**Natural Language Processing Task**

**Main Task:** Movie Recommender system

**Dataset Used:** The Movies dataset.

**Download Link:** <https://www.kaggle.com/rounakbanik/the-movies-dataset>

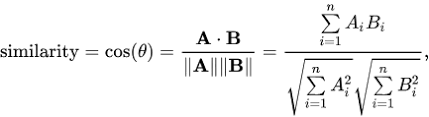
**How NLP is used in this Model:**

1. Contents of title, overview and genre were concatenated for each movie and made another column combined feature column to find similarity between the movies.
2. Extra character other than a-z and A-Z and space were removed from combined features.
3. All uppercase letters were converted into lowercase letters.
4. Applied stemming (All unnecessary words were removed from the combined features.
5. Cosine Similarity and TfidfVectorizer are used to find similarity between the movies combined feature.
6. Using sparse matrix similarity were found.
7. On the basis of similarity and another parameters Movie was Recommended by model.

**Steps involved in building the used models:**

1. Dataset Cleaning:
2. metadata.csv file have some faulty data at 19730, 29503, 35587 which was removed.
3. metadata.csv file had 45463 movies data which was reduced in small dataset having 9099 using small\_link.csv.
4. Extra character other than a-z and A-Z and space were removed from combined features.
5. All uppercase letters were converted into lowercase letters.
6. Applied stemming (All unnecessary words were removed from the combined features.
7. Contents of Title, Overview, Genre of metadata.csv file was concatenated for each movie and made a combined\_feature column to find similarity among the movies.
8. Ratings was calculated for each movieId using rating.csv and ratings\_small.csv.
9. Find Similarity between the movies:

TfidfVectorizer:



1. **Models Implemented:**

Following models are implemented:

Model 1

Content Based Recommenders based only on Movie Overviews and Genre.

Recommendation on the basis of similarity sparse matrix. It was not considering the Rating and popularity of the movie. So, I implemented model 2.

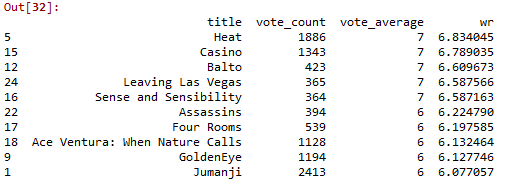
Model 2

Content Based Recommenders based on Movie Overviews and Genre and add a mechanism to remove bad movies and return movies which are popular and have had a good critical response. It will take the top 25 movies based on similarity scores and calculate the vote of the 60th percentile movie. Then, using this as the value of m, we will calculate the weighted rating of each movie using below formula. It used vote\_count and vote\_average for prediction of the movie.



where,

* *v* is the number of votes for the movie
* *m* is the minimum votes required to be listed in the chart
* *R* is the average rating of the movie
* *C* is the mean vote across the whole report



**Model 3:**

Surprise was used to calculate prediction and uses ratings for evaluation of the model.

Recall@k = (# of recommended items @k that are relevant) / (total # of relevant items)

Precision@k = (# of recommended items @k that are relevant) / (# of recommended items @k)

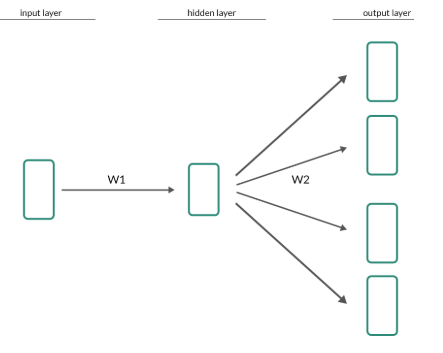
**Model 4:**

Recommendation based on the user’s interest using wrd2vec

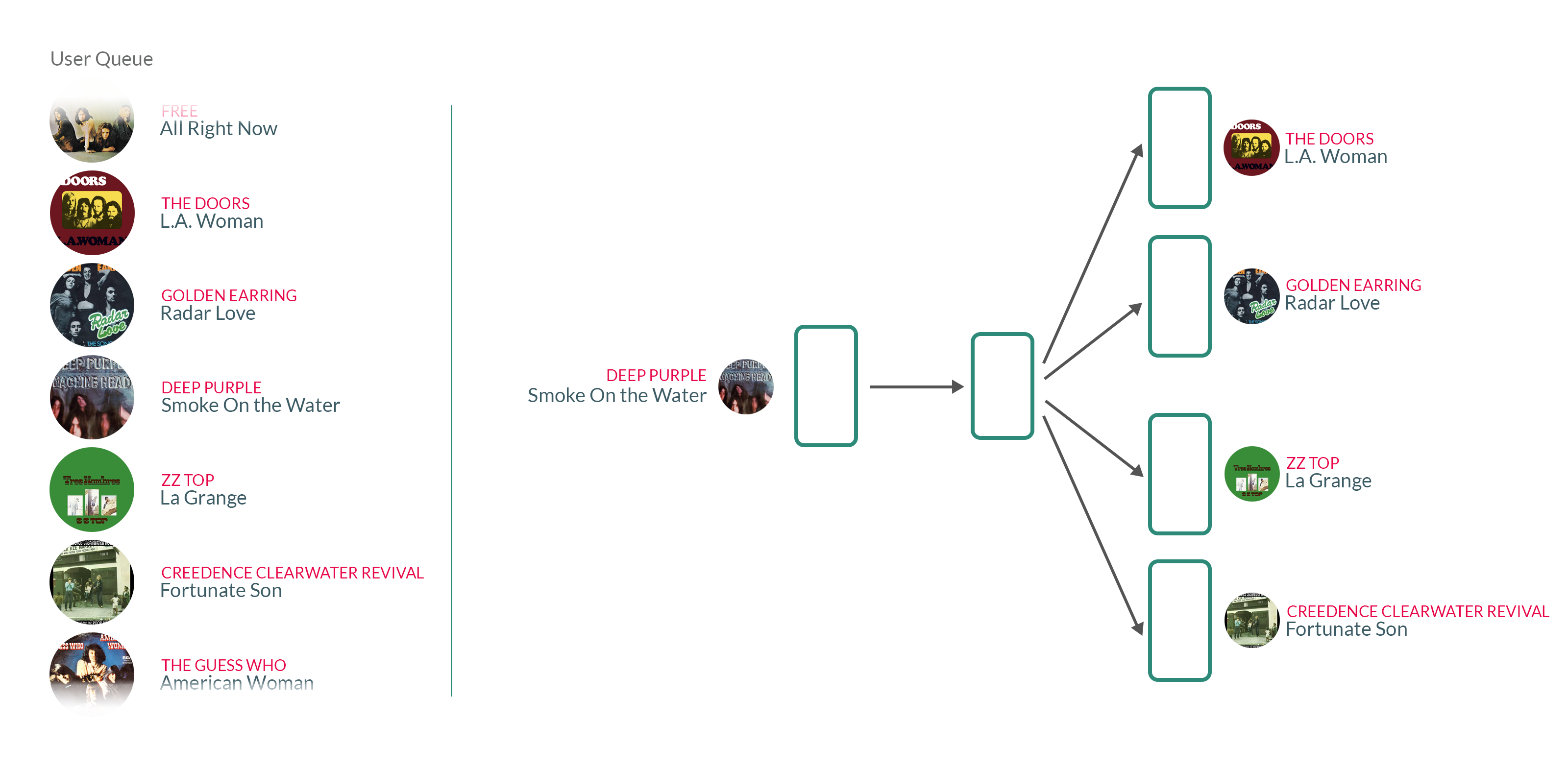
Word2vec is a class of neural network models that are highly useful for Natural Language Processing tasks. The neural network takes in a large corpus of text, analyses it, and for each word in the vocabulary, generates a vector of numbers that represent that word. Those vectors of numbers encode important information about the meaning of the word in relation to the context in which it appears.

There are two main models defined:

1. **Continuous Bag-of-Words** model
2. **Skip-gram** model: shallow neural network with a single hidden layer that takes in a word as input and tries to predict the context of words around it as output.



**W1** and **W2** represent weight matrices that control the weights of the successive transformation. Below Figure shows how it works.



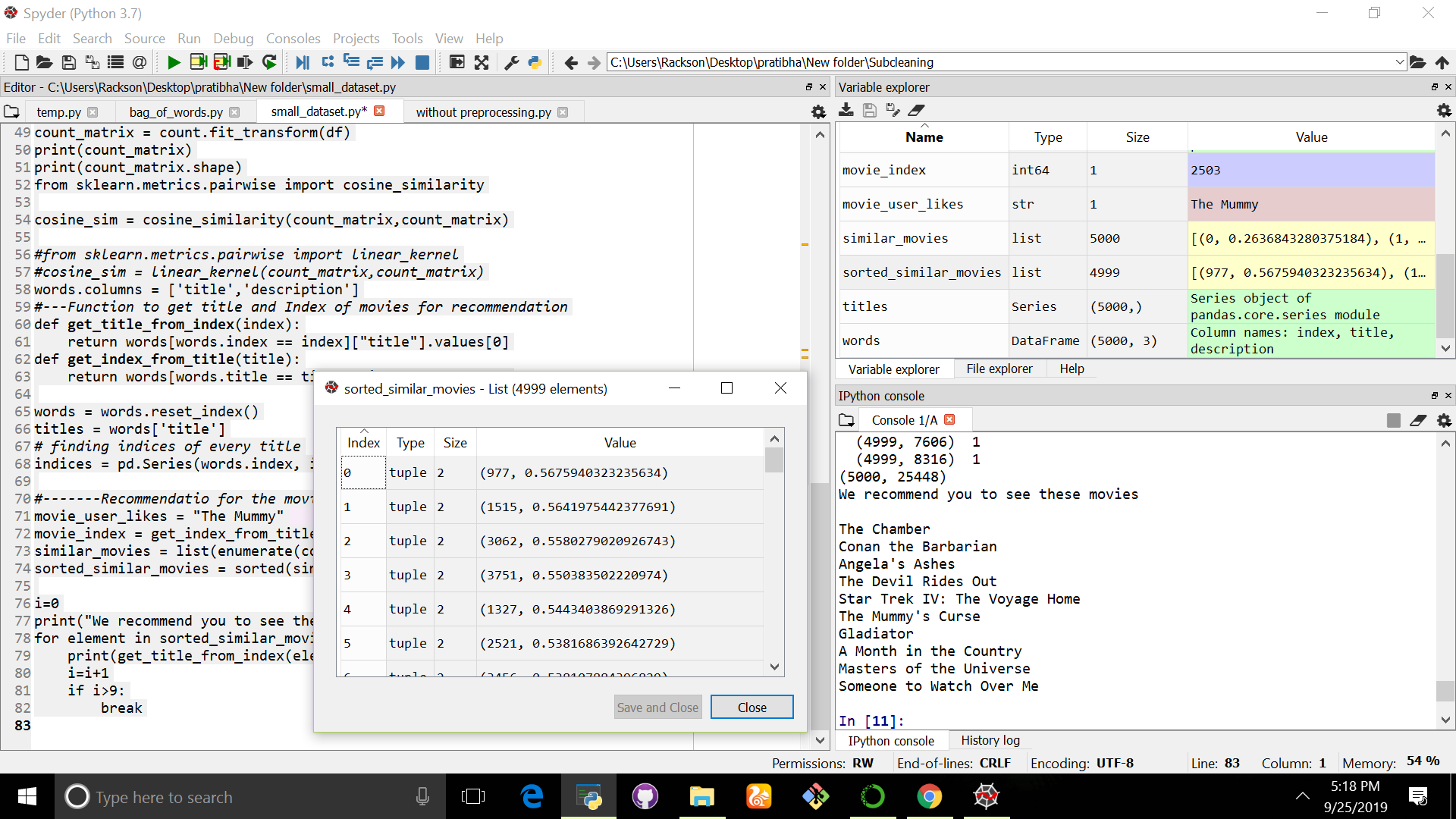
**Observation/ Result:**

Confidence Score upto 73.5%

As we increase no. Of movies the top 10 suggested movies remained unchanged after 8000 movies included. This may be faulty or maybe due to the fact that out of all 45000 movies the most similar movies are amongst top 5k-6k.

Model 3 was best suited for content-based Recommendation according to movie.

Model 4 was best suited for content based Recommendation according to user.



**Refrences:**

1. <https://www.analyticsvidhya.com/blog/2019/07/how-to-build-recommendation-system-word2vec-python/>
2. <https://towardsdatascience.com/using-word2vec-for-music-recommendations-bb9649ac2484>
3. <https://www.kaggle.com/fabiendaniel/film-recommendation-engine>
4. <https://www.kaggle.com/ibtesama/getting-started-with-a-movie-recommendation-system>
5. <https://www.kaggle.com/fabiendaniel/film-recommendation-engine>
6. <http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/>
7. <http://sudiptakar.info/learning-from-mistakes1-memory-error-in-tfidfvectorizer/>
8. <https://www.tensorflow.org/tutorials/representation/word2vec>
9. <https://medium.com/scaleabout/a-gentle-introduction-to-doc2vec-db3e8c0cce5e>
10. <https://www.kaggle.com/rounakbanik/the-movies-dataset/kernels>
11. <https://www.kaggle.com/gspmoreira/recommender-systems-in-python-101>
12. <https://towardsdatascience.com/how-to-build-a-simple-recommender-system-in-python-375093c3fb7d>
13. <https://towardsdatascience.com/evaluation-metrics-for-recommender-systems-df56c6611093>
14. <https://surprise.readthedocs.io/en/latest/FAQ.html#how-to-compute-precision-k-and-recall-k>
15. <https://surprise.readthedocs.io/en/latest/prediction_algorithms.html>

etc.