Sentence Similarity using FastText and Cosine Similarity.

Importing Csv to Python.

Csv files are imported into python using panda and text columns are stored as series of strings.

Tokenization of Text.

After importing texts, sentences are converted into arrays of strings by removing special characters and non-letter characters. Stop words are also removed from sentences.

Corpus.

After tokenizing the words, all sentence form text1 and text2 are added to prepare sample output for our vectorization model.

FastText.

Now our corpus is fed into genism FastText model with default parameter and taking min_count=1 (minimum count of word in the document to appear in FastText). The fast model takes input as n-grams (example- if trigram is selected then joke is divided as <jo,oke,ke>). For minimizing data, the n-grams are converted into hash dictionary and each n-gram is represented by an integer between 1 and B (B typically range in millions). A given word vector is calculated by summing all its n-grams.

The default recapping method is CBOW (Continuous Bag of Word Model) where center word is predicted by giving neighboring words. The vectorization gives a 100-dimension vector for each word in our corpus.

Document Vectorization.

Each sentence is converted into dictionary with words as keys and count of word in the given sentence as values. The mean vector is calculated for all the words in the sentence and it is used to represent the document.

$$\text{Centroid} = \frac{\sum_{i=1}^{n} N * W_i}{\sum_{i=1}^{n} N}$$

Where N = count of word, W_i

W_i = Vector of i_{th} word in sentence

n = number of different words

Then cosine similarity for each sentence pair is calculated as:

$$\text{Cosine Similarity} = \frac{u.v}{||u|| * ||v||}$$

Where u and v are two vectors.