**NLP – TEXT CLASSIFICATION**

The project was divided into 4 parts:

1. Given text, determine the nationality of the speaker.
2. Given text, determine what is the level of his expertise in English.
3. Given text, determine what is the grade of the speaker.
4. Given text, determine the topic of discussion.

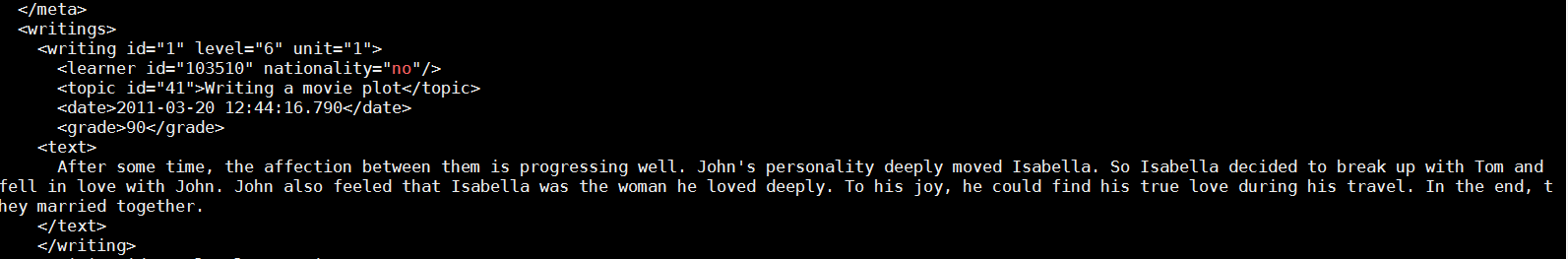
**Dataset:**

The primary dataset is the xml file attached. The file consists of text, learner\_id, nationality, level, grade and topic as either tags or attributes.

**Methodology:**

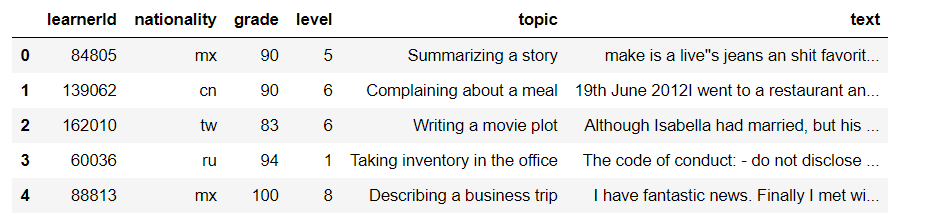
1. **XML To CSV:**

The first task was to convert the xml file to csv file. Since I am using ElementTree, it was essential that the start and stop tags of the entire file need to be <root> </root> respectively. The change was done in linux using vi.



The XML file consists of many unknown undefined tags. As such it is necessary that the file be first cleaned of all such tags. By debugging, it was found that a certain tag <br> was occurring multiple times in the context of the <text></text>. This led to the development of an xml cleaning script which would develop a new xml.

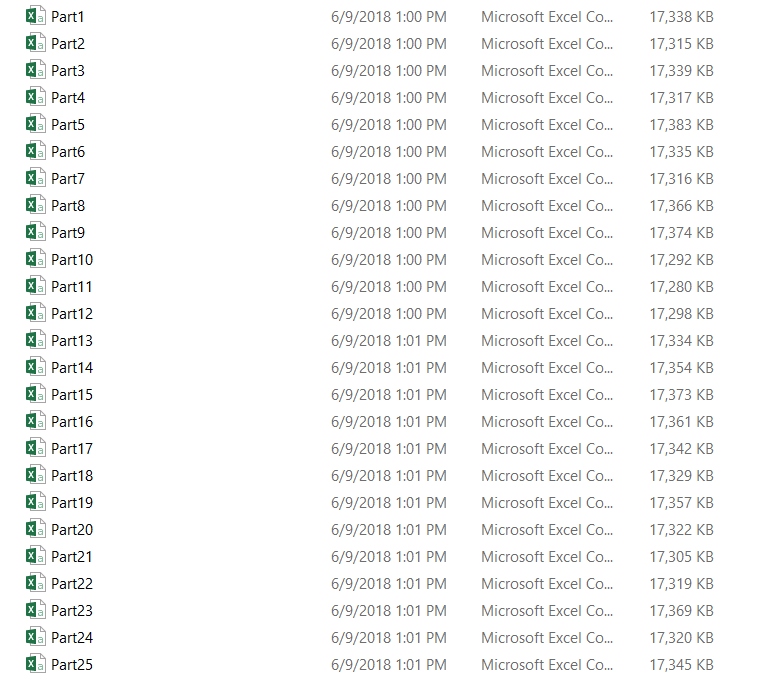
I then used parser script to use the cleaned xml and give a csv output.



1. **Splitting CSV into 25 parts:**

It was a requirement to split the dataset into 25 equal parts with equal representation in grade, level, nationalities and topic. The first point of check was if the data was sorted, and it was. Then I checked for how many observations from different nationalities was present. The observation that came was there are 198 nations, and there were many nations with less than 10 observations. As such no differentiation can be made which are representative of all countries.

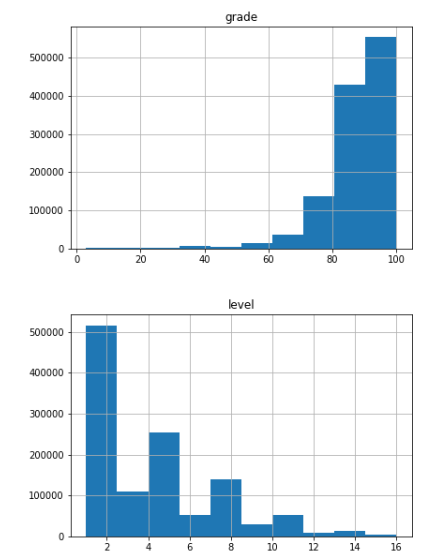
The approach chosen was that the data was first randomized and then the data was split into 25 different parts. The top 80% of the data was just plotted to check if it was representative of the complete sample which it was. 10 part was used as training, and the 2 of the remaining 15 are test set.

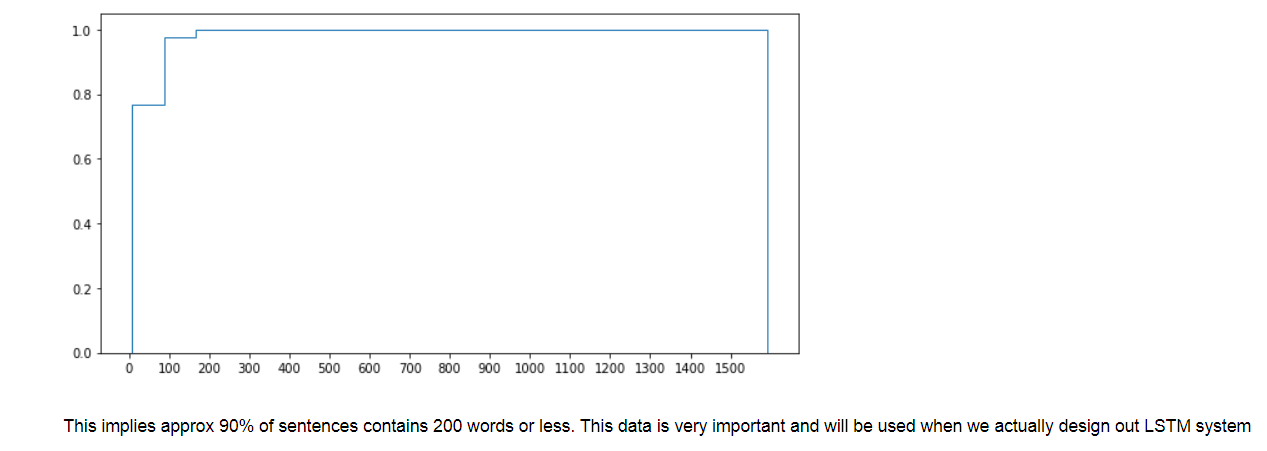


1. **Exploratory Data Analysis:**

Exploratory data analyses were done to get a context of the data. It was helpful to determine how to approach the model development. One was to check what the population distribution was in the dataset regarding nationalities. How was the sample distribution of dataset for levels, grades, etc. Similar words between GloVe corpus and datasets were checked.

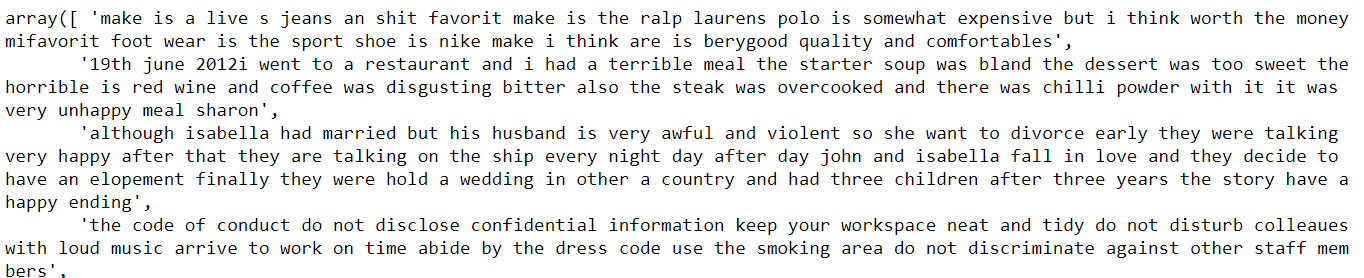
It was also checked as to how much should be my limit to the text layers be. A cumulative distribution plot was plotted how we can have 90% of the words in the dataset using 200 words from each text





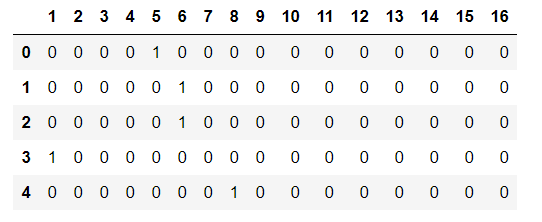
1. **Data PreProcessing:**

The words from the dataset and GloVe corpus was very different. While there are many words that do not necessarily match with the words at all, some of them were due to various words in the text, like “end.As” which was considered as one word in dataset corpus, which could not be matched with two different words in GloVe corpus – “end” and “as”. As such, the preprocessing of text was done where the punctuations were changed to a space.



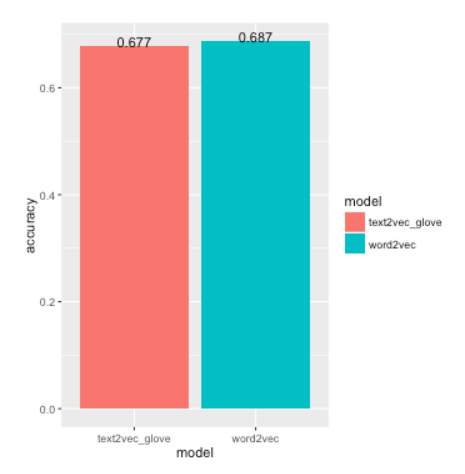
1. **One Hot Encoding:**

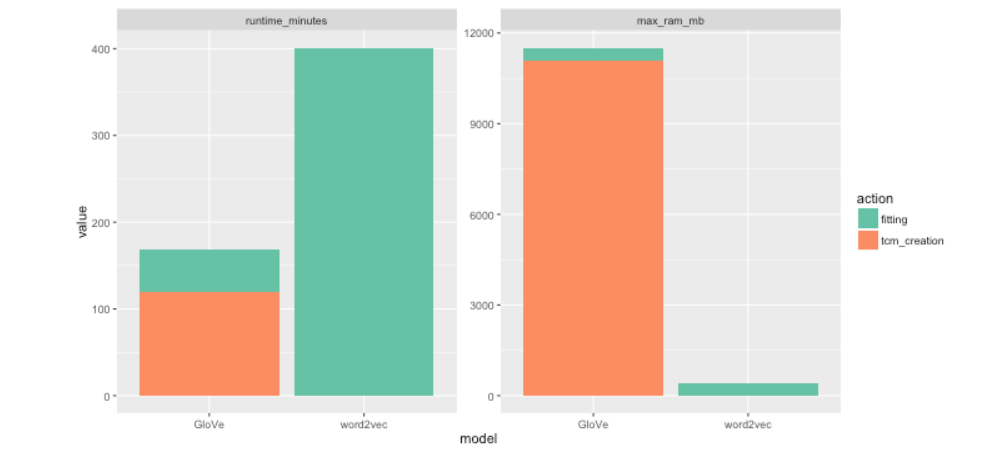
Since there were many classes, both character level and numeric, one hot encoding was implemented to encode categorical feature. The figure below represents One-Hot Encoding of 16 different levels in the levels column.

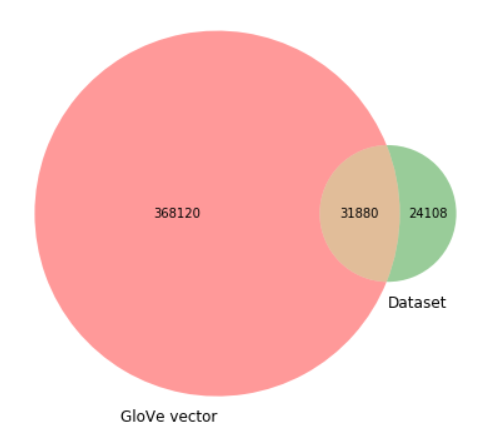


1. **Word Embeddings:**

GloVe was used for word embeddings of text. The reasons for using GloVe over word2vec was while the accuracy of both of them are almost similar, the run time complexity of GloVe was much less than word2vec (Source: <http://dsnotes.com/post/glove-enwiki/>)

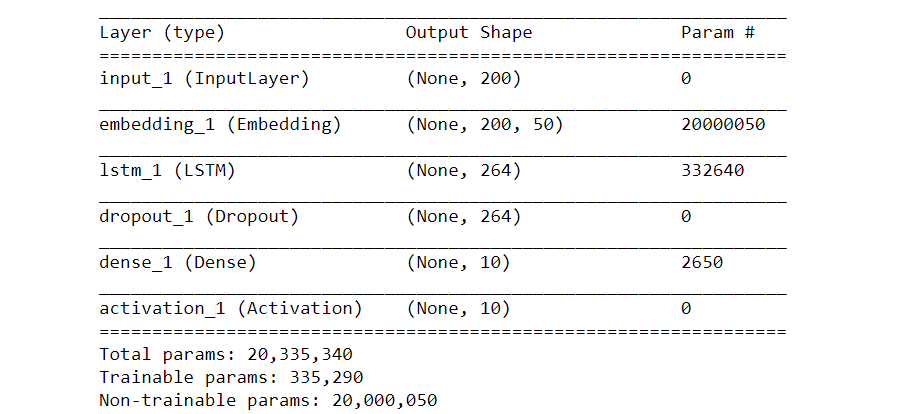




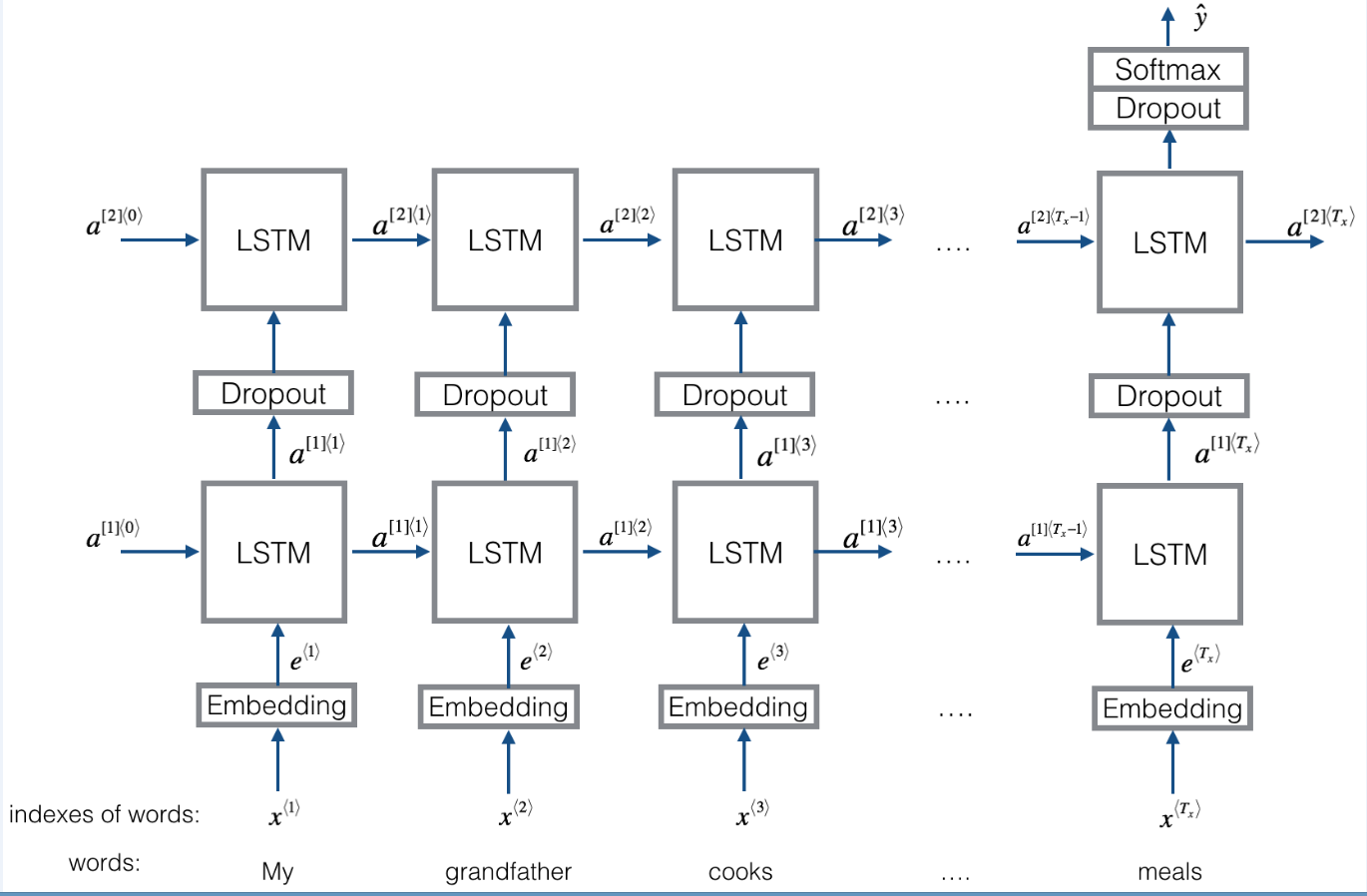


1. **Model Architecture:**

LSTM was used as a model architecture. One layer LSTM was used which was propagated through a dropout and a dense layer with softmax activation.



The architecture layer can be compared with the below given layer taken from Sequence Models certification by Andrew Ng on Coursera, although with one LSTM layer:



**FUTURE SCOPE:**

The project can be further extended to implement thesewhich might result in better accuracy:

1. **Removing Class-Bias:**

The data has a lot of skewness and its only fair to assume that there will be class bias. We must try to remove it. Let us consider the dataset of nationalities and text. There are 198 classes. We can build 198 models, where each model has a certain number of rows of one class, and equivalent number of rows of all other rows combined. This is one of the ways to remove class bias.

1. **Different combinations of LSTM/GRU networks:**

Other way to check/improve model accuracy is by using different layers (LSTM), change it using bidirectional rnn, etc. Only one layer is used here for computational complexity and time constraints.

1. **Different Neural Networks:**

Different Neural networks other than rnn could have been compared. In recent times, it has found that cnn to have equivalent performance to lstm.

1. **Hyperparameter Tuning:**

The hyperparameters needs to be fine-tuned and see which configuration has better results.

1. **Increase number of epochs/k-fold:**

We can improve the number of epochs or k-folds to increase performance.

1. **Use of other features:**

I have only used text to determine various features. However, only

**STEPS ON HOW TO REPLICATE:**

1. Put all the files in a single folder which is your home directory.
2. Make sure you have similar xml file.
3. Run the cleanXML file as it has been pointed out that the xml is not clean.
4. Now the cleaner file will generate a new xml which is free of unbiased tags.
5. Now run the 1\_parseXML file. This will parse the cleaned xml file and generate a csv file.
6. Next run the “Splitting” file. This will divide the data into 25 different datasets, all stored in csv format.
7. Run the EDA file.
8. Run the Training Nationality Model file.
9. Run the Training Level Model file.
10. Run the Training topic Model.
11. Run the Training Grade Model.

To use any of the 15 datasets in the training model, go to the “Training the model on any validation data set” in the training models. There is a statement: “part\_no = X”. Change X between any of 25 parts as a validation part.