**Conclusion**

In this lesson, you got your hands on some of the most important ideas associated with recommendation systems:

**Recommender Validation**

You looked at methods for validating your recommendations (when possible) using offline methods. In these cases, you could split your data into training and testing data. Frequently this split is based on time, where events earlier in time are in the training data, and events later in time are in a testing dataset.

We also quickly introduced the idea of being able to see how well your recommendation engine works by simply throwing it out into the world to directly see the impact.

**Matrix Factorization with SVD**

Next, we looked at matrix factorization as a technique for making recommendations. Traditional singular value decomposition a technique can be used when your matrices have no missing values. In this decomposition technique, a user-item (**A**) can be decomposed as follows:

A = U\Sigma V^T*A*=*U*Σ*VT*

Where

* U*U* gives information about how users are related to latent features.
* \SigmaΣ gives information about how much latent features matter towards recreating the user-item matrix.
* V^T*VT* gives information about how much each movie is related to latent features.

Since this traditional decomposition doesn't actually work when our matrices have missing values, we looked at another method for decomposing matrices.

**FunkSVD**

FunkSVD was a new method that you found to be useful for matrices with missing values. With this matrix factorization you decomposed a user-item (**A**) as follows:

A = UV^T*A*=*UVT*

Where

* U*U* gives information about how users are related to latent features.
* V^T*VT* gives information about how much each movie is related to latent features.

You found that you could iterate to find the latent features in each of these matrices using gradient descent. You wrote a function to implement gradient descent to find the values within these two matrices.

Using this method, you were able to make a prediction for any user-movie pair in your dataset. You also could use it to test how well your predictions worked on a train-test split of the data. However, this method fell short with new users or movies.

**The Cold Start Problem**

Collaborative filtering using FunkSVD still wasn't helpful for new users and new movies. In order to recommend these items, you implemented content based and ranked based recommendations along with your FunkSVD implementation.