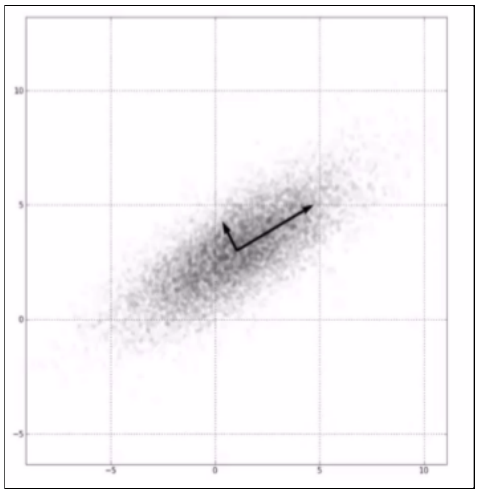
Principal component analysis

Usually, when people talk about dimensionality reduction, they're talking about a technique called principal component analysis. This is a much more-fancy technique, it gets into some pretty involved mathematics. But, at a high-level, all you need to know is that it takes a higher dimensional data space, and it finds planes within that data space and higher dimensions.

These higher dimensional planes are called hyper planes, and they are defined by things called eigenvectors. You take as many planes as you want dimensions in the end, project that data onto those hyperplanes, and those become the new axes in your lower dimensional data space:



You know, unless you're familiar with higher dimensional math and you've thought about it before, it's going to be hard to wrap your head around! But, at the end of the day, it means we're choosing planes in a higher dimensional space that still preserve the most variance in our data, and project the data onto those higher dimensional planes that we then bring into a lower dimensional space, okay?

You don't really have to understand all the math to use it; the important point is that it's a very principled way of reducing a dataset down to a lower dimensional space while still preserving the variance within it. We talked about image compression as

one application of this. So you know, if I want to reduce the dimensionality in an image, I could use PCA to boil it down to its essence.

Facial recognition is another example. So, if I have a dataset of faces, maybe each face represents a third dimension of 2D images, and I want to boil that down, SVD and principal component analysis can be a way to identify the features that really count in a face. So, it might end up focusing more on the eyes and the mouth, for example, those important features that are necessary for preserving the variance within that dataset. So, it can produce some very interesting and very useful results that just emerge naturally out of the data, which is kind of cool!

To make it real, we're going to use a simpler example, using what's called the Iris dataset. This is a dataset that's included with scikit-learn. It's used pretty commonly in examples, and here's the idea behind it: So, an Iris actually has 2 different kinds of petals on its flower. One's called a petal, which is the flower petals you're familiar with, and it also has something called a sepal, which is kind of this supportive lower set of petals on the flower.

We can take a bunch of different species of Iris, and measure the petal length and width, and the sepal length and width. So, together the length and width of the petal, and the length and width of the sepal are 4 different measurements that correspond to 4 different dimensions in our dataset. I want to use that to classify what species an Iris might belong to. Now, PCA will let us visualize this in 2 dimensions instead of 4, while still preserving the variance in that dataset. So, let's see how well that works and actually write some Python code to make PCA happen on the Iris dataset.

So, those were the concepts of dimensionality reduction, principal component analysis, and singular value decomposition. All big fancy words and yeah, it is kind of a fancy thing. You know, we're dealing with reducing higher dimensional spaces down to smaller dimensional spaces in a way that preserves their variance. Fortunately, scikit-learn makes this extremely easy to do, like 3 lines of code is all you need to actually apply PCA. So let's make that happen!

Support / Sign Out

PREV
Dimensionality reduction

NEXT
A PCA example with the Iris dataset