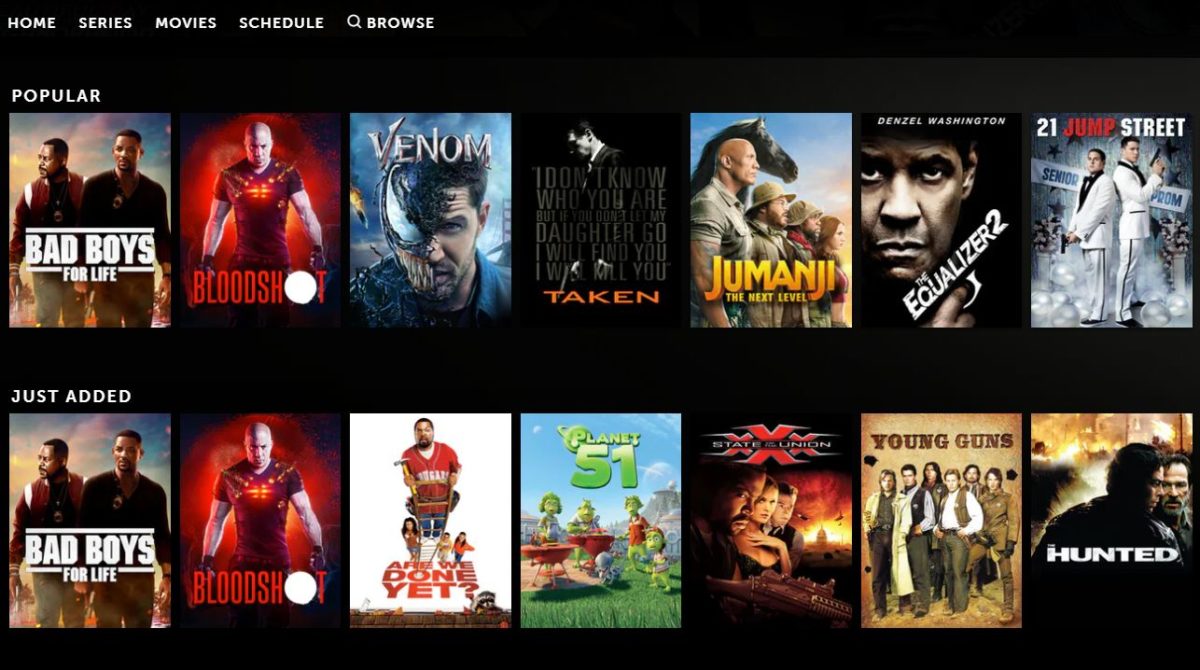
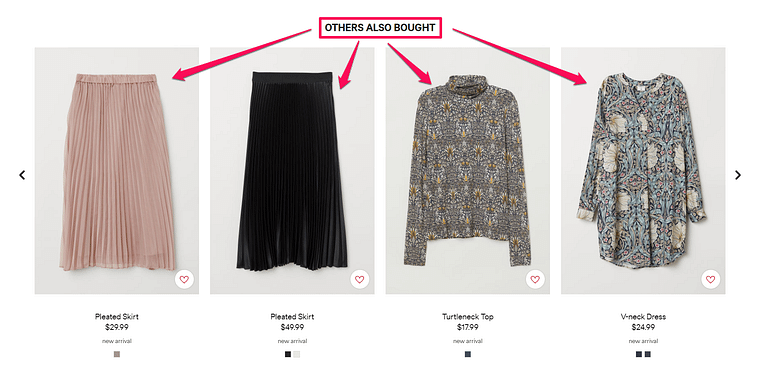
MOVIE RECOMMENDATION SYSTEM



Have you ever been on an online streaming platform like Netflix, Amazon Prime? I watched a movie and after some time, that platform started recommending me different movies and TV shows. I wondered, how the movie streaming platform could suggest me content that appealed to me. Then I came across something known as Recommendation System. Its time to understand What is Recommendation System?

Recommendation System is an information filtering technique, which provides users with information, which he/she may interested in. Recommendation system shows only user desired data which is based on his or similar users preferences and interests rather than overload of information.

Here’s an example of a recommender system in e-commerce. H&M served the following recommendations to users who clicked on “pleated skirt” as a potential buy:



**Why Recommendation systems?**

– They help the user find items of their interest  
– Helps the item provider to deliver their items to the right user  
       – To identify the most relevant products for each user  
       – Showcase personalised content to each user  
       – Suggest top offers and discounts to the right user  
– Websites can improve user-engagement  
– It increases revenues for business through increased consumption

How Recommender Systems Provide Users with Suggestions

Using machine learning, recommender systems provide you with suggestions in a few ways:

* **Collaborative Filtering**
* **Content-based Filtering**

**Collaborative Filtering**

This method finds a subset of users who have similar tastes and preferences to the target user and use this subset for offering recommendations.

Basic Assumptions:

-Users with similar interests have common preferences.

-Sufficiency large number of user preferences are available.

MAIN APPROACHES:

-User Based

-Item Based

User Based Collaborating Filtering

The other kind of collaborative filtering takes the similarity of user tastes into consideration.

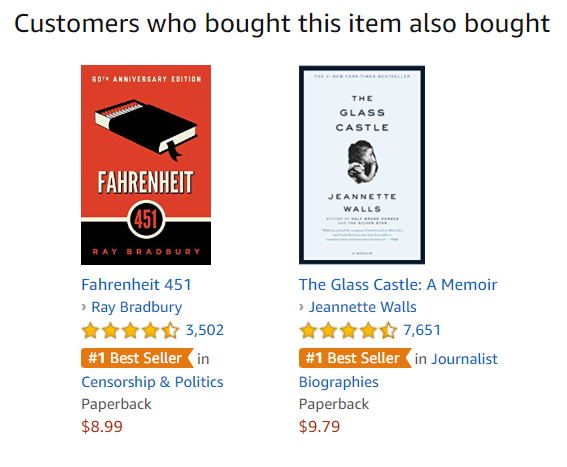
So, user-user collaborative filtering doesn’t serve you items with the best ratings. Instead, you join a cluster of other people with similar tastes and you see content based on historic choices.

Techniques : Recommendation Generation
User Based Collaborative Filtering
Item
User
I1 I2 I3 I4 I5
U1 5 8 7 8
U2 10 1
U3 2...

Here as you see user a is highly correlated with user 3.User 3 like I5 item so we recommend I5 to user a because both has similar tastes. We have high probability that user a also like I5 item.

Lets see real life example-

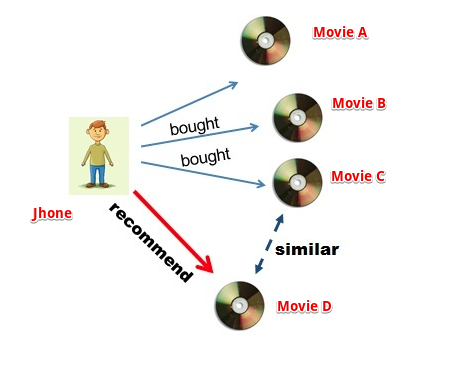
Let’s say you buy Fitzgerald book. The system clusters you with other users who also buy Fitzerald. Then the Amazon recommendation system shows you other books which is buy by users in your cluster. The more choices you make, the more relevant the results.



Here the system served you Fahrenheit 451. That’s because past Fitzgerald customers must have also bought Bradbury.

Item-item collaborative Filtering

An item-item filtering algorithm analyzes product associations taken from user ratings.



Item-item collaborative filtering is one kind of recommendation method which looks for similar items based on the items users have already liked or positively interacted with. It was developed by Amazon in 1998 and plays a great role in Amazon’s success.

But what does that mean when we say item-item similarity? In this case we don’t mean whether two items are the same by attribute like Fountain pen and pilot pen are similar because both are pen. Instead, what similarity means is how people treat two items the same in terms of like and dislike.

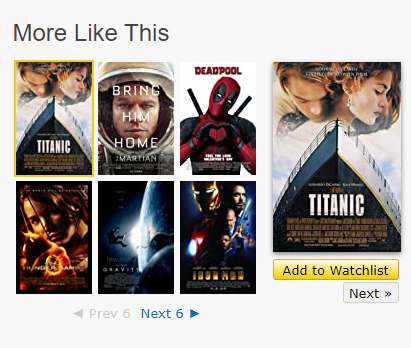
Techniques : Recommendation Generation
Item Based Collaborative Filtering
Item
User
I1 I2 I3 Item I I5
U1 5 8 7 8
U2 10 1
...

Here you see in table Item1 is highly correlated with I5 based on users ratings.

Example-

You search for Avatar movie so it will recommend you similar movies based on users ratings.





**Content Based Recommender Systems**

Content based filtering uses the characteristics or properties of an item to serve recommendations. Characteristic information includes:

* **Characteristics of Items (Keywords and Attributes)**
* **Characteristics of Users (Profile Information)**

Let’s use a movie recommendation system as an example. Characteristics for the item *Harry Potter and the order of the phoenix* might include:

Director: David Yates

Writers:Michael Goldenberg(screenplay),J.K Rowling (novel)

Stars:Daniel Radcliffe,Emma Watson,Rupert Grint

**Genres:Action,Adventure,Family**



A content based recommender system can now serve the user:

* **More Harry Potter Movies**
* **More Adventure, Family, or Action Movies**
* **More David Yates Movies**
* **More Daniel Radcliffe Movies**

Of course, the list is not exhaustive. Once the user makes choices, the recommender system can serve more targeted results.

**How is the Similarity Computed between the different products?**

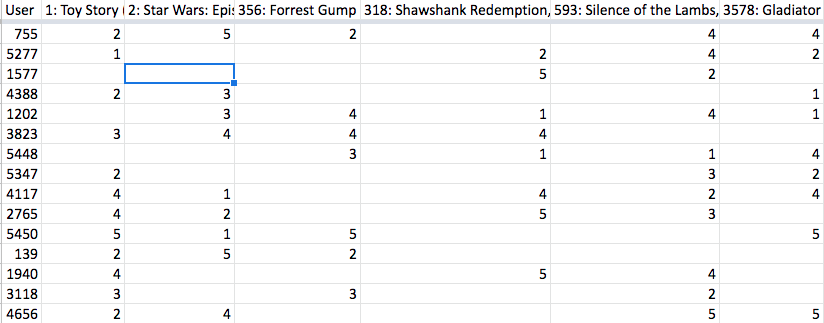
The similarity is the main key fundamental in the case of content-based recommendation systems. A most similar thing to what we are currently watching gets recommended to us. The question is how? Let us understand how similarity between different products is computed. There are different techniques or similarity measures that are used to compute the similarity. Let us understand them:-

**Euclidean Distance:-** This distance metric is used when we have numeric data. For example, if I want to compute the similarity between One plus 6 and other one plus variants based on ram and camera. The values of ram and camera for each variant would be in numbers. In these cases, we calculate Euclidean distance if the results of this distance come out to be 0 then both the two are considered to be similar whereas if the distance is anything other than 0 then are not similar.

**Cosine Similarity:-**  This type of metric is used to compute the similarity textual data. Consider an example where we have to find similar news or similar movies. How is it done? We convert these textual data in the form of vectors and check for cosine angle between those two vectors if the angle between them is 0. It means they are similar or else they are not. Most used similarity measures when we talk about the similarity between any textual content. There are other different metrics as well like Jaccard Similarity that is used when we have categorical data.

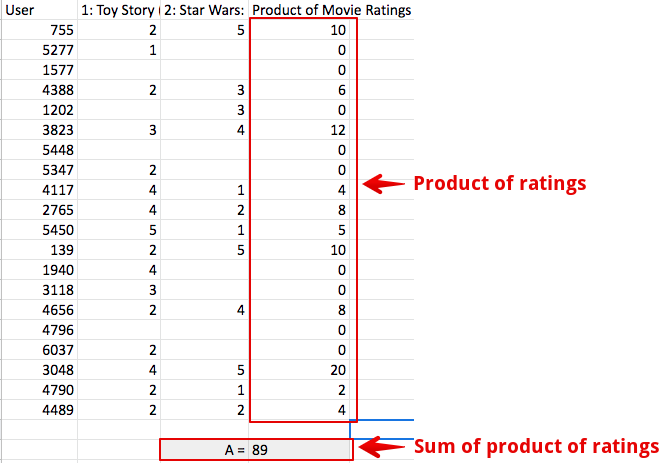
Lets see an example how it find the similarity between items

Suppose we have a table of users and their ratings for movies

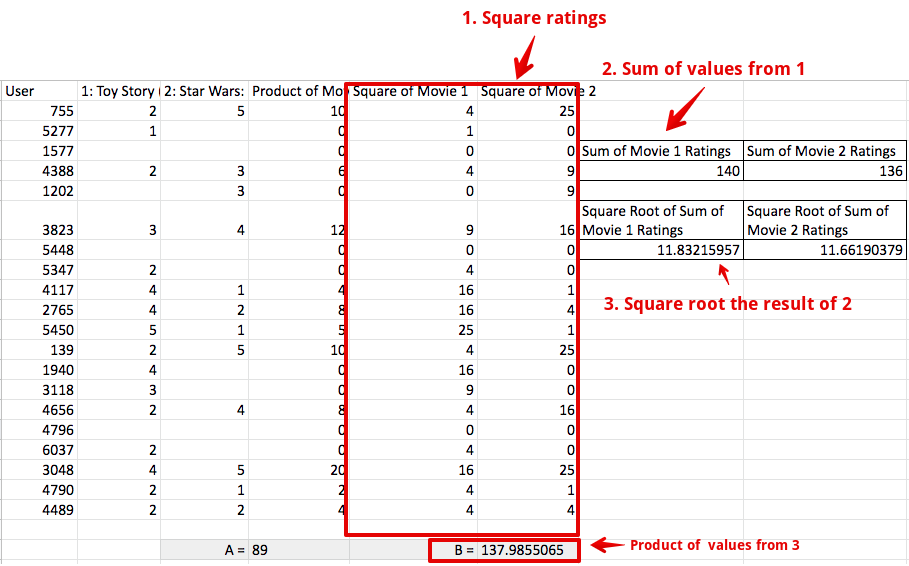


Let’s pick two movies (Id 1: Toy Story and Id 2: Star Wars) for which we have to calculate the similarity i.e. how much these two movies are comparable to one another in terms of there likeness by users. To compute this we will:

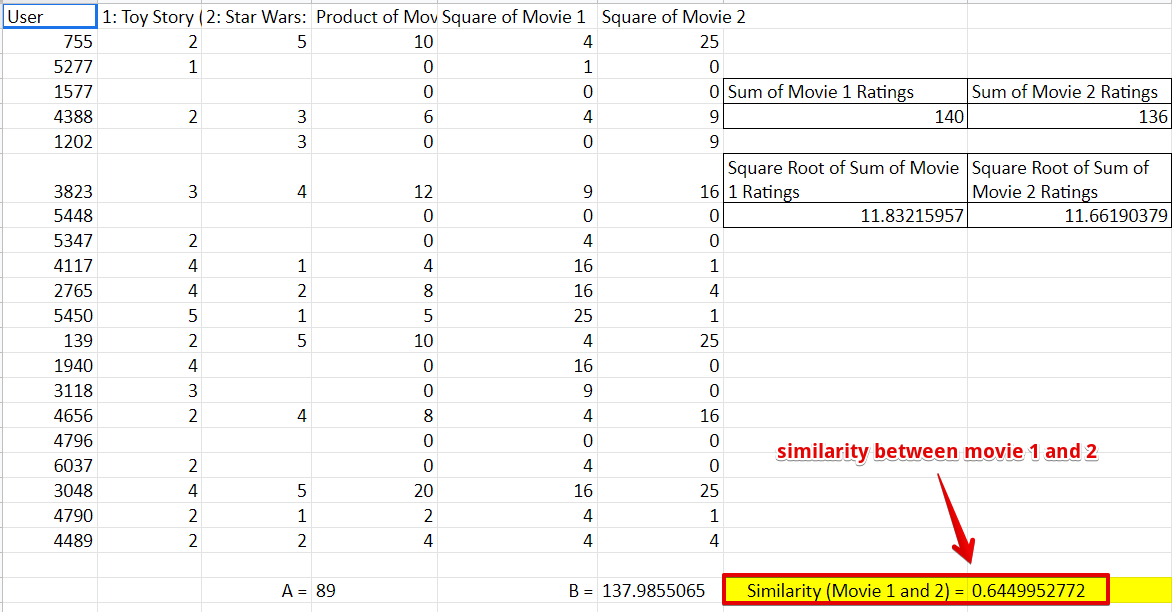
First multiple ratings of both the movies with each other and the sum the result. Let’s call this value ‘**A’**.



Second, we‘ll sum the squared movie ratings and then take the square root of them. So square all movie 1 ratings, sum them and then take the square root to get final value (do same of movie 2). Doing so will get us two values i.e. square root value of movie 1 and movie 2. Multiply both the values. We call this final value ‘**B’**

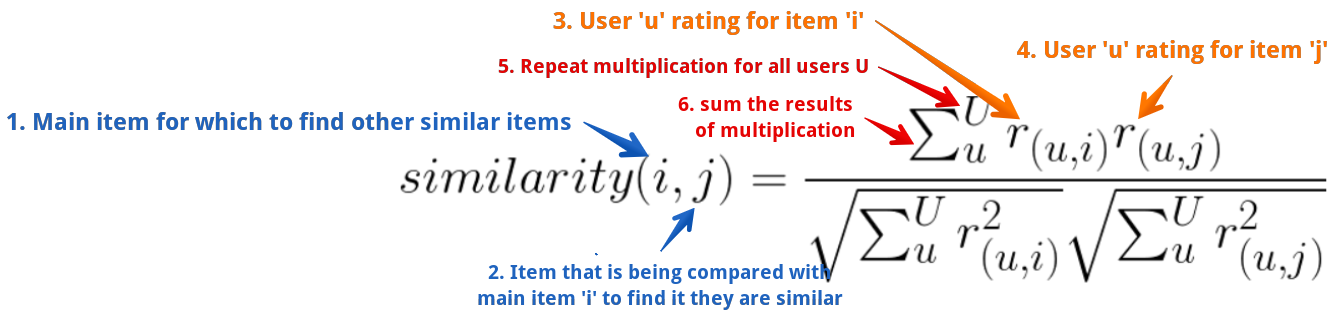


Third, divide A and B, this will gets us a score that indicates how close movie 1 and movie 2 are to one another

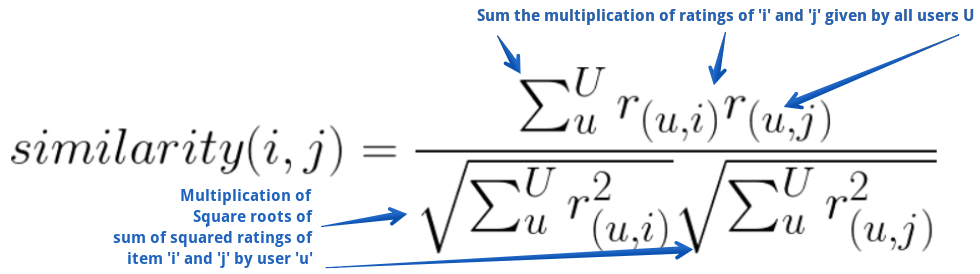


Repeating the above process for all the movies will result in a table with similarities between each movie(in general we call them items).

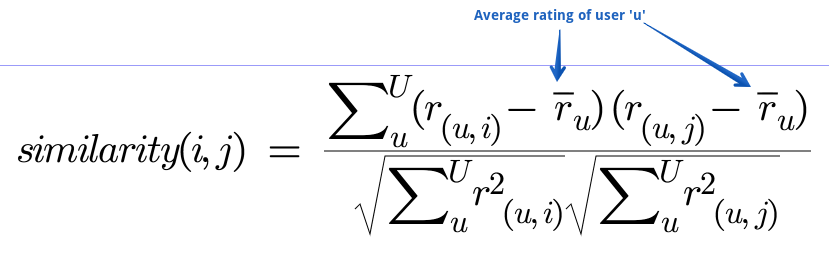
Here is how the above process is depicted in mathematical form.



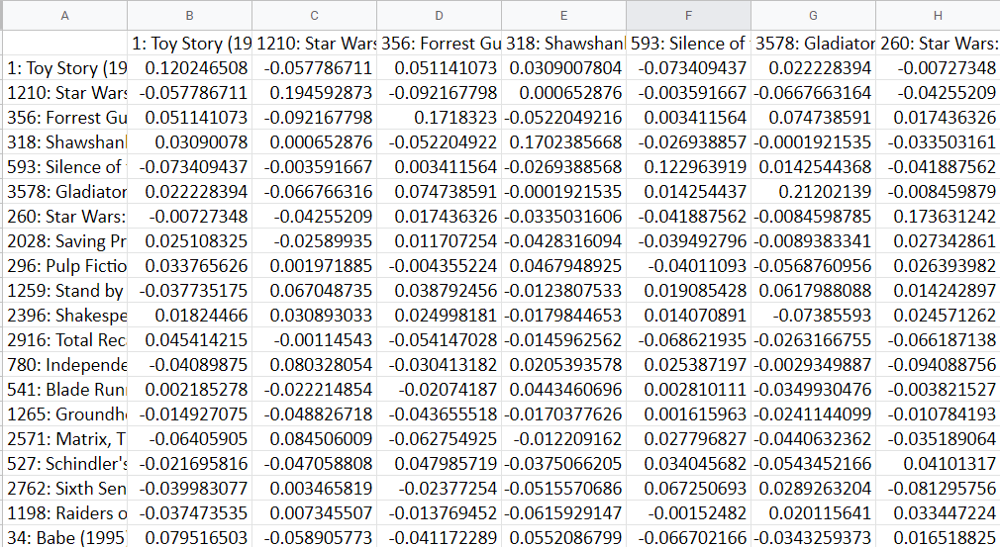
Starting from label 1 (blue on left), above illustrates that in order to calculate the similarity between an item ‘i’ and an item ‘j’ (consider ‘i’ and ‘j’ as movie id 1 and 2) multiple all ratings of item ‘i’ and ‘j’ given by users ‘u’ and sum them. Divided the result with the product of square root of the sum of squared ratings of individual items given by user ‘u’.



Cosine similarity works but it doesn’t take into account the optimistic behavior of users. Different users can rate the same item differently depending upon how optimistic they are. On the scale of 5, one could rate an item 5 while another could rate 3 even though they both very much liked the item. To account for this we have to make a small change to our similarity formula. Here is what it looks like:



Applying adjusted cosine similarity equation on ratings for items will produce a table or matrix that’ll show how similar one item is to another. It should look something like this:



## Let’s start coding up our own Movie recommendation system

In this implementation, when the user searches for a movie we will recommend the top 10 similar movies using our movie recommendation system. We will be using **an item-based collaborative filtering** algorithm for our purpose. The dataset used in this demonstration is taken from [https://grouplens.org/datasets/movielens/](https://grouplens.org/datasets/movielens/%20)

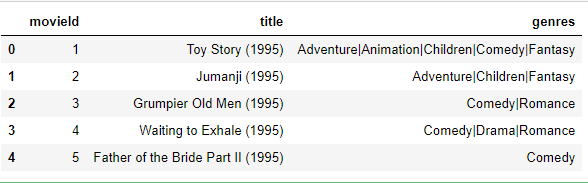
First, we need to import libraries which we’ll be using in our movie recommendation system. Also, we’ll import the dataset by adding the path of the **CSV** files.



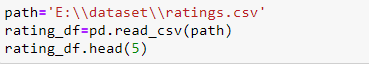


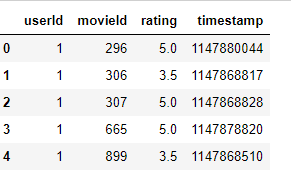
Let’s have a look at the files using the**dataframe.head()**command to print the first 5 rows of the dataset.





Similarly,We will do same for ratings data



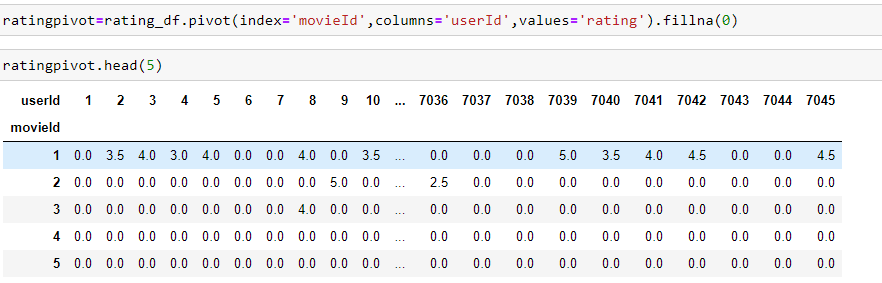


Ratings dataset has-

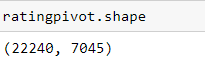
* userId – unique for each user.
* movieId – using this feature, we take the title of the movie from the movies dataset.
* rating – Ratings given by each user to all the movies using this we are going to predict the top 10 similar movies.
* We don’t need timestamp column for recommending movies.

Here, we can see that userId 1 has **watched**movieId 1 & 3 and rated both of them 4.0 but has **not rated**movieId 2 at all. This interpretation is harder to extract from this dataframe. Therefore, to make things easier to understand and work with, we are going to make a new dataframe where each column would represent each unique userId and each row represents each unique movieId.Make a table to visualize data more clearly.

We have to need replace Nan data with 0

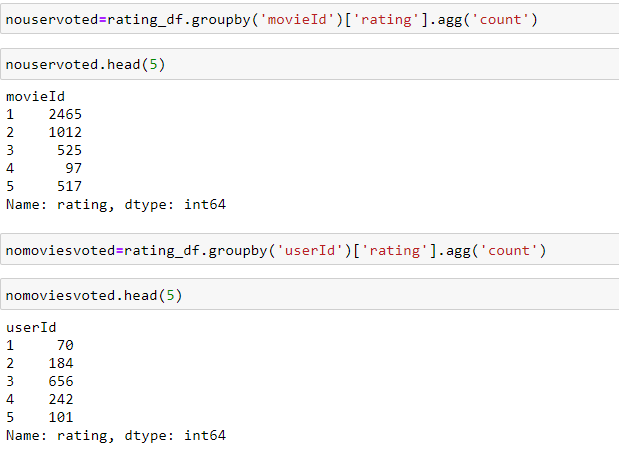


Lets check the shape of our data



We wouldn’t want movies that were rated by a small number of users because it’s not credible enough. Similarly, users who have rated only a handful of movies should also not be taken into account .So its time to filter the data.

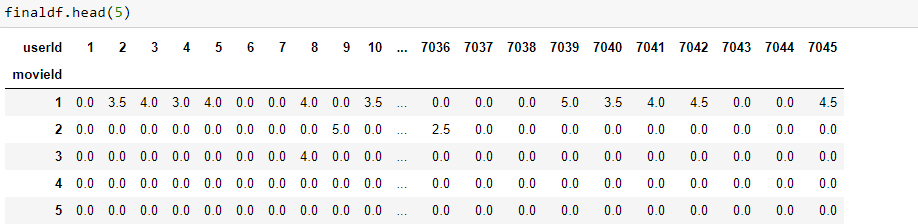
1.Aggregating the number of users who voted and the number of movies that were voted.



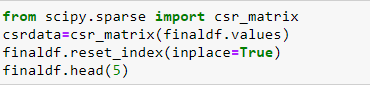
* To qualify a movie, a minimum of **50** users should have voted a movie.
* To qualify a user, a minimum of **20** movies should have voted by the user.

Now its time to filter the final data:



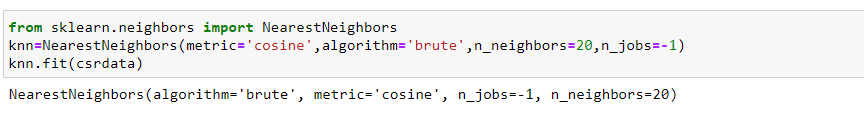


Applying the csr\_matrix method to the dataset :



**Making the movie recommendation system model**

We will be using [the KNN](https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/) algorithm to compute similarity with [cosine distance](https://www.sciencedirect.com/topics/computer-science/cosine-similarity#:~:text=Cosine%20similarity%20measures%20the%20similarity,document%20similarity%20in%20text%20analysis.) metric which is very fast and more preferable than [pearson coefficient.](http://learntech.uwe.ac.uk/da/Default.aspx?pageid=1442" \l ":~:text=Correlation%20is%20a%20technique%20for,association%20between%20the%20two%20variables." \t "_blank)



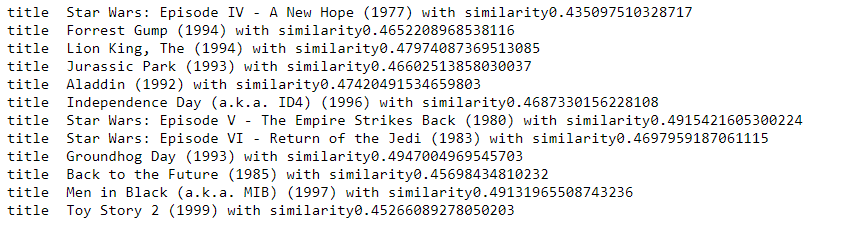
**Making the recommendation function**

The working principle is very simple. We first check if the movie name input is in the database and if it is we use our recommendation system to **find similar movies** and sort them based on their similarity distance and output only the **top 12** movies with their distances from the input movie.



Wow you have done it!!!!

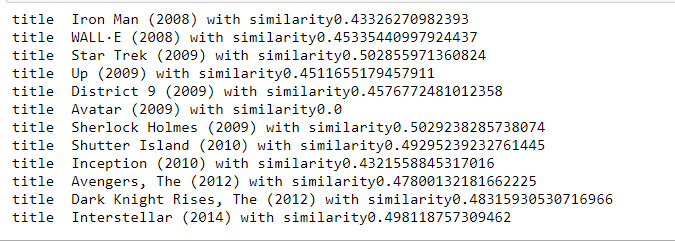
Now it is time to check the recommendation--------



I personally think the results are pretty good. All the movies at the top are superhero or animation movies which are ideal for kids as is the input movie “Toy Man”

Lets check one more





All are action/adventure/kids movies therefore I think the result, in this case, is also good.

Our model works quite well- a movie recommendation system based on user behaviour. Hence, we conclude our collaborative filtering here. So, what are you waiting for? Make your own movie recommendation system and enjoy it !!