Fast derivation of Shapley based feature importances through feature extraction methods for nanoinformatics

IOP Publishing Mach. Learn.: Sci. Technol. 2 (2021) 035034 https://doi.org/10.1088/2632-2153/ac0167

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
9 March 2021

REVISED
4 May 2021

ACCEPTED FOR PUBLICATION
14 May 2021

PUBLISHED
14 July 2021

PAPER

Fast derivation of Shapley based feature importances through feature extraction methods for nanoinformatics

Tommy Liu* and **Amanda S Barnard**

School of Computing, Australian National University, Canberra, ACT 2601, Australia
* Author to whom any correspondence should be addressed.

E-mail: tommy.liu@anu.edu.au

Keywords: feature ranking, Shapley values, graphene oxide, PCA, dimension reduction, nanoinformatics

Supplementary material for this article is available online

Abstract

This work presents an alternative model-agnostic attribution method to compute feature importance rankings for high dimensional data requiring dimension reduction. We make use of Shapley values within the Shapley additive explanation framework to determine the importance values of each of the feature in the data set. We then demonstrate that it is possible to significantly reduce the computational complexity of ranking features in high dimensional spaces by first applying principal component analysis. This transformation into lower dimensional spaces in conjunction with our normalisation approach does not yield a significant loss of information when performing feature selection tasks beyond a threshold. The efficacy of our approach is demonstrated on several examples of nanomaterial data, in particular graphene oxide. Our approach is ideal for the applied physical science communities where datasets are of high dimensionality and computational complexity is a matter for concern.

1. Introduction

In fields of applied sciences it is highly desirable to use interpretable machine learning methods that give additional insights into predictions that can be used to revise experimental or computational research strategies. This is particularly important when studying real world applications, and when ever humans need to make decisions about physical systems. This includes researchers designing new materials that underpin numerous technologies and applications in energy, electronics, environment and health. Materials design using machine learning methods (an area of materials informatics) typically involves a large number of variables that can be tuned in order to improve performance, provided researchers can interpret the correlations from the models [1].

Schleder *et al* [2] states that model interpretability decreases as the complexity of the model increases, which is derived from Kuhn [3] and motivates the need for simple models with low dimensionality, as well as models that are easier to understand. However, with datasets in the applied sciences it is difficult to build simple models to accurately model phenomena using data containing a significant number of features [4].

In recent years it has been demonstrated that Shapley values provide a good basis for understanding the decisions made by models [5], but the decisions made by the model are not the only decisions made in a data analysis workflow. Feature engineering involves numerous researcher decisions that can have a significant impact on the outcome, including dimension reduction to simplify models and reduce the computational intensity. It is difficult to interpret how data is transformed through some of the more performant dimensional reduction methods such as Universal Manifold Approximation and Projection (UMAP) [6], and the tuning of particular parameters may further increase the complexity of such a task and model. This makes feature selection methods more desirable, but entirely data-driven selection can be complicated.

Feature selection methods are highly varied and model performance depends on the data at hand. Formally, given a data set $X \in \mathbb{R}^{n \times m}$ and target labels $Y \in \mathbb{R}^{n}$, feature selection methods seek to find a subset $Z \in \mathbb{R}^{n \times k}$ such that $k << m$ which can fulfil similar machine learning needs as the original data X. While

there exist a significant body of work on feature selection methods [7–9], there is no obvious 'best' approach to any one task. Currently there are two main approaches to feature selection: supervised, and unsupervised. In unsupervised feature selection, information about the data itself, such as statistical correlation and entropy, are used to select the optimal features; in supervised feature selection indicators that relate the data to the target features are used, such as importance to some given model. Supervised approaches tend to be superior in the sense that predictive models typically improve based on feedback, but at a risk of overfitting to the data. Another issue arises in determining the optimal number of features to retain, in addition to which features to retain. While it may be possible to increase the predictive performance in a task with a minimal number of features, this comes at the cost of losing the opportunity to understand other factors such as the marginal contributions of less important features. Features that a model deems less important may be important to a researcher, or vice versa.

Such a feature selection landscape presents a problem for applied sciences such as materials informatics and determining the best approach is not straightforward. This brings us back to the imperative that simple and effective approaches are needed. A popular approach involves feature ranking based on feature importance characteristics, which are available to a large number of models implemented in numerous scientific computing packages. Feature ranking fulfils the need for availability, simplicity and interpretability, but the results are model specific and lack the generality sought by researchers in the application domains. Therefore, in this paper we present an alternative approach to feature rankings using Shapley values which can act as a general indicator for feature importance for any model, and can be used consistently to a series of models during model selection and evaluation. While such methods have been employed before [10], we present an alternative approach that incorporates dimension reduction which significantly reduces the computational complexity of the problem.

To demonstrate our feature selection and attribution method apply our method to two computational data sets of graphene oxide nanomaterials. Nanomaterials data typically contains few samples but are of high dimensionality [1], due to the limiting cost of synthesis experiments, and the comparatively low cost of materials characterization. For example, the simulation of experimentally realistic graphene oxide using first principles electronic structure methods consumes months of supercomputing resources, but the statistical analysis of the outputs required for feature extraction takes only a few hours. This leads to the 'curse of dimensionality', but not all of the features are actually informatic. In our case study data we seek to predict the energy of the Fermi level of the graphene oxide samples, based on the physical structure, since controlling this property in the lab opens up significant opportunities to advance materials science research. The Fermi level of a material is partially responsible for electrical conduction and important to many applications in energy, electronics, and sensing. The properties of the data set, and the properties of the material, make this an ideal and important exemplar.

In section 2 we present background information regarding the data set along with additional information on the methods discussed above. In section 3 we describe the original methodology employed by this work along with some justification of why these methods were selected. In section 4 we present our evaluation methodology and introduce our data. In section 5 we present and discuss the results of our method.

2. Background

Shapley values [11] are a measure of marginal contributions from a given feature. Shapley values measures feature importances based on the relative contribution of a feature across all possible subsets of the data [11] and is formally defined via a value function ν over the data instances given by:

$$\phi_i(\nu) = \sum_{S \subset F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)] \tag{1}$$

where f_S is a model trained without the particular feature, $f_{S\cup\{i\}}$ is a model trained with that particular feature, and $x_S, x_{S\cup\{i\}}$ are the values of the particular feature inputs. There are several frameworks which directly use or approximate this classic Shapley equation in order to provide model explanations.

Shapley additive explanation (SHAP) is a framework provides a unified measure of feature contributions which discusses the class of additive model explanations. Given a model $f(z)$, SHAP provides an explanation of how features additively combine to offset the conditional expectation of the model $E[f(z)|z = x]$ from the base value $E[f(z)]$. In particular, SHAP provides several different implementations for various models including models for linear regression and decision trees/random forests [5, 12].

SHAP values also provide a model agnostic method of computing feature importances, since we only need to consider how much of a contribution a particular feature has across each of the data instances in X.

The following algorithm is used to calculate these importances:

$$\varphi_i = \frac{1}{n}\sum_{j=1}^{n} |\mathbf{\Phi}_{ji}| \tag{2}$$

where φ_i denotes the feature importance of the ith feature, $\mathbf{\Phi}_{ji}$ is the SHAP value of the ith feature of the jth data point.

From these feature importances derived from the Shapley values, we can make use of contribution-based feature selection methods in a model agnostic manner. Cohen *et al* introduces the CSA algorithm making use of an iterative strategy to select or remove features from a working set [10]. We find that in many cases, one does not need to apply the iterative strategy but rather an in-place approach is sufficient.

Principal component analysis (PCA) is a popular and well known dimension reduction method that captures the directions of greatest variance in the data [13]. Consider a set of observations $X = \{x_1, x_2, \ldots, x_n\}$, each observation of dimensionality M, then the standard PCA seeks to find a projection onto a subspace of dimension $N < M$ such that the variance of the data projections is maximised (or least-squares error is minimised). This projection is defined by an orthonormal basis which consists of a set of unitary basis vectors which are called the principal components. There are multiple methods of computing these principal components however the singular value decomposition (SVD) is the most popular [4].

An alternate interpretation of PCA is that of a low rank model [14] which interprets PCA as a optimisation problem in which the following problem must be solved:

$$\text{Minimise} \qquad ||X - UZ||_{\mathcal{F}}^2. \tag{3}$$

We make use of this interpretation to gain further insights to formulate our attribution method for original features whilst making use of PCA.

3. Methodology

In this section we present an overview of our attribution and evaluation methods. A feature selection workflow making use of Shapley values typically involves first training a (possibly less complex) model on the original data [5]. From this model, the Shapley values are computed and the features with the largest Shapley values are selected. The strategy for selecting the optimal number of features can vary based on the desired outcome, for example aiming for the best model performance, or finding the smallest number of features to obtain a sufficiently accurate model. In our approach, to further reduce the need for computation, we make use of a novel method which we call low rank attribution (LRA). Instead of training a model from the original data, we first apply PCA so that model training can be significantly faster, the Shapley values of these components are attributed to original features using equation (4).

3.1. Low rank attributions (LRAs)

Motivated by the specific form of low rank representation of the PCA problem discussed in section 1. If we must first apply PCA to our data, then normalise contributions by some squared factor of all their component values. Therefore, given a set of attributions ψ and PCA components γ, the attribution of the ith feature denoted α_i is given by:

$$\alpha_i = \frac{\sum_{j=0}^{k} (\psi_j \cdot \gamma_j)_i}{\sqrt{\frac{1}{k}\sum_{j=0}^{k} \gamma_{ji}^2}} \tag{4}$$

which is exactly the total contributions of a particular feature across all components, normalised by the root mean squared component value across all components. This allows feature importances of the original data to be computed, without requiring that training to be carried out on the original data. Instead it is sufficient to train on the projection which may be of significantly smaller dimensionality than the original data.

4. Evaluation

To evaluate our results, we conducted subset searches using the algorithm described in figure 1 on two graphene oxide datasets. We compare and plot the cross-validated scores achieved by our approach and the baseline consisting of feature importances derived from the original data. Fifty components were used for the PCA transformation of our LRA approach. We also provide learning curves showing training and testing set RMSE scores which demonstrates that no over-fitting is occurring in our experiments.

Algorithm 1: Best Subset Search

Input: L: ML Algorithm; F: Feature set; Y: Target labels; $\boldsymbol{\phi}$: SHAP values; t: Maximum number of features

Result: S: Feature subset with greatest contributions

$S = \{\}$;
$\Lambda = \{\}$;
$i = 0$;
while $i < t$ **do**
  Set S to the i features in F with the largest ϕ values;
  Evaluate S according to $L(S, Y)$;
  Append score to Λ;
  i++;
end
Return S with best score L;

Figure 1. Algorithm for selecting best subset based on SHAP values.

Our testing consists of two cleaned datasets containing key characteristics of graphene oxide samples. Data generation and feature extraction was not conducted as part of this study. Dataset A consists of 390 physical features and 1617 samples [15]. Dataset B consists of a subset of 673 features and 776 samples [16]. From the number of samples and number of features, we can see that the curse of dimensionality will apply since the ratio of samples to features is non-ideal for machine learning applications. Further information about these datasets is presented in the supplementary materials (available online at stacks.iop.org/MLST/2/035034/mmedia) and in the meta data available on the repositories. Additional results using a third nanomaterials data set (gold nanoparticles) is also provided in Supplementary Materials for comparison.

We make use of several regression methodologies for the model L in our model evaluation for Algorithm 1, and is provided by the scikit-learn python package [17]. In particular we make use of: Random Forests, Decision Tress, Ridge and Kernelized Ridge Regression, and multi-layer perceptrons.

In our experiments decision tress and random forests are built using mean squared error as the impurity metric and create fully grown un-pruned trees. We make use of both cross-validation (CV) and testing set evaluation of root mean squared error (RMSE) scores in order to evaluate our models. We use the SHAP Github provided, this package makes use of various 'explainers' which implement both model agnostic and specific methodologies to explain machine learning models. We derive SHAP values from random forests using TreeSHAP which is a model specific method provided by the SHAP package for decision trees and random forests [12]. Alternative explainers such as the KernelExplainer, Explainer, and LinearExplainer are used for the alternative models used.

While TreeSHAP has a fast method to approximate the contribution values directly from the tree structures, it only works for the cases of tree based machine learning models. SHAP does provide both model specific and agnostic implementations to derive SHAP values from various types of models which we make use of for non tree-based models. However, it is the case that many of these explainers do not scale well in the number of features in the data resulting in poor performance. We evaluate the data efficiency of both model training time, and the derivation of feature importances through Shapley values in section 6.

5. Results and discussion

We first consider our problem in the context of supervised machine learning by considering different training-test sizes for our dataset A. Figure 2 demonstrates the random forest model performance for various proportions of the training and testing sets. We see that the score on both sets tends to fall as the training set proportion increases until approximately 85%.

5.1. Feature selection using SHAP values and low rank attribution

We first compare our approach using testing set scores for feature ranking against the baseline of feature ranks derived from the original data. In figure 3 we see that our approach yields comparable performance past certain thresholds which are different for the two tested training set sizes.

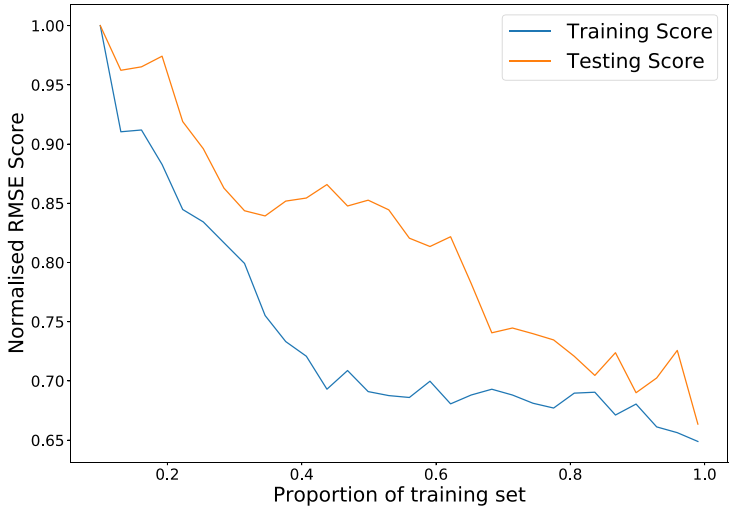


Figure 2. Training and testing scores for various training set proportions of dataset A for the Random Forest model ($n = 1000$).

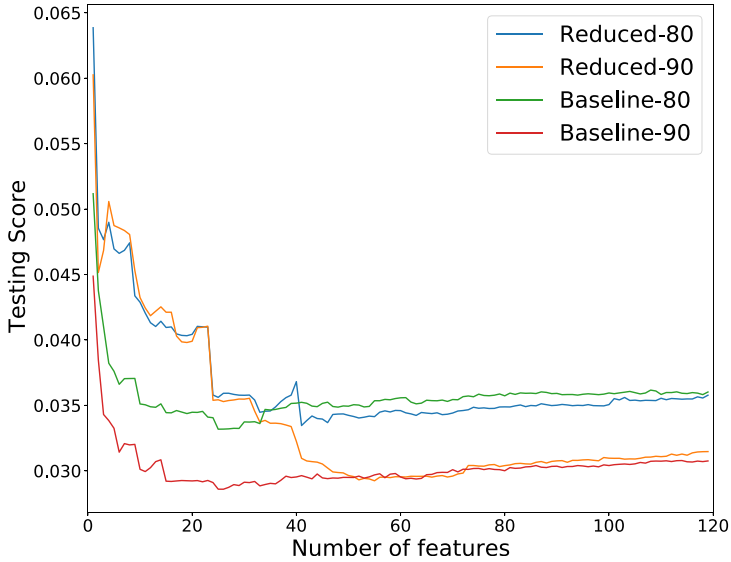


Figure 3. Testing set scores derived from LRA against baseline approach.

In figure 4 we compare our approach using cross-validation scores against feature importances derived from the original features using Dataset A. We again see that beyond a certain threshold (42 features) the performance difference between using feature contributions computed using the original data and the reduced data is insignificant. In fact, the minimum values achieved by the two methods is exactly the same. The same approach for Dataset B can be seen in figure 5, where the minimum value achieved by the reduced method is slightly higher than that of the baseline model, but in the fourth decimal place. These curves demonstrate that the effect of adding the most important features has a significant impact reducing performance, and conversely past a certain point adding additional (less important) features has little effect on model performance. Full curves containing the entire feature space are provided in supplementary materials.

The amount of overlap in the feature-sets obtained by the two different approaches can be seen in figure 6 for Dataset A. It can be seen that the degree of overlap between the two methodologies is less than half yet, when training a model, the performance is the same. This can be explained by the high degree of correlation between important features selected within each subset, the average correlation between the top ten features in one set and the top ten features in the other is 0.793.

We now consider the peak performances obtained by various models trained using features derived from LRA, and feature importances trained on the original data. We express the best CV and testing RMSE scores

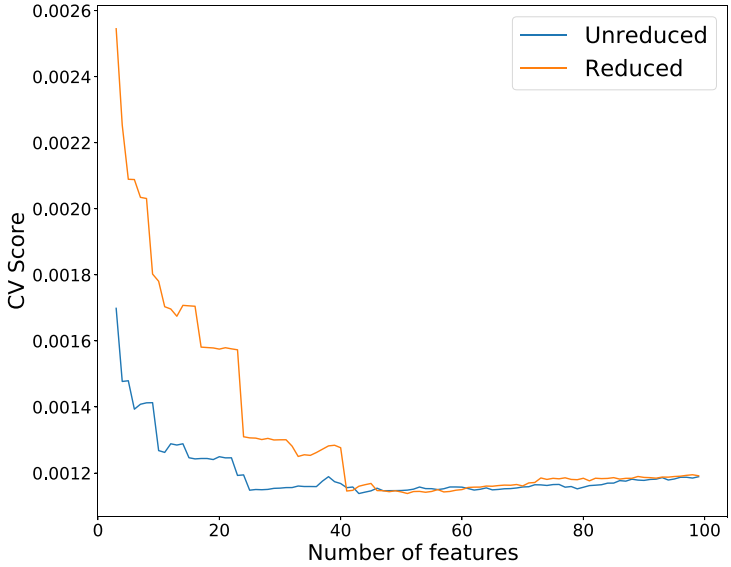


Figure 4. Cross validated results of algorithm 1, with (reduced) and without (unreduced) application our LRA approach applied to Dataset A using a Random Forest Regressor with 1000 trees.

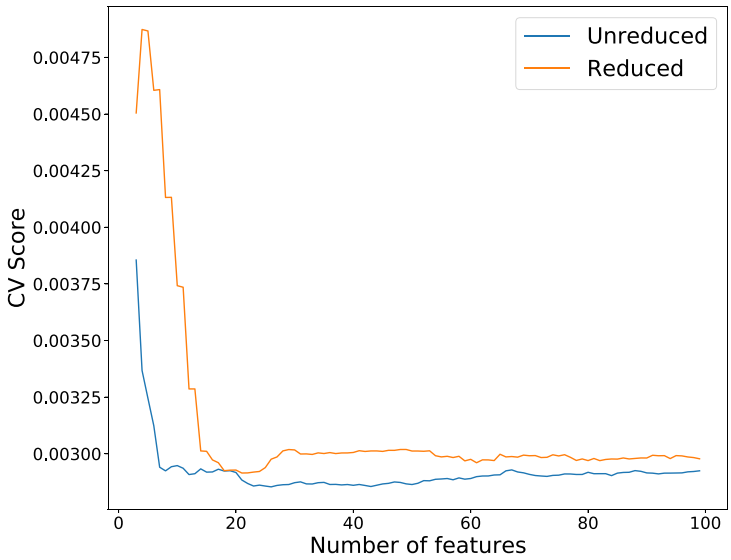


Figure 5. Cross validated results of algorithm 1, with (reduced) and without (unreduced) application of our LRA approach applied to Dataset B using a Random Forest Regressor with 1000 trees.

obtained in table 1. We see that the in majority of models our approach does not result in significantly worse peak performance, and in some cases even performs better (linear regression). The exception when our approach performs significantly worse seems to be that of both cases of Ridge Regression, it may be the case that the penalties imposed by these approaches results in a sub-optimal ranking of features when the number of samples is very low (as the case with dataset B). Note that these best scores for each model occur for different number of most important features selected, and we only consider the 100 most important features. In figure 7 we consider the entire training curve with consideration to CV and RMSE scores. From these curves we can see both the effects of including the most important features, but also the effects of removing the least important features. We see that the effects of the least important features is relatively consistent across both the original and LRA approaches to feature ranking.

While our approach yields similar performances to a model trained on the original data past a certain number of features in our subset search, it is difficult to determine this threshold without running an exhaustive search as we have here since it varies based on the data and the choice of machine learning model. While this limits the practicality of our approach when we want a very small number of features to analyse (i.e. < 20), there are strategies which can be employed in order to determine the best value, such as simulated annealing where the optimal number of features is treated as a hyper-parameter. Such an approach can also

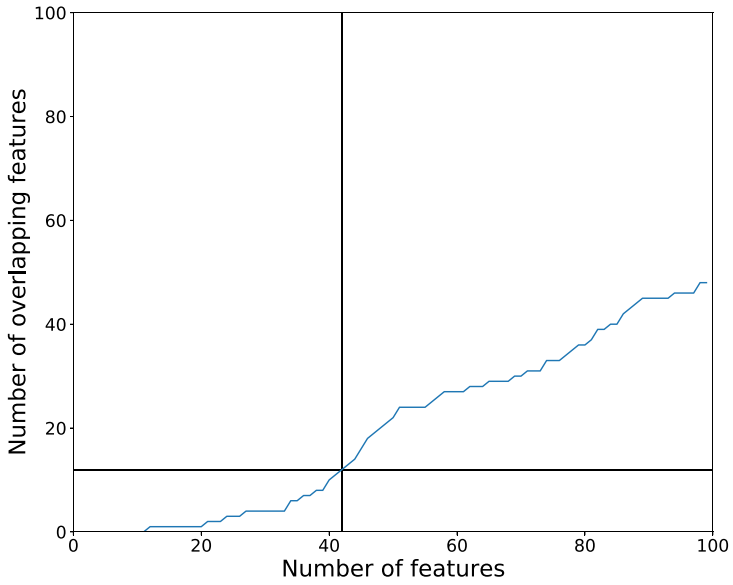


Figure 6. Number of overlapping features when using the k most important features for the two approaches for computing feature importances.

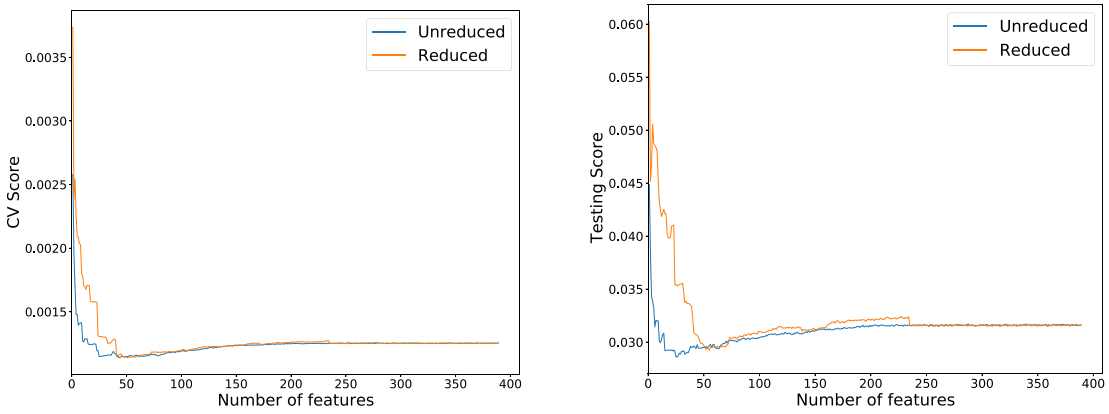
Table 1. Table showing best scores achieved by using feature importances derived from original data (O), compared with feature importances derived from our approach (LRA) on both dataset A and B.

A	Best CV Score (O)	Best CV Score (LRA)	Best Testing Score (O)	Best Testing Score (LRA)
Decision Tree	0.00 196	0.00 203	0.0374	0.0388
Random Forest ($n = 1000$)	0.00 113	0.00 114	0.0337	0.0348
Linear Regression	0.00 182	0.000 730	0.0280	0.0251
Ridge Regression	0.00 123	0.00 165	0.0338	0.0369
Kernel Ridge Regression	0.00 133	0.00 145	0.0340	0.0359
Multi-Layer Perceptron*	0.00 278	0.00 202	0.0505	0.0416
B	**Best CV Score (O)**	**Best CV Score (LRA)**	**Best Testing Score (O)**	**Best Testing Score (LRA)**
Decision Tree	0.00 379	0.00 376	0.0634	0.0606
Random Forest ($n = 1000$)	0.00 287	0.00 305	0.0602	0.0579
Linear Regression	0.00 383	0.00 292	0.0548	0.0473
Ridge Regression	0.00 295	0.00 360	0.0544	0.0588
Kernel Ridge Regression	0.00 293	0.00 422	0.0558	0.0637
Multi-Layer Perceptron*	0.00 387	0.00 412	0.0613	0.0632

*4 layers, 128, 64, 32, 8, neurons in the layers, trained using the Adam optimiser with an adaptive learning rate

be taken advantage of along with other feature selection methodologies such as the Boruta algorithm where repeated simulations are used to determine which features to retain [18]. In particular, our approach can be used with a highly conservative estimate of the number of features to keep (such as $3\sqrt{n}$) and further refinements to the feature set through algorithms such as Boruta can be made.

A drawback of our approach is the current limitation to using PCA for its transformation into a lower dimensional space. While PCA may work well for some datasets, there are limited applications since it is a orthogonal linear transformation into a subspace and makes many assumptions about the data. Furthermore, despite all our experiments being carried out using SHAP values, our LRA approach can be applied to any metric which computes feature importances including: decrease in impurity; feature permutation importance [19]; and SAGE values [20]. Both represent opportunities for further research.



(a) CV score against number of features

(b) Testing RMSE score against number of components

Figure 7. Full testing set (90%) and CV scores for the Random Forest model ($n = 1000$) for dataset A.

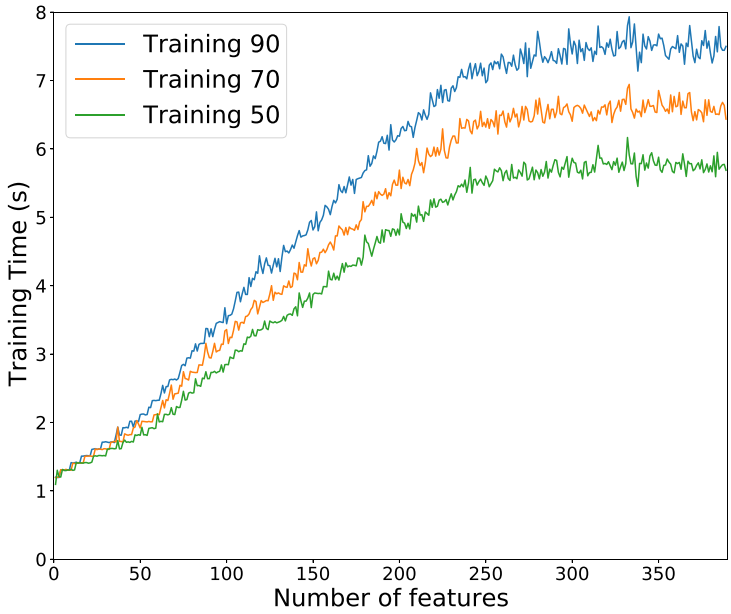


Figure 8. Timing curves for testing set scores as a function of training set size (% number shown in legend). Running time computed with AMD R9-3900X with 24 threads.

6. Data efficiency

The benefits of our approach comes in two aspects, both with regards to the feature ranking process: firstly by reducing the number of features, the learning algorithm can see a significant speed-up for model training (i.e. Random Forests); and secondly for algorithms which do not have an inherent concept of 'feature importance' (such as kernel ridge regression) the time it takes to explain the data (using SHAP explainers) can be significantly sped up.

Consider a learning algorithm that scales with the number of features such as a Random Forest with time complexity order $O(kmn \log n)$. For datasets with a significant number of features it may be slow or possibly infeasible to train a model on the original data. In these cases, by reducing the feature set with PCA and making use of our LRA approach, training time can be significantly faster. While training fully grown trees may be unrealistic in real world scenarios, the time complexity of such a random forest is of the order $O(kmn \log n)$ where k,m,n are the number of trees, features, and samples respectively [21]. In our case study by reducing the number of features m from 390 to 50 using our LRA method the complexity is reduced by $\times$ 7.8. In figure 8 we see the actual runtimes of the random forest model for various feature sizes for different training set sizes. We see that the speedup moving from 390 to 50 features is approximately $\times$ 5.5.

Table 2. Table of running times for training various models and explainers in seconds Model training time includes PCA+model training, explainer training time includes LRA calculations. Models trained with AMD R9-3900X with 24 threads.

A	Model Training Time (O)	Model Training Time (LRA)	Explainer Training Time (O)	Explainer Training Time (LRA)	Explainer
Decision Tree	0.147	0.0711	0.997	0.466	TreeExplainer
Random Forest ($n = 1000$)	8.40	3.28	1.40	1.32	TreeExplainer
Random Forest ($n = 5000$)	39.5	15.6	11.9	10.8	TreeExplainer
Linear Regression	0.0230	0.0210	0.00 801	0.0220	LinearExplainer
Ridge Regression	0.0230	0.0220	0.0110	0.0235	LinearExplainer
Kernel Ridge Regression	0.105	0.0941	551.0	32.4	KernelExplainer
Multi-Layer Perceptron*	0.882	1.67	483.2	45.5	Explainer

Consider the permutation explainer provided by the SHAP package, which is model agnostic and works by iteration over complete permutations of the features forward and reversed [5]. The number of permutations that must be considered to produce and accurate explanation for data is significantly reduced if we can lower the number of features to be considered using our LRA approach. This is important because not all models have a concept of 'feature importances' and therefore to derive feature rankings, SHAP explainers must be used. In table 2 we show the computational time required to train and explain a particular model using the original data compared with our LRA approach. We see that our approach yields significantly faster speeds in almost all regards. Further timing curves can be found in supplementary information. In conjunction with our performance results in section 5, our result demonstrates significant advantages to the feature ranking task. However, in the case of simpler models such as Ridge Regression, the sub-par performance results and the lack of significant speed-up demonstrates that our methodology may not apply in all possible machine learning tasks.

7. Conclusion

The task of feature selection is a complex one, yet simply taking the most important features contributing to a model typically yields good initial results. However, the task of training a model, or a feature ranking may be computational intensive. It may also be the case that a strong enough predictor may be infeasible given the dimensionality of the data. Our approach allows this feature selection step to be carried out on such high dimensionality data without significant adverse effects on model performance, providing a type of data triage that speeds up computation. This factor will become more critical as we see the development of real-time automatic and autonomous laboratories and the adoption of machine learning methods in many areas of applied science and technology.

This approach to interpretable dimension reduction and feature selection has great potential in other areas of applied machine learning. As our approach is currently limited to PCA and data which lie on linear manifolds, we plan to extend to other transformations into lower dimensional spaces, in particular non-linear methods such as the interpretable kernel dimensional reduction presented by Wu *et al* [22]. If used consistently this approach is model and data-set agnostic and can aid in model selection and validation.

ORCID iD

Tommy Liu https://orcid.org/0000-0003-4800-8418

References

[1] Barnard A, Motevalli B, Parker A, Fischer J, Feigl C and Opletal G 2019 *Nanoscale* **11** 19190–201
[2] Schleder G R, Padilha A C M, Reily Rocha A, Dalpian G M and Fazzio A 2019 *J. Chem. Inf. Model.* **60** 452–9
[3] Kuhn T S 2012 *The Structure of Scientific Revolutions* (Chicago, IL: University of Chicago press)
[4] Bishop C M 2006 *Pattern Recognition and Machine Learning* (Berlin: Springer)
[5] Lundberg S M and Lee S I 2017 A unified approach to interpreting model predictions *Advances in Neural Information Processing Systems* pp 4765–74
[6] McInnes L, Healy J and Melville J 2018 (arXiv:1802.03426)
[7] Li J, Cheng K, Wang S, Morstatter F, Trevino R P, Tang J and Liu H 2017 Feature selection: a data perspective *ACM Computing Surveys (CSUR)* **50** 1–45

[8] Nguyen T T, Huang J Z and Nguyen T T 2015 *Sci. World J.* **2015** 471371
[9] Li J, Cheng K, Wang S, Morstatter F, Trevino R P, Tang J and Liu H 2017 *ACM Comput. Surv. (CSUR)* **50** 1–45
[10] Cohen S, Ruppin E and Dror G 2005 *Other Words* **1** 98Eqr
[11] Shapley L S 1953 *Contrib. Theory Games* **2** 307–17
[12] Lundberg S M, Erion G G and Lee S I 2018 (arXiv:1802.03888)
[13] Wold S, Esbensen K and Geladi P 1987 *Chemometr. Intell. Lab. Syst.* **2** 37–52
[14] Udell M, Horn C, Zadeh R and Boyd S 2014 (arXiv:1410.0342)
[15] Barnard A, Motevalli Soumehsaraei B and Sun B 2019 Periodic graphene oxide data set. v1. CSIRO Data Collection (available at: https://doi.org/10.25919/5e30b45f9852c)
[16] Barnard A, Motevalli Soumehsaraei B, Sun B and Lai L 2019 Neutral graphene oxide data set. v1. CSIRO Data Collection (available at: https://doi.org/10.25919/5e30b44a7c948)
[17] Pedregosa F *et al* 2011 *J. Mach. Learn. Res.* **12** 2825–30
[18] Kursa M B, Jankowski A and Rudnicki W R 2010 *Fundam. Inform.* **101** 271–85
[19] Breiman L 2001 *Mach. Learn.* **45** 5–32
[20] Covert I, Lundberg S and Lee S I 2020 *Adv. Neural Inf. Process. Syst.* **33** 17212–23
[21] Biau G 2012 *J. Mach. Learn. Res.* **13** 1063–95
[22] Wu C, Miller J, Chang Y, Sznaier M and Dy J 2019 (arXiv:1909.03093)