**Singular Value Decomposition Takeaways**

Three main takeaways from the previous notebook:

1. The latent factors retrieved from SVD aren't actually labeled.
2. We can get an idea of how many latent factors we might want to keep by using the Sigma matrix.
3. SVD in NumPy will not work when our matrix has missing values. **This makes this technique less than useful for our current user-movie matrix.**

### SVD Closed Form Solution

#### What Is A Closed Form Solution?

A closed form solution is one where you can directly find the solution values (unlike iterative solutions, which are commonly used in practice). There isn't an iterative approach to solving a particular equation. One of the most popular examples of a closed form solution is the solution for multiple linear regression. That is if we want to find an estimate for \beta*β* in the following situation:

y = X\beta*y*=*Xβ*

We can find it by computing the **Best Linear Unbiased Estimate (BLUE)**. It can be found **in closed form**using the equation:

\hat{\beta} = (X'X)^-X'y*β*^​=(*X*′*X*)−*X*′*y*

where **X** is a matrix of explanatory inputs and **y** is a response vector.

Another common example of a closed form solution is the quadratic equation. If we want to find **x** that solves:

ax^2 + bx + c = 0*ax*2+*bx*+*c*=0

We can find these values using the quadratic formula:

x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}*x*=2*a*−*b*±*b*2−4*ac*​​

**Each of these is an example of a closed form solution, because in each case we have an equation that allows us to solve directly for our values of interest.**

#### Closed Form Solutions for SVD

It turns out there is a closed form solution for Singular Value Decomposition that can be used to identify each of the matrices of interest (U, \Sigma, V*U*,Σ,*V*). The most straightforward explanation of this closed form solution can be found at [**this MIT link**](http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm).

As put in the paper -

"Calculating the SVD consists of finding the eigenvalues and eigenvectors of AA'*AA*′ and A'A*A*′*A*. The eigenvectors of A'A*A*′*A* make up the columns of V*V*, the eigenvectors of AA'*AA*′ make up the columns of U*U*. Also, the singular values in \SigmaΣ are square roots of eigenvalues from AA'*AA*′ or A'A*A*′*A*. The singular values are the diagonal entries of the \sigma*σ* matrix and are arranged in descending order. The singular values are always real numbers. If the matrix A*A* is a real matrix, then U*U* and V*V* are also real."

Again, you can see a fully worked example of the closed form solution at the [**MIT Link here**](http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm).

#### A More Common Approach

The main issue with the closed form solution (especially for us) is that it doesn't actually work when we have missing data. Instead, Simon Funk (and then many followers) came up with other solutions for finding our matrices of interest in these cases using **gradient descent**.

So all of this is to say, people don't really use the closed form solution for SVD, and therefore, we aren't going to spend a lot of time on it either. The link above is all you need to know. Now, we are going to look at the main way that the matrices in SVD are estimated, as this is what is used for estimating values in FunkSVD.

#### Additional Resources

Below are some additional resources in case you are looking for others that go beyond what was shown in the simplified MIT paper.

* [**Stanford Discussion on SVD**](http://infolab.stanford.edu/~ullman/mmds/ch11.pdf)
* [**Why are Singular Values Always Positive on StackExchange**](https://math.stackexchange.com/questions/2060572/why-are-singular-values-always-non-negative)
* [**An additional resource for SVD in Python**](https://machinelearningmastery.com/singular-value-decomposition-for-machine-learning/)
* [**Using Missing Values to Improve Recommendations in SVD**](https://www.hindawi.com/journals/mpe/2015/380472/)