# Recommending Movies



Netflix is very interested in making good movie recommendations to their subscribers. One rudimentary way to make movie recommendations is to measure the similarity between subscribers based on the genre of film that they prefer, classifying subscribers as “horror watchers” or “rom-com viewers”. Then Netflix could recommend films that other users with a similar classification liked.

To begin to think about this, consider the viewing histories over the last month for the following three subscribers:

* **Subscriber 1:** Watched 3 Dramas, 1 Rom-Com, and 1 Action movie
* **Subscriber 2:** Watched 3 Dramas, 1 Documentary, and 1 Horror film
* **Subscriber 3:** Watched 9 Dramas, 3 Rom-Coms, and 3 Action movies

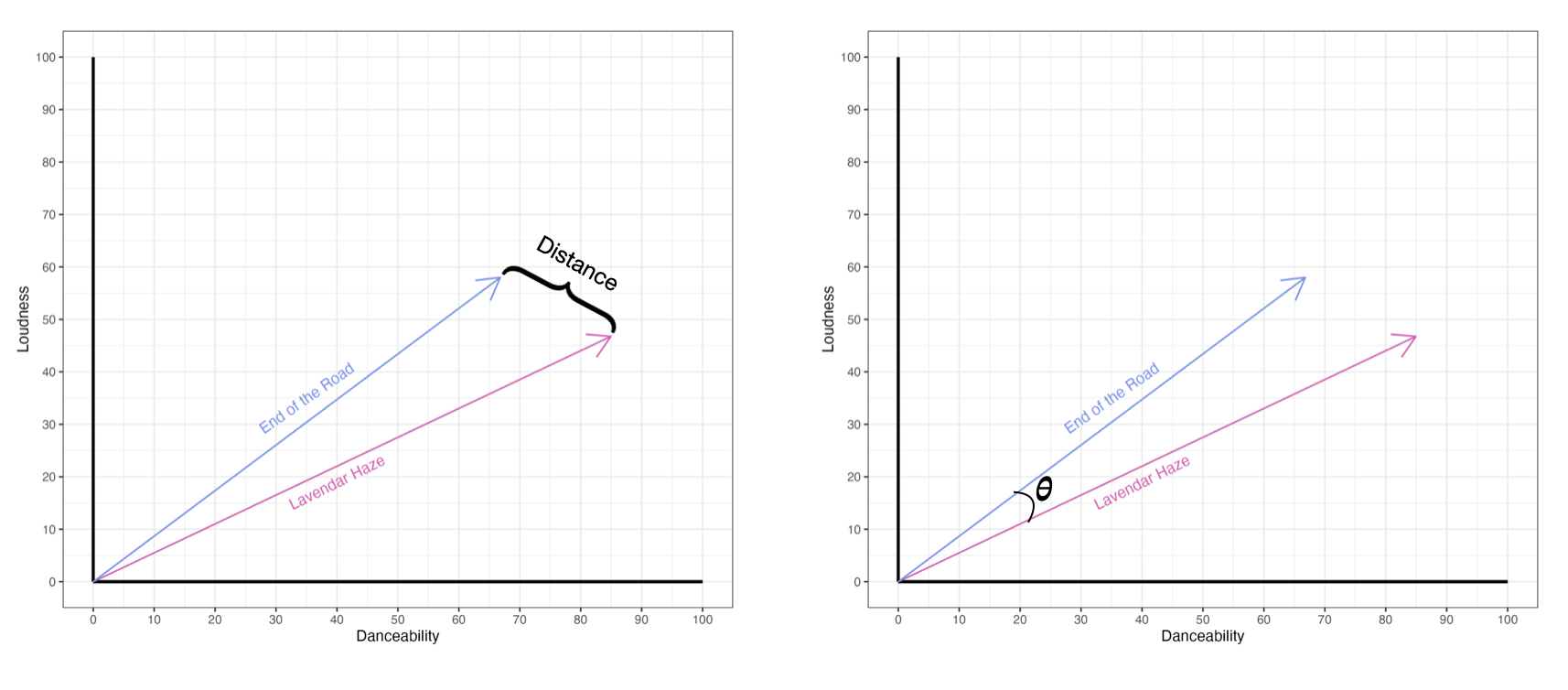
1. Which two subscribers are *most similar* in their viewing profiles? Explain. (Don’t use any mathematics to answer this.)
2. Create a vector to express each subscriber’s viewing history and compute the Euclidean distance between each pair of users.
3. Based on the distances you computed in Question #2, which two subscribers are *most similar* in their viewing profiles?

Measuring similarity between user profiles using Euclidean distance leads to results that don’t match our intuition that Subscriber 1 and 3 have a similar profile. The issue is that the number of movies viewed has a big impact on the Euclidean distance, and since the magnitude of movies is quite different between Subscriber 1 and 3, the Euclidean distance suggests that those subscribers are not similar. In contrast, the magnitude of movies viewed between Subscriber 1 and 2 are much closer, which is why the Euclidean distance indicated those two subscribers are more similar, despite having different viewing profiles.

In this situation, what is of more use in gauging similarity of subscribers’ viewing profiles is the ratio of movies watched in each genre rather than the raw number of movies. (The choice of whether the raw values or the ratio of these values is more important is really context dependent and is a decision that needs to be made by the data analyst.) To offset this issue we can do one of two things: (1) scale the number of movies viewed for each subscriber so that they have a comparable magnitude on which to compute similarity, or (2) use a different measure of similarity.

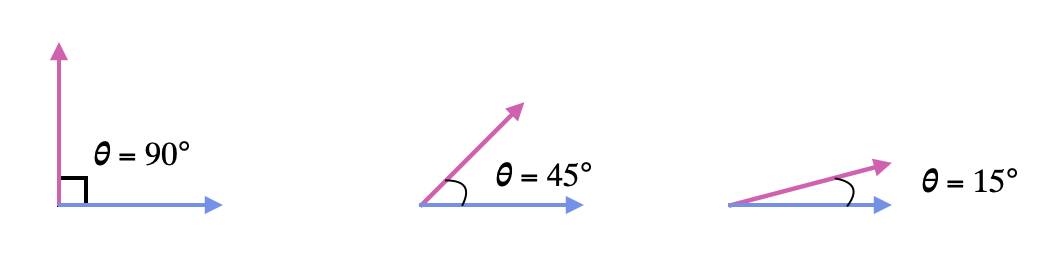
## Cosine Similarity

One method of measuring similarity that deals with this issue is called *cosine similarity*. The idea behind cosine similarity is that instead of measuring distance between two vectors, we instead measure the angle between the vectors. You can see an example of this in the right-hand plot in the figure below. Hopefully you can see in your mind’s eye that if we increase the angle (𝛳) between the vectors, the distance between the vectors will also increase. Thus, we say that two vectors are more similar if the angle between them is smaller.



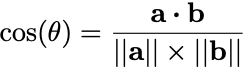
Rather than determining the angle between the two vectors, we actually compute the cosine of that angle in practice. In the Taylor Swift and Netflix applications, since the values on the different metrics (e.g., danceability, number of films watched) can only be positive, 𝛳 can only range from 0 to 90. This implies that taking the cosine of 𝛳 will result in a value between 0 and 1.

1. Consider the following three sets of vectors. Compute the cosine of the angle 𝛳 for each set of vectors.



1. Describe the relationship between the value of the cosine and the similarity of two vectors.

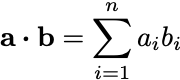
There is a nice mathematical formula[[1]](#footnote-0) that allows you to compute the cosine of the angle between two vectors **a** and **b**. This formula is:



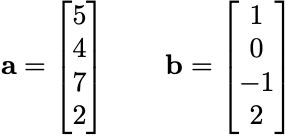
where **a·b** is the dot product of vectors **a** and **b**, and ||**a**|| and ||**b**|| are the norms (lengths) of vector **a** and vector **b**, respectively.

### **Math Extension: Dot Product**

The dot product of vectors **a** and **b** is defined as:



In other words, the dot product is calculated by multiplying together the corresponding elements of each vector, and summing those products. Consider the vectors,



The dot product, or **a·b**, is calculated as:

**a·b** =5(1) + 4(0) + 7(−1) + 2(2)

= 2

1. Compute the cosine between each pair of Netflix subscribers:
   * **Subscriber 1:** Watched 3 Dramas, 1 Rom-Com, and 1 Action movie
   * **Subscriber 2:** Watched 3 Dramas, 1 Documentary, and 1 Horror film
   * **Subscriber 3:** Watched 9 Dramas, 3 Rom-Coms, and 3 Action movies
2. Which two subscribers are *most similar* in their viewing profiles? Explain.

## Item-Based Collaborative Filtering

In the previous sections, the method for recommending movies was based on similarity between subscribers—if Subscriber A and Subscriber B are similar, movies liked by Subscriber A should be recommended to Subscriber B. Another method of making recommendations is item-based collaborative filtering[[2]](#footnote-1). The idea underlying this method is that the similarity between movies (based on subscriber ratings) is measured, and then recommendations are made to correspond to how a subscriber rates particular movies. For example, if you rate the movie *Toy Story* highly, then other movies that are similar to *Toy Story* (based on their ratings) will be recommended to you; you often see these recommendations in a “You may also enjoy…” section!

[MovieLens](https://movielens.org/) is a personal movie recommender site that is run by [GroupLens Research](https://grouplens.org/) at the University of Minnesota. Users rate movies they have seen using 1–5 stars. (A rating of zero indicates that the user has not seen/rated the movie.) Below are 10 users’ ratings for two movies[[3]](#footnote-2).

| **Title** | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Toy Story | 2.5 | 0 | 4.5 | 4 | 4 | 3.5 | 4 | 5 | 4 | 3 |
| RoboCop | 2 | 0.5 | 4 | 2.5 | 3.5 | 0 | 0 | 5 | 3 | 0 |

1. Compute the cosine distance between *Toy Story* and *Robocop*. Show your work.
2. Use the data in the [MovieLens-data](https://docs.google.com/spreadsheets/d/1cClcZn9FDslRC3A7rrfDyWumRZfq5GgZaJtdKnC_53I/edit?usp=sharing) Google Sheet to compute the cosine distance between *Toy Story* and *Robocop*. Do this by writing a formula in the spreadsheet to compute this. Make sure that the resulting cosine similarity matches your answer to Question #8 (within rounding).
3. Apply your formula to measure the cosine similarity between *Toy Story* and each of the other movies in theGoogle Sheet.
4. Provide the Top 5 recommendations for a user who gave *Toy Story* a high rating based on your cosine similarity scores.
5. Provide the Top 5 recommendations for a user who gave *Toy Story* a low rating based on your cosine similarity scores.

## Bonus Features

We can also use cosine similarity to classify cases in the same way we did with Euclidean distance. Rather than computing Euclidean distance, we instead compute cosine similarity, and then use those similarity measures in the *k-*nearest neighbors algorithm.

1. Go back to Activity 5: Blank (Vector) Space and re-do Questions #5–8 using cosine similarity rather than Euclidean distance.
2. Did any of the songs get classified differently? Explain.

1. This formula is directly related to the Law of Cosines that you may have learned about in a trigonometry class. [↑](#footnote-ref-0)
2. Item-based collaborative filtering (a.k.a., item-to-item collaborative filtering) was developed by Amazon in 1998. You can read more [here](https://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf). [↑](#footnote-ref-1)
3. These data were extracted from the data set provided by: Harper, F. M., & Konstan, J. A. (2015). The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS) 5*(4): 19:1–19:19. <https://doi.org/10.1145/2827872> [↑](#footnote-ref-2)