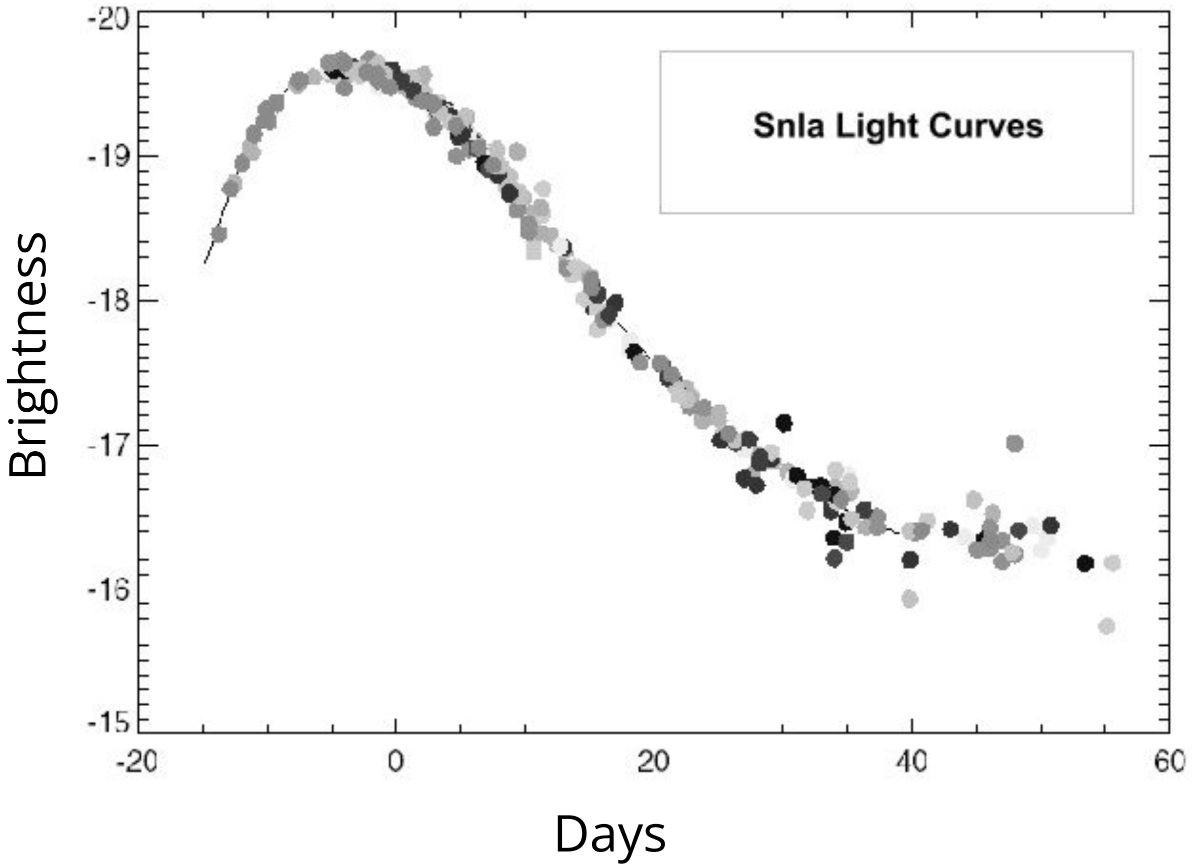
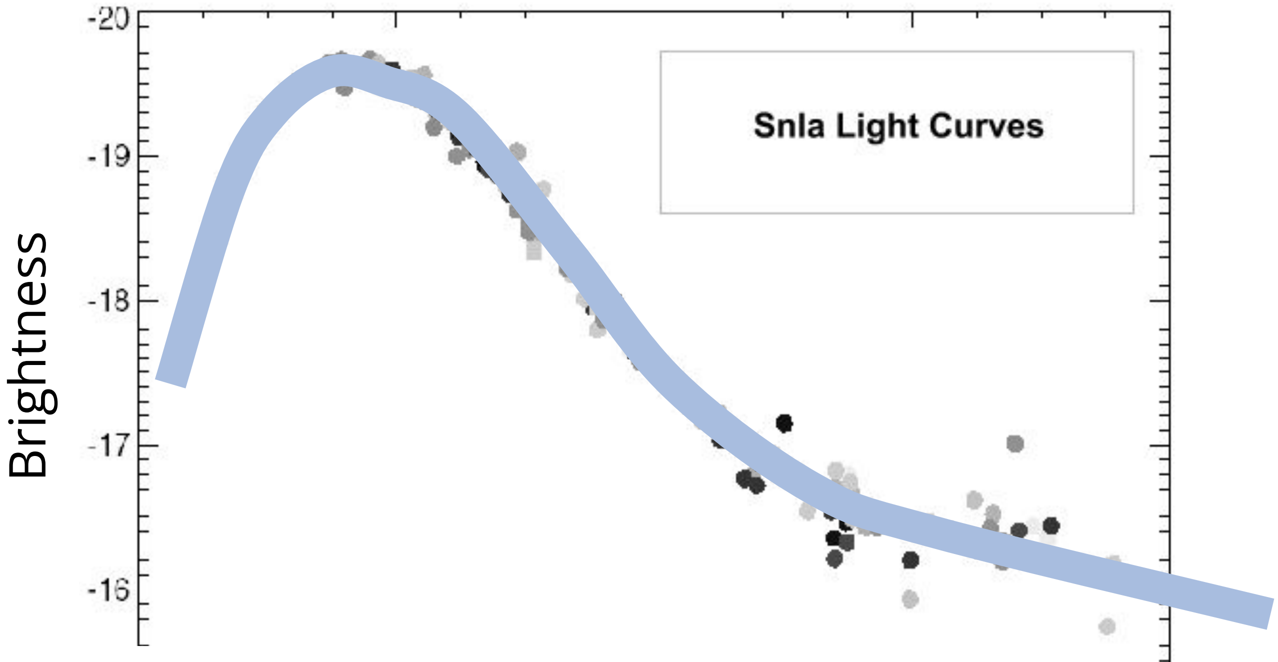
Dimensionality Reduction and Representation Learning

June 14, 2022

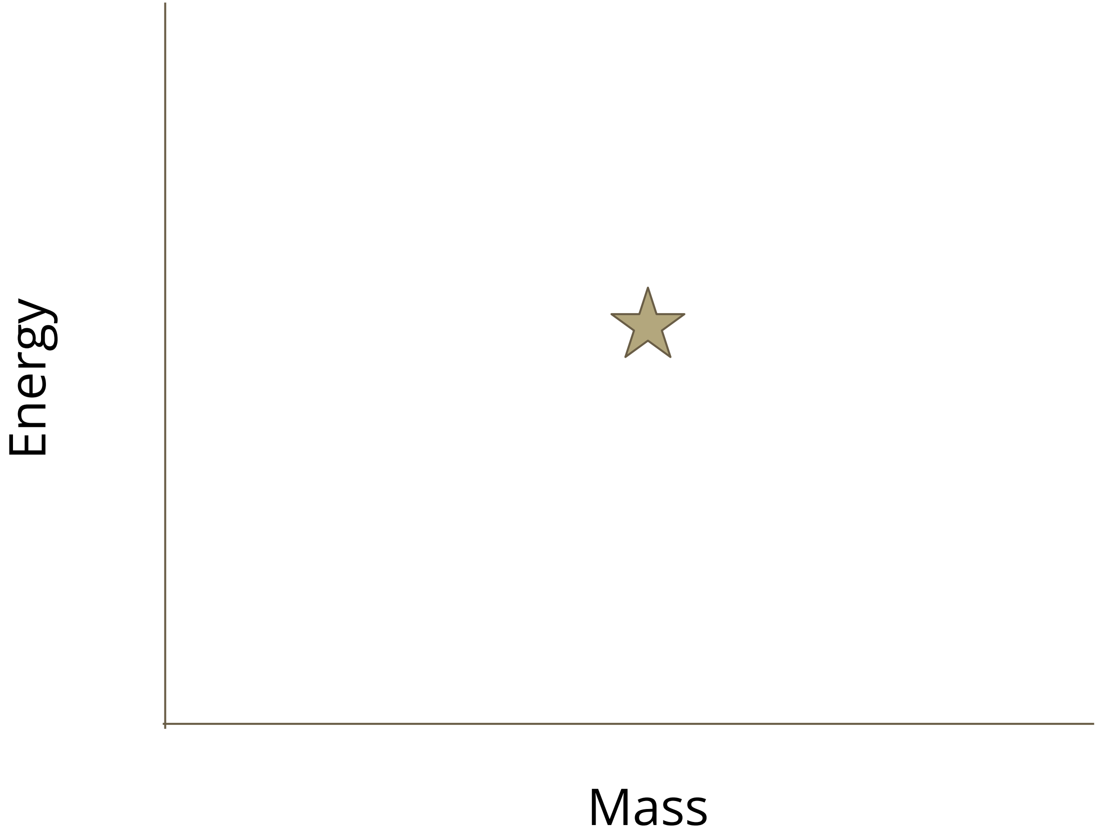
How to fit a model using MOO

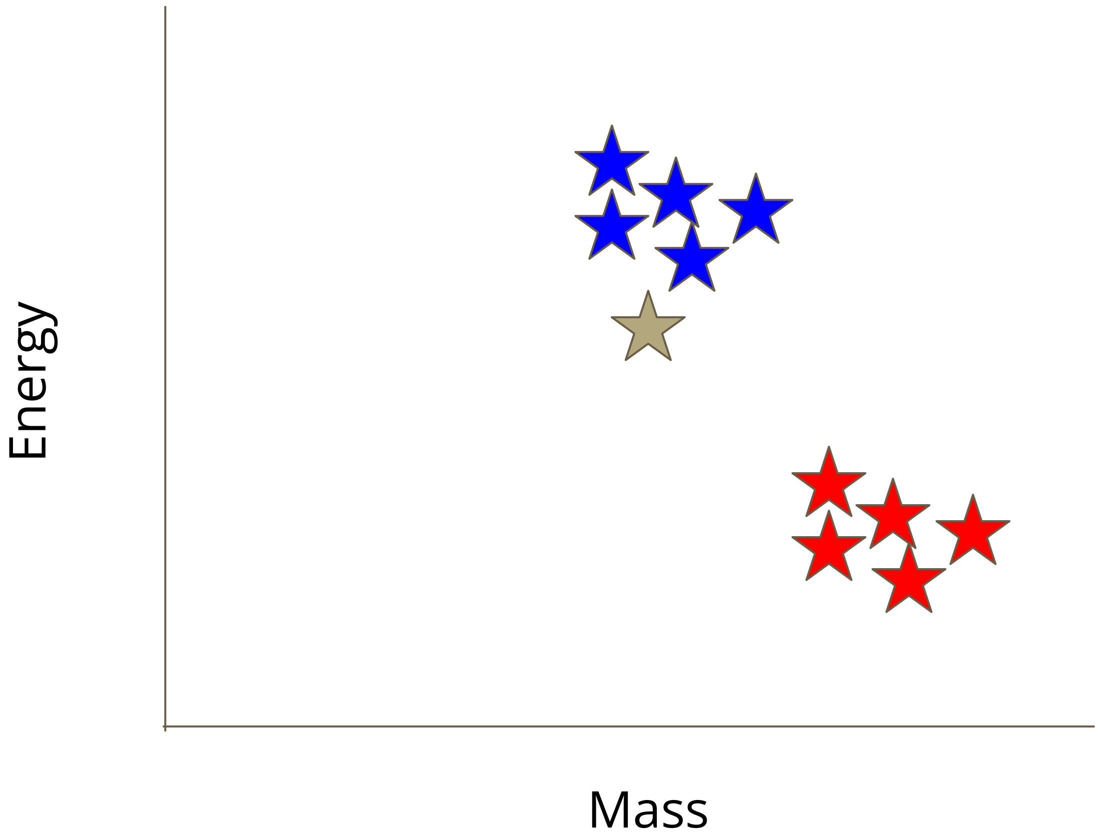
1. **Model** to fit the data (e.g. physics).
2. **Objective Function** (or ‘loss/cost function’) which is a metric that you will choose to quantify how well the model fits the data (e.g. chi-squared).
3. **Optimization Method** which you will use to find the best model (e.g. gradient descent).


SnIa Light Curves
Brightness
Days
-20
-19
-18
-17
-16
-15
-20
0
20
40
60



I can write down a model which only has 3 parameters:
Energy,
Mass,
Amount of radioactive material

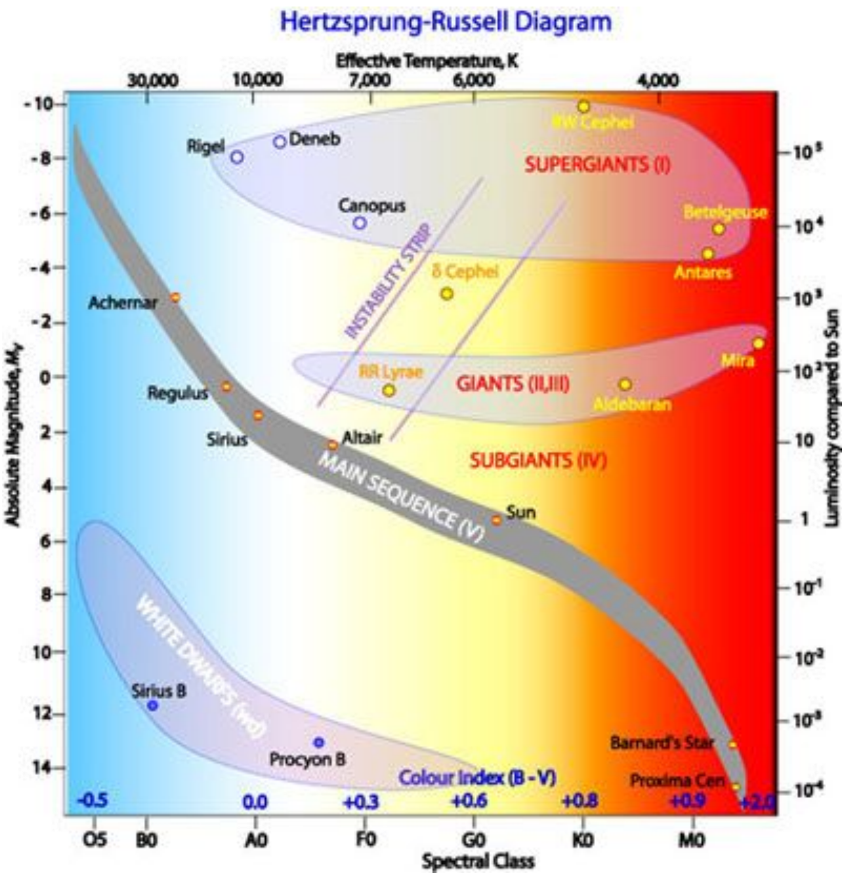

Energy
Mass


Energy
Mass

Let’s call this space a “latent” space. Why “latent”?

How is this different (if at all) from physical model parameters (e.g., mass and energy)?

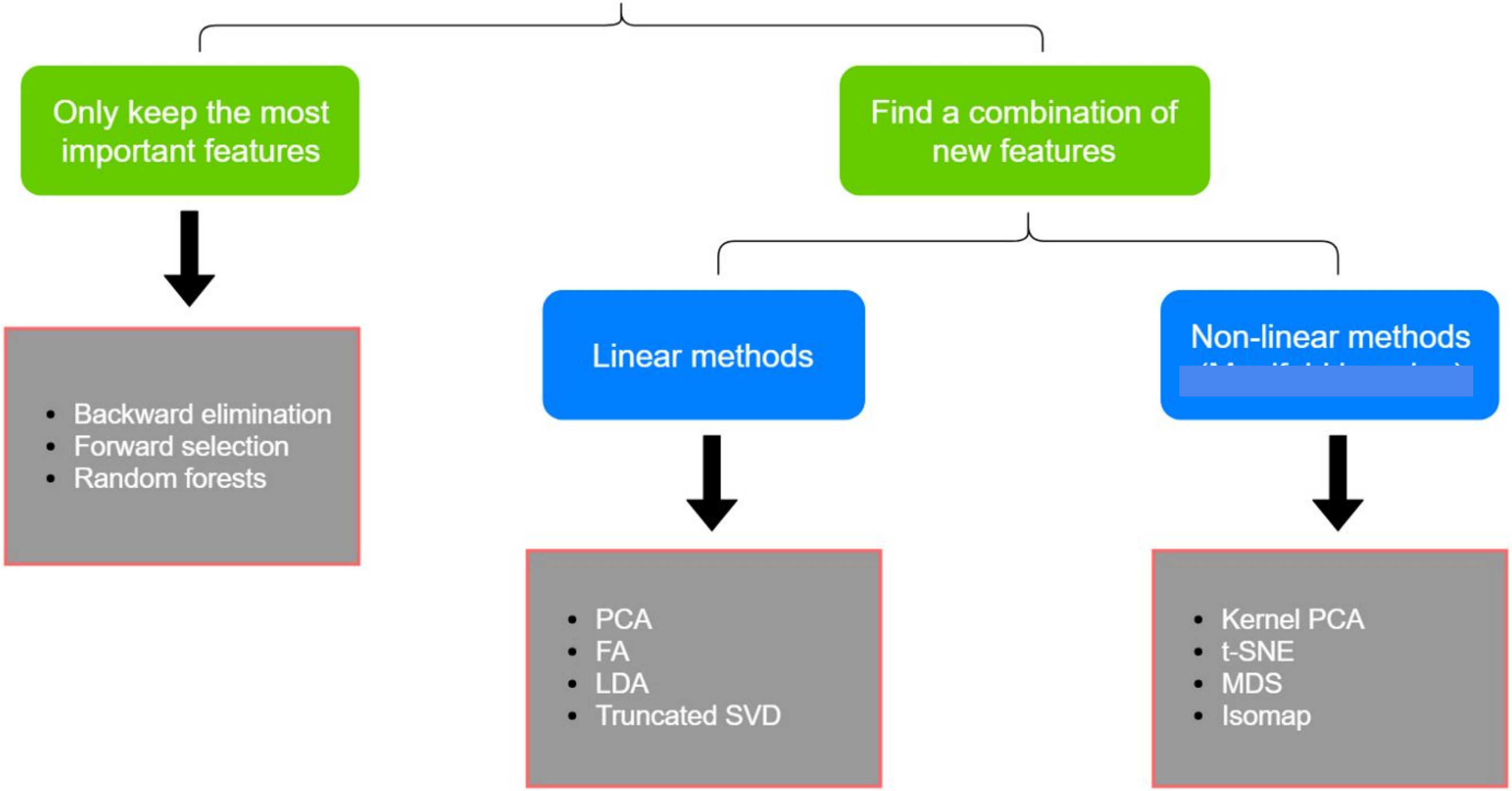
Why do we care about latent spaces?



Once our data is in a low-dimensional space, we can complete a number of tasks:

1. Better feature selection for classification
2. Anomaly detection
3. Data simulations
4. Physical interpretability

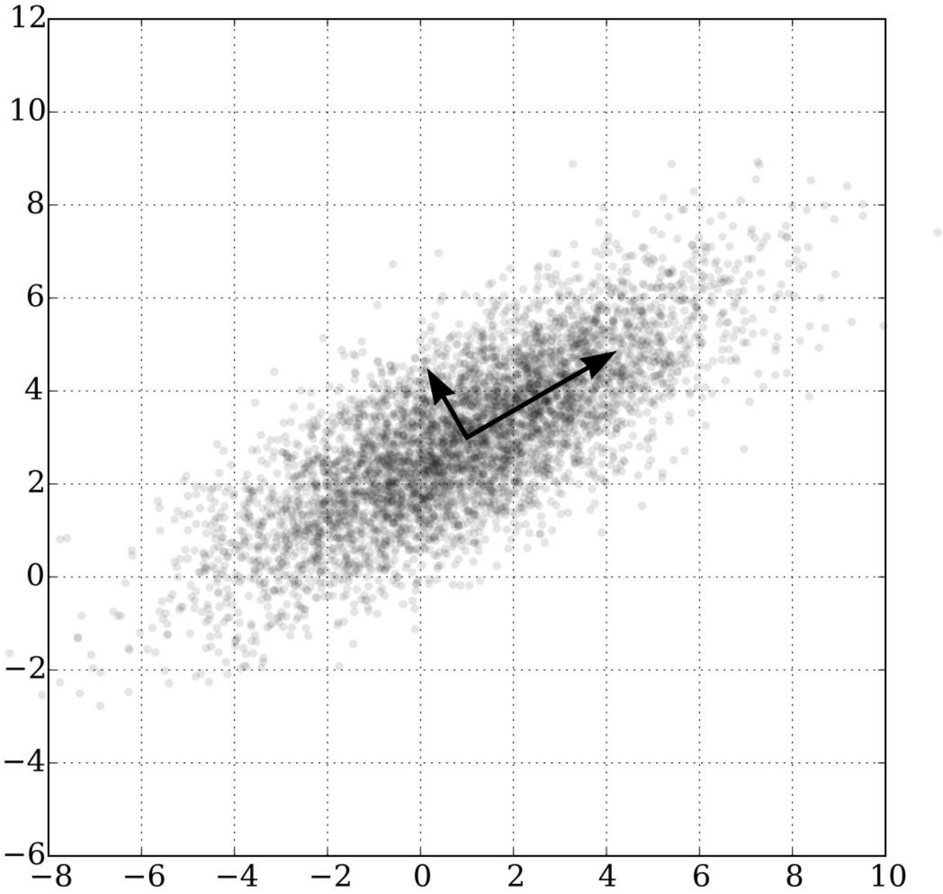
Dimensionality reduction methods



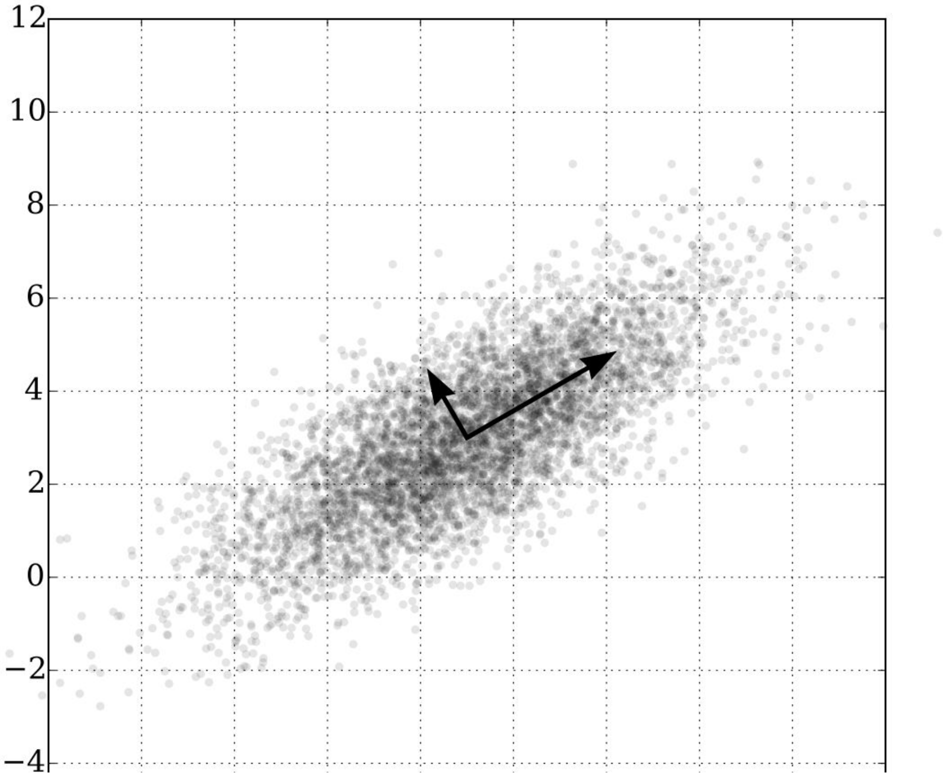
Removing information can be an effective way to remove dimensionality

How can we create a low-dimensional representation without directly fitting a physical model?

The simplest solution is to break down data into **basis vectors**

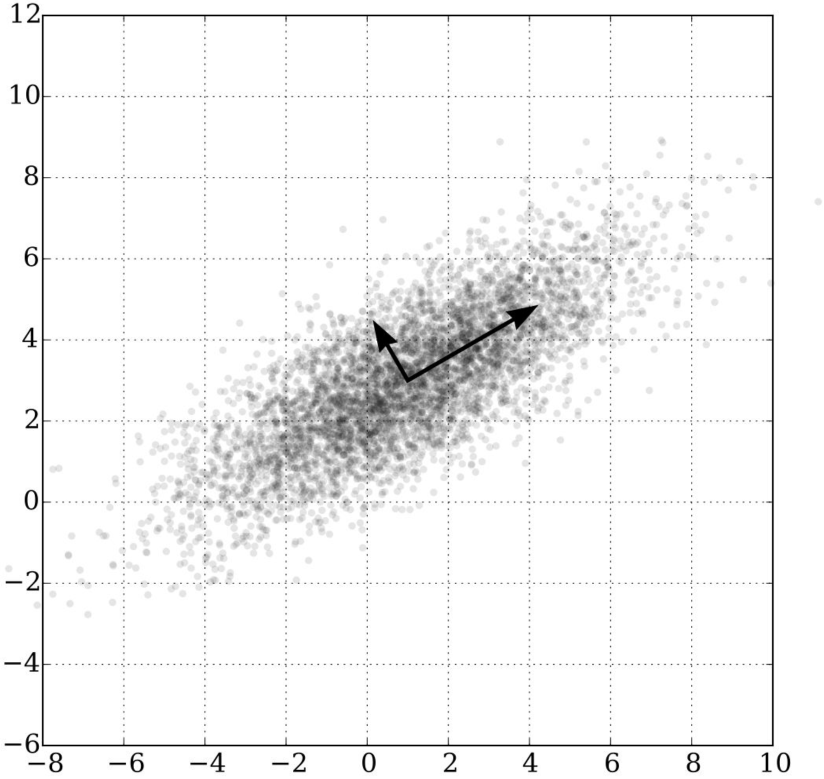


The simplest solution is to break down data into **basis vectors**

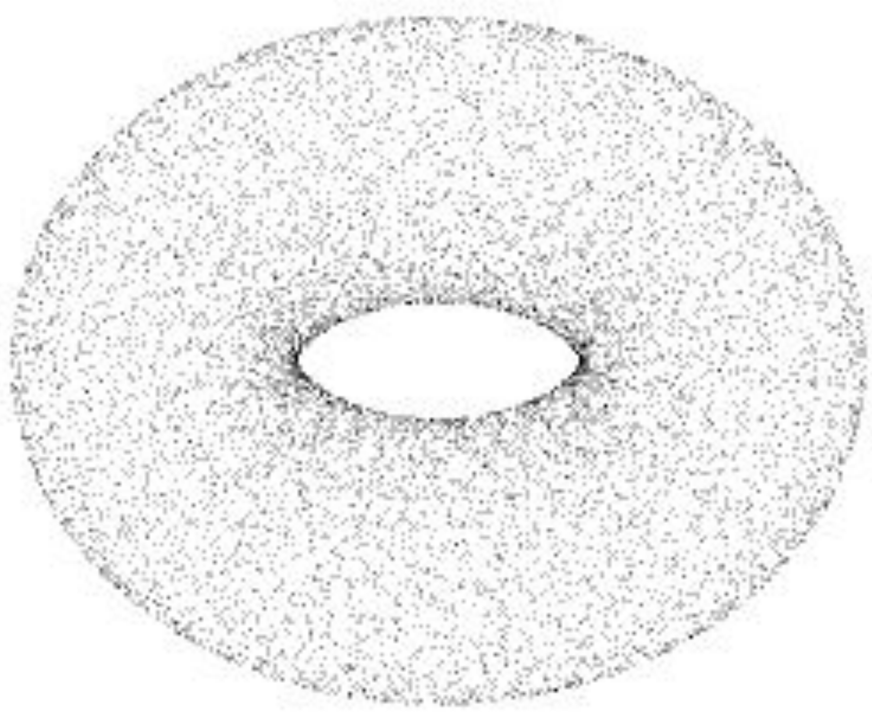


An eigenbasis is one with orthogonal, normalized vectors

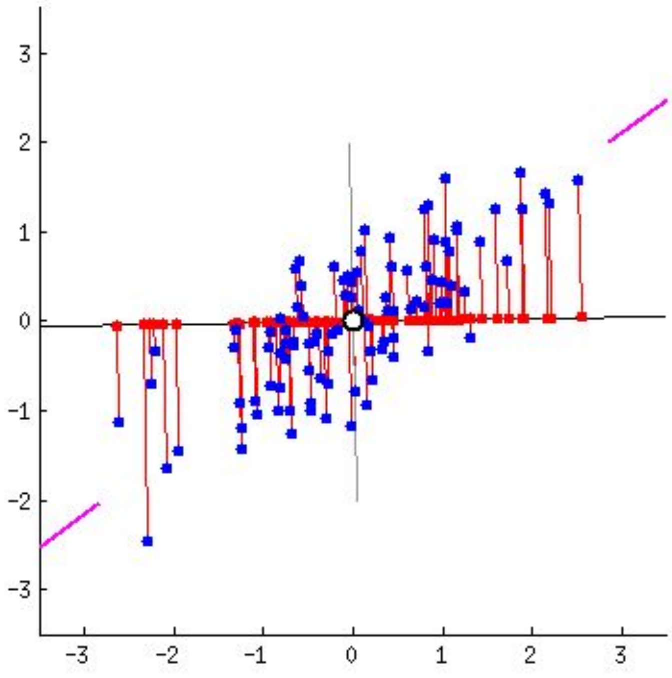
In this 2D example, every point is exactly described by the sum of two eigenvectors



In higher dimensions, 2 detectors may describe ‘most’ of the variance within our data



Each observed datapoint can be projected onto a basis vector

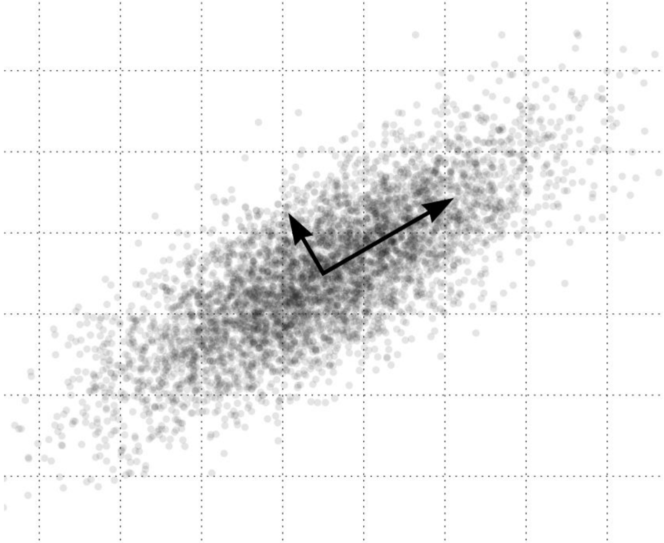


Principal Component Analysis (roughly) has the following steps:

- Find the eigenvectors of the dataset

Principal Component Analysis (roughly) has the following steps:

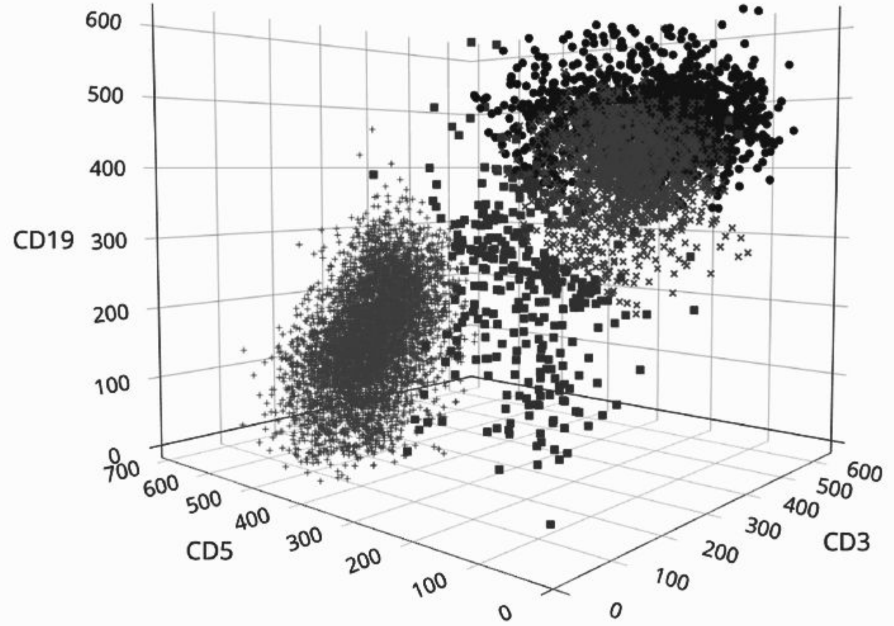
- Find the eigenvectors of the dataset
- Sort the eigenvectors by explained variance
  - Which ones explain the majority of the scatter within the data?

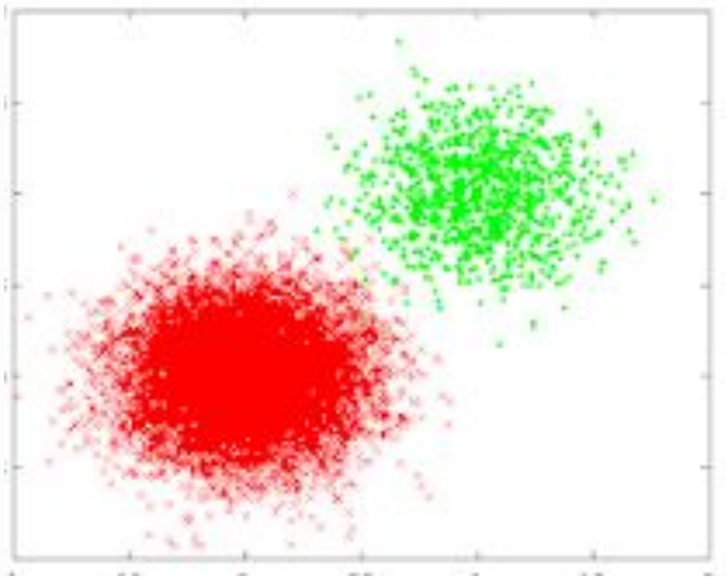


Principal Component Analysis (roughly) has the following steps:

- Find the eigenvectors of the dataset
- Sort the eigenvectors by explained variance
- Project the data onto each basis, tracking the weights
    - For N-dimensional data, each point should be exactly described with N-vectors. So we want to grab the M vectors which describe most of the variance in our data, with M<N

PCA transforms our high-dimensional observed space, to a low-dimension **latent space**



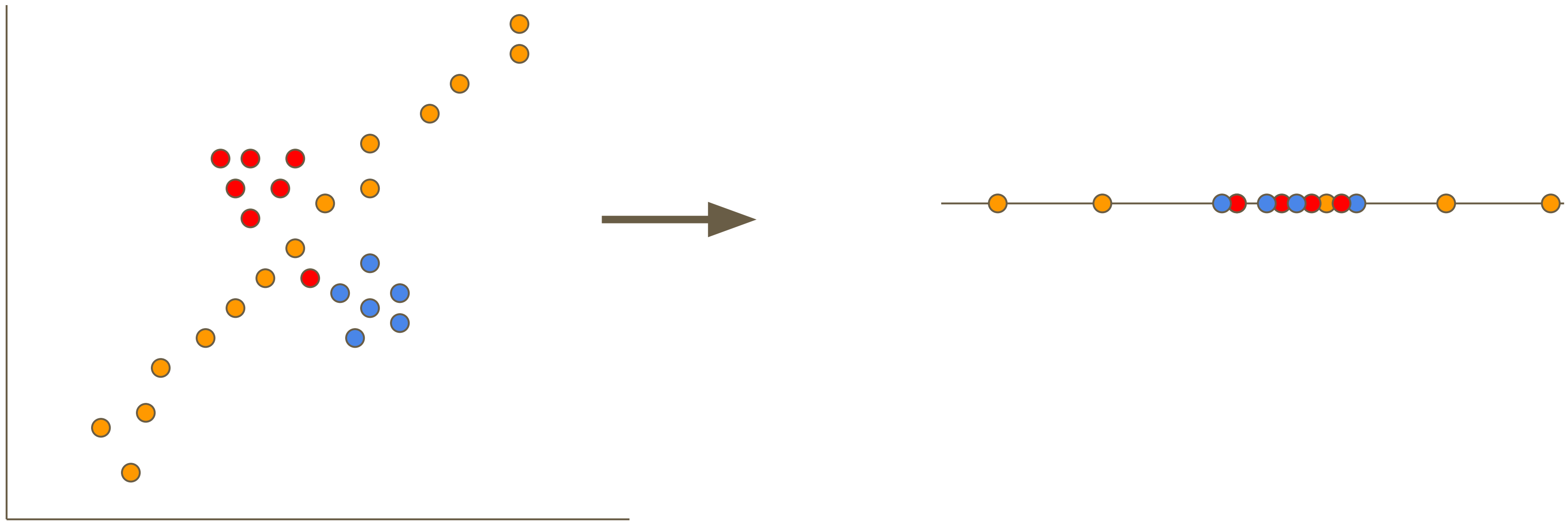


What if my data is high dimensional?

It can be computational expensive to find *every* eigenvector if our data-space is high-dimensional. Instead, we can iteratively find the top k eigenvectors using the **power iteration method**:

1. Given M=XTX, select a random vector v_0
2. For i = 1, 2, . . . , let $v_i = Miv_0$.
3. If $v_i / |v_i| \approx v_{i-1} / |v_{i-1}|$, then return $v_i/|v_i|$ as an approximation for the first component (w_1)
4. Project our data orthogonally to w_1 Repeat steps 1-3 to find the next PCA component. Repeat for k components

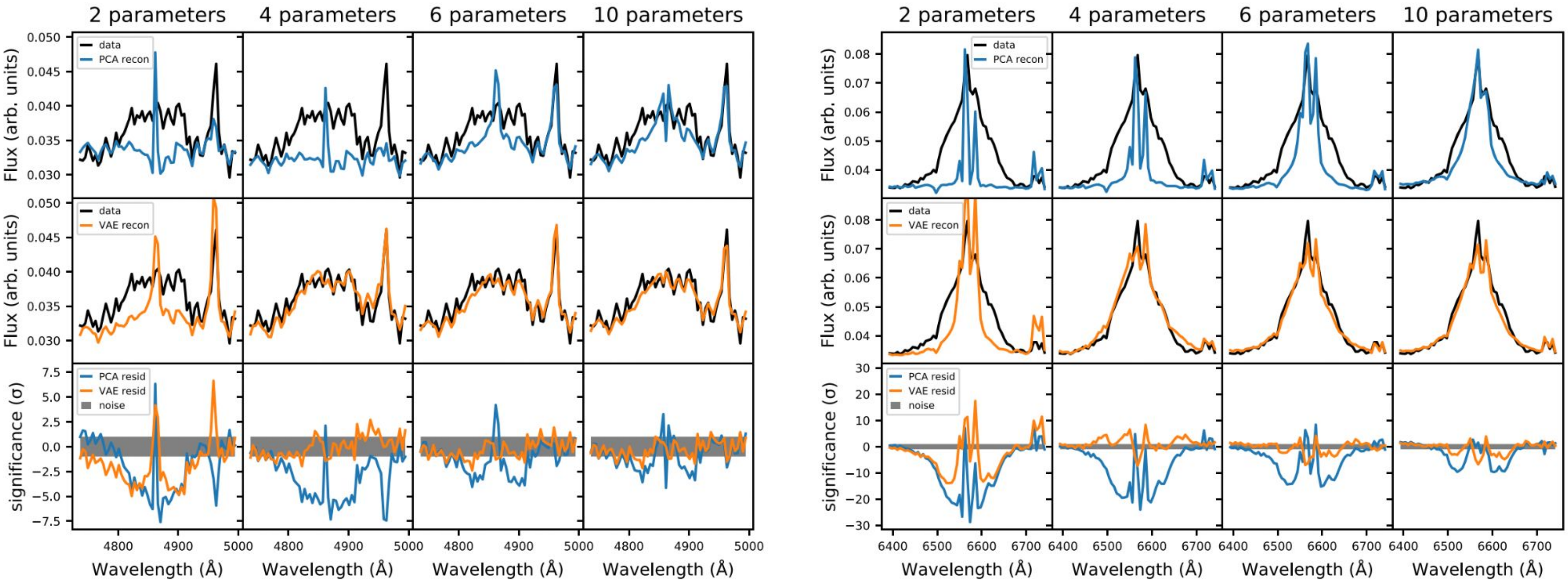
PCA is limited by it linear nature



Group chat - discuss the following questions

1. What does it mean to have a ‘linear’ transformation of our data?
2. Can you give an example of a non-linear transformation in astronomical data?

If a nonlinear transformation affects our data, PCA is insufficient



Portillo et al 2020

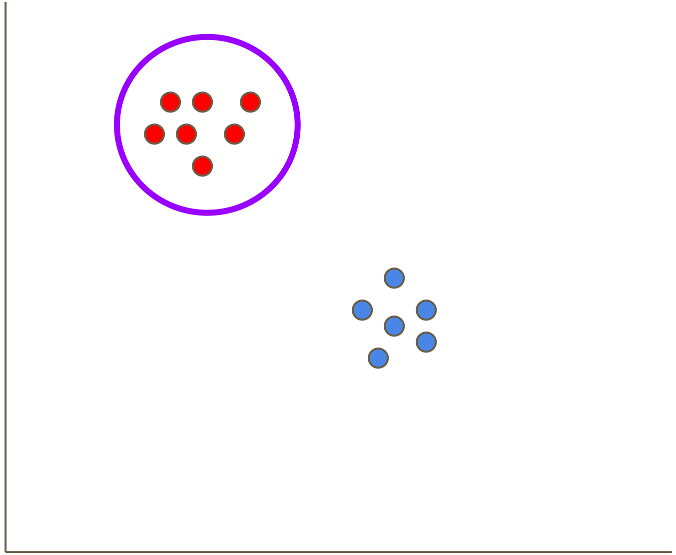
PCA is limited in its linear assumption, but other, more sophisticated,
methods exist as well

t-distributed stochastic neighbor embedding

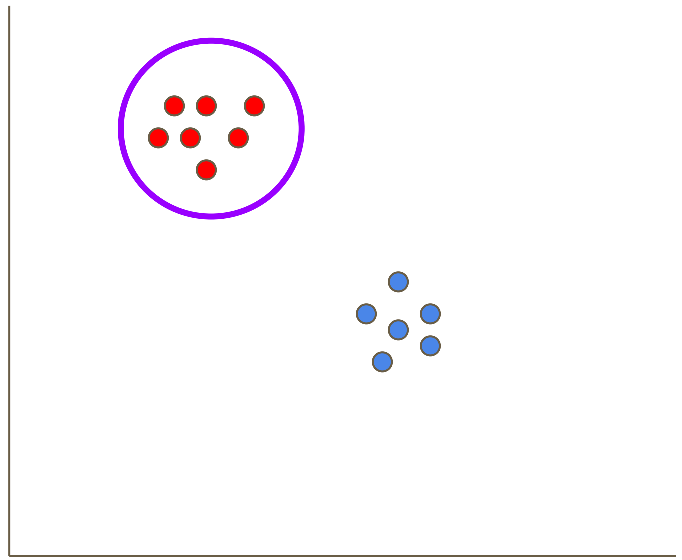
(t-SNE or “tee-snee”)

t-distributed stochastic neighbor embedding

Neighbor embedding: quantify which observations are similar

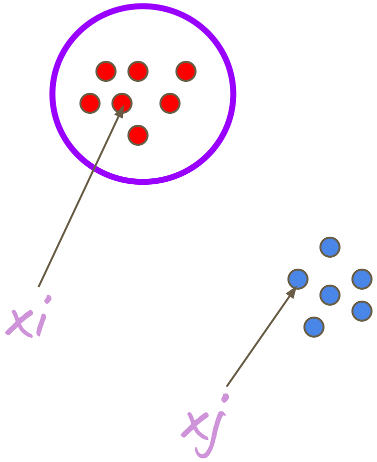


Define a distance metric (e.g., Euclidean distance)



$$d_{i,j}^2 = ||x_i - x_j||^2$$

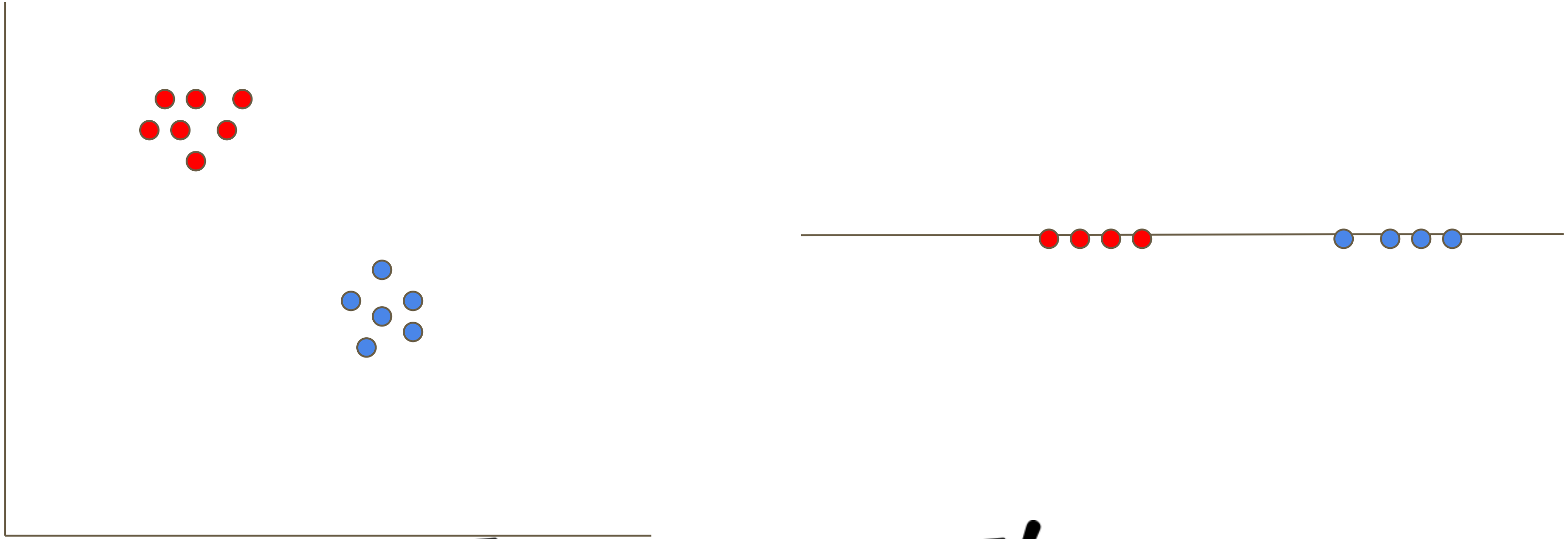
Think of this distance as being proportional to the chance that observations are "neighbors"



$$p_{i|j} \propto \exp\left(-\frac{||x_i - x_j||^2}{2\sigma_j^2}\right)$$

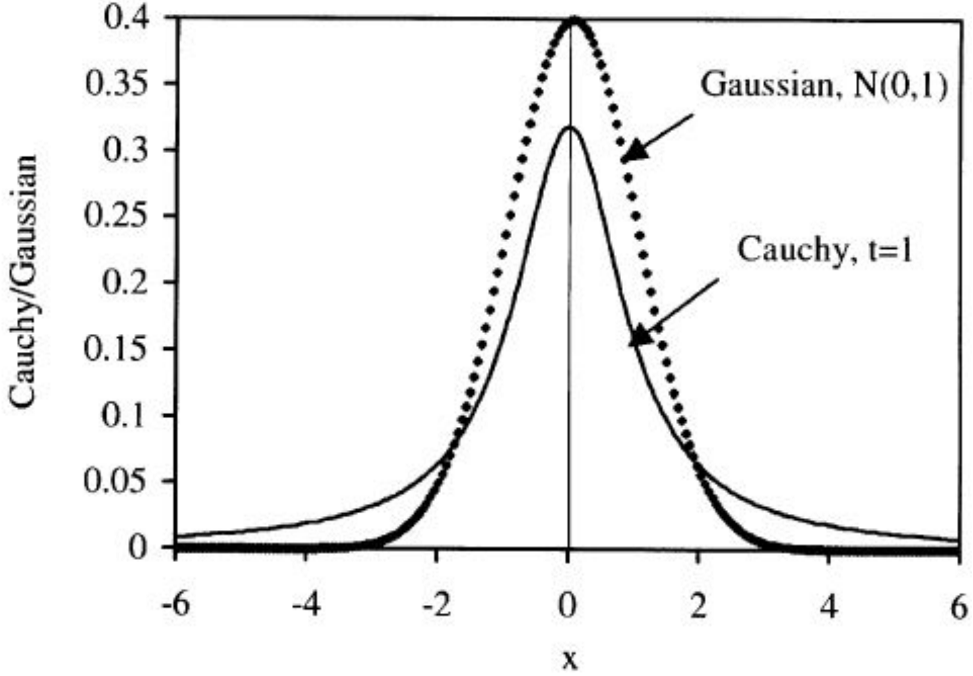
Free parameter to tune

t-SNE aims to learn an embedding which preserves distance measures in the latent space



$$d_{i,j} \simeq d'_{i,j}$$

There are cases in which we *can't* achieve this embedding perfectly, but to help, we will use a Student **t-distribution**



Intuition: Try to match distances for neighbors, but the distance between neighborhoods can be fudgey

In observed space:

$$p(x_i|x_j)$$

In observed space:

$$p(x_i|x_j)$$

In latent space:

$$q(x'_i|x'_j)$$

In observed space:

$$p(x_i|x_j)$$

In latent space:

$$q(x'_i|x'_j)$$

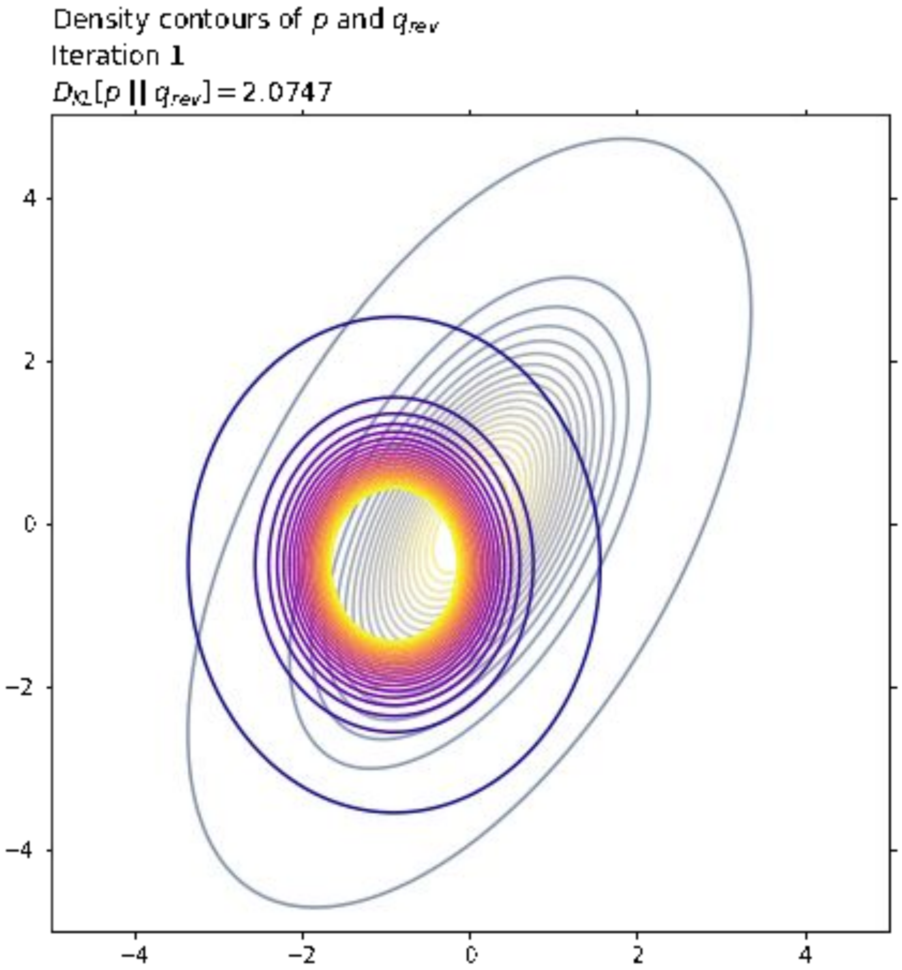
We want to minimize the difference between these distributions

Kullback–Leibler divergence

$$D_{KL}(P||Q) = \sum_{x_i \in X} P(x_i) \log \left(\frac{P(x_i)}{Q(x_i)} \right)$$

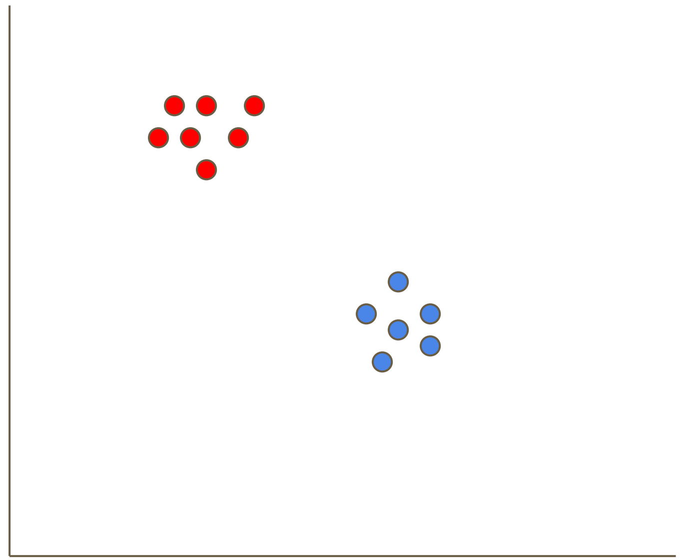
KL divergence of KL(p|q) tries to match the probability distributions of p(xi|xj) and q(x’i|x’j)

We will minimize the KL divergence using gradient descent



t-distributed stochastic neighbor embedding

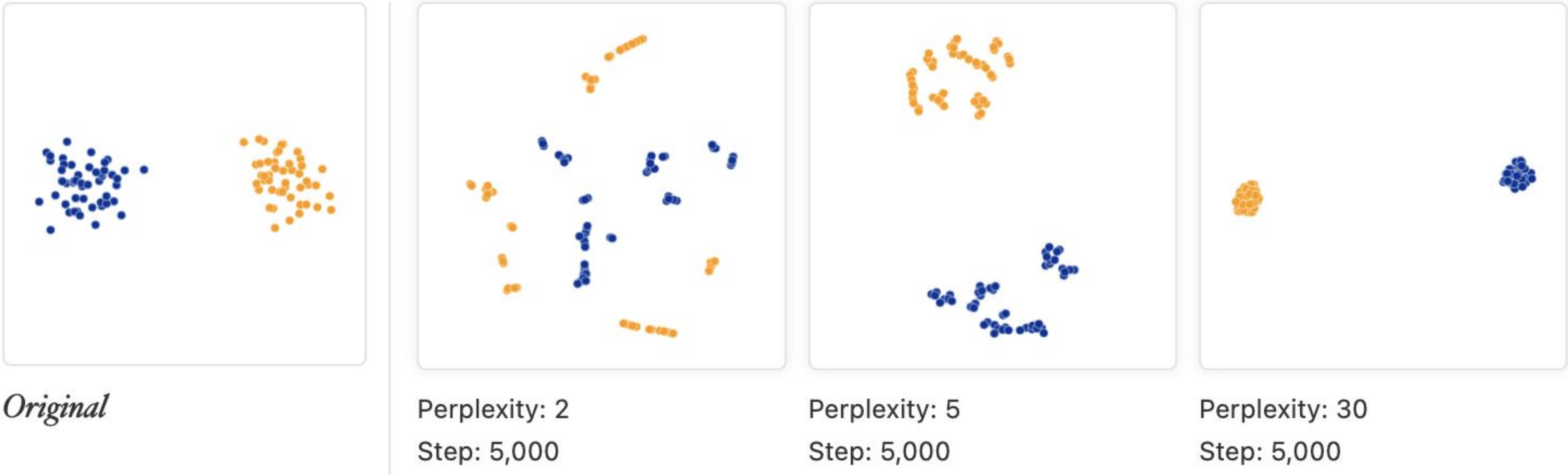
Randomly choose points while optimizing distance metrics….



So while PCA is deterministic, t-SNE is a stochastic method

“Perplexity” is important!

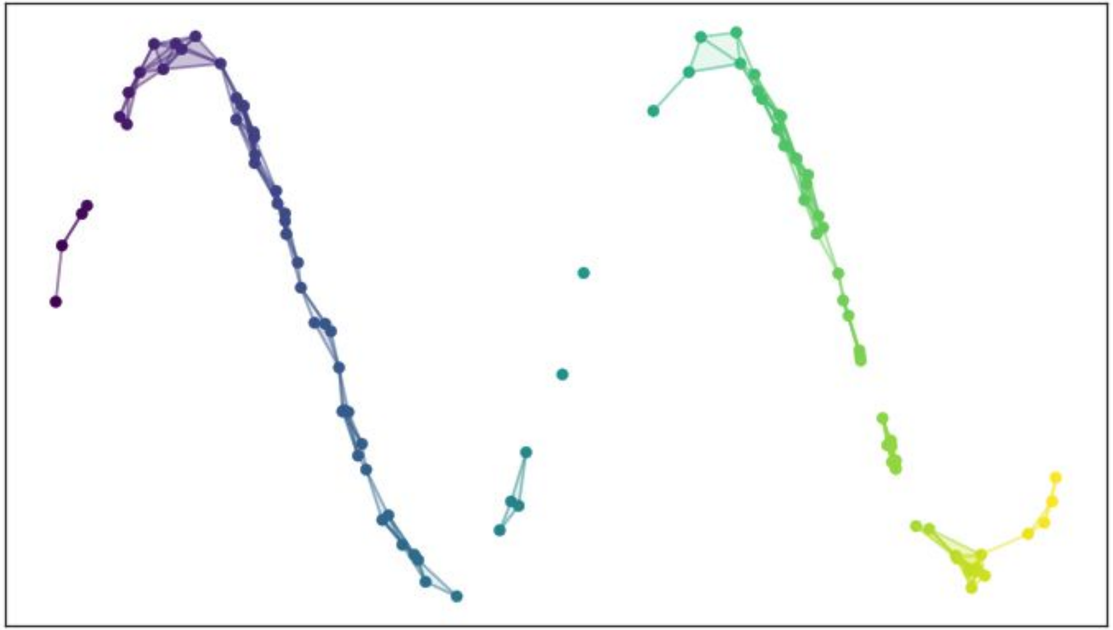
Perplexity ~ number of points expect in each cluster, a hyperparameter



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

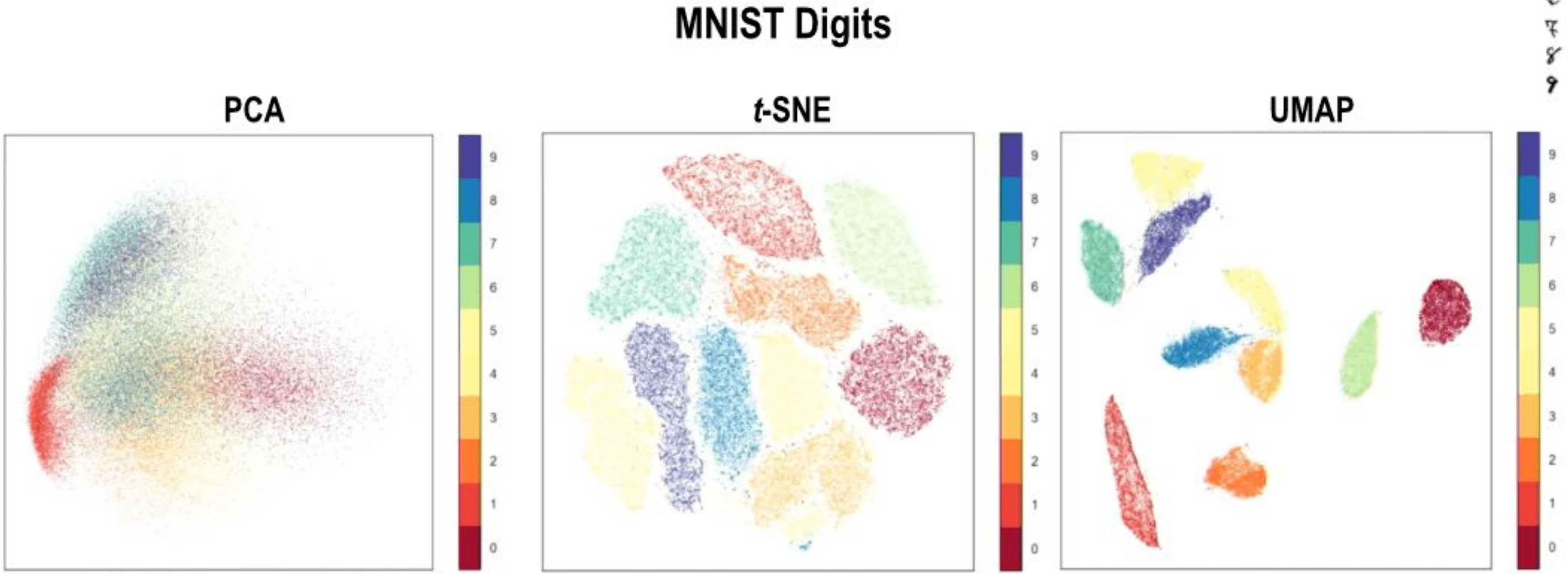
Another common dimensionality reduction techniques

UMAP: Uniform Manifold Approximation and Projection

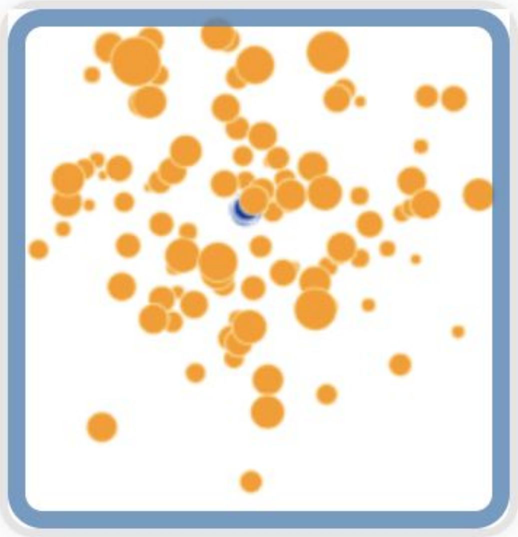


Read more: https://pair-code.github.io/understanding-umap/

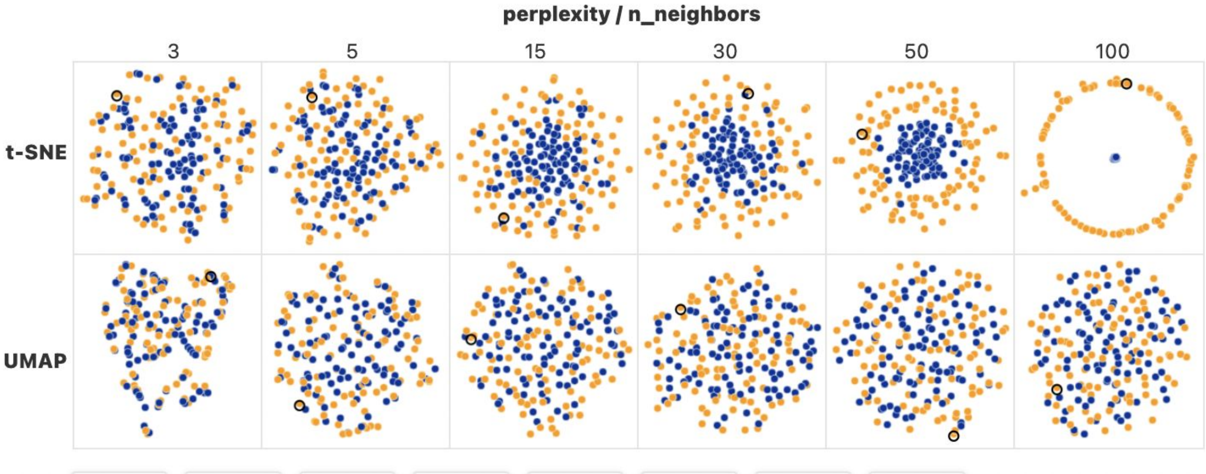
Clustering MNIST (digits)



So is UMAP always superior? No!



A dense, tight cluster inside of a wide, sparse cluster.



Pros and Cons of each method

PCA

Computationally fast

Simple interpretations of latent spaces

Limited expressiveness due to linearity

t-sne

Preservation of local structure

Performs especially well in 2D cases

Extremely expensive

Complex interpretation

U-MAP

Better preservation of local and *global* structure

Computationally much faster than t-sne, slower than PCA

Complex interpretation

What can we do with a low-dimensional latent space?

Clustering: Look for groups without knowing underlying “labels”

What can we do with a low-dimensional latent space?

Clustering: Look for groups without knowing underlying “labels”

Classification: With labels, train a supervised method to use the learned latent space to guess the class of many objects

What can we do with a low-dimensional latent space?

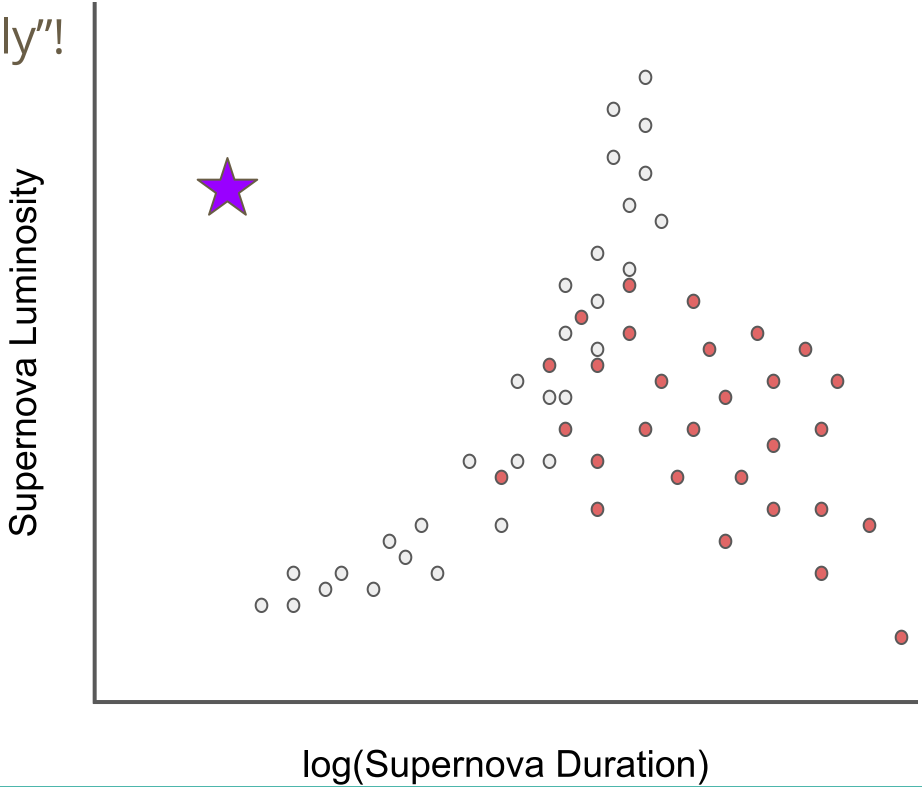
Clustering: Look for groups without knowing underlying “labels”

Classification: With labels, train a supervised method to use the learned latent space to guess the class of many objects

Anomaly detection

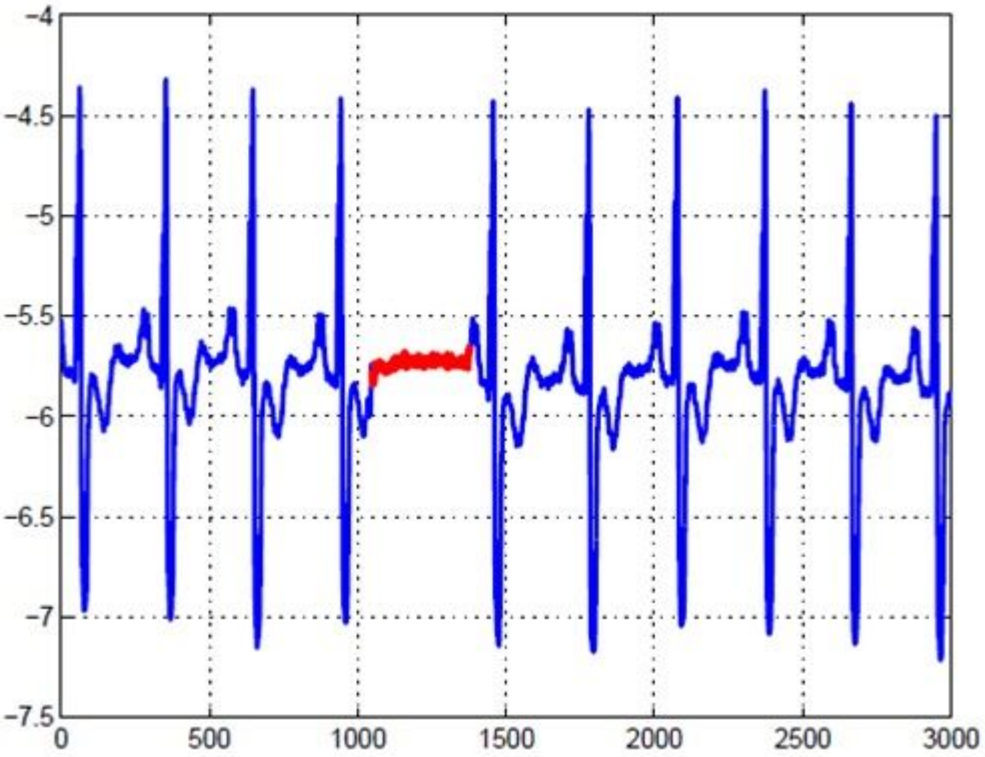
Anomaly Detection - three types

Point Anomalies - An object which standard out from all others in some space. This is the type we'll focus on, and probably your first thought when you hear "anomaly"!



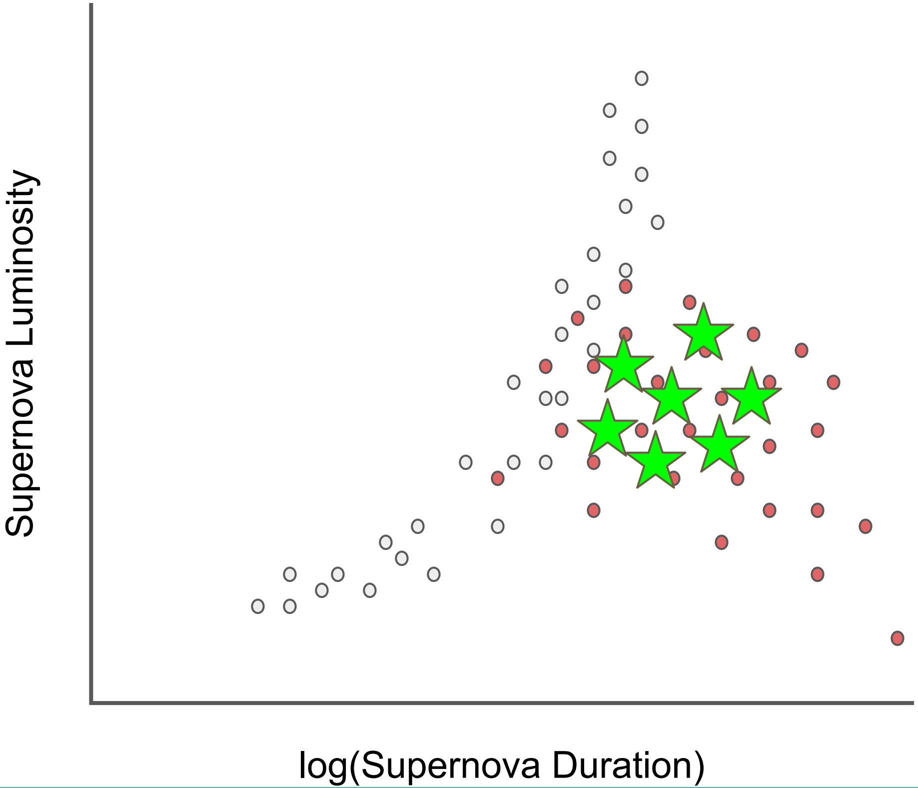
Anomaly Detection - three types

Contextual Anomalies - An object which is anomalous given some context.



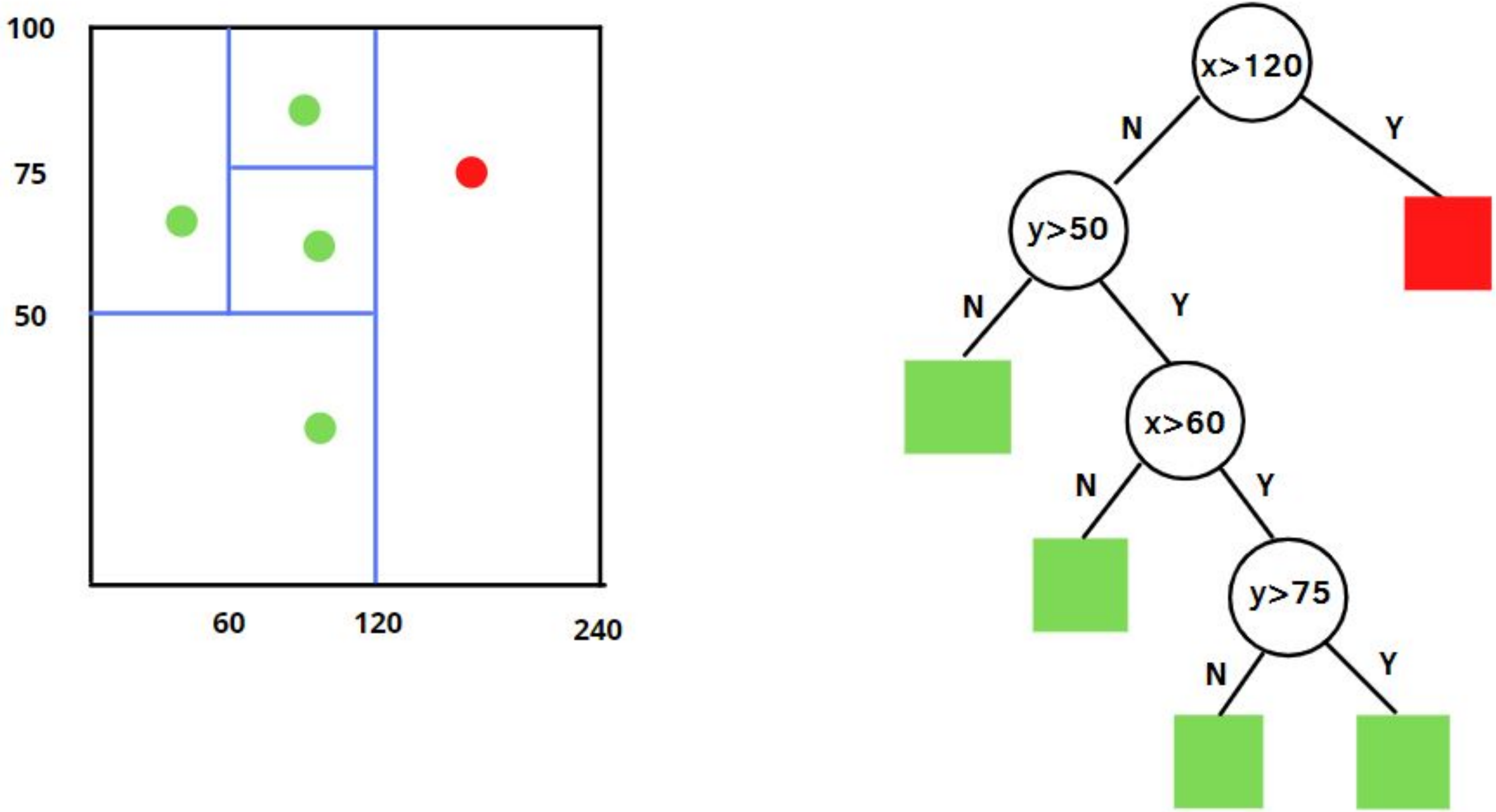
Anomaly Detection - three types

Collective Anomalies - An anomalous pileup of objects, which stick out when pooled together.



One simple way to find out-of-distribution (point) anomaly detection: Isolation Forests

Isolation Forest: A Random Forest w/o Classification



What makes this different than a random forest?

What makes this different than a random forest?

- An isolation forest is unsupervised vs supervised
- The decision splits are random and do not minimize an impurity

Each object is assigned an anomaly score

This score is related to how many splits it takes to isolate an object (averaged across trees). More anomalous objects require fewer splits.

Although there is some care in the normalization of this “score”, it is generally not meant to be **interpretable**. Instead, use this score to *rank* objects.

See the original paper, Liu, Ting & Zhou 2008

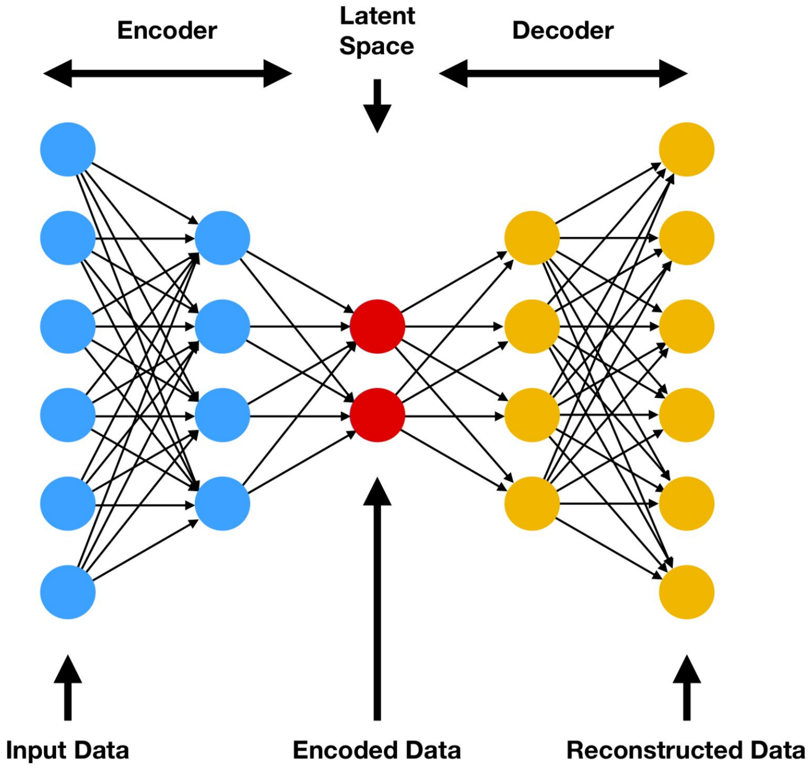
What are some considerations for creating a latent space?

- Orthogonality: Do you want to ensure that no features correlate within this space? (like PCA)

- Orthogonality: Do you want to ensure that no features correlate within this space? (like PCA)
- Maintaining some manifold shape: Do you care that similar objects in the original space still appear similar (like t-sne)? What about dissimilar objects?

- Orthogonality: Do you want to ensure that no features correlate within this space? (like PCA)
- Maintaining some manifold shape: Do you care that similar objects in the original space still appear similar (like t-sne)? What about dissimilar objects?
- Interpretability: Do each of your latent dimensions need to have some clear, physical interpretation? What limits your ability to fit that model to the data?

A quick teaser for autoencoders



Conclusions

- Dimensionality reduction (or “representation learning”) is an invaluable tool for understanding “Big Data”
- We covered linear and nonlinear methods (PCA, t-SNE, UMAP)
- We discussed potential applications of low-dimensional latent spaces (anomaly detection)
- We discussed some considerations in building these latent space

Questions?