PCA

Principal Component Analysis is a method of extracting important features (in the form of components) from a large set of features available in a dataset. PCA finds the directions of maximum variance in high-dimensional data and project it onto a smaller dimensional subspace while retaining most of the information. By projecting our data into a smaller space, we're reducing the dimensionality of our feature space. In simple words, Principal Component Analysis (PCA) is a statistical techniques used to reduce the dimensionality of the data (reduce the number of features in the dataset) by selecting the most important features that capture maximum information about the dataset. Original features of the dataset are converted to the Principal Components which are the linear combinations of the existing features. The feature that causes highest variance is the first Principal Component. The feature that is responsible for second highest variance is considered the second Principal Component and so on.

PCA helps to remove correlated features. Finding correlation manually in thousands of features is nearly impossible, frustrating and time consuming. After implementing the PCA on your dataset, all the Principal Components are independent of one another. There is no correlation among them.

PCA is a good way of speeding up a machine learning algorithm. If your learning algorithm is too slow because the input dimension is too high, then using PCA to speed it up can be a reasonable choice.

PCA improves model performance. Model would obviously perform better with less features having almost entire variance explained by the earlier huge dataset. Model would also take less time to get trained.

PCA reduces overfitting. Overfitting mainly occurs when there are too many variables in the dataset. So, PCA helps in overcoming the overfitting issue by reducing the number of features.

PCA leads to Information Loss. Although Principal Components try to cover maximum variance among the features in a dataset, if we don't select the number of Principal Components with care, it may miss some information as compared to the original list of features.

Independent variables become less interpretable. After implementing PCA on the dataset, your original features will turn into Principal Components. Principal Components are the linear combination of your original features. Principal Components are not as readable and interpretable as original features.

You need to scale the features in your data before applying PCA. For instance, if a feature set has data expressed in units of Kilograms, Light years, or Millions and if PCA is applied on such a feature set, the principal components will be biased towards features with high magnitude and would lead to false results. Use Standard Scaler to standardize the dataset features onto unit scale (mean = 0 and standard deviation = 1).

How it works

- Normalize the data

First step is to normalize the data that we have so that PCA works properly. This is done by subtracting the respective means from the numbers in the respective column. This produces a dataset whose mean is zero.

- Calculate the covariance matrix

Since the dataset we took is 2-dimensional, this will result in a 2*2 Covariance matrix.

$$\sum = \begin{bmatrix} Var(x) & Cov(x,y) \\ Cov(y,x) & Var(y) \end{bmatrix}$$

Please note (Cov(x,x)=Var(x)) and (Cov(x,y)=Cov(y,x))

What do the covariances that we have as entries of the matrix tell us about the correlations between the variables?

It's actually the sign of the covariance that matters

if positive, then the two variables increase or decrease together (correlated)

if negative, then One increases when the other decreases (Inversely correlated)

- Calculate the eigenvalues and eigenvectors

Next step is to calculate the eigenvalues and eigenvectors for the covariance matrix. λ is an eigenvalue for a matrix A if it is a solution of the characteristic equation

det(λI - A) = 0

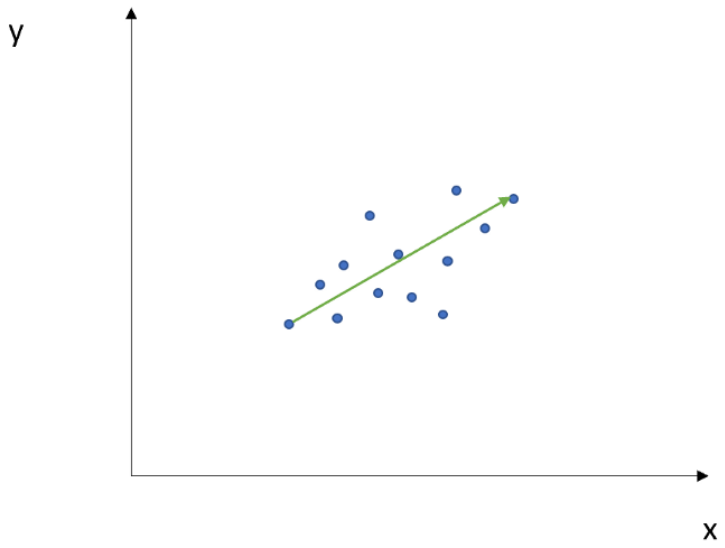
Where, I is the identity matrix of the same dimension as A which is a required condition for the matrix subtraction as well in this case and 'det' is the determinant of the matrix. For each eigenvalue λ, a corresponding eigen-vector v can be found by solving

(λI – A)v = 0

If we consider our example of two features (x and y), we will obtain the following:

$$\begin{bmatrix} Var(x) & Cov(x,y) \\ Cov(y,x) & Var(y) \end{bmatrix} \begin{bmatrix} \boldsymbol{v_1} \\ \boldsymbol{v_2} \end{bmatrix}' = \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{v_1} \\ \boldsymbol{v_2} \end{bmatrix}$$

Let's visualize them:



The direction in green is the eigenvector and it has a corresponding value called eigenvalue which describes its magnitude

- Choosing components and forming a feature vector

We order the eigenvalues from largest to smallest so that it gives us the components in order or significance. Here comes the dimensionality reduction part. If we have a dataset with n variables, then we have the corresponding n eigenvalues and eigenvectors. It turns out that the eigenvector corresponding to the highest eigenvalue is the principal component of the dataset and it is our call as to how many eigenvalues we choose to proceed our analysis with. To reduce the dimensions, we choose the first p eigenvalues and ignore the rest. We do lose out some information in the process, but if the eigenvalues are small, we do not lose much.

So, the feature vector is simply a matrix that has as columns the eigenvectors of the components that we decide to keep. This makes it the first step towards dimensionality reduction, because if we choose to keep only p eigenvectors (components) out of n, the final data set will have only p dimensions.

Continuing with the example from the previous step, we can either form a feature vector with both of the eigenvectors v1 and v2

$$\begin{bmatrix} 0.6778736 & -0.7351785 \\ 0.7351785 & 0.6778736 \end{bmatrix}$$

Or discard the eigenvector v2, which is the one of lesser significance, and form a feature vector with v1 only

$$\begin{bmatrix} 0.6778736 \\ 0.7351785 \end{bmatrix}$$

Discarding the eigenvector v2 will reduce dimensionality by 1, and will consequently cause a loss of information in the final data set. But given that v2 was carrying only 4% of the information, the loss will be therefore not important and we will still have 96% of the information that is carried by v1.

- Forming Principal Components

This is the final step where we actually form the principal components using all the math we did till here. For the same, we take the transpose of the feature vector and left-multiply it with the transpose of scaled version of original dataset.

NewData = ScaledData * Feature Vector

NewData is the Matrix consisting of the principal components

ScaledData is the scaled version of original dataset

FeatureVector is the matrix we formed using the eigenvectors we chose to keep