

⚠ Your bundle subscription to Data Scientist Nanodegree will end on Nov 27, 2021, and will be auto-renewed on a monthly basis. [Check your subscriptions](#) to learn more.



SEARCH	Q
RESOURCES	▲
CONCEPTS	
✓ 18. Notebook: How Are We Doing?	
✓ 19. Screencast: How Are We Doing?	
✓ 20. Video: The Cold Start Problem	
✓ 21. Notebook: The Cold Start Prob...	
✓ 22. Screencast: The Cold Start Pro...	
✓ 23. Video: Putting It All Together	
✓ 24. Screencast: Code Walkthrough	
✓ 25. Workspace: Recommender M...	
✓ 26. Video: Conclusion	
✓ 27. Text: Review	



Mentor Help

Ask a mentor on our Q&A platform



Peer Chat

Chat with peers and alumni

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 these cases, you could split your data into training and testing data. Frequently the time, where events earlier in time are in the training data, and events later in time are in the test dataset.

We also quickly introduced the idea of being able to see how well your recommender performs 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. Singular value decomposition is a technique that can be used when your matrices have a rectangular shape. This decomposition technique, a user-item (**A**) can be decomposed as follows:

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

Where

- U gives information about how users are related to latent features.
- Σ gives information about how much latent features matter towards recreating the matrix.
- V^T 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. For matrix factorization you decomposed a user-item (**A**) as follows:

$$A = UV^T$$

Where

- U gives information about how users are related to latent features.
- V^T 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 of the latent features in the matrices.

Using this method, you were able to make a prediction for any user-movie pair in the dataset. You also could use it to test how well your predictions worked on a train-test split of the dataset. The 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. To recommend these items, you implemented content based and ranked based recommendations with your FunkSVD implementation.