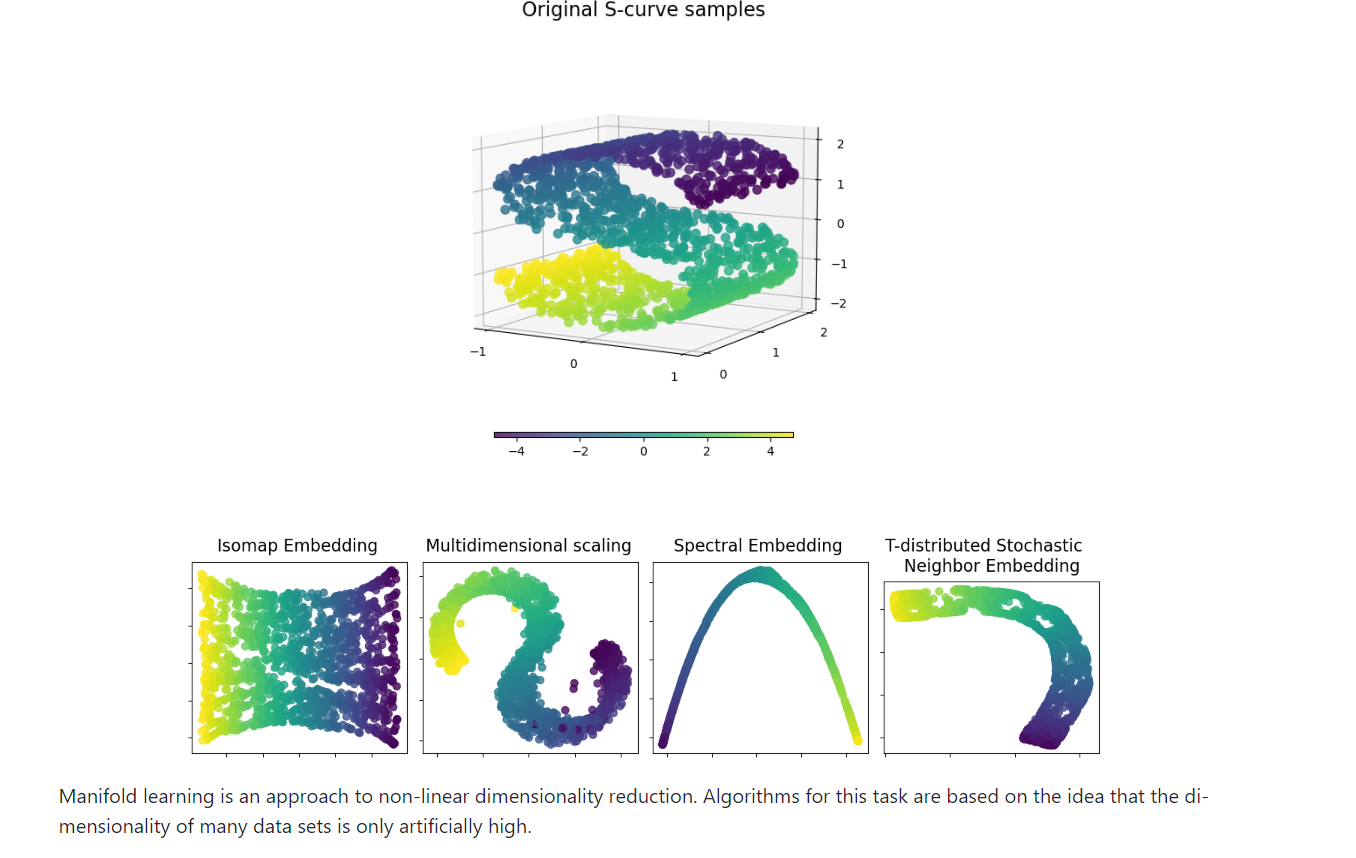
**Spectral Embedding**

Spectral Embedding is an approach to calculating a non-linear embedding. Scikit-learn implements Laplacian Eigenmaps, which finds a low dimensional representation of the data using a spectral decomposition of the graph Laplacian. The graph generated can be considered as a discrete approximation of the low dimensional manifold in the high dimensional space. Minimization of a cost function based on the graph ensures that points close to each other on the manifold are mapped close to each other in the low dimensional space, preserving local distances.

****

**Parameters:**

[**https://scikit-learn.org/stable/modules/generated/sklearn.manifold.SpectralEmbedding.html**](https://scikit-learn.org/stable/modules/generated/sklearn.manifold.SpectralEmbedding.html)**:**

**n\_components*int, default=2***

The dimension of the projected subspace.

**affinity*{‘nearest\_neighbors’, ‘rbf’, ‘precomputed’, ‘precomputed\_nearest\_neighbors’} or callable, default=’nearest\_neighbors’***

**How to construct the affinity matrix.**

* ‘nearest\_neighbors’ : construct the affinity matrix by computing a graph of nearest neighbors.
* ‘rbf’ : construct the affinity matrix by computing a radial basis function (RBF) kernel.
* ‘precomputed’ : interpret X as a precomputed affinity matrix.
* ‘precomputed\_nearest\_neighbors’ : interpret X as a sparse graph of precomputed nearest neighbors, and constructs the affinity matrix by selecting the n\_neighbors nearest neighbors.
* callable : use passed in function as affinity the function takes in data matrix (n\_samples, n\_features) and return affinity matrix (n\_samples, n\_samples).

**gamma*float, default=None***

Kernel coefficient for rbf kernel. If None, gamma will be set to 1/n\_features.

**random\_state*int, RandomState instance or None, default=None***

A pseudo random number generator used for the initialization of the lobpcg eigen vectors decomposition when eigen\_solver == 'amg', and for the K-Means initialization. Use an int to make the results deterministic across calls (See [Glossary](https://scikit-learn.org/stable/glossary.html#term-random_state)).

**Note**

When using eigen\_solver == 'amg', it is necessary to also fix the global numpy seed with np.random.seed(int) to get deterministic results. See <https://github.com/pyamg/pyamg/issues/139> for further information.

**eigen\_solver*{‘arpack’, ‘lobpcg’, ‘amg’}, default=None***

The eigenvalue decomposition strategy to use. AMG requires pyamg to be installed. It can be faster on very large, sparse problems. If None, then 'arpack' is used.

**eigen\_tol*float, default=”auto”***

Stopping criterion for eigendecomposition of the Laplacian matrix. If eigen\_tol="auto" then the passed tolerance will depend on the eigen\_solver:

* If eigen\_solver="arpack", then eigen\_tol=0.0;
* If eigen\_solver="lobpcg" or eigen\_solver="amg", then eigen\_tol=None which configures the underlying lobpcg solver to automatically resolve the value according to their heuristics. See, **[scipy.sparse.linalg.lobpcg](https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.linalg.lobpcg.html" \l "scipy.sparse.linalg.lobpcg" \o "(in SciPy v1.11.3))** for details.

Note that when using eigen\_solver="lobpcg" or eigen\_solver="amg" values of tol<1e-5 may lead to convergence issues and should be avoided.

*New in version 1.2.*

**n\_neighbors*int, default=None***

Number of nearest neighbors for nearest\_neighbors graph building. If None, n\_neighbors will be set to max(n\_samples/10, 1).

**n\_jobs*int, default=None***

The number of parallel jobs to run. None means 1 unless in a **[joblib.parallel\_backend](https://joblib.readthedocs.io/en/latest/generated/joblib.parallel_backend.html" \l "joblib.parallel_backend" \o "(in joblib v1.4.dev0))** context. -1 means using all processors. See [Glossary](https://scikit-learn.org/stable/glossary.html#term-n_jobs) for more details.