Published in The Andela Way · Follow

Patrick Luboobi
Feb 5, 2018 · 6 min read

Foundations of Machine Learning : Singular Value Decomposition (SVD)

Linear Algebra is fundamental in many areas of Machine learning and one of the most important concepts is; Singular Value Decomposition(SVD). The motivation element behind this article is to get Software Engineers to ameliorate their basic understanding of SVD, and its real-world application.

Singular Value Decomposition(SVD) is one of the most widely used Unsupervised learning algorithms, that is at the center of many recommendation and Dimensionality reduction systems that are the core of global companies such as Google, Netflix, Facebook, Youtube, and others.

Specifically, for this article, we shall be looking at a movie recommendation system. But before that, let's see how SVD works.

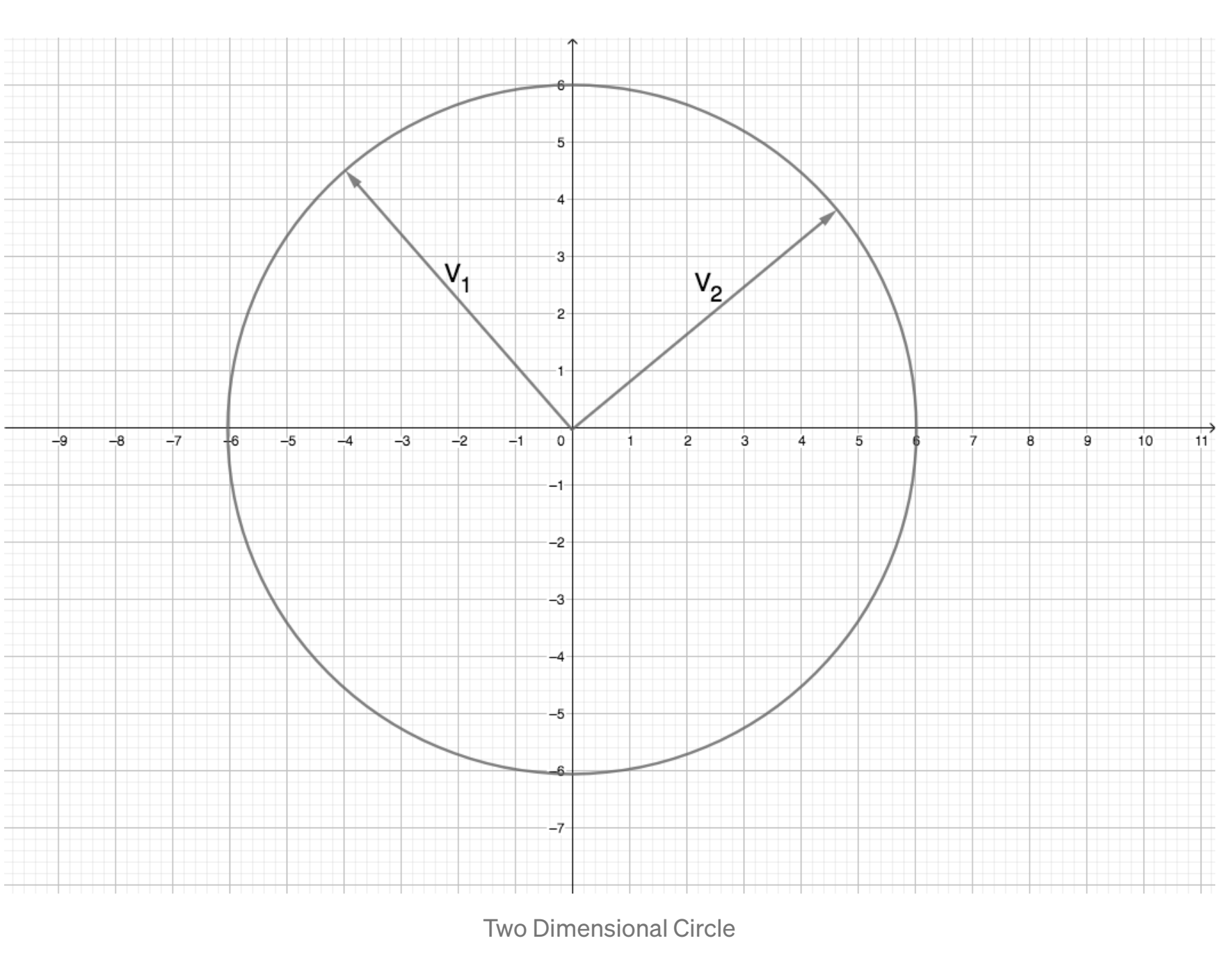
In simple terms, SVD is the factorization of a matrix into 3 matrices. So if we have a matrix **A**, then its SVD is represented by:

$$A = U\Sigma V^T$$

> *Where **A** is an m x n matrix, **U** is an (m x m) orthogonal matrix, **Σ** is an (m x n) nonnegative rectangular diagonal matrix, and **V** is an (n x n) orthogonal matrix.*
>
> ***U** is also referred to as the left singular vectors, **Σ** the singular values, and **V** the right singular vectors*

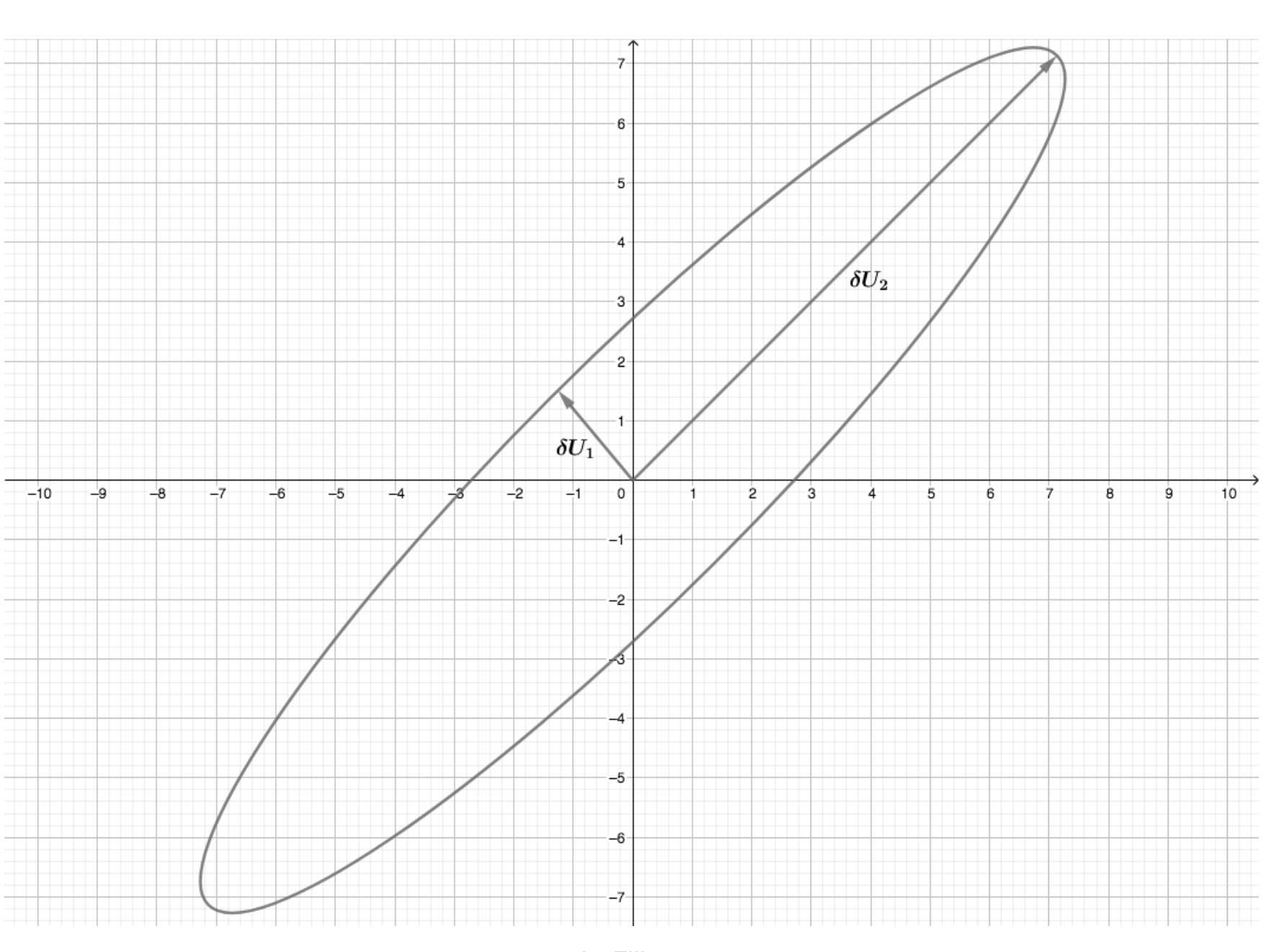
So first let's see how this comes about and then we'll look at an example:

Imagine a circle in two dimensions represented by vectors `V1` and `V2` undergoing a matrix transformation as illustrated on the cartesian coordinates below:



Two Dimensional Circle

After Matrix Multiplication



An Ellipse

From the images above, you can tell that when a matrix multiples a vector, it simply stretches it and then rotates it.

So if we generalize this from just two dimensions to n-dimensions, the vector space:

$$V_1, V_2, ...V_n \quad becomes \quad U_1, U_2, ...U_n$$

after the multiplication, and we have:

$$\sigma_1, \sigma_2, ...\sigma_n$$

representing the space of the individual stretching factors.

Therefore from this we can write the equation:

$$AV_n = \sigma_n U_n$$

which we can write more generally as:

$$AV = U\Sigma$$

Where **Σ** represents the space of all stretching factors(σ's).

But for an orthogonal matrix,

$$V^{-1} = V^T$$

Also, note that the product of a matrix and its inverse is the **identity matrix** (An identity matrix is a diagonal matrix with only 1's). This concept can be represented by the equation below:

$$VV^{-1} = 1$$

Combining the above three equations leads us to the **Reduced Singular Value Decomposition**.

$$A = U\Sigma V^T$$

Where **V** is a rotation, **Σ** a stretching and **U** another rotation. Also, the entries of **U** are the principle axis while **Σ** are the singular values.

So this is how you can decompose a matrix into three lower rank matrices.

Let's look at a classical application of this. Imagine that we have a matrix A whose columns represent movies and the rows different users. The entries of the matrix are numbers 0 to 5 where 0 means a user does not like a certain movie and 5 means they really like a given movie as illustrated below:

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}$$

Now imagine that the first 3 columns are the movies Avengers, StarWars and IronMan respectively(Sci-Fi movies). While the last 2 columns are the movies Titanic and Notebook (Romance movies).

After performing SVD on matrix A we get the matrices UΣV as illustrated below(using a tool like or sklearn):

$$U = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & -0.32 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix}$$

$$V^T = \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

Let's take a closer look at these three matrices starting with U:

So the first column of U represents weights that would match each user's preference to movies categorized under Sci-Fi while the second column of U represents weights that would match each user's preference to movies under the romance category. For example, the first user greatly prefers sci-fi movies(0.13 score) compared to romance(0.02 score). As for the third column, we won't consider it for now.

And for Σ,

The first diagonal entry represents the weight of the Sci-Fi category and the second diagonal entry represents the weight of the romance category.

And for V,

The columns depict the degree to which a movie belongs to a category. So, for example, we can see from the first column of **V** that the first movie(this would be Avengers) belongs heavily to the Sci-Fi(0.56 score) category and very little to the romance category(0.12 score).

> ***Note:** we have not considered the third dimension of each matrix at all. Well, this is because when you look at matrix **Σ**, the third diagonal entry which represents the weight of a movie category has a small value(1.3 score). This is understandable because we only have two categories of movies. So most of the third dimension is considered as noise.*

So its the above note that we use to perform dimensionality reduction to the matrices **A**. We do this by eliminating the third dimension of **Σ**, this would also mean eliminating the third column of **U** and the third row of **V** to produce the following new **U, Σ** and, **V**:

$$U = \begin{bmatrix} 0.13 & 0.02 \\ 0.41 & 0.07 \\ 0.55 & 0.09 \\ 0.68 & 0.11 \\ 0.15 & -0.59 \\ 0.07 & -0.73 \\ 0.07 & -0.29 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}$$

$$V^T = \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & -0.12 & -0.69 & -0.69 \end{bmatrix}$$

So as you can see, the final matrices UΣV are smaller than the initial ones since we have eliminated the third dimension.

To confirm that eliminating the given rows and columns as we have done only affects the initial matrix A to a small extent. Let's multiply the above three matrices to get matrix B below:

$$B = \begin{bmatrix} 0.92 & 0.95 & 0.92 & 0.01 & 0.01 \\ 2.91 & 3.01 & 2.91 & -0.01 & -0.01 \\ 3.90 & 4.04 & 3.90 & 0.01 & 0.01 \\ 4.82 & 5.00 & 4.82 & 0.03 & 0.03 \\ 0.70 & 0.53 & 0.70 & 4.11 & 4.11 \\ -0.69 & 1.34 & -0.69 & 4.78 & 4.78 \\ 0.32 & 0.23 & 0.32 & 2.01 & 2.01 \end{bmatrix}$$

So Let's compare this matrix B with the original matrix **A** below:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}$$

Just by looking at the above two matrices, you can tell that the difference between their elements is very small, in other words, the product of our final three matrices **B**(after SVD) ≈ **A**(before SVD):

Or mathematically this can be represented as the Frobenius norm. Which is the square root of the summation of the squares of the differences between the individual matrix entries. It can be represented by the equation below:

$$||A - B||_F = \sqrt{\sum_{ij}(A_{ij} - B_{ij})^2}$$

So this is how we are able to decompose a matrix into lower rank matrices without losing much of the important data.

It also helps to analyse and acquire important information concerning the matrix data.

There are many other applications of SVD other than the ones talked about in this article. Some of the others include data compression, solving the pseudo-inverse and search engines like Google use SVD to compute approximations of enormous matrices that provide compression ratios of millions to one. So searching for a term is much quicker.

So hopefully this reading can give you a clear picture of this fundamental linear algebra concept and its application in Machine Learning.

756 5

Get started Sign In
Search
Patrick Luboobi
38 Followers
Follow
Related
Introduction To Machine Learning Pipeline
Prerequisite: Basic kno...
A Beginner's Guide to Activation Functions
In an artificial neural net...
Naïve Bayes Classifier
A Mathematical Explanation of Naiv...
Algo Learning: Linear Regression
Series Topic 1: A mathe...
Help Status Writers Blog Careers Privacy Terms About Knowable