new folder in the same location as the .zip file, called “MetabWorkflowTutorial-master”. The contents of this folder are the files visible in the repository at the GitHub site. We will refer to this folder as the “repository root”, or just “root”.

The Jupyter Notebooks in the repository require several Python packages to be installed in order to be run successfully. It would be possible to install these on the local computer so that they are visible to, and accessible by, all notebooks on the computer. However, it is often the case that different repositories and projects require alternative, incompatible versions of these packages. So, in practice, it is not usually possible to install a single set of packages that meets the needs of all the projects that a user would want to run. A technical solution to this is to create a new “virtual environment” that contains only the packages necessary for a project to run, and keeps them separate (“sandboxes” them) from any other projects. Environments can be created when required, and deleted when no longer necessary, without affecting other projects or the operation of the computer. It is good practice to create a new virtual environment

for each project, and typical that multiple such environments are set up, and exist simultaneously on the same computer. The Anaconda Navigator application provides an interface for creating and managing these virtual environments.

To create a new virtual environment for the tutorial, first open the Anaconda Navigator application and click on “Environments” in the left-hand sidebar. The main panel will change to list any *virtual environments* that have been created using Anaconda. If no environments have been created only “base (root)” will be listed. To the right of each virtual environment Anaconda Navigator lists the packages that have been installed in that environment.

It is common to create a new environment “from scratch” by specifying individual packages in the Anaconda Navigator, but for this tutorial we will use a configuration file called “environment.yml” that is part of the GitHub repository. This file describes all the packages that are necessary to reproduce an environment for running the tutorial notebooks. To create a new environment from this configuration file, click on “Import” (at the bottom of the main panel of Anaconda Navigator) and navigate to the repository root folder. By default Anaconda Navigator expects configuration files with “.yaml” or “.yml” file extensions, so only the file named “environment.yml” should be highlighted in the file dialog box. Select this file and click “Open”. The “Import new environment” dialogue box will have autocompleted the “Name:” field for the new environment (“MetabWorkflowTutorial”). To complete creation of the new environment, click on the “Import” button. Anaconda Navigator will show a progress bar in the main panel as it creates the new environment.

Once the environment has been created, click on the “Home” icon in the left-hand sidebar. In the main panel, the dropdown should now read “Applications on [MetabWorkflowTutorial]”,

which indicates that the MetabWorkflowTutorial environment which was just created is now active. If "MetabWorkflowTutorial" is not visible, click on the dropdown menu and select that environment. Click on the "Launch" button under Jupyter Notebook in the main panel, to launch Jupyter in your web browser.

The Jupyter landing page will start in your home folder. To use the tutorial notebooks, navigate to the repository root. The notebooks for Tutorial 1 and 2 can now be run on your own computer, just as on Binder, by selecting the appropriate notebook file. However any output or changes to the contents of a notebook file will now be saved persistently in the local computer and can be reused at any time.

As an alternative you may wish to try to create a virtual environment and launch Jupyter in your web browser through a terminal window (command window). To do this open the terminal window (type '*terminal*' in your computer's search box), then type the following five lines of code:

```
git clone https://github.com/cimcb/MetabWorkflowTutorial
cd MetabWorkflowTutorial
conda env create -f environment.yml
conda activate MetabWorkflowTutorial
jupyter notebook
```

Line one creates an exact copy of the github file directory on your local machine in the folder 'MetabWorkflowTutorial'. Line two moves you into that folder. Line three creates the virtual environment called "MetabWorkflowTutorial" using the contents of the environment.yml file.

Line four activates the virtual environment. Line five launches a local Jupyter notebook server and opens the Jupyter landing page in your web browser, from which you can run the tutorials.

To close the local Jupyter notebook server press "<control>c" *twice* in the terminal window and it will ask you to confirm the action. You may then close the virtual environment by typing:

```
conda deactivate
```

When you no longer need the virtual environment, the following will delete it from your computer:

```
conda remove --name MetabWorkflowTutorial --all
```

If you created a virtual environment using Anaconda Navigator you will have to delete the environment before creating a fresh version.

3.3.2.4. Tutorial 4: Creating a New Jupyter Notebook on a Local Computer

Tutorial 4 builds on tutorial 3. Please ensure that the Anaconda Python distribution is installed on your computer.

In this tutorial we will create a new Jupyter notebook that demonstrates the use of visualisation methods available in Anaconda Python without the need to install additional third-party packages. We will upload a generic metabolomics data set and write code to produce four graphical outputs:

1. A histogram of the distribution of QC_{RSD} across the data set.
2. A kernel density plot of QC_{RSD} vs. D-ratio across the data set.
3. A PCA scores plot of the data set labelled by sample type.
4. A bubble scatter plot of molecular mass vs. retention time, with bubble size proportional to QC_{RSD}

The data set included in this tutorial is previously unpublished, and of arbitrary biological value. It describes serum data acquired using a C18+ LC–MS platform consisting of 3084 unidentified peaks and 91 samples. Of the 91 samples, 23 are pooled QCs injected every 5th sample across the experimental run. The Peak table contains information on the molecular mass, retention time of each detected metabolite, and the associated QCRSD and D-ratio values calculated following recommended quality control procedures (Broadhurst *et al.*, 2018). The data are presented in an Excel file using the previously-described "tidy data" format.

Tutorial 4 is available in a GitHub repository at https://github.com/cimcb/MetabSimpleQcViz. Download and unzip the repository to a folder on your own computer, using the method described in Tutorial 3 (the location of this folder will now be the "*repository root*"). This copy (clone) of the repository is for reference only as we will be recreating the contents of this directory under a different name as we move through this tutorial and Tutorial 5.

First create a new Jupyter notebook. To do this, start the Anaconda Navigator application if it is not already open. Ensure that "[base (root)]" is selected in the "Applications on" dropdown list of the main panel, then launch Jupyter Notebook. This will start a new Jupyter notebook server in your browser and show files from the home directory on the landing page. Navigate to the repository root (the "MetabSimpleQcViz" folder). To create a new notebook, click on

the “New” button in the top right corner of the page. This will list supported Jupyter Notebook languages in the drop-down. Select “Python 3” from this list. A new tab will open in your browser, showing a blank notebook called “Untitled” (at the top of the page). Rename the notebook by clicking on the text “Untitled” and replacing it with “myExample”. This will create a new file in the repository called “myExample.ipynb”

When the “myExample.ipynb” notebook is launched, it contains a single empty code cell. We will use this cell to add a title to the notebook. To do this we need to convert the cell type to be a Markdown cell, then type a header in the cell, and execute it. First, select the empty cell by clicking anywhere within the cell. To convert the cell type, click on the dropdown field marked “Code” in the top menu bar and select “Markdown”. The “In[]:” prompt should disappear from the left-hand side of the cell. Now click inside the cell to see the flashing cursor that indicates the cell is ready to accept input. Type “# Tutorial 4” and click on the “Run” button in the top menu. The formatting of the first cell should change, and a new code cell should appear beneath it.

In the new code cell, we will place Python code that:

1. Imports the Pandas package (necessary to load the Excel spreadsheet).
2. Loads the dataset into variables called “data” and “peak”.
3. Report the number of rows and column in the tables.
4. Displays the first few lines of the resulting table.

The required code is provided in the static supplementary html file Tutorial4.*html* (https://cimcb.github.io/MetabSimpleQcViz/Tutorial4.html) and “Tutorial4.ipynb” notebook

and can be copy-and-pasted or typed in manually, as preferred. When the code is complete, click on the “Run” button again to execute the cell. On completion, two tables should be visible below the code cell (one for “data”, one for “peak”), and a new empty code cell should be placed beneath this.

Next we add the code required to draw a histogram of the RSD values across all the detected peaks in this data set. Using the Tutorial4.html file as a guide, add in the required explanatory text and Python code and click on the “Run” button after each step.

Continue adding in the remaining explanatory text and Python code using the Tutorial4.html file. After completion you will have a Jupyter notebook that takes a metabolomics dataset through the process of generating diagnostic plots for quality control. Once you are satisfied with the state of the notebook, it can be saved by clicking on the floppy disk icon (far left on the menu). The notebook can then be closed by clicking “File” and then “Close and Halt” from the top Jupyter menu. The notebook tab will be closed, showing the Jupyter landing page. The Jupyter session can be closed by clicking on “Quit” on the Jupyter landing page tab of your web browser (this tab may not close automatically).

3.3.2.5. Tutorial 5: Deploying a Jupyter Notebook on Binder via GitHub

Tutorial 5 builds on tutorial 3 and 4. To complete this tutorial, we will create a new GitHub repository. A GitHub account is required for this. If you do not already have a GitHub account, please follow the instructions on GitHub at https://help.github.com/en/articles/signing-up-for-a-new-github-account.

To create a new repository, log into the GitHub site (if you are not already logged in) and navigate to your profile page (https://github.com/<yourusername>), then click on the "Repositories" link at the top of the page. To start a new repository, click on the "New" button at the top right of the page. This will open a new page titled "Create a new repository." Each repository requires a name, and this should be entered into the "Repository name" field; use the name "JupyterExample". Beneath the Repository Name field there is an optional Description box, and then below this a choice of public or private repository. Ensure that the 'Public' option is chosen. Select the checkbox to "Initialize this repository with a README" (this is a file in which you will write useful information about the repository, later). Below this is the option to "Add a license" file. There are many alternative licences to choose from (https://choosealicense.com/), and the choice for your own projects may be constrained by funder, home organisation, or other legal considerations. We strongly recommend that all projects carry a suitable licence, and that you add the MIT License to this tutorial repository. Now, to create the repository, click the "Create repository" button.

On successful creation of the repository, GitHub will present the new repository's home page (this will be at https://github.com/<yourusername>/JupyterExample), with some options for "Quick setup". Under the "Quick setup" notice, the LICENSE and README.md file will be shown, and clicking on either will open them. The README.md file for a repository is automatically displayed on the homepage, but in this case, it is empty (we can add text later). Now we need to add the new Jupyter notebook and the Excel data file from tutorial 4 to the repository. We will do this using the GitHub "Upload files" interface, though there are several other ways to perform this action. To use the GitHub interface, click on the 'Upload files' button and either drag files from your computer, or click on "choose your files" to select files with a file dialogue box. Add the 'myExample.ipynb' and 'data.xlsx' files from your repository

root. These files will be placed in the “staging area”, visible on the webpage but not yet committed to the repository.

GitHub imposes version control as a form of best practice on the repositories it hosts. One of the features of version control best practice is that a description of the changes made to a repository should accompany every “commit” to that repository. To do this, enter the text “Add data and notebook via upload” to the top field under “Commit changes.” Then, to commit the files to the repository, click on the “Commit changes” button.

Now that there is a publicly hosted GitHub repository containing a notebook and dataset, we are nearly ready to make the notebook available interactively through Binder. The final necessary component required is a configuration file. This file is vital, as it defines the environment Binder will build, with a specified programming language and all the necessary packages for the notebook to successfully operate. This configuration file is an Anaconda YAML file called ‘*environment.yml*’ and it contains a list of *dependencies* (the programming language version and a list of packages used in the notebook) and *channels* (the location of these resources in the Anaconda cloud library). Detailed consideration of how to create these files is beyond the scope of the tutorial. Upload the environment.yml file from Tutorial 4 (it is also included in the Supplementary File, to cut and paste if required) to the repository in the same way that the notebook and data files were uploaded.

We are now ready to build and launch a Binder virtual machine for this repository. To do this, open https://mybinder.org in a modern web browser. The landing page presents a set of fields to be completed for Binder to build a virtual machine. The minimal requirement is to specify a GitHub repository URL in the “GitHub repository name or URL” field Enter the path to the

home page of your repository (https://github.com/<yourusername>/JupyterExample) in this field, and click on the 'Launch' button. Binder will use the configuration file in the root directory to build and store a Docker image for your repository. This process often takes several minutes.

Once the Binder repository is built, the URL shown in the field "Copy the URL below and share your Binder with others" (here: https://mybinder.org/v2/gh/<yourusername>/JupyterExample/master) can be shared with colleagues. A button to launch the Binder can also be added into the README file on GitHub (we also strongly recommend this). Anyone using this URL in their browser, will be provided with an individual interactive session (1 CPU, 2 GB RAM running on Google Cloud) making available the notebooks of your repository in an interactive and editable form.

Congratulations, you have created your first Binder notebook! Now share it with your colleagues!

It is important to remind users that data uploaded to a public GitHub repository is indeed public. If the user wants to share Jupyter Notebooks but not any associated metabolomics data (or other sensitive data) then clear instructions on how to securely access and download the data needs to be included in the notebook text, and the location of that downloaded data be included in the requisite notebook code block (this could be a local hard drive, or uploaded to Binder while in session). If institutional security concerns preclude using a collaborative workspace such as Binder, then alternative cloud solutions such as Microsoft Azure can be investigated. Before doing so it is probably best that to consult with your institute IT representative.

3.4. Summary

Due to the rate at which data are generated and new analysis and visualisation methods are developed, the omics sciences have become highly vulnerable to irreproducibility. In attempt to ameliorate this, the metabolomics community has made several efforts to align with FAIR data standards in the areas of open data formats, data repositories, online spectral libraries, and metabolite databases. While there are also a number of open options for data analysis, these tend to exist as prescriptive and inflexible workflows that inadvertently enable users to apply data science methods without fully understanding their underlying principles and assumptions. For FAIR data science to exist in metabolomics, presentation of methods and results needs to be rapid, transparent, reusable, and recoverably attached to published work. Furthermore, any framework enabling this must be intuitive and accessible to computational novices.

In this tutorial review, we have illustrated one possible solution for achieving open, transparent, yet intuitive data science within the metabolomics community. Jupyter Notebooks are an open-source, interactive web tool for creating seamless integration of text, code, and outputs (tables, figures) into a single live executable document. When used alongside data repositories, such as GitHub, and open cloud-based deployment services, such as Binder, these computational notebooks can greatly enhance transparent dissemination of data science methods and results during the publication process. In addition to the benefit of increased transparency, computational notebooks provide a valuable tool for open collaboration. Rather than exchanging multiple individual data, code, methods, and results files, computational notebook environments provide a single mechanism for collaborators (both within and beyond a single research group) to share and interact with the data science workflow. Moreover, this interactive nature, combined with the ability to provide extensive documentation, provides a valuable

opportunity for enhanced learning in the computer programming and data science contexts. Given that they are increasingly recognised as being foundational to contemporary research, it is imperative that scientists continue to enhance these skills over the duration their career. This open and interactive framework enables scientists to continue to learn and also keep up-to-date with latest data science methods and trends without reinstalling the wheel.

Acknowledgments

This work was partly funded through an Australian Research Council funded LIEF Grant No. (LE170100021).

Author Contributions

All authors contributed equally to this work. KMM developed the code for the Python cimcb-lite package. KMM & DIB developed the tutorial Jupyter notebook code. LP and SNR edited the learning tutorials to align to current pedagogical best practices. KMM wrote the initial draft of the review section. All authors edited the manuscript.

Data and Software Availability

The metabolomics and metadata used in this paper were retrieved from Metabolights (https://www.ebi.ac.uk/metabolights/) study identifier: MTBLS290, and Metabolomics Workbench (https://www.metabolomicsworkbench.org/) project id: PR000699. This data were converted from the original data format to a clean format compliant with the Tidy Data framework, this is available at the CIMCB GitHub project page (https://github.com/CIMCB).

All software developed for this paper is available at the CIMCB GitHub project page (https://github.com/CIMCB).

Conflict of Interest

The authors have no disclosures of potential conflicts of interest related to the presented work.

Research Involving Human or Animal Rights

No research involving human or animal participants was performed in the construction of this manuscript.

Open Access

References

Afgan, E., Baker, D., Batut, B., van den Beek, M., Bouvier, D., Cech, M., Chilton, J., Clements, D., Coraor, N., Gruning, B.A., Guerler, A., Hillman-Jackson, J., Hiltemann, S., Jalili, V., Rasche, H., Soranzo, N., Goecks, J., Taylor, J., Nekrutenko, A. and Blankenberg, D. (2018) The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. *Nucleic Acids Research* **46,** W537-w544.

Anderson, L.W., Krathwohl, D.R., Airasian, P.W., Cruikshank, K.A., Mayer, R.E., Pintrich, P.R., Raths, J. and Wittrock, M.C. (2001) *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives, abridged edition*. Longman, White Plains, NY.

Baker, M. (2016) 1,500 scientists lift the lid on reproducibility. *Nature* **533,** 452-454.

Baumer, B., Cetinkaya-Rundel, M., Bray, A., Loi, L. and Horton, N.J. (2014) R Markdown: Integrating a reproducible analysis tool into introductory statistics, *Technology Innovations in Statistics Education*.

Beaker X Development Team (2018). Beaker X. http://beakerx.com/ Accessed May 1 2019

Bokeh Development Team (2018). Bokeh: Python library for interactive visualization. http://www.bokeh.pydata.org Accessed May 1 2019

Broadhurst, D., Goodacre, R., Reinke, S.N., Kuligowski, J., Wilson, I.D., Lewis, M.R. and Dunn, W.B. (2018) Guidelines and considerations for the use of system suitability and quality control samples in mass spectrometry assays applied in untargeted clinical metabolomic studies. *Metabolomics* **14,** 72.

Broadhurst, D.I. and Kell, D.B. (2006) Statistical strategies for avoiding false discoveries in metabolomics and related experiments. *Metabolomics* **2,** 171-196.

Chan, A.W., Mercier, P., Schiller, D., Bailey, R., Robbins, S., Eurich, D.T., Sawyer, M.B. and Broadhurst, D. (2016) (1)H-NMR urinary metabolomic profiling for diagnosis of gastric cancer. *British journal of cancer* **114,** 59-62.

Considine, E.C., Thomas, G., Boulesteix, A.L., Khashan, A.S. and Kenny, L.C. (2017) Critical review of reporting of the data analysis step in metabolomics. *Metabolomics* **14,** 7.

Davidson, R.L., Weber, R.J.M., Liu, H., Sharma-Oates, A. and Viant, M.R. (2016) Galaxy-M: a Galaxy workflow for processing and analyzing direct infusion and liquid chromatography mass spectrometry-based metabolomics data. *GigaScience* **5,** 10.

Gehlenborg, N., O'Donoghue, S.I., Baliga, N.S., Goesmann, A., Hibbs, M.A., Kitano, H., Kohlbacher, O., Neuweger, H., Schneider, R., Tenenbaum, D. and Gavin, A.C. (2010) Visualization of omics data for systems biology. *Nature Methods* **7,** S56-68.

Giacomoni, F., Le Corguillé, G., Monsoor, M., Landi, M., Pericard, P., Pétéra, M., Duperier, C., Tremblay-Franco, M., Martin, J.-F., Jacob, D., Goulitquer, S., Thévenot, E.A. and Caron, C. (2015) Workflow4Metabolomics: a collaborative research infrastructure for computational metabolomics. *Bioinformatics (Oxford, England)* **31,** 1493-1495.

GitHub (2019). About GitHub. https://github.com/about Accessed April 30 2019

Goodacre, R., Broadhurst, D., Smilde, A.K., Kristal, B.S., Baker, J.D., Beger, R., Bessant, C., Connor, S., Capuani, G., Craig, A., Ebbels, T., Kell, D.B., Manetti, C., Newton, J., Paternostro, G., Somorjai, R., Sjöström, M., Trygg, J. and Wulfert, F. (2007) Proposed minimum reporting standards for data analysis in metabolomics. *Metabolomics* **3,** 231-241.

Gruber, J. (2004). Markdown. https://daringfireball.net/projects/markdown/ Accessed April 30 2019

Haug, K., Salek, R.M., Conesa, P., Hastings, J., de Matos, P., Rijnbeek, M., Mahendraker, T., Williams, M., Neumann, S., Rocca-Serra, P., Maguire, E., González-Beltrán, A., Sansone, S.-A., Griffin, J.L. and Steinbeck, C. (2012) MetaboLights--an open-access general-purpose repository for metabolomics studies and associated meta-data. *Nucleic acids research* **41,** D781-D786.

Holten, D. (2006) Hierarchical edge bundles: visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics* **12,** 741-8.

Horai, H., Arita, M., Kanaya, S., Nihei, Y., Ikeda, T., Suwa, K., Ojima, Y., Tanaka, K., Tanaka, S., Aoshima, K., Oda, Y., Kakazu, Y., Kusano, M., Tohge, T., Matsuda, F., Sawada, Y., Hirai, M.Y., Nakanishi, H., Ikeda, K., Akimoto, N., Maoka, T., Takahashi, H., Ara, T., Sakurai, N., Suzuki, H., Shibata, D., Neumann, S., Iida, T., Tanaka, K., Funatsu, K., Matsuura, F., Soga, T., Taguchi, R., Saito, K. and Nishioka, T. (2010) MassBank: a public repository for sharing mass spectral data for life sciences. *Journal of Mass Spectrometry* **45,** 703-14.

Hunter, J.D. (2007) Matplotlib: a 2D graphics environment. *Computing in Science & Engineering* **9,** 90-95.

Jones, M. (2013) *Python for biologists*. CreateSpace Independent Publishing Platform, Scotts Valley, CA, USA.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S. and Willing, C. (2016) Jupyter Notebooks – a publishing format for reproducible computational workflows in Loizides, F.a.S., Birgi (Ed), *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, IOS Press, pp. 87 - 90.

Knuth, D.E. (1984) Literate programming. *The Computer Journal* **27,** 97-111.

Kolb, D. (1984) *Experiential Learning: Experience as the Source of Learning and Development*. Prentice Hall, Englewood Cliffs, NJ.

Lantz, B. (2013) *Machine Learning with R*, First edn. Packt Publishing, Birmingham, UK.

Lee, A.H., Shannon, C.P., Amenyogbe, N., Bennike, T.B., Diray-Arce, J., Idoko, O.T., Gill, E.E., Ben-Othman, R., Pomat, W.S., van Haren, S.D., Cao, K.L., Cox, M., Darboe, A., Falsafi, R., Ferrari, D., Harbeson, D.J., He, D., Bing, C., Hinshaw, S.J., Ndure, J., Njie-Jobe, J., Pettengill, M.A., Richmond, P.C., Ford, R., Saleu, G., Masiria, G., Matlam, J.P., Kirarock, W., Roberts, E., Malek, M., Sanchez-Schmitz, G., Singh, A., Angelidou, A., Smolen, K.K., Consortium, E., Brinkman, R.R., Ozonoff, A., Hancock, R.E.W., van den Biggelaar, A.H.J., Steen, H., Tebbutt, S.J., Kampmann, B., Levy, O. and Kollmann, T.R. (2019) Dynamic molecular changes during the first week of human life follow a robust developmental trajectory. *Nature Communications* **10,** 1092.

McKinney, W. (2017) *Python for Data Analysis*, 2nd edn. O'Reilly Media, Inc.

Müller, A.C. and Guido, S. (2017) *Introduction to machine learning with Python : a guide for data scientists*, First edn. O'Reilly Media, Inc, California, USA.

Open Science MOOC (2018). Make your code citable using github and zenodo: a how-to guide. https://genr.eu/wp/cite/ Accessed 14 August 2019

Passey, D. (2017) Computer science (CS) in the compulsory education curriculum: Implications for future research. *Education and Information Technologies* **22,** 421-443.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R. and Dubourg, V. (2011) Scikit-learn: machine learning in python. *Journal of machine learning research* **12,** 2825-2830.

Pedrioli, P.G., Eng, J.K., Hubley, R., Vogelzang, M., Deutsch, E.W., Raught, B., Pratt, B., Nilsson, E., Angeletti, R.H., Apweiler, R., Cheung, K., Costello, C.E., Hermjakob, H., Huang, S., Julian, R.K., Kapp, E., McComb, M.E., Oliver, S.G., Omenn, G., Paton, N.W., Simpson, R., Smith, R., Taylor, C.F., Zhu, W. and Aebersold, R. (2004) A common open representation of mass spectrometry data and its application to proteomics research. *Nature Biotechnology* **22,** 1459-66.

Peters, K., Bradbury, J., Bergmann, S., Capuccini, M., Cascante, M., de Atauri, P., Ebbels, T.M.D., Foguet, C., Glen, R., Gonzalez-Beltran, A., Gunther, U.L., Handakas, E., Hankemeier, T., Haug, K., Herman, S., Holub, P., Izzo, M., Jacob, D., Johnson, D., Jourdan, F., Kale, N., Karaman, I., Khalili, B., Emami Khonsari, P., Kultima, K., Lampa, S., Larsson, A., Ludwig, C., Moreno, P., Neumann, S., Novella, J.A., O'Donovan, C., Pearce, J.T.M., Peluso, A., Piras, M.E., Pireddu, L., Reed, M.A.C., Rocca-Serra, P., Roger, P., Rosato, A., Rueedi, R., Ruttkies, C., Sadawi, N., Salek, R.M., Sansone, S.A., Selivanov, V., Spjuth, O., Schober, D., Thevenot, E.A., Tomasoni, M., van Rijswijk, M., van Vliet, M., Viant, M.R., Weber, R.J.M., Zanetti, G. and Steinbeck, C. (2019) PhenoMeNal: processing and analysis of metabolomics data in the cloud. *Gigascience* **8,** giy149.

Pinu, F.R., Beale, D.J., Paten, A.M., Kouremenos, K., Swarup, S., Schirra, H.J. and Wishart, D. (2019) Systems biology and multi-omics integration: viewpoints from the metabolomics research community. *Metabolites* **9**.

Project Jupyter (2019). Jupyter. https://jupyter.org/ Accessed 19 March 2019

Project Jupyter, Bussonnier, M., Forde, J., Freeman, J., Granger, B., Head, T., Holdgraf, C., Kelley, K., Nalvarte, G., Osheroff, A., Pacer, M., Panda, Y., Perez, F., Ragan-Kelley, B. and Willing, C. (2018) Binder 2.0 - Reproducible, interactive, sharable environments for science at scale, *SCIPY 2018*, Proceedings of the 17th Python in Science Conference, pp. 113-120.

Ramalho, L. (2015) *Fluent Python: Clear, Concise, and Effective Programming*. O'Reilly Media, Inc., Sebastopol, CA, USA.

Reinke, S.N., Galindo-Prieto, B., Skotare, T., Broadhurst, D.I., Singhania, A., Horowitz, D., Djukanović, R., Hinks, T.S.C., Geladi, P., Trygg, J. and Wheelock, C.E. (2018) OnPLS-based multi-block data integration: a multivariate approach to interrogating biological interactions in asthma. *Analytical Chemistry* **90,** 13400-13408.

Rohart, F., Gautier, B., Singh, A. and Lê Cao, K.-A. (2017) mixOmics: An R package for 'omics feature selection and multiple data integration. *PLOS Computational Biology* **13,** e1005752.

Sicilia, M.-A., García-Barriocanal, E. and Sánchez-Alonso, S. (2017) Community curation in open dataset repositories: insights from zenodo. *Procedia Computer Science* **106,** 54-60.

Smith, C.A., O'Maille, G., Want, E.J., Qin, C., Trauger, S.A., Brandon, T.R., Custodio, D.E., Abagyan, R. and Siuzdak, G. (2005) METLIN: a metabolite mass spectral database. *Therapeutic Drug Monitoring* **27,** 747-51.

Spicer, R.A., Salek, R. and Steinbeck, C. (2017) A decade after the metabolomics standards initiative it's time for a revision. *Scientific Data* **4,** 170138.

Sud, M., Fahy, E., Cotter, D., Azam, K., Vadivelu, I., Burant, C., Edison, A., Fiehn, O., Higashi, R., Nair, K.S., Sumner, S. and Subramaniam, S. (2016) Metabolomics Workbench: An international repository for metabolomics data and metadata, metabolite standards, protocols, tutorials and training, and analysis tools. *Nucleic Acids Research* **44,** D463-70.

Teschendorff, A.E. (2019) Avoiding common pitfalls in machine learning omic data science. *Nature Materials* **18,** 422-427.

The Carpentries (2019). Lessons. https://software-carpentry.org/lessons/ Accessed May 20 2019

van der Walt, S., Colbert, S.C. and Varoquaux, G. (2011) The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering* **13,** 22-30.

VanderPlas, J. (2016) *Python Data Science Handbook: Essential Tools for Working with Data*. O'Reilly Media, Inc., Sebastopol, CA, USA.

Waskom, M., Botvinnik, O., O'Kane, D., Hobson, P., Ostblom, J., Lukauskas, S., Gemperline, D.C., Augspurger, T., Halchenko, Y., Cole, J.B., Warmenhoven, J., Ruiter, J.d., Pye, C., Hoyer, S., Vanderplas, J., Villalba, S., Kunter, G., Quintero, E., Bachant, P., Martin, M., Meyer, K., Miles, A., Ram, Y., Brunner, T., Yarkoni, T., Williams, M.L., Evans, C., Fitzgerald, C., Brian and Qalieh, A. (2018). mwaskom/seaborn: v0.9.0. DOI:10.5281/ZENODO.1313201 Accessed May 1 2019

Wickham, H. (2014) Tidy Data. *Journal of Statistical Software* **59,** 1-23.

Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L.B., Bourne, P.E., Bouwman, J., Brookes, A.J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C.T., Finkers, R., Gonzalez-Beltran, A., Gray, A.J.G., Groth, P., Goble, C., Grethe, J.S., Heringa, J., 't Hoen, P.A.C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S.J., Martone, M.E., Mons, A., Packer, A.L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M.A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J. and Mons, B. (2016) The FAIR guiding principles for scientific data management and stewardship. *Scientific Data* **3,** 160018.

Wishart, D.S., Feunang, Y.D., Marcu, A., Guo, A.C., Liang, K., Vazquez-Fresno, R., Sajed, T., Johnson, D., Li, C., Karu, N., Sayeeda, Z., Lo, E., Assempour, N., Berjanskii, M., Singhal, S., Arndt, D., Liang, Y., Badran, H., Grant, J., Serra-Cayuela, A., Liu, Y., Mandal, R., Neveu, V., Pon, A., Knox, C., Wilson, M., Manach, C. and Scalbert, A. (2018) HMDB 4.0: the human metabolome database for 2018. *Nucleic Acids Research* **46,** D608-D617.

Xia, J., Broadhurst, D.I., Wilson, M. and Wishart, D.S. (2013) Translational biomarker discovery in clinical metabolomics: an introductory tutorial. *Metabolomics* **9,** 280-299.

Xia, J. and Wishart, D.S. (2011) Metabolomic data processing, analysis, and interpretation using MetaboAnalyst. *Current Protocols in Bioinformatics* **Chapter 14,** Unit 14.10.

Supplementary Information

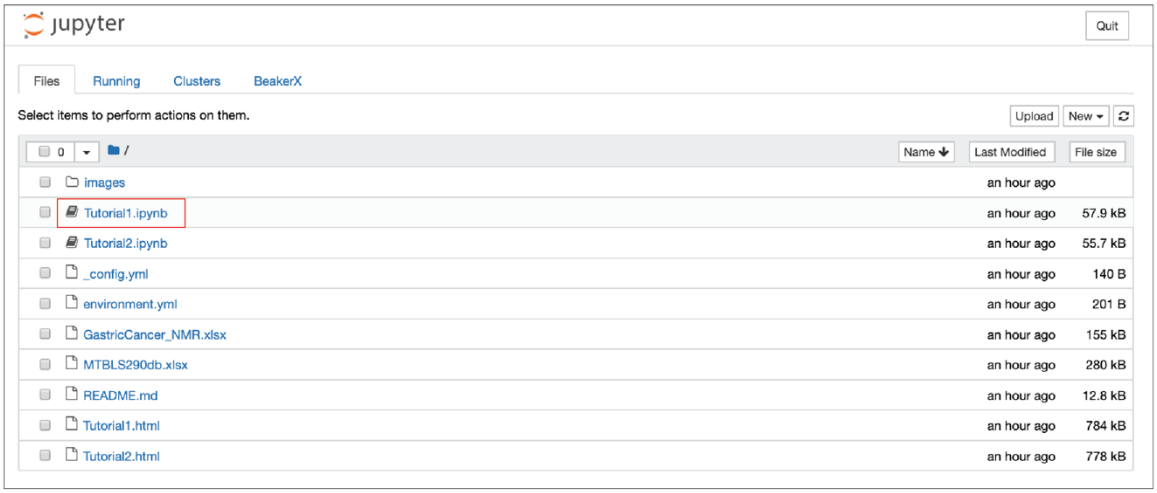
List of supplementary html files:

- Tutorial 1: Tutorial1.html
    - https://static-content.springer.com/esm/art%3A10.1007%2Fs11306-019-1588-0/MediaObjects/11306_2019_1588_MOESM2_ESM.html
- Tutorial 2: Tutorial2.html
    - https://static-content.springer.com/esm/art%3A10.1007%2Fs11306-019-1588-0/MediaObjects/11306_2019_1588_MOESM3_ESM.html
- Tutorial 4: Tutorial4.html
    - https://static-content.springer.com/esm/art%3A10.1007%2Fs11306-019-1588-0/MediaObjects/11306_2019_1588_MOESM4_ESM.html

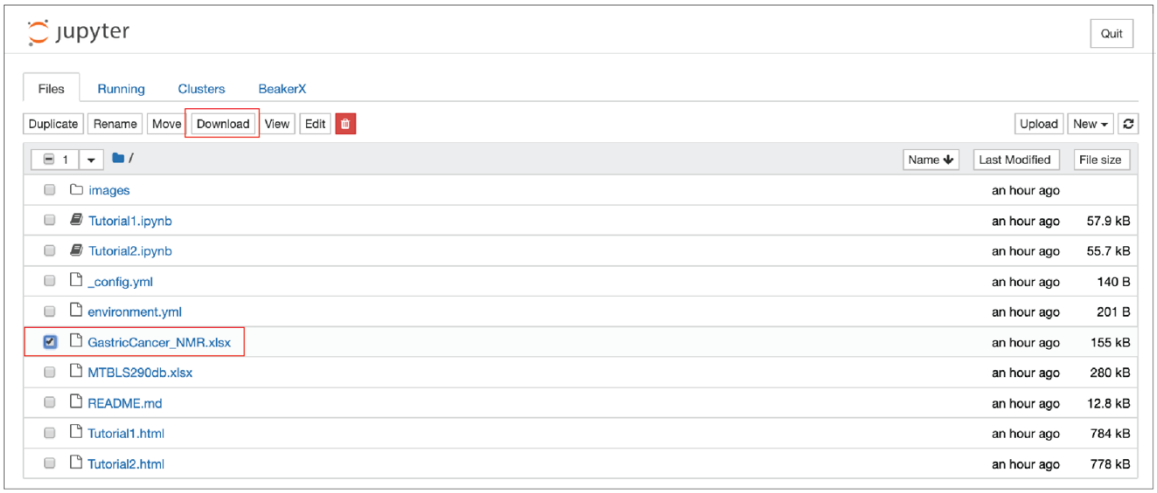
List of supplementary figures:

- Supplementary Fig 3.1
- Supplementary Fig 3.2
- Supplementary Fig 3.3

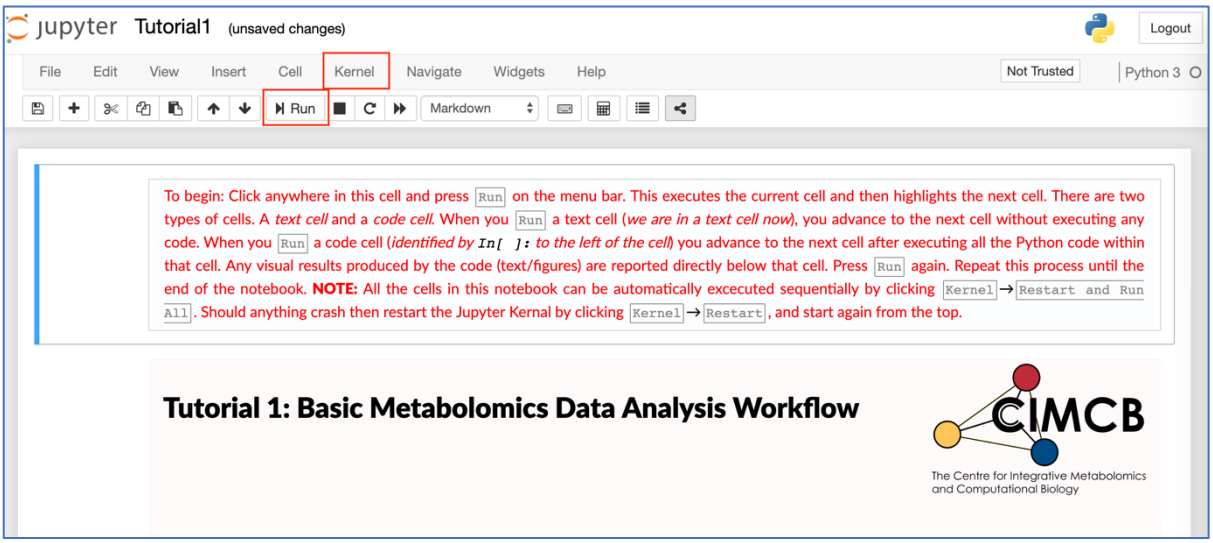
(A)



(B)

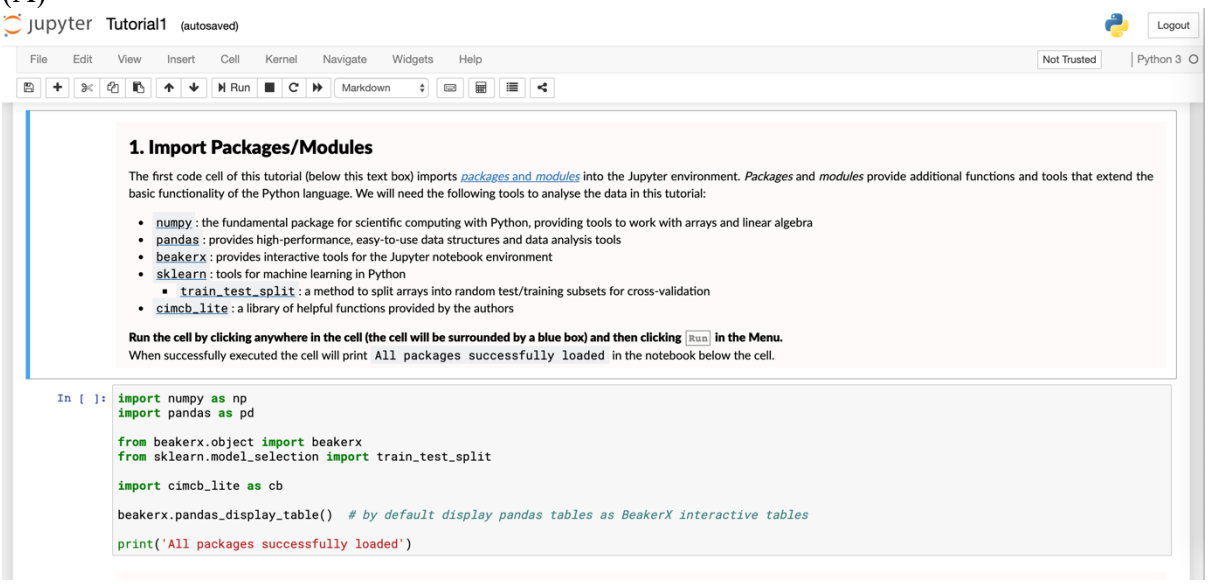


Supplementary Figure 3.1: Jupyter Notebook landing page. (**a**) The Jupyter Notebook landing page containing all of the files present in this copy of the GitHub repository. To launch Tutorial #1, Click on "Tutorial1.ipynb" outlined in red. (**b**) To download any of the files, including the Excel workbook, select them by checking the appropriate boxes on the left side and then click "Download" at the top of the screen.

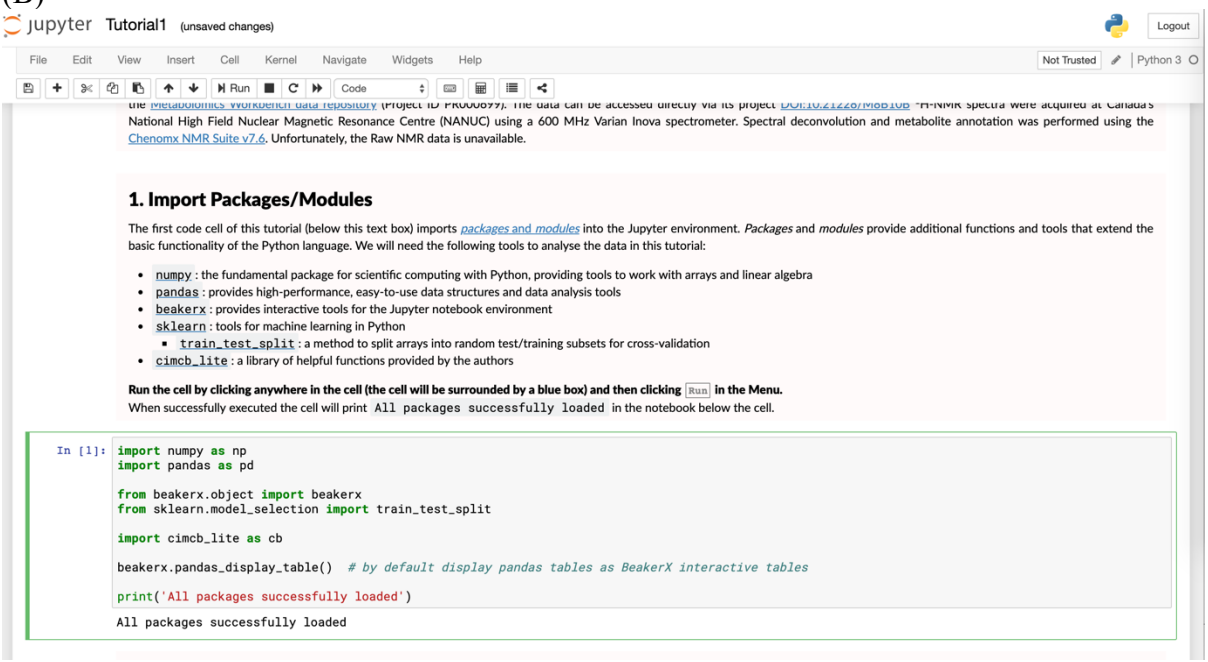


Supplementary Figure 3.2: The top of the Jupyter notebook Tutorial 1. At the top of the page there is a menu bar and ribbon of action buttons similar to those found in other GUI-based software, such as Microsoft Word. The interface is powerful, and it is worth taking time to become familiar with it, but for this tutorial only the "Run" button and the "Cell" and "Kernel" drop down menus are required.

(A)



(B)



Supplementary Figure 3.3: To run a single text cell, first select it by clicking anywhere within the cell, which will then be outlined by a blue box (**a**). To run a single code cell, first select it by clicking anywhere within the cell, which will then be outlined by a green box (**b**). Once a cell is selected, the code in the cell can be executed by clicking on the "Run" button in the top menu. Multiple cells can also be run in sequence by choosing options from the dropdown list in the "Cell" menu item. The options include "Run All" (runs all the cells in the notebook, from top to bottom), and "Run all below" (run all cells below the current selection). These can be used after changing the code or values in one cell to recalculate the contents of subsequent cells in the notebook.

Chapter Four: A Comparative Evaluation of the Generalised Predictive Ability of Eight Machine Learning Algorithms across Ten Clinical Metabolomics Data Sets for Binary Classification

Authors

Kevin M Mendez[1], Stacey N Reinke[1*], David I Broadhurst[1*]

[1]Centre for Integrative Metabolomics & Computational Biology, School of Science, Edith Cowan University, Joondalup, 6027 Australia

*Corresponding authors:

email: d.broadhurst@ecu.edu.au, stacey.n.reinke@ecu.edu.au

phone: +61 (0)8-6304-2705

This is a post-peer-review, pre-copyedit version of an article published in *Metabolomics*. The final authenticated version is available online at: https://doi.org/10.1007/s11306-019-1612-4.

Abstract

Introduction: Metabolomics is increasingly being used in the clinical setting for disease diagnosis, prognosis and risk prediction. Machine learning algorithms are particularly important in the construction of multivariate metabolite prediction. Historically, partial least squares (PLS) regression has been the gold standard for binary classification. Nonlinear machine learning methods such as random forests (RF), kernel support vector machines (SVM) and artificial neural networks (ANN) may be more suited to modelling possible nonlinear metabolite covariance, and thus provide better predictive models.

Objectives: We hypothesise that for binary classification using metabolomics data, non-linear machine learning methods will provide superior generalised predictive ability when compared to linear alternatives, in particular when compared with the current gold standard PLS discriminant analysis.

Methods: We compared the general predictive performance of eight archetypal machine learning algorithms across ten publicly available clinical metabolomics data sets. The algorithms were implemented in the Python programming language. All code and results have been made publicly available as Jupyter notebooks.

Results: There was only marginal improvement in predictive ability for SVM and ANN over PLS across all data sets. RF performance was comparatively poor. The use of out-of-bag bootstrap confidence intervals provided a measure of uncertainty of model prediction such that the quality of metabolomics data was observed to be a bigger influence on generalised performance than model choice.

Conclusion: The size of the data set, and choice of performance metric, had a greater influence on generalised predictive performance than the choice of machine learning algorithm.

Key Words: Metabolomics. Partial least squares, Support vector machines, Random forest, Artificial neural network, Machine learning, Jupyter, Open source

4.1. Introduction

The multidisciplinary field of *data science* is concerned with extracting insights from data using a diverse set of computational methodologies, theories, and technologies (Blei and Smyth, 2017). Within data science, there are two competing scientific philosophies: *classical statistics* and *machine learning* (Breiman, 2001b). Classical statistics aims to formalise relationships between dependent and independent variables based on a clearly defined set of assumptions from which mathematical models are parametrised. The aim is to derive meaningful statistical inference (properties of an underlying probability distribution) for the measured variables, assuming that the observed data is sampled from a larger population. Conversely, machine learning uses ad-hoc computational algorithms that iteratively optimise (or 'learn') without necessarily relying on any formal statistical assumptions (Bishop, 1995). Here, the aim is typically prediction rather than explanation, and inference is replaced by validation through testing the model with new data. Both approaches add insight into a given data set. Ideally, one would like a machine learning method that can be used for both prediction and statistical inference. Historically, for metabolomics (Gromski *et al.*, 2015), that method has been partial least squares regression (PLS) (Wold, 1975; Wold *et al.*, 1993).

PLS has become the standard multivariate machine learning algorithm in metabolomics for several reasons. Firstly, PLS is a projection method, where highly multivariate data is projected into a smaller coordinate space (latent variables) before regressing to a dependent variable. This not only allows data sets with more variables than samples to be modelled without resorting to prefiltering variables (essential for hypothesis-generating experiments), it also plays to the strength of metabolomics, over other 'omic platforms, in that there is inherently a large amount of inter-metabolite covariance in any biological system (Dunn *et al.*, 2011), which is likely best represented as latent structure. Secondly, once optimised, a PLS model can be reduced to the form of a standard linear regression, from which inference about the importance of constituent metabolites can be made (Gromski *et al.*, 2015). Finally, the algorithm is computationally inexpensive, and historically excellent software has been readily available through companies such as Umetrics (Umeå, Sweden) and Eigenvector Research (Washington, USA). This has accelerated its widespread adoption across the metabolomics community.

While easily interpretable PLS is inherently a linear algorithm, capable of modelling only linear latent covariance. As biological data are often non-linear(Mosconi *et al.*, 2008) it is probable that metabolomics data also has a non-linear latent structure. As such, more complex non-linear machine learning methods such as random forest (RF), kernel support vector machine (SVM), and artificial neural networks (ANNs) may be more applicable for analysing metabolomics data. These alternative methods have spasmodically appeared in metabolomics literature, but never really gained much traction. This could be due to convoluted methods for determining metabolite inference, but equally because historically these methods have been computationally expensive, and software lacked widespread availability. As metabolomics experiments continue to become more complex in design, with increasingly large data sets, the opportunity to exploit concomitant advances in computational power and availability of open

source software means that non-linear machine learning algorithms have become a viable alternative to PLS, particularly in situations where predictive performance is more important than inference.

The aim of this study was to compare the general predictive performance of an archetypal set of linear and non-linear machine learning algorithms evaluated across a representative number of clinical metabolomics data sets. The number of data sets was limited to ten and represented a cross section of current published data in terms of measurement instrument, number of samples, and complexity of biological question. This allowed the study to be small enough to be tractable (providing all data and code as interactive Jupyter notebooks) but also large enough to extract some general conclusions. We hypothesise that for binary classification using metabolomics data, non-linear machine learning methods will provide superior generalised predictive ability when compared to linear alternatives, in particular when compared with the current gold standard PLS discriminant analysis.

It is important to note that it is not the aim of this study to challenge the published results related to these data sets, or to pitch data sets against each other. All interpretations should be based only on the relative performance of competing algorithms for a given data set, and then a generalised meta-analysis of performance rankings across data sets. Also, the aim is to compare predictive performance, not metabolite inference, thus no biological interpretation of models is considered.

4.2. Methods

4.2.1. Data Sets

The following criteria were used to identify ten metabolomics data sets for this comparative evaluation:

1. Data were of clinical origin.
2. Data were previously published.
3. Data publicly available at either *MetaboLights* or *Metabolomics Workbench* data repositories (www.ebi.ac.uk/metabolights ; www.metabolomicsworkbench.org)
4. Metabolite data available in a form amenable for direct modelling (All feature selection/deconvolution performed and the resulting data matrix available in either a flat text file or common format of spreadsheet—e.g. Microsoft Excel).
5. Experimental data (e.g. Clinical Outcome) available in a form amenable for direct modelling.
6. A clear binary outcome available to model (either a primary or secondary outcome of the publication, or a subset of a multi-class study) and the number samples in each class are reasonably balanced.
7. Data representative of the three primary metabolomics technologies (nuclear magnetic resonance; gas chromatography mass spectrometry; liquid chromatography mass spectrometry).
8. Data representative of multiple biofluids (e.g. blood, urine, faeces).
9. A range of samples sizes (from less than 50 to more than 500).

The computational framework for this study (Sect. 4.2.3) required data to be converted to a standardised Microsoft Excel file format, using the Tidy Data framework (Wickham, 2014), where each variable forms a column, each observation forms a row, and each type of observational unit forms a table. To this end, for each study, data are split into two linked tables. The first, named **Data**, contains data values related to each observation. i.e. metabolite concentrations M_1 ... M_n, together with metadata such as: *injection order*, *sample type*, *sample identifier* and *outcome class*. The second table, named **Peak**, contains data that links each metabolite identifier (M_i) to a specific annotation (metabolite name) and optional metadata (e.g. *mass*, *retention time*, *MSI identification level*, *number of missing values*, *quality control statistics*). Standardising the data format before data analysis enabled clear presentation, and efficient reuse, of computer code.

4.2.2. Machine Learning Algorithms

The following eight machine learning methods were considered for this study:

1. Partial least squares regression (a.k.a. projection to latent structures).
2. Principal components regression.
3. Principal components logistic regression.
4. Linear kernel support vector machines.
5. Radial basis function kernel support vector machines.
6. Random forests.
7. Linear artificial neural networks.
8. Non-linear artificial neural networks.

All methods were implemented in the Python programming language using standard packages where possible. Python packages*: Sci-kit learn* (Pedregosa et al., 2011*)*, *Numpy* (Kristensen and Vinter, 2010), *Pandas (McKinney, 2010)*, *Bokeh* (Bokeh Development Team, 2018), *Keras* (Chollet, 2015*)*, *Theano (Theano Development Team, 2016)*. Details are provided in the supplementary files.

Before providing a brief overview of each method it is important to understand the concept of a hyperparameter. In machine learning, a *hyperparameter* is a parameter that is used to either configure the structure of the underlying model or the characteristics of the learning process. Its value is fixed before the learning process begins. All other parameters (coefficients, or weights) are determined through the training process. Different algorithms require different, and possibly multiple hyperparameters. Some simple algorithms (such as logistic regression) require none, many require only one (PLS requires only the optimisation of the number of latent variables), and others (such as artificial neural networks and random forests) require many. The number, type and function are described below.

4.2.2.1. Partial Least Squares Regression

Partial least squares regression (PLS) (Wold, 1975; Wold *et al.*, 1993) is a widely used technique for constructing predictive models with metabolomics data (Broadhurst and Kell, 2006; Gromski *et al.*, 2015), especially when the number of independent variables (metabolites) is much larger than the number of data points (samples). PLS uses the *projection to latent space* approach to modelling the linear covariance structure between two matrices (**X** and **Y**). A PLS model will try to find the multidimensional direction in the **X** space that explains the maximum multidimensional variance direction in the **Y** space. In lay terms: if the **X** matrix

is thought of as a set of N data points in M-dimensional space (where, N is the number of samples and M is the number of metabolites), and **Y** is a binary vector, length N, describing the classification of samples (e.g. case = 1 & control = 0), then PLS rotates and projects those data points into a lower dimensional space (typically 2 or 3 dimensions) such that discrimination (covariance) between the two labelled groups in the subspace is maximised.

Classification PLS is generally referred to as PLS *discriminant analysis* (PLS-DA). Importantly, PLS-DA is considered a linear regression method as the final predictive model can be reduced to the standard linear form $y^* = \beta_0 + \beta_1 x_1 + \beta_1 x_2 + \cdots + \beta_n x_N$, where $\beta_0 \ldots \beta_N$ is a vector of PLS coefficients and y^* is the model prediction (typically, we define a positive classification if $y^* > 0.5$ and a negative classification if $y^* < 0.5$). For this study, each PLS model was optimised using the SIMPLS algorithm (de Jong, 1993). PLS models have a single tuning hyperparameter: the number of latent variables (i.e. the number discriminant dimensions the **X** matrix is projected).

4.2.2.2. Principal Component Regression

Principal component regression (PCR) (Hastie *et al.*, 2009; Jolliffe, 1982) was a mathematical precursor to PLS. It builds upon the widely used multivariate descriptive statistical model: principal components analysis (PCA) (Jolliffe, 2002). In PCA the **X** matrix is rotated and projected into a lower dimensional space based on orthogonal covariance, such that principle component 1 (PC_1) describes the direction of maximal variance in **X**, principal component 2 (PC_2) describes the second orthogonal direction of maximal variance, PC_3 the third direction … etc. PCA is converted into a predictive model by using the principal components as independent variables, and **y** as the dependent variable, in a multiple linear regression (MLR),

with coefficients estimated by the least-squares method (Seber, 2004). As with PLS, PCR is considered a linear regression method as the independently calculated PCA + MLR coefficients can be combined and reduced to the standard linear form $y^* = \beta_0 + \beta_1 x_1 + \beta_1 x_2 + \cdots + \beta_n x_N$, where $\beta_0 \ldots \beta_N$ is a vector of PCR coefficients and y^* is the model prediction (typically, we define a positive classification if $y^* > 0.5$ and a negative classification if $y^* < 0.5$). PCR models have a single tuning hyperparameter: the number of principal components to use in the MLR.

4.2.2.3. Principal Component Logistic Regression

PLS and PCR are usually solved by minimizing the least squares error of the model fit to the data. As such, errors are penalized quadratically. The underlying assumption of this method is that model residuals are normally distributed ($\boldsymbol{y} - \boldsymbol{Xb} = \mathcal{N}(0, \sigma)$). For a binary classification problem this may not be a valid (or useful) assumption. Consider a model for categorical outcomes ($y \in \{0,1\},$), where we define a positive classification if $y^* > 0.5$ and a negative classification if $y^* < 0.5$. If the model predicts the outcome to be 23 when truth is 1, or the model predicts the outcome to be -43 when the truth is 0, nothing has been lost. Having an extremely large absolute error of prediction is not detrimental to the classification. However, least squares regression will consider this error important (remember all errors are penalized quadratically) and try to reduce it – unnecessarily. An alternative modelling technique is to make the binary outcome prediction a probability of correct classification, rather than a regression. To do this we use logistic regression. For logistic regression, observations $y \in \{0,1\}$ are assumed to follow a Bernoulli distribution, and uses a logistic loss function to model the dependent variable. The logistic function acts as a squashing function for extreme positive or

negative values, causing large errors to be penalized asymptotically to a constant value (Menard, 2002).

Accordingly, principal component logistic regression (PCLR) differs from PCR only in the change in loss function (logistic rather than quadratic), which can be visualised as a linear regression pushed through a logistic transformation (squashing function). So for PCLR, PCA is converted into a predictive model by using the principal components as independent variables, and $\boldsymbol{y}$ as the dependent variable ($y \in \{0,1\}$), in a logistic regression (LR), with coefficients estimated using the maximum likelihood method (Menard, 2002). PCLR is also considered a linear regression method as the independently calculated PCA + MLR coefficients can be combined and reduced to a model that is "linear in the coefficients" of the form $ln\left(\frac{p_+}{1-p_+}\right) = \beta_0 + \beta_1 x_1 + \beta_1 x_2 + \cdots + \beta_n x_N$, where $\beta_0 \ldots \beta_n$ is a vector of PCLR coefficients and p_+ is the predicted probability of positive outcome. PCLR models have a single tuning hyperparameter: the number of principal components to use in the MLR.

4.2.2.4. Linear Kernel Support Vector Machines

The objective of the linear kernel support vector machine (SVM-Lin) algorithm is to find a hyperplane in an *M*-dimensional space (*M* = the number of features) that distinctly classifies the *N* data points in the $\mathbf{X}$ matrix ($N \times M$). To separate two classes of data points, there are many possible hyperplanes that can be chosen. The role of the SVM algorithm is to determine the direction (or rotation) of the hyperplane that maximises the margin of discrimination (i.e. the distance between the closest data points at the edge of each class is made as large as possible). The support vectors are the data points that best define this margin. Importantly, and what makes SVM unique, is the process of maximising the margin makes the SVM robust to

correctly classifying new data that may lie within that margin either side of the classification hyperplane (acting like a classification buffer). The loss function that enables SVM to maximize the margin is called the *hinge loss* function. SVM-Lin models have a single tuning hyperparameter called the regularization parameter (termed '*C'* for 'cost' by the Python library used in this study). The regularization parameter allows some flexibility regarding the number of misclassifications made by the hyperplane margin (and can be thought of as the degree in which the buffer of a given thickness is enforced - Supplementary Fig 4.1). For a large value of C, the SVM will choose a small margin for the hyperplane if that hyperplane does a better job of getting all the training points classified correctly (hard margin). Conversely, a small value of C will cause the SVM to optimise to a larger margin separating hyperplane, even if that hyperplane misclassifies more points (soft margin). This regularisation is very important for allowing the SVM to generalise well and not over inflate the importance of individual data points in the optimisation process. An excellent detailed, and more mathematical, explanation of SVM is provided by Steinwart and Christmann (2008).

4.2.2.5. Radial Basis Function Kernel Support Vector Machines

SVMs can also be configured to perform non-linear classification by implicitly mapping input data into a high-dimensional feature space. This process is known as the *kernel trick*. The idea is to gain linearly separation by mapping the data to a higher dimensional space (see Supplementary Fig 4.2). There are many kernel functions available, but the most popular is the *radial basis function* (RBF). An RBF, $\varphi(x, y)$, maps the distance between two points into the range $[0,1]$ using a nonlinear transformation such that $\varphi(x, y) = \varphi(\| x - y \|)$. The standard RBF function is the Gaussian function: $\varphi(x, y) = e^{-(\gamma \| x - y \|)^2}$, where γ is a shaping parameter to be tuned. The optimisation process for SVM-RBF is then identical to the SVM-Lin except

now the optimal linear hyperplane is found with the assistance of the additional radial dimension, equivalent to a nonlinear hyperplane in the original data space. SVM-RBF models have a two tuning hyperparameters: (i) the regularization parameter *C* (as described in 4.2.2.4) and (ii) the gaussian shape parameter, γ. If γ is large the Gaussian shape is very tight leading to over-fitting. Conversely, if γ is very small, the transformation is ineffective. The two hyperparameters are somewhat interdependent. A small value of C can compensate for a large value for γ. An excellent explanation of kernel methods applied to SVM is provided by Schölkopf and Smola (2001).

4.2.2.6. Random Forests

Random Forest (RF) classifiers are radically different to the other ML methods used in this study. They are a type of ensemble classifier, where multiple *base classifiers* are trained and then aggregated to generate a single prediction. To avoid strong correlation between base classifiers, which in turn leads to overfitting, each base classifier must be unique, and thus differ in either the algorithm used, hyperparameter settings, or the training data. With RFs the base classifier is a *decision tree*. Thus, we are dealing with an ensemble of many decision trees (a forest of random decision trees).

A decision tree is top-down hierarchical structure of nodes connected by branches visualised as an inverted tree (Supplementary Fig 4.3). Each node contains a logical question that sends a sample down one of two branches (a binary split), which in turn leads to another node, and on, and on, until it reaches a terminal node, which will provide a predicted classification. For example, to classify a new sample (say, based on a metabolite profile of 300 metabolites: $m_1 \dots m_{300}$) we start at the *root node* and performs the split described therein (e.g. **if** $m_5 > 52$

then *Branch 1*, **else** *Branch 2*). Depending on the result we then descend the tree to the next *internal node* (e.g. **if** $m_{254} > 22$ **then** *Branch 3*, **else** *Branch 4*). Eventually we reach a *leaf node* at which time a classification is made (e.g. **if** $m_{42} > 12$ **then** *Case*, **else** *Control*). The result is a complex, but intuitive, multivariate binary-logic based predictive classification algorithm. However, inherently, the deeper the tree the fewer data points are used to split the samples into different classes, and as such they are prone to overfitting unless very large data sets are employed.

Random forest classifiers aggregate multiple trees (typically 100+ trees) to ameliorate the overfitting problem. Specifically, it uses Classification and Regression Tree (CART) optimisation (Breiman *et al.*, 1984). The algorithm also reduces the previously mentioned correlation issue by allowing only a random subset of features on which to base the split at each node (typically the number in this subsample is equal to the square root of the total number of available features). To avoid any additional overtraining, trees can be constrained to a maximum depth and, during training, the minimum number of samples at each split and a minimum number of samples at each leaf node can be fixed. It has been shown that averaging the classification across many overtrained shallow CARTs produces a robust multivariate classifier (Breiman, 2001a). For this comparative study using metabolomics data our preliminary analysis showed that varying many of the hyperparameters had minimal impact on final RF performance (i.e. 'number of trees'; 'number of features sampled during training'; 'minimum number of samples at each split'), thus they were kept constant at their default values. This reduced the number of tuneable hyperparameters to: (i) *tree depth*, and (ii) *minimum number of samples classified at each leaf node during training (percentage)*.

4.2.2.7. Linear Artificial Neural Network

Artificial neural networks (ANNs), inspired by the biological interconnections in the brain, consist of a layered weighted network of interconnected mathematical operators (neurons). The most common ANN is the feed-forward neural network. Here, each neuron acts as a weighted sum of the outputs of the previous layer applied multiplied to an activation function (typically linear or logistic function). Thus, a neuron with a linear activation is equivalent to a multiple linear regression, and a neuron with a logistic activation function is equivalent to logistic regression. A two-layer ANN (Supplementary Fig. 4.4) with a small number of linear neurons in the 1st layer (hidden layer) and a single linear neuron in the 2nd layer (output layer) is mathematically equivalent to PLS-DA, PCR. Moreover, a two-layer ANN with a small number of linear neurons in the hidden layer and a single logistic neuron in the output layer is mathematically equivalent to PCLR.

During ANN training, the interconnection weights between each layer of neurons (equivalent to coefficients in a regression) are iteratively optimised in a two-phase cycle. Firstly, data is projected though the model to generate a prediction (forward propagation), after which an error term is calculated based on the difference between the target and predicted outputs for all available data. This error is then projected back through the network, and individual weights are adjusted along the way (backward propagation). The aim is to optimise the classification performance by minimising misclassification using an appropriate loss function. For binary classification the best ANN loss function is cross-entropy: $loss = -(y \times \ln(p_+) + (1-y) \times \ln(1-p_+))$ where p_+ is the predicted probability of positive classification and y is the expected binary outcome. For ANN this loss function is then optimised using a gradient descent method (calculating the local loss function gradient and adjusting weights

accordingly). The effectiveness of these methods is dependent on parameters that determine the rate and momentum of traversing the local error gradients (specifically 'learning rate', 'momentum', and 'decay' of the learning rate over time). This unique training method, known as backpropagation, allows for flexibility of ANN network architectures and a multitude of activation functions. For a detailed introduction to feedforward ANN please refer to Bishop (1995). When many layers on neurons are stacked in sequence the ANN is known as deep learning. Deep learning networks are beyond the scope of this study, but clearly warrant further investigation.

For this comparative study, a linear two-layer ANN with a small number of linear neurons in the hidden layer and a single logistic (sigmoidal) neuron in the output layer (ANN-LS) was implemented using stochastic gradient descent, with a binary cross-entropy loss function. Preliminary explorative analysis indicated that hyperparameters: *momentum*, and *decay*, could be set to a constant value (0.5 and 0 respectively) with little variation on performance. The hyperparameters *epochs* (number of training iterations), and *learning rate* are interdependent. Thus, we fixed the number of epochs (400) and varied the learning rate. This reduced the number of tuneable hyperparameters to: (i) the *number of neurons in the hidden layer*, and (ii) the *learning rate*.

4.2.2.8. Non-Linear Artificial Neural Network

To make the linear ANN into a non-linear ANN the hidden layer neurons can be changed to a non-linear activation function. In effect this is similar to the kernel trick described to SVM except the extra dimension is added to the latent variable space (hidden neuron space) rather than directly to the problem space. Although ANN with RBF hidden neurons were one of the

first ever reported kernel methods (Broomhead and Lowe, 1988; Park and Sandberg, 1991) the more popular ANN with sigmoidal hidden neurons proved to be more effective (Bishop, 1995; Wilkins *et al.*, 1994). Thus, the final ML method in our collection is a two-layer ANN with a small number of sigmoidal hidden neurons and a single sigmoidal output neuron (ANN-SS) implemented using stochastic gradient descent, with a binary cross-entropy loss function. Again, the *momentum*, *decay* and *epochs* hyperparameters could be set to a constant value (0.5, 0, 400 respectively) without any detriment to performance. This reduced the number of tuneable hyperparameters to: (i) the *number of neurons in the hidden layer*, and (ii) the *learning rate*.

4.2.3. Computational Workflow

All workflows were implemented using the Python scripting language, presented in the form of interactive Jupyter notebooks following standard guidelines (Mendez *et al.*, 2019b). All data and notebooks are publicly available on GitHub (https://cimcb.github.io/MetabComparisonBinaryML). Details of minor variations in the workflow for each individual model are provided at the top of each notebook (also provided in static html format as supplementary data). The standardised workflow for building, optimising, evaluating, and reporting each of the 80 models generated in this study is summarised below.

4.2.3.1. Splitting Data into Training and Test Sets

Multivariate predictive models are prone to overfitting. In order to provide some level of independent evaluation it is common practice to split the source data set into two parts: training data (X_{train} and Y_{train}) and test data (X_{test} and Y_{test}). The model is then optimised using the

training data and independently evaluated using the test data. The true effectiveness of a model can only be assessed using the test data (Broadhurst and Kell, 2006; Xia *et al.*, 2013). It is imperative that both the training and test data are equally representative of the sample population, or else the test prediction will prone to sampling bias. For these workflows each data set is split with a ratio of 2:1 (2/3 training, 1/3 test) using stratified random selection. The data is split once and then applied to each ML method.

4.2.3.2. Optimisation

Using the training data only, each model was optimised either using a linear search of a single hyperparameter, or a grid search of two hyperparameters, depending on the model type. Following fivefold cross-validation with 10 Monte Carlo repartitions (Broadhurst and Kell, 2006; Hastie *et al.*, 2009), plots of $|R^2 - Q^2|\ vs. Q^2$ were generated to determine the optimal hyperparameter values (where R^2 is the coefficient of determination for the full data set, and Q^2 is the mean coefficient of determination for cross-validated prediction data across the 10 MC repartitions). The optimal hyperparameter was selected at the point of inflection of the outer convex hull of the $|R^2 - Q^2|\ vs. Q^2$ data (i.e. Pareto optimization (Miettinen, 1999)) (Fig 4.1). If a clear inflection point was not present the hyperparameter (outcome) sitting on the Pareto front closest to the line $|R^2 - Q^2| = 0.2$ was deemed optimal, based on the general rule that a difference between training and validation performance greater than 20% is indicative of overtraining (Eriksson *et al.*, 2013). It has been previously shown (Szymańska *et al.*, 2012) that for binary PLS-DA a more appropriate measure of performance is the area under the receiver operating characteristic curve (AUC). As such, plots of $|AUC_{Full} - AUC_{CV}|\ vs.\ AUC_{CV}$ were also provided and utilised as appropriate.

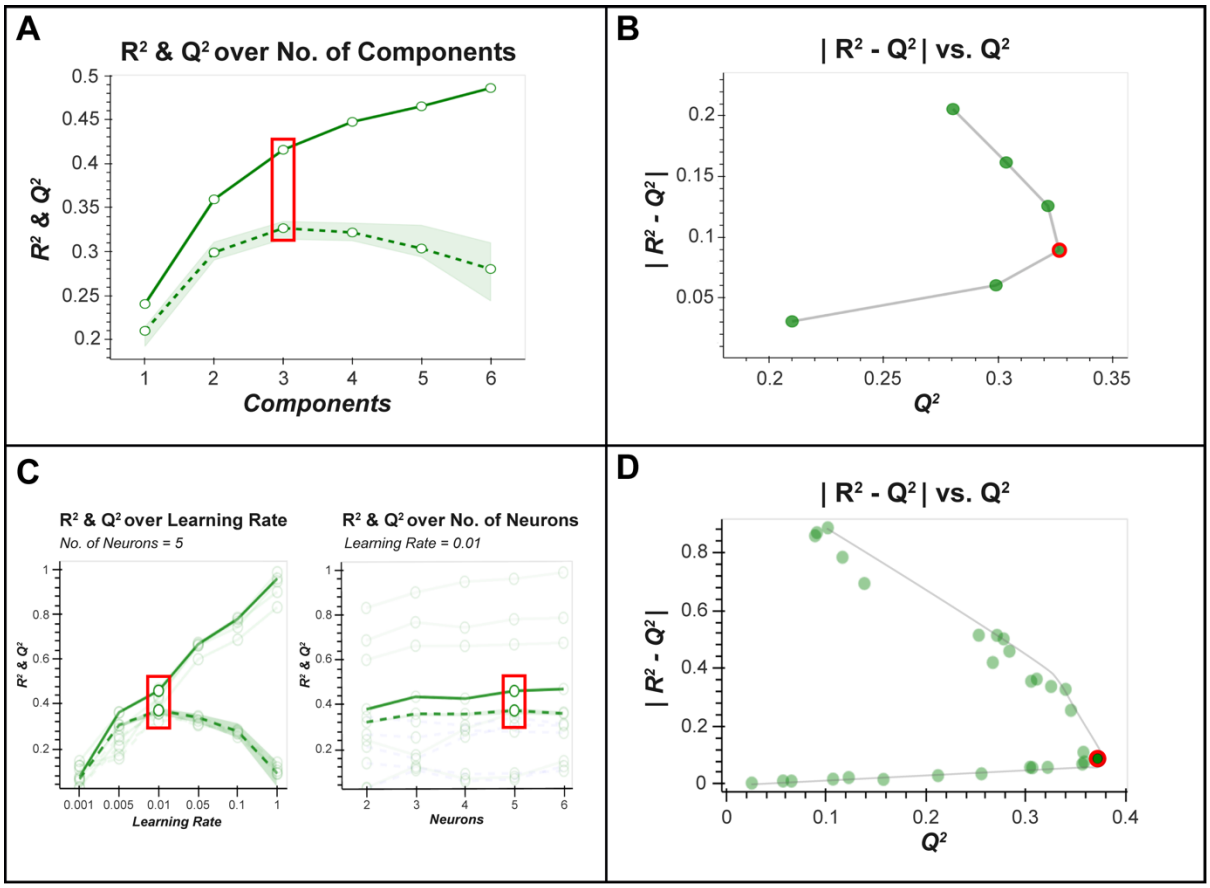


Figure 4.1: Hyperparameter optimisation. **a** An example of a standard R^2/Q^2 plot used for single hyperparameter optimisation (e.g. PLS). The optimum hyperparameter value (number of latent variables) indicated by the red square. **b** The corresponding generalised $|R^2 - Q^2|$ $vs. Q^2$ plot used for hyperparameter optimisation that is extended from (**a**), where the optimal number hyperparameter value (red circle) lies at the inflection of the data curve. **c** An example of a standard R^2/Q^2 plot used for multiple hyperparameter optimisation (e.g. ANN – one plot for "number of neurons" and another for "learning rate"). These plots are difficult to interpret as there are multiple curves for a give fixed value of the 1st hyperparameter across all the possible values of the 2nd hyperparameter. **d** The corresponding $|R^2 - Q^2|$ $vs. Q^2$ plot where each point corresponds to the evaluation for a pair of hyperparameter values. The optimal point, at the infection of the Pareto curve, is labelled as a red circle and this corresponds to the two red squares in (**c**), and optimal hyperparameter pair: number of neurons = 5 & learning rate = 0.01.

4.2.3.3. Model Evaluation using Test Data

Using the optimal hyperparameters, a new model is fit using the training data (X_{train} and Y_{train}). When X_{train} is applied to the model it produces a training prediction data (Y^*_{train}). The similarity of Y_{train} to Y^*_{train} gives an indication of training performance. The model is then

independently evaluated by applying the test metabolite data (X_{test}; transformed and scaled using the metrics applied to X_{train}). This produces a test prediction (Y^*_{test}). The similarity of Y_{test} to Y^*_{test} gives an indication of test performance. For binary classification the best performance indicator is the receiver operator characteristic (ROC) curve (i.e. ROC_{train}, ROC_{test}) which can be further reduced to a single statistic using the area under the ROC curve (i.e. AUC_{train}, AUC_{test}).

4.2.3.4. Generalised Predictive Ability

Although the above 'test data evaluation' gives a good estimate of the true model performance when data sets are large, it potentially gives a biased estimate of performance when data sets are small. All sampled data sets are subject to sampling bias, such that they may not be truly representative of the generalised relationship being modelled (e.g. the metabotype for a specific disease). The smaller the sample data set the higher the probability of bias. This problem is only compounded when an already small sample is split into training and test data set. This bias can result in overly optimistic, or overly pessimistic evaluation, depending on the random chance of selecting an unrepresentative test set.

A measure of this uncertainty in prediction can be determined empirically by calculating confidence intervals of both the training and test evaluation metrics using bootstrap resampling (DiCiccio and Efron, 1996; Efron, 2000). The theoretical details of bootstrapping are beyond the scope of this paper. Briefly, this methodology allows accurate estimation of sampling distributions for almost any statistic by repeated random sampling. Each random sample selects ~ 2/3rd of the data points (called the in-bag sample, IB) leaving ~ 1/3rd (the out-of-bag sample, OOB). As such, bootstrapping can be useful for the evaluation of the optimal ML model

configuration in metabolomics (Broadhurst and Kell, 2006; Mendez *et al.*, 2019b; Xia *et al.*, 2013).

In this study, for each workflow, a model with the fixed optimal hyperparameter values (derived in 4.2.3.3) is retrained on data randomly sampled (IB sample) from the complete data set, and then evaluated on the unused data (OOB sample) for 100 resamples. This produces 100 different models, and therefore 100 IB predictions, and 100 OOB predictions. These predictions can then be translated into ROC curves from which 95% confidence interval can be calculated.

Note: The most effective way to get a true estimate of general performance is to ask a candidate model to predict scores for independently measured data (independent test data). Unfortunately, for the studies used in this paper, independent test data were unavailable. As such, the metrics presented are only **estimates**; however, the variability presented though confidence intervals allows some understanding of the uncertainty of any explicit single model performance metric, particularly when metrics are being compared across multiple competing ML algorithms (Xia *et al.*, 2013)

4.3. Results

4.3.1. Data Sets

The ten data sets curated for this study are described in Table 4.1. Six of the data sets were retrieved from *Metabolights* and four from *Metabolomics Workbench* data repositories. Six data sets acquired using LC–MS, two using NMR, and two using GC–MS. There was a cross

section of biofluids (Plasma, Serum, Urine, Caecal, Saliva, Stool). The size of data set ranged from 59 to 968 subjects (data sets were reasonably balanced in outcome). Number of metabolites included in each data set ranged from 29 to 689. The outcome comparison (binary classification) performed is briefly described in the table and explained in detail at the top of each Jupyter notebook in the supplementary html files. Each data set was split into 2/3 training and 1/3 test using stratified random selection. The identical training and test sets were applied to each ML method so that comparison was unbiased.

Table 4.1: The ten data sets curated for this study.

Study ID	***Publication***	***Platform***	***Type***	***No. of Samples (pos/neg)***	***No. of Peaks***	***Case/Control***
MTBLS90†	Ganna *et al.* (2014); Ganna *et al.* (2015)	LC-MS	Plasma	968 (485/483)	189	Sex (M/F)
MTBLS92†	Hilvo *et al.* (2014)	LC-MS	Plasma	253 (142/111)	138	Breast Cancer Chemotherapy (Before/After)
MTBLS136†	Stevens *et al.* (2018)	LC-MS	Serum	668 (337/331)	689	Postmenopausal hormone (Estrogen / Estrogen + Progesterone)
MTBLS161†	Armstrong *et al.* (2015)	NMR	Serum	59 (34/25)	29	Chronic fatigue syndrome (case/control)
MTBLS404†	Thévenot *et al.* (2015)	LC-MS	Urine	184 (101/83)	120	Sex (M/F)
MTBLS547†	Zheng *et al.* (2017)	LC-MS	Caecal	97 (46/51)	42	High fat diet (case/control)
ST000369*	Fahrmann *et al.* (2015)	GC-MS	Serum	80 (49/31)	181	Adenocarcinoma (case/control)
ST000496*	Sakanaka *et al.* (2017)	GC-MS	Saliva	100 (50/50)	69	Debridement (pre/post)
ST001000*	Franzosa *et al.* (2019)	LC-MS	Stool	121 (68/53)	747	Inflammatory bowel diseases (Crohn's Disease/ Ulcerative Colitis)
ST001047*	Chan *et al.* (2016)	NMR	Urine	83 (43/40)	149	Gastric Cancer (Gastric Cancer/Healthy)

* Indicates data sourced from Metabolomics Workbench (https://www.metabolomicsworkbench.org/)
† Indicates data sourced from Metabolights (https://www.ebi.ac.uk/metabolights/)

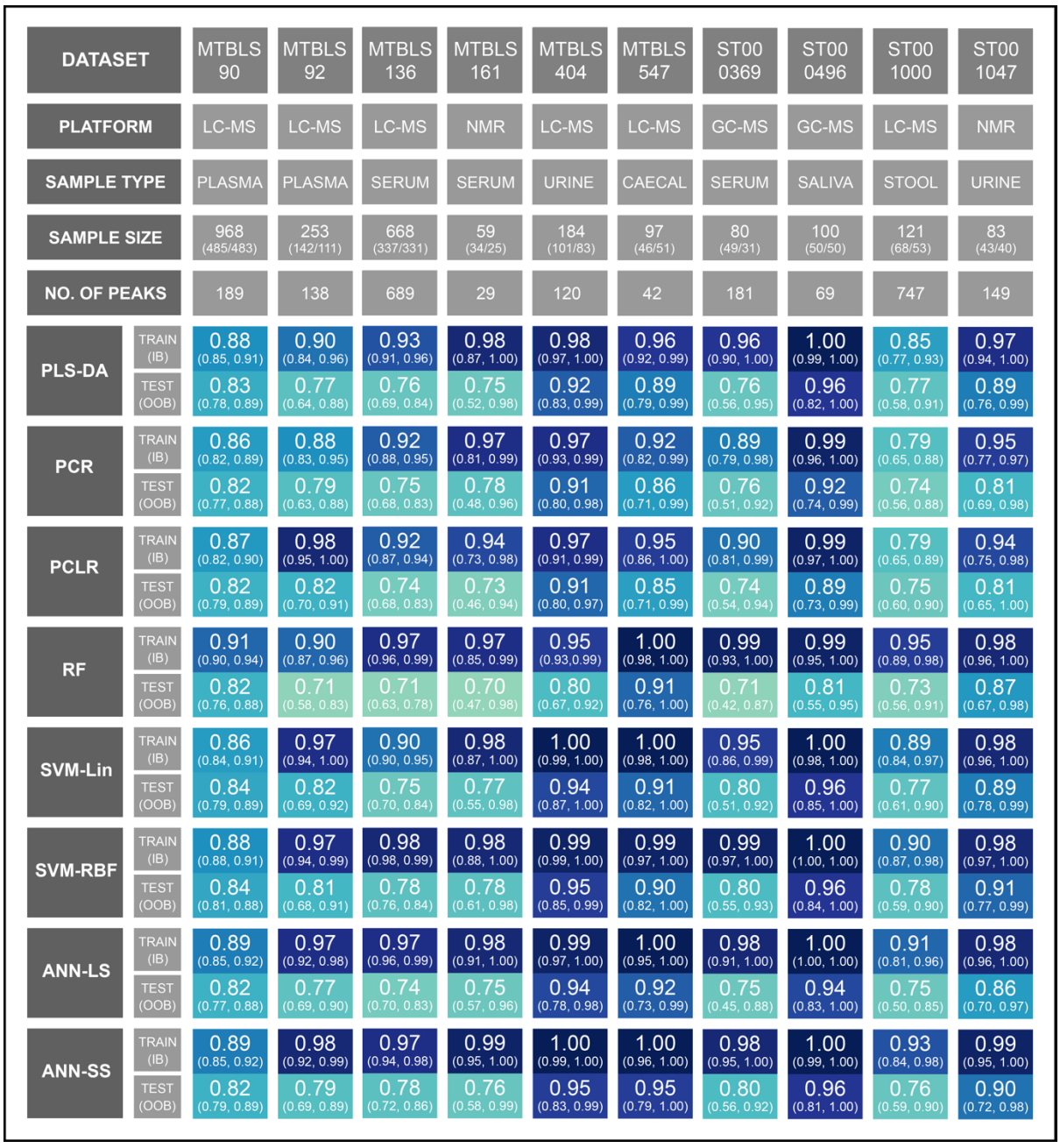
4.3.2. Comparative Evaluation of Generalised Predictive Ability across ML Methods

The hyperparameters for all 80 models were successfully optimised (see supplementary html files). For each optimally configured model, training/test data ROC curves was constructed and AUC_{train} / AUC_{test} calculated. Bootstrap resampling/retraining (n = 100) was performed and in-bag (IB) / out-of-bag (OOB) 95% confidence intervals were calculated. These results are presented as an annotated heatmap in Fig. 4.2. An interactive version of this figure linking each performance metric to a unique Jupyter notebook (including multiple statistics and visualisations) is available here: https://cimcb.github.io/MetabComparisonBinaryML/.

If the 95% confidence intervals are initially ignored, and the comparative evaluation across ML methods is based exclusively on the explicit test set predictions (AUC_{test}), then SVM-RBF performs best across all data sets, closely followed by the nonlinear ANN-SS; however, the mean difference in AUC_{test} between SVM-RBF and ANN-SS across all data sets was only 0.004 (0.4%). The mean difference in AUC_{test} between SVM-RBF and PLS-DA was 0.02 (2%).

The mean difference in AUC_{test} between SVM-Lin and SVM-RBF was 0.006 (0.6%). The mean difference in AUC_{test} between ANN-LS and ANN-SS was 0.023 (2.3%).

When the OOB 95% confidence intervals is used for test prediction then no single ML method is superior. ANN-LS, ANN-SS, SVM-Lin, SVM-RBF, and PLS-DA have very similar confidence intervals for each data set (for example, Fig. 4.3 shows the complete set of ROC curves for data set MTBLS404).



DATASET		MTBLS 90	MTBLS 92	MTBLS 136	MTBLS 161	MTBLS 404	MTBLS 547	ST00 0369	ST00 0496	ST00 1000	ST00 1047
PLATFORM		LC-MS	LC-MS	LC-MS	NMR	LC-MS	LC-MS	GC-MS	GC-MS	LC-MS	NMR
SAMPLE TYPE		PLASMA	PLASMA	SERUM	SERUM	URINE	CAECAL	SERUM	SALIVA	STOOL	URINE
SAMPLE SIZE		968 (485/483)	253 (142/111)	668 (337/331)	59 (34/25)	184 (101/83)	97 (46/51)	80 (49/31)	100 (50/50)	121 (68/53)	83 (43/40)
NO. OF PEAKS		189	138	689	29	120	42	181	69	747	149
PLS-DA	TRAIN (IB)	0.88 (0.85, 0.91)	0.90 (0.84, 0.96)	0.93 (0.91, 0.96)	0.98 (0.87, 1.00)	0.98 (0.97, 1.00)	0.96 (0.92, 0.99)	0.96 (0.90, 1.00)	1.00 (0.99, 1.00)	0.85 (0.77, 0.93)	0.97 (0.94, 1.00)
	TEST (OOB)	0.83 (0.78, 0.89)	0.77 (0.64, 0.88)	0.76 (0.69, 0.84)	0.75 (0.52, 0.98)	0.92 (0.83, 0.99)	0.89 (0.79, 0.99)	0.76 (0.56, 0.95)	0.96 (0.82, 1.00)	0.77 (0.58, 0.91)	0.89 (0.76, 0.99)
PCR	TRAIN (IB)	0.86 (0.82, 0.89)	0.88 (0.83, 0.95)	0.92 (0.88, 0.95)	0.97 (0.81, 0.99)	0.97 (0.93, 0.99)	0.92 (0.82, 0.99)	0.89 (0.79, 0.98)	0.99 (0.96, 1.00)	0.79 (0.65, 0.88)	0.95 (0.77, 0.97)
	TEST (OOB)	0.82 (0.77, 0.88)	0.79 (0.63, 0.88)	0.75 (0.68, 0.83)	0.78 (0.48, 0.96)	0.91 (0.80, 0.98)	0.86 (0.71, 0.99)	0.76 (0.51, 0.92)	0.92 (0.74, 0.99)	0.74 (0.56, 0.88)	0.81 (0.69, 0.98)
PCLR	TRAIN (IB)	0.87 (0.82, 0.90)	0.98 (0.95, 1.00)	0.92 (0.87, 0.94)	0.94 (0.73, 0.98)	0.97 (0.91, 0.99)	0.95 (0.86, 1.00)	0.90 (0.81, 0.99)	0.99 (0.97, 1.00)	0.79 (0.65, 0.89)	0.94 (0.75, 0.98)
	TEST (OOB)	0.82 (0.79, 0.89)	0.82 (0.70, 0.91)	0.74 (0.68, 0.83)	0.73 (0.46, 0.94)	0.91 (0.80, 0.97)	0.85 (0.71, 0.99)	0.74 (0.54, 0.94)	0.89 (0.73, 0.99)	0.75 (0.60, 0.90)	0.81 (0.65, 1.00)
RF	TRAIN (IB)	0.91 (0.90, 0.94)	0.90 (0.87, 0.96)	0.97 (0.96, 0.99)	0.97 (0.85, 0.99)	0.95 (0.93,0.99)	1.00 (0.98, 1.00)	0.99 (0.93, 1.00)	0.99 (0.95, 1.00)	0.95 (0.89, 0.98)	0.98 (0.96, 1.00)
	TEST (OOB)	0.82 (0.76, 0.88)	0.71 (0.58, 0.83)	0.71 (0.63, 0.78)	0.70 (0.47, 0.98)	0.80 (0.67, 0.92)	0.91 (0.76, 1.00)	0.71 (0.42, 0.87)	0.81 (0.55, 0.95)	0.73 (0.56, 0.91)	0.87 (0.67, 0.98)
SVM-Lin	TRAIN (IB)	0.86 (0.84, 0.91)	0.97 (0.94, 1.00)	0.90 (0.90, 0.95)	0.98 (0.87, 1.00)	1.00 (0.99, 1.00)	1.00 (0.98, 1.00)	0.95 (0.86, 0.99)	1.00 (0.98, 1.00)	0.89 (0.84, 0.97)	0.98 (0.96, 1.00)
	TEST (OOB)	0.84 (0.79, 0.89)	0.82 (0.69, 0.92)	0.75 (0.70, 0.84)	0.77 (0.55, 0.98)	0.94 (0.87, 1.00)	0.91 (0.82, 1.00)	0.80 (0.51, 0.92)	0.96 (0.85, 1.00)	0.77 (0.61, 0.90)	0.89 (0.78, 0.99)
SVM-RBF	TRAIN (IB)	0.88 (0.88, 0.91)	0.97 (0.94, 0.99)	0.98 (0.98, 0.99)	0.98 (0.88, 1.00)	0.99 (0.99, 1.00)	0.99 (0.97, 1.00)	0.99 (0.97, 1.00)	1.00 (1.00, 1.00)	0.90 (0.87, 0.98)	0.98 (0.97, 1.00)
	TEST (OOB)	0.84 (0.81, 0.88)	0.81 (0.68, 0.91)	0.78 (0.76, 0.84)	0.78 (0.61, 0.98)	0.95 (0.85, 0.99)	0.90 (0.82, 1.00)	0.80 (0.55, 0.93)	0.96 (0.84, 1.00)	0.78 (0.59, 0.90)	0.91 (0.77, 0.99)
ANN-LS	TRAIN (IB)	0.89 (0.85, 0.92)	0.97 (0.92, 0.98)	0.97 (0.96, 0.99)	0.98 (0.91, 1.00)	0.99 (0.97, 1.00)	1.00 (0.95, 1.00)	0.98 (0.91, 1.00)	1.00 (1.00, 1.00)	0.91 (0.81, 0.96)	0.98 (0.96, 1.00)
	TEST (OOB)	0.82 (0.77, 0.88)	0.77 (0.69, 0.90)	0.74 (0.70, 0.83)	0.75 (0.57, 0.96)	0.94 (0.78, 0.98)	0.92 (0.73, 0.99)	0.75 (0.45, 0.88)	0.94 (0.83, 1.00)	0.75 (0.50, 0.85)	0.86 (0.70, 0.97)
ANN-SS	TRAIN (IB)	0.89 (0.85, 0.92)	0.98 (0.92, 0.99)	0.97 (0.94, 0.98)	0.99 (0.95, 1.00)	1.00 (0.99, 1.00)	1.00 (0.96, 1.00)	0.98 (0.95, 1.00)	1.00 (0.99, 1.00)	0.93 (0.84, 0.98)	0.99 (0.95, 1.00)
	TEST (OOB)	0.82 (0.79, 0.89)	0.79 (0.69, 0.89)	0.78 (0.72, 0.86)	0.76 (0.58, 0.99)	0.95 (0.83, 0.99)	0.95 (0.79, 1.00)	0.80 (0.56, 0.92)	0.96 (0.81, 1.00)	0.76 (0.59, 0.90)	0.90 (0.72, 0.98)

Figure 4.2: Bootstrap Model Performance. Training and test area under the Receiver Operator Characteristic curve (95% in-bag and out-of-bag bootstrap confidence intervals) for the complete matrix of datasets and machine learning methods.

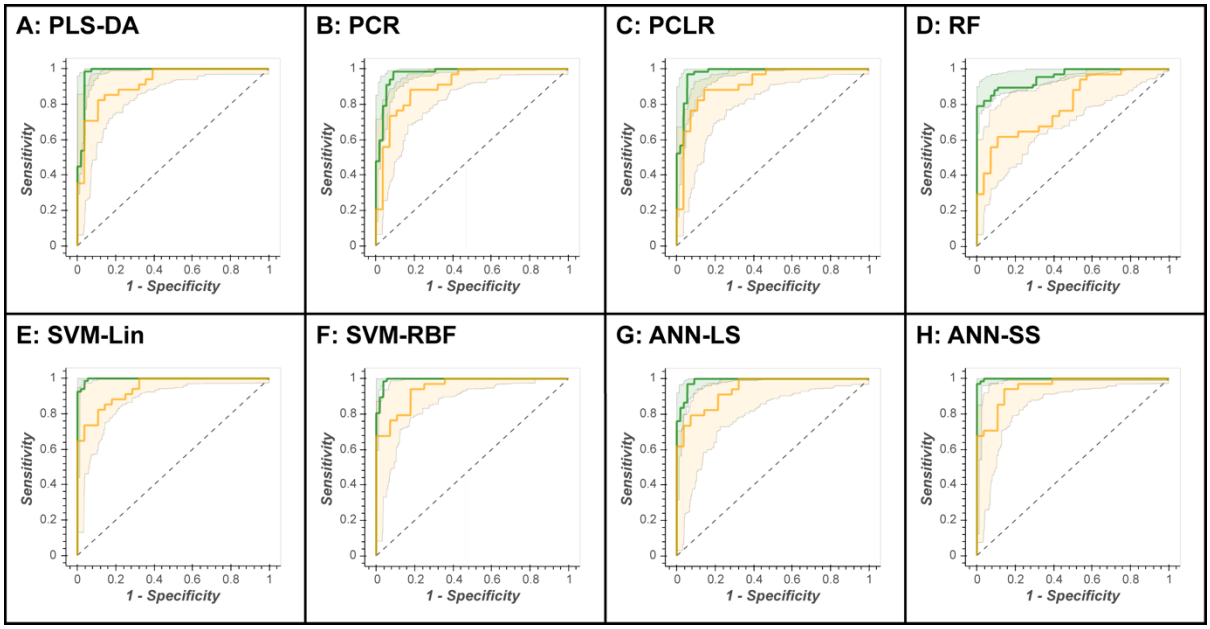


Figure 4.3: Illustration of the similarity of test prediction across all ML algorithms. The complete set Receiver Operator Characteristic curves for Data Set MTBLS404. Green line=ROC_{train}, green shading=in-bag 95% confidence interval, yellow line=ROC_{test}, yellow shading=out-of-bag 95% confidence interval. This resulted in: **a** AUC_{test} = 0.92; **b** AUC_{test} = 0.91; **c** AUC_{test} = 0.91; **d** AUC_{test} = 0.80; **e** AUC_{test} = 0.94; **f** AUC_{test} = 0.95; **g** AUC_{test} = 0.94; **h** AUC_{test} = 0.95.

If a single ML method is compared across multiple data sets, there is an observable inverse correlation between sample size and OOB 95% confidence interval (the fewer the samples the broader the confidence interval). This is illustrated in Fig. 4.4, where the ANN-SS ROC curves are presented for 3 different size data sets (n = 968, n = 235, and n = 83). Note that there is no observed correlation between performance and the number of metabolites modelled.

4.4. Discussion

The primary hypothesis of this study was that for binary classification using metabolomics data, non-linear machine learning methods would provide superior generalised predictive ability when compared to linear alternatives, in particular when compared with the current gold standard partial least squares discriminant analysis (PLS-DA). Based on the ten data sets

curated for this study, and the eight chosen machine learning methods, this primary hypothesis was disproved. Although support vector machines using a non-linear radial basis function kernel (SVM-RBF) and the fully sigmoidal feed-forward artificial neural network (ANN-SS) proved to be superior for all compared data sets with respect to AUC_{test}, the difference in performance against their linear counterparts (ANN-LS and SVM-Lin) and PLS-DA was marginal once generalised confidence intervals were calculated. These results suggest that in general, for binary classification, metabolomics data is linearly separable, particularly when projected into a latent space. There is no need for the "kernel trick" described in Sect. 4.2.2.5. The poor overall performance of random forests (RFs) will be surprising to some, given claims that RFs cannot overfit. However, as Hastie *et al.* (2009) prove "when the number of variables is large, but the fraction of relevant variables small, random forests are likely to perform poorly with small m [number of samples]". The inherent covariance in metabolomics data, which is an advantage to projection methods, hold no advantage for the random feature selection and data splitting performed by RF.

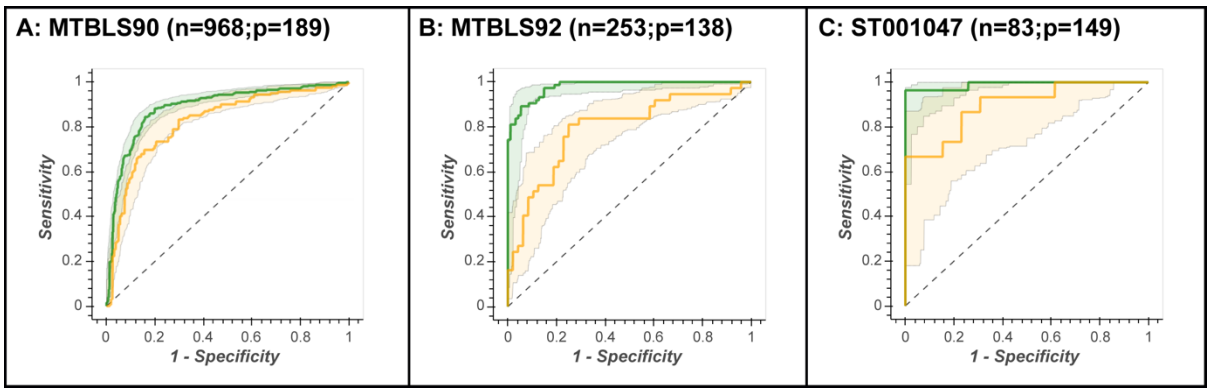


Figure 4.4: Inverse correlation between sample size and confidence intervals of models. SVM-RBF Receiver Operator Characteristic curves for three different size data sets (n=968, n=235, and n=83). Green line=ROC_{train}, green shading=in-bag 95% confidence interval, yellow line=ROC_{test} yellow shading=out-of-bag 95% confidence interval.

A second important observation from this study was that, despite standard k-fold cross-validation for optimisation, every model overtrained, and that the more complex the ML method the more severe the overtraining. This will be unsurprising to experts in the field, but it is worth noting. This is most strikingly observed in Fig. 4.2. The performance metrics (AUC_{train}& AUC_{test}) for each model/data pair should, if not overtrained, be of the same value (same hue of blue in Fig. 4.3). Clearly, for several data sets the RF, SVM-RBF and ANN-SS are severely overtrained (reflected in differences between AUC_{train} and AUC_{test} of up to 25%). This is further illustrated in Fig. 4.3, where the in-bag ROC curves showed AUC_{train} > 0.98 for the PLS-DA, SVM-Lin, SVM-RBF and ANN-SS models applied to data set MTBLS404, but AUC_{test} were more conservative (0.92–0.95). As such, it is imperative that an estimate of generalised predictive ability is presented alongside any published model, preferably using an independently measured test data set or alternatively a methodology similar to the train/test or out-of-bag bootstrap method described herein. It is misleading to only present the confidence interval for the training data as a measure generalised prediction.

Thirdly, it is important to discuss the utility of calculating the bootstrap confidence interval for each candidate model configuration for the applied data. When data sets are small and potentially heterogeneous (as often observed in clinical studies) the use of random data splitting (e.g. 2/3 training, 1/3 test) to provide an unbiased performance evaluation can be dangerous. For truly unbiased evaluation the test set must exactly represent the training data. This may not be possible by random methods (even when stratified by outcome). This is illustrated in Fig. 4.5 where, for data set ST001047, the random split is repeated 5 times with dramatically different performance for a PLS-DA model using two latent variables. The bootstrap resampling enables the modeller to estimate this uncertainty. It is worth noting that for all 80 of the models presented in this paper the ROCtest curve lay within the bounds of the respective

OOB 95% confidence interval (see supplementary notebooks). Even so, such bootstrapping provides only an estimate and care must be taken as there is a certain amount of data leakage as the same data that is being used to select the hyperparameters is being used to evaluate the model.

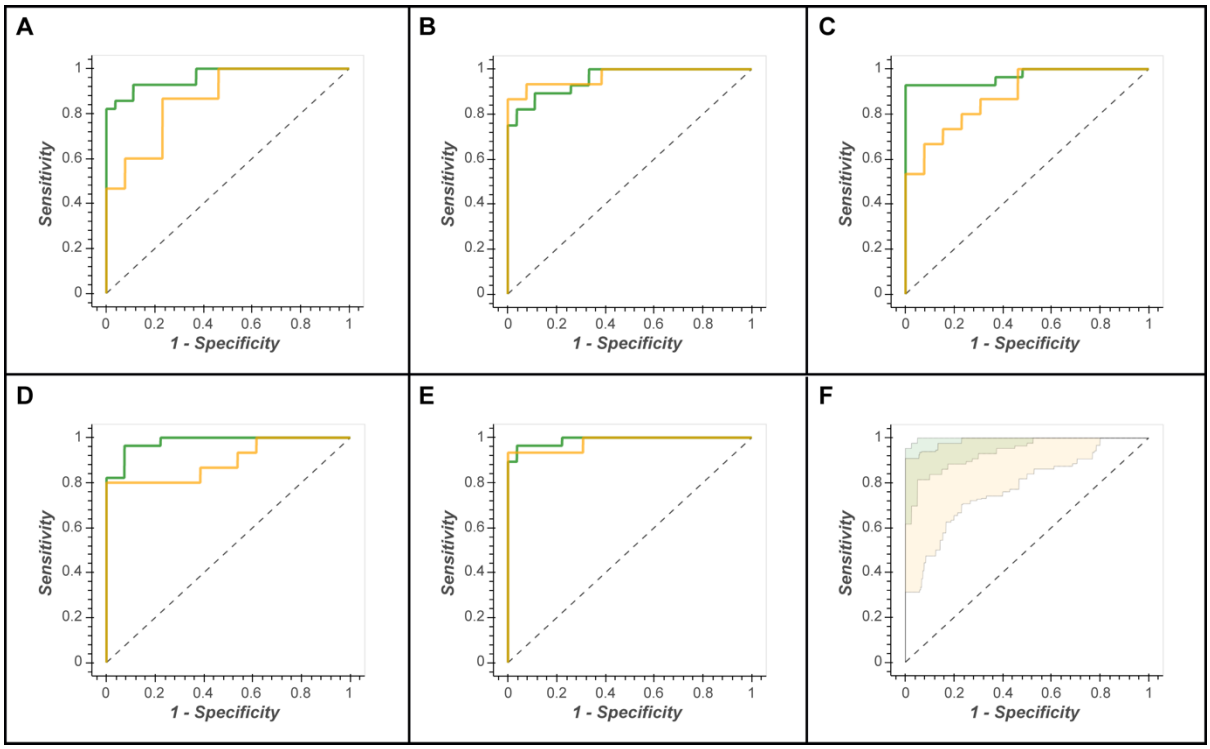


Figure 4.5: Prediction uncertainty when using train/test data splitting for validation. Receiver Operator Characteristic curves for training/ test performance of PLS-DA on data set ST001047 for five iterations of stratified random splitting (2/3 training and 1/3 test). Green line=ROC_{train}, yellow line=ROC_{test}. This resulted in: **a** AUC_{train}= 0.96, AUC_{test} = 0.87; **b** AUC_{train} = 0.97, AUC_{test} = 0.96; **c** AUC_{train}= 0.97, AUC_{test} = 0.88; **d** AUC_{train} = 0.98, AUC_{test} = 0.90; **e** AUC_{train}= 0.99, AUC_{test} = 0.98. **d** The 95% OOB confidence interval for the same data. Note all **a–e** ROC_{test} curves lie within the 95% confidence interval.

A final, but equally important, observation from this study was that the stability of a model was dependent on the number of samples available for training. This is best illustrated in Fig. 4.4. Here the generalised predictive ability of an ANN-SS model is compared across three data sets of increasing size. For data set ST001047 (n = 83) the out-of-bag ROC curves vary

dramatically from AUC_{OOB} = 0.75–0.98). This implies that the underlying model parameters varying massively due to heterogeneity of the in-bag training sets. Which leads to the question: Is the complete data set a representative sample of the biological question? (in this case classifying gastric cancer). This phenomena, known as the Rashomon Effect, has been discussed at length by Breiman (2001b), Broadhurst & Kell (2006) and Broadhurst (2017). In contrast, data set MTBLS90 (n = 968) has extremely stable out-of-bag ROC curves implying that there is sufficient data to robustly model the biological question.

4.5. Limitations of the Study

While the results of this study will hopefully prove useful to the metabolomics research community, it is important to list some limitations. Firstly, focusing on binary classification we may have oversimplified the problem space. Non-linear ML methods may be more effective in multi-class problems, so results need to be interpreted with this in mind. Secondly, by focussing on published data there is a possibility that the results are biased (publication bias). All the data set used in this study were successfully published using a linear model. Given that, generally, only positive results are published it may be that, despite our best efforts, we did not have access to data sufficiently complex to require a non-linear model. Finally, the ML algorithms with more than two hyperparameters (i.e. ANN and RF) are presented in the Jupyter notebooks such that we limit the search strategy to a grid search of the two most sensitive hyperparameters, fixing the other hyperparameters at a constant value. A full parameter search was performed for each individual model under cross-validation conditions, and repeatedly the same hyperparameters had little effect on optimisation, so for clarity of presentation they were fixed at the same value across all data sets in the Jupyter notebooks provided. Interested readers are encouraged to download the data and notebooks and verify our findings.

4.6. Conclusions

In this study of binary classification across ten publicly available metabolomics datasets we have shown that using non-linear machine learning showed no general improvement in predictability over linear methods. If we use the principle of Occam's razor, where the simplest model wins out, PLS-DA remains a sensible first choice. However, improved computational power and open availability of high-quality software libraries means that comparing multiple models of a given data set is tractable. Our results clearly demonstrate that of equal importance to the choice of machine learning method is the way that each method is optimised, and how its generalised performance is evaluated. It is far too easy to overtrain a complex model and erroneously report misleading results. We have provided a generalised framework to investigate eight machine learning algorithms and a generalised optimisation and evaluation workflow that can be applied to any multivariate data with a binary outcome variable.

The likely most important conclusion from this study is a reiteration of the well-established machine learning trope a model is only as good as the data that is used to train it. We consider the 10 datasets used in this study are representative, both in sample size and scope, of biomarker studies published in metabolomics. The results presented here suggest that for robust predictive models the most important consideration is statistical power. There is no magic formula for calculating the number of samples needed for robust metabolomics multivariate machine learning, where estimates are dependent on many factors, including: the dimensionality of the data, the strength of effect, the degree of covariance (strength of latent structure), the heterogeneity of the sample population, the repeatability of the measurement instrument, and the complexity of the model. However, as pointed out by Breiman (2001b), the curse of dimensionality dictates that the expected generalization error is proportional to the complexity

of the model and inversely proportional to the number of samples used to build the model. Thus, for high dimensional data a complex model trained on a small data set will tend to have poor generalised performance as a classifier. Put simply, the larger and better curated (cleaned and identified) the data set, the more amenable it will be to non-linear machine learning algorithms.

4.7. Future Perspectives

In order for machine learning to have a meaningful impact on metabolomics then larger data sets need to be collated, and those data have to be pass stringent quality control checks (Broadhurst *et al.*, 2018). It is important to note that an increasing number of metabolomics researchers, particularly in the clinical domain, outsource metabolomics data acquisition. Companies such as Metabolon (https://www.metabolon.com/), Nightingale Health (https://nightingalehealth.com/), and Biocrates (https://www.biocrates.com/) have built business models that depend on providing high-quality fully annotated data sets in a format amenable for data science. Most large academic laboratories also provide some level of similar service. This is illustrated by the recent successful ring trial for the Biocrates AbsoluteIDQ p400HR assay (Thompson *et al.*, 2019) which will allow data sets from multiple labs to be potentially combined into one data analysis. Other approaches to data fusion have most recently been reported in the American Journal of Epidemiology by Yu *et al.* (2019) "Consortium of Metabolomics Studies (COMETS) Metabolomics in 47 Prospective Cohort Studies".

As machine learning methods get more complex the demands for data get greater. The recent successes of deep learning in image processing, peak deconvolution and metabolite identification (Mendez *et al.*, 2019a) means it is likely that such methods will also be applied

to predictive modelling. As a community it is important that mechanisms are put in place to avoid over optimistic reporting of results, and that it is not simply assumed that a complex model is the best model. There is an urgent need for transparent and consistent reporting of all aspects of the metabolomics study lifecycle. The metabolomics community has made substantial efforts to align with FAIR (Findable, Accessible, Interoperable, and Reusable) data principles by utilizing open data formats [e.g. mzXML (Pedrioli *et al.*, 2004)], developing data repositories [e.g. MetaboLights (Haug *et al.*, 2012) and Metabolomics Workbench (Sud *et al.*, 2015)], and with online spectral reference [e.g. METLIN (Smith *et al.*, 2005), mzCloud (https://www.mzcloud.org/), MassBank (Horai *et al.*, 2010), GNPS (Wang *et al.*, 2016)], and online databases for metabolite identification and biochemical association [e.g. HMDB (Wishart *et al.*, 2017)]. However, significant efforts are required to find ways to make metabolomics data modelling FAIR. One such approach is through Jupyter notebooks (Mendez *et al.*, 2019b). Hopefully, the 80 Jupyter notebooks provided for this study will help inspire more open reporting of predictive modelling in metabolomics (https://cimcb.github.io/MetabComparisonBinaryML).

Acknowledgments

This work was partly funded through an Australian Research Council funded LIEF Grant (LE170100021).

Author Contributions

KMM, SNR & DIB conceived of the idea. KMM developed the Jupyter notebooks. KMM developed the Python packages. KMM performed the optimisation and evaluation of all models. DIB & SNR independently checked each model. KMM wrote the manuscript. DIB and SNR edited the manuscript.

Data Availability

The metabolomics and metadata used in this paper were retrieved from Metabolights (https://www.ebi.ac.uk/metabolights/) Project IDs: MTBLS90 MTBLS92 MTBLS136 MTBLS161 MTBLS404 MTBLS547 and Metabolomics Workbench (https://www.metabolomicsworkbench.org/) Project IDs: ST000369 ST000496 ST001000 ST001047. This data were converted from the original data format to a clean format compliant with the Tidy Data framework, this is available at the CIMCB GitHub project page: https://github.com/CIMCB/MetabComparisonBinaryML.

Software Availability

All software developed for this paper is available at the CIMCB GitHub project page: https://github.com/CIMCB/MetabComparisonBinaryML.

Conflict of Interest

The authors have no disclosures of potential conflicts of interest related to the presented work.

Research Involving Human or Animal Rights

No research involving human or animal participants was performed in the construction of this manuscript.

Open Access

References

Bishop, C.M. (1995) *Neural networks for pattern recognition*. Oxford University Press, New York, United States of America.

Blei, D.M. and Smyth, P. (2017) Science and data science. *Proceedings of the National Academy of Sciences of the United States of America* **114,** 8689-8692.

Bokeh Development Team (2018). Bokeh: Python library for interactive visualization. https://bokeh.pydata.org/en/latest/

Breiman, L. (2001a) Random forests. *Machine learning* **45,** 5-32.

Breiman, L. (2001b) Statistical modeling: The two cultures. *Statistical Science* **16,** 199-231.

Breiman, L., Friedman, J., Stone, C.J. and Olshen, R.A. (1984) *Classification and Regression Trees*, 1st edn. Chapman & Hall - CRC, New York, United States of America.

Broadhurst, D. (2017) A clash of two cultures: The juxtaposition of biostatistics & machine learning in metabolomics data diagnostics item, *Metabomeeting December 2017, University of Birmingham, UK*, https://doi.org/10.6084/m9.figshare.5696494.v3.

Broadhurst, D., Goodacre, R., Reinke, S.N., Kuligowski, J., Wilson, I.D., Lewis, M.R. and Dunn, W.B. (2018) Guidelines and considerations for the use of system suitability and quality control samples in mass spectrometry assays applied in untargeted clinical metabolomic studies. *Metabolomics* **14,** 72.

Broadhurst, D.I. and Kell, D.B. (2006) Statistical strategies for avoiding false discoveries in metabolomics and related experiments. *Metabolomics* **2,** 171-196.

Broomhead, D.S. and Lowe, D. (1988) Radial basis functions, multi-variable functional interpolation and adaptive networks. *Technical report, Royal Signals and Radar Establishment Malvern (United Kingdom)*.

Chollet, F. (2015). Keras. https://keras.io/ Accessed August 27 2019

de Jong, S. (1993) SIMPLS: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems* **18,** 251-263.

DiCiccio, T.J. and Efron, B. (1996) Bootstrap confidence intervals. *Statistical Science* **11,** 189-212.

Dunn, W.B., Broadhurst, D.I., Atherton, H.J., Goodacre, R. and Griffin, J.L. (2011) Systems level studies of mammalian metabolomes: the roles of mass spectrometry and nuclear magnetic resonance spectroscopy. *Chemical Society Reviews* **40,** 387-426.

Efron, B. (2000) The bootstrap and modern statistics. *Journal of the American Statistical Association* **95,** 1293-1296.

Eriksson, L., Byrne, T., Johansson, E., Trygg, J. and Vikström, C. (2013) *Multi- and megavariate data analysis: basic principles and applications*, 3rd edn. Umetrics Academy, Malmö, Sweden.

Gromski, P.S., Muhamadali, H., Ellis, D.I., Xu, Y., Correa, E., Turner, M.L. and Goodacre, R. (2015) A tutorial review: Metabolomics and partial least squares-discriminant analysis--a marriage of convenience or a shotgun wedding. *Analytica Chimica Acta* **879,** 10-23.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning*, 2nd edn. Springer, New York, United States of America.

Haug, K., Salek, R.M., Conesa, P., Hastings, J., de Matos, P., Rijnbeek, M., Mahendraker, T., Williams, M., Neumann, S., Rocca-Serra, P., Maguire, E., González-Beltrán, A., Sansone, S.-A., Griffin, J.L. and Steinbeck, C. (2012) MetaboLights—an open-access general-purpose repository for metabolomics studies and associated meta-data. *Nucleic Acids Research* **41,** D781-D786.

Horai, H., Arita, M., Kanaya, S., Nihei, Y., Ikeda, T., Suwa, K., Ojima, Y., Tanaka, K., Tanaka, S., Aoshima, K., Oda, Y., Kakazu, Y., Kusano, M., Tohge, T., Matsuda, F., Sawada, Y., Hirai, M.Y., Nakanishi, H., Ikeda, K., Akimoto, N., Maoka, T., Takahashi, H., Ara, T., Sakurai, N., Suzuki, H., Shibata, D., Neumann, S., Iida, T., Tanaka, K., Funatsu, K., Matsuura, F., Soga, T., Taguchi, R., Saito, K. and Nishioka, T. (2010) MassBank: a public repository for sharing mass spectral data for life sciences. *Journal of Mass Spectrometry* **45,** 703-714.

Jolliffe, I.T. (1982) A note on the use of principal components in regression. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **31,** 300-303.

Jolliffe, I.T. (2002) *Principal Component Analysis*, 2nd edn. Springer, New York, United States of America.

Kristensen, M.R.B. and Vinter, B. (2010) Numerical Python for scalable architectures, *Proceedings of the Fourth Conference on Partitioned Global Address Space Programming Model*, Association for Computing Machinery, pp. 1-9.

McKinney, W. (2010) Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference***,** 51-56.

Menard, S. (2002) *Applied Logistic Regression Analysis*, 2nd edn. SAGE Publications, California, United States of America.

Mendez, K.M., Broadhurst, D.I. and Reinke, S.N. (2019a) The application of artificial neural networks in metabolomics: a historical perspective. *Metabolomics* **15,** 142.

Mendez, K.M., Pritchard, L., Reinke, S.N. and Broadhurst, D.I. (2019b) Toward collaborative open data science in metabolomics using Jupyter Notebooks and cloud computing. *Metabolomics* **15,** 125.

Miettinen, K. (1999) *Nonlinear Multiobjective Optimization*. Springer, New York, United States of America.

Mosconi, F., Julou, T., Desprat, N., Sinha, D.K., Allemand, J.-F., Croquette, V. and Bensimon, D. (2008) Some nonlinear challenges in biology. *Nonlinearity* **21,** 131-147.

Park, J. and Sandberg, I.W. (1991) Universal Approximation Using Radial-Basis-Function Networks. *Neural Computation* **3,** 246-257.

Pedregosa, F., Ga, Varoquaux, l., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, D. (2011) Scikit-learn: Machine Learning in Python. *The Journal of Machine Learning Research* **12,** 2825-2830.

Pedrioli, P.G.A., Eng, J.K., Hubley, R., Vogelzang, M., Deutsch, E.W., Raught, B., Pratt, B., Nilsson, E., Angeletti, R.H., Apweiler, R., Cheung, K., Costello, C.E., Hermjakob, H., Huang, S., Julian, R.K., Kapp, E., McComb, M.E., Oliver, S.G., Omenn, G., Paton, N.W., Simpson, R., Smith, R., Taylor, C.F., Zhu, W. and Aebersold, R. (2004) A common open representation of mass spectrometry data and its application to proteomics research. *Nature Biotechnology* **22,** 1459-1466.

Schölkopf, B. and Smola, A.J. (2001) *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, Massachusetts, United States of America.

Seber, G.A.F. (2004) *Multivariate Observations*, 2nd edn. Wiley, New Jersey, United States of America.

Smith, C.A., Maille, G.O., Want, E.J., Qin, C., Trauger, S.A., Brandon, T.R., Custodio, D.E., Abagyan, R. and Siuzdak, G. (2005) METLIN: A metabolite mass spectral database. *Therapeutic Drug Monitoring* **27,** 747-751.

Steinwart, I. and Christmann, A. (2008) *Support Vector Machines*. Springer, New York, United States of America.

Sud, M., Fahy, E., Cotter, D., Azam, K., Vadivelu, I., Burant, C., Edison, A., Fiehn, O., Higashi, R., Nair, K.S., Sumner, S. and Subramaniam, S. (2015) Metabolomics Workbench: An international repository for metabolomics data and metadata, metabolite standards, protocols, tutorials and training, and analysis tools. *Nucleic Acids Research* **44,** D463-D470.

Szymańska, E., Saccenti, E., Smilde, A.K. and Westerhuis, J.A. (2012) Double-check: validation of diagnostic statistics for PLS-DA models in metabolomics studies. *Metabolomics* **8,** 3-16.

Theano Development Team (2016) Theano: A python framework for fast computation of mathematical expressions. *arXiv:1605.02688*.

Thompson, J.W., Adams, K.J., Adamski, J., Asad, Y., Borts, D., Bowden, J.A., Byram, G., Dang, V., Dunn, W.B., Fernandez, F., Fiehn, O., Gaul, D.A., Hühmer, A.F.R., Kalli, A., Koal, T., Koeniger, S., Mandal, R., Meier, F., Naser, F.J., O'Neil, D., Pal, A., Patti, G.J., Pham-Tuan, H., Prehn, C., Raynaud, F.I., Shen, T., Southam, A.D., St. John-

Williams, L., Sulek, K., Vasilopoulou, C.G., Viant, M., Winder, C.L., Wishart, D., Zhang, L., Zheng, J. and Moseley, M.A. (2019) International Ring Trial of a High Resolution Targeted Metabolomics and Lipidomics Platform for Serum and Plasma Analysis. *Analytical Chemistry* **91,** 14407-14416.

Wang, M., Carver, J.J., Phelan, V.V., Sanchez, L.M., Garg, N., Peng, Y., Nguyen, D.D., Watrous, J., Kapono, C.A., Luzzatto-Knaan, T., Porto, C., Bouslimani, A., Melnik, A.V., Meehan, M.J., Liu, W.-T., Crüsemann, M., Boudreau, P.D., Esquenazi, E., Sandoval-Calderón, M., Kersten, R.D., Pace, L.A., Quinn, R.A., Duncan, K.R., Hsu, C.-C., Floros, D.J., Gavilan, R.G., Kleigrewe, K., Northen, T., Dutton, R.J., Parrot, D., Carlson, E.E., Aigle, B., Michelsen, C.F., Jelsbak, L., Sohlenkamp, C., Pevzner, P., Edlund, A., McLean, J., Piel, J., Murphy, B.T., Gerwick, L., Liaw, C.-C., Yang, Y.-L., Humpf, H.-U., Maansson, M., Keyzers, R.A., Sims, A.C., Johnson, A.R., Sidebottom, A.M., Sedio, B.E., Klitgaard, A., Larson, C.B., Boya P, C.A., Torres-Mendoza, D., Gonzalez, D.J., Silva, D.B., Marques, L.M., Demarque, D.P., Pociute, E., O'Neill, E.C., Briand, E., Helfrich, E.J.N., Granatosky, E.A., Glukhov, E., Ryffel, F., Houson, H., Mohimani, H., Kharbush, J.J., Zeng, Y., Vorholt, J.A., Kurita, K.L., Charusanti, P., McPhail, K.L., Nielsen, K.F., Vuong, L., Elfeki, M., Traxler, M.F., Engene, N., Koyama, N., Vining, O.B., Baric, R., Silva, R.R., Mascuch, S.J., Tomasi, S., Jenkins, S., Macherla, V., Hoffman, T., Agarwal, V., Williams, P.G., Dai, J., Neupane, R., Gurr, J., Rodríguez, A.M.C., Lamsa, A., Zhang, C., Dorrestein, K., Duggan, B.M., Almaliti, J., Allard, P.-M., Phapale, P. *et al.* (2016) Sharing and community curation of mass spectrometry data with global natural products social molecular networking. *Nature Biotechnology* **34,** 828-837.

Wickham, H. (2014) Tidy data. *Journal of Statistical Software* **59,** 1-23.

Wilkins, M.F., Morris, C.W. and Boddy, L. (1994) A comparison of radial basis function and backpropagation neural networks for identification of marine phytoplankton from multivariate flow cytometry data. *Computer Applications in the Biosciences* **10,** 285-94.

Wishart, D.S., Feunang, Y.D., Marcu, A., Guo, A.C., Liang, K., Vázquez-Fresno, R., Sajed, T., Johnson, D., Li, C., Karu, N., Sayeeda, Z., Lo, E., Assempour, N., Berjanskii, M., Singhal, S., Arndt, D., Liang, Y., Badran, H., Grant, J., Serra-Cayuela, A., Liu, Y., Mandal, R., Neveu, V., Pon, A., Knox, C., Wilson, M., Manach, C. and Scalbert, A. (2017) HMDB 4.0: the human metabolome database for 2018. *Nucleic Acids Research* **46,** D608-D617.

Wold, H. (1975) Path models with latent variables: The NIPALS approach, *Quantitative sociology*, Elsevier. 307-357.

Wold, S., Johansson, E. and Cocchi, M. (1993) PLS: Partial least squares projections to latent structures, *3D QSAR in Drug Design: Theory, Methods and Applications.*, Kluwer/Escom, Dordrecht, The Netherlands.

Xia, J., Broadhurst, D.I., Wilson, M. and Wishart, D.S. (2013) Translational biomarker discovery in clinical metabolomics: an introductory tutorial. *Metabolomics* **9,** 280-299.

Yu, B., Zanetti, K.A., Temprosa, M., Albanes, D., Appel, N., Barrera, C.B., Ben-Shlomo, Y., Boerwinkle, E., Casas, J.P., Clish, C., Dale, C., Dehghan, A., Derkach, A., Eliassen, A.H., Elliott, P., Fahy, E., Gieger, C., Gunter, M.J., Harada, S., Harris, T., Herr, D.R., Herrington, D., Hirschhorn, J.N., Hoover, E., Hsing, A.W., Johansson, M., Kelly, R.S., Khoo, C.M., Kivimäki, M., Kristal, B.S., Langenberg, C., Lasky-Su, J., Lawlor, D.A., Lotta, L.A., Mangino, M., Le Marchand, L., Mathé, E., Matthews, C.E., Menni, C., Mucci, L.A., Murphy, R., Oresic, M., Orwoll, E., Ose, J., Pereira, A.C., Playdon, M.C., Poston, L., Price, J., Qi, Q., Rexrode, K., Risch, A., Sampson, J., Seow, W.J., Sesso, H.D., Shah, S.H., Shu, X.-O., Smith, G.C.S., Sovio, U., Stevens, V.L., Stolzenberg-Solomon, R., Takebayashi, T., Tillin, T., Travis, R., Tzoulaki, I., Ulrich, C.M., Vasan, R.S., Verma, M., Wang, Y., Wareham, N.J., Wong, A., Younes, N., Zhao, H., Zheng, W. and Moore, S.C. (2019) The consortium of metabolomics studies (COMETS): Metabolomics in 47 prospective cohort studies. *American Journal of Epidemiology* **188,** 991-1012.

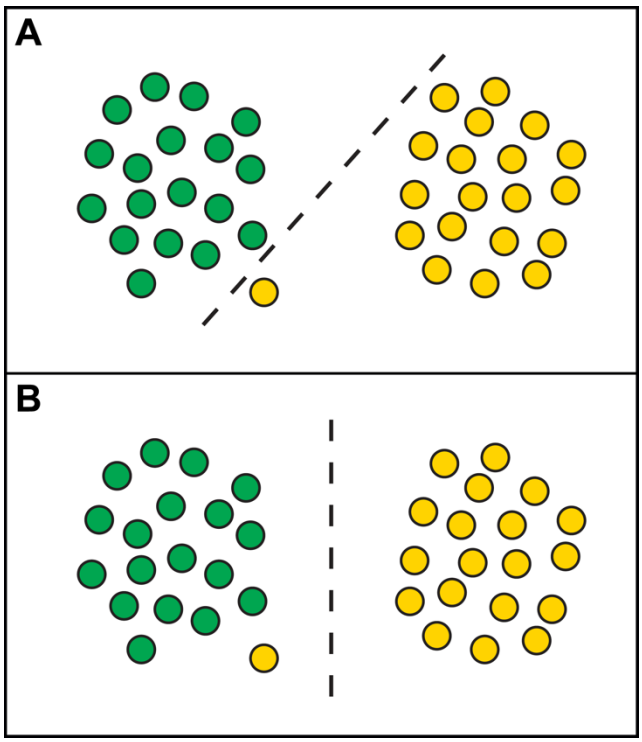
Supplementary Information

List of supplementary html files:

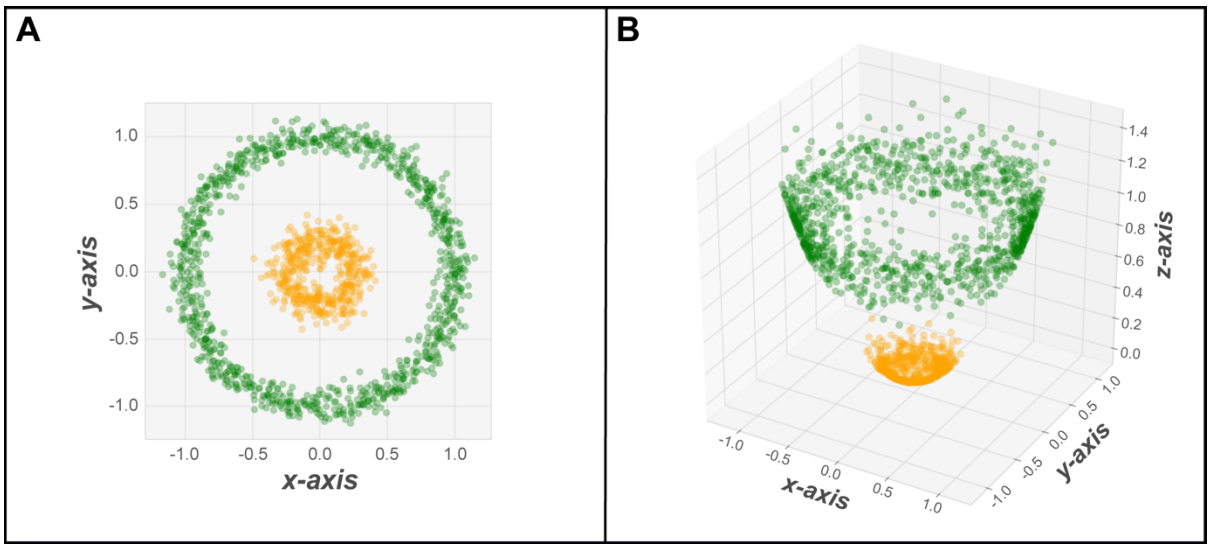
- 80 html files as [Model]_[DataSet].html
    - Model: ANNLinSig; ANNSigSig; PCLR; PCR; PLSDA; RF; SVMLin; SVMRBF
    - Dataset: MTBLS136; MTBLS161; MTBLS404; MTBLS547; MTBLS90; MTBLS92; ST000369; ST000496; ST001000; ST001047
    - https://github.com/CIMCB/MetabComparisonBinaryML/zipball/master (html folder)

List of supplementary figures:

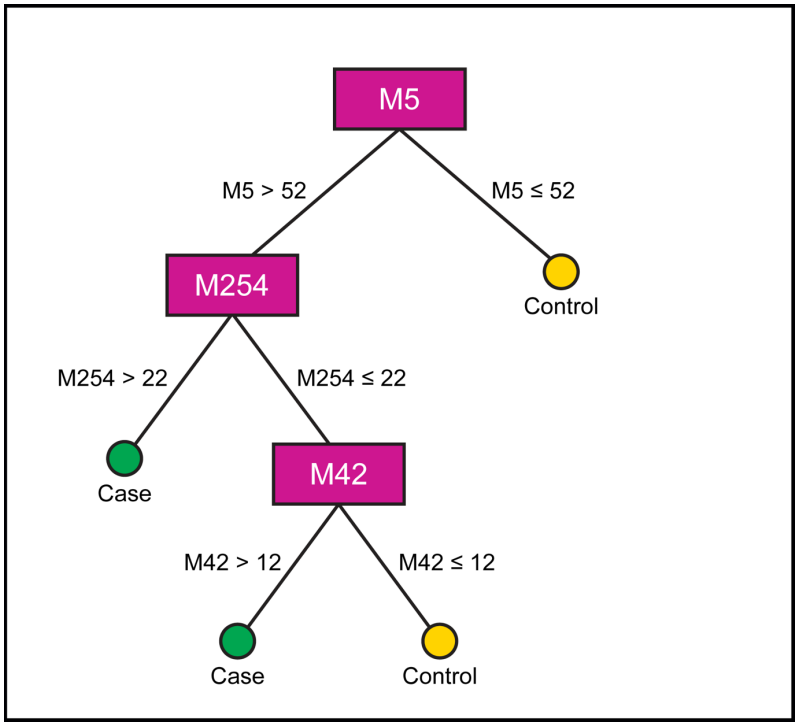
- Supplementary Fig 4.1
- Supplementary Fig 4.2
- Supplementary Fig 4.3
- Supplementary Fig 4.4



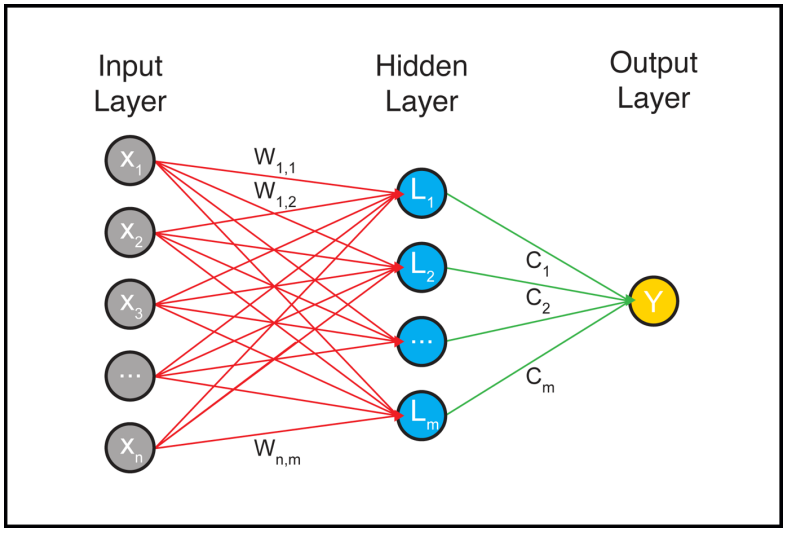
Supplementary Figure 4.1: Illustration of how the regularization parameter, C, in support vector machine (SVM) optimisation allows some flexibility regarding the number of misclassifications made by the hyperplane margin. For a large value of C (**a**), the SVM will choose a small margin for the hyperplane if that hyperplane does a better job of getting all the training points classified correctly (hard margin). Conversely, a small value of C (**b**) will cause the SVM to optimise to a larger margin separating hyperplane, even if that hyperplane misclassifies more points (soft margin).



Supplementary Figure 4.2: SVMs can be configured to perform non-linear classification by implicitly mapping input data into a high-dimensional feature space. This process is known as the *kernel trick*. The idea is to gain linearly separation by mapping the original data (**a**) to a higher dimensional space (**b**). The data is now linearly separable in the z-axis.



Supplementary Figure 4.3: A decision tree is top-down hierarchical structure of nodes connected by branches visualised as an inverted tree. Each node contains a logical question that sends a sample down one of two branches (a binary split), which in turn leads to another node, and on, and on, until it reaches a terminal node, which will provide a predicted classification. For example, to classify a new sample (say, based on a metabolite profile of 300 metabolites: $m_1 \ldots m_{300}$) we start at the *root node* and performs the split described therein (e.g. **if** $m_5 > 52$ **then** *Branch 1*, **else** *Branch 2*). Depending on the result we then descend the tree to the next *internal node* (e.g. **if** $m_{254} > 22$ **then** *Branch 3*, **else** *Branch 4*). Eventually we reach a *leaf node* at which time a classification is made (e.g. **if** $m_{42} > 12$ **then** *Case*, **else** *Control*).



Supplementary Figure 4.4: A two-layer ANN (Supplementary Figure 4) with a small number of linear neurons in the 1st layer (hidden layer) and a single linear neuron in the 2nd layer (output layer).

Chapter Five: Migrating from Partial Least Squares Discriminant Analysis to Artificial Neural Networks: A Comparison of Functionally Equivalent Visualisation and Feature Contribution Tools using Jupyter Notebooks.

Authors

Kevin M Mendez[1], David I Broadhurst[1*], Stacey N Reinke[1*]

[1]Centre for Integrative Metabolomics & Computational Biology, School of Science, Edith Cowan University, Joondalup, 6027 Australia

*Corresponding authors:

email: d.broadhurst@ecu.edu.au, stacey.n.reinke@ecu.edu.au

phone: +61 (0)8-6304-2705

This is a post-peer-review, pre-copyedit version of an article published in *Metabolomics*. The final authenticated version is available online at: https://doi.org/10.1007/s11306-020-1640-0.

Abstract

Introduction: Metabolomics data is commonly modelled multivariately using partial least squares discriminant analysis (PLS-DA). Its success is primarily due to ease of interpretation, through projection to latent structures, and transparent assessment of feature importance using regression coefficients and Variable Importance in Projection scores. In recent years several non-linear machine learning (ML) methods have grown in popularity but with limited uptake essentially due to convoluted optimisation and interpretation. Artificial neural networks (ANNs) are a non-linear projection-based ML method that share a structural equivalence with PLS, and as such should be amenable to equivalent optimisation and interpretation methods.

Objective: We hypothesise that standardised optimisation, visualisation, evaluation and statistical inference techniques commonly used by metabolomics researchers for PLS-DA can be migrated to a non-linear, single hidden layer, ANN.

Method: We compared a standardised optimisation, visualisation, evaluation and statistical inference techniques workflow for PLS with the proposed ANN workflow. Both workflows were implemented in the Python programming language. All code and results have been made publicly available as Jupyter notebooks on GitHub.

Results: The migration of the PLS workflow to a non-linear, single hidden layer, ANN was successful. There was a similarity in significant metabolites determined using PLS model coefficients and ANN Connection Weight Approach.

Conclusion: We have shown that it is possible to migrate the standardised PLS-DA workflow to simple non-linear ANNs. This result opens the door for more widespread use and to the investigation of transparent interpretation of more complex ANN architectures.

Keywords: Metabolomics, Partial Least Squares, Artificial Neural Networks, Machine Learning, Jupyter, Variable Importance in Projection.

5.1. Introduction

Within a biological system, metabolite concentrations are highly interdependent (Dunn *et al.*, 2011). As such, the usefulness of multivariate data analysis in metabolomics stems from the need to extract biological information from inherently complex covariant data, where metabolite interaction is as important as individual changes in concentration. Historically, partial least squares (PLS), a.k.a. projection to latent structures (Wold, 1975; Wold *et al.*, 1993), has been the standard multivariate machine learning (ML) method used to construct predictive models to classify metabolite profiles. The underlying theory of PLS, and its utility to metabolomics, has been documented many times (Geladi and Kowalski, 1986; Gromski *et al.*, 2015; Wold *et al.*, 1993; Wold *et al.*, 2001). A key benefit of PLS is the ability to visualise (via a latent variable score plot) the projected metabolomic relationship (clustering) between individual samples before classification.

There are many machine learning (ML) alternatives to PLS, several of which have been applied to metabolomics data. The most popular include support vector machines (Steinwart and Christmann, 2008), random forests (Breiman, 2001), and artificial neural networks (Bishop, 1995; Wilkins *et al.*, 1994); however, despite coexisting for a similar length of time, none of

these methods have gained the popularity of PLS. A survey of publications listed on the Web of Science using the keywords metabolite*, metabolom* or metabonom* reveals that up to and including 2018, 2,224 publications list the use of PLS as a key term, whereas the alternatives were listed < 500 times (combined number). The key to the popularity of PLS over alternative methods can be distilled into a single word - *interpretability*. Historically, the primary aim of machine learning (ML) has been accurate prediction, not statistical inference (Mendez *et al.*, 2019a). As such, methods for statistically interpreting either the similarities between each individual metabolite profile, or the importance of individual metabolites across multiple samples, have been a secondary consideration. The ability for PLS to visualise and infer statistical confidence intervals upon the latent relationships within and between sample classes, together with the fact that a PLS model can be reduced to a simple linear regression (and thus exposed to multiple well established post-hoc statistical tests), means that it sits alone as an effective hybrid prediction-inference algorithm for high dimensional data (Eriksson *et al.*, 2013; Wold, 1975; Wold *et al.*, 1993).

Artificial neural networks (ANNs) are also of particular interest because in their simplest form, as with PLS, they can be considered as a combination of dimensionality reduction and multiple linear regression. In fact, for a linear ANN, with a single hidden layer, the only difference between ANN and PLS is the manner in which the constituent model parameters are optimised (Fig. 5.1). ANNs can be generally considered a projection-based method which share a *structural equivalence* with PLS (Mendez *et al.*, 2019a). With non-linear ANNs the *projection to latent structures* ethos is preserved but now non-linear, rather than linear, latent structures can be modelled.

ANNs were first applied to metabolomic profiling ca. 1992 by Goodacre *el al.* (1992). At that time, due to lack of compute power and poor software availability, ANNs were very slow to train and considered difficult to interpret. As such, by the early 2000s they had been widely disregarded and relegated to an intellectual curiosity not considered able to provide meaningful biological insight (Goodacre, 2003). With recent advancements in computational power, the availability of easily accessible yet powerful open-source packages (e.g. TensorFlow and PyTorch), and the general success within industry and other research fields, the reintroduction of ANNs warrants renewed investigation. We recently showed that ANNs have similar predictive ability to PLS across multiple diverse metabolomics data sets (Mendez *et al.*, 2019c). However, within the domain of metabolomics, if ANNs are to become a truly viable alternative to PLS it will be necessary to develop similar standardised and robust methods for data visualisation, evaluation, and statistical inference (Mendez *et al.*, 2019a).

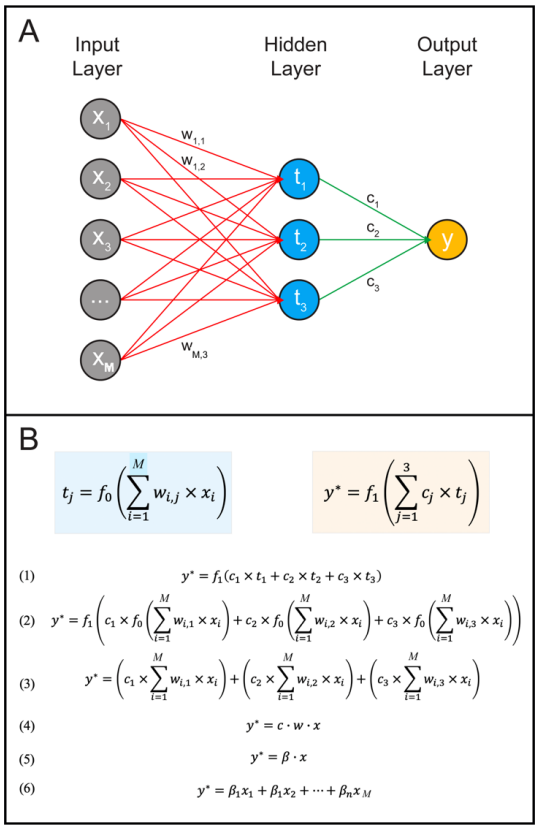


Figure 5.1: Illustration of an ANN as a regression model. **a** Network representation of a 2-layer ANN. **b** Representation of a 2-layer ANN with linear activation functions, as a set of equations, simplified to a linear regression model.

Recently, the increased availability of well curated open-source software libraries, particularly from R and Python programming communities, has increased the availability and utility of many ML methods, including ANNs. Moreover, the massive increase in available computer power has reduced compute times such that methods previously intractable due to computational expense, such as bootstrap confidence intervals (Efron, 1988), have enabled non-parametric statistical inference to be derived for previously considered uninterpretable 'black box' methods. This opens the door for the development of an ANN framework comparable to that of PLS-DA.

The aim of this study is to migrate the standardised *optimisation, visualisation, evaluation,* and *statistical inference* techniques commonly used in a PLS-DA binary classification over to a nonlinear, single hidden layer, ANN algorithm, and then conduct a direct comparison of utility. We provide two functionally equivalent workflows (PLS-DA vs. ANN) implemented using the Python programming language, and presented as open-access Jupyter Notebooks (https://cimcb.github.io/MetabProjectionViz/). The workflows were applied to two previously published metabolomics datasets by Chan *et al.* (2016) & Ganna *et al.* (2016), but are written to be used with any data set suitably formatted following previous guidelines (Mendez *et al.*, 2019b). Both workflows include cross-validated hyperparameter optimisation, latent variable projection scores plots, classification evaluation using receiver operator characteristic curves, bootstrap resampling for statistical inference of feature contribution and generalisability of prediction metrics.

5.2. Methods

5.2.1. Partial Least Squares Discriminant Analysis (PLS-DA)

PLS-DA (Wold, 1975; Wold *et al.*, 1993) is a widely used multivariate ML algorithm used for classifying and interpreting metabolomics data, especially applicable when the number of metabolites (independent variables) is much larger than the number of data points (samples). PLS uses the *projection to latent space* approach to model the linear covariance structure between two matrices (**X** and **Y**). If the **X** matrix is thought of as a set of N data points in M-dimensional space (where, N = number of samples, and M = number of metabolites), and **Y** is a binary vector (length N) describing the class of each samples (e.g. case = 1 and control = 0), and if we consider the algorithm geometrically, the PLS algorithm rotates and projects **X** into a lower K dimensional space (typically K = 2 or 3), represented by the scores matrix **T**, such that discrimination (covariance) between the two labelled groups in the subspace is maximised (Eriksson *et al.*, 2013). For this study, PLS-DA models was optimised using the iterative SIMPLS algorithm (de Jong, 1993). **T** can be derived from **X** using Eq. (1), where **W,** the X-weight matrix, describes how the X-variables are linearly combined, or geometrically rotated, to form the score vectors, $t_1, t_2 \ldots t_K$.

$$\mathbf{T} = \mathbf{XW} \tag{1}$$

The predicted classification (**Y***) can then be calculated from **T** using Eq. (2), where **C** is the Y-weights matrix describing how the **Y** vector is rotated to map to the covariance described by **T**.

$$\mathbf{Y}^* = \mathbf{TC}' \tag{2}$$

These matrix equations, Eq. (1) and Eq. (2), can be combined and simplified to a single linear regression, Eq. (3), where $\mathbf{B_{PLS}}$ is a vector of coefficient values.

$$\begin{aligned} \mathbf{Y}^* &= \mathbf{TC}' \\ \mathbf{Y}^* &= \mathbf{XWC}' \\ \mathbf{Y}^* &= \mathbf{XB_{PLS}} \end{aligned} \tag{3}$$

This matrix equation, Eq. (3), can also be described as a single linear regression in standard form, Eq. (4), where $\beta_0 \ldots \beta_N$ is a vector of linear coefficients.

$$y^* = \beta_0 + \beta_1 x_1 + \beta_1 x_2 + \cdots + \beta_M x_M \tag{4}$$

5.2.1.1. PLS-DA Optimisation

The optimal number of latent variables, *K*, is determined such that the **T** matrix is just sufficient to accurately describe the underlying latent structure in **X** but not so large as to also model random correlation and produce a model that is a poor classification tool for new X-data (see cross-validation in Section 5.3.3). In machine learning terminology any parameter which is used to define a model's structure, or an optimisation algorithm characteristic, is known as a *hyperparameter*. Thus, the number of latent variables is the single PLS-DA hyperparameter.

5.2.1.2. PLS-DA Evaluation

In order to provide some level of independent model evaluation it is common practice to split the source data set into two parts: training set and test set (typically, ⅔ training and ⅓ test). Once the optimal number of latent variables has been determined using the training data only ($\mathbf{X_{train}}$ and $\mathbf{Y_{train}}$), the resulting model, $\mathbf{Y}^* = \mathbf{XB_{PLS}}$, is then independently evaluated by applying the test data ($\mathbf{X_{test}}$; suitably transformed and scaled) to the model, $\mathbf{Y^*_{Test}} = \mathbf{X_{Test}B_{PLS}}$. A measure of the predictive ability of the model can then be calculated by comparing the training prediction ($\boldsymbol{Y^*_{train}}$) to the expected training outcome ($\mathbf{Y_{train}}$), and the test prediction ($\mathbf{Y^*_{test}}$) to the expected test outcome ($\mathbf{Y_{test}}$).

While true effectiveness of a model can only be assessed using test data (Westerhuis *et al.*, 2008; Xia *et al.*, 2013), for small data sets it is dangerous to use a single random data split as the only means of model evaluation, as the random test data set may not accurately represent the training data set (Mendez *et al.*, 2019c). An alternative is to use bootstrap resampling. Bootstrap resampling is a method for calculating confidence intervals using random sampling with replacement (DiCiccio and Efron, 1996; Efron, 1981, 2000). The theoretical details of this methodology are beyond the scope of this paper. Briefly, this technique allows the accurate estimation of the sampling distribution of almost any statistic using repeated random sampling. Each random sample selects ~⅔ of the data points (called the in-bag sample) leaving ~⅓ (the out-of-bag sample).

Bootstrapping can be used to calculate confidence measurements for the evaluating the optimal ML model configuration for a given metabolomics data set (Broadhurst and Kell, 2006; Mendez *et al.*, 2019b; Xia *et al.*, 2013). A model with fixed hyperparameter values is retrained

on data, randomly sampled with replacement (in-bag), and then evaluated on the unused data (out-of-bag) for r resamples (typically r = 100). The predicted outcome from each in-bag bootstrap resample as well as other outputs, including the predicted outcome, latent scores, latent loadings, and feature contribution metrics are stored after each resampling. The out-of-bag prediction of classification is also stored, as this can be considered an unbiased estimate of the model's performance when shown new data. Using these stored outputs, 95% confidence intervals are calculated using the commonly-used bias-corrected and accelerated (BCa) method; this method adjusts the percentiles to account for the bias and skewness in the bootstrap distribution (Efron, 1987). Following bootstrap resampling, a measure of generalised prediction of each model is calculated as the median and 95% confidence intervals of the in-bag and out-of-bag predictions.

5.2.1.3. PLS-DA Visualisation

For a given PLS-DA model it is common practice to visualise the projection of **X** into the latent variable space to provide a generalised understanding of the metabolomic relationship (clustering) between individual samples before classification. For this, the scores matrix, **T**, described in Eq. (1), can be represented as a scatter plot (scores plot) such that each axis of the plot represents a column of the T-matrix. For example, a scatter plot of t_1 vs. t_2 will represent the projections of X onto the first two latent variables (i.e. each data point represents a projection of a given sample's metabolite profile). It is in this latent variable space that one would expect to see different metabotypes cluster. The associated weight vectors (columns of **W**) can also be visualised individually and interpreted as an indication of how the X-variables are linearly combined to create each score vector, Eq. (5).

$$
\begin{aligned}
t_1 &= w_{0,1} + w_{1,1}x_1 + w_{2,1}x_2 + \cdots + w_{M,1}x_M \\
t_2 &= w_{0,2} + w_{1,2}x_2 + w_{2,2}x_2 + \cdots + w_{M,2}x_M \\
&\cdots \\
t_K &= w_{0,K} + w_{1,K}x_1 + w_{2,K}x_2 + \cdots + w_{M,K}x_M
\end{aligned} \tag{5}
$$

For a single optimised model, latent scores plots can be generated for training, cross-validation, and test X-data sets independently. This is a useful method for determining if overtraining has occurred (see supplementary Jupyter Notebooks).

5.2.1.4. PLS-DA Variable Contribution

For PLS-DA, there are two common methods used to estimate variable contribution. First, as discussed, a PLS-DA model can be reduced to a single multiple linear regression, Eq. (3), thus feature contribution can be inferred directly from the model's regression coefficients, $\mathbf{B_{PLS}}$. Second, for more of a focus on the importance of the X-variables on the latent projection, the *variable influence on projection* (VIP) scores can be calculated using Eq. (6) (Favilla *et al.*, 2013). VIP is the weighted, w_i^2, combination of the sum of squares of Y explained by each latent variable, SSY_i, normalised to the cumulative sum of square, SSY_{cum}, where M is the total number of metabolites, and K is the total number of latent variables.

$$
\mathbf{VIP} = \sqrt{M \times \frac{\sum_{i=1}^{K} w_i^2 \times SSY_i}{SSY_{cum}}} \tag{6}
$$

The average VIP score is equal to 1 because the sum of squares of all VIP scores is equal to the number of variables in **X**. Thus, if all X-variables have the same contribution to the model, they will have a VIP score equal to 1. VIP scores larger than 1 indicate the most relevant variables. Bootstrap resampling (Section 5.2.1.2) can be applied to calculate 95% confidence

intervals for both the $\mathbf{B_{PLS}}$ coefficient values and **VIP** scores, from which estimates of significant contribution to the model can be determined.

5.2.2. Artificial Neural Network (ANN)

ANNs consist of layered weighted networks of interconnected mathematical operators (neurons). The most prevalent ANN is the feed-forward neural network. Here, each neuron acts as a weighted sum of the outputs of the previous layer (or input data) transformed by an activation function (typically linear or logistic function). This is described in Eq. (7), using notation from Fig. 5.1a, where t_j is the output for the j^{th} neuron in the hidden layer, f_0 is the activation function, x is a vector of input variables ($x_1, x_2, \ldots, x_M$), $w_{i,j}$ is the weight from input variable, x_i, to the neuron, and $w_{0,j}$ is a constant offset value.

$$t_j = f_0\left(w_{0,j} + \sum_{i=1}^{M} w_{i,j} \times x_i\right) \tag{7}$$

A neuron with a linear activation function connected to multiple input variables is mathematically equivalent to a linear regression with multiple independent variables, Eq. (8), where $w_{0,j} \ldots w_{N,j}$ is a vector of linear coefficients.

$$t_j = w_{0,j} + w_{1,j}x_1 + w_{2,j}x_2 + \cdots + w_{M,j}x_M \tag{8}$$

A neuron with a logistic activation function, $f_0()$, is equivalent to the multivariate logistic regression describe in Eq. (9).

$$t_j = \frac{1}{1 + e^{-\left(w_{0,j} + \sum_{i=1}^{M} w_{i,j} \times x_i\right)}} \quad (9)$$

An ANN with a single linear hidden layer and a single linear output neuron is mathematically equivalent to a PLS-DA model (Fig. 5.1). Replacing all the linear neurons with logistic neurons in the two-layer ANN results in a complex non-linear projection-based discriminant model. For this study, we use a two-layer ANN with logistic activations in both layers.

5.2.2.1. ANN Optimisation

During ANN training, the interconnection weights between each layer of neurons are optimised using an iterative algorithm known as *back-propagation*. This algorithm has been described in detail elsewhere (Bishop, 1995). The effectiveness of this optimisation method is dependent on a set of *hyperparameters.* A two-layer feedforward ANN has 5 hyperparameters: 1 parameter to determine the model structure, the *number of neurons* in the hidden layer (equivalent to number of latent variables) and 4 parameters that characterise the learning process. These determine the rate and momentum of traversing local error gradients (specifically *learning rate*, *momentum*, and *decay* of the learning rate over time) and the number of times the back-propagation is applied to the ANN (the number of training *epochs*). For this study, preliminary explorative analysis indicated that hyperparameters: *momentum*, *decay*, *epochs* could be set to a constant value (*0.5*, *0* and *400* respectively) with little variation on performance. This reduced the number of tuneable hyperparameters to: (i) the *number of neurons in the hidden layer*, and (ii) the *learning rate*.

5.2.2.2. ANN Evaluation

Model evaluation using a test set and model evaluation using bootstrap resampling is identical to that described in Section 5.2.1.2 except replacing the PLS-DA prediction, $\mathbf{Y}^*$, with the ANN equivalent.

5.2.2.3. ANN Visualisation

For an equivalent representation of the PLS-DA projection to latent space, we provide a projection to neuron space. Each hidden neuron represents a transformed weighted sum of the X-variables (Eq. 7). Thus, for each pairwise combination of neurons, plotting the weighted sum before transformation provides a similar means to PLS-DA for visualising and interpreting any clustering between individual samples before classification. Similarly, associated weight vectors can also be visualised individually and interpreted as an indication of how the X-variables are linearly combined to create each neuron scores vector before transformation.

5.2.2.4. ANN Variable Contribution

For ANN, several variable contribution metrics have been proposed (Olden *et al.*, 2004); however, the two most comparable metrics to the PLS-DA $\mathbf{B_{PLS}}$ coefficients and VIP scores are the Connection Weight Approach (CWA) (Olden and Jackson, 2002) and Garson's Algorithm (GA) (Garson, 1991), respectively. Similar to $\mathbf{B_{PLS}}$, for a two-layer ANN with linear activation functions (Fig. 5.1b), feature contribution can be inferred directly from a model's linear coefficients, $\mathbf{B_{ANN}}$, as shown in Eq. (10), where $\mathbf{C}$ is the weights for the hidden-output layer, and $\mathbf{W}$ is the weights for the input-hidden layer.

$$\mathbf{CWA} = \mathbf{B_{ANN}} = \mathbf{CW} \tag{10}$$

This equation can be used to calculate variable contribution for two-layer non-linear ANNs, renamed as CWA, and describes *relative* (and *directional*) metabolite contribution.

While VIP may not be directly applied to non-linear ANNs, a similar measure of weighted *absolute relative* contribution of each metabolite per neuron can be calculated using Garson's Algorithm (Garson, 1991). First, absolute $CWA_{i,j}$ values are calculated across the network by multiplying each neuron input weight, $w_{i,j}$, to the corresponding output weight, c_j and converting to an absolute value.

$$\left|CWA_{i,j}\right| = \left|w_{i,j} \times c_j\right| \tag{11}$$

Second, as shown in Eq. (12), for each hidden neuron the total absolute connection weight value is calculated, where M is the total number of metabolites.

$$\left|CWA_{\,j}\right| = \sum_{i=1}^{M} \left|CWA_{i,j}\right| \tag{12}$$

Then, the overall contribution for each input variable, GA_i, is calculated as shown in Eq. (13), where K is the total number of hidden layer neurons.

$$GA_i = \sum_{j=1}^{K} \left(\frac{\left|CWA_{i,j}\right|}{\left|CWA_j\right|} \right) \tag{13}$$

Unlike VIP there is no general threshold of importance for Garson's Algorithm, so we propose using the average GA score as a comparable equivalent to indicate metabolites of importance in the model.

5.3. Computational Workflow

The standard workflow for the PLS visualisation and interpretation, and the proposed equivalent ANN visualisation and interpretation is described in Fig. 5.2. Both the PLS-DA and ANN workflows were implemented in the Python programming language using a package called 'cimcb' (https://github.com/CIMCB/cimcb) developed by the authors. This package contains tools for the analysis and visualisation of untargeted and targeted metabolomics data. The package is based on existing well curated open-source packages (including *numpy* (Kristensen and Vinter, 2010), *scipy* (Virtanen et al., 2019), *bokeh* (Bokeh Development Team, 2018), *keras* (Chollet, 2015*)*, *pandas (McKinney, 2010)*, *scikit-learn* (Pedregosa et al., 2011*)*, and *Theano (Theano Development Team, 2016)*). It utilises these packages through *helper functions* specifically designed to simplify the application to metabolomics data, following guidelines previously described (Mendez *et al.*, 2019b).

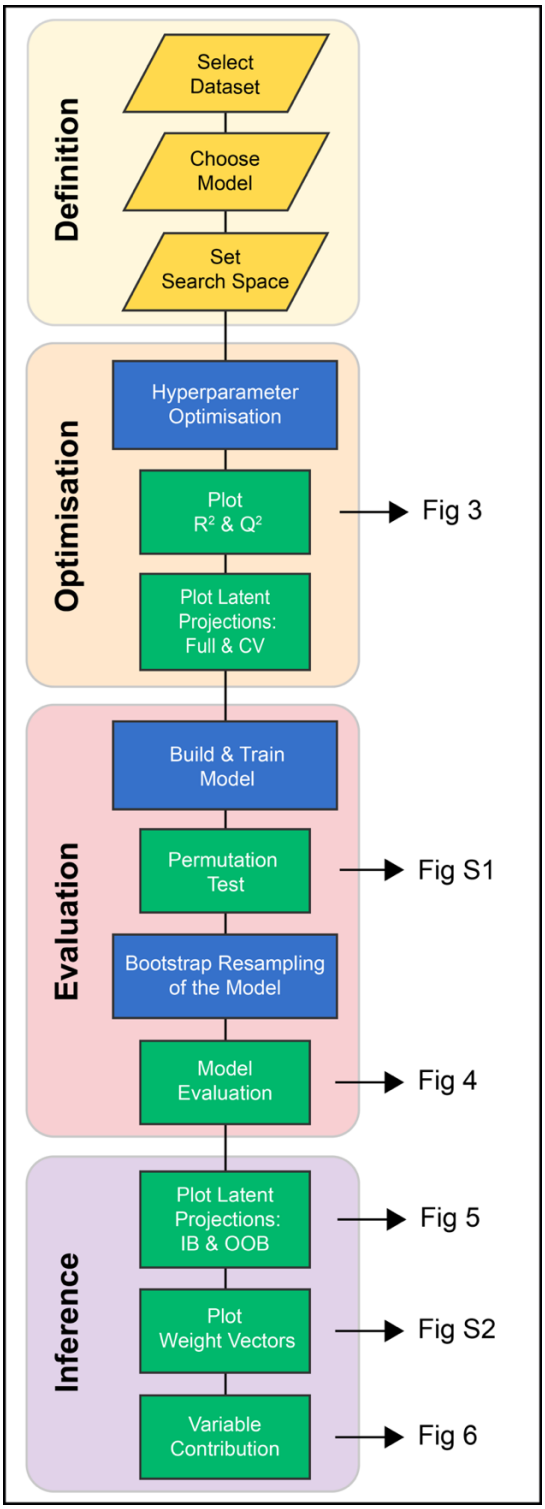


Figure 5.2: Data analysis workflow. Flowchart of the data analysis workflow used for the PLS and ANN methods. Arrows identify the figure corresponding to the respective workflow step.

Each step of the respective PLS-DA and ANN workflow is described in detail in the associated Jupyter Notebook file (included in supplementary material and https://cimcb.github.io/MetabProjectionViz/). The method of embedding explanatory text within functional code and visualisations follows previously published guidelines (Mendez *et al.*, 2019b). The generic workflow is now briefly described.

5.3.1. Prepare Data

For an adequate comparison of visualisation and interpretation methods, across PLS and ANN, it was important that identical data were used in both models. The **X** matrix of metabolite concentrations, and associated **Y** vector of classification labels (case = 1, control = 0) were extracted from the excel spreadsheet. Metabolites in **X** were included for modelling if they had a QC relative standard deviation (RSD_{QC}) < 20% and <10% missing data (Broadhurst *et al.*, 2018). The datasets were split using a ratio of 2:1 (⅔ training, ⅓ test) using stratified random selection. After splitting the data into training and test sets, the columns of **X** were natural log transformed, mean centred, and scaled to unit variance with missing values imputed using k-nearest neighbour prior to modelling following standard protocols for metabolomics (Broadhurst and Kell, 2006). The means and standard deviations calculated from the training set were applied to scale the test set data.

5.3.2. Hyperparameter Optimisation

For both PLS-DA and ANN algorithms the optimal hyperparameter values were determined using 5-fold cross-validation (CV) with 10 Monte Carlo repartitions (Broadhurst and Kell, 2006; Hastie *et al.*, 2009; Xia *et al.*, 2013). For the PLS-DA workflow, a linear search was used

to optimise the number of latent variables (1 to 6). For the ANN workflow, a grid search was used to optimise the number of neurons (2 to 6) and the learning rate (0.001 to 1). The optimal hyperparameter values were determined by evaluating plots of R^2 and Q^2 statistics. Two plots were generated: (i) a standard R^2 and Q^2 plot against hyperparameter values, and (ii) an alternative plot of $|R^2 - Q^2|$ $vs. Q^2$. Using the later plot, the optimal hyperparameter was selected at the point of inflection of the outer convex hull. The area under the receiver operating characteristic curve (AUC) is a recommended alternative non-parametric measure of classification performance (Szymańska *et al.*, 2012), thus equivalent plots of AUC_{Full} and AUC_{CV} metrics are also generated for comparison.

5.3.3. Permutation Test

Following hyperparameter optimisation, a permutation test was applied to the optimal model configuration. In a permutation test, the expected outcome label is randomised (permuted), and the model with fixed hyperparameter values is subsequently trained and evaluated (Lindgren *et al.*, 1996). For both PLS-DA and ANN, this process was repeated (n=100) using 5-fold CV to construct a distribution of the permuted model statistics. While R^2 and Q^2 statistics are commonly used in permutation testing (Eriksson *et al.*, 2013), AUC_{Full} and AUC_{CV} metrics were also included for ANNs, given its common usage as a measure of non-linear classification performance.

5.3.4. Model Evaluation using Test Set

As previously described in Section 5.2.1.2, the measure of the predictive ability of the model using a test set is calculated by comparing the training score ($\mathbf{Y}^*_{\mathbf{train}}$) to the expected outcome

($\mathbf{Y_{train}}$) classification, and the test score ($\mathbf{Y^*_{test}}$) to the expected outcome ($\mathbf{Y_{test}}$) classification. This is visualised using three plots:

1. A violin plot that shows the distribution of the predicted score, by outcome, for the training and test set.
2. A probability density plot that shows the distribution of the predicted score, by outcome, for the training and test set via overlapping probability density functions.
3. A receiver operator characteristic (ROC) curve of the training and test sets.

5.3.5. Model Evaluation using Bootstrap Resampling

Model evaluation using bootstrap resampling is described in Section 5.2.1.2. Following bootstrap resampling (n=100), a measure of generalised prediction of each model is calculated and visualised using the protocol described in 5.3.4, except this time presenting the 95% confidence intervals of the 100 in-bag and out-of-bag predictions.

5.3.6. Model Visualisation: Scores Plot and Weights Plot

Pairwise latent variable scores plots and associated weight vector plots are also provided. The scores plots are similar in construction to those generated during hyperparameter optimisation, except they are based on the in-bag and out-of-bag scores averaged across repeated prediction for each sample (aggregate score). 95% confidence intervals for each class are calculated using standard parametric methods. The 95% confidence intervals for each weight vector plots were constructed using the distribution of each weight variable across the 100 bootstrap resampled

models. Any metabolite weight with a confidence interval crossing the zero line (coloured blue) are considered non-significant to the latent variable (or neuron).

5.3.7. Variable Contribution Plots

The $\mathbf{B_{PLS}}$ coefficients and VIP scores for the PLS models were calculated using the methods described in Section 5.2.1.4. The CWA and Garson scores were calculated for the ANNs using the methods described in Section 5.2.2.4. There metrics were also applied to all 100 models of each type generated during the bootstrap resampling. Variable contribution plots were constructed. The 95% confidence intervals for each vector plots were calculated using the distribution of each variable's metric across the 100 bootstrap resampled models. Any metabolite weight with a confidence interval crossing the zero line are considered non-significant to the latent variable (or neuron).

The variable contribution metrics for each model type was compared and contrasted through visual inspection of a scatter plots of $\mathbf{B_{PLS}}$ *vs.* $\mathbf{CWA_{ANN}}$ and of $\mathbf{VIP_{PLS}}$ *vs.* $\mathbf{Garson_{ANN}}$ scores, and by calculating the associated Pearson's correlation coefficient.

5.4. Results

5.4.1. Datasets

In this study, a previously published dataset by Chan *et al.* (2016) was used to illustrate the standardised PLS workflow and the proposed equivalent ANN workflow. This urine nuclear magnetic resonance (NMR) dataset, comprised of 149 metabolites, is publicly available on

Metabolomics Workbench (Study ID: ST0001047). For the work described herein a binary classification was performed: gastric cancer (n=43) vs. healthy controls (n=40).

The computational libraries developed for this study require data to be converted to a standardised format using the *tidy data* framework (Wickham, 2014). This standardised format has been previously described (Mendez *et al.*, 2019b; Mendez *et al.*, 2019c), and allows for the efficient reuse of these workflows for other studies. To demonstrate this, we include the application of the identical workflows and visualisation techniques to a second previously published dataset (Ganna *et al.*, 2016) as a supplementary document. This plasma liquid chromatography-mass spectrometry (LC-MS) dataset, comprised of 189 named metabolites, is publicly available on *MetaboLights* (Study ID: MTBLS90), and for this study, samples were split into two classes by sex: males (n=485) and females (n=483). This dataset did not report QC measurements and therefore the data cleaning step was unable to be performed.

Following data cleaning, for the urine NMR gastric cancer data set 52 metabolites were included in data modelling (case = 43 vs. control = 40). Figs. 5.3-5.6 (and Supplementary Figs. 5.1-5.2) show the optimisation, visualisation, evaluation and statistical inference for the PLS-DA compared to the ANN algorithms. Similar plots are provided in supplementary documentation for the plasma LC-MS data set (males=485 vs. females=483). All 4 workflows are also available as interactive Jupyter notebooks (https://cimcb.github.io/MetabProjectionViz/), either to be downloaded or to be run in the cloud through mybinder.org. See Mendez *et al.* (2019b) for guidance.

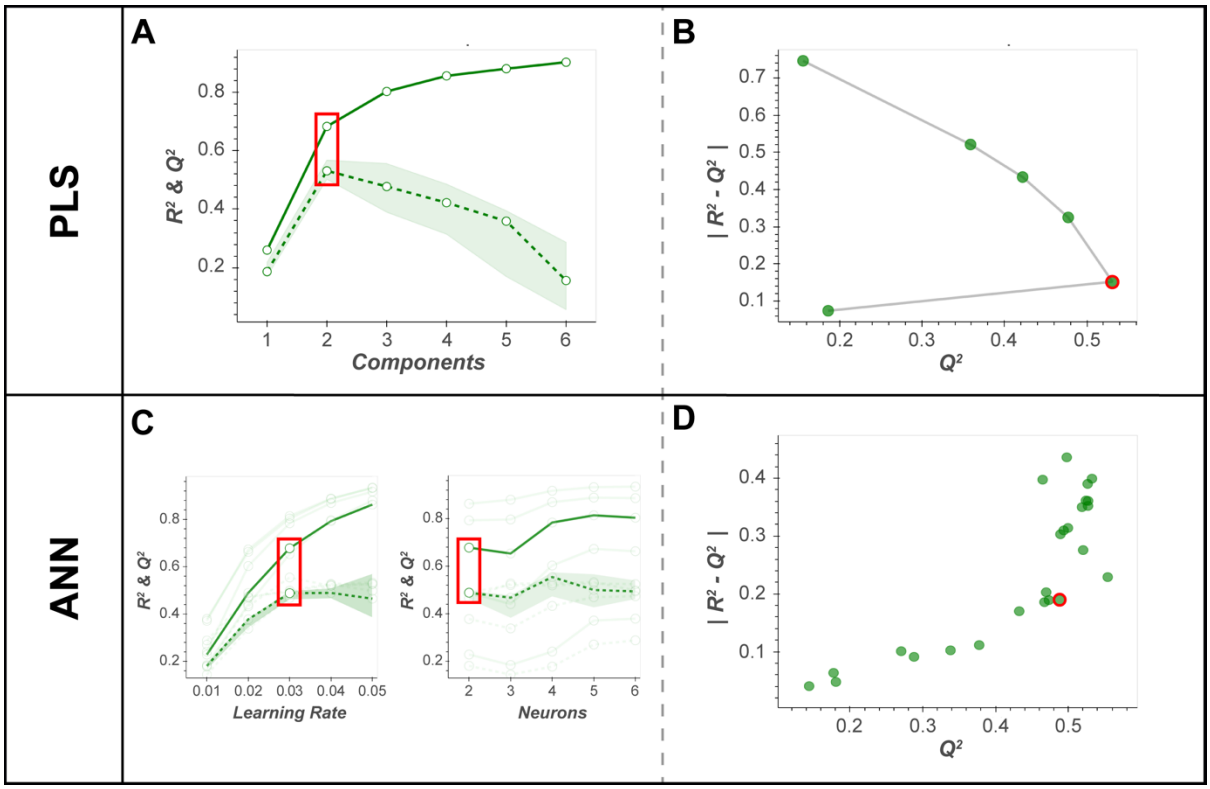


Figure 5.3: Hyperparameter optimisation. Plots of R^2 and Q^2 statistics; red circle, optimal hyperparameter value(s). **a & c** Standard R^2 and Q^2 vs hyperparameter values plot for PLS and ANN, respectively. Solid line, R^2; dashed line, Q^2. **b & d** The alternate $|R^2 - Q^2|$ $vs.$ Q^2 plot for PLS and ANN, respectively. The optimal hyperparameters shown in panel **c** were identified using the plot in panel **d**.

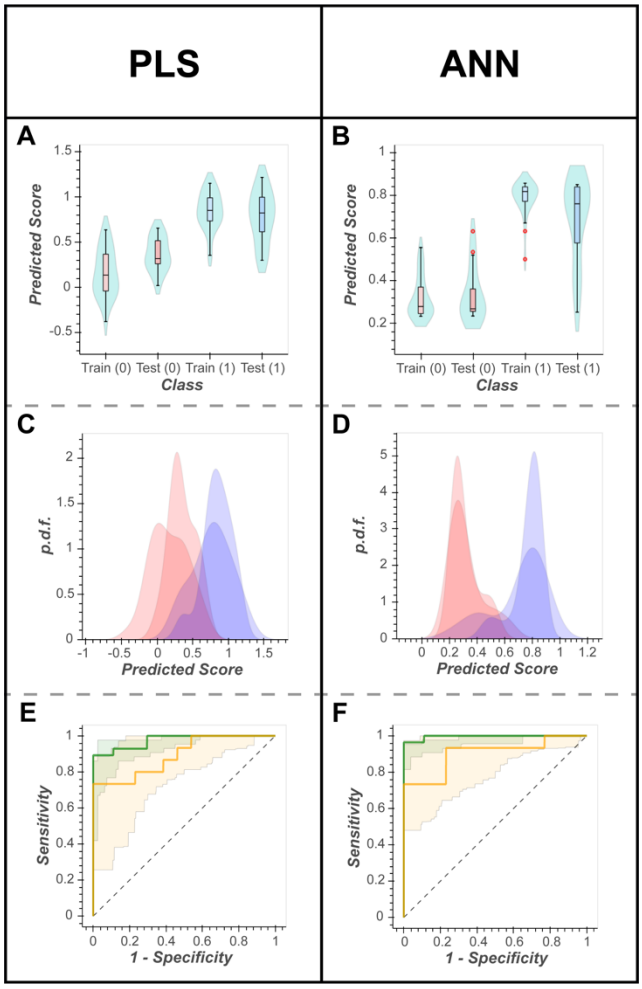


Figure 5.4: Visualisations of model evaluation. Predicted scores (train and test) split into the respective binary classification, visualised in three different ways. **a & b** Violin plots; **c & d** probability distribution function (pdf) plots. Red, healthy controls (control); blue, gastric cancer (case). **e & f** ROC curves with 95% CIs derived from 100 iterations of bootstrap resampling. Green line predicted scores for training set; green 95% CIs, IB predictions; yellow line, prediction scores for test set; yellow 95% CIs, OOB predictions. PLS-DA $AUC_{Train} = 0.97$, $AUC_{Test} = 0.89$, $AUC_{IB} = 0.92\text{-}0.99$, $AUC_{OOB} = 0.72\text{-}0.98$. ANN $AUC_{Train} = 1.00$, $AUC_{Test} = 0.90$, $AUC_{IB} = 0.95\text{-}0.99$, $AUC_{OOB} = 0.77\text{-}1.00$.

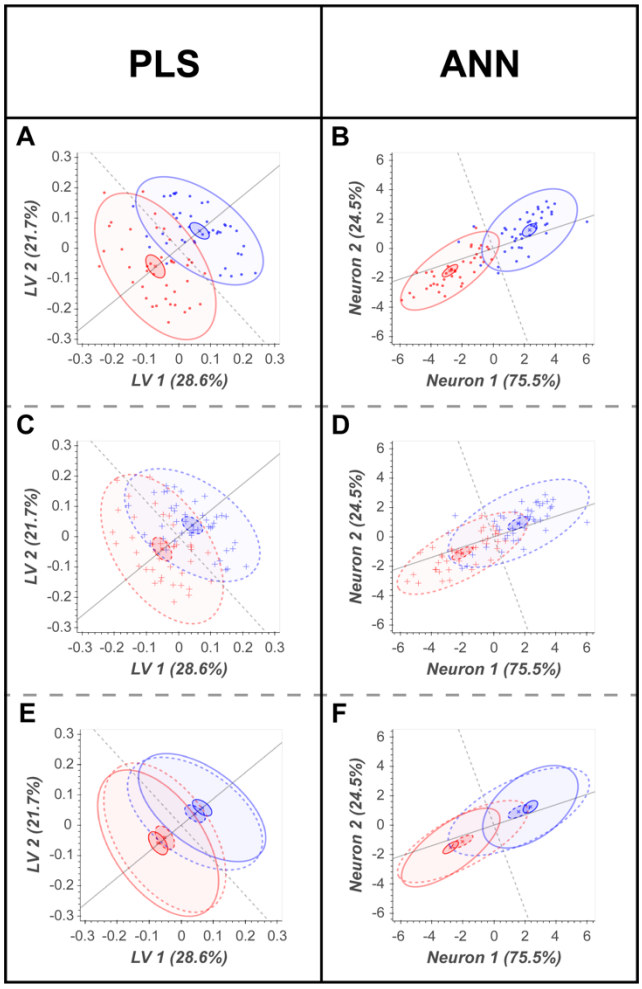


Figure 5.5: Bootstrap projection (scores) plots. Projection plots show LV2 vs LV1 for PLS and Neuron 2 vs Neuron 1 for ANN. **a & b** projected scores of the median IB; **c & d** projected scores for median OOB; **e & f** median IB and median OOB scores overlaid. Red, healthy control (control); blue, gastric cancer (case). Inner ellipses, 95% CI of the mean; outer ellipses, 95% CI of the population. Solid lines, IB predictions; dashed lines, OOB predictions.

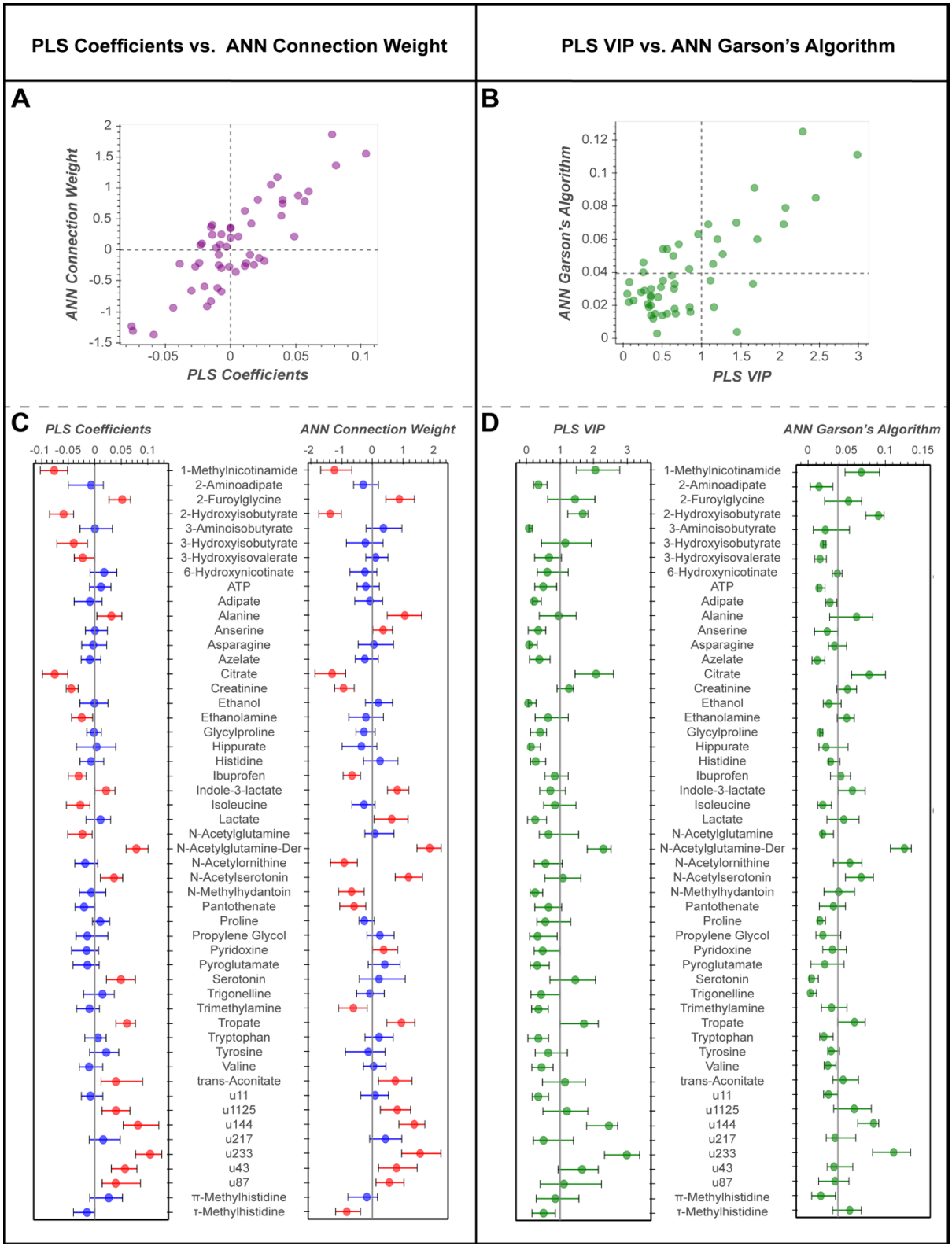


Figure 5.6: Variable contribution. Visualisation of variable contribution for PLS (coefficients and VIP) and ANN (CWA and Garson's algorithm). **a** Scatterplot of ANN_{CWA} vs. $\boldsymbol{B}_{PLS}$, Pearson's r = 0.85 (p-value = $2.79e^{-15}$). **b** Scatterplot of $Garson_{ANN}$ vs. VIP_{PLS}, Pearson's r = 0.75 (p-value = $1.33e^{-10}$). Dashed lines at respective "importance" cut-off: $Garson_{ANN}$ = 0.038, VIP_{PLS} = 1.00. **c** Median (and 95% CI) $\boldsymbol{B}_{PLS}$ (left) and ANN_{CWA} (right). Blue, contribution not significant based on 95% CIs; red, contribution significant based on 95% CIs. **d** Median (and 95% CI) VIP_{PLS} (left) and $Garson_{ANN}$ (right).

5.4.2. Model Optimisation

Using the = $|R^2 - Q^2|\ vs. Q^2$ plot, both the number of latent variables (LV=2; Fig. 5.3a) and ANN hyperparameters (learning rate = 0.03 & hidden neurons =2; Fig. 5.3d) were clearly interpretable. These findings were verified using permutation testing (Supplementary Fig. 5.1).

5.4.3. Model Evaluation and Visualisation

Strategies for model evaluation and visualisation were successfully transferred from PLS-DA to ANNs. For both example data sets the ANN model performed slightly better than the PLS-DA for both the training and test data sets (Fig. 5.4). Both models somewhat overtrained despite rigorous cross-validation. For the PLS-DA model the AUC_{Train} = 0.97 and the AUC_{Test} = 0.89. For the ANN model the AUC_{Train} = 1.00 and AUC_{Test} = 0.90. Bootstrap remodelling also showed similar results. The PLS-DA model had an in-bag area under the ROC curve (AUC) with 95% CI of 0.92-0.99. Similarly, the ANN produced an in-bag AUC with 95% CI of 0.95-0.99. The out-of-bag predictions showed that both models overtrained with out-of-bag AUC 95% CI of 0.72-0.98 (PLS-DA) and 0.77-1.00 (ANN). The bootstrap projections confirmed these findings and illustrated that the models were still able to project significant mean differences between classes, for both the in-bag and out-bag projections (Fig. 5.5).

5.4.4. Model Inference

Feature contribution was determined by calculating bootstrap confidence intervals for the model coefficients $\boldsymbol{B_{PLS}}$ (or equivalent CWA_{ANN}) and of the VIP_{PLS} (or equivalent $Garson_{ANN}$). Across the two models, $\boldsymbol{B_{PLS}}$ and CWA_{ANN} showed a high degree of correlation (Fig. 5.6a;

Pearson's r = 0.85, p = 2.8 x 10^{-15}). Twenty-three metabolites significantly contributed to the PLS-DA model and 25 metabolites significantly contributed to the ANN model, with an overlap of 17 metabolites being significant in both models (Fig. 5.6a). The VIP_{PLS} and $Garson_{ANN}$ values showed a reduced, but still significant, degree of correlation with each other (Fig. 5.6b; Pearson's r = 0.75, p = 1.33 x 10^{-10}). Based on median values alone (Fig. 5.6b), 12 metabolites were deemed as "important" across both models and an additional 12 metabolites were "important" in one, but not both models. When taking into consideration bootstrapped confidence intervals (Fig. 5.6d) VIP_{PLS} and $Garson_{ANN}$ yielded 7 and 8 "important" metabolites, respectively. Six metabolites deemed "important" by $Garson_{ANN}$ were also deemed important by VIP_{PLS}. Although mathematical calculations for variable contribution were different for the two models, Fig. 5.6 shows that the overall visualisation strategy was transferrable.

5.5. Discussion

The migration of the PLS-DA optimisation, evaluation, and interpretation workflow to a single hidden layer ANN was successful. The strategy for visualising hyperparameter optimisation was adapted to the $|R^2 - Q^2|\ vs. Q^2$ plot (Fig. 5.3c-d) and readily employable to both model types. Not only did it allow for simultaneous interpretation of 2 hyperparameters (ANNs), but it provides an alternate interpretation strategy for PLS-DA optimisation if the standard R^2 and Q^2 vs hyperparameter value plot is ambiguous. Model evaluation and projection (scores) plots were directly transferrable from PLS-DA to ANNs. Projecting the neuron weights (in place of latent variables) before the transfer function allows for a comparative and clear visual disruption of sample similarity. The bootstrap resampling/remodelling enabled both the PLS-DA and ANN models' predictions to be interpreted with statistical rigor. Both models had

similar performance, but as described (and expected) in the bootstrap projections (Fig. 5.5) and loadings (Supplementary Fig. 5.2).

CWA and *Garson* provided suitable variable contribution metrics for the ANN model. The surprising similarity between $\boldsymbol{B_{PLS}}$ and CWA_{ANN} and VIP_{PLS} and $Garson_{ANN}$ indicates the validity of both CWA_{ANN} and $Garson_{ANN}$ as methods of determining feature importance. These findings are validated by the second study (supplementary documentation). It is important to note that no one ML method will be superior for identifying the most biological plausible metabolites. The high level of overlap between comparable variable contribution methods, in these results, suggest that deviations are likely random false discoveries due to lack of power (as reflected in the 95% CIs are how close they are to the zero line). As the cut-off for both *VIP* and $Garson_{ANN}$ are not statistically justified limits (Tran *et al.*, 2014), we recommend opting for $\boldsymbol{B_{PLS}}$ for PLS and CWA_{ANN} for ANN, and using the 95% CI from bootstrap resampling to determine statistically significant metabolites.

As a side note, it is worth discussing two additional points. First, there is an advantage of using bootstrap resampled predictions and projections once the optimal hyperparameters are fixed. This is particularly important if the sample size is small and there may be large differences in results depending on how the samples are split into training and test sets. The out-of-bag predictions provide an unbiased estimate of model performance, and the averaged out-of-bag projections a more realistic estimate of generalised class-based cluster similarity. Bootstrapping can also aid in preventing false discoveries regarding metabolite significance, as the resulting 95% CIs will identify metabolites with unstable contributions to the model. Second, model outcomes and resulting interpretations can affected by the quality of the input data. We have previously shown that PLS and ANNs show similar predictive ability, when

using the same input data, and that sample size is an important determinant of model stability (Mendez *et al.*, 2019c). However, to our knowledge, an extensive comparison of different data cleaning (Broadhurst *et al.*, 2018), pre-treatment (van den Berg *et al.*, 2006), and imputation (Di Guida *et al.*, 2016; Do *et al.*, 2018) procedure options has not been performed for ANNs. As such, individual users should consider and test these effects prior to modelling their own data.

5.6. Conclusion and Future Perspectives

We have shown that for binary discrimination using metabolomics data it is possible to migrate the workflow from PLS-DA to a single hidden layer non-linear ANN. For the two presented examples the ANN does not perform any better than PLS-DA, and based on coefficient plots there is very similar feature contribution. However, these results show that ANNs can be evaluated alongside PLS-DA for any data set (using the provided Jupyter notebooks it is possible to evaluate any binary classification data set provided it is formatted appropriately before uploading). If a highly non-linear relation should arise, then ANN may be a better approach to PLS. This remains to be proven.

More importantly these result open the door to investigating more complex models. As discussed previously (Mendez *et al.*, 2019a), an area of increasing interest to the metabolomics community is multi-block data integration (e.g. multi-omic or multi-instrument). Currently, methods employed are based on hierarchical application of multiple linear projection models. For example, OnPLS (Löfstedt and Trygg, 2011; Reinke *et al.*, 2018) is a combinatorial amalgamation of multiple PLS models, and Mixomics (Rohart *et al.*, 2017) is a stepwise integration of canonical correlation analysis and sparse PLS. The inherent flexibility of ANN

architecture allows complex relationships to be combined into a single model. It may be possible to build an ANN to combine multiple data blocks into a single model without resorting to over-simplified data concatenation. For these types of models to be useful will be necessary to incorporate feature importance, and interpretable visualisation strategies. The work presented here is a first step to applying statistical rigor and interpretability to more complex ANN models.

Acknowledgements

This work was partly funded through an Australian Research Council funded LIEF grant (LE170100021).

Conflicts of Interest

The authors have no disclosures of potential conflicts of interest related to the presented work.

Authors Contributions

All authors conceived of the idea. KMM and DIB developed the software. KMM wrote the manuscript. DIB and SNR edited the manuscript.

Data Availability

The metabolomics and metadata used in this paper were retrieved Metabolomics Workbench (https://www.metabolomicsworkbench.org/) Study ID: ST0001047, and from Metabolights (https://www.ebi.ac.uk/metabolights/) study identifier: MTBLS90. This data were converted from the original data format to a clean format compliant with the Tidy Data framework, this is available at the CIMCB GitHub project page: https://github.com/CIMCB/MetabProjectionViz.

Software Availability

All software developed for this paper is available at the CIMCB GitHub project page: https://github.com/CIMCB

Compliance with Ethical Standards

No research involving human or animal participants was performed in the construction of this manuscript.

References

Bishop, C.M. (1995) *Neural networks for pattern recognition*. Oxford University Press, New York, United States of America.

Bokeh Development Team (2018). Bokeh: python library for interactive visualization. https://bokeh.pydata.org/en/latest/

Breiman, L. (2001) Random forests. *Machine Learning* **45,** 5-32.

Broadhurst, D., Goodacre, R., Reinke, S.N., Kuligowski, J., Wilson, I.D., Lewis, M.R. and Dunn, W.B. (2018) Guidelines and considerations for the use of system suitability and quality control samples in mass spectrometry assays applied in untargeted clinical metabolomic studies. *Metabolomics* **14,** 72.

Broadhurst, D.I. and Kell, D.B. (2006) Statistical strategies for avoiding false discoveries in metabolomics and related experiments. *Metabolomics* **2,** 171-196.

Chan, A.W., Mercier, P., Schiller, D., Bailey, R., Robbins, S., Eurich, D.T., Sawyer, M.B. and Broadhurst, D. (2016) (1)H-NMR urinary metabolomic profiling for diagnosis of gastric cancer. *British Journal of Cancer* **114,** 59-62.

Chollet, F. (2015). Keras. https://keras.io/

de Jong, S. (1993) SIMPLS: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems* **18,** 251-263.

Di Guida, R., Engel, J., Allwood, J.W., Weber, R.J.M., Jones, M.R., Sommer, U., Viant, M.R. and Dunn, W.B. (2016). Non-targeted UHPLC-MS metabolomic data processing methods: A comparative investigation of normalisation, missing value imputation, transformation and scaling. *Metabolomics* **12**, 93.

DiCiccio, T.J. and Efron, B. (1996) Bootstrap confidence intervals. *Statistical Science* **11,** 189-212.

Do, K.T., Wahl, S., Raffler, J., Molnos, S., Laimighofer, M., Adamski, J., Suhre, K., Strauch, K., Peters, A., Gieger, C., Langenberg, C., Stewart, I.D., Theis, F.J., Grallert, H., Kastenmüller, G. and Krumsiek, J. (2018). Characterization of missing values in untargeted MS-based metabolomics data and evaluation of missing data handling strategies. *Metabolomics* **14,** 128.

Dunn, W.B., Broadhurst, D.I., Atherton, H.J., Goodacre, R. and Griffin, J.L. (2011) Systems level studies of mammalian metabolomes: the roles of mass spectrometry and nuclear magnetic resonance spectroscopy. *Chemical Society Reviews* **40,** 387-426.

Efron, B. (1981) Nonparametric estimates of standard error - the jackknife, the bootstrap and other methods. *Biometrika* **68,** 589-599.

Efron, B. (1987) Better bootstrap confidence intervals. *Journal of the American Statistical Association* **82,** 171-185.

Efron, B. (1988) Bootstrap confidence-intervals - good or bad. *Psychological Bulletin* **104,** 293-296.

Efron, B. (2000) The bootstrap and modern statistics. *Journal of the American Statistical Association* **95,** 1293-1296.

Eriksson, L., Byrne, T., Johansson, E., Trygg, J. and Vikström, C. (2013) *Multi- and megavariate data analysis: basic principles and applications*, 3rd edn. Umetrics Academy, Malmö, Sweden.

Favilla, S., Durante, C., Vigni, M.L. and Cocchi, M. (2013) Assessing feature relevance in NPLS models by VIP. *Chemometrics and Intelligent Laboratory Systems* **129,** 76-86.

Ganna, A., Fall, T., Salihovic, S., Lee, W., Broeckling, C.D., Kumar, J., Hagg, S., Stenemo, M., Magnusson, P.K.E., Prenni, J.E., Lind, L., Pawitan, Y. and Ingelsson, E. (2016) Large-scale non-targeted metabolomic profiling in three human population-based studies. *Metabolomics* **12**.

Garson, G.D. (1991) Interpreting neural network connection weights. *AI Expert* **6,** 47-51.

Geladi, P. and Kowalski, B.R. (1986) Partial least-squares regression: a tutorial. *Analytica Chimica Acta* **185,** 1-17.

Goodacre, R. (2003) Explanatory analysis of spectroscopic data using machine learning of simple, interpretable rules. *Vibrational Spectroscopy* **32,** 33-45.

Goodacre, R., Kell, D.B. and Bianchi, G. (1992) Neural networks and olive oil. *Nature* **359,** 594-594.

Gromski, P.S., Muhamadali, H., Ellis, D.I., Xu, Y., Correa, E., Turner, M.L. and Goodacre, R. (2015) A tutorial review: Metabolomics and partial least squares-discriminant analysis--a marriage of convenience or a shotgun wedding. *Analytica Chimica Acta* **879,** 10-23.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning*, 2nd edn. Springer, New York, United States of America.

Kristensen, M.R.B. and Vinter, B. (2010) Numerical Python for scalable architectures, *Proceedings of the Fourth Conference on Partitioned Global Address Space Programming Model*, Association for Computing Machinery, pp. 1-9.

Lindgren, F., Hansen, B., Karcher, W., Sjöström, M. and Eriksson, L. (1996) Model validation by permutation tests: Applications to variable selection. *Journal of Chemometrics* **10,** 521-532.

Löfstedt, T. and Trygg, J. (2011) OnPLS—a novel multiblock method for the modelling of predictive and orthogonal variation. *Journal of Chemometrics* **25,** 441-455.

McKinney, W. (2010) Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference***,** 51-56.

Mendez, K.M., Broadhurst, D.I. and Reinke, S.N. (2019a) The application of artificial neural networks in metabolomics: a historical perspective. *Metabolomics* **15,** 142.

Mendez, K.M., Pritchard, L., Reinke, S.N. and Broadhurst, D.I. (2019b) Toward collaborative open data science in metabolomics using Jupyter Notebooks and cloud computing. *Metabolomics* **15,** 125.

Mendez, K.M., Reinke, S.N. and Broadhurst, D.I. (2019c) A comparative evaluation of the generalised predictive ability of eight machine learning algorithms across ten clinical metabolomics data sets for binary classification. *Metabolomics* **15,** 150.

Olden, J.D. and Jackson, D.A. (2002) Illuminating the "black box": a randomization approach for understanding variable contributions in artificial neural networks. *Ecological Modelling* **154,** 135-150.

Olden, J.D., Joy, M.K. and Death, R.G. (2004) An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. *Ecological Modelling* **178,** 389-397.

Pedregosa, F., Ga, Varoquaux, l., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, d. (2011) Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research* **12,** 2825-2830.

Reinke, S.N., Galindo-Prieto, B., Skotare, T., Broadhurst, D.I., Singhania, A., Horowitz, D., Djukanović, R., Hinks, T.S.C., Geladi, P., Trygg, J. and Wheelock, C.E. (2018) OnPLS-based multi-block data integration: A multivariate approach to interrogating biological interactions in asthma. *Analytical Chemistry* **90,** 13400-13408.

Rohart, F., Gautier, B., Singh, A. and Lê Cao, K.-A. (2017) mixOmics: An R package for 'omics feature selection and multiple data integration. *PLOS Computational Biology* **13,** e1005752.

Steinwart, I. and Christmann, A. (2008) *Support Vector Machines*. Springer, New York, United States of America.

Szymańska, E., Saccenti, E., Smilde, A.K. and Westerhuis, J.A. (2012) Double-check: validation of diagnostic statistics for PLS-DA models in metabolomics studies. *Metabolomics* **8,** 3-16.

Theano Development Team (2016) Theano: A python framework for fast computation of mathematical expressions. *arXiv:1605.02688*.

Tran, T.N., Afanador, N.L., Buydens, L.M.C. and Blanchet, L. (2014) Interpretation of variable importance in partial least squares with significance multivariate correlation (sMC). *Chemometrics and Intelligent Laboratory Systems* **138,** 153-160.

van den Berg, R.A., Hoefsloot, H.C.J., Westerhuis, J.A., Smilde, A.K., and van der Werf, M.J. (2006). Centering, scaling, and transformations: improving the biological information content of metabolomics data. *BMC Genomics* **7,** 142.

Virtanen, P., Gommers, R., Oliphant, T., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Walt, S., Brett, M., Wilson, J., Millman, K., Mayorov, N., Nelson, A., Jones, E., Kern, R., Larson, E. and SciPy 1.0 Contributors (2019) SciPy 1.0--fundamental algorithms for scientific computing in python. *arXiv:1907.10121*.

Westerhuis, J.A., Hoefsloot, H.C.J., Smit, S., Vis, D.J., Smilde, A.K., van Velzen, E.J.J., van Duijnhoven, J.P.M. and van Dorsten, F.A. (2008) Assessment of PLSDA cross validation. *Metabolomics* **4,** 81-89.

Wickham, H. (2014) Tidy data. *Journal of Statistical Software* **59,** 1-23.

Wilkins, M.F., Morris, C.W. and Boddy, L. (1994) A comparison of Radial Basis Function and backpropagation neural networks for identification of marine phytoplankton from multivariate flow cytometry data. *Computer Applications in the Biosciences* **10,** 285-94.

Wold, H. (1975) Path models with latent variables: The NIPALS approach, *Quantitative sociology*, Elsevier. 307-357.

Wold, S., Johansson, E. and Cocchi, M. (1993) PLS: Partial least squares projections to latent structures, *3D QSAR in Drug Design: Theory, Methods and Applications.*, Kluwer/Escom, Dordrecht, The Netherlands.

Wold, S., Sjöström, M. and Eriksson, L. (2001) PLS-regression: a basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems* **58,** 109-130.

Xia, J., Broadhurst, D.I., Wilson, M. and Wishart, D.S. (2013) Translational biomarker discovery in clinical metabolomics: an introductory tutorial. *Metabolomics* **9,** 280-299.

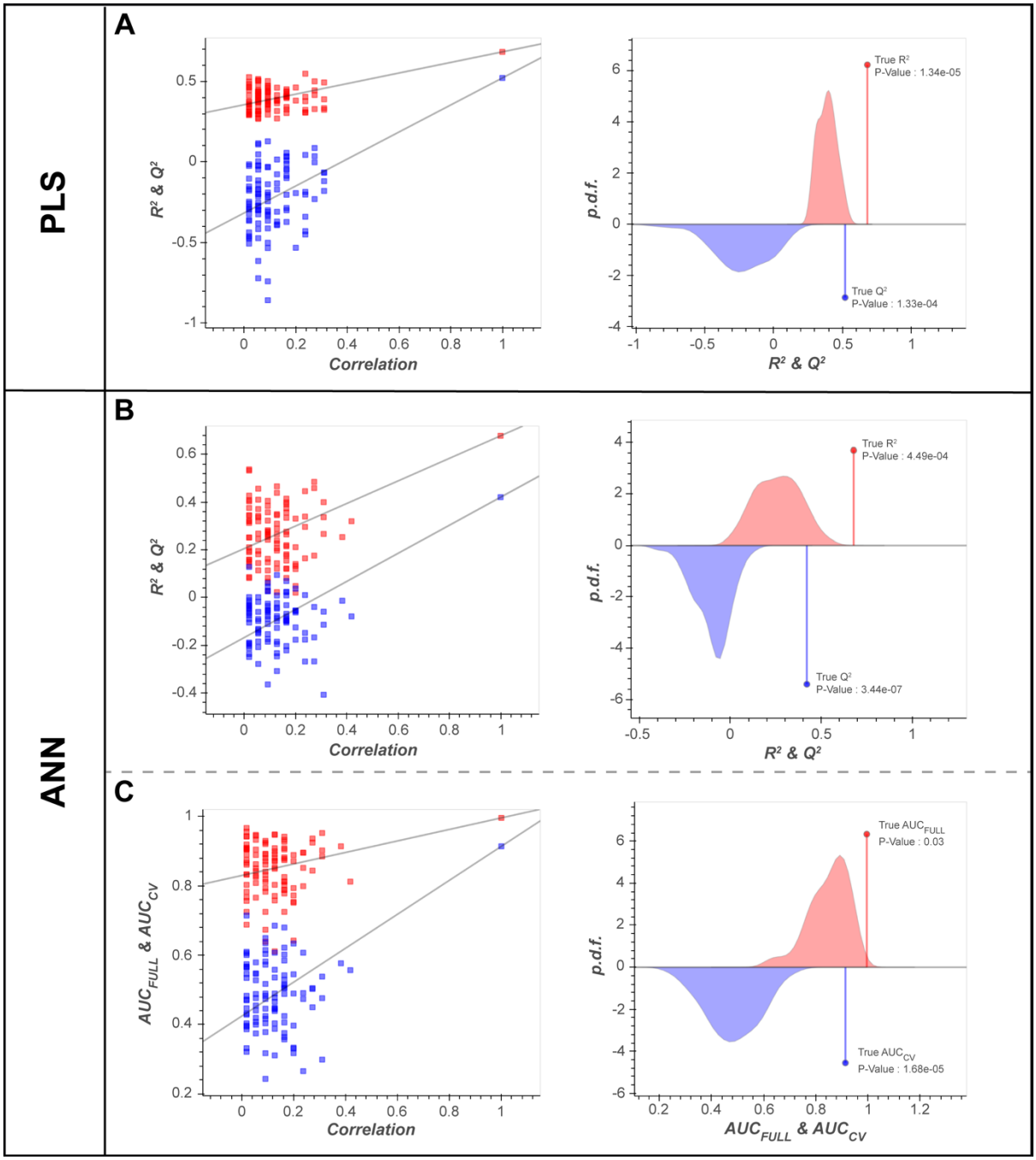
Supplementary Information

List of supplementary html files:

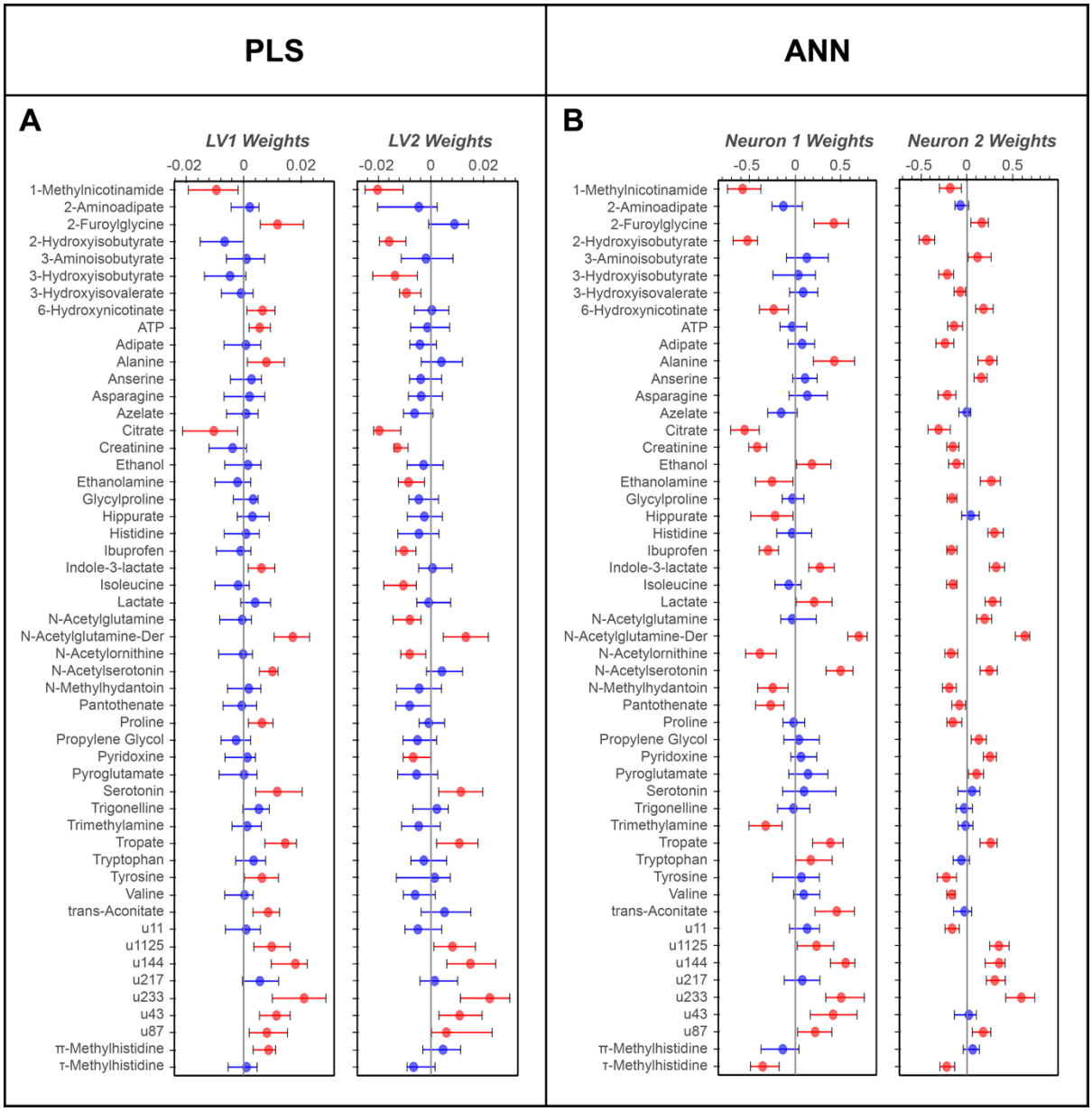
- PLSDA_ST001047.html
    - https://cimcb.github.io/MetabProjectionViz/html/PLSDA_ST001047.html
- ANNSigSig_ST001047.html
    - https://cimcb.github.io/MetabProjectionViz/html/ANNSigSig_ST001047.html
- PLSDA_MTBLS90.html
    - https://cimcb.github.io/MetabProjectionViz/html/PLSDA_MTBLS90.html
- ANNSigSig_MTBLS90.html
    - https://cimcb.github.io/MetabProjectionViz/html/ANNSigSig_MTBLS90.html

List of supplementary figures:

- Supplementary Fig 5.1
- Supplementary Fig 5.2



Supplementary Figure 5.1: Permutation Test. R^2 and Q^2 against correlation between permuted outcomes and original outcomes (left), and probability density functions for R^2 and Q^2, with the R^2 and Q^2 values of the model trained on the original data presented as a ball-and-stick, and p-values from a one-tailed t-test (right). **a** Permutation test figures for PLS. **b** Permutation test figures for ANN. **c** Permutation test figure for ANN using AUC_{FULL} and AUC_{CV} as an alternative metric to R^2 and Q^2.



Supplementary Figure 5.2: Bootstrap weight vectors. Median (and 95% CI) weight vectors of metabolites calculated using BCa (n=100). Red, significant contribution; blue, no significant contribution. **a** LV1 and LV2 weight vectors for PLS. **b** Neuron 1 and neuron 2 weight vectors for ANN.

Chapter Six: Concluding Discussion

6.1. Discussion

The overarching goal of this project was to evaluate the utility of non-linear artificial neural networks (ANNs) in deriving statistical inference from clinical metabolomics data sets. This task was broken down into the specific aims: (1) develop an open source computational framework to allow for interactive modelling and assessment of common machine learning methods specifically tailored for metabolomics studies; (2) compare the generalised predictive ability of non-linear ANNs against alternative machine learning models for binary classification in clinical metabolomic studies; (3) investigate visualisation and variable importance techniques for artificial neural networks that are comparable to current linear projection methods; and (4) critically compare any novel ANN visualisation and variable importance techniques against existing linear projection workflows publicly available metabolomics data sets.

6.1.1. Open Data Science

There is a 'reproducibility crisis' across scientific disciplines highlighting the increasing need for more transparency (Baker, 2016). To align with FAIR data principles (Wilkinson *et al.*, 2016), the dissemination of data analysis steps need to extend beyond a brief description in the methods section. A solution, as described in Chapter Three, is the use of 'computational lab books' such as Jupyter Notebooks (Kluyver *et al.*, 2016). Jupyter Notebooks allow the user to add descriptions of the methodology alongside the code, results, and figures of the data analysis workflow. These 'computational lab books' can be included as a supplemental file to the

manuscript, thereby enabling transparent dissemination and re-use. When combined with a data repository (e.g. GitHub) and an open cloud-based deployment service (e.g. Binder), transparency and ease of re-use are further enhanced. A basic framework and experiential learning tutorials are provided in Chapter Three. It is a challenge for busy practising researchers to adopt these practices towards open data science. However, by providing a tutorial that requires little or no computational expertise, we encourage the broad use of open frameworks within the metabolomics community. This strength of open frameworks is highlighted in Chapter Four and Chapter Five. Without the need for any software installation, readers can re-analyse and re-use of the provided 84 computational workflows.

6.1.2. Generalised Predictive Ability of ANNs

PLS, a linear-based projection method, is the current gold standard ML in metabolomics (Gromski *et al.*, 2015). As biological data are often non-linear (Mosconi *et al.*, 2008), we hypothesised that metabolomic data may have a non-linear latent structure. A non-linear latent structure would make non-linear ML methods more appropriate (i.e. higher predictability) than linear ML methods. Based on the 10 clinical metabolomic datasets in Chapter Four, there was only marginal improvement in predictive ability for ANNs over PLS across all data sets. The use of out-of-bag bootstrap confidence intervals provided a measure of uncertainty of model prediction such that the quality of metabolomics data was observed to be a bigger influence on generalised performance than model choice. Using the principle of Occam's razor, the simplest method (i.e. linear ML method such as PLS-DA) can often be the preferred method. However, there may be clinical metabolomic datasets (not included in this study) with more complex biological questions that clearly warrant the use of non-linear ML methods such as ANN. Thus,

enabling the effective application of ANN to metabolomics data sets can only enrich the available set of data science tools available to the research community.

By providing the 80 data analysis workflows in supplementary, we encourage re-analysis and re-use by the reader. The open source interactive framework developed for this thesis is designed to be reused and be easily accessible. Any new data set with a binary outcome can be applied rapidly to all ten machine learning methods, and where non-linear modelling is appropriate, the supplied non-linear supplementary workflows (for example, with ANN or SVM-RBF) can be copied, repurposed and published by the user.

6.1.3. Statistical Inference of ANNs

ANNs have been used in metabolomics since the early 1990s (Goodacre and Kell, 1993; Goodacre *et al.*, 1992, 1993); however, its usage has stagnated while the use of PLS has continued to grow (Fig. 2.4; Chapter Two). This largely stems from difficulty in deriving and visualising statistical inference for ANNs (Goodacre, 2003), which is still an ongoing issue (Samek *et al.*, 2017; Zhang and Zhu, 2018; Zhang *et al.*, 2018). The approach we proposed to derive and visualise statistical inference is based on the *structural equivalence* between a single hidden layer linear ANN and PLS (described in Chapter Two). In Chapter Five, we successfully migrated the PLS-DA optimisation, evaluation, and interpretation workflow to a single layer ANN. This workflow includes *functional equivalent* feature importance measures to identify significant metabolites (biomarkers).

To allow for a comparison between the PLS-DA and proposed ANN workflow, two clinical metabolomic datasets from Chapter Four were included in this study. The second dataset was

primarily included to demonstrate that these workflows can be re-used for other studies, provided data is in, or converted to, the described standardised format used in Chapter Three, Four, and Five. Therefore, this study provides an open framework for other metabolomics community members to apply to their own practice to derive and visualise statistical inference for non-linear single hidden layer ANNs, as well as PLS-DA. Combined with the open data analysis workflows in Chapter Three and Four, we hope to ease of re-usability and extension of this workflows will help the community adopt more transparent practices, and align with FAIR data principles. Additionally, the success of this approach for single hidden layer ANNs, it opens the door to the investigation of extending statistical inference techniques to more complex models including multi-block models.

6.1.4. Conclusion

In this project, we have shown that non-linear single hidden layer ANNs **can** be used to derive statistical inference from clinical metabolomics data sets. However, non-linear ML methods were **not necessary** across the ten data sets in this study (based on Occam's razor). This was unexpected given the highly dimensional and complex covariance structure of metabolomic data. There may be data sets not included in this study that have sufficiently complex non-linear latent structure to warrant to use non-linear ANNs. By providing an open framework, such data sets within the metabolomics community can now be appropriately analysed with relative ease and published with transparency. Importantly, this project (with the provided workflows) give an indication of the necessary rigour and appropriate steps in optimising and evaluating ML. All ML models are prone to overfitting, especially more complex (non-linear) ML models. Overfit (or underfit) ML models are not generalisable, leading to incorrect statistical inference, and therefore, incorrect biological interpretation. Using Jupyter

Notebooks, we have provided extensive prose for each step in the workflow to help mitigate incorrect modelling and misinterpretation within the metabolomics community. Moving forward, this framework provides a foundation for open data science, which is becoming increasingly important as we investigate more complex ANN architectures.

6.2. Future Perspectives

In other research areas such as genomics, the use of ANNs has grown exponentially and is now well established (Fig. 2.4). Based on this trajectory, it is likely that there would be continued growth and interest for ANN application in other 'omics sciences including metabolomics and integrative systems-biology. ANNs have a high degree of flexibility allowing for various types of network architectures. One area that is yet to be exploited is the use of ANNs to model the covariance data structures within and between multiple data blocks. These blocks can include multiple analytical platforms, multiple biofluids, or multiple omics (i.e. integrative systems-biology).

The inherent flexibility of ANNs allows for various model architecture to model multi-block data (Fig. 2.5B and Fig. 2.5C; Chapter Two). This allows ANNs to have a single model architecture and model non-linear covariance data structures, in contrast to currently used hierarchical applications of linear projection-based models. Several publications use a variant of the architecture described in Fig. 2.5B (Bica *et al.*, 2018; Chaudhary *et al.*, 2018; Huang *et al.*, 2019; Sharifi-Noghabi *et al.*, 2019). These publications focus on the predictive ability of the model but provide no method for statistical inference. For multi-block ANNs to be a viable alternative to hierarchical linear projection-based models, there is a need to incorporate visualisation and feature importance techniques. With the success of migrating the statistical

inference approaches to a single hidden layer ANN (Chapter Five), there is high potential to extend this approach to multi-block ANN models. Key to this approach is the visualisation of neuron scores prior to transformation (i.e. before activation function), and the use of neuron connection weights (input-hidden and hidden-output weights) to derive feature (metabolite) contributions to the overall model and to each neuron. While multi-block ANNs can be extended to many hidden layers (i.e. deep learning), to maintain transparency the depth may need to be limited. If deriving and visualising statistical inference is extended to multi-block ANNs, it is likely to become standard in modelling multi-block data for metabolomics and integrative systems-biology.

References

Baker, M. (2016) 1,500 scientists lift the lid on reproducibility. *Nature* **533,** 452-454.

Bica, I., Velickovic, P., Xiao, H. and Li, P. (2018). Multi-omics data integration using cross-modal neural networks. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.*, pp. 385-390.

Chaudhary, K., Poirion, O.B., Lu, L. and Garmire, L.X. (2018) Deep learning-based multi-omics integration robustly predicts survival in liver cancer. *Clinical Cancer Research* **24,** 1248-1259.

Géron, A. (2017) *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Inc., California, United States of America.

Goodacre, R. (2003) Explanatory analysis of spectroscopic data using machine learning of simple, interpretable rules. *Vibrational Spectroscopy* **32,** 33-45.

Goodacre, R. and Kell, D.B. (1993) Rapid and quantitative analysis and bioprocesses using pyrolysis mass spectrometry and neural networks: application to indole production. *Analytica Chimica Acta* **279,** 17-26.

Goodacre, R., Kell, D.B. and Bianchi, G. (1992) Neural networks and olive oil. *Nature* **359,** 594-594.

Goodacre, R., Kell, D.B. and Bianchi, G. (1993) Rapid assessment of the adulteration of virgin olive oils by other seed oils using pyrolysis mass spectrometry and artificial neural networks. *Journal of the Science of Food and Agriculture* **63,** 297-307.

Gromski, P.S., Muhamadali, H., Ellis, D.I., Xu, Y., Correa, E., Turner, M.L. and Goodacre, R. (2015) A tutorial review: Metabolomics and partial least squares-discriminant analysis--a marriage of convenience or a shotgun wedding. *Analytica Chimica Acta* **879,** 10-23.

Huang, Z., Zhan, X., Xiang, S., Johnson, T.S., Helm, B., Yu, C.Y., Zhang, J., Salama, P., Rizkalla, M., Han, Z. and Huang, K. (2019) SALMON: Survival analysis learning with multi-omics neural networks on breast cancer. *Frontiers in Genetics* **10,** 166.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S. and Willing, C. (2016) Jupyter Notebooks – a publishing format for reproducible computational workflows in Loizides, F.a.S., Birgi (Ed), *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, IOS Press, pp. 87 - 90.

Mosconi, F., Julou, T., Desprat, N., Sinha, D.K., Allemand, J.-F., Croquette, V. and Bensimon, D. (2008) Some nonlinear challenges in biology. *Nonlinearity* **21,** 131-147.

Samek, W., Wiegand, T. and Müller, K.-R. (2017) Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv:1708.08296*.

Sharifi-Noghabi, H., Zolotareva, O., Collins, C.C. and Ester, M. (2019) MOLI: Multi-Omics Late Integration with deep neural networks for drug response prediction. *Bioinformatics* **35,** i501–i509.

Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L.B., Bourne, P.E., Bouwman, J., Brookes, A.J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C.T., Finkers, R., Gonzalez-Beltran, A., Gray, A.J.G., Groth, P., Goble, C., Grethe, J.S., Heringa, J., 't Hoen, P.A.C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S.J., Martone, M.E., Mons, A., Packer, A.L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M.A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J. and Mons, B. (2016) The FAIR guiding principles for scientific data management and stewardship. *Scientific Data* **3,** 160018.

Zhang, Q.-s. and Zhu, S.-c. (2018) Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering* **19,** 27-39.

Zhang, Z., Beck, M.W., Winkler, D.A., Huang, B., Sibanda, W., Goyal, H. and written on behalf of, A.M.E.B.-D.C.T.C.G. (2018) Opening the black box of neural networks: methods for interpreting neural network models in clinical applications. *Annals of translational medicine* **6,** 216-216.

Appendix 1: Endorsement of Author Contributions (Publication 1)

Publication: Mendez, K.M., Broadhurst, D.I., Reinke, S.N. (2019) The Application of Artificial Neural Networks in Metabolomics: A Historical Perspective. *Metabolomics* **15,** 142.

Author Contributions: All authors conceived of the idea. KMM wrote the manuscript. DIB and SNR edited the manuscript.

I, as a co-author, endorse the above stated contribution of work:

David I Broadhurst

Stacey N Reinke

Appendix 2: Endorsement of Author Contributions (Publication 2)

Publication: Mendez, K.M., Pritchard, L., Reinke, S.N., Broadhurst, D.I. (2019) Toward Collaborative Open Data Science in Metabolomics using Jupyter Notebooks and Cloud Computing. *Metabolomics,* **15,** 125.

Author Contributions: All authors contributed equally to this work. KMM developed the code for the Python cimcb-lite package. KMM & DIB developed the tutorial Jupyter notebook code. LP and SNR edited the learning tutorials to align to current pedagogical best practices. KMM wrote the initial draft of the review section. All authors edited the manuscript.

I, as a co-author, endorse the above stated contribution of work:

Leighton Pritchard

Stacey N Reinke

David I Broadhurst

Appendix 3: Endorsement of Author Contributions (Publication 3)

Publication: Mendez, K.M, Reinke, S.N., Broadhurst, D.I. (2019) A Comparative Evaluation of the Generalised Predictive Ability of Eight Machine Learning Algorithms across Ten Clinical Metabolomics Data Sets for Binary Classification. *Metabolomics,* **15,** 150

Author Contributions: All authors conceived of the idea. KMM developed the Jupyter notebooks. KMM developed the Python packages. KMM performed the optimisation and evaluation of all models. DIB & SNR independently checked each model. KMM wrote the manuscript. DIB and SNR edited the manuscript.

I, as a co-author, endorse the above stated contribution of work:

Stacey N Reinke

David I Broadhurst

Appendix 4: Endorsement of Author Contributions (Publication 4)

Publication: Mendez, K.M, Broadhurst, D.I., Reinke, S.N. (2019) Migrating from Partial Least Squares Discriminant Analysis to Artificial Neural Networks: A Comparison of Functionally Equivalent Visualisation and Feature Contribution Tools using Jupyter Notebooks. *Metabolomics,* **16,** 17

Author Contributions: All authors conceived of the idea. KMM and DIB developed the software. KMM wrote the manuscript. DIB and SNR edited the manuscript.

I, as a co-author, endorse the above stated contribution of work:

David I Broadhurst

Stacey N Reinke

This page is intentionally left blank.