SLISEMAP: Combining supervised dimensionality reduction with local explanations *

Anton Björklund[0000−0002−7749−2918], Jarmo Mäkelä[0000−0002−8788−3939], and Kai Puolamäki[0000−0003−1819−1047]

University of Helsinki, Helsinki, Finland

Abstract. We introduce a Python library, called SLISEMAP, that contains a supervised dimensionality reduction method that can be used for global explanation of *black box* regression or classification models. SLISEMAP takes a data matrix and predictions from a *black box* model as input, and outputs a (typically) two-dimensional embedding, such that the black box model can be approximated, to a good fidelity, by the same interpretable *white box* model for points with similar embeddings. The library includes basic visualisation tools and extensive documentation, making it easy to get started and obtain useful insights. The SLISEMAP library is published on GitHub and PyPI under an open source license.

Keywords: Manifold visualisation · Explainable AI

1 Introduction

In our recent manuscript [3] we introduce an algorithm, SLISEMAP, that extends [1,2] and combines manifold visualization (e.g., [8,6,7]) with local, model-agnostic explanations of regression or classification models (see [5] for a review). The idea of the latter is to find an interpretable *white box* surrogate model that locally approximates a complex *black box* model for a given data point.

SLISEMAP produces a non-linear embedding of the data into d dimensions (typically $d = 2$), such that data points projected nearby can, with good fidelity, be explained by the same white box model. Each data point have an embedding and an associated white box model. Together the white box models and the visual embedding provide a *global* explanation of the black box model.

In this paper we describe a Python library, called SLISEMAP, that implements the algorithm by the same name.

The SLISEMAP library can be used by all who want to explore datasets or are interested in global explanations for complex black box models.

While there are plethora of software for manifold embeddings or local explanations, none exist that combine these two.

* Support by Academy of Finland (grants 320182, 346376) & Future Makers Program.

2 Problem definition

Formally, input to SLISEMAP is given as a dataset of n points $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$, where the covariates are given by real vectors $\mathbf{x}_i \in \mathbb{R}^m$ and the responses $\mathbf{y}_i = f(\mathbf{x}_i) \in \mathbb{R}^p$, where $f : \mathbb{R}^m \to \mathbb{R}^p$ is a pre-trained black box regression or classification model that we wish to explain. For regression problems $p = 1$ and for classification problems p is the number of classes, where $\mathbf{y}_i$ represents the predicted class probabilities.

We also need a type of easy-to-understand, *white box*, surrogate model, $g_i : \mathbb{R}^m \to \mathbb{R}^p$, that we use to approximate the black box model f in the neighbourhood (as defined by the embedding) of the data point $i \in \{1, \ldots, n\}$. We collect the parameters of the white box models into a matrix $\mathbf{B} \in \mathbb{R}^{n \times q}$ such that the ith row $\mathbf{B}_{i\cdot}$ contains the parameters of the white box model g_i. As g_i for regression problems we use a simple linear model and for classification problems a multinomial logistic regression. Additionally, the loss function $l : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}_{\geq 0}$ quantifies the mismatch between the black box and white box models. We use quadratic loss for regression problems and Hellinger loss (which is related to log-loss) for classification problems. Formally, the SLISEMAP algorithm finds an embedding of a given radius by solving the following computational problem.

Problem 1. [3] Given the definitions above, regularization parameters $\lambda_{lasso} \geq 0$ and $\lambda_{ridge} \geq 0$, and the radius of the embedding $z_{radius} > 0$, find the parameters $\mathbf{B} \in \mathbb{R}^{n \times q}$ and embedding of data points $\mathbf{Z} \in \mathbb{R}^{n \times d}$ that minimise the loss given by $\mathcal{L} = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{W}_{ij} \mathbf{L}_{ij} + \sum_{i=1}^{n} \sum_{j=1}^{q} \left(\lambda_{lasso} |\mathbf{B}_{ij}| + \lambda_{ridge} \mathbf{B}_{ij}^2 \right)$, where $\mathbf{L}_{ij} = l(g_i(\mathbf{x}_j), \mathbf{y}_j)$, $\mathbf{W}_{ij} = e^{-\mathbf{D}_{ij}} / \sum_{k=1}^{n} e^{-\mathbf{D}_{ik}}$, and $\mathbf{D}_{ij} = (\sum_{k=1}^{d} (\mathbf{Z}_{ik} - \mathbf{Z}_{jk})^2)^{1/2}$, with the constraint that $(\sum_{i=1}^{n} \sum_{k=1}^{d} \mathbf{Z}_{ik}^2 / n)^{1/2} = z_{radius}$.

This means that the local models are optimised using weights. The weights are based on distances between the data points in the embedding. Incompatible local models are, thus, pushed away from each other. Conversely, the constraint on the embedding size leads to interchangable local models forming clusters.

We refer to [3] for a detailed summary of related work, description and analysis of the algorithm, as well as experimental validation.

3 The SLISEMAP library

SLISEMAP is implemented in *Python* using *PyTorch* for the optimisation, enabling automatic differentiation and optional GPU-acceleration. However, the library also interfaces with standard *Numpy*. For the built-in visualisation, exploration, and diagnostics tools we use *Seaborn*.

The design goals of the library are flexibility, performance, and ease of use. This is accomplished through optional parameters, closures, and just-in-time compilation, while providing extensive documentation, sane defaults, and helpful warning messages.

The SLISEMAP library is open source and available under an MIT license at https://github.com/edahelsinki/slisemap. The repository also includes a demonstration video and an extended version of the example discussed below in the

Table 1. Descriptions and default values for the most important parameters.

Parameter	Default Value	Description
`X`		Data matrix, in $\mathbb{R}^{n \times m}$
`y`		Response vector / matrix, in $\mathbb{R}^{n \times p}$
`d`	2	Number of embedding dimensions
`radius`	3.5	Spread of the embedding, (z_{radius})
`lasso`	0.0	L1 regularisation coefficient (λ_{lasso})
`ridge`	0.0	L2 regularisation coefficient (λ_{ridge})
`local_model`	Linear regression	Prediction function for the white box model (g_i)
`local_loss`	Least squares	Loss function for the white box model (l)

form of a *Jupyter notebook*. The package can also be installed using `pip install slisemap`.

4 Usage example

The AUTOMPG dataset [4] is a multivariate real-valued dataset with eight attributes describing the properties of 398 distinct cars (6 rows with missing values removed). The covariates are in a (normalised) Numpy array `X`, that consists of seven ordinal attributes for each car. The response vector `y` contains the fuel consumption (miles per gallon), as estimated by a random forest regressor. Code 1 shows how we apply SLISEMAP on this dataset.

```
sm = Slisemap(X, y, lasso=0.01) # Slisemap object
sm.optimise() # Optimise the solution
sm.plot(title="Slisemap with local model clusters",
        clusters=4, bars=6, jitter=0.1, variables=names)
```

Code 1. Basic SLISEMAP usage.

We make the interpretation of the local models easier by clustering (using k-means) the local model coefficients (rows of matrix $\mathbf{B}$) and colour-code the embedding based on the cluster indices. Furthermore, we add some jitter (since some points are on top of each other), and show only the five most meaningful attributes.

The result is shown in Figure 1.

We can now identify which attributes in a given cluster are the most important in getting the predictions correct. For example, model year is an important indicator of fuel economy for cluster 0, but it is less important in cluster 3. Further analysis of the clusters reveals that cluster 3 consists of mostly heavy, U.S.-made cars with poor fuel economy, where the weight is the primary determinant for fuel consumption. On the other hand, cluster 0 has primarily non-U.S. cars, which are, on average, newer and lighter. Here horsepower is also an important attribute in predicting fuel consumption.

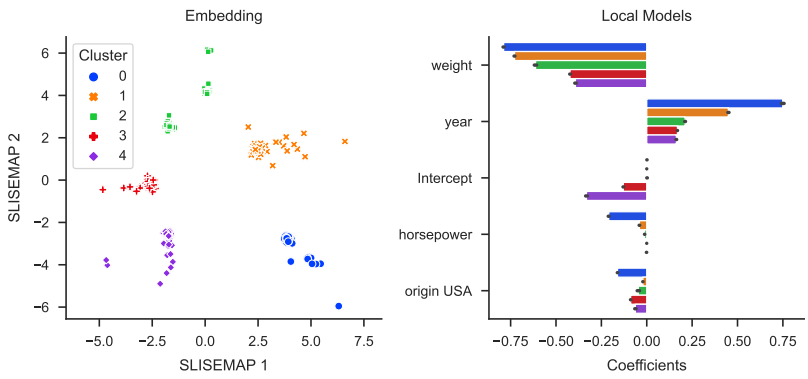

Fig. 1. Two-dimensional Slisemap embedding (left) with clusters based on *white box* surrogate models. The bar plot (right) shows the cluster centroids for the coefficients (rows of matrix $\mathbf{B}$) of the *white box* models.

After optimising an embedding and finding local models with SLISEMAP, it is possible to investigate, with a built-in command, how new data items would be projected onto the same embedding and what their local white box models would be. This is useful for faster embedding of large datasets (using subsampling) or to detect concept drift. Also, the same command can highlight alternative explanations (locations in the embedding) for existing data points.

Classification To use SLISEMAP for classification tasks we only have to replace the white box model (`local_model` in Table 1) with a classifier, such as logistic regression (included in the library). Alternatively we can transform the predictions of a black box model from $[0, 1]$ to $[-\infty, \infty]$ with a logit transformation, $y' = \log(y/(1-y))$, and use linear regression for the approximation. A classification example on a larger dataset is also included in the GitHub repository (https://github.com/edahelsinki/slisemap).

References

1. Björklund, A., Henelius, A., Oikarinen, E., Kallonen, K., Puolamäki, K.: Sparse Robust Regression for Explaining Classifiers. In: Discovery Science, vol. 11828, pp. 351–366 (2019)
2. Björklund, A., Henelius, A., Oikarinen, E., Kallonen, K., Puolamäki, K.: Robust regression via error tolerance. Data Min Knowl Discov (2022)
3. Björklund, A., Mäkelä, J., Puolamäki, K.: SLISEMAP: Supervised dimensionality reduction through local explanations. arXiv:2201.04455 [cs] (2022)
4. Dua, D., Graff, C.: UCI machine learning repository (2017)
5. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A Survey of Methods for Explaining Black Box Models. ACM Comput Surv **51**(5), 1–42 (2019)

6. van der Maaten, L., Hinton, G.: Visualizing Data using t-SNE. J Mach Learn Res **9**(86), 2579–2605 (2008)
7. McInnes, L., Healy, J., Melville, J.: UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv:1802.03426 [cs, stat] (2020)
8. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science **290**(5500), 2319–2323 (2000)