

Manifold Learning and Dimensionality Reduction for Data Visualization and Feature Engineering

London, October 2018, mcubed conference

Dr. Stefan Kühn

https://www.xing.com/profile/Stefan_Kuehn46

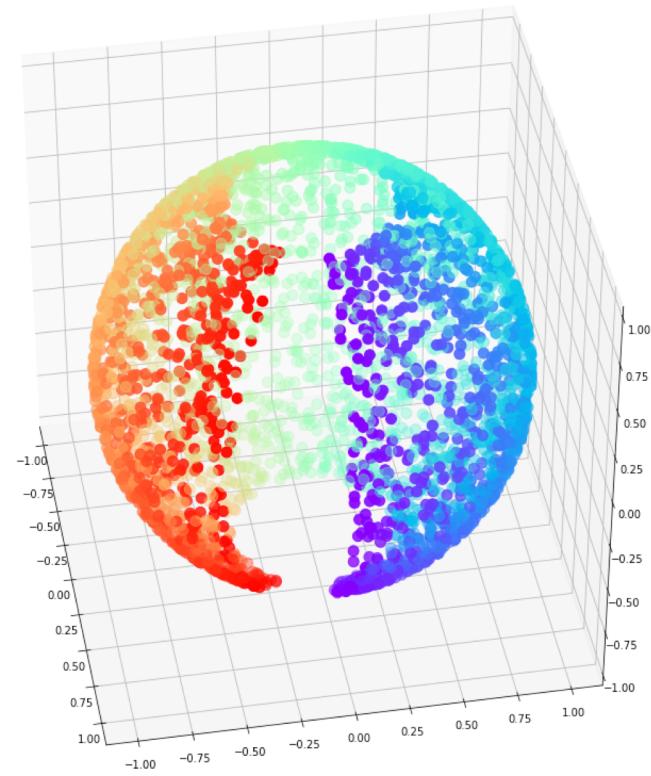
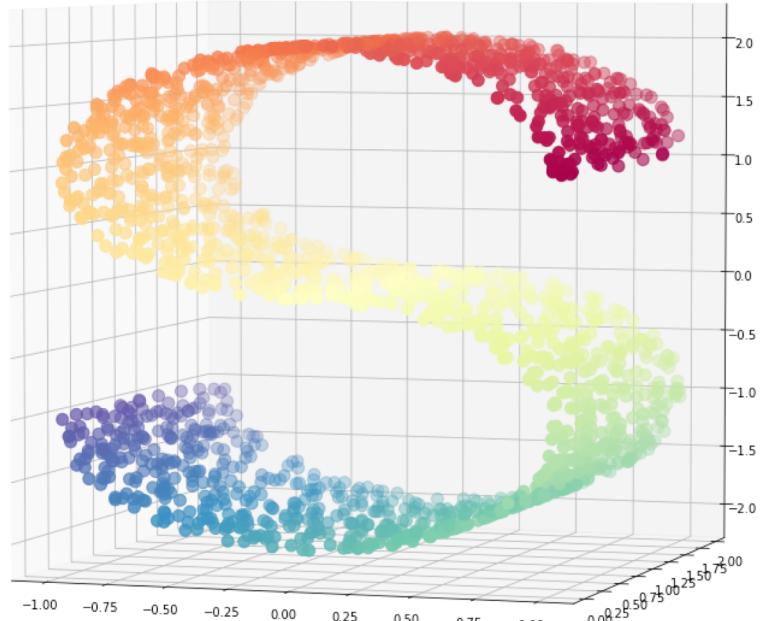
<https://www.linkedin.com/in/stefan-k%C3%BChn-020a34119/>

<https://de.slideshare.net/StefanKhn4>



What is a manifold?

Mathematical concept from Differential Geometry





Why is it a useful concept?

Relativity theory, black holes, and the Poincaré conjecture

- A Manifold locally resembles Euclidean Space
- Allows to generalize useful concepts like Calculus
- Allows to describe geometric properties by means of Calculus
- Einstein's general theory of relativity
- Gravitational lensing and black holes
- Perelman and the proof of the Poincaré conjecture

Ok, that's Differential geometry, but where is Machine Learning?



What are properties of a manifold?

Important Properties – Topology and more

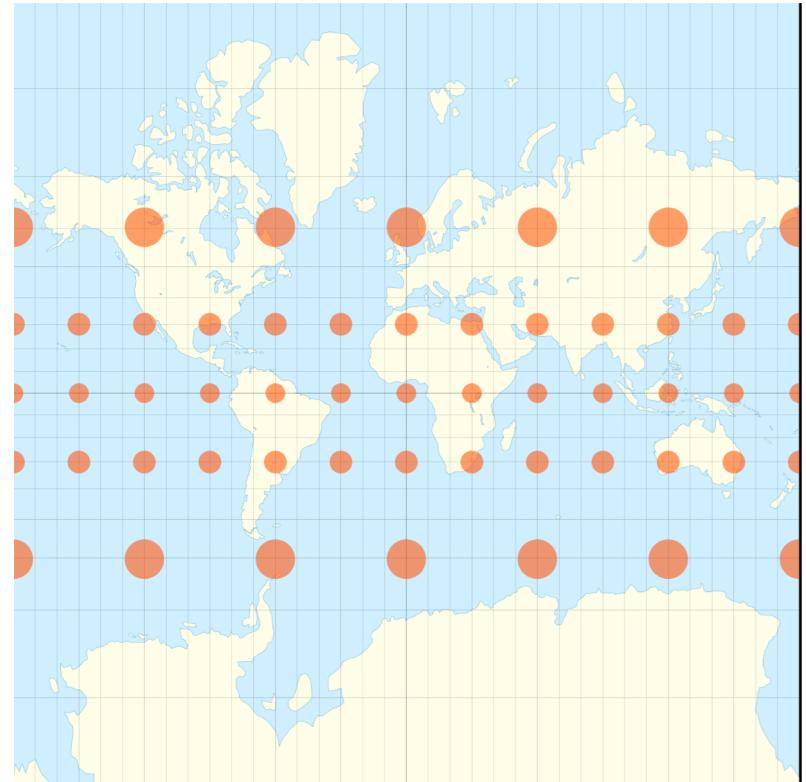
- Number of Connected Components
- Holes
- Curvature
- Smoothness
- Dimensionality
- ...you_name_it...

What are properties of a good visualization?

Preserve important properties

- Number of connected components?
- Holes?
- Curvature?
- Smoothness?
- Dimensionality?
- Distances between points?
- Angles, orientations?
- Local versus global properties?

You cannot have it all!





Which one is correct?

World Mercator projection with country going to true size



@neilrkaye

World Mercator projection with country going to true size



@neilrkaye



What does „Learning“ mean in Manifold Learning?

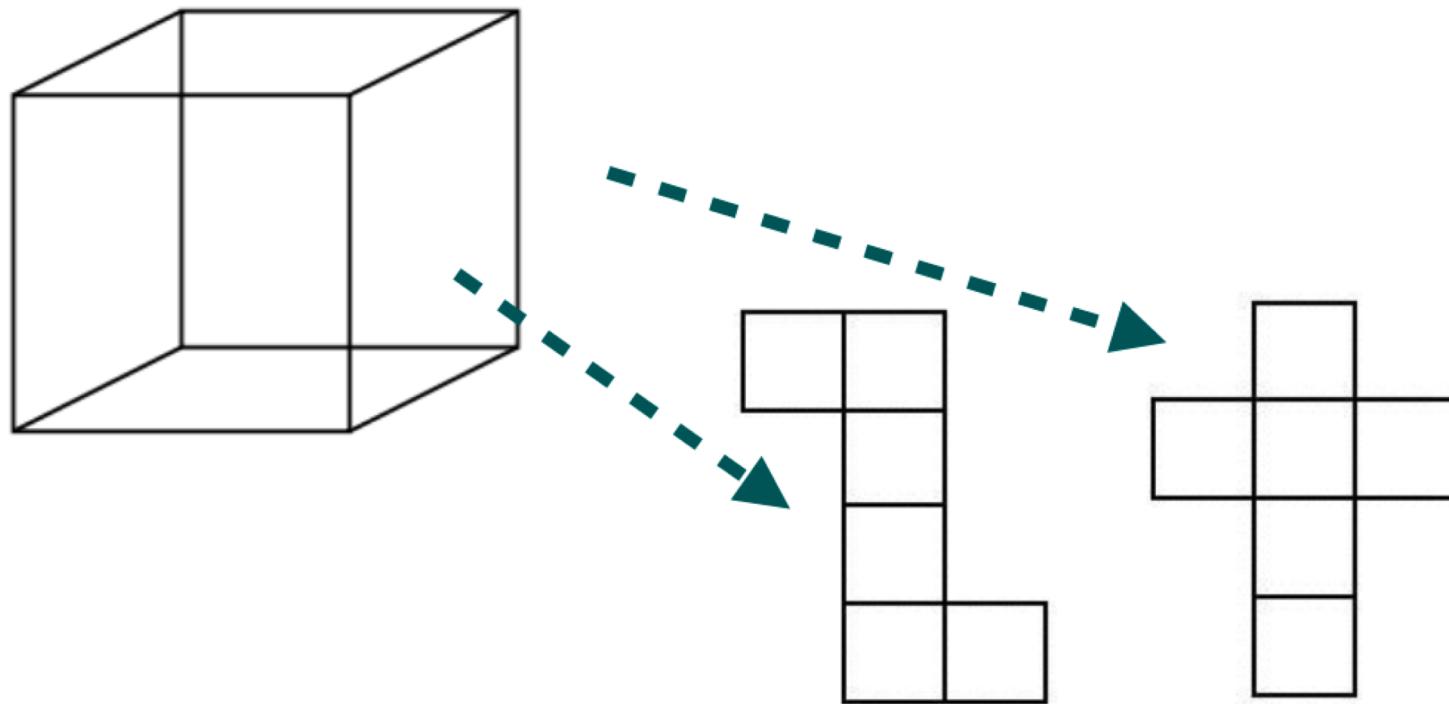
- **Learning the high-dimensional manifold**
 - Somewhat independent from low-dim approximation
 - Mathematical theories can be complex
 - Local versus global properties, distance vs. density etc.
- **Learning the low-dimensional approximation (or embedding)**
 - Depends on properties to be preserved
 - Depends on choice of similarity/distance measure
- **Learning the mapping from high-dim to low-dim**
 - Depends on everything above
- **Learning the structure without assumptions?**
 - *No. All methods make use of explicit and / or implicit assumptions. Sometimes it's hard to figure these out but they determine what you get / see in the end.*



Manifold Learning Methods in sklearn

- **Locally Linear Embedding**
 - Neighborhood-preserving
- **Isomap**
 - Quasi-isometric
- **Multi-Dimensional Scaling (MDS)**
 - Quasi-isometric
- **Spectral Embedding**
 - Spectral clustering based on similarity (Laplacian Eigenmaps)
- **Local Tangent Space Alignment (LTSA)**

Local Tangent Space Alignment





Dimensionality Reduction Methods

- **PCA**
 - Preserves variance
 - Best approximation by linear subspace
 - Comes with quantitative estimation of loss
 - Solves multiple approximations at once
- **Gaussian Random Projections**
 - Preserves distances
 - Based on Johnson-Lindenstrauss lemma
 - Simple and reliable
 - Generates a random projection matrix using a Gaussian
- **Sparse Random Projections**
 - Preserves distances and enforces sparsity
 - Uses a much simpler probability distribution



„Advanced“ Methods – Advanced Math

- **tSNE or t-Distributed Stochastic Neighbor Embedding**
 - Transforms Euclidean distances between points into conditional probabilities for similarity
 - Idea is to find a 2D/3D probability distribution that is similar to the high-dim distribution
 - Learns a mapping that preserves these probabilities
 - Minimizes Kullback-Leibler divergence (somehow)
- **UMAP or Uniform Manifold Approximation and Projection**
 - Based on Differential Geometry and Algebraic Topology
 - Preserves important structural properties
 - If you really want to know more:
 - *Category Theory*
 - *Functors*
 - *Fuzzy Topological Representation*



Demo Time

Sometimes, words are insufficient...

In mathematics, a **manifold** is a topological space that locally resembles Euclidean space near each point. More precisely, One-dimensional manifolds include **lines** and **circles**, but not **figure eights** (because they have *crossing points* that are not local self-intersections) in three dimensional real space, but also the **Klein bottle** and **real projective plane**, which will always self-intersect.

So let's use code!

```
print(__doc__)

from time import time

import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.ticker import NullFormatter
%matplotlib inline
from sklearn import manifold, datasets
```



Resources I

Scikit-learn documentation

<http://scikit-learn.org/stable/modules/manifold.html>

http://scikit-learn.org/stable/auto_examples/manifold/plot_compare_methods.html

http://scikit-learn.org/stable/auto_examples/manifold/plot_manifold_sphere.html

http://scikit-learn.org/stable/modules/random_projection.html

<http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>

Github repo with worked examples

https://github.com/cc-skuehn/Manifold_Learning

Jupyter Lab

<https://jupyterlab.readthedocs.io/en/stable/index.html>

Citation

Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

<http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html>



Resources II

tSNE

[Original tSNE paper \[PDF\]](#)

<https://distill.pub/2016/misread-tsne/>

[Linear tSNE for the Web \[PDF\]](#)

UMAP

[Original UMAP paper \[PDF\]](#)

[Github repo for UMAP \[Python implementation\]](#)

distill.pub is a great resource – there is a lot more to find

Thank you!

London, October 2018

XING 