

|  |
| --- |
| Sentiment Analysis Twitter  Kaggle |
|  |
| July 2  IIT KANPUR Summer Internship Project  Authored by: Aman Srivastava aman13799@gmail.com |

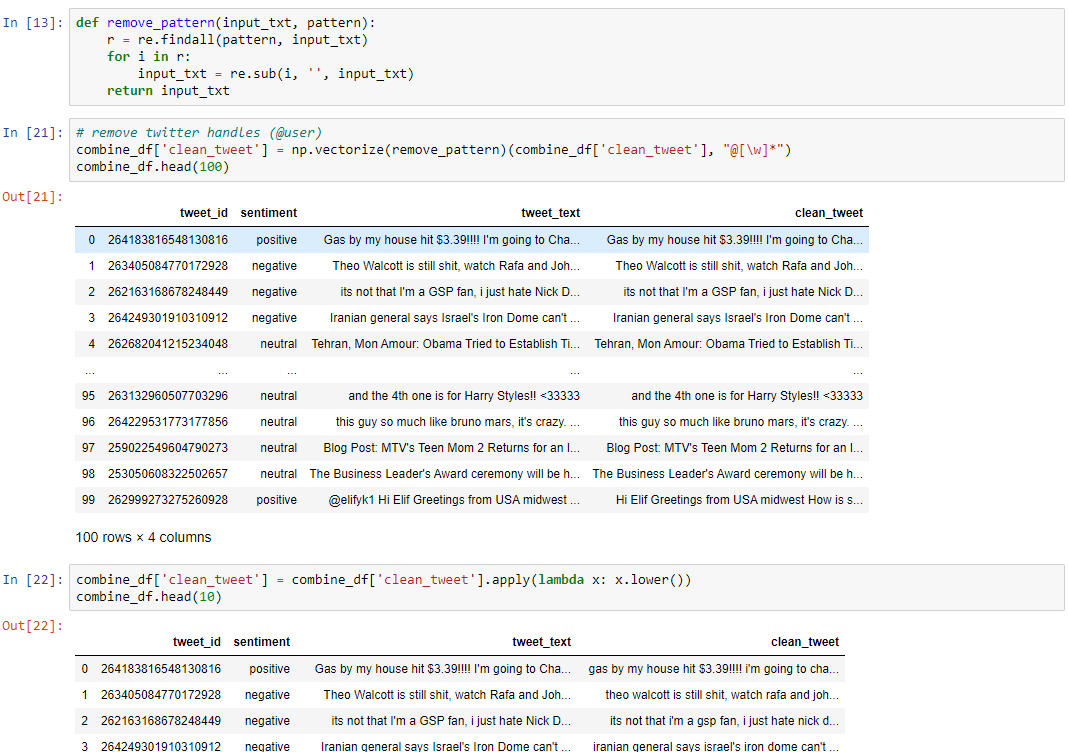


# Tweet Polarity Classification:

|  |
| --- |
| Given a message, classify whether the message is of positive, negative, or neutral sentiment. Sentiment analysis is the interpretation and classification of emotions (positive, negative and neutral) within text data using text analysis techniques. Sentiment analysis allows businesses to identify customer sentiment toward products, brands or services in online conversations and feedback. |
| *“The meaning of your communication is the response that you get.”* |
| **INTRODUCTION**  **Types of Sentiment Analysis**  Sentiment analysis models focus on polarity (positive, negative, neutral) but also on feelings and emotions (angry, happy, sad, etc), and even on intentions (e.g. interested v. not interested).  Here are some of the most popular types of sentiment analysis:  **Fine-grained Sentiment Analysis**  If polarity precision is important to your business, you might consider expanding your polarity categories to include:  Very positive  Positive  Neutral  Negative  Very negative  This is usually referred to as fine-grained sentiment analysis, and could be used to interpret 5-star ratings in a review, for example:  Very Positive = 5 stars  Very Negative = 1 star  **Emotion detection**  This type of sentiment analysis aims at detecting emotions, like happiness, frustration, anger, sadness, and so on. Many emotion detection systems use lexicons (i.e. lists of words and the emotions they convey) or complex machine learning algorithms.  One of the downsides of using lexicons is that people express emotions in different ways. Some words that typically express anger, like bad or kill (e.g. your product is so bad or your customer support is killing me) might also express happiness (e.g. this is bad ass or you are killing it).  **Aspect-based Sentiment Analysis**  Usually, when analyzing sentiments of texts, let’s say product reviews, you’ll want to know which aspects or features people are mentioning in a positive, neutral, or negative way. That's where aspect-based sentiment analysis can help, for example in this text: "The battery life of this camera is too short", an aspect-based classifier would be able to determine that the sentence expresses a negative opinion about the feature battery life.  **Multilingual sentiment analysis**  Multilingual sentiment analysis can be difficult. It involves a lot of preprocessing and resources. Most of these resources are available online (e.g. sentiment lexicons), while others need to be created (e.g. translated corpora or noise detection algorithms), but you’ll need to know how to code to use them.  *OUR APPROACH*  *After exploratory analysis of the given dataset it was found that the dataset was imbalanced. So, after cleaning the data, I applied data augmentation to make dataset balanced and then trained a Neural Network on it.*  *Cleaning the dataset:*  Tweets are noisy data and the sentences may contain many different types of short hand words instead of complete words. This is maily because tweets have a maximum limit of 250 charecters. So people try to use shorthand words/emoticons as much as possible to fit the tweets under the max limit.  Also, tweets contain hyperlinks which should be removed as they are of little use to us.   * Importing the dataset from kaggle        * We now merge both train and test dataset as both dataset needs to be cleaned. |



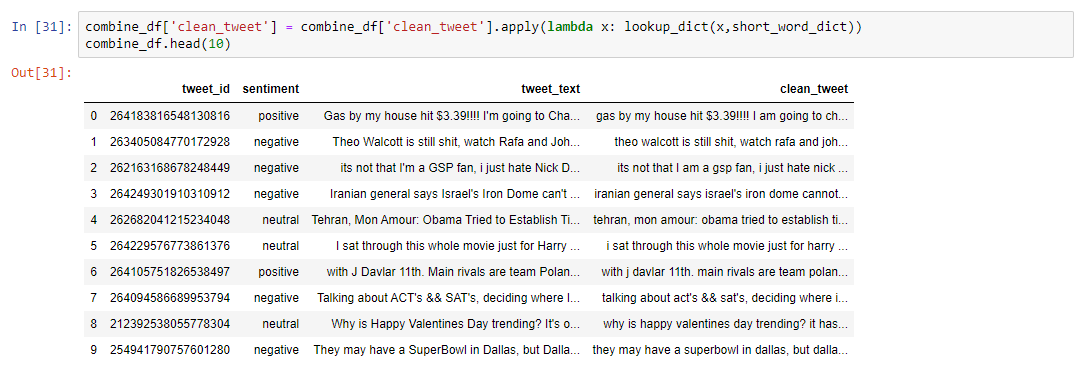
* Next we wrote a simple function which removes pattern from the dataset by passing regex pattern and text into it.



We apply the following modifications to tweet:

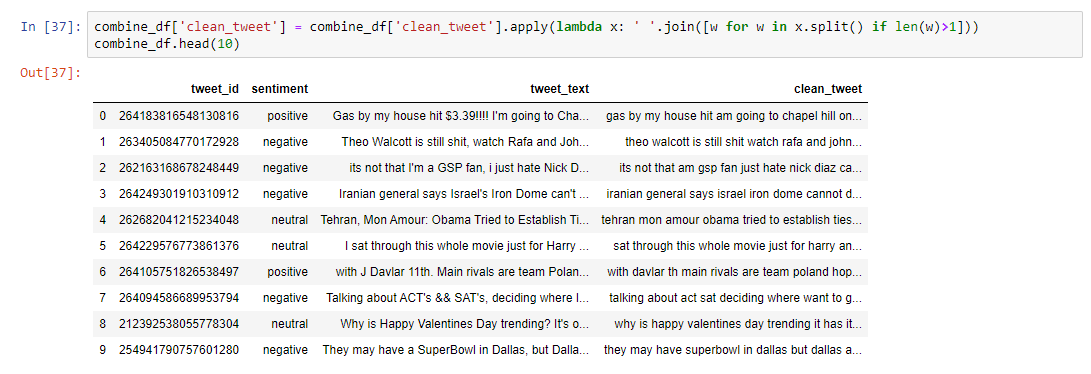
* Remove Usernames (@user)
* Convert tweets to lowercase
* Converting apostrophe words to set of words. (“I’m” to “I” am etc.) using a apostrophe words dictionary
* Remove hyperlinks
* Convert short words to complete words using a short word dictionary
* Converting emoticons to word emotions using a emoticon dictionary
* Removing all special characters
* Removing all numbers
* Removing single characters (word length < 1)



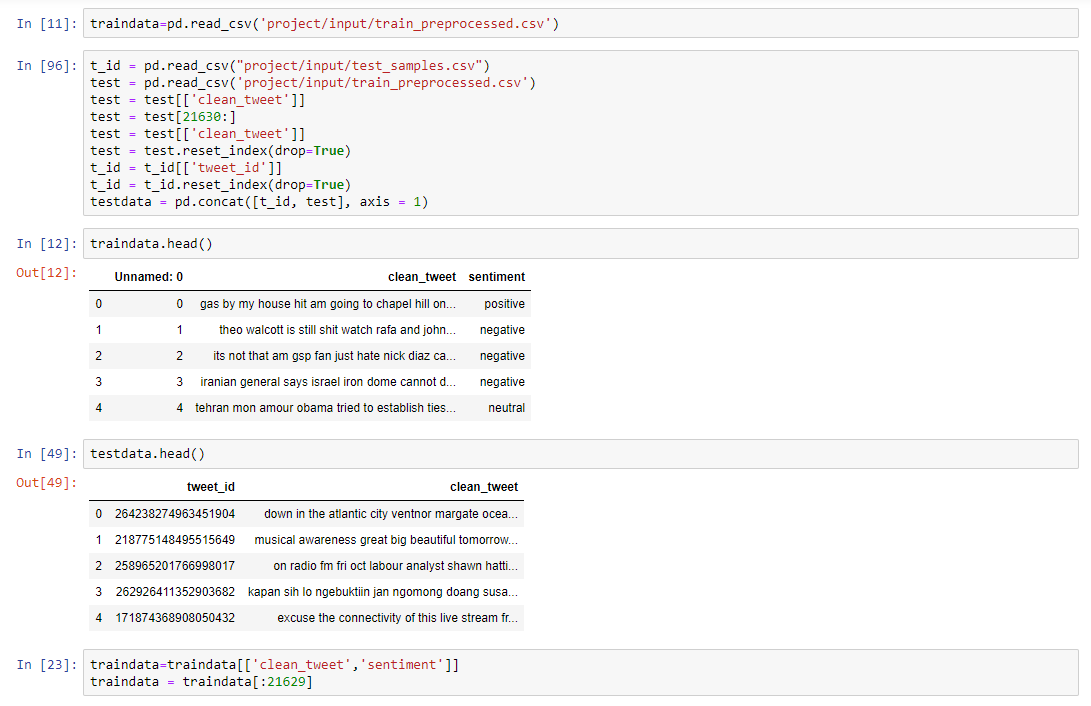








We then save the processed dataset as a csv.

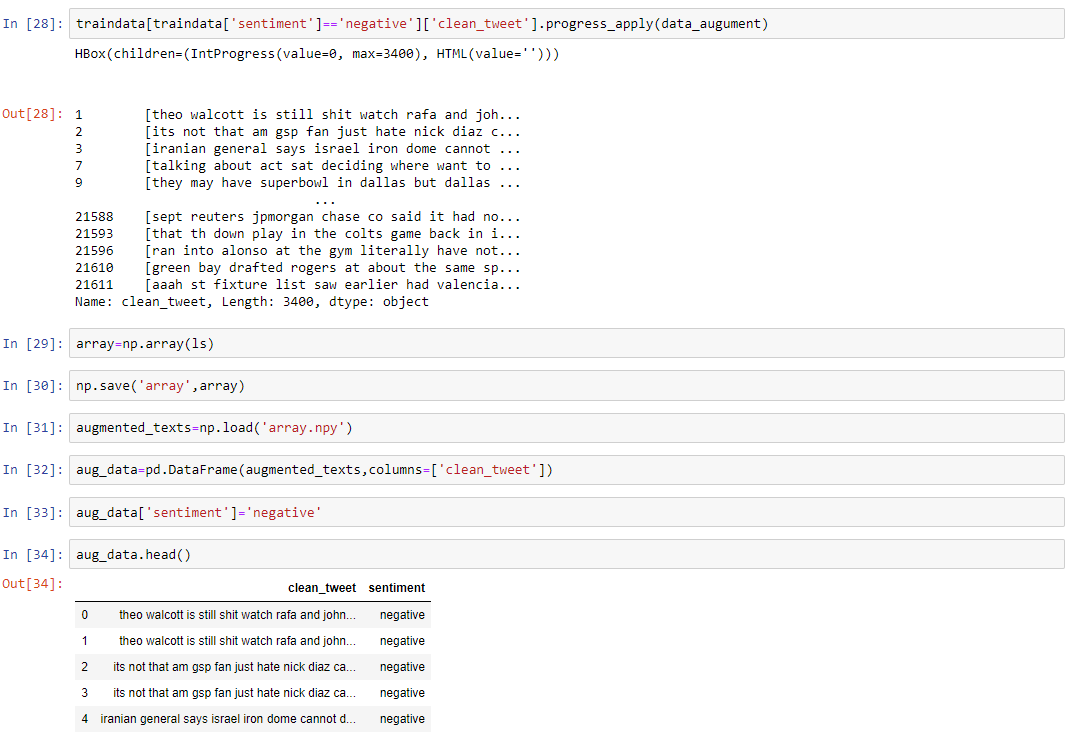


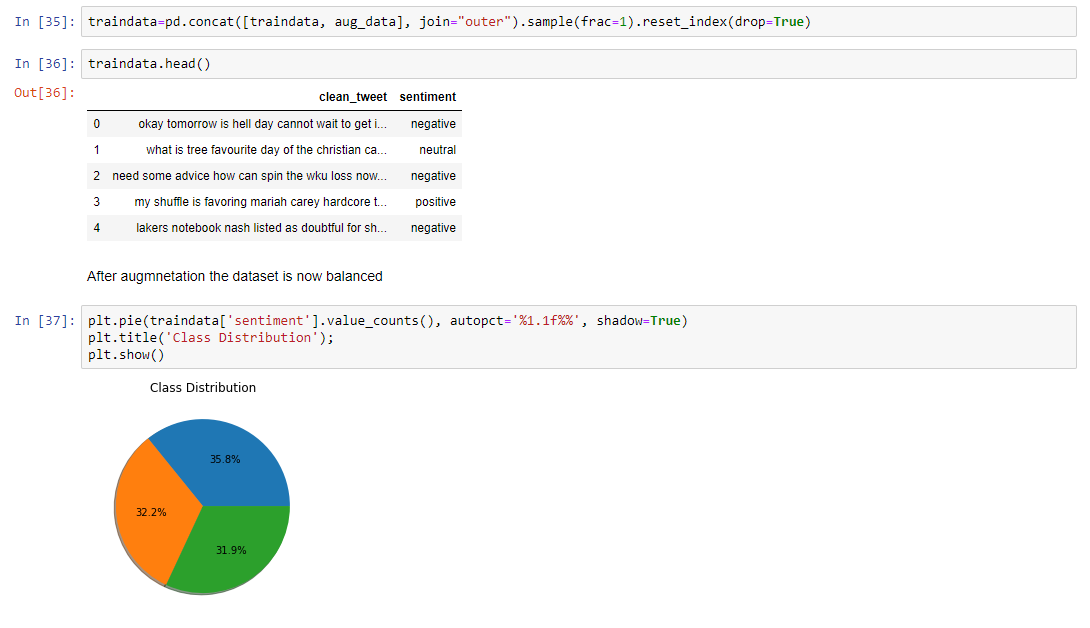
As we can see that the value counts of negative sentiments is a lot less than positive and negative sentiments. This makes the dataset heavily imbalanced.



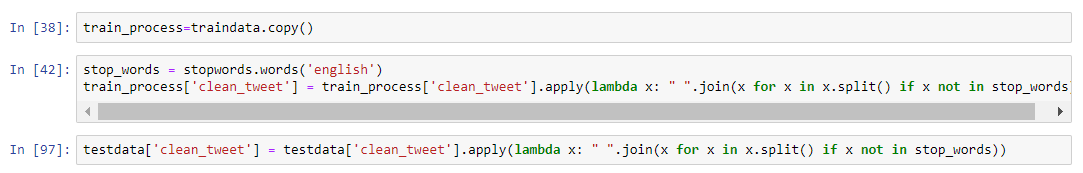
Data Augmentation using nlpaug

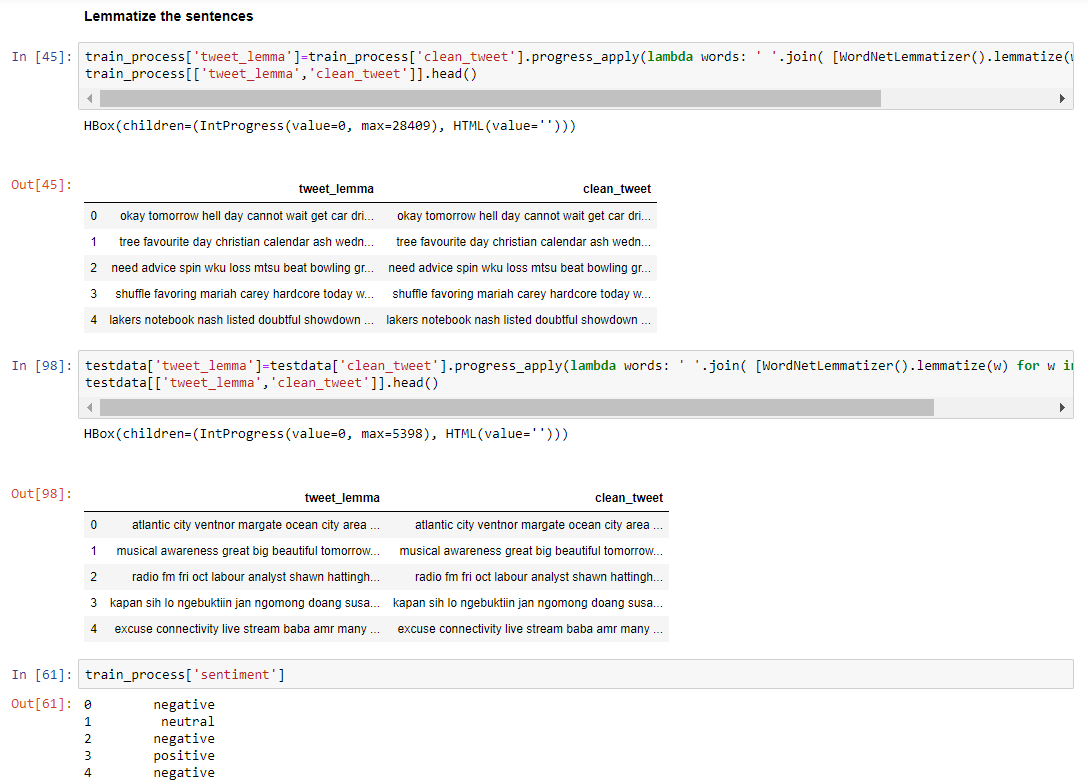






After text augmentation we remove stop words and lemmatize the dataset again as augmented tweets might have stop words in them.

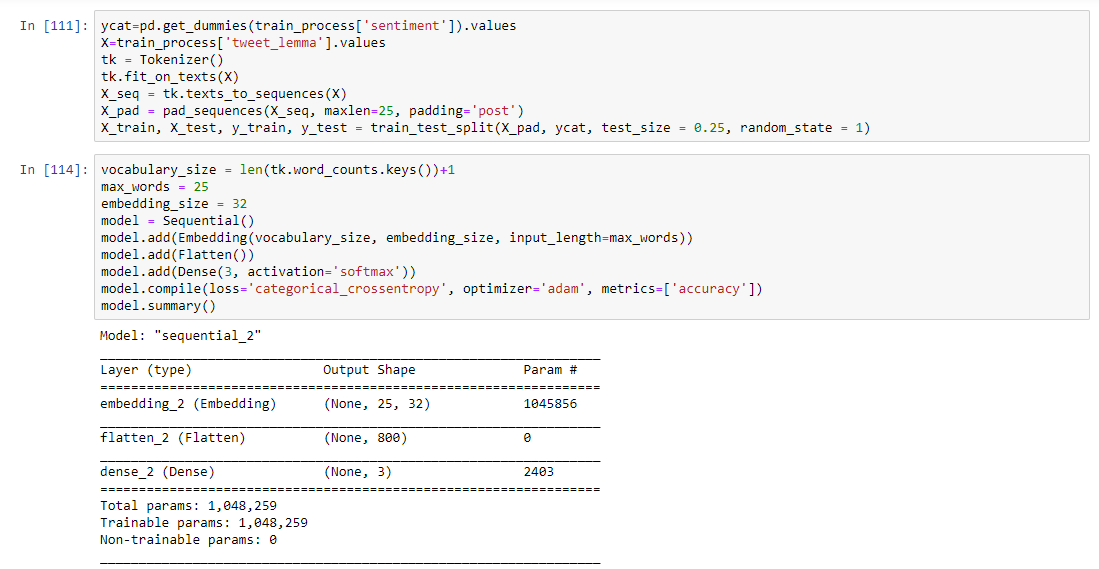


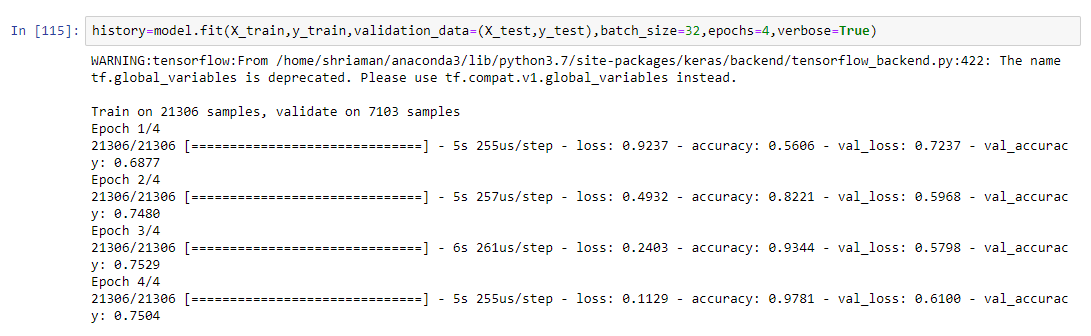


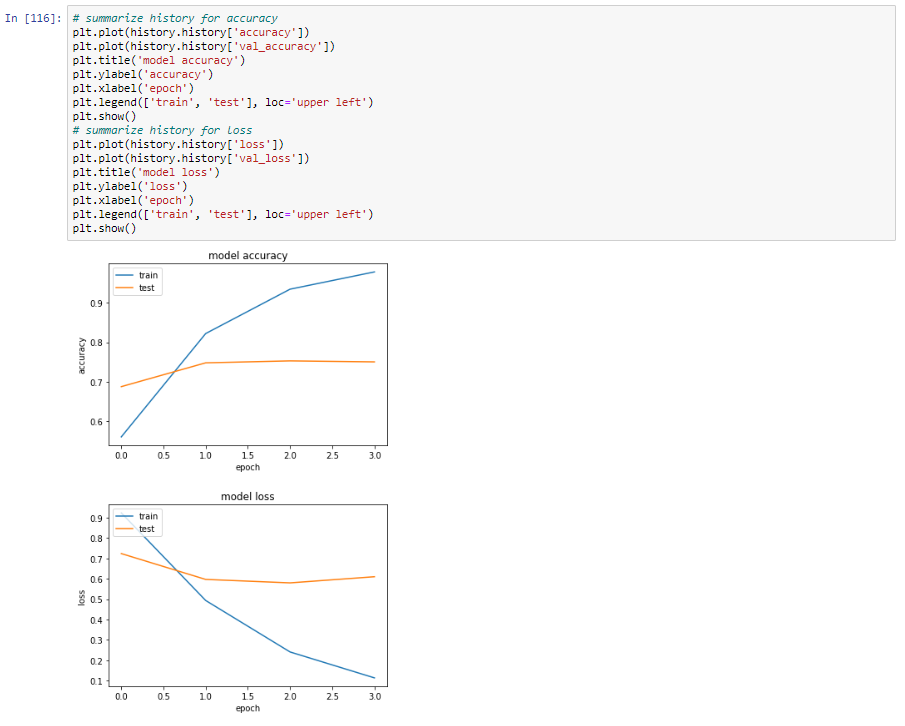
Next, we convert sentiment values into numbers.



Deep learning Model

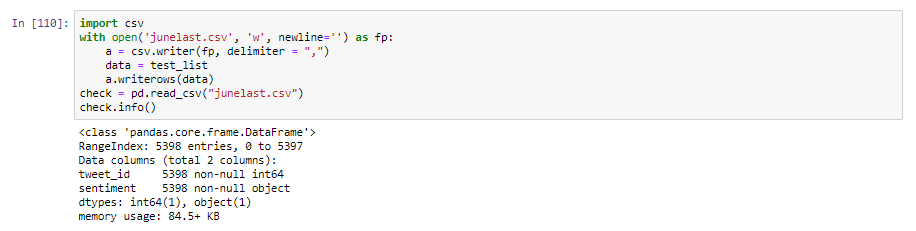






The trained model obtained an accuracy of 75% on verification set. We now run the model on test dataset to upload the output of model on Kaggle.





On Kaggle, this approach of test augmentation and a simple neural network obtained an F1 score of .63

We can try to improve the model further by changing the model to a RNN or try with randomForrest.