# Introduction

**Principal Component Analysis** (PCA) is a dimensionality reduction algorithm. As the name says, PCA reduces the dimensions of the data into a lower dimension while trying to preserve as much variance as possible. This is done by minimizing the squared projection errors (or squared distances) between the data and the projected **principal components** (PC). But to make the computations more efficient while still achieving the same objective, PCA usually prefers to minimize the sum of squared distances between the projected points (projected onto the PC) and the origin.

The number of principal components is generally less than or equal to the number of features, or number of samples, whichever lower. The principal components are also ranked in descending order, in which the first PC would retain the most amount of variance from the original data.

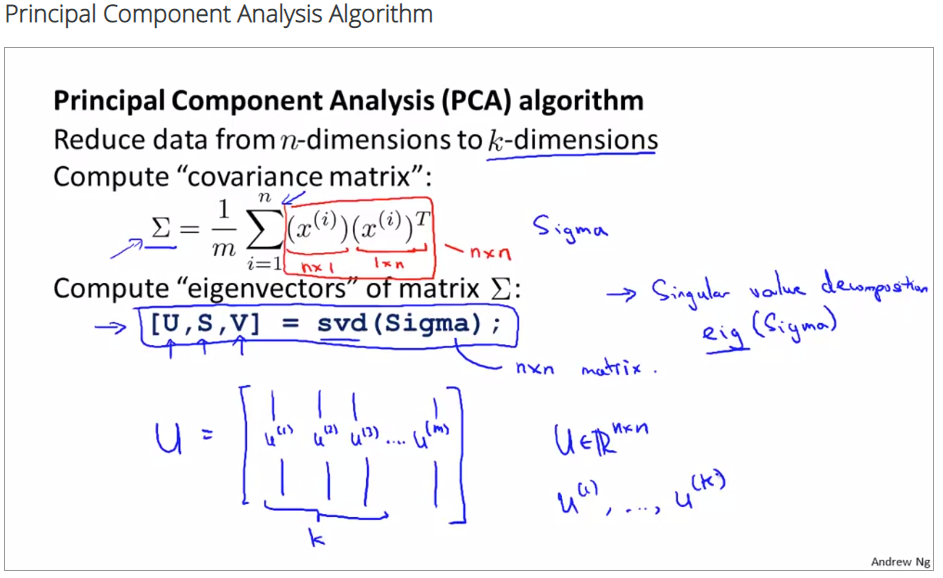
# Objective

Find a transformation such that:

* The transformed features are linearly independent
* The newly found dimensions (or hyperplanes) should minimize the squared projection error
* They should also be ranked by the amount of variance preserved.

# Algorithm procedures

The procedures can essentially be summarized in the figure below.



## Covariance Matrix

Where *m* = number of samples.

## Eigenvectors and eigenvalues

The ***eigenvectors*** are the unit vectors that point in the direction of each PC. Also refer images below.

The ***eigenvalue*** for a PC is the sum of squared projection error (sum of squared distances) between the PC and the data points. But to make the computations more efficient while still achieving the same result, the ***eigenvalues*** are usually computed using the sum of squared distances between the projected points (projected onto the PC) and the origin, like what explained in the introduction section above.

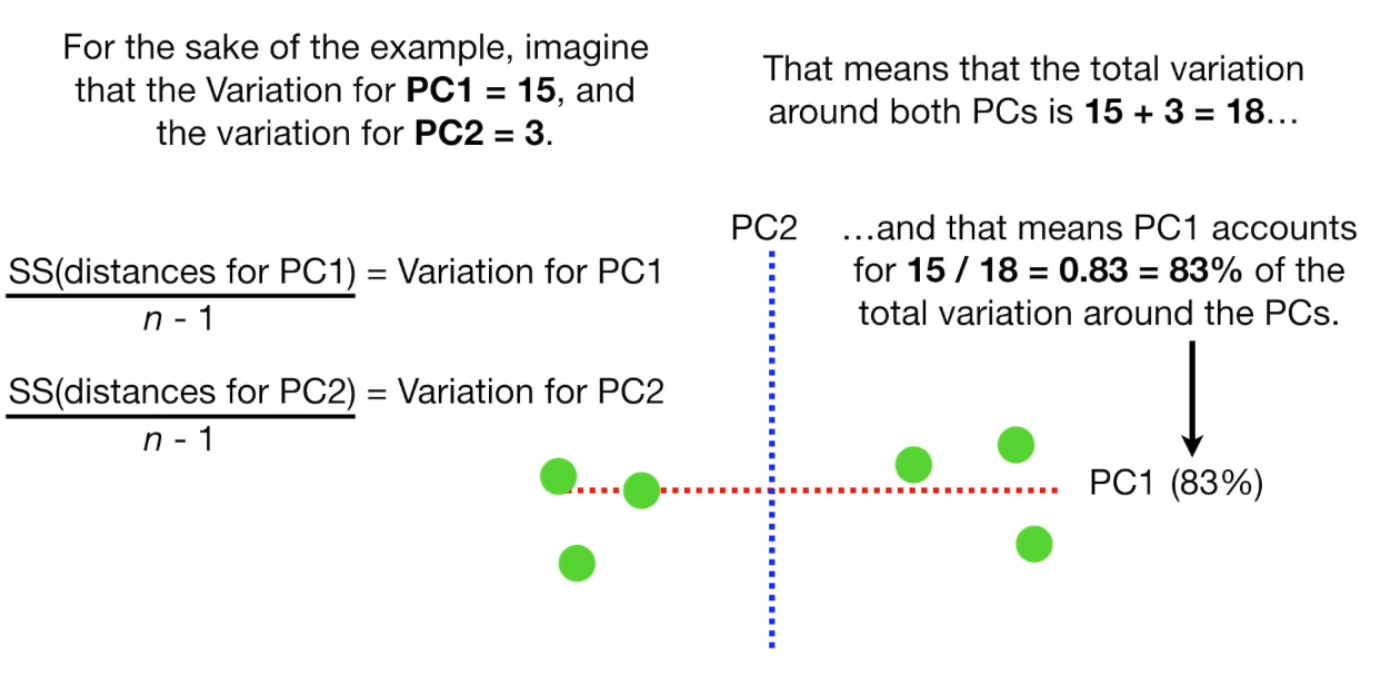
## Variance

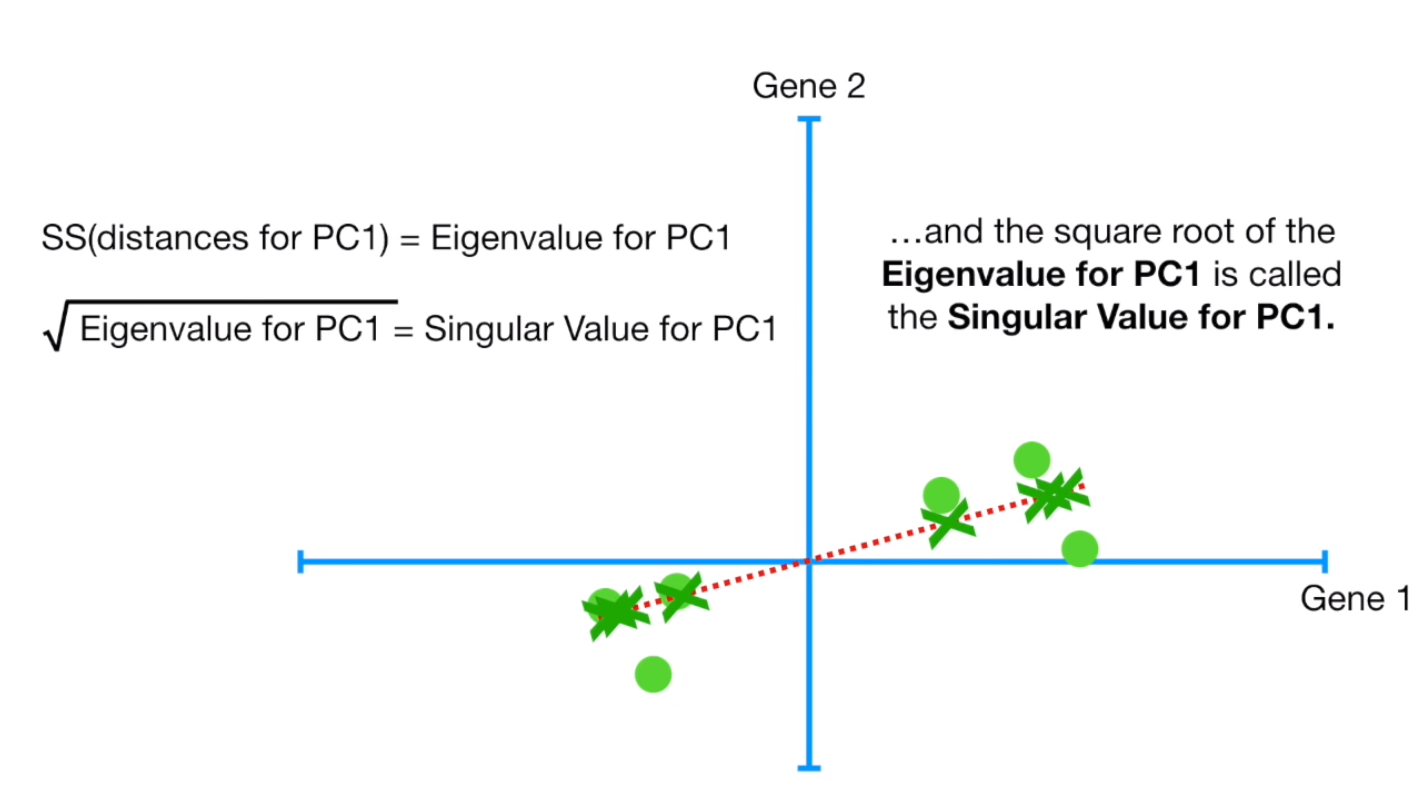
In Scikit-learn PCA, it is the attribute -- . We calculate them by:

## Detailed steps

1. Subtract the mean from X (mean centering), and optionally scale by standard deviation
2. Calculate the covariance matrix
3. Calculate the eigenvectors and eigenvalues of the covariance matrix
4. Sort the eigenvectors by their eigenvalues in descending order. NOTE: steps 2-4 are known as the *Singular Vector Decompositi*on (SVD) technique, can be simplified using .
5. Choose first eigenvectors to be our PCs.
6. Transform data: Project the dataset onto the hyperplane defined by the first *d* PCs (by using dot product between the data and the eigenvectors).

[YouTube link](https://youtu.be/FgakZw6K1QQ?list=PLblh5JKOoLUICTaGLRoHQDuF_7q2GfuJF) for the StatQuest video.





[Coursera link](https://www.coursera.org/learn/machine-learning/lecture/S1bq1/choosing-the-number-of-principal-components).

