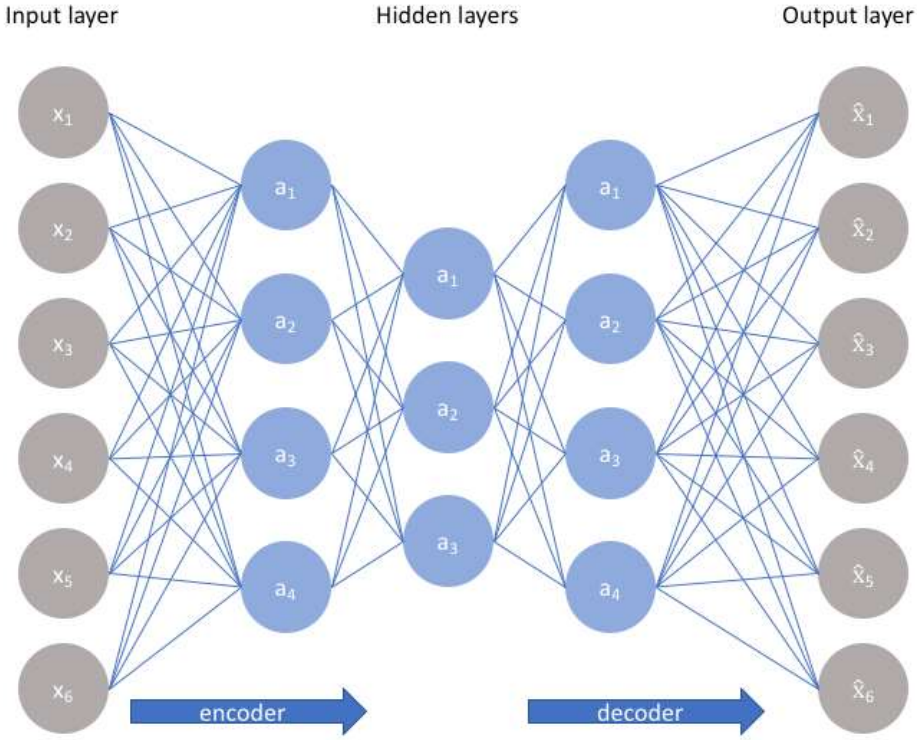
Module 3

Undercomplete autoencoder

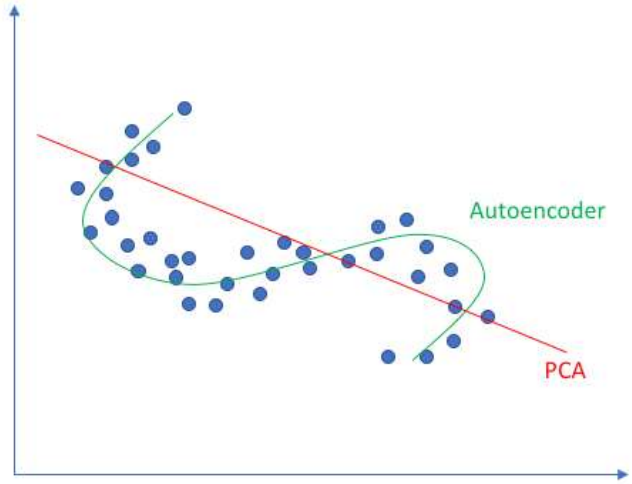
The simplest architecture for constructing an autoencoder is to constrain the number of nodes present in the hidden layer(s) of the network, limiting the amount of information that can flow through the network. By penalizing the network according to the reconstruction error, our model can learn the most important attributes of the input data and how to best reconstruct the original input from an "encoded" state. Ideally, this encoding will **learn and describe latent attributes of the input data**.



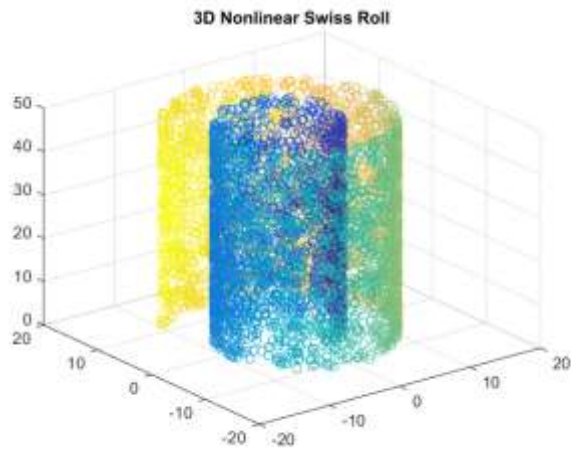
Because neural networks are capable of learning nonlinear relationships, this can be thought of as a more powerful (nonlinear) generalization of PCA. Whereas PCA attempts to discover a lower dimensional hyperplane which describes the original data, autoencoders are capable of learning nonlinear manifolds (a manifold is defined in *simple* terms as a continuous, non-intersecting surface). The difference between these two approaches is visualized below.

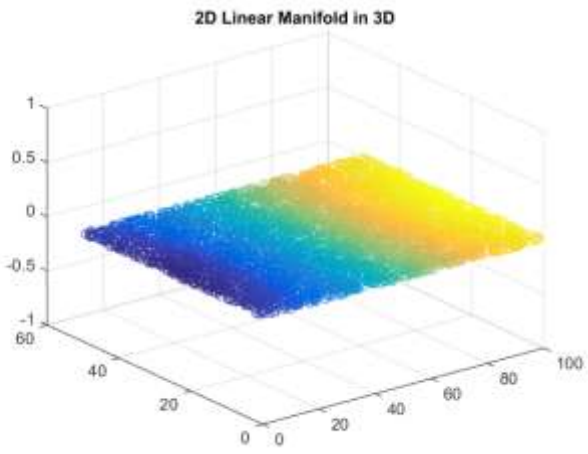
Module 3

Linear vs nonlinear dimensionality reduction



For higher dimensional data, autoencoders are capable of learning a complex representation of the data (manifold) which can be used to describe observations in a lower dimensionality and correspondingly decoded into the original input space.





An undercomplete autoencoder has no explicit regularization term - we simply train our model according to the reconstruction loss. Thus, our only way to ensure that the model isn't memorizing the input data is the ensure that we've sufficiently restricted the number of nodes in the hidden layer(s).

For deep autoencoders, we must also be aware of the *capacity* of our encoder and decoder models. Even if the "bottleneck layer" is only one hidden node, it's still possible for our model to memorize the training data provided that the encoder and decoder models have sufficient capability to learn some arbitrary function which can map the data to an index.

Module 3

Given the fact that we'd like our model to discover latent attributes within our data, it's important to ensure that the autoencoder model is not simply learning an efficient way to memorize the training data. Similar to supervised learning problems, we can employ various forms of regularization to the network in order to encourage good generalization properties; these techniques are discussed below.