**ASSIGNMENT 2**

**VECTORIZATION**

**Introduction:**

There exists vast amount of data on social media platforms on virtually any topic today. Different posts capture different sentiments, or a single post can get mixed reviews. This data in the form of tweets from Twitter can be almost impossible for computers to analyze as a human would do based on the context of the words, way of saying them. To be able to use this huge data on Artificial Intelligence, we need to perform tokenization, vectorization and use some techniques to reduce the vocabulary list to better make out of the sentiment.

**Data Description:**

To collect the data, I have used tags API to collect tweets about Artificial Intelligence. The tweets pulled by this method has mixed tweets where some feel it is boon and some feel it cannot replace humans altogether.

The data comprises of more than 2000 tweets pulled from Twitter using the #Artificial Intelligence and #Future and #Trends to get the data refined based on the trending things people are talking about Artificial Intelligence and its future. The data is collected in a .csv file which needs to be cleaned before doing any analysis. The data has tweets from wide spectrum of fields where AI is used and we can use it as a whole when talking about people’s opinion.

This Tweet collection is an unstructured text data which cannot be directly used. The data needs to be cleaned for it to be processed and looked at. The data from .csv file is copied to .txt file and then manually cleaned using regular expressions filter in Notepad++ which includes removing https links, handle names of people tagged, removing hash tags, removing duplicate tweets(re-tweets) etc. before using this huge data. The cleaned data is then converted into .csv format again to perform vectorization on it.

**Methods:**

The cleaned data .csv file is loaded into python as a numpy array to perform various vectorization techniques available. Vectorization is to convert documents into word vectors(numbers) to be able to count them. There are various techniques which can be used to do this.

Count the number of times a word appears in a document, the frequency of word in a document is called term frequency. To vectorize, we can use Weka or python. I used python library called Sklearn to perform vectorization. There can be different types of vectorization.

Boolean Representation is where each word is given 1 if present and 0 for absence. Thus, it basically represents the words in documents in Boolean creating a dictionary/bag of words. Term Frequency where each word’s frequency of occurrence is calculated. We can have normalized term frequency which is basically the term frequency divided by the length of the document to normalize it. Unigram, Bi-gram or Trigram where words are taken as combination of 1, 2 or 3 words respectively to create bag of meaningful words. Another technique is TFIDF vectorization which is taking Term Frequency and multiplying it by taking log of Inverse of Document Frequency (flipping numerator and denominator for Document frequency). The value of TFIDF shows frequent the word is or how rare it is.

Stop words are removed from the text because having the stop words such as a, an, the, all, been, ours etc. would take up a huge memory to store. These words being stored would unnecessarily increase the work of vectorizing them which doesn’t make any sense as they themselves don’t create or change any meaning of the word or sentences. We can add our own stop words based on the topic or context we are talking about.

After vectorization we would have a bag or vocabulary of words to look at. We can reduce the vocab list to reduce the burden of handling too many words by using various vocabulary techniques if we want to. Stemming is one of the techniques which can be performed to achieve vocab reduction, where each word is replaced by its root by removing suffix and it may cause certain issue while interpreting.

**Results:**

After loading the .csv file, to vectorize I used sklearn python library. I used Boolean vectorization which shows total documents 2930 and created a bag(vocab) of 690 words. For Unigram and Bi gram method, the bag of words increased from 690 to 1005 almost increasing by 30% while taking one word at minimum and 2 words combined at max to do vectorization which tells us that there are more meaningful combinations when using this technique. Also, the count of individual words increased. For example, the word “machine learning” appeared 359 times before and increased to 549 in the uni and bi-gram technique. The word came once and sometimes in combination with other word. This is one of the interpretations which needs to be looked upon more closely. Vectorization can count words differently and in different combinations which cannot be ruled out. The combinations can be more meaningful rather than only single word. For this single word, alone 359 occurrences and taking it other word increases by 200 occurrences of the same word.

The TFIDF vectorization technique also had vocabulary of 690 words but need to look at the value which depicts the rareness of the words even though number of words is the same. The count was reduced to around 600 using stemming technique. Stemming treated each word of the sentence separately as it was tokenized and then each token was stemmed to its root. There can be 2 techniques to do stemming. Stemmer and Lemmatizer. Stemmer removes postfixes from the word and stems it to its root form without taking into consideration Part of Speech. This creates issue while interpreting as we cannot find out what the word was based on the word being used easily. Lemmatizer on the other hand, transform the word to a real word ‘lemma’ which can be looked at, understood as it considers part of speech and the context the word is used. For example, while performing stemming using stemmer after vectorization using TFIDF method words like actually and activities were named ‘actual’ and ‘activ’ and by using lemmatizer they were given names as ‘act’ and ‘act’. So clearly there is issue with identifying the original word. Even the lemmatizer is not perfect always which is seen in above example. I used Porter stemmer and Lancaster stemmer to see the differences of using the 2 stemming techniques. Others like Snowball stemmer can also be used.

There were few times where linguistic information was omitted by stemming like developers was taken as develop which changes the whole meaning of the sentence. Developers mean people whereas develop means developing not the actual person. Customers and customer both were coded as custom which makes it indistinguishable and unidentifiable as to what the original word meant along with getting rid of plural form as required for customers. Other example of showing plural omission issue is computer and computers being named as comput which changes the rubrics of countable number of things such as computers. Coming and comment were labelled as com using stemming so we cannot make sense of what is being said. This also highlights that it mis-classified the root word for the 2 words. Proportion of number of words reduced by stemming are very less in this case which makes one think it is not useful to try to do stemming using either stemmer or lemmatizer or any other stemmer. It does not reduce the vocabulary size drastically for it to be considered effective to be used on top of vectorization.

**Conclusion:**

Using stemming in this case is not effective and creates issues such as loss of meaning of words, words numerical sense cannot be quantified properly, misclassification or wrong classification of root words, proportion of vocabulary reduction is very less as compared to the total words. So, it is better not to use stemming on this type of small data.

Removing stop words first and then doing stemming or vice versa also changes as per the situation. Removing stop words can also be looked at as type of stemming or filtering as it removes and reduces the vocab word count.

Among the vectorization techniques used TFIDF is better in my opinion as it normalizes the document term frequency at the same time increases the weight of terms present in a particular document and assigns less weight to words which are found in most of the documents. This weight reduction to most common words comes from Inverse of Document Frequency. It improves the performance. We can compare the similarity of documents by comparing TFIDF. The matching of the documents is more influenced by least frequent words.