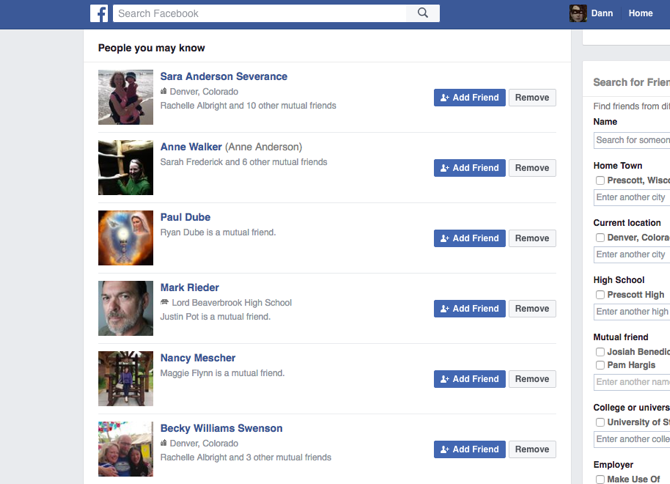
**Content Based Recommender System – A Beginners Guide**

**Introduction:**

So, I will start with an example. Right, so we all have been using YouTube, Facebook, LinkedIn and many other sites. What’s so common about these sites? If you guess it, then you might be aware. But let me tell you. They all use a Recommender system. Yes, you heard it right. Isn’t it amazing that we have been using this concept for a while now and we were not even aware. Now, you might have a question as to how? So, to answer that, let me give you an example.

Facebook: You might have seen on sight Facebook suggesting more people using “People You May Know” Section.



Similarly, LinkedIn goes on to suggest you people that you might want to connect to and YouTube shows the recommended list of videos based on your browsing history. All of these use recommender systems. So far so good. Brace yourself, we are building up to some amazing behind the scenes about these recommender systems.

Well, to be honest, most of the people might already be aware about this feature or may be not. But I am pretty sure that many might not know the logic and algorithm that are being used behind these systems. ‘Recommender Systems’ is the algorithm that is used behind all these suggestions that you get on these channels. So, if you ask what does a Recommender Systems does? Well, they are like those friends who go on to suggest some stuff they liked. Not exactly this but its something, that recommends a personalised content based on what the user’s past interaction and preferences. Broadly there are two kinds of Recommender systems, Content based and Collaborative Filtering. Well there is one more, Popularity Based recommender systems, but to be honest it’s a very outdated method of suggesting. Because Let’s say some movie was popular and was a good hit, you might end up disliking it because you have a whole different set of taste. So, in this article I will move ahead with Content Based Recommender Systems.

But before we go ahead with this, let’s understand what’s an item and what is Attribute.

**Item:** Think of item as a parent, whose traits or say attributes are being inherited and used in Recommender Systems. These could be like a movie, book, songs, etc.

**Attribute:**  It is a trait or characteristic of an item. For example, a movie tag, a song tag, words in a document and the list can go on.

**What are Recommender Systems/Content Recommender Systems?**

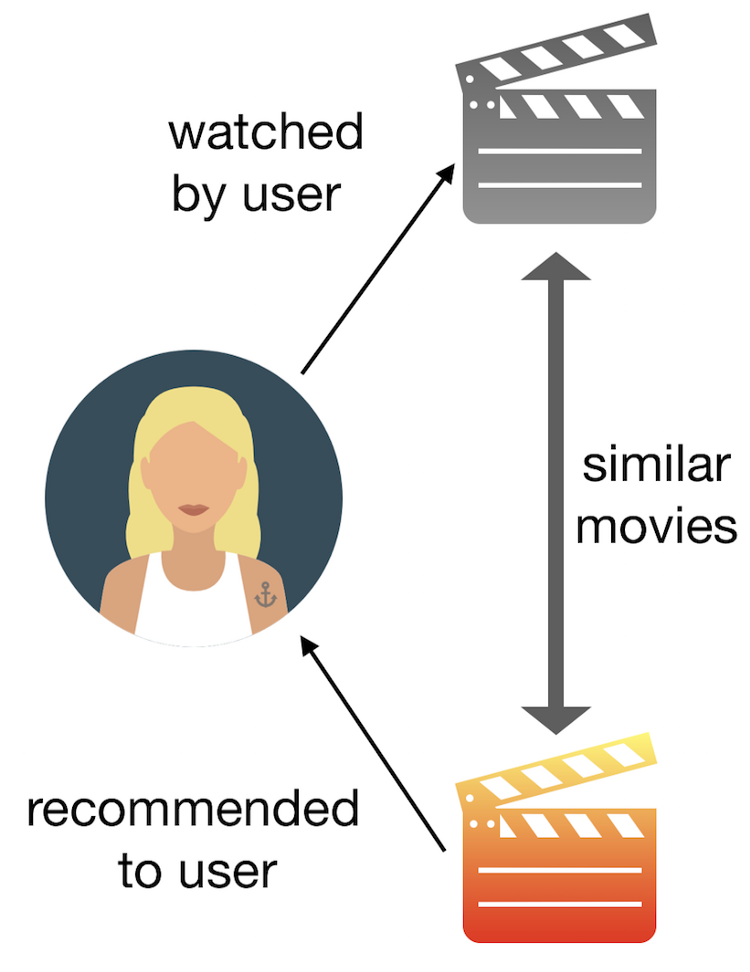
According to Wikipedia, “A recommender system, or a recommendation system (sometimes replacing 'system' with a synonym such as platform or engine), is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item. They are primarily used in commercial applications.”

So again, what did we get from Recommender Systems definition? Well, it can be thought of as an active information filtering system, which personalises the information coming to a used based on his interest, past interaction and preferences etc. These systems are widely used in recommending movies, articles, restaurants etc. We can find use of such system in many places like IMDB, Gaana, Zomato, Amazon, Flipkart and the list never ends.

Let’s move ahead!!

**How do Content Based recommender Systems Work?**

A content-based recommender system is the one that works with the data provided by user either explicitly (you can say ratings) or implicitly (let’s say clicking on links). So, based on these data, the system builds a user profile which in turn is used to make suggestions. The system grows smarter with the amount of user input received. I guess you didn’t get the last part. Well, for example, I watched some series, say crime genre, the system remembers that I might like crime genre and it recommends. The more I watch, the systems gets more information about my preferences, the more accurate the system will get. As simple as this. We all know the proverb like practice makes us stronger, it’s the same but in different context. Here, the proverb you can create is, “ The more the data, the merrier/accurate the system will be.”

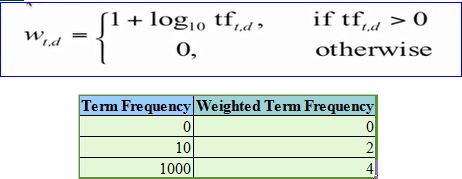


Let’s move ahead into more of mathematical and technical part.

**What are the concepts used in Content Based Recommenders?**

The concept of Term Frequency (TF) and Inverse Document Frequency (IDF) are used in content-based systems. As described on Wikipedia “In information retrieval, tf–idf or TF-IDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in searches of information retrieval, text mining, and user modelling.” So, by the definition itself, its quite clear that these are used to determine the relative importance of a document/news/movie/song etc.

TF is nothing but the frequency of an individual word occurring in a document. IDF is the inverse of the document frequency. The main agenda of using tf-idf revolves around two reasons: Suppose we search “the evolution of ML” and look at words like “the” and “ML”. We know that “the” word will occur a number of times, even while creating this content, the is occurring multiple times while “ML” will occur a few times not regular but the importance of “ML” will be higher as compared to “the”. Also, tf-idf neglects the effect of highly occurring words by using log. Log dampens the effect of highly occurring words.



See the magic here. The weighted term frequency is now more comparable than the term frequency because of the dampening effect of log.

Now that we have calculated tf-idf, how do we determine which item is closer to each other, rather than closer to user profile? We are going to find out next with Vector Space Model.

**How does Vector Space Model works?**

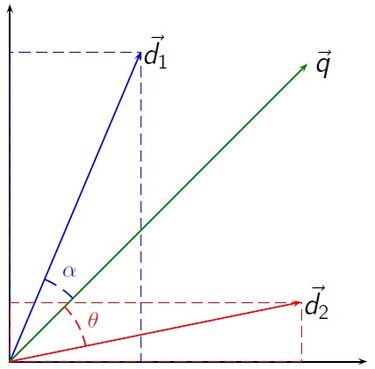
Let’s start with Wikipedia definition, “**Vector space model** or term vector model is an algebraic model for representing text documents (and any objects, in general) as [vectors](https://en.wikipedia.org/wiki/Vector_space) of identifiers, such as, for example, index terms. It is used in [information filtering](https://en.wikipedia.org/wiki/Information_filtering), [information retrieval](https://en.wikipedia.org/wiki/Information_retrieval), [indexing](https://en.wikipedia.org/wiki/Index_(search_engine)) and relevancy rankings.”

In this model, each item is stored as a vector of its attribute in the n-dimensional space. Okay, I might have got you confused with all these terminologies here, but have patience, it will all fall into places. Keep reading on. I am trying to make this as simplistic as possible.

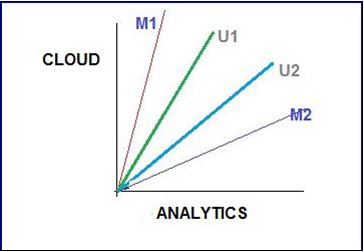
So, what we do here is calculate the angle between the vectors to determine the similarity among vectors. We are getting there. Just hold on.

Next, the user profile vectors are also created based on his past interaction and preferences. Afterwards, we try to create a similarity between an item and a user.

Here is a Wikipedia image to represent what I said earlier.



Let’s understand above with an example.

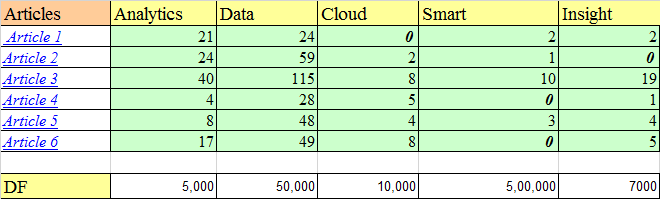


Above image depicts 2-D representation of a two attribute, which are Cloud and Analytics. So, M1 and M2 are documents and U1 and U2 are users. From the diagram its quite clear that M2 is more inclined towards Analytics and M1 towards Cloud.

Coming to the interesting part as in how the relative importance comes into picture here. From the diagram, its safe to say that U1 is more inclined towards M1 and in turn likes articles on “Cloud” more than he likes the article on “Analytics” and vice versa. So, we calculate whether user likes/dislikes a particular article is by taking cosine of angle between the user profile vector (*Ui*) and the document Vector (*di*). The reason behind choosing cosine is very simple. The **value of cosine will increase with decreasing value of angle.** Well, I am pretty sure you all might have know this from the trigonometry. If not, well, you might need a trigonometry lesson. Just kidding.

**Calculating TF-IDF:**

Slowly we are moving towards our final goal. Let’s work with an example and see how the recommender systems works. Let us assume that we search some technology and top five links that appears contains the following word distribution.



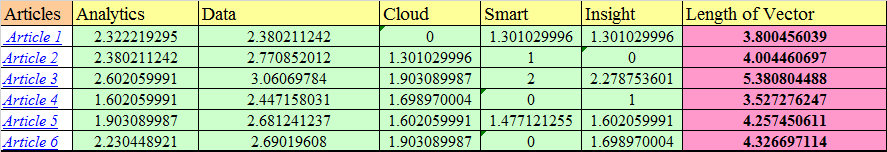
Okay, let me describe the above picture. In above dataset, it says analytics word appeared 5000 times, similarly data appeared 50000 times and so forth. Let us assume that the total count of the corpus it contained is 1 million (10^6).

**Term Frequency:**

As discussed earlier, log dampens the effect of higher frequency and that is what we are going to use as a formula for Term Frequency.

**TF = 1+log1021**

So, using above formula, we calculate TF for every attribute of an individual item.

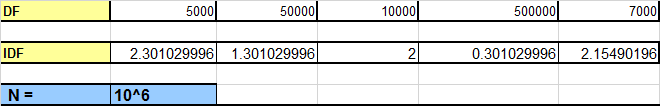


**Inverse Document Frequency:**

IDF is calculated by taking logarithmic inverse of document frequency among the whole corpus. For example, as said, we had 10^6 documents.

**IDF = log10(Total Corpus/DF)**

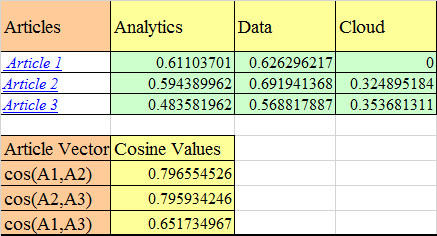
So, for every attribute, we calculate IDF = log10(10^6/DF).



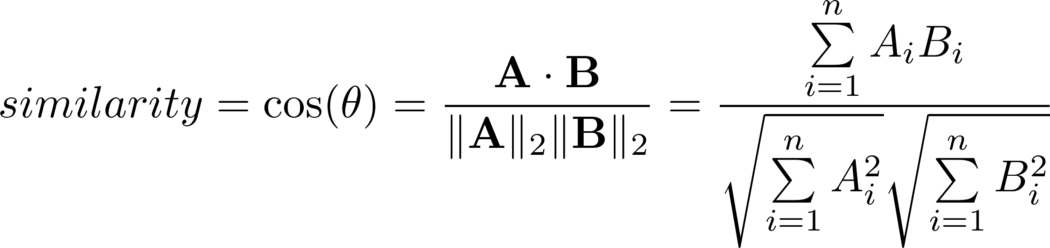
Did you guys observe something? I guess no. Okay, I will break this down to you. Smart word appeared 500000 times and observe the IDF it has got. It’s the lowest. Why, because Smart is not a relevant word in context of this. See the beauty. The word with low relevance is assigned lowest IDF. This was something amazing. Also, we calculated Length of vector using square root of sum of squared values of each attribute in the vector. So, what’s the use of this Length of vector we just calculated. That’s what I am going to show you next. It is useful in context when you want to normalise the vectors.

So, each term vector is divided by length of vector in order to get the normalised vector. For Article 1, word Analytics, normalised vector becomes 2.322/3.8 = 0.611. When you work this out for each vector, the table will look something like below and sum of all will be a normalised length of 1.

One last part into this since we calculated the normalised vector. Let’s use cosine to find the similarity. Right.



Let me first tell you about the dot product.



Above is the formula used to calculate the dot product for our vector. From above, we can conclude that Article 1 and 2 are both most similar and hence it appears at the top two position of search result.

References:

<https://en.wikipedia.org/wiki/Vector_space_model>

<https://en.wikipedia.org/wiki/Recommender_system>

<http://towardsdatascience.com/>

<https://www.analyticsvidhya.com/>