**Assignment Execution**

* **Importing libraries and downloading dictionaries**

Pandas, Scikit Learn, bs4, nltk and few dictionaries like stopwords, Wordnet

* **Data Preparation**: - The Data is downloaded from the given link and the Parsing of the sgm files is done using BeautifulSoup. The relevant tags like body, newid, lewissplit, topic is fetched and stored as features in Data Frame.

In the Process of **Data Cleaning**, regex expressions are implemented to strip the stopwords and unwanted characters like tags, “,’, “.’ from the body feature. Nan Values are dropped.

Topic Feature is melted down to rows.

* **Feature Engineering**

1. **Vectorization**

Raw text has to be converted to feature Vectors in this step. Several techniques can be used here like CountVectors ,TFID, WordEmbedding , but in this assignment , TFID has been used.

TF-IDF score represents the relative importance of a term in the document and the entire corpus. TF-IDF score is composed by two terms: the first computes the normalized Term Frequency (TF), the second term is the Inverse Document Frequency (IDF), computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document)

IDF(t) = log\_e (Total number of documents / Number of documents with term t in it)

1. **Normalization**

Lemmatization is done in this step using WordNet database since the text is in English.

* **Model Building & Training**

Next step is to train a classifier using features created as vectors. Again, several Classifiers can be used like Naïve Bayes, Linear Classifier, SVM, Bagging, Boosting, DNN but in this assignment, Logsitic regression OneVsRest Classifier has been implemented.

* **Prediction & Results**

Model is able to predict on test data with the accuracy of 66%. To improve it, Model is again trained by feeding only TOP 10 documents and is then able to predict the test data with the accuracy of 90%.

**Improving the Model**

To achieve better accuracy, several improvements can be made.

Following are some tips to do so.

1. Text Cleaning: text cleaning can still be improvised.

For eg, in case of dropping Nan Value, they can be filled to make out training dataset stronger.

1. Stratified Sampling can be done too
2. Earlier mentioned Methods of Feature Engineering can also be tried.
3. Named Entity Recognition can be added.
4. correlation between the topics can be found using chi square test.
5. Stemming Technique for Token Normalization can be tested too.
6. Other Classifiers can be tried too and in case of Neural networks, accuracy can be checked by hyperparameter tuning.
7. Ensemble Models can be implemented by stacking different models