Bernard Kurka

42 Followers · About Follow

Photo by Christian Wiediger on Unsplash

Building a movie recommender system with Python

Bernard Kurka May 7, 2019 · 7 min read

Hello reader! In this post, I will walk through how I used Python to build a movie recommender system. In the first part, I will explain how cosine similarity works, and in the second I will apply the Data Science process to build a simple recommender system.

The problem came up because I was not satisfied with Netflix recommendations. I believe their recommendation system was inaccurate because of the four factors below.

1. A limited number of Movies

There are plenty of good films for me to watch but not all of them are available on Netflix.

2. I dislike most Netflix originals

Most of the recommendations I receive are on Netflix Originals and in my opinion, they deliver several bad movies for each good movie they are able to create.

3. I was not rating the movies

I have never rated the movies after I've watched them.

4. Two people sharing the same profile

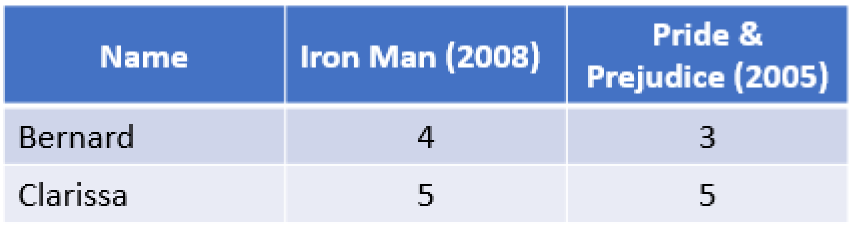
When people share their account the recommender algorithm gets less effective in capturing the entertainment preferences of each user.

An interesting fact is that the Netflix recommender system is responsible for directing 80% of the content watched.

Part 1: How cosine similarity works for building Recommenders

Cosine similarity is a method to measure the difference between two non zero vectors of an inner product space. See the example below to understand.

Suppose I want to check if Bernard and Clarissa have similar movie preferences, and I only have two movie reviews. The reviews are scores from 1 to 5, where 5 is the best score and 1 the worst, and 0 means that a person has not watched the movie.



Name	Iron Man (2008)	Pride & Prejudice (2005)
Bernard	4	3
Clarissa	5	5

I can represent each person's reviews in a separate vector.

$$\vec{b} = \begin{bmatrix} 4 \\ 3 \end{bmatrix} \qquad \vec{c} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

Vector b represents Bernard and vector c Clarissa.

The cosine similarity will measure the similarity between these two vectors which is a measurement of how similar are the preferences between these two people.

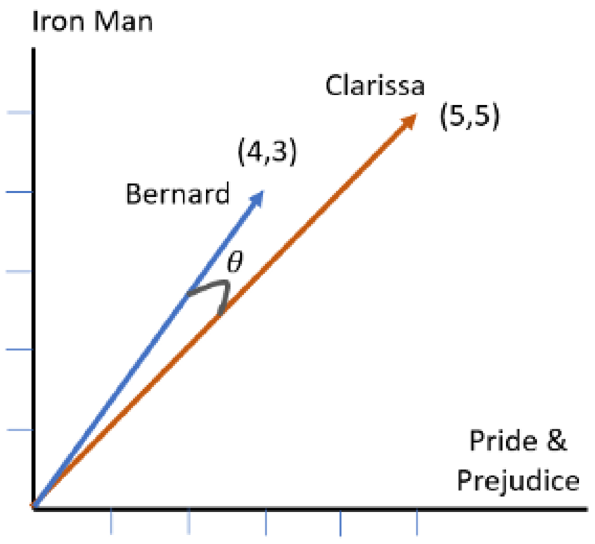
$$similarity = \cos\theta = \frac{b.c}{\|b\|\|c\|}$$

$$b.c \Rightarrow Is\ the\ Dot\ product\ of\ the\ two\ vectors$$

$$\|b\|\|c\| \Rightarrow Is\ the\ product\ of\ each\ vector's\ magnitude$$

In the image, below each vector represents a person's preferences and they have an angle θ between them. Similar vectors will have a lower angle θ, and dissimilar vectors (different film preferences) will have bigger θ.

✓



Calculating:

$$b.c = \sum_{i=1}^{n} b_i c_i = (4 \times 5) + (3 \times 5) = 35$$

$$\|b\| = \sqrt{4^2 + 3^2} = 5$$

$$\|c\| = \sqrt{5^2 + 5^2} = 5\sqrt{2}$$

$$similarity = \frac{35}{5 \times 5\sqrt{2}} \sim 0.989$$

In the example above the similarity 0.989 is close to the maximum value of 1, this means that given only two movie reviews the two users have similar preferences.

Theoretically, the cosine similarity can be any number between -1 and +1 because of the *image* of the cosine function, but in this case, there will not be any negative movie rating so the angle θ will be between 0° and 90° bounding the cosine similarity between 0 and 1. If the angle θ = 0° =>cosine similarity = 1, if θ = 90° => cosine similarity =0.

The cosine similarity can be calculated for more than 2 movies. In the example below I will add the ratings for a movie that Bernard liked and Clarissa disliked this should decrease cosine similarity value.

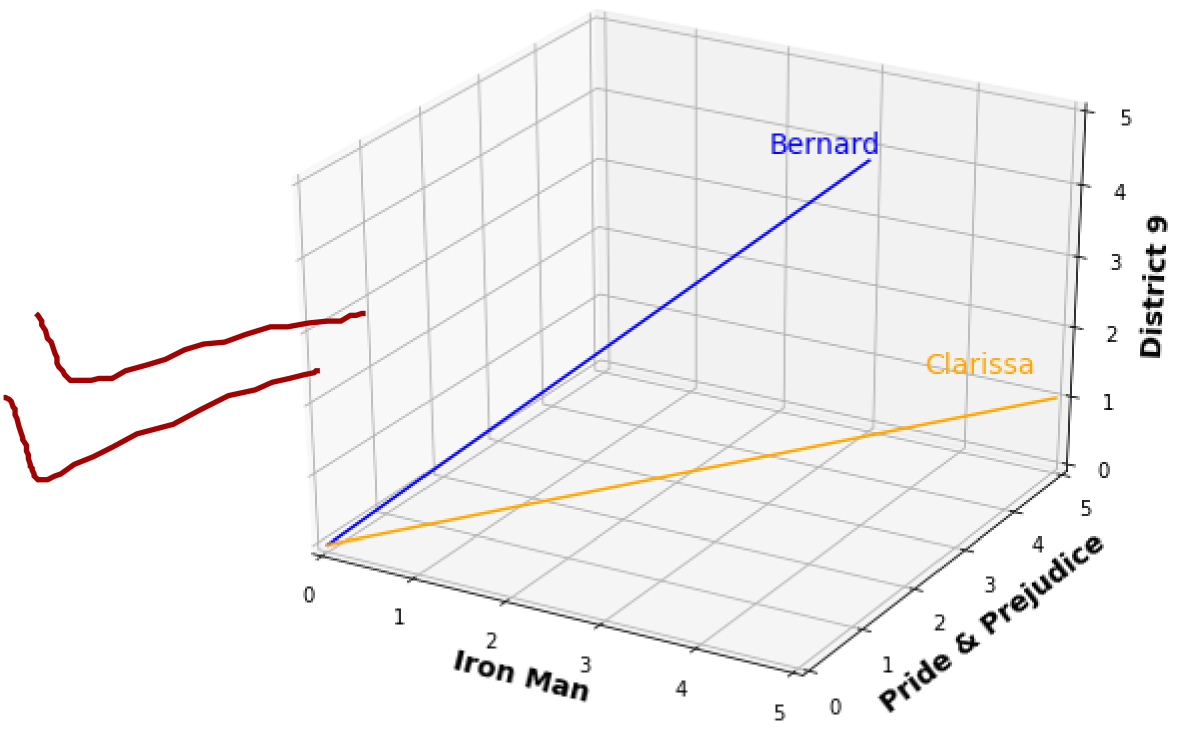
Name	Iron Man	Pride & Prejudice	Ditrict 9
Bernard	4	3	5
Clarissa	5	5	1

The new vectors are:

$$\vec{b} = \begin{bmatrix} 4 \\ 3 \\ 5 \end{bmatrix} \qquad \vec{c} = \begin{bmatrix} 5 \\ 5 \\ 1 \end{bmatrix}$$

Vector b represents Bernard and vector c Clarissa.

The plot has 3 dimensions now:



Calculating the similarity:

$$b.c = (4 \times 5) + (3 \times 5) + (5 \times 1) = 40$$

✓

$$\|b\| = \sqrt{4^2 + 3^2 + 5^2} = 5\sqrt{2}$$

$$\|c\| = \sqrt{5^2 + 5^2 + 1^2} = \sqrt{51}$$

$$similarity = \frac{40}{5\sqrt{2} \times \sqrt{51}} \sim 0.792$$

The similarity has reduced from 0.989 to 0.792 due to the difference in ratings of the District 9 movie. The cosine can also be calculated in Python using the Sklearn library.

```
import sklearn.metrics.pairwise as pw
pw.cosine_similarity([[4,3,5]],[[5,5,1]])

Or

from sklearn.metrics.pairwise import pairwise_distances
pairwise_distances([[4,3,5]],[[5,5,1]],metric='cosine')
```

The latter (pariwise_distances) will result in 1- cosine similarity. Both functions can receive a SciPy 2-d sparse matrices. A SciPy 2-d sparse matrix is a more efficient way of representing a matrix in which most elements are zero.

Part 2: Data Science process to build a recommender system

Problem Statement:

Create a list of movie recommendations based on a movie I liked or a list of rated movies.

Data:

The data set contained 100.000 movie ratings, produced by 600 users. The data included 9.000 movie titles and is available at the GroupLens Research website.

I also imported an excel file with a list of 65 movies rated by a friend.

Cleaning the data:

The data set is pretty neat. I only found 5 duplicated `movieID` values and removed the ones that had the lowest count of user ratings.

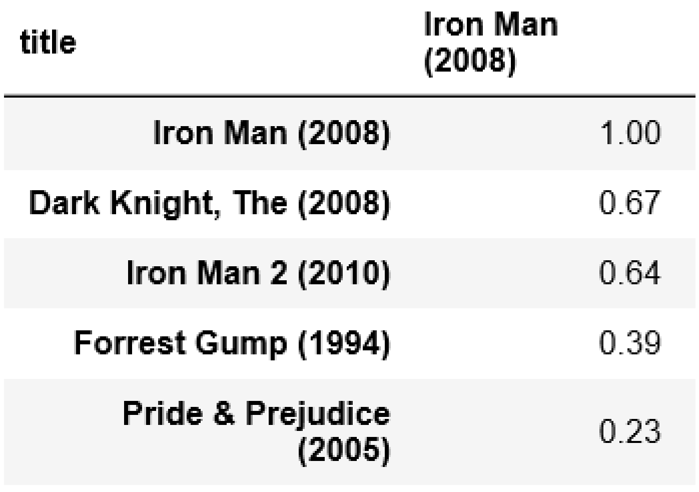
Building a Model\Recommender:

I’ve built 3 different recommenders based on the cosine similarity.

1. Item-Based Recommender:

Invented in 1998 by Amazon, item-to-item or item-based recommender is calculated based on the similarity between items using people’s ratings on those items. In this project, each item is represented by a movie. For example, in order to calculate the similarity between Iron Man (2008) and Iron Man 2 (2010) I created two vectors with all the reviews for each movie and then calculated the vector’s cosine similarity. The vector contains several 0 values to fill in null values when a move was not rated by a user. I calculated the cosine similarity between all 9.000 movies.

In the table below I’ve compared 4 popular movies with Iron Man.



title	Iron Man (2008)
Iron Man (2008)	1.00
Dark Knight, The (2008)	0.67
Iron Man 2 (2010)	0.64
Forrest Gump (1994)	0.39
Pride & Prejudice (2005)	0.23

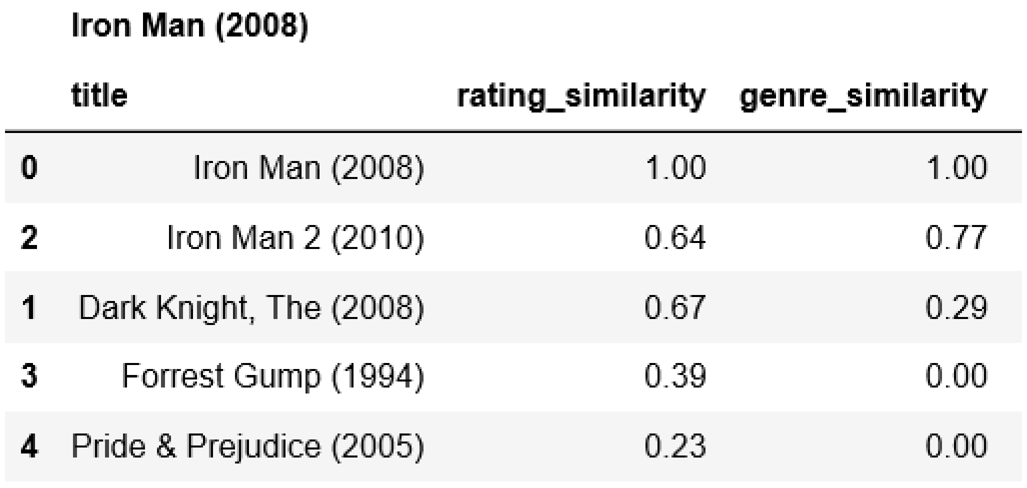
Cosine similarity between Iron Man and 4 popular movies

If I enjoyed watching Iron Man (2008) the information above shows that Iron man (2008) and The Dark Night are the most similar movies with 0.64 and 0.67 similarity scores. Forest Gump (score 0.39) is a better recommendation than Pride & Prejudice (score 0.23). Keep in mind that the similarity scores relies on only 610 user ratings, which is a small sample for creating a robust recommender system.

2. Item-Based and Genre Recommender:

In the previous example, we noticed that Dark Night was a better recommendation than Iron Man 2. But if we look into each film’s genre we find that they are slightly different. Iron Man’s genre is Action, Adventure, and Sci-Fi. Dark Night’s genre is Action, Crime, and Drama.

I added a new layer to the recommender, first I will find movies with similar genres and then select the best rating similarities. In order to do that I've added a new column containing the genre cosine similarity.



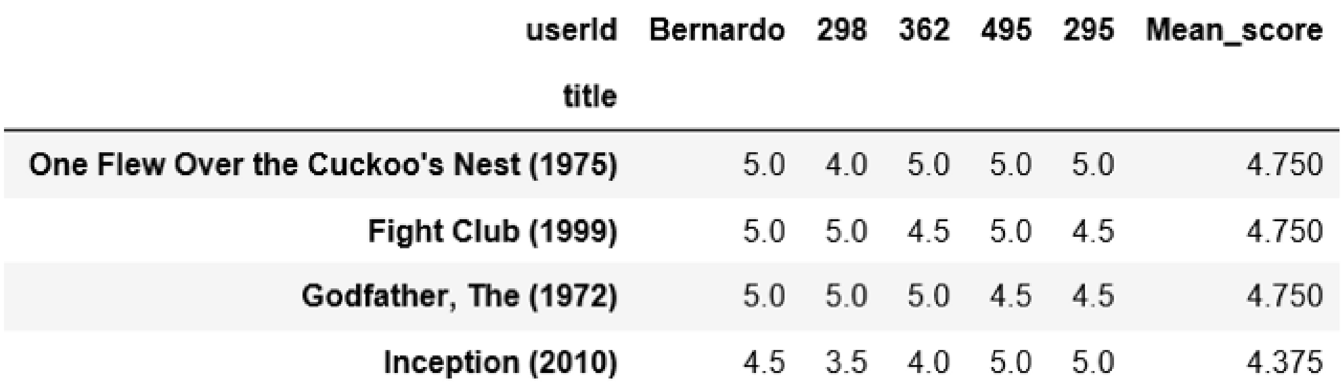
	Iron Man (2008)		
	title	**rating_similarity**	**genre_similarity**
0	Iron Man (2008)	1.00	1.00
2	Iron Man 2 (2010)	0.64	0.77
1	Dark Knight, The (2008)	0.67	0.29
3	Forrest Gump (1994)	0.39	0.00
4	Pride & Prejudice (2005)	0.23	0.00

Cosine similarity using ratings and genre

So the now the Iron Man 2 will come first in the recommendation order because it has a 0.77 genre similarity vs 0.29 of The Dark Knight.

3. User-Based Recommender:

Using two vectors with each person's scores for the 9.000 movies, I can calculate the cosine similarity between these two users. My friend Bernardo rated 65 movies, I've imported his ratings and found the 4 users that had the most similar cosine similarity. And then calculated the mean score for each movie considering their rating. See the table below:



userId	**Bernardo**	**298**	**362**	**495**	**295**	**Mean_score**
title						
One Flew Over the Cuckoo's Nest (1975)	5.0	4.0	5.0	5.0	5.0	4.750
Fight Club (1999)	5.0	5.0	4.5	5.0	4.5	4.750
Godfather, The (1972)	5.0	5.0	5.0	4.5	4.5	4.750
Inception (2010)	4.5	3.5	4.0	5.0	5.0	4.375

Comparing Bernardo's scores with the highest Mean_Score

The table above is filtered in descending order of Mean_score. The numbers (298,362,495 and 295) represent users that have similar preference than Bernardo. The last column is the mean of their movie ratings. The films above were all watched by Bernardo and had good scores. The next step was to filter only movies that Bernardo has not watched already.

Evaluating Recommenders and Answering the problem:

I did not create an evaluation metric nor did build any statistics on its performance. I did manual explorations selecting multiple movies I've like and observed the results. Below I've printed the recommendations for Iron Man 2(2010) and Inception (2010).

Let's see the recommendations for Inception (2010):

User rating-based recommender

User and Genre-based recommender

Let's see the recommendations for Iron Man 2(2010):

Item rating-based recommender

Item and Genre-based recommender

Analyzing the tables above I can say the movies recommended are movies that I enjoyed, and would score them in a range from 3.5 to 5.

The user-based recommender had the following results:

User-based recommender

Talking with my friend Bernardo Bianchi the movies recommended based on his previous ratings are movies that he likes.

Improving the recommender:

1. GroupLens has a data set with more movies and user reviews; using this data set would improve the cosine similarity recommenders.

2. To improve the user-based recommender, it would be important to have more plenty of reviews from multiple friends.

3. To evaluate the user-based recommender it's possible to separate the data into split and test in order to compare recommendations with actual scores from the user.

4. Using more qualitative information about each film might improve the recommender, such as movie's decade, main actor, director, length, and movie cost.

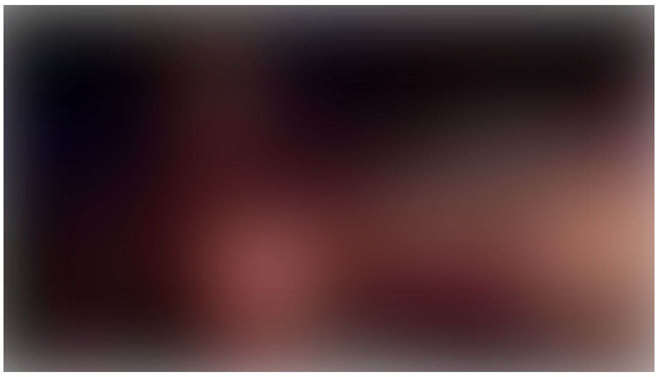
Next Steps:

I will build an online dashboard where a person can select a movie and check out the recommendations.

GitHub project folder: https://github.com/berkurka/MovieRecommender

GitHub Python code:
https://github.com/berkurka/MovieRecommender/blob/master/Movie_Recommender.ipynb

Thanks for reading!



From Giphy.com

Data Science Python Netflix Movies