Author :-Rahul Nyati

Singular Value Decomposition (SVD)

IN machine learning Singular value decomposition is generally used for Dimension Reduction Techniques, image compression, and denoising data. In essence, SVD states that a matrix can be represented as the product of three other matrices.

Singular Value Decomposition (SVD), a method from straight algebra that has been widely used as a method of reducing size in machine learning. SVD is a matrix factorization process, which reduces the number of data elements by reducing the size of the space from N-dimension to K-dimension (where K <N). In the context of the recommendation process, SVD is used as a participatory filtering method. It uses a matrix structure where each row represents a user, and each column represents an object. The properties of this matrix are the dimensions given by the users.

The factorisation of this matrix is done by the singular value decomposition. It finds factors of matrices from the factorisation of a high-level (user-item-rating) matrix. The singular value decomposition is a method of decomposing a matrix into three other matrices as given below:

$$A = USV^T$$

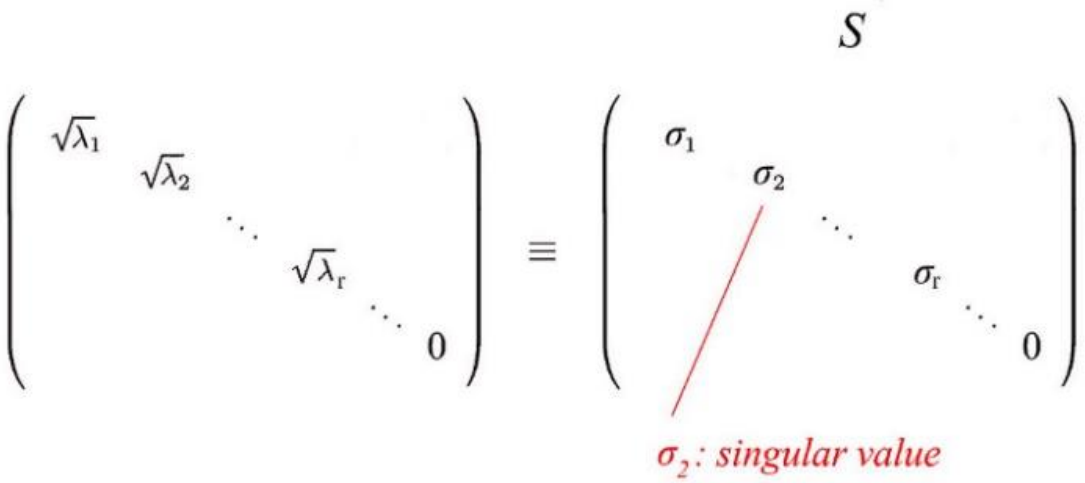
Where A is a m x n utility matrix, U is a m x r orthogonal left singular matrix, which represents the relationship between users and latent factors, S is a r x r diagonal matrix, which describes the strength of each latent factor and V is a r x n diagonal right singular matrix, which indicates the similarity between items and latent factors. The

latent factors here are the characteristics of the items, for example, the genre of the music. The SVD decreases the dimension of the utility matrix A by extracting its latent factors. It maps each user and each item into a r-dimensional latent space. This mapping facilitates a clear representation of relationships between users and items.

Let's start with the hard part first. SVD states that **any** matrix A can be factorized as:

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

where U and V are orthogonal matrices with orthonormal eigenvectors chosen from $\boldsymbol{AA^T}$ and $\boldsymbol{A^TA}$ respectively. $\boldsymbol{S}$ is a diagonal matrix with r elements equal to the root of the positive eigenvalues of AA^T or A^TA (both matrics have the same positive eigenvalues anyway). The diagonal elements are composed of singular values.



$$A_{m\times n} = U_{m\times m} S_{m\times n} V^{T}_{n\times n}$$

$$\underset{m\times n}{A} = \begin{pmatrix} x_{11} & x_{12} & & x_{1n} \\ & & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix} = \underset{m\times m}{U} \begin{pmatrix} u_{11} & & u_{m1} \\ & \ddots & \\ u_{1m} & & u_{mm} \end{pmatrix} \underset{m\times n}{S} \begin{pmatrix} \sigma_1 & \ddots & 0 \\ & \sigma_r & \ddots \\ 0 & & 0 \end{pmatrix} \underset{n\times n}{V^{T}} \begin{pmatrix} v_{11} & & v_{1n} \\ & \ddots & \\ v_{n1} & & v_{nn} \end{pmatrix}$$

We can arrange eigenvectors in different orders to produce U and V. To standardize the solution, we order the eigenvectors such that vectors with higher eigenvalues come before those with smaller values.

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m$$

$$\begin{pmatrix} | & & | \\ u_1 & \ldots & u_m \\ | & & | \end{pmatrix}$$

Comparing to eigendecomposition, SVD works on non-square matrices. U and V are invertible for any matrix in SVD and they are orthonormal which we love it.

Suppose we had a matrix $\boldsymbol{A}$.

$$A = \begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix}$$

$$AA^T = \begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 2 & 1 \end{pmatrix} = \begin{pmatrix} 8 & 4 \\ 4 & 2 \end{pmatrix} = W$$

Recall how the definition for eigenvectors and eigenvalues is:

$$Wx = \lambda x$$

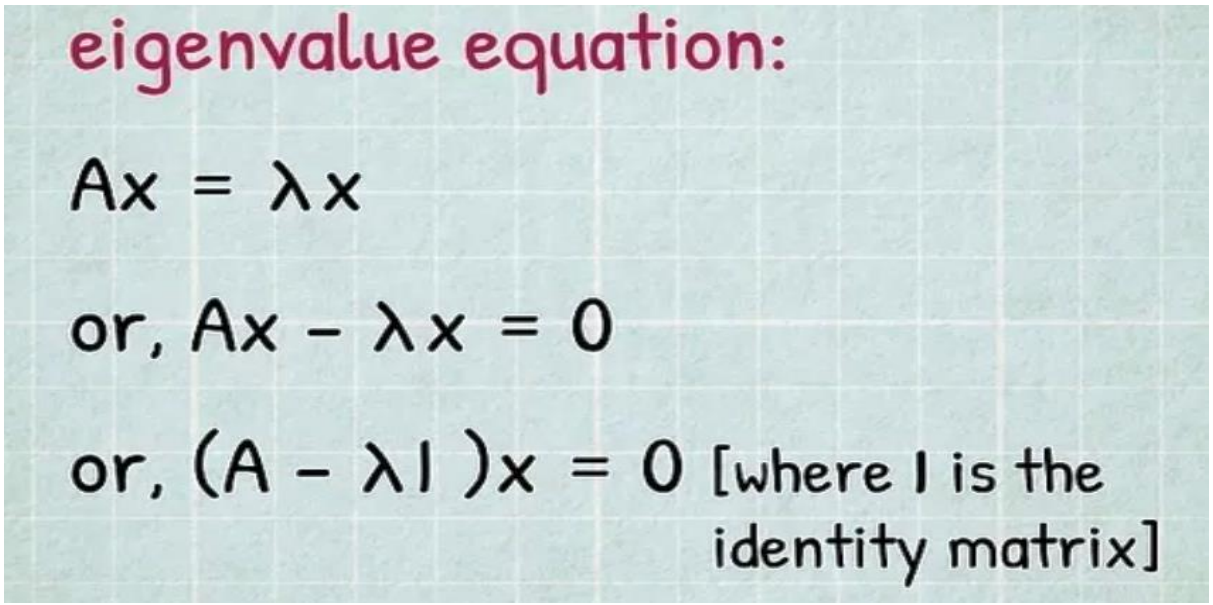
The latter can also be expressed as:

$$Wx = (W - \lambda I)x$$

How can we compute an SVD of a matrix A ?

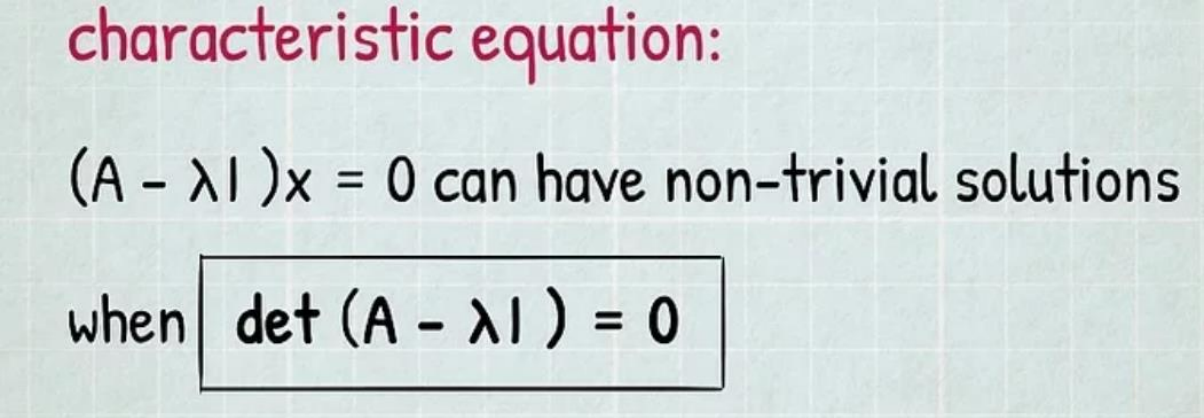
1. Evaluate the n eigenvectors V_i and eigenvalues λ_i of AA^T

1 **Understand determinants.** The determinant of a matrix $\det A = 0$ when A is non-invertible. When this occurs, the null space of A becomes non-trivial - in other words, there are non-zero vectors that satisfy the homogeneous equation $\mathbf{Ax} = 0$.[1]



2 **Write out the eigenvalue equation.** As mentioned in the introduction, the action of A on $\mathbf{x}$ is simple, and the result only differs by a multiplicative constant λ, called the eigenvalue. Vectors that are associated with that eigenvalue are called eigenvectors.[2]

- $A\mathbf{x} = \lambda\mathbf{x}$
- We can set the equation to zero, and obtain the homogeneous equation. Below, I is the identity matrix.
- $(A - \lambda I)\mathbf{x} = 0$



3 **Set up the characteristic equation.** In order for $(A - \lambda I)\mathbf{x} = 0$ to have non-trivial solutions, the null space of $A - \lambda I$ must be non-trivial as well.

- The only way this can happen is if $\det(A - \lambda I) = 0$. This is the characteristic equation.

4 **Obtain the characteristic polynomial.** $\det(A - \lambda I)$ yields a polynomial of degree n for $n \times n$ matrices.

$$\det \begin{vmatrix} 8-\lambda & 4 \\ 4 & 2-\lambda \end{vmatrix} = 0 \quad \text{or,} \quad (8-\lambda)(2-\lambda)-4*4=0$$

eigen values are λ=10 or 0

Now we calculate the eigen vectors

λ=10

$$(AA^T - 10)x=0 = \begin{pmatrix} -2 & 4 \\ 4 & 8 \end{pmatrix} \begin{pmatrix} X1 \\ X2 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$X_1=2X_2$

$[\ 2 \quad 1\ ]^T$

λ=0

$$(AA^T)x=0= \begin{pmatrix} 8 & 4 \\ 4 & 2 \end{pmatrix} \begin{pmatrix} X1 \\ X2 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$2x_1=-x_2$

$[1 \quad -2]^T$

Now we calculate the norm of vector

$\sqrt{(1^2+(-2)^2)}$ or $\sqrt{(2^2+1^2)}$

Which is $\sqrt{5}$

Now our U matrix is

$$U=1/\sqrt{5} \begin{pmatrix} 2 & 1 \\ 1 & -2 \end{pmatrix}$$

Now we calculate the eigen vector from A^TA which gives the V matrix

$$A^TA = \begin{pmatrix} 2 & 1 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 5 & 5 \\ 5 & 5 \end{pmatrix}$$

$$\text{Det}\begin{vmatrix} 5-\lambda & 5 \\ 5 & 5-\lambda \end{vmatrix} = 0 \quad \text{or,} \quad (5-\lambda)(5-\lambda)-25=0$$

Eigen values are $\lambda=10$ or 0

Now we calculate the eigen vectors

$\lambda=10$

$$(AA^T - 10)x=0 = \begin{pmatrix} -5 & 5 \\ 5 & -5 \end{pmatrix} \begin{pmatrix} X1 \\ X2 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$X1=X2$

$[\ 1 \quad 1\ ]^T$

$\lambda=0$

$$(AA^T)x=0 = \begin{pmatrix} 5 & 5 \\ 5 & 5 \end{pmatrix} \begin{pmatrix} X1 \\ X2 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$X1=-X2$

$[\ 1 \quad -1]^T$

Now we calculate the norm of vector

$\sqrt{(1^2+(-1)^2)}$ or $\sqrt{(1^2+1^2)}$

Which is $\sqrt{2}$

Now our V matrix is

$$V=1/\sqrt{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$S= \begin{pmatrix} \sqrt{10} & 0 \\ 0 & 0 \end{pmatrix}$$

Now we can Write the A in the form of

$$A = U\,S\,V^T = \begin{pmatrix} 2/\sqrt{5} & 1/\sqrt{5} \\ 1/\sqrt{5} & -2/\sqrt{5} \end{pmatrix} \begin{pmatrix} \sqrt{10} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -2/\sqrt{2} \end{pmatrix}$$

Applications of SVD

According to me SVD is the most fascinating factorization!

It is extremely powerful! There are a lot of applications:

1. **Principal Component Analysis (PCA)**, explained by the other answers
2. **Image compression**. Thanks to SVD we can compress images, which are matrices, calculating them k-rank approximation
3. **Least squares**. You can approximate the solution of overdetermined systems using SVD both in the full and deficient rank case
4. **Dictionaries for sparse representations of signals**.
5. **Latent Semantic Analysis (LSA)**
6. denoising data.
7. ...and so on!

Dimensionality reduction is a process for reducing the number of features to be used in an analysis or modeling exercise.

Intuitively, SVD is a procedure that allows one to (in a sense) "rotate the axes" of the feature space in which a dataset has been plotted so that instead of the different axes corresponding to specific features (e.g. age, income, gender, etc. of a customer)

the new axes point in directions that track linear combinations of the original feature space. These “new axes” are useful because they systematically break down the variance in the data points (how widely the data points are distributed) based on each directions contribution to the variance in the data:

The result of this process is a ranked list of "directions" in the feature space ordered from most variance to least. The directions along which there is greatest variance are referred to as the "principal components" (of variation in the data) and the common wisdom is that by focusing on the way the data is distributed along these dimensions exclusively, one can capture most of the information represented in in the original feature space without having to deal with such a high number of dimensions (but see below on the difference between feature selection and dimensionality reduction).

What are the benefits of dimensionality reduction tools like SVD?

There are a number benefits to reducing the dimension of a feature space when working with data:

First, by reducing the dimensionality of the features per data point, a reduction in computational cost can be achieved. In many data sets, most of the variance can be explained by a relatively small number of principal components, when compared to the dimensionality of the entire feature set. In such cases the computational cost per data point may be reduced by many orders of magnitude with a procedure like SVD without

ostensibly (but see below) losing much of the information stored by looking at the data in the unreduced feature space.

Second, such a rotation in data can eliminate colinearity in the original feature spaces. For instance, if two features in a data set tend to vary in the same (or inverse) way for most of the data points (consider square footage and room count of a house) then (depending on the data set and use case) including these "collinear" features can have a negative impact if one were to use both of them in building a classifier. By reducing the number of features with SVD, this such collinearity may be eliminated by attending to a single axis (e.g an " $\beta 1|SqFootage>+\beta 2|RoomCt>$ " axis).

Third, it reduces the impact of a phenomenon called the "curse of dimensionality" (What is the curse of dimensionality?). In short, as we increase the dimension of a feature space, the number of data points needed to "fill in" that space with the same density explodes (exponentially). Because of this many machine learning algorithms will falter if the dimensionality is too high (relative to the number of available data points). By performing SVD to reduce the the number of dimensions (without ostensibly losing much of the variance in the data set) the effects of this "curse of dimensionality" can be mitigated.

Is dimensionality reduction the same as feature selection?

No, but this is a common misconception...

While feature selection and dimensionality reduction both have the effect of feeding a lower number of features to “get an answer from one’s model,” feature selection techniques (like regularization) are an exercise in finding (and selecting) those features that are significant when it comes to signaling a given target variable. Dimensionality reduction on the other hand, blindly reduces the count of features (dimensions) that are being used, without paying attention to their effect in predicting a given target variable. So, it is entirely possible, that in some circumstances one might use SVD to identify and drop information captured by certain lower variance features that could have been none the less very significant when it came to classifying a certain target variable.

One (coarse grained) way of thinking about the difference between dimensionality reduction and feature selection is along the lines of the difference between unsupervised and supervised learning models. Often one can use unsupervised learning techniques to find “natural patterns” (e.g. clusters) in the data, but if the goal is to find patterns that signal a particular target, it is not guaranteed that these “natural patterns” will directly correspond to signaling the given target.

Are there other use cases for SVD than dimensionality reduction?

Yes. One of the most prominent in the context of Machine Learning and Data Science is found in the case of recommendation engines. SVD is a popular “Matrix

Factorization" technique in Collaborative Filtering recommender engines.

When using SVD for Collaborative Filtering, the idea is that one has a data matrix M relating each User (rows) with potential Items (columns) that could be recommended. The matrix M (often referred to as the utility matrix) can be filled in either with explicit feedback (stars) or implicit feedback (e.g. purchase counts). What happens in the SVD Collaborative Filtering case is that the non-missing entries that are in M (e.g. the movies a user has rated) are used to "factor" M into three component matrices:

1-U -- A matrix relating the users to detected latent features.

2-V -- A matrix relating the items to these same latent features.

3-S -- A diagonal matrix giving the relative significance of these latent features.

These matrices can then be recombined through multiplication to "fill in" the missing values in the original utility matrix $M=USV^T$.

Note this is very different from the case of dimensionality reduction: In the recommendation case, the goal is to "fill in" the missing entries in the data matrix M . In the case of dimensionality reduction, there is no presumption of missing data. In both cases, the goal is achieved by finding a way of factoring the matrix into component matrices, but the methodology in the case of dimensionality reduction is fundamental linear algebra, whereas in the recommendation

case, a number of techniques (e.g. alternating least squares, and others) can be used to leverage the non-missing data in order to generate ostensible factors of the utility matrix.