SLISEMAP

SUPERVISED DIMENSIONALITY REDUCTION THROUGH LOCAL EXPLANATIONS

Anton Björklund, Jarmo Mäkelä, Kai Puolamäki.
SLISEMAP: Supervised dimensionality reduction through local explanations.
arXiv: 2201.04455 [cs] (2022). DOI: 10.48550/arXiv.2201.04455.

`https://github.com/edahelsinki/slisemap`

These notes (on the side) contain the spoken parts

SLISEMAP is a new supervised dimensionality reduction method, that can be used to explain black box classification or regression models. SLISEMAP is open source and available as a Python library via GitHub.

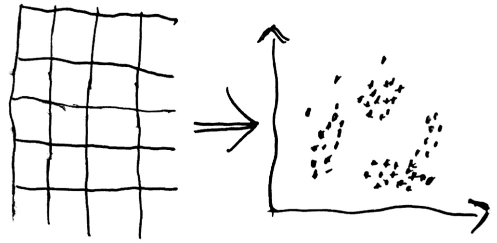
BACKGROUND

SLISEMAP = Supervised dimensionality reduction + Local explanations

To understand SLISEMAP better, lets start by briefly looking at two techniques that inspired SLISEMAP.

BACKGROUND

Dimensionality reduction



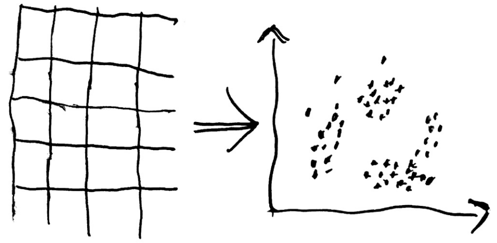
Dimensionality reduction can be used for manifold visualisation.

SLISEMAP = Supervised dimensionality reduction + Local explanations

The goal of dimensionality reduction is to turn a high-dimensional dataset into a lower-dimensional embedding that preserves some of the patterns in the data. This is used for, for example, visualisations.

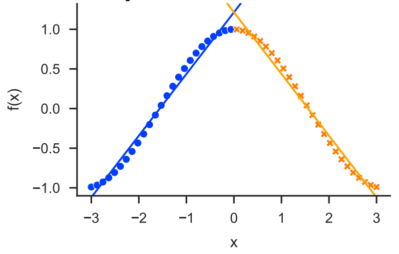
Background

Dimensionality reduction



Dimensionality reduction can be used for manifold visualisation.

Local Explanations



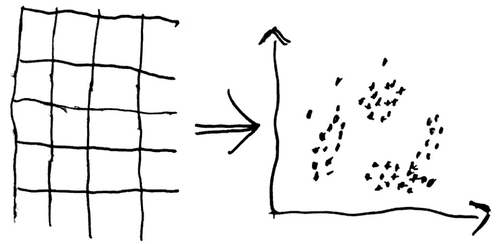
We can approximate this non-linear function with two local linear models.

SLISEMAP = Supervised dimensionality reduction + Local explanations

Explainable artificial intelligence is the field dedicated to investigate how complex black box models work. Local explanations is a type of explanation where we investigate particular predictions, one at a time. A common approach is to locally approximate the complex model with a simpler one.

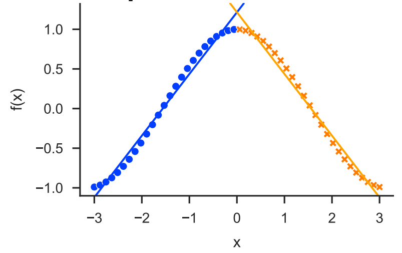
Background

Dimensionality reduction



Dimensionality reduction can be used for manifold visualisation.

Local Explanations



We can approximate this non-linear function with two local linear models.

SLISEMAP = Supervised dimensionality reduction + Local explanations
– *"Find an embedding such that data items with similar local models are next to each other".*

With SLISEMAP we combine these two by finding embedding coordinates and a local model for every data item, such that data items with similar models end up next to each other in the embedding.

PROBLEM DEFINITION

- Given a dataset of n items: $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$.
- Find the embedding coordinates $\mathbf{z}_1, \ldots, \mathbf{z}_n$ and the local models $g_1, \ldots, g_n$ that minimise:

$$\mathcal{L} = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{e^{-\|\mathbf{z}_i - \mathbf{z}_j\|_2}}{\sum_{k=1}^{n} e^{-\|\mathbf{z}_i - \mathbf{z}_k\|_2}} l(g_i(\mathbf{x}_j), \mathbf{y}_j)$$

- Where l is a loss function for the local models.
- Under the constraint $(\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{d} \mathbf{z}_{ik}^2)^{\frac{1}{2}} = z_{radius}$.

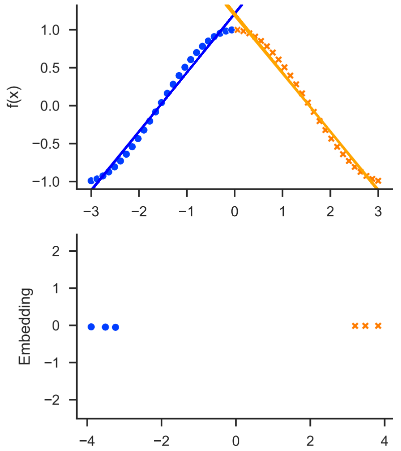
Formally, we express this as a loss function, where the losses from the local models are weighted based on the distances between embedding coordinates. This means that data items with incompatible local models gets pushed far apart. And since we limit the radius of the embedding, data items with similar local models form clusters.

PROBLEM DEFINITION

- Given a dataset of n items: $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$.
- Find the embedding coordinates $\mathbf{z}_1, \ldots, \mathbf{z}_n$ and the local models $g_1, \ldots, g_n$ that minimise:

$$\mathcal{L} = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{e^{-\|\mathbf{z}_i - \mathbf{z}_j\|_2}}{\sum_{k=1}^{n} e^{-\|\mathbf{z}_i - \mathbf{z}_k\|_2}} l(g_i(\mathbf{x}_j), \mathbf{y}_j)$$

- Where l is a loss function for the local models.
- Under the constraint $(\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{d}\mathbf{z}_{ik}^2)^{\frac{1}{2}} = z_{radius}$.



And here we can see the SLISEMAP solution for the simple example from the previous slide. The embedding has two clusters corresponding to the two local models.

USAGE

Installation

```
pip install slisemap
```

Code

```
from slisemap import Slisemap
# Use Lasso regularisation
sm = Slisemap(X, y, lasso=0.01)
# Remember to optimise
sm.optimise()
# Plot the solution
sm.plot(clusters=5, bars=5,
    jitter=0.01, variables=names)
```

SLISEMAP is implemented in Python, using Pytorch for the optimisation, and can be installed via pip. Using SLISEMAP requires only a couple of lines. The two most important being the creation of the SLISEMAP object and the optimisation.

Usage

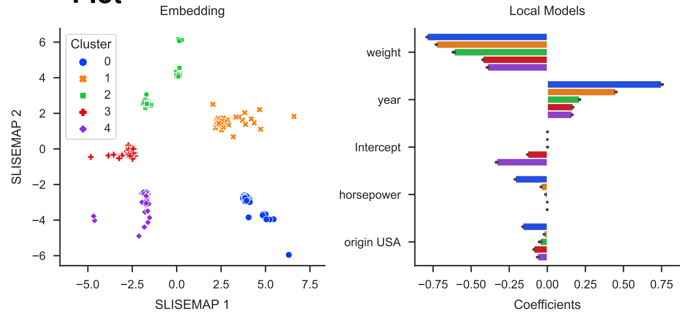
Installation

```
pip install slisemap
```

Code

```
from slisemap import Slisemap
# Use Lasso regularisation
sm = Slisemap(X, y, lasso=0.01)
# Remember to optimise
sm.optimise()
# Plot the solution
sm.plot(clusters=5, bars=5,
    jitter=0.01, variables=names)
```

Plot



Cluster the local models to make them easier to read (using a dataset about cars).

The built-in plotting function returns two plots. The plot on the left is the embedding. And on the right are the local models. To make the models easier to read we have clustered the models by their coefficients, and show the mean models in a barplot. The points in the embedding are colour-coded with the same clusters, showing that the embedding clusters match the model clusters.

SUMMARY

SLISEMAP is a novel supervised manifold visualisation method that embeds data items into a lower-dimensional space such that the same white box model models nearby data items.

A. Björklund, J. Mäkelä, K. Puolamäki.
SLISEMAP: Supervised dimensionality reduction through local explanations.
arXiv: 2201.04455 [cs] (2022).
DOI: 10.48550/arXiv.2201.04455.

`https://github.com/edahelsinki/slisemap`

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

To summarise, SLISEMAP is a new supervised manifold visualisation method that simultaneously finds local models for every data item and constructs an embedding based on the local models.

Please read our arXiv publication for a more in-depth treatment of SLISEMAP. And on GitHub you can find documentation for the Python library, as well as tutorial notebooks.

Thank you for your time and interest.