

Introduction to Data Science

**DATA UNDERSTANDING:
GLOBAL STRUCTURE**

BRIAN D'ALESSANDRO

Fine Print: these slides are, and always will be a work in progress. The material presented herein is original, inspired, or borrowed from others' work. Where possible, attribution and acknowledgement will be made to content's original source. Do not distribute, except for as needed as a pedagogical tool in the subject of Data Science.

TYPES OF STRUCTURE

Univariate – does a single variable follow a predictable pattern?

Pairwise (i.e., between 2 variables)

Global (i.e., across a matrix, or within a set of variables)

Supervised (technically includes the above, but with special emphasis on a single target variable)



DATA IS MULTIVARIATE

We usually want to go beyond pairwise similarity and understand how much “information” is actually embedded in a matrix.

We also might want to know what groups of features are related.

In an $N \times 2$ matrix, this problem reduces to the pairwise methods just discussed.

SINGULAR VALUE DECOMPOSITION

Although this is not a linear algebra course, this equation happens so often in data analysis that it is worth learning (again).

$$\begin{matrix} X & = & U & \Sigma & V^T \\ N \times M & & N \times M & M \times M & (M \times M)^T \end{matrix}$$

We will not dive into all of the theoretical aspects of SVD and related topics. Our goal here is to understand it intuitively and be able to use it as a tool for solving other common Data Science problems.

SINGULAR VALUE DECOMPOSITION

Lets go through this piece by piece:

U

U holds the left singular vectors. Each row contains k elements that correspond to the latent factors of each row of X. U is orthonormal.

Σ

Σ is a diagonal matrix that holds the singular values (in descending order). The singular values are essentially weights that determine how much that latent factor contributes to the matrix.

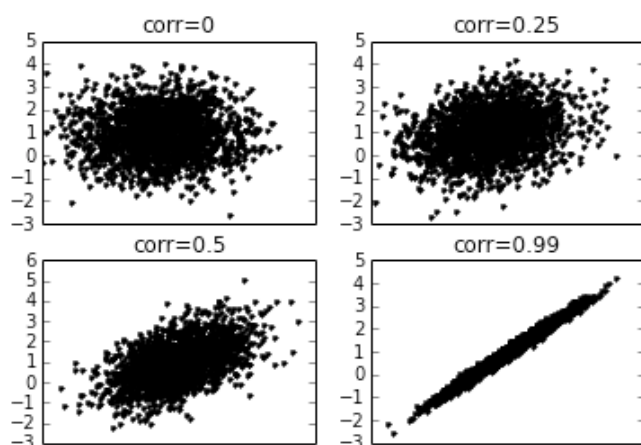
V^T

V holds the right singular vectors. Each row contains k elements that correspond to the latent factors of each column of X. V is orthonormal.

SINGULAR VALUE DECOMPOSITION

How does this relate to structure?

Lets recall these scatter plots. Each plot can be expressed as an $N \times 2$ Matrix. The SVD is a tool that can help us understand how much information or structure is actually in the matrix.

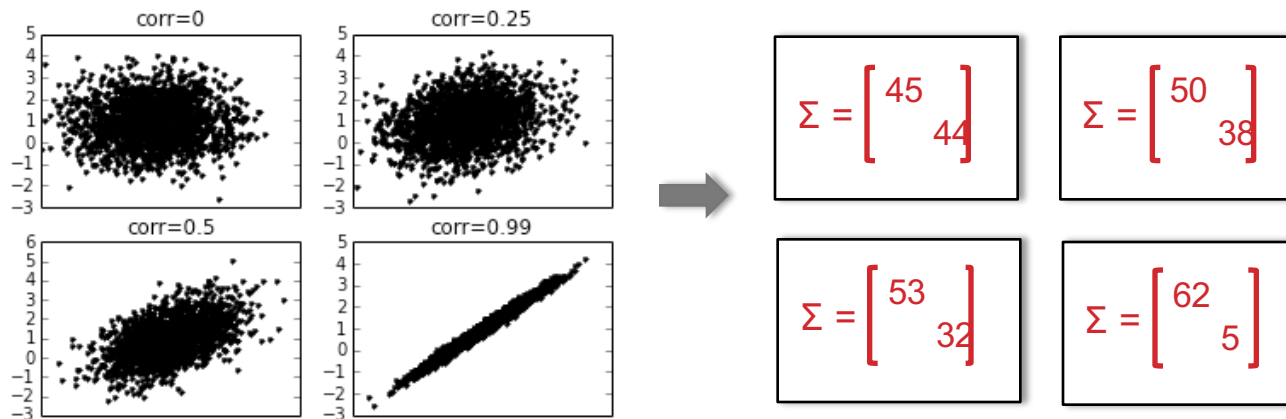


More independence of columns = more information or degrees of freedom.

Less independence of columns = less information or degrees of freedom.

SINGULAR VALUE DECOMPOSITION

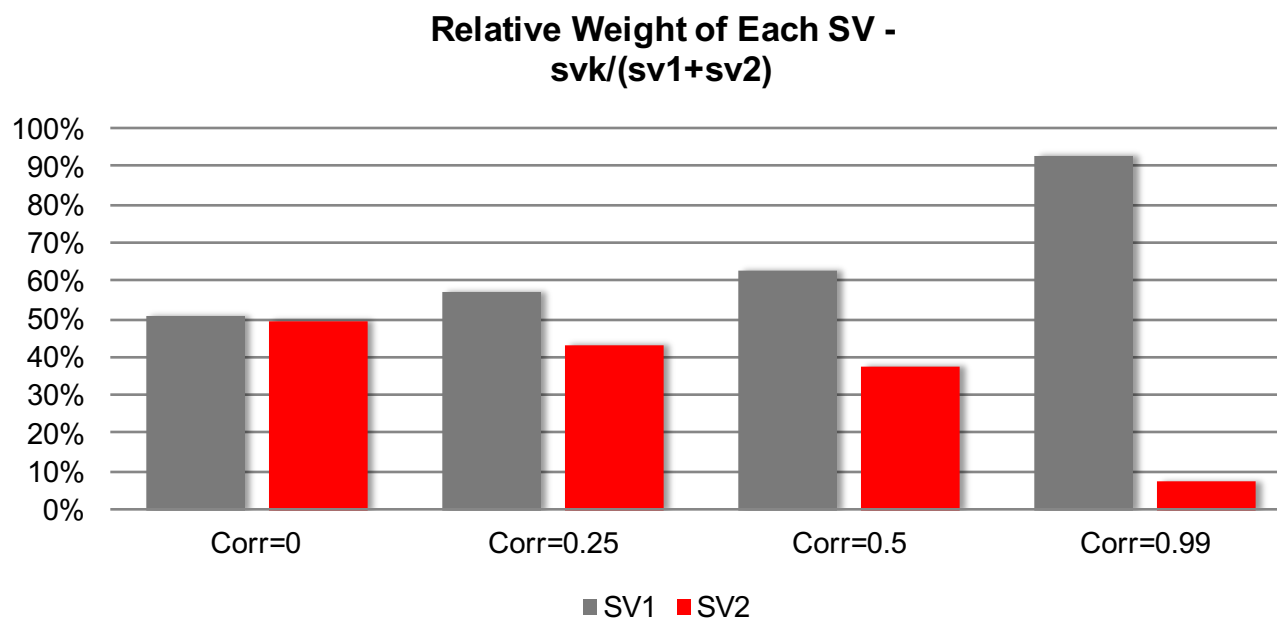
Using Scipy we can easily compute the SVD.:
`U,Sig,Vt = scipy.linalg.svd(matrix)`



The magnitude of the singular-values are generally determined by the magnitude of the values in X. The relative difference between singular values is a function of the level of independence of the columns.

SINGULAR VALUE DECOMPOSITION

We can see that less independence of the columns leads to singular values with more skew from each other.

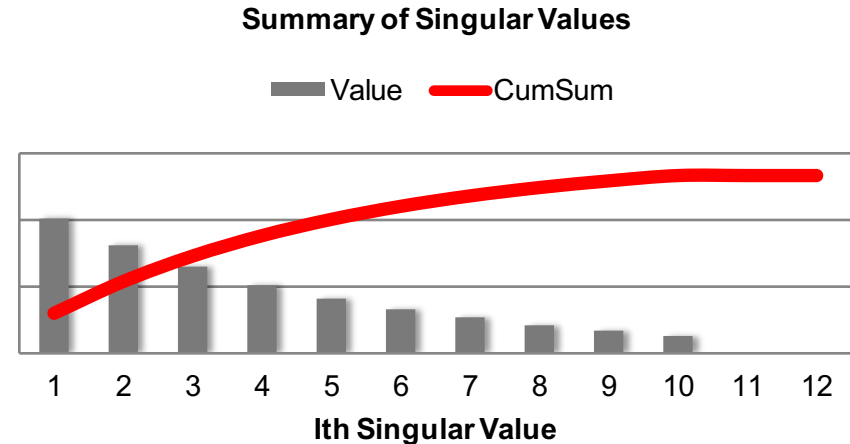
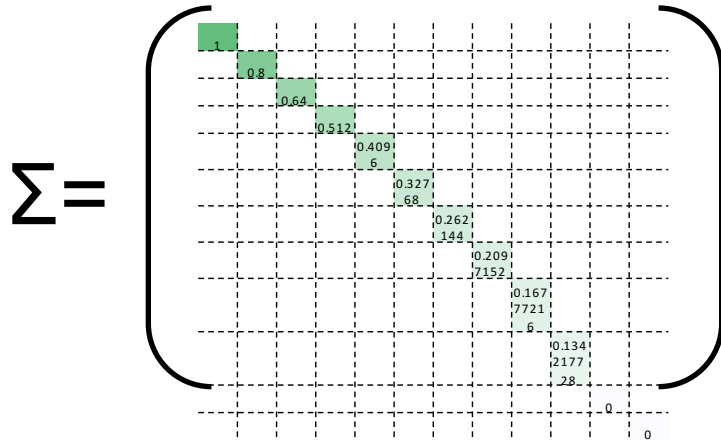


Copyright: Brian d'Alessandro, all rights reserved

SINGULAR VALUE DECOMPOSITION

This idea generalizes to more dimensions, which is where the SVD is extremely useful.

The skew of the singular values, and the shape of the cumulative sum curve can give us a sense of the degree of independence in a multi-dimensional matrix.



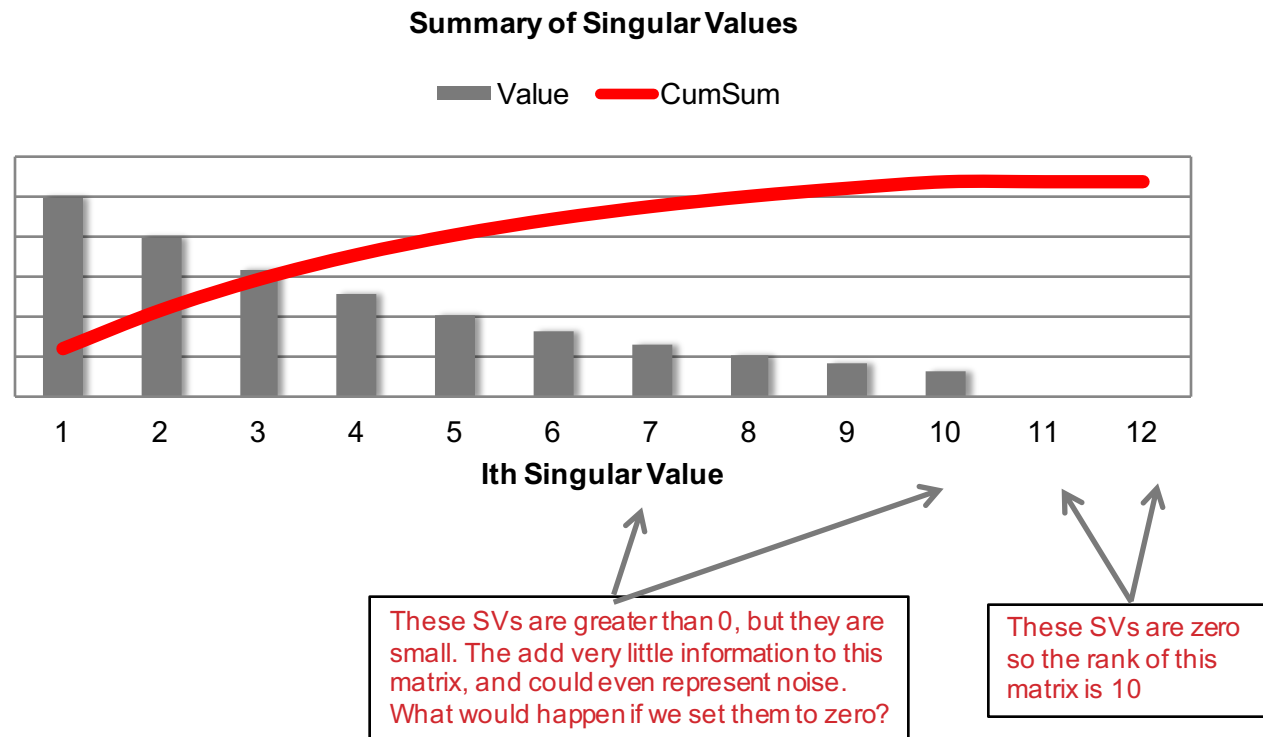
AN INCREDIBLY USEFUL APPLICATION OF SVD

One of the most powerful applications of the SVD is creating a low rank approximation of a data matrix. Many of the following methods are based on using the SVD to get a low rank approximation.

- **Data Compression**
- **Dimensionality Reduction**
- **Recommender Systems**
- **Clustering in High Dimensions**

THE LOW RANK APPROXIMATION

The rank of a matrix is the size of the largest number of independent columns of a matrix. The rank can be found by counting the number of singular values > 0 .



THE LOW RANK APPROXIMATION

We can build a matrix X_k that approximates our original matrix by doing the following:

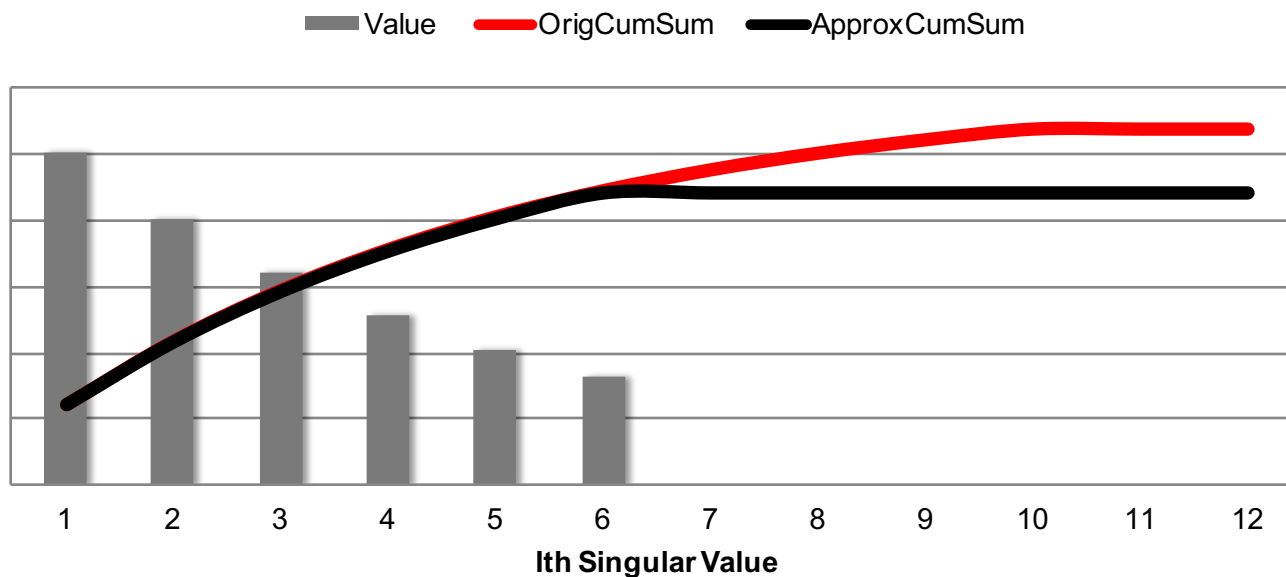
1. Compute the SVD of X
2. Truncate U , Σ and V to take only the k highest columns & singular values
3. Multiply back together to get X_k

$$\begin{array}{c} \text{Low Rank Approximation – } K \ll M \\ \mathbf{X}_k = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^T \\ \begin{array}{cccc} N \times M & N \times K & K \times K & (M \times K)^T \end{array} \end{array}$$

THE LOW RANK APPROXIMATION

Our original matrix had 12 columns with a rank of 10. In our approximation, we decided to use $k=6$. The cumulative sum of singular values is related to the amount of information contained in the matrix. By using $k=6$, we lost some information, but not half.

What do we gain with the low rank approximation?
What do we lose?




Copyright: Brian d'Alessandro, all rights reserved

EXAMPLE: DATA COMPRESSION

Some facts

- A floating point number uses 8 bytes of memory
- An $M \times N$ matrix of floating point numbers uses $8 \cdot M \cdot N$ bytes of memory.

Instead of storing X , lets use X_k , a low rank approximation

U_k	...this is $8 \cdot N \cdot K$ bytes		When separated...
Σ_k	...this is $8 \cdot 1 \cdot K$ bytes		$8 \cdot K \cdot (N + M + 1)$ bytes
V_k^T	...this is $8 \cdot K \cdot K$ bytes		

If you choose $K \ll M$, you could save a ton of space...

i.e. $k=100, N=1\text{MM}, M=10k$, the full X is 80 GB, whereas storing the decomposed components of X_k would only be 0.8 GB!

THE COST

This result comes from the Eckart-Young-Mirsky Theorem

If X is an $n \times m$ matrix and X_k is the rank- k approximation derived from the SVD, then the sum-of-squares error between the entries of X and X_k equals the square-root of the sum-of-squares of the singular values $> k$.

i.e.

$$\|X - X_k\|_F = \sqrt{\sum_{r=k+1}^{\min(m,n)} \sigma_r^2}$$

Where the Frobenius norm is defined as:

$$\|A\|_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^m a_{ij}^2}$$

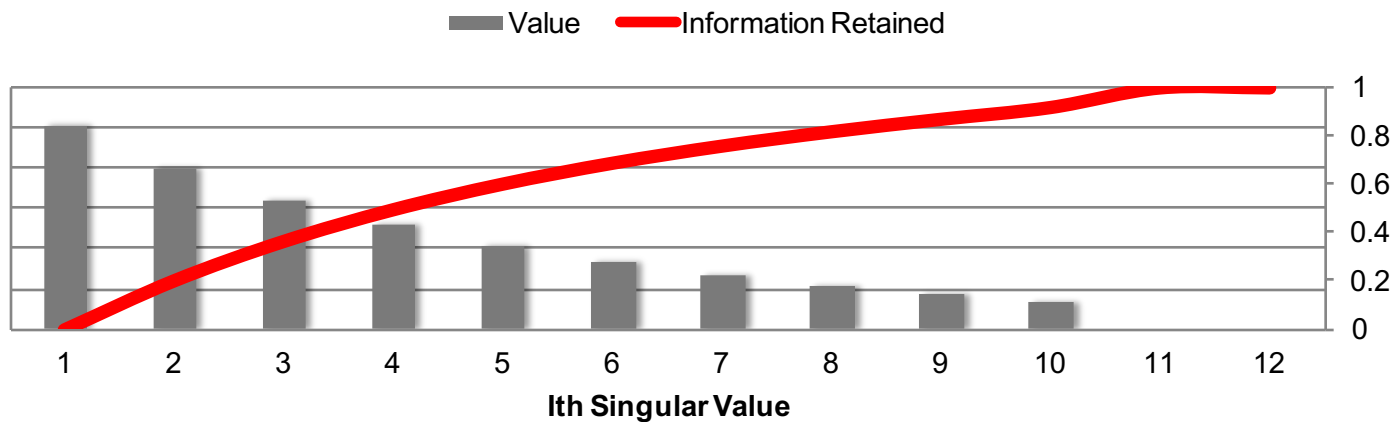
THE COST - EXAMPLE

Given a matrix with the Singular Values shown below, the “Information Retained” shows:

$$1 - \frac{\|X - X_k\|_F}{\|X\|_F}$$

This is very similar to the R^2 metric in linear regression, and it tells us how well our rank- k approximation “fits” the the original matrix from a least squares sense. Analyzing a rank- k approximation like this gives us a tool to choose k .

Summary of Singular Values



Copyright: Brian d'Alessandro, all rights reserved

DIMENSIONALITY REDUCTION

Sometimes a dataset has too many features for various algorithms (or storage systems to handle). Often times there is also a lot of redundancy of the data. The SVD gives us a way to reduce the number of variables without losing too much information.

Compute rank-k SVD $X_k = U_k \Sigma_k V_k^T$

We can create an NxK reduced matrix with $X V_k$. This effectively becomes our new “X” matrix, and we do any analysis (clustering, modeling, etc.) with the reduced matrix.

We can project any new data into the reduced space by, $X' V_k$ and then use this in our algorithms.

Again, the optimal choice of K is dependent on the problem and will be a tradeoff between information loss vs. constraint tolerance.

TO BE CONTINUED...

Other applications of SVD are in recommender systems as well as clustering.

These will be presented in a later lecture.