Supervised Dimensionality Reduction for the Algorithm Selection Problem

Danielle Notice[1], Nicos G. Pavlidis[2], and Ahmed Kheiri[2]

[1] STOR-i Centre for Doctoral Training, Lancaster University
d.a.notice@lancaster.ac.uk
[2] Department of Management Science, Lancaster University
{n.pavlidis,a.kheiri}@lancaster.ac.uk

Abstract. Instance space analysis extends the algorithm selection framework by enabling the visualisation of problem instances via dimensionality reduction (DR). The lower dimensional projection can also be used as input to predict algorithm performance, or to perform algorithm selection. In this paper we consider two supervised DR methods – partial least squares (PLS) and linear discriminant analysis (LDA) – both as visualisation tools and for the purpose of constructing classification models for algorithm selection. Multinomial logistic regression models are used for the classification problem. We compare PLS and LDA to DR methods previously used in this context on three combinatorial optimisation problems, and show that these methods are as competitive.

Keywords: supervised dimensionality reduction, algorithm selection, classification, combinatorial optimisation.

1 Introduction

In the algorithm selection problem (ASP), we consider a problem space, $\mathcal{P}$, that contains a set of instances of a problem in an application domain; and an algorithm space, $\mathcal{A}$, which consists of k algorithms to solve the instances in $\mathcal{P}$. The aim is to predict which algorithm in $\mathcal{A}$ will achieve the best performance y for a given problem instance $x \in \mathcal{P}$. Exhaustive testing may be computationally infeasible to identify the most suitable algorithm for each instance $x \in \mathcal{P}$. So instead we try to understand the relationship between a *description* of problem instances and algorithm performance. Under the ASP framework proposed in [1] we use the following components to formulate a supervised learning problem:

- the feature matrix $\mathbf{F} \in \mathbb{R}^{m \times n}$ contains the values of m features that describe each of the $n = |\mathcal{P}|$ problem instances;
- the performance matrix $\mathbf{Y} \in \mathbb{R}^{k \times n}$ contains the performance metric value for each of the k algorithms in solving each of the n problem instances;
- the best algorithm label $\mathbf{Y}_{\text{best}} \in \{1, 2, \ldots, K\}^n$ is a vector which reports the index of the best performing algorithm for each $x \in \mathcal{P}$. Note that, as we will explain in Section 3, K need not be equal to $|\mathcal{A}|$.

Instance space analysis (ISA) [2] is a methodology that extends the ASP framework by (among other things) enabling the visualisation of problem instances. Elements of the ISA methodology have been applied to several combinatorial optimisation problems [3,4,5]. In ISA, the problem space is visualised via dimensionality reduction (DR) of the feature space. Principal component analysis (PCA) is a common DR technique which aims to find a projection of $\mathbf{F}$ which captures the maximum data variability, or equivalently minimise the reconstruction error. While PCA is widely used [4], the directions of maximum variance in $\mathbf{F}$ may not necessarily be useful to explain the variability in $\mathbf{Y}$, which is ultimately what we are most interested in. **Supervised dimensionality reduction** methods take into account the performance space when seeking to project the feature space to a lower dimension subspace.

The authors of [6] proposed PILOT, a method for supervised DR for ISA. PILOT is primarily used for visualisation. However, the lower dimensional projection of $\mathbf{F}$ can also be used as input to predict algorithm performance, or to perform algorithm selection. DR is particularly useful for training prediction models when (i) the number of problem instances is less than number of features, (ii) there exist irrelevant or strongly correlated features (columns of $\mathbf{F}$). DR also allows for reduced model complexity with less loss of information compared to feature selection (although this comes at a cost in terms of interpretability).

In this paper we consider two other supervised DR methods and compare them to PILOT and PCA on three combinatorial optimisation problems. To the best of our knowledge these methods have not been previously considered in the DR for ISA literature, although they are very well known in statistics and machine learning. Our aim is to evaluate these methods both as visualisation tools and for the purpose of constructing classification models.

2 Dimensionality reduction

DR methods aim to represent a dataset of n observations, $\mathbf{F} \in \mathbb{R}^{m\times n}$, in a lower dimensional space $\mathbf{Z} \in \mathbb{R}^{d\times n}$ where $d < m$, such that $\mathbf{Z}$ retains the most *important properties* of the original data matrix $\mathbf{F}$. In linear DR methods, the matrix $\mathbf{Z}$ is given by,

$$\mathbf{Z} = \mathbf{A}\mathbf{F},$$

where $\mathbf{A} \in \mathbb{R}^{d\times m}$ is called the *transformation matrix*. In words, each new component (row of $\mathbf{Z}$) is a linear combination of the original features in $\mathbf{F}$. When $d \leqslant 3$, $\mathbf{Z}$ is useful for visualising the problem space.

PCA is the most commonly used DR method. PCA finds a projection of $\mathbf{F}$ that maximises the variance of the projected data (or equivalently minimises the reconstruction error) subject to the constraint that projection vectors (rows of $\mathbf{A}$) are mutually orthogonal. The solution to this problem is given by setting the d rows of $\mathbf{A}$ equal to the d eigenvectors of the covariance matrix of $\mathbf{F}$ that correspond to the largest eigenvalues. As an unsupervised DR method, PCA does not take into account information from the response, $\mathbf{Y}$. Consequently, there is no guarantee that the directions of greatest variation in input data will be useful to explain variability in $\mathbf{Y}$.

We next describe three supervised DR methods that take into account the performance space when seeking lower dimensional projections.

2.1 PILOT

The algorithm for projecting instances with linearly observable trends (PILOT) aims to find a projection space that simultaneously captures the linear relationships between each of the features and the performance of each algorithm across the instance space [2]. This is a low dimensional subspace of the input space that is useful to minimise the reconstruction error for both the feature vectors and the response.

Specifically, PILOT seeks a transformation matrix $\mathbf{A}_r \in \mathbb{R}^{d \times m}$, and two linear functions represented as $\mathbf{B}_r \in \mathbb{R}^{m \times d}$ and $\mathbf{C}_r \in \mathbb{R}^{k \times d}$ to solve the following problem,

$$\min_{\mathbf{A}_r, \mathbf{B}_r, \mathbf{C}_r} \|\mathbf{F} - \mathbf{B}_r \mathbf{Z}\|_F^2 + \|\mathbf{Y} - \mathbf{C}_r \mathbf{Z}\|_F^2, \tag{1}$$

$$\text{where } \mathbf{Z} = \mathbf{A}_r \mathbf{F}. \tag{2}$$

The matrix $\mathbf{A}_r$ is given by [6],

$$\mathbf{A}_r = \mathbf{V}^\top \bar{\mathbf{X}} \mathbf{F}^\top \left(\mathbf{F}\mathbf{F}^\top\right)^{-1},$$

where $\bar{\mathbf{X}} = [\mathbf{F}; \mathbf{Y}] \in \mathbb{R}^{(m+k) \times n}$, and $\mathbf{V} \in \mathbb{R}^{(m+k) \times d}$ contains the first d eigenvectors of $\bar{\mathbf{X}}\bar{\mathbf{X}}^\top$. PILOT is primarily used for visualisation, so by default $d = 2$.

2.2 Partial least squares

The objective of partial least squares (PLS) regression is to recursively find a pair of directions $\mathbf{w}$ and $\mathbf{t}$ to project $\mathbf{F}$ and $\mathbf{Y}$, respectively, such that the covariance between these one-dimensional vectors, $\mathbf{w}^\top \mathbf{F}$ and $\mathbf{t}^\top \mathbf{Y}$ is maximised [7]. For PLS the rows of $\mathbf{F}$ and $\mathbf{Y}$ are first standardised.

$$\max_{\mathbf{w_1}, \mathbf{c_1}} \mathbf{w}_1^\top \mathbf{F} (\mathbf{c}_1^\top \mathbf{Y})^\top, \text{ s.t. } \|\mathbf{w}_1\|_2 = 1, \ \|\mathbf{t}_1\|_2 = 1.$$

Subsequent pairs of projection vectors $(\mathbf{t}_i, \mathbf{w}_i)$, $i > 1$ are found sequentially by solving the same optimisation problem after $\mathbf{F}$ and $\mathbf{Y}$ are deflated. The termination criterion is either that the required number of dimensions is reached, or $\mathbf{F}^\top \mathbf{Y} = \mathbf{0}$. PLS can also be used for classification in mind by transforming $\mathbf{Y}_{\text{best}}$ to dummy variables and then fitting the projection. This is sometimes referred to as PLS discriminant analysis [8].

2.3 Linear discriminant analysis

Linear discriminant analysis (LDA) is both a method for classification and supervised DR. Its objective is to maximise the separability among known classes in the data. Let N_k and $\bar{x}_k$ be the number of instances and class means for class k. Fisher's criterion is to identify the projection matrix that solves the following problem:

$$\mathbf{A}_{\texttt{LDA}} = \arg\max_{\mathbf{A}} \frac{\det(\mathbf{A}^\top S_b \mathbf{A})}{\det(\mathbf{A}^\top S_w \mathbf{A})},$$

where

$$S_b = \sum_{k=1}^{K} N_k (\bar{x}_k - \bar{x})(\bar{x}_k - \bar{x})^\top, \qquad S_w = \sum_{k=1}^{K} \sum_{x \in K_i} (x - \bar{x}_k)(x - \bar{x}_k)^\top.$$

S_b is the between-class scatter matrix and S_w is the common, within-class, covariance matrix. The optimal projection maximises the variance of the across-class means and minimises the within-class variance. $\mathbf{A}_{\texttt{LDA}}$ is composed of the eigenvectors of $S_w^{-1} S_b$, of which there are at most $K-1$. Therefore when using LDA, $\mathbf{F}$ can be reduced to at most $K-1$ dimensions.

3 Comparison of methods

We consider three combinatorial optimisation problems: the travelling salesman problem (TSP); the vehicle routing problem (VRP) and the knapsack problem.

The **travelling salesman problem (TSP)** aims to find the cyclic tour with minimum cost which visits each of N nodes in the problem instance exactly once. We use 2143 problem instances and 87 features from the datasets generated by [9]. The performance matrix $\mathbf{Y}$ is the median cost of the tour found after a fixed time for two metaheuristic algorithms. $\mathbf{Y}_{\text{best}}$ has three classes – one for when either algorithm is the best, and a third when they perform nearly the same. There are three datasets based on different termination criteria. In `tsp60` and `tsp1800` the termination criterion is a time budget of 60 seconds and 1800 seconds, respectively. In `tspWin` performance is measured at the point at which the first algorithm terminates. We consider all three datasets as relative algorithm performance greatly varies depending on the choice of termination criterion.

The **capacitated vehicle routing problem (CVRP)** aims to find the optimal routes for a fleet of vehicles with fixed maximum capacity to visit a depot and a set of nodes each with a fixed demand. We use the performance data for eight algorithms published in [10] for 100 problem instances. The performance of each algorithm is measured by the mean total distance of the solution found after a fixed time. For some problem instances, there are ties for best performance; in those cases, we set the best class label as the winning algorithm which was first developed. This leads to there being five unique labels in $\mathbf{Y}_{\text{best}}$. To characterise these problem instances, we use 119 features, some of which we constructed and most of which are found in [11].

The **0-1 knapsack problem** involves deciding which items of known size and profit to select from a finite set of items to fill a knapsack with a fixed capacity constraint, so as to maximise profit. We use the dataset from the Melbourne algorithm test instance library with data analytics (MATILDA) [12] generated in [5]. The dataset contains 4000 problem instances, characterised by 44 features, 21 of which are transformations of the original features. The performance matrix $\mathbf{Y}$ contains the time to obtain and verify the optimal solution for each problem instance for three algorithms.

Table 1 summarises the information about the feature and performance spaces for each problem. Note that the number of unique classes K in $\mathbf{Y}_{\text{best}}$, need not equal $|\mathcal{A}|$ if there exist ties in the performance metrics. For each dataset, the training and test data are stratified using these class labels $\mathbf{Y}_{\text{best}}$. The proportion of instances in each class are also included in Table 1.

Table 1. Summary of datasets

	n	m	$\|\mathcal{A}\|$	K	Class Proportions
`tsp60`	2143	87	2	3	(0.45, 0.35, 0.20)
`tsp1800`	2143	87	2	3	(0.43, 0.48, 0.08)
`tspWin`	2143	87	2	3	(0.27, 0.36, 0.37)
`vrp`	100	119	5	5	(0.52, 0.20, 0.13, 0.10, 0.05)
`knapsack`	4000	44	3	3	(0.64, 0.23, 0.13)

For each of the DR methods described in Section 2, the transformation matrix is fitted on the training data. Then the full dataset is projected to the lower dimensional subspace. Three variants of PLS are fitted: PLS using $\mathbf{Y}$, relative PLS using the performance of each algorithm relative to the best $\mathbf{Y}_{\text{rel}}$, and PLS-DA using the dummy encoding of $\mathbf{Y}_{\text{best}}$. Each of these variants use representations of the performance matrix which are more directly suited to different tasks – $\mathbf{Y}$ for performance prediction (regression), $\mathbf{Y}_{\text{rel}}$ for performance comparison (regression) and $\mathbf{Y}_{\text{best}}$ for algorithm selection (classification).

The analytic solution to PILOT is used as it is time-consuming to solve the problem numerically for greater than two dimensions (as it was designed to be used). A step was added to the algorithm which removes features that are linearly dependent in order for the problem to be solvable analytically. When it is not possible to do so PILOT is not included in the analysis.

To perform algorithm selection, the following models are trained to predict $\mathbf{Y}_{\text{best}}$. The baseline models are a naïve classifier which always predicts the most frequent class for all problem instances, and a multinomial logistic regression model using the constructed features. For each DR method, a multinomial logistic regression model is fit given $\mathbf{Z}$. The number of components used to fit each of these models was determined by cross-validation. For PCA, PILOT and LDA, the number of components which led to the best logistic regression model in terms of accuracy, precision and recall was selected. For the PLS variants, the number of components which minimised the root mean squared error of the predictions from the projection model was selected.

To evaluate each of the classification models, the following metrics are used. Accuracy is the proportion of instances which were correctly classified across all classes. Weighted precision is average of the per-class ratio of correctly classified instances to all instances predicted to be in that class, weighted by the proportion of instances actually in the class. Recall is the average of the fraction of correctly classified instances in each class. We also considered the ***relative regret*** of the misclassified instances. This is the percentage difference between the performance for the predicted algorithm and the actual best algorithm for a given instance.

To evaluate the visualisations, we use the ***neighbourhood hit*** [13]. This is the average proportion of the k neighbours of each point with the same class label as that point. Neighbourhood hit in effect measures how separable the data is in the projected subspace. We calculated this on first two dimensions of the projected coordinates for each DR method.

4 Results

4.1 TSP

We first evaluate the projections' usefulness for visualisations. The 2D visualisations of all the problem instances in the different projection spaces for each dataset are included in the supplementary materials and code for the analysis is available[3]. Fig. 1 shows that for all of the projections, the more neighbours considered, the lower the neighbourhood hit rate. For these datasets, all the projections except LDA have similar trends how the neighbourhood hit rate decreases. For LDA, the rate of decrease is much slower, to the extent that there are more common neighbours when $k = 25$ (which is approximately 5% of the dataset) than with the other projections which have a higher hit rate for fewer neighbours. For `tsp60`, all PLS projections perform the best, with a high 57.9% hit rate for the nearest neighbour, and a low of 52.4% for the nearest 25 neighbours. Similarly for `tsp1800`, PLS and PLS-DA performed better than the other projections.

Next we evaluate all DR methods for classification. These performance metrics are reported in Table 2. Based on this, the set of features is better suited for predictions for `tspWin`. All three TSP datasets have the same $\mathbf{F}$, but different $\mathbf{Y}$ and $\mathbf{Y}_{\text{best}}$. Because PCA is unsupervised, the projections are the same for all datasets, however the predictions on `tspWin` require more than twice as many components than are needed for `tsp60`, and this number increases further for `tsp1800`. While there is no single standout across the metrics, LDA can be considered the best DR method for `tsp60` and `tspWin` when considering the number of components needed. The relative PLS models have inferior performance for all three of these datasets. For both algorithms in all of the datasets, the values of $\mathbf{Y}_{\text{rel}}$ are predominantly very close to zero, which resulted in the relative PLS projection being uninformative. PILOT requires more components than the

[3] https://github.com/danotice/Sup-DR-for-ASP

other supervised DR techniques, but it does achieve better performance than the model with all features in two out of the three datasets.

There is no objective way to define "good" performance in absolute terms for the algorithms in this data because of the nature of the performance metric, so the relative performance is considered. Across all three datasets, both algorithms almost always find solutions within one percent of each other. Considering the relative regret of the misclassified instances, all the models (excluding the naïve classifier) have a similar distribution. The relative regret shows that when given enough time to run, both algorithms will achieve the same performance. The median relative regret for `tsp1800` – which had a maximum time budget of 1800 seconds – was on average 0.007%, while the median for `tsp60` was on average 0.012%. All of the models misclassify 1 or 2 instances with abnormally large relative regret, which skew the means. The relative regret being so small for most instances gives an indication that without an absolute threshold (such as difference from the known optimal solution), we cannot reasonably try to predict "good" performance for the algorithms.

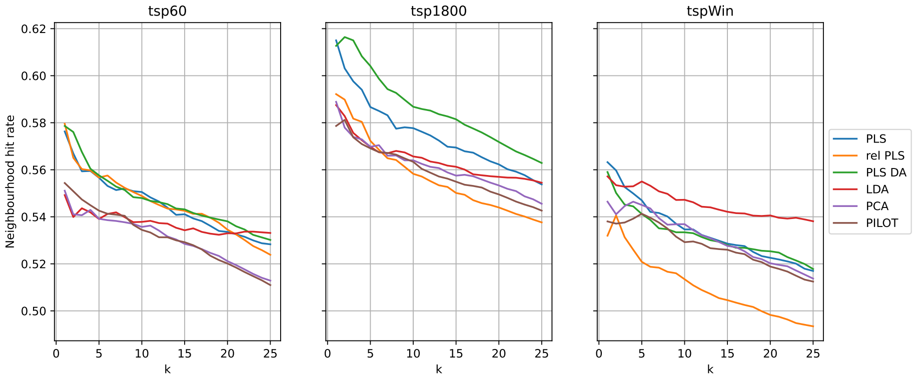

Fig. 1. Neighbourhood hit rate of actual class labels for TSP datasets

4.2 VRP

For the `vrp` dataset, LDA appears to separate the classes well as seen in the visualisation of the problem space in Fig. 2. The average neighbourhood hit rate for up to 8 neighbours for LDA is above 80%, and less than 60% for all the other methods. However, looking at Fig. 2 the training data are projected to very separate clusters, but the test data are not separated as well.

Considering the classification models, PCA has similar performance to the model with all the features. The PLS variants however both achieve better performance, using far fewer components. Notably, the PLS projection using the relative performance improves the predictions made by PLS, and has the best evaluation metrics overall. PLS-DA, while notably worse in terms of recall, is

Table 2. Prediction model evaluation metrics for test data

						rel	PLS	
	Naïve	**All**	**PCA**	**PILOT**	**PLS**	**PLS**	**DA**	**LDA**
`tsp60`								
Components	-	86	22	20	15	2	7	**2**
Accuracy	44.99	56.18	56.41	55.01	52.68	46.15	53.85	55.24
Precision	75.25	56.75	56.64	55.17	52.70	46.98	53.76	56.09
Recall	33.33	55.43	56.73	54.72	52.63	38.92	53.84	54.26
`tsp1800`								
Components	-	86	63	74	22	1	7	**1**
Accuracy	48.25	59.91	59.91	**61.77**	59.21	57.11	59.21	56.64
Precision	75.03	59.26	59.50	**61.25**	56.53	60.90	58.32	51.89
Recall	33.33	46.94	47.70	**48.24**	43.36	40.78	43.50	40.77
`tspWin`								
Components	-	86	42	50	9	8	20	**2**
Accuracy	36.83	62.47	63.40	**63.64**	62.24	60.84	62.94	62.47
Precision	76.73	58.78	59.80	**60.27**	56.56	54.01	58.50	59.51
Recall	33.33	58.88	59.72	**60.07**	57.73	56.30	58.94	59.09
`vrp`								
Components	-	117	50	-	9	7	**1**	4
Accuracy	50.00	65.00	65.00	-	65.00	**80.00**	70.00	60.00
Precision	75.00	79.00	71.86	-	66.39	**86.00**	78.33	70.15
Recall	20.00	69.00	72.00	-	67.33	**76.33**	43.33	67.00
`knapsack`								
Components	-	44	39	31	41	**22**	24	**2**
Accuracy	63.62	73.88	73.88	73.00	73.62	72.25	72.75	72.62
Precision	76.86	72.41	72.41	71.55	72.12	70.10	71.26	71.26
Recall	33.33	62.14	62.14	61.79	61.64	58.65	61.44	56.98

comparable to the relative PLS projection in terms of the other performance metrics, achieving 70% accuracy and 78% weighted precision, while using a single component. Overall, this dataset benefits the most from the use of the supervised DR methods, most likely because of its small size.

Also, recall that performance data is available for eight algorithms, although only five of them are contenders for the best in this dataset. Including the performance of the other three algorithms in $\mathbf{Y}$ worsened the performance of PLS and relative PLS, as they were accounting for variance that was irrelevant for the classification problem. The other methods were unaffected by this as PCA is unsupervised, and LDA and PLS-DA use $\mathbf{Y}_{\text{best}}$.

There are many ties in algorithm performance in `vrp`, which makes it reasonable to set a small relative threshold for "good" performance of the algorithms. Based on the distribution of the relative regret shown in Fig. 3, setting this threshold to greater than 0.3% would be inappropriate as all of the classification models are able to make predictions with a lower margin of error.

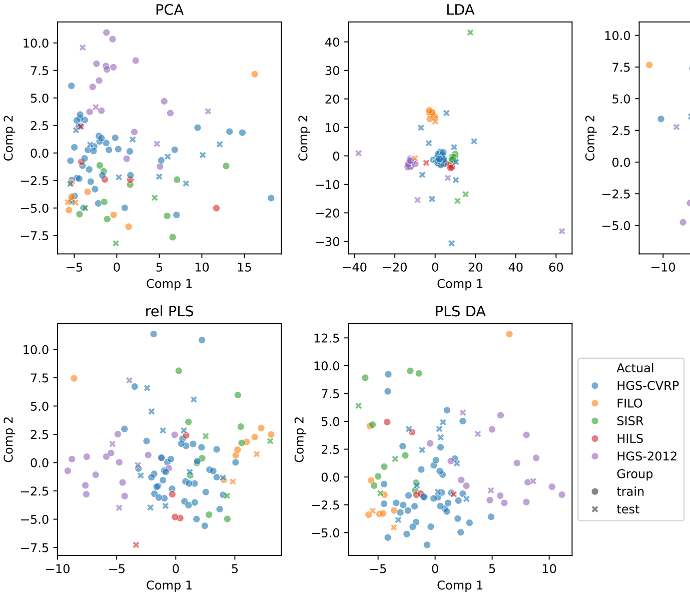

Fig. 2. 2D projections of `vrp` instance space showing actual best algorithm

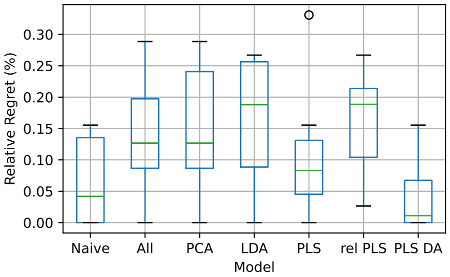

Fig. 3. Distribution of relative regret for misclassified instances from `vrp` dataset

4.3 Knapsack

The dense problem space for the `knapsack` dataset makes seeing any class separability difficult in Fig. 4. Based on the neighbourhood hit rates in Fig. 5 however, LDA is the best, having the largest clusters of instances with the same class. While PCA and PILOT projections are quite similar (but mirrored), there is greater separation of some of the clusters along the axis of the first component, which is reflected in the neighbourhood hit rate.

Considering the classification models, we see in Table 2 that both PCA and PLS require almost as many components as there are features for the best predictions. With PCA in particular, using this many components leads to predictions nearly identical to those made by the model with all the features - resulting in

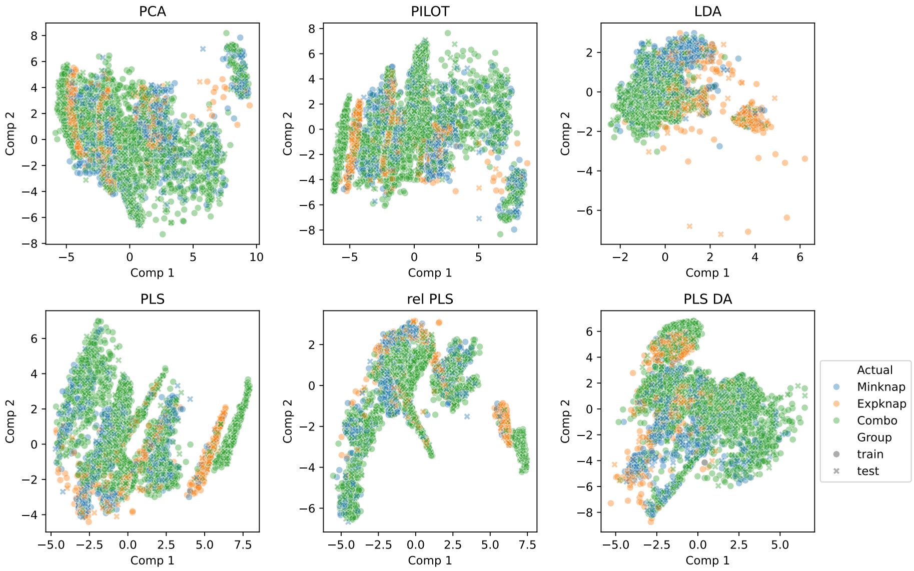

Fig. 4. 2D projections of `knapsack` instance space showing actual best algorithm

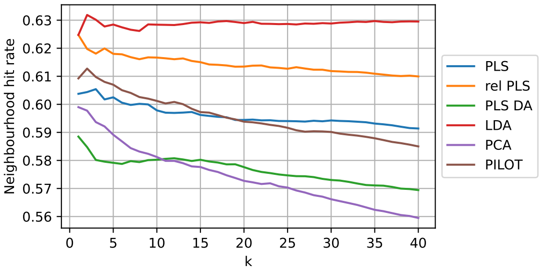

Fig. 5. Neighbourhood hit rate of actual class labels for `knapsack` dataset

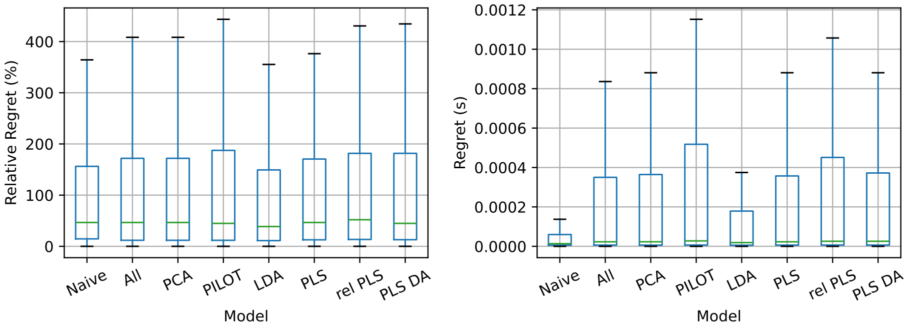

Fig. 6. Distribution of relative and absolute regret (excluding outliers) for misclassified instances from `knapsack` dataset

the evaluation metrics we considered to all be equal. PILOT and the PLS variants produce predictions of similar quality using fewer components. Once again LDA, despite the strict limit on the maximum number of components remains competitive.

The performance metric for this dataset is a measure of time, and it varies greatly between algorithms for each problem instance. As this is a measure that is not instance specific, we consider both the relative regret and the absolute regret in seconds. Table 3 shows the number of misclassified instances from the test data and the total absolute regret for each model. While all have multiple outliers, we see that the other models have a total absolute regret which is orders of magnitude greater than that of LDA, despite having fewer misclassified instances.

Table 3. Total absolute regret for each classification model for `knapsack`

						rel	**PLS**	
	Naïve	**All**	**PCA**	**PILOT**	**PLS**	**PLS**	**DA**	**LDA**
No. misclassified	291	209	209	216	211	222	218	219
No. outliers	46	29	29	35	29	33	34	**23**
Total regret	0.228	94.868	94.870	139.879	94.870	90.239	139.865	**0.233**

5 Conclusion

In this paper we considered PCA and a number of supervised dimensionality reduction techniques for visualisation and algorithm selection. While performing DR on the feature space reduces the interpretability of the classification model used for algorithm selection, it does alleviate the need for elegant feature construction and selection.

In the ISA literature PCA and PILOT are used for visualisation, but not classification. In this paper we found that PCA generally required considerably more components than the supervised DR methods to achieve comparable classification performance. Moreover, the two-dimensional visualisations do not capture the class structure as well as supervised DR methods can. PILOT required more components than other supervised DR methods generally, while having performance similar to PCA.

LDA and PLS are not new techniques, but to the best of our knowledge they have not been previously applied in this context. LDA was notable as it was just as competitive as the other projections but used far fewer components. In terms of the visualisation, it best showed the separability of the classes for all the datasets measured in terms of neighbourhood hit.

Using fewer components, PLS and PLS-DA were always able to achieve prediction performance comparable to the logistic regression model with all features.

The PLS-DA projections were better for the predictions than the PLS regression input. This is not an unexpected result as PLS-DA is directly connected to the classification (rather than the regression) problem. A more notable result was that the relative PLS projections were better than the usual PLS projections for the VRP and knapsack data. Using the relative performance led to a projection better suited comparison of the algorithm performance as opposed to prediction, which is a more difficult task.

Acknowledgements

This paper is based on work completed while Danielle Notice was part of the EPSRC funded STOR-i Centre for Doctoral Training (EP/S022252/1). This work was also funded in part by Tesco Stores Limited.

References

1. Rice, J.R.: The algorithm selection problem. In: Rubinoff, M., Yovits, M.C. (eds.) Advances in Computers, vol. 15, pp. 65–118. Elsevier (1976)
2. Smith-Miles, K., Muñoz, M.A.: Instance space analysis for algorithm testing: Methodology and software tools. ACM Computing Surveys (2022)
3. Alipour, H., Muñoz, M.A., Smith-Miles, K.: Enhanced instance space analysis for the maximum flow problem. European Journal of Operational Research 304(2), 411–428 (2023)
4. Smith-Miles, K., Baatar, D., Wreford, B., Lewis, R.: Towards objective measures of algorithm performance across instance space. Computers & Operations Research 45, 12–24 (2014)
5. Smith-Miles, K., Christiansen, J., Muñoz, M.A.: Revisiting where are the hard knapsack problems? via instance space analysis. Computers & Operations Research 128 (2021)
6. Muñoz, M.A., Villanova, L., Baatar, D., Smith-Miles, K.: Instance spaces for machine learning classification. Machine Learning 107(1), 109–147 (2017)
7. Abdi, H.: Partial least squares regression and projection on latent structure regression (pls regression). WIREs Computational Statistics 2(1), 97–106 (2010)
8. Barker, M., Rayens, W.: Partial least squares for discrimination. Journal of Chemometrics 17(3), 166–173 (2003)
9. Pihera, J., Musliu, N.: Application of machine learning to algorithm selection for TSP. In: 2014 IEEE 26th International Conference on Tools with Artificial Intelligence. pp. 47–54 (2014)
10. Vidal, T.: Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. Computers & Operations Research 140 (2022)
11. Rasku, J., Kärkkäinen, T., Musliu, N.: Feature extractors for describing vehicle routing problem instances. In: OpenAccess Series in Informatics. vol. 50, pp. 7.1–7.13 (2016)
12. Smith-Miles, K., Muñoz, M., Neelofar: Melbourne algorithm test instance library with data analytics (MATILDA) (2020), https://matilda.unimelb.edu.au/
13. Espadoto, M., Martins, R.M., Kerren, A., Hirata, N.S.T., Telea, A.C.: Toward a quantitative survey of dimension reduction techniques. IEEE Trans Vis Comput Graph 27(3), 2153–2173 (2021)