simpleppt

Release 1.1.3

Louis Faure

Aug 14, 2023

CONTENTS

A python implementation of SimplePPT algorithm, with GPU acceleration.

CHAPTER

ONE

INSTALLATION

```
pip install -U simpleppt
```

CHAPTER

TWO

USAGE

```
from sklearn.datasets import make_classification
import simpleppt

X1, Y1 = make_classification(n_features=2, n_redundant=0, n_informative=2,
                             n_clusters_per_class=1, n_classes=3)

SP = simpleppt.ppt(X1,Nodes=30,seed=1,progress=False,lam=10)
simpleppt.project_ppt(SP, X1, c=Y1)
```

CHAPTER

THREE

CITATION

Please cite the following paper if you use it:

```
Mao et al. (2015), SimplePPT: A simple principal tree algorithm
SIAM International Conference on Data Mining.
```

CHAPTER

FOUR

GPU DEPENDENCIES (OPTIONAL)

If you have a nvidia GPU, simpleppt can leverage CUDA computations for speedup in tree inference. The latest version of rapids framework is required (at least 0.17) it is recommanded to create a new conda environment:

```
conda -n SimplePPT-gpu -c rapidsai -c nvidia -c conda-forge -c defaults cuml=23.04␣
↪cugraph=23.04 python=3.8 cudatoolkit=11.8 -y
conda activate SimplePPT-gpu
pip install simpleppt
```

4.1 API

class simpleppt.**SimplePPT**(*F*, *R*, *B*, *L*, *d*, *score*, *lam*, *sigma*, *nsteps*, *metric*, *tips=None*, *forks=None*, *root=None*, *pp_info=None*, *pp_seg=None*)

A python object containing the data used for dynamical tracks analysis.

Parameters

- **F** (array) – coordinates of principal points in the learned space.
- **R** (array) – soft assignment of datapoints to principal points.
- **B** (array) – adjacency matrix of the principal points.
- **L** (array) – Laplacian matrix.
- **d** (array) – Pairwise distance matrix of principal points.
- **score** (float) – Score minimized during the tree learning.
- **tips** (Optional[Iterable]) – Node IDs of the tree that have degree 1.
- **forks** (Optional[Iterable]) – Node IDs of the tree that have a degree of more than 1.
- **root** (Optional[int]) – Selected node ID as the root of the tree for distance calculations.
- **pp_info** (Optional[DataFrame]) – Per node ID info of distance from the root, and segment assigment.
- **pp_seg** (Optional[DataFrame]) – Per segment info with node ID extremities and distance.

__init__(*F*, *R*, *B*, *L*, *d*, *score*, *lam*, *sigma*, *nsteps*, *metric*, *tips=None*, *forks=None*, *root=None*, *pp_info=None*, *pp_seg=None*)

`set_tips_forks()`

Obtains the tips and forks of the tree.

Returns

adds to SimplePPT object the following fields –

.tips
Node IDs of the tree that have degree 1..

.forks
Node IDs of the tree that have a degree of more than 1.

Return type
simpleppt.SimplePPT

`set_branches(root=None)`

Assign branches/segments to nodes.

Returns

adds to SimplePPT object the following fields –

.pp_info
Per node ID info of distance from the root, and segment assigment.

.pp_seg
Per segment info with node ID extremities and distance.

Return type
simpleppt.SimplePPT

`simpleppt.ppt(X, W=None, Nodes=None, init=None, sigma=0.1, lam=1, metric='euclidean', nsteps=50, err_cut=0.005, device='cpu', gpu_tbp=16, seed=None, progress=True)`

Generate a principal tree.

Learn a simplified representation on any space, composed of nodes, approximating the position of the datapoints on a given space.

Parameters

- **X** – n-dimensionnal matrix to be learned.
- **W** – weight matrix, having the same dimensions as X.
- **Nodes** (Optional[int]) – Number of nodes composing the principial tree.
- **init** (Optional[DataFrame]) – Initialise the point positions.
- **sigma** (Union[float, int, None]) – Regularization parameter.
- **lam** (Union[float, int, None]) – Penalty for the tree length.
- **metric** (str) – The metric to use to compute distances in high dimensional space. For compatible metrics, check the documentation of sklearn.metrics.pairwise_distances if using cpu or cuml.metrics.pairwise_distances if using gpu.
- **nsteps** (int) – Number of steps for the optimisation process.
- **err_cut** (float) – Stop algorithm if proximity of principal points between iterations less than defined value.
- **gpu_tpb** – Threads per block parameter for cuda computations.
- **seed** (Optional[int]) – A numpy random seed.

- **progress** (bool) – Show progressbar of the tree learning.

Returns

SimplePPT object with the following fields –

.F
coordinates of principal points in the learned space.

.R
soft assignment of datapoints to principal points.

.B
adjacency matrix of the principal points.

.L
Laplacian matrix.

.d
Pairwise distance matrix of principal points.

.score
Score minimized during the tree learning.

Return type
simpleppt.SimplePPT

simpleppt.**project_ppt**(*SP*, *emb*, *size_nodes=None*, *plot_datapoints=True*, *alpha_seg=1*, *alpha_nodes=1*, *ax=None*, *show=None*, ***kwargs*)

Project principal graph onto embedding.

Parameters

- **SP** – SimplePPT object.
- **emb** – embedding to project the tree onto.
- **size_nodes** (Optional[float]) – size of the projected prinicpal points.
- **alpha_seg** – segment alpha
- **alpha_nodes** – node alpha.
- **ax** – Add plot to existing ax
- **show** (Optional[bool]) – show the plot.
- **kwargs** – arguments to pass to scanpy functions plt.scatter

Return type
If *show==False* a Axes

PYTHON MODULE INDEX

S

INDEX

Symbols

M

P

S