**RECOMMENDER SYSTEM**

Recommender systems are the systems that are designed to recommend things to the user based on many different factors. These systems predict the most likely product that the users are most likely to purchase and are of interest to. Companies like [Netflix](https://www.analyticssteps.com/blogs/using-data-handling-and-digital-marketing-maximise-customer-experience-netflix-case-study), Amazon, etc. use recommender systems to help their users to identify the correct product or movies for them.

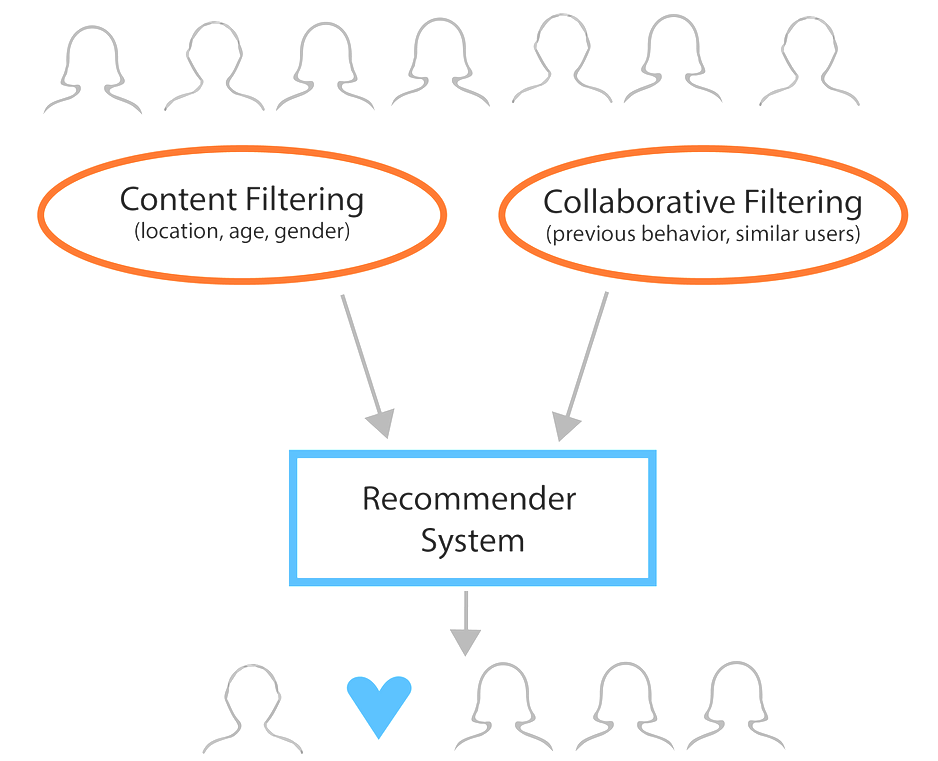
**WHAT IS THE NEED?**

Recommender system has the ability to predict whether a particular user would prefer an item or not based on the user's profile. Recommender systems are beneficial to both service providers and users [3]. They reduce transaction costs of finding and selecting items in an online shopping environment.

**MOVIE RECOMMENDER**

A movie recommendation system, or a movie recommender system, is an [ML-based approach](https://labelyourdata.com/articles/achieving-digital-maturity-with-ai-in-business#machine_learning_and_business_use_cases) to filtering or predicting the users’ film preferences based on their past choices and behaviour. It’s an [advanced filtration mechanism](https://labelyourdata.com/articles/how-to-choose-a-machine-learning-algorithm#supervised_ml_algorithms) that predicts the possible movie choices of the concerned user and their preferences towards a domain-specific item, aka movie.

**TYPES OF MOVIE RECOMMENDER**

****

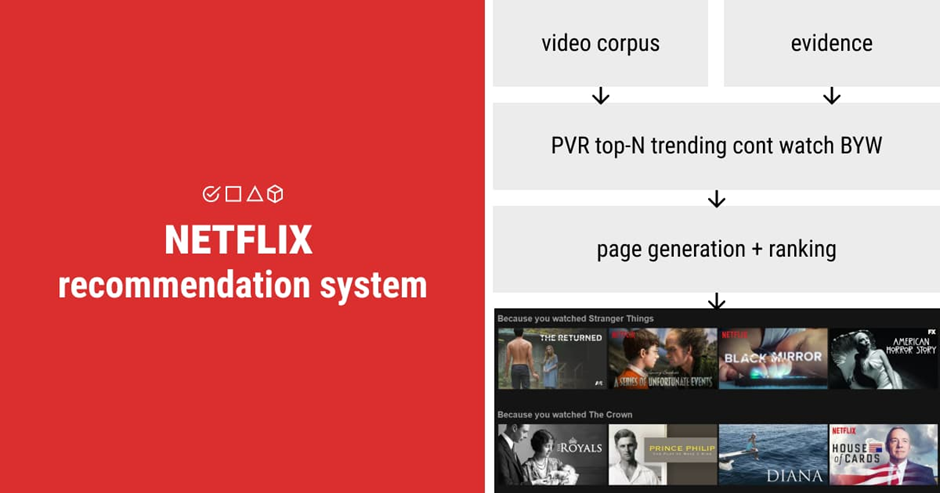
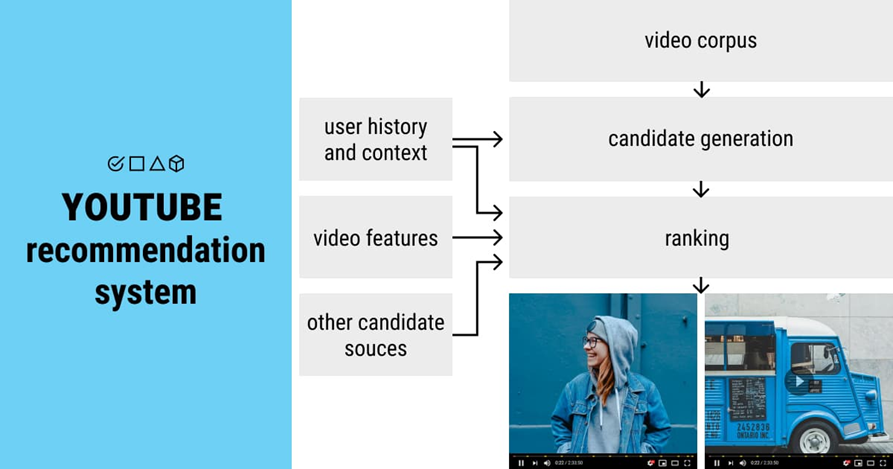
**CONTENT BASED FILTERING**

A filtration strategy for movie recommendation systems, which uses the data provided about the items (movies). This data plays a crucial role here and is extracted from only one user. An ML algorithm used for this strategy recommends motion pictures that are similar to the user’s preferences in the past. Therefore, the similarity in content-based filtering is generated by the data about the past film selections and likes by only one user.

**COLLABORATIVE FILTERING**

As the name suggests, this filtering strategy is based on the combination of the relevant user’s and other users’ behaviours. The system compares and contrasts these behaviours for the most optimal results. It’s a collaboration of the multiple users’ film preferences and behaviours. The core element in this movie recommendation system and the ML algorithm it’s built on is the history of all users in the database. Basically, collaborative filtering is based on the interaction of all users in the system with the items (movies).

**EXAMPLES**

** **

**Datasets**

We have two movie datasets fetched form Kaggle.com.

We have 2 datasets:(of 5000 movies)

* TMDB (the movie database) 5000 Movie datasets

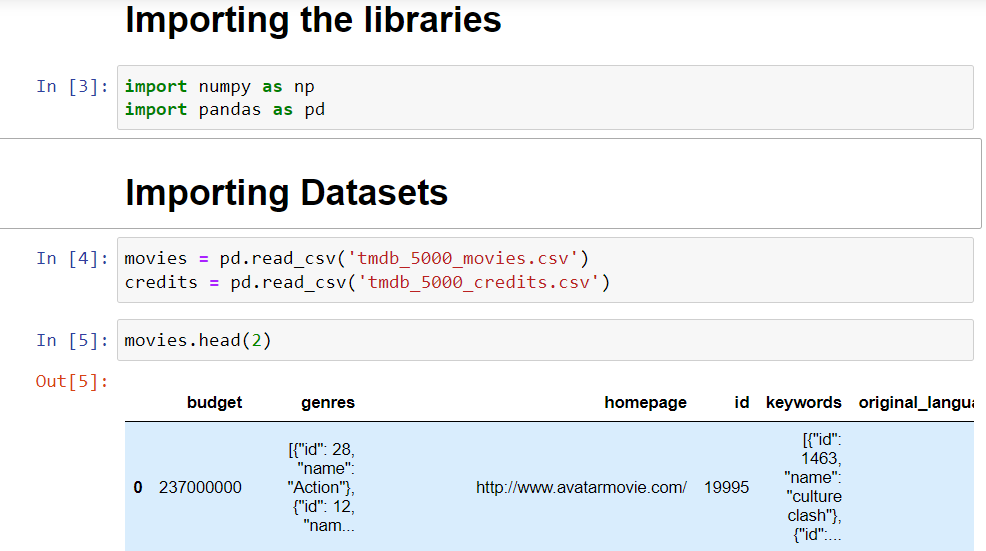
We have data related to budget, genres, popularity, etc.

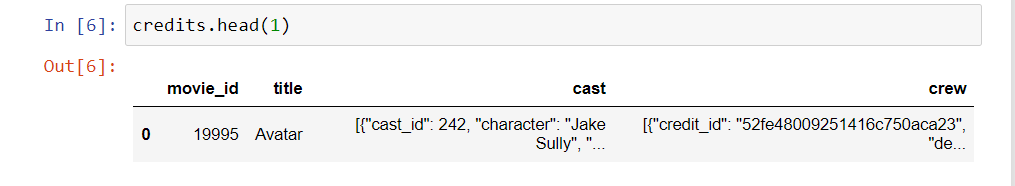
* TMDB 5000 credits datasets

We have data related to cast and crew.

**Importing the requirements of Project**

Firstly, we will import our libraries and datasets in our Jupyter Notebook.



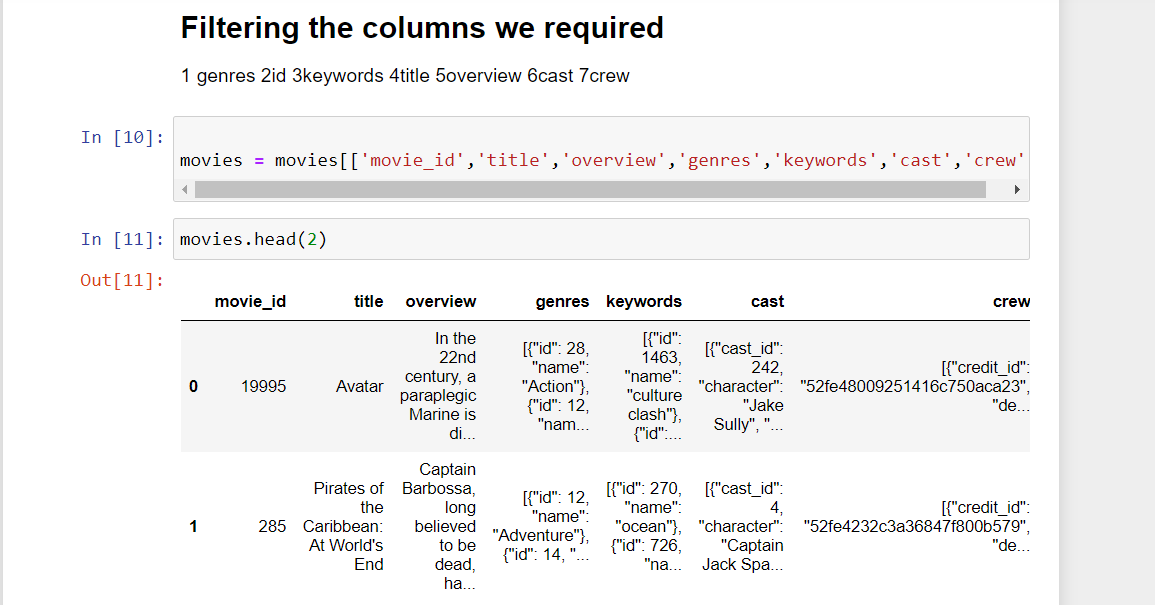


**Data-Preprocessing**

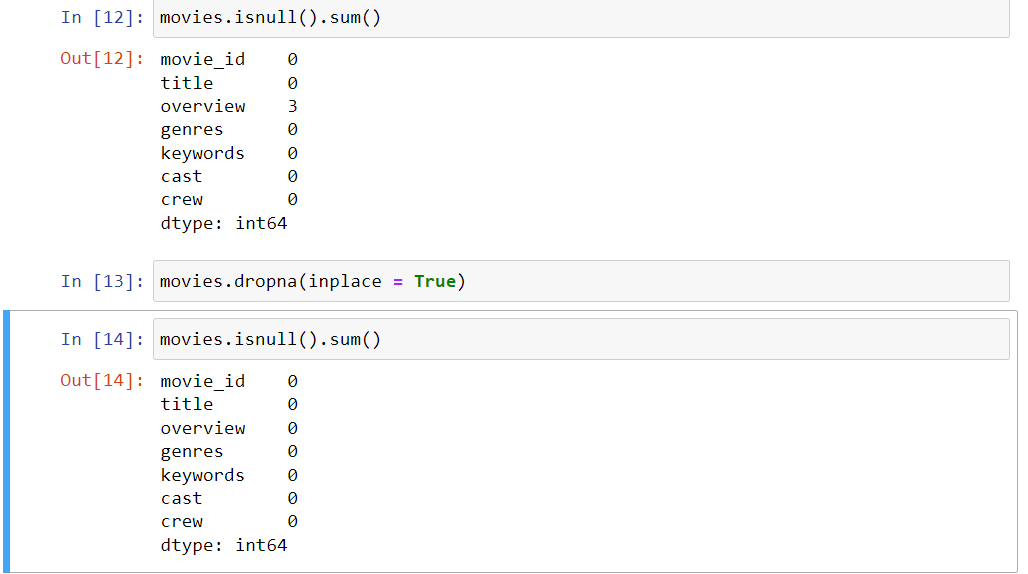
As we have two datasets so we will merge them in one dataset.

****

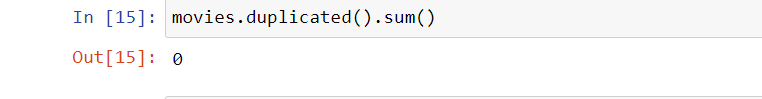
Further, there are too many columns out of which there are only few which we require so we will filter out the columns we require. These columns are Movie\_id, title, overview, genres, keywords, cast and crew.



Now we will check for the null values (missing data), and as there are only three missing data in overview so we will drop those three.



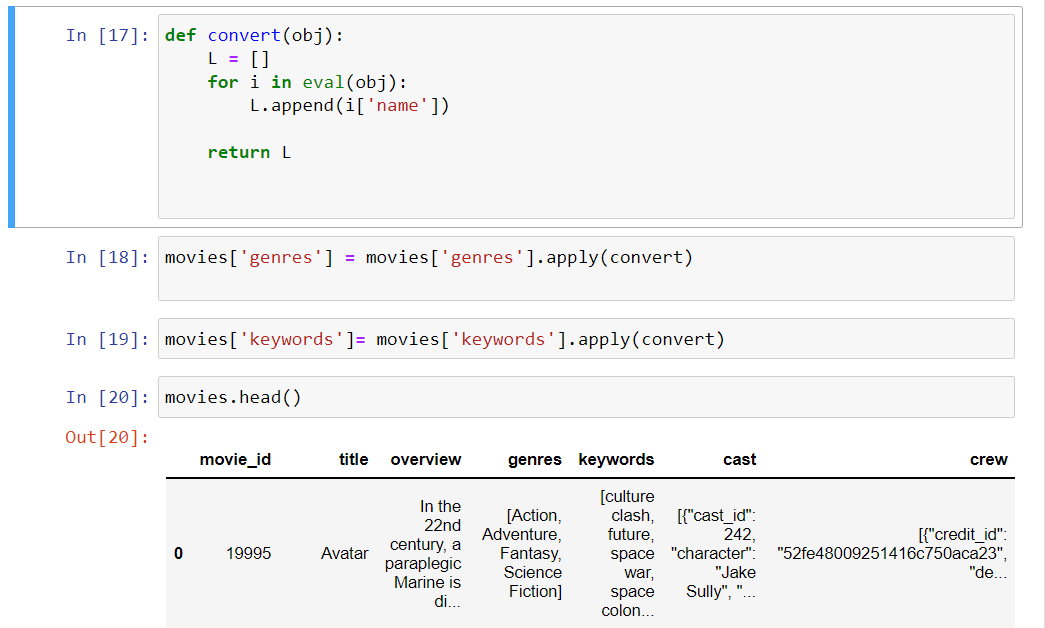
Further we will check for duplicate data.



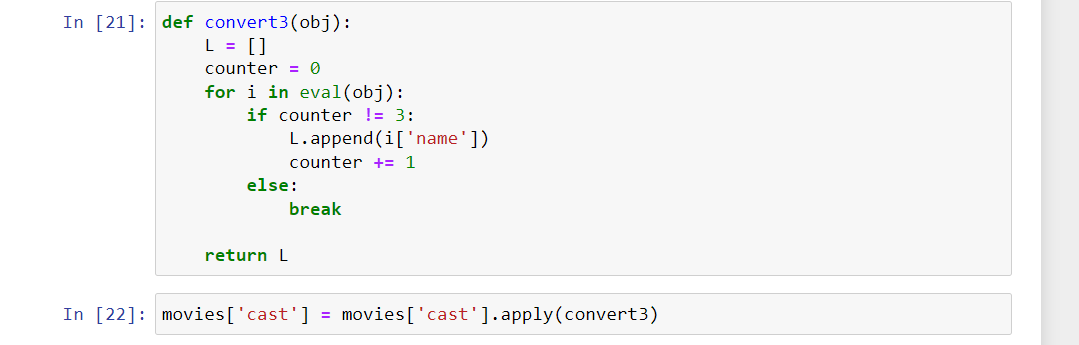
There is not a single duplicated data.

Now we will bring our columns in proper format one by one starting from genres. Genres is in a very typical format, it is a list of dictionaries, we will turn it in a simple list.

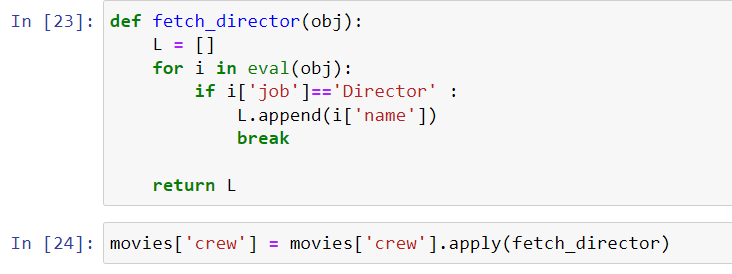
Same situation is with keyword column so we will apply same function to it.



Now we will move to cast column, here we don’t need all the cast members as only top actors are the one who are focussed by the viewers, so we are only taking top-3 cast members. In cast column we only require the real name of cast not its id or name in the movie so will make our function on the basis of that.

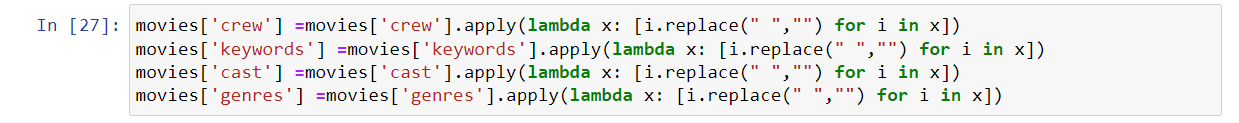


In crew column, we only require director-name so we will fetch director and eliminate all the other crew members.

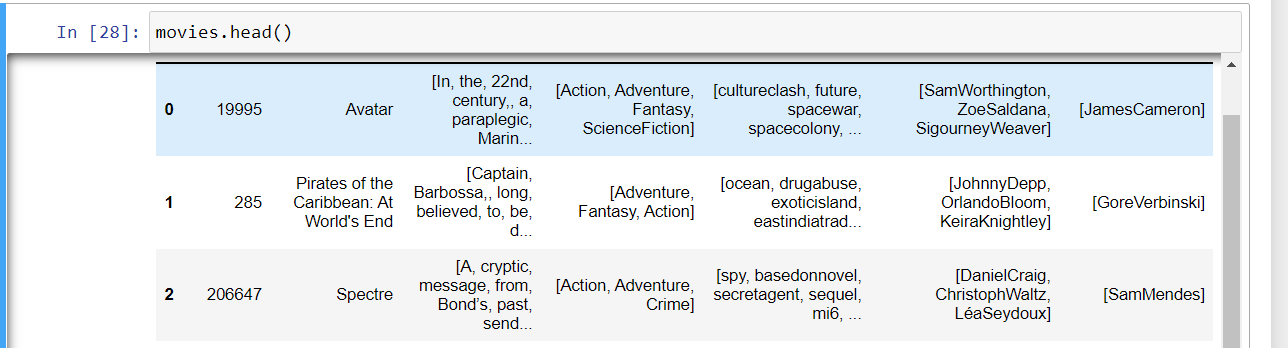


The overview column is in form of string to concatenate with others we need to turn it into the list. Why we will concatenate, we will understand it further. 

There is a transformation required in our datasets, suppose there are two names “John Smith” and “John Doe” our recommender system will take time and may treat them as same, so we need to remove the space which will make them totally different entities.



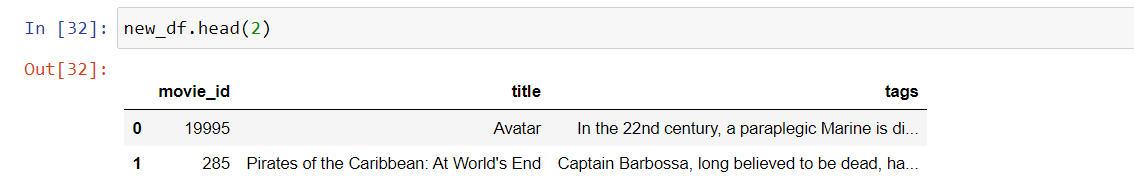
Below you can see all the codes implemented to our dataset to make it clean enough to work upon.



Now, we will concatenate all the columns in tags column and make a different dataframe having movie\_id, title and tags column.







In tags column we will change their format from list to strings.

We will also change their form into lower form for easy comparisons.



**Vectorization**

We have data in form of movie\_id, name and tags. Tags are in text form so we have to represent these tags in dimensional forms so we have to make vectors of these tags. To make vectors we will use Bag of Words technique.

BAG OF WORDS

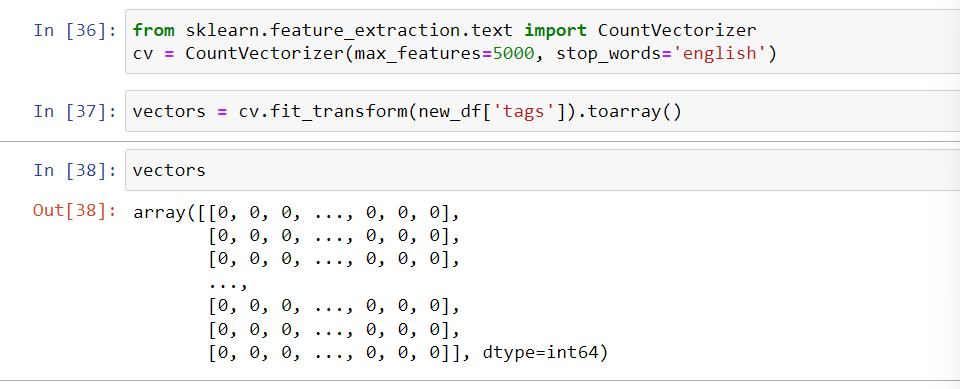
The **bag-of-words model** is a simplifying representation used in [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing) and [information retrieval](https://en.wikipedia.org/wiki/Information_retrieval) (IR). In this model, a text (such as a sentence or a document) is represented as the [bag (multiset)](https://en.wikipedia.org/wiki/Multiset) of its words, disregarding grammar and even word order but keeping [multiplicity](https://en.wikipedia.org/wiki/Multiplicity_(mathematics)). The bag-of-words model has also been [used for computer vision](https://en.wikipedia.org/wiki/Bag-of-words_model_in_computer_vision).[[1]](https://en.wikipedia.org/wiki/Bag-of-words_model#cite_note-sivic-1)

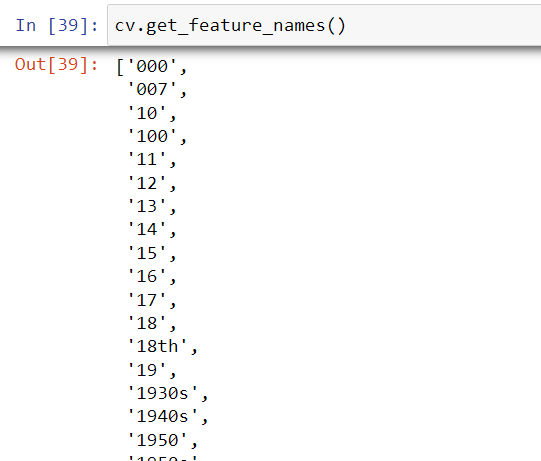
The bag-of-words model is commonly used in methods of [document classification](https://en.wikipedia.org/wiki/Document_classification) where the (frequency of) occurrence of each word is used as a [feature](https://en.wikipedia.org/wiki/Feature_(machine_learning)) for training a [classifier](https://en.wikipedia.org/wiki/Statistical_classification).[[2]](https://en.wikipedia.org/wiki/Bag-of-words_model#cite_note-2)

An early reference to "bag of words" in a linguistic context can be found in [Zellig Harris](https://en.wikipedia.org/wiki/Zellig_Harris" \o "Zellig Harris)'s 1954 article on *Distributional Structure*.[[3]](https://en.wikipedia.org/wiki/Bag-of-words_model#cite_note-3)

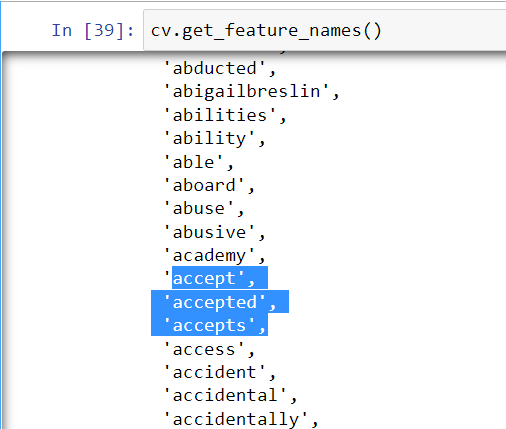
The Bag-of-words model is one example of a [Vector space model](https://en.wikipedia.org/wiki/Vector_space_model).

During vectorization we will not use Stop words (like a, an, the, of, from, etc) as these words doesn’t give any special meaning to sentences and thus, they won’t be helping in recommendations.





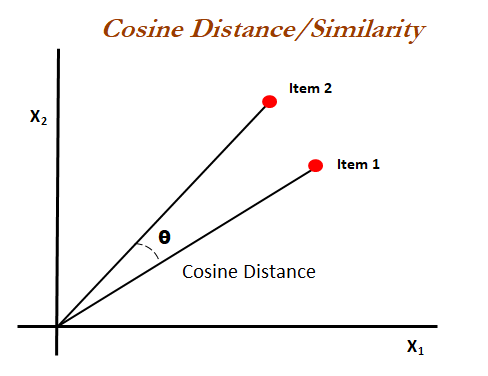
It might feel that numeric terms are useless but we are still considering them because some terms like 1980s might be depicting a different era, meaning to say that some of them might hold a great value so we are not doing anything to them.

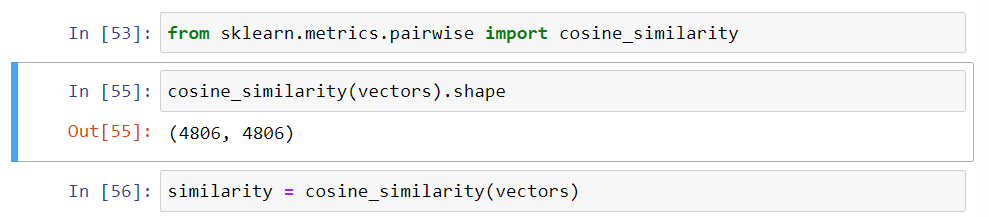


As you can see there are some words like accept and accepted which have same meaning, so we shouldn’t be making different attributes for the similar purpose. We will apply stemming to our “tags” to solve these issues. Stemming is the process of reducing a word to its stem that affixes to suffixes and prefixes or to the roots of words known as "lemmas".



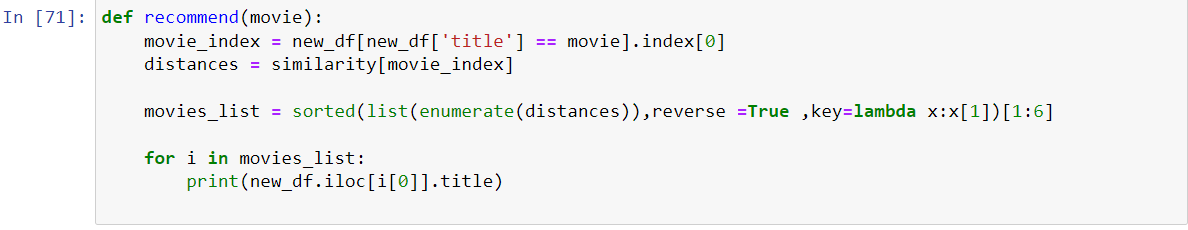


Now, further we will calculate the distance between all vectors through Cosine distance (the angle between two vectors). Distance is inversely proportional to similarity. The farther the two vectors are the less similarity is there and closer the vectors that is lesser the theta, vectors are more similar.

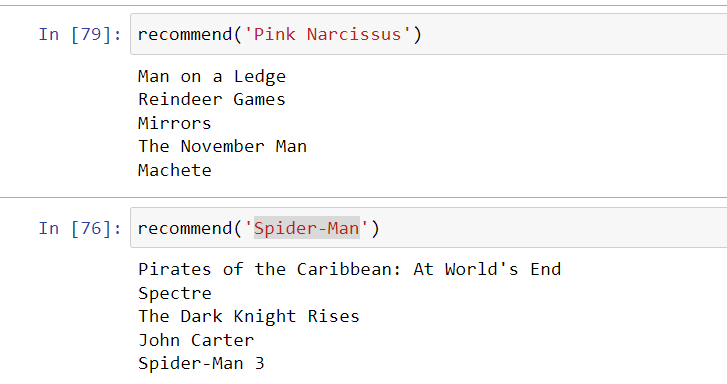


Now we will make our recommend function in which first we will sort all the distances between vectors and out of which we will select the first 5 movies which are closest to out selected vector(movie).

Then we will fetch the name of movie through iloc method.



So, now our recommendation system is ready.



Names of some movie

|  |
| --- |
| George Washington |
| Smiling Fish & Goat On Fire |
| Dawn of the Crescent Moon |
| Raymond Did It |
| The Last Waltz |
| Run, Hide, Die |
| The Exploding Girl |
| The Legend of God's Gun |
| Mutual Appreciation |
| Her Cry: La Llorona Investigation |
| Down Terrace |
| Clerks |
| Pink Narcissus |
| Funny Ha Ha |
| In the Company of Men |
| Manito |
| Rampage |
| Slacker |
| Dutch Kills |
| Dry Spell |
| Flywheel |
| Backmask |
| The Puffy Chair |
| Stories of Our Lives |
| Breaking Upwards |
| All Superheroes Must Die |
| Pink Flamingos |
| Clean |
| The Circle |
| Tin Can Man |
| Cure |
| On The Downlow |
| Sanctuary: Quite a Conundrum |
| Bang |
| Primer |
| Cavite |
| El Mariachi |
| Newlyweds |
| Signed, Sealed, Delivered |
| Shanghai Calling |
| My Date with Drew |