**Sentiment Analysis on Twitter data**

**Abstract**:

Identifying opinions and sentiments from tweets is called as “Twitter Sentiment Analysis”. The main idea behind Twitter sentiment analysis is to determine the polarity of the tweet and classifying them into positive or negative sentiment tweet. In this project we used Bidirectional Encoder Representations from Transformers (BERT) model for to determine the polarity of the twitter data. At first, the raw tweets are pre-processed by removing the urls, white spaces, characters that are not letters. Then, we utilized pre-trained BERT pre-processor and encoder from tensor flow for sentiment classification. We have evaluated the model performance with metrics like confusion matrix, accuracy, precision, recall, f1 score, and auc.

**Introduction**:

Now a days, Internet is becoming very popular and more important. It is serving as a cost-effective platform and used of social media become a necessary daily activity as it is used for social interaction. Several social media platforms like Facebook, Instagram, blogs, reviews, tweets are being processed for extracting the people’s opinions about any particular product feedback, organization, medical experience, professional knowledge, or any situation. The attitude and feeling behind any opinion plays a vital role in evaluating behaviour of an individual which is known as sentiment.

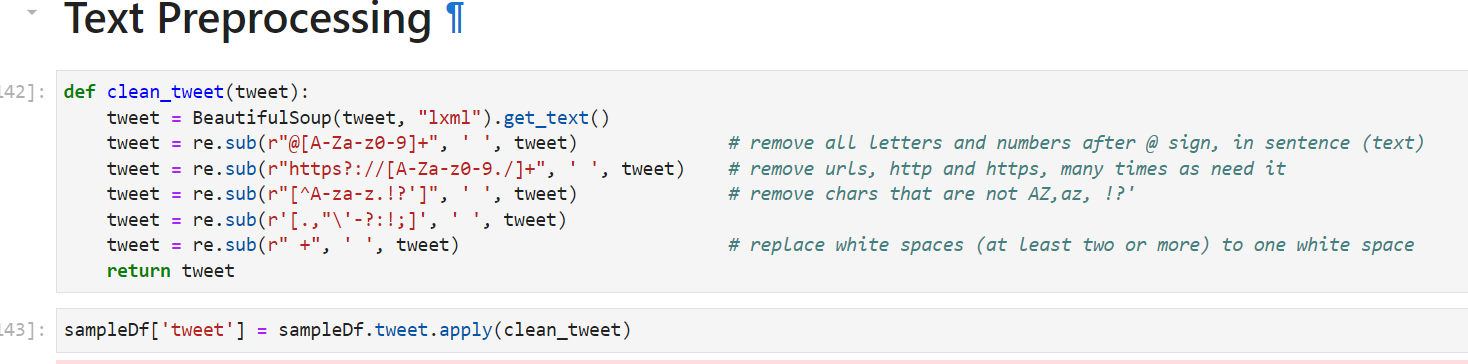
Different Machine learning methods were introduced to perform sentiment analysis those are categorized under unsupervised and supervised leaning methods. This project is mainly contributed to perform sentiment analysis using Bidirectional Encoder Representations from Transformers (BERT), a popular and effective model which makes used of Transformer that learns **contextual relation** between words in a text. The BERT models use the Transformer encoder architecture to process each token of input text in the full context of all tokens before and after, hence the name: Bidirectional Encoder Representations from Transformers.

The objective of this project was to build an effective approach based on the BERT model for Twitter sentiment analysis. First, we explored the raw twitter data and pre-processing is done by creating a balanced dataset and removing the urls, white spaces, characters that are not letters. then we build word embedding layers with an input layer, dropout and dense layers to build BERT model by utilizing pre-trained BERT model from Tensor flow on plain text. Trained model with training data, fine-tuned and evaluated on test dataset and predicted the output using test data. Performance metrics carried out to measure the performance of the model.

**Technical Approach**:

Step-1: Pre-processing of data.

The raw tweets in the dataset generally result in a very noisy due to people’s random and creative use of social media. Tweets have certain special features, i.e., emojis, emoticons, hashtags and user mentions, coupled with typical web constructs, such as email addresses and URLs, and other noisy sources, such as phone numbers, percentages, money amounts, time, date, and generic numbers. The pre-processing step is needed in order to remove all the words that are irrelevant for sentiment analysis.



Step-2: BERT Model

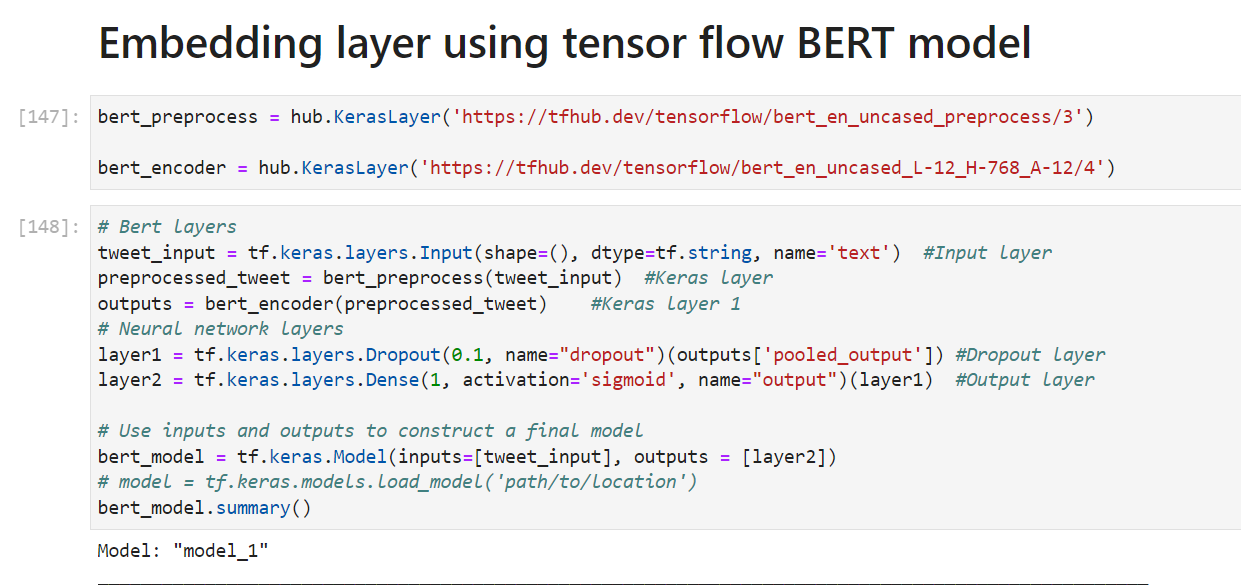
BERT model is used for emotion classification. The meaning of a word in a given sentence depends on the other words surrounding it. The BERT feeds all input at once to handle dependencies among words. The BERT-base model uses 12 transformer encoders, we can easily fine-tune the BERT model to get the desired results.

In this project we followed the following steps to create a mask and the encoder representation of the BERT using Tensorflow with Keras library for emotion classification:

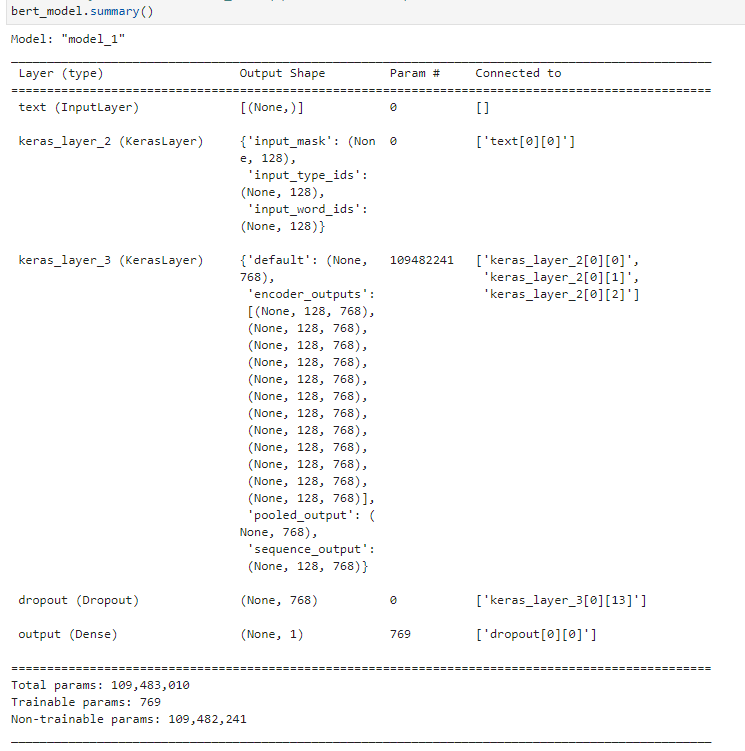
* Dividing the collected data into training and testing sets using train-test split.



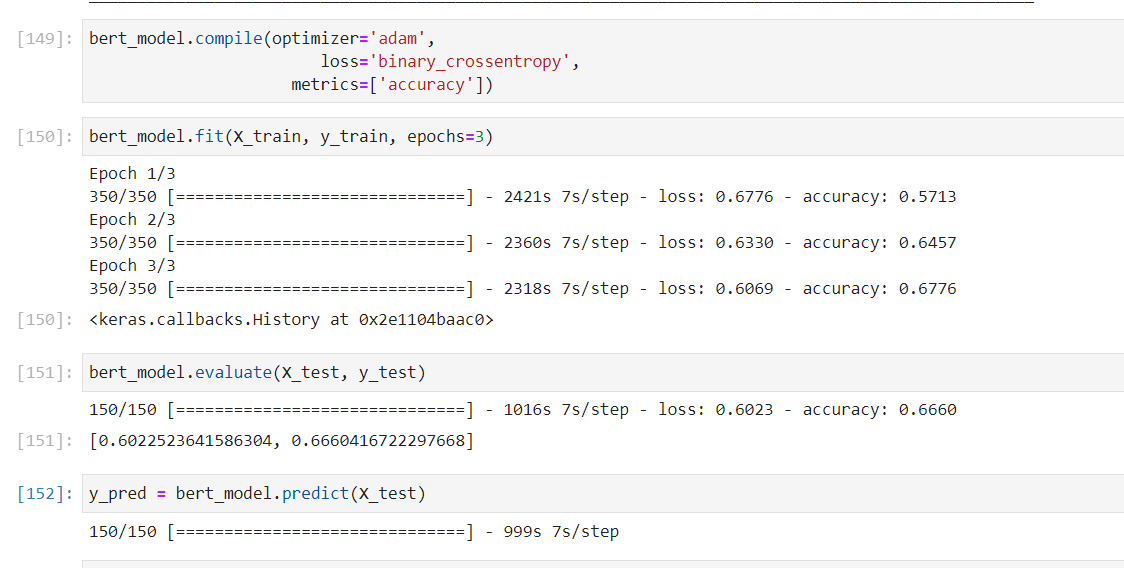
* we build word embedding layers on pre-processed twitter data with an input layer, dropout and dense layers to build BERT model.



Summary of the model created with embedding layers –



* Compiling, model fitting on training data and evaluating the model on test data.
* We used Adam as our optimizer, binary\_crossentropy as our loss function, and accuracy as our accuracy metric. Fine-tuning the model for 3 epochs.



* Evaluating the performance of the model.

**Confusion matrix** -

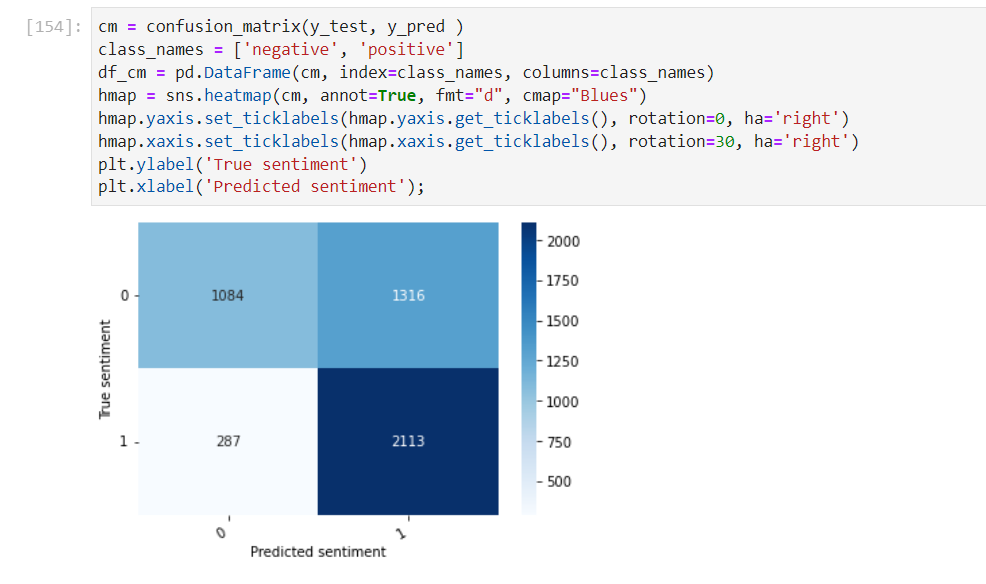
It is a table with 4 different combinations of predicted and actual values.

True Positives: Model predicted positive and it’s true.

True Negatives: Model predicted negative and it’s true.

False Positives : Model predicted positive and it’s false.

False Negatives: Model predicted negative and it’s false.



**Accuracy -** From all the classes (positive and negative), how many of them model have predicted correctly.

**Recall** - From all the positive classes, how many model predicted correctly.

**Precision** - From all the classes (positive and negative), model have predicted as positive, how many are actually positive.

**F1 score** - The F1-score combines the precision and recall of a classifier into a single metric by taking their harmonic mean.

**AUC** **(Area Under the ROC Curve)** - it is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve.



**Conclusion**:

The BERT model predicted sentiment of tweets with F1 Score – 0.72 and AUC score – 0.77 which is reasonably good compared random model with auc of 0.50.

**References**:

1. Alammar, J (2018). The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning) [Blog post]. Retrieved from <http://jalammar.github.io/illustrated-bert/>
2. A Survey on Concept-Level Sentiment Analysis Techniques of Textual Data <https://ieeexplore.ieee.org/abstract/document/9454169/citations#citations>
3. M. G. Sousa, K. Sakiyama, L. d. S. Rodrigues, P. H. Moraes, E. R. Fernandes and E. T. Matsubara, "BERT for Stock Market Sentiment Analysis," 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), 2019, pp. 1597-1601, doi: 10.1109/ICTAI.2019.00231.
4. <https://tfhub.dev/google/collections/bert/1>
5. <https://www.tensorflow.org/text/tutorials/classify_text_with_bert>
6. <https://www.tensorflow.org/tfmodels/nlp/fine_tune_bert>