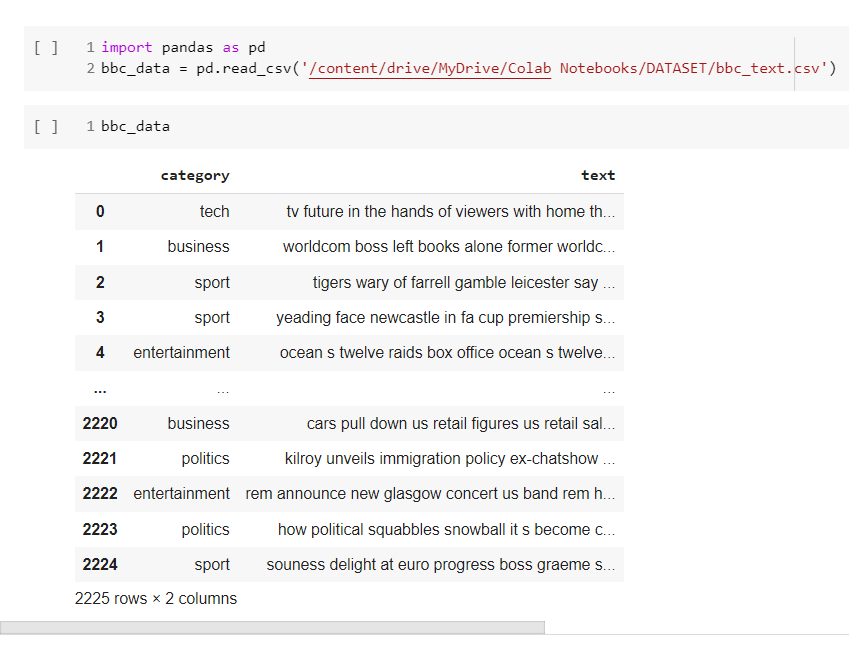
Analytical Result

1. **DATA IMPORT**

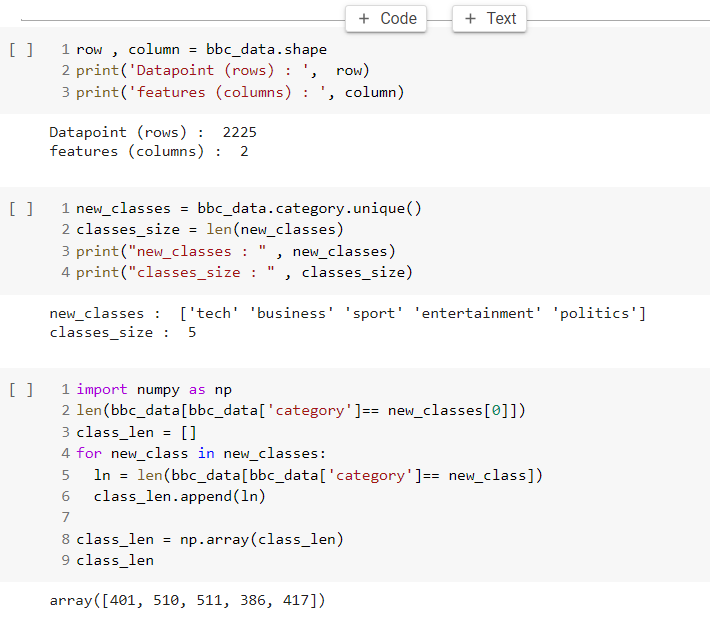
The BBC csv file is downloaded from Kaggle repository and its imported into the Google Colab environment for data science operation. The fig 1 shows the code snippet for data import and sample datapoint.



**Fig 1 data import and Sample data point.**

1. **Data exploration (Getting to know the dataset)**

The BBC news dataset contain one input feature (new content) and the output feature (the target class). The total BBC news samples is 2225 datapoint and each datapoint is group into five different classes or news group. This include; tech news, business news, sport news, entertainment news, and political news making 5 groups in total. The aim of the experiment is to efficiently classify news articles into the earlier mentioned groups. Based on the statistical data after exploration its identify that; the dataset is imbalance.

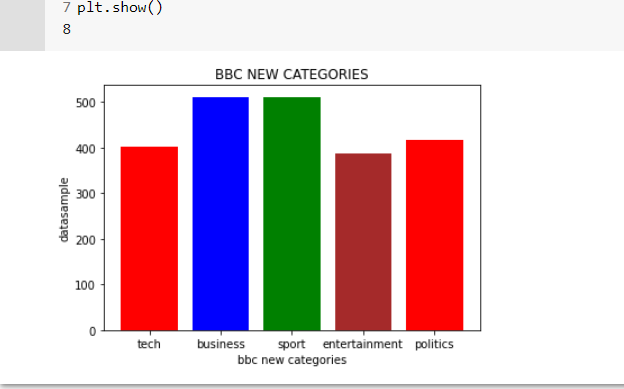


**Fig 2 Data exploration**

Based on fig 2 it’s shown that the:

1. Tech news class contain 411 datapoint
2. Business news class contain 510 datapoint
3. Sport news class contain 511 datapoint
4. Entertainment news class contain 386 datapoint
5. Politics news class with 417 datapoint

However, the fig 3 below show that sport and business news categories as the largest datapoint, while the entertainment news class has the minimum datapoint. In a glace it can be deduced that the datapoint for the class are not the same.



**Fig 3 Graphical Exploration of BBC News Categories**

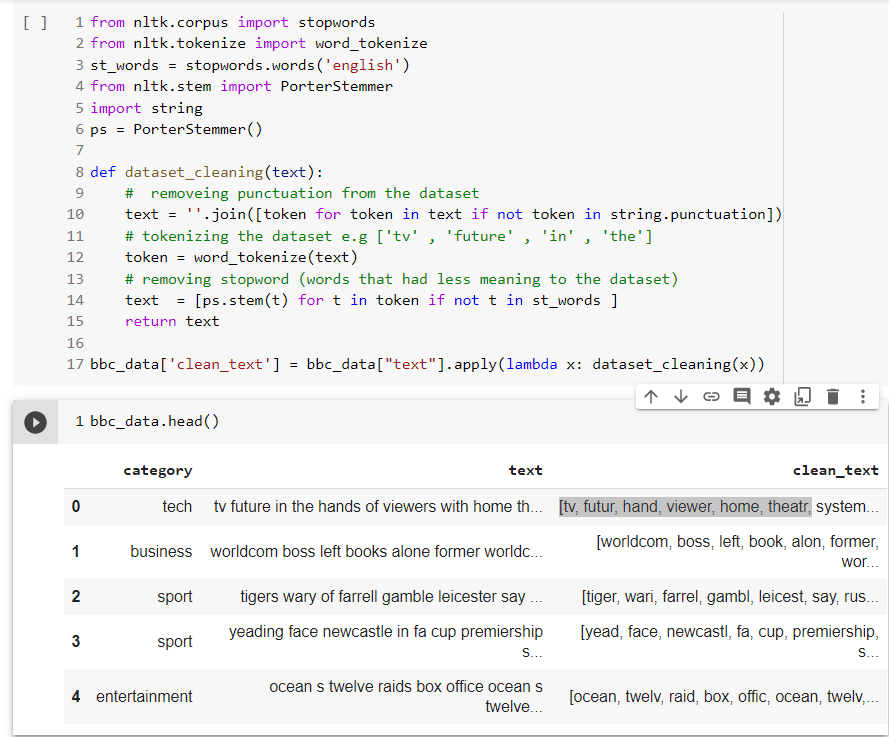
1. **DATA PREPROCESSING AND CLEANING**

the data cleaning or preprocessing is essential because irrespective of how good a Model is if the supply data is not clean or full of unnecessary content the model will not perform efficiently.

The data preprocessing stage includes

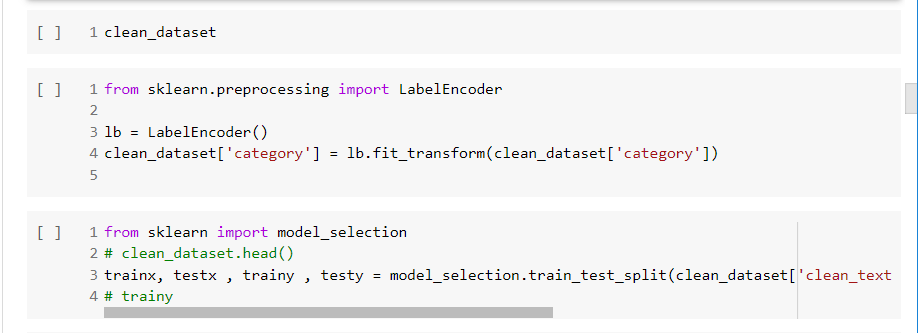
1. Tokenization of sentences. E.g [tv, futur, hand, viewer, home, theatr]
2. Removal of punctuation such as (.,-?@#$^ etc)
3. Removal of stop words or word extraction to candidate terms
4. Stemming of candidate terms.

However, the resulted clean text and code fragment after preprocessing is illustrated with the fig 4 below.



**Fig 4 Data Cleaning and Preprocessing**

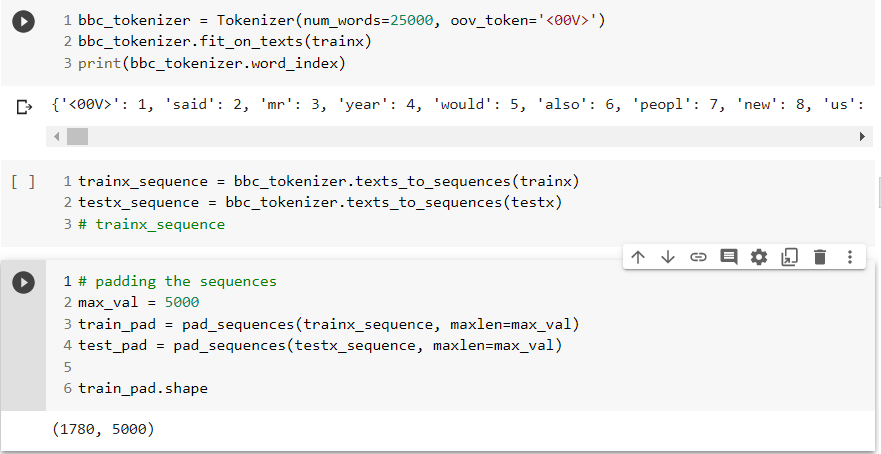
The Preprocessed data and cleaned text is represented with the feature name ‘clean\_text’. Further preprocessing includes converting the classes (tech, business, sport, entertainment, political) into number representation (tech = 0, business = 2). This is done or achieved using Label-Encoding. The fig 5 shows the code snippet for label encoding conversion and splitting of dataset for training and testing purpose.



**Fig 5 Category Label Encoding and BBC news data splitting.**

Additionally, for any deep learning classification the following step are critically important to maintain consistencies among data.

1. **Tokenizer and Word Index:** the clean text and train each word in other to generate a vocabulary of words represented with a unique number.
2. **Sequencing:** this includes represent each datapoint text with the equivalent unique ID in the vocabulary, something like vectorizing but in this situation the order of word is in place.
3. **Padding:** this process is essential in other to maintain equal length among all sample data point. Padding is done by adding redundant zeros to words with less threshold length or cutting words that has higher threshold length. For this experiment we uses 5000 word to set the padding threshold.

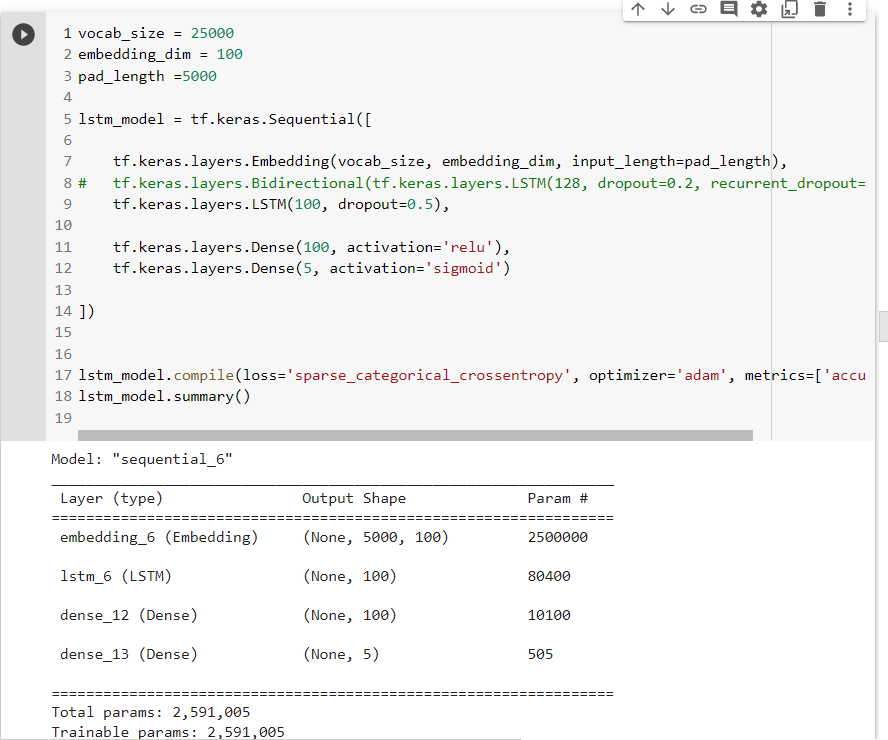


**Fig 7 Word Index, Sequences and Padding.**

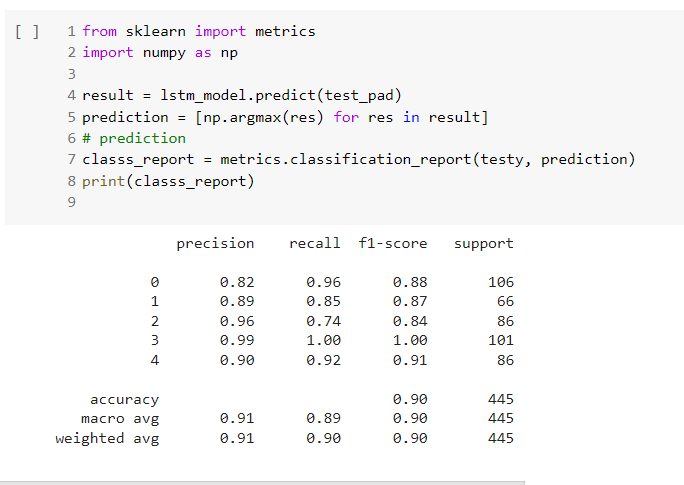
1. **LSTM BENCHMARK TRAINING MODEL (WITH NO AUGMENTED DATA)**

All the LSTM model used in this experiment are place under the same configuration, such as

1. Number of embedding network layer
2. Number of LSTM layer
3. Number of Dense layer
4. Same activation function is considered.
5. Same loss, optimizer and metrics are use all through.
6. And same 8 epochs (Number of training iteration) is used.



**Fig 8 LSTM Benchmark model Configuration.**



**Fig 9 LSTM Benchmark Evaluation Report after 8 Epoch.**

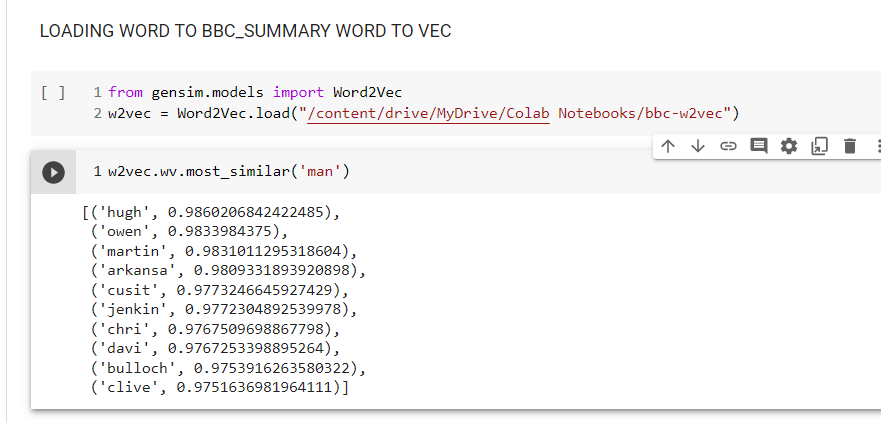
The fig 9 show that 90% accuracy is achieved with the LSTM model.

1. **LSTM TRAINING MODEL (WITH WORD2VEC AUGMENTED DATA)**

this section illustrate how augmented data are generated using Word2Vec model to enhance the dataset for better prediction accuracy.

* 1. **Data Augmentation using Word2Vec**

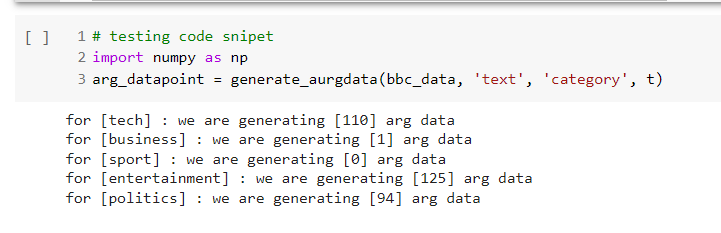
The original BBC data is used in training the Word2vec model, this encode every word using N dimensional vector (in this experiment 100 dimension is used). Meaning each word has a 100 numeric value for describing that particular word, this is achieved using the Gensim module in python to train the Word2vec model. After training the model can now be loaded for data augmentation. Code fragment in the fig 10 below shows how the Word2Vec model is loaded and testing the similarity score of a word.



**Fig 10 Loading Word-to-vector model and checking similarity score.**

Considering the similarity score result, ‘owen’, ‘martin’, ‘hugh’, and ‘Jenkin’ must have been used in similar context as the word ‘man’. However, in reality those names are players names and players are ‘man’ in nature, this shows that the word embedding is producing significant result.

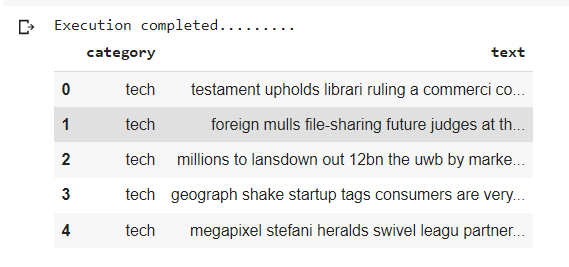
The Word2Vec model is now utilize in generating augmented data from existing BBC news data, by simply substituting similar word used in the same context.



**Fig 11 balancing the data set using Word2Vec (Augmented datapoint).**

**Sequential steps taking for generating augmented data**

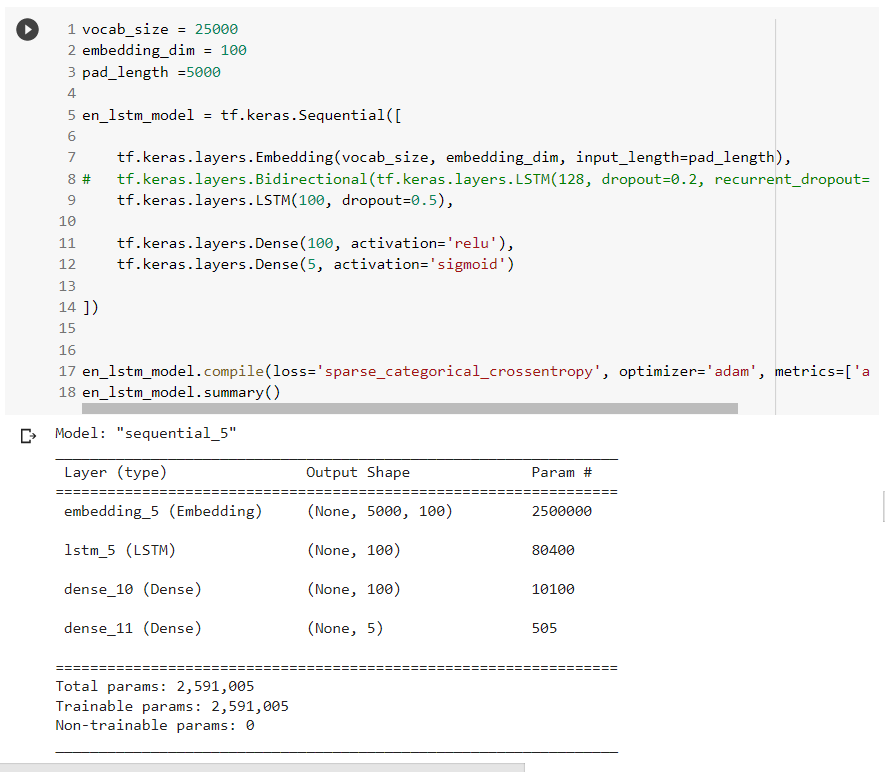
1. Train Word-2-Vec model using Gensim module
2. Import Word-2-Vec model
3. Loop through all news category classes and compute the number of samples to be generated to balance the class datapoint.
4. Generate augmented data for each BBC news class category using similarity score to substitute words.
5. Finally, merge the augmented data with the main BBC news data



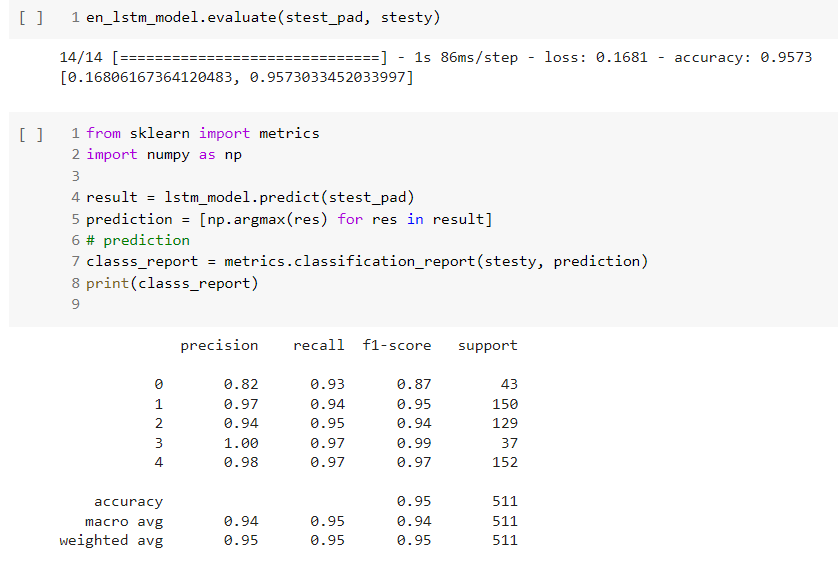
**Fig 12 Augmented sample datapoint**

* 1. **LSTM model with enhance dataset (Word-2-Vec Augmented Data)**

NOTE: before model training preprocessing such as, tokenization, stemming, removal of stop, generating vocabulary, sequencing, padding is carried out on the new Generated dataset has earlier mention in this experiment. However, the same model configuration is used to but with the improved dataset to see if the added dataset has significant impact in the training accuracy.



**Fig 13 Model Configuration**



**Fig 14 Result Analysis**

After training and evaluation of the LSTM + Word-2-Vec (augmented dataset) model resulted in an accuracy of 95%.

1. **LSTM TRAINING MODEL (WITH WORDNET AUGMENTED DATA)**

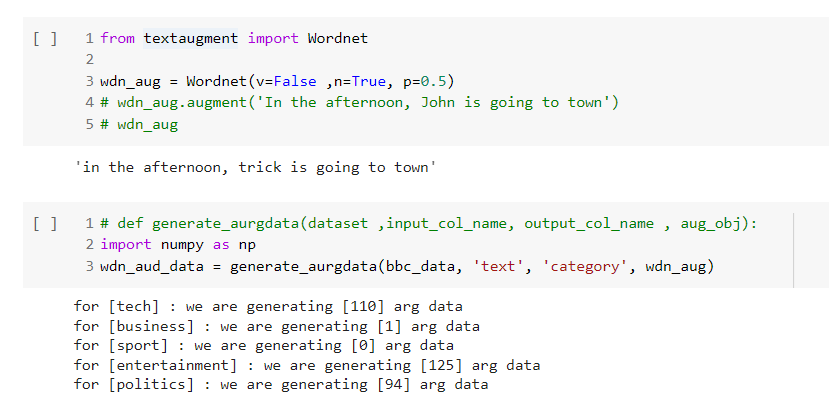
this section illustrate how augmented data are generated using Wordnet Synsets to enhance the dataset for better prediction accuracy.

* 1. **Data Augmentation using WordNet**

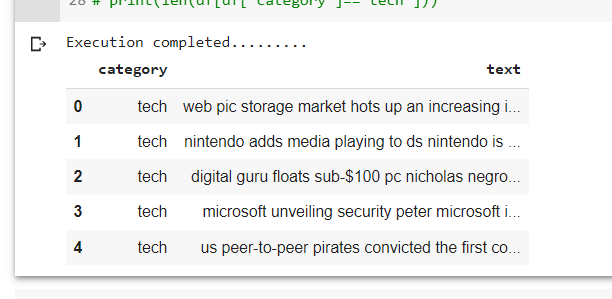
**The Sequential Step in Using Wordnet to Generate Augmented Data.**

1. Select Random samples from the main BBC news dataset.
2. Select random candidate term for substitution from the selected sample datapoint.
3. Using Synsets (synonyms words) and Part of Speech Tagging to substitute words for generating augmented datapoint.
4. Merging the augmented datapoint to the main BBC datapoint.

The fig 14 shows the function call to wordnet routing to generate the augmented datapoint.



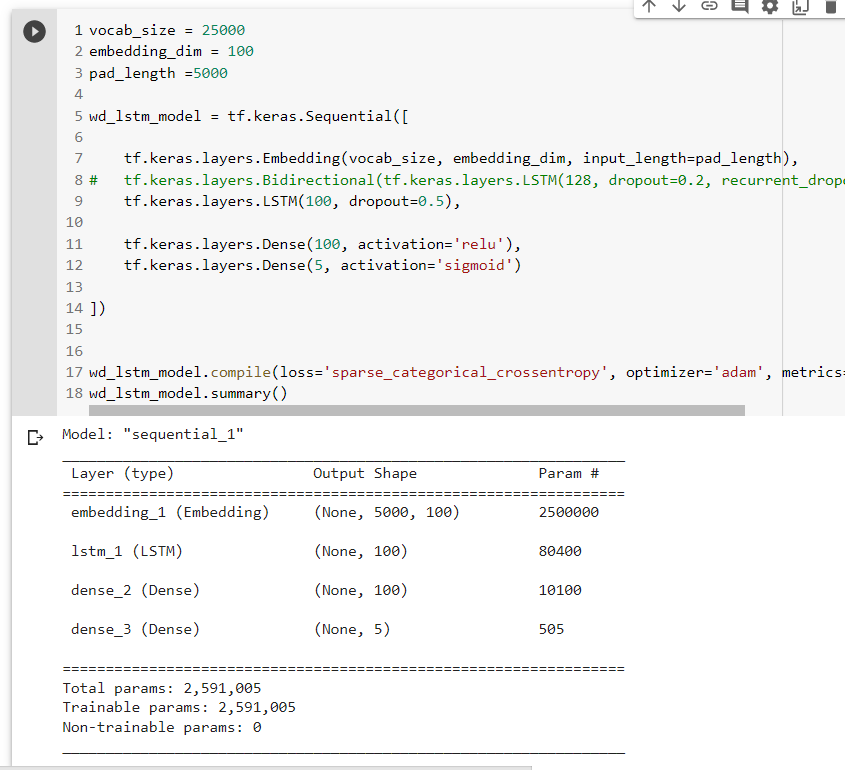
**Fig 15 Wordnet Augmented Data**



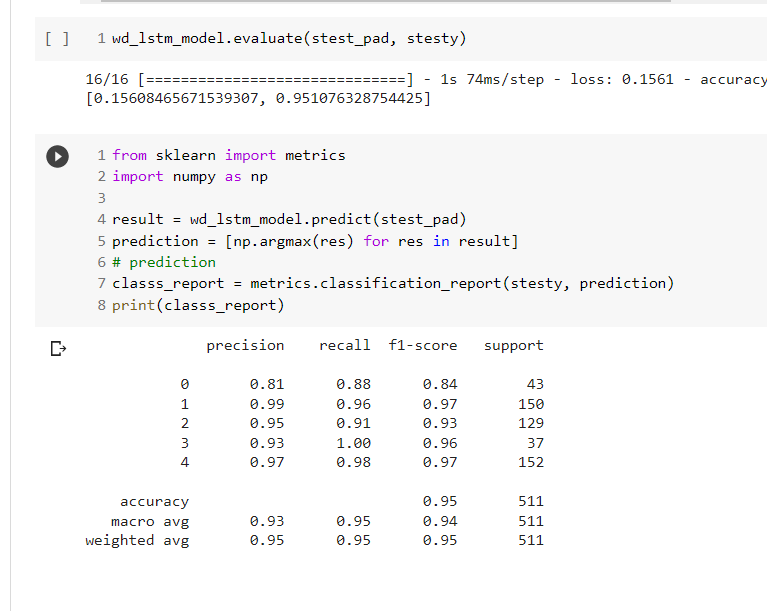
**Fig 16 Sample Data Generated by WordNet**

* 1. **LSTM model with enhance dataset (WordNet Augmented Data)**

NOTE: before model training preprocessing such as, tokenization, stemming, removal of stop, generating vocabulary, sequencing, padding is all done has explained earlier in this experiment. However, the same model configuration is used to but with the improved dataset to see if the added dataset has significant impact in the training accuracy.



**Fig 17 LSTM + WordNet Model configuration**



**Fig 18 Result Analysis**

After training and evaluation of the LSTM + WordNet (Augmented dataset) model resulted in an accuracy of 95%.

1. **FINAL COMPARISION**

|  |  |  |  |
| --- | --- | --- | --- |
| **S/N** | **Augmented techniques** | **Model** | **Accuracy** |
| **1** | **None** | **LSTM** | **90%** |
| **2** | **Word embedding Techniques** | **LSTM+Word-2-Vec** | **95%** |
| **3** | **WordNet Synset Approach** | **LSTM+ WordNet** | **95%** |