Nscript

Got It!

1. Visualizing the PCA transformation

In the next two chapters you'll learn techniques for dimension reduction.

2. Dimension reduction

Dimension reduction finds patterns in data, and uses these patterns to re-express it in a compressed form. This makes subsequent computation with the data much more efficient, and this can be a big deal in a world of big datasets. However, the most important function of dimension reduction is to reduce a dataset to its "bare bones", discarding noisy features that cause big problems for supervised learning tasks like regression and classification. In many real-world applications, it's dimension reduction that makes prediction possible.

3. Principal Component Analysis

In this chapter, you'll learn about the most fundamental of dimension reduction techniques. It's called "Principal Component Analysis", or "PCA" for short. PCA performs dimension reduction in two steps, and the first one, called "de-correlation", doesn't change the dimension of the data at all. It's this first step that we'll focus on in this video.

4. PCA aligns data with axes

In this first step, PCA rotates the samples so that they are aligned with the coordinate axes. In fact, it does more than this: PCA also shifts the samples so that they have mean zero. These scatter plots show the effect of PCA applied to two features of the wine dataset. Notice that no information is lost - this is true no matter how many features your dataset has. You'll practice visualizing this transformation in the exercises.

5. PCA follows the fit/transform pattern

scikit-learn has an implementation of PCA, and it has fit and transform methods just like StandardScaler. The fit method learns how to shift and how to rotate the samples, but doesn't actually change them. The transform method, on the other hand, applies the transformation that fit learned. In particular, the transform method can be applied to new, unseen samples.

6. Using scikit-learn PCA

Let's see PCA in action on the some features of the wine dataset. Firstly, import PCA. Now create a PCA object, and fit it to the samples. Then use the fit PCA object to transform the samples. This returns a new array of transformed samples.

7. PCA features

This new array has the same number of rows and columns as the original sample array. In particular, there is one row for each transformed sample. The columns of the new array correspond to "PCA features", just as the original features corresponded to columns of the original array.

8. PCA features are not correlated

It is often the case that the features of a dataset are correlated. This is the case with many of the features of the wine dataset, for instance. However, PCA, due to the rotation it performs, "de-correlates" the data, in the sense that the columns of the transformed array are not linearly correlated.

9. Pearson correlation

Linear correlation can be measured with the Pearson correlation. It takes values between -1 and 1, where larger values indicate a stronger correlation, and 0 indicates no linear correlation. Here are some examples of features with varying degrees of correlation.

10. Principal components

Finally, PCA is called "principal component analysis" because it learns the "principal components" of the data. These are the directions in which the samples vary the most, depicted here in red. It is the principal components that PCA aligns with the coordinate axes.

11. Principal components

After a PCA model has been fit, the principal components are available as the components attribute. This is numpy array with one row for each principal component.