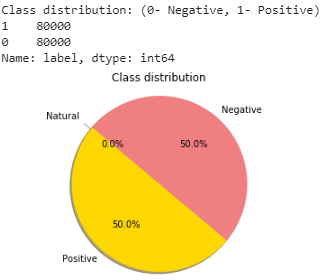
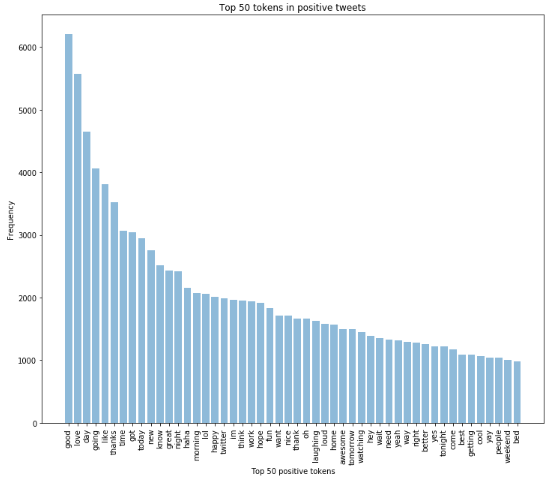
**Twitter Sentiment Classification report:**

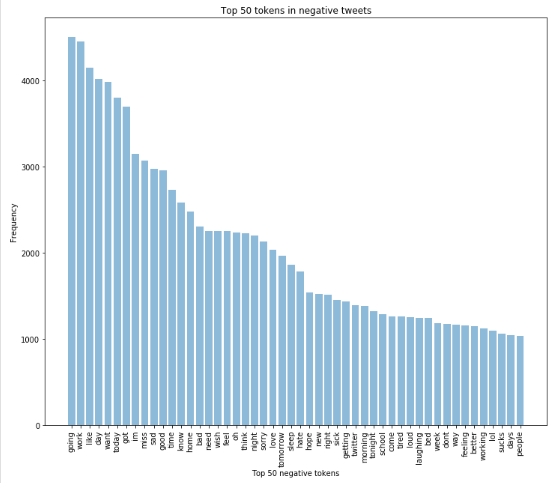
**Question 1:**

After reading the 1,600,000 processed noemoticons from Kaggel’s dataset, we decided to decrease the volume of the data to 10% from its original size into 160,000 noemoticons. We kept an equal class distribution (80,000 from each class- positive and negative). This was done in order to enhance training speed, although it results in inferior accuracy.



The following are top 50 term frequencies for each of sentiments:





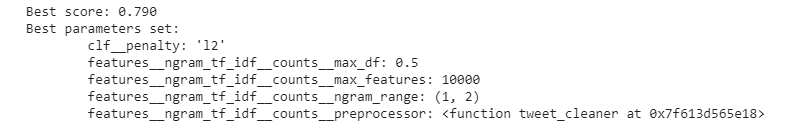
**Question 2:**

We had struggled to find the best way to preprocess the tweets given. Finally, we had concluded, using numerous gridsearches on smaller data, that the best combination would be to use TFIDF representation for the logistic regression and for the preprocessing itself we employed stemming, de-contraction of contracted phrases (don’t -> do not, etc.), resolving emojis in the tweets to indicative words (I.e., “:)" -> “happy”), dropping hashtags, relating to other users (“@myId” were dropped), etc.

We trained two models:

* Logistic regression-

We tried several parameters and got the best result when using the following:



In addition, we used a 5-fold CV to find a representative result of our model.

The features we used were TFIDF representation of the documents, the number of exclamation marks and the hour of day in which the document was published.

The highest average score we got: **0.790** and this is the train set accuracy of the logistic regression model.

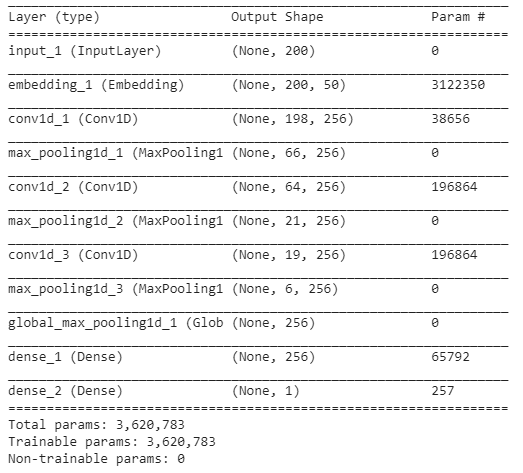
Accuracy of test set: **0.79315625**

* CNN-

At first we tried training an word embedding + LSTM model, but this was taking a long while to train.

We decided to go for a CNN based model, and not a LSTM one since we have experienced a satisfactory performance out of the CNN model.

The model summary:



We used 3 epochs while fitting.

Training accuracy: 0.8521

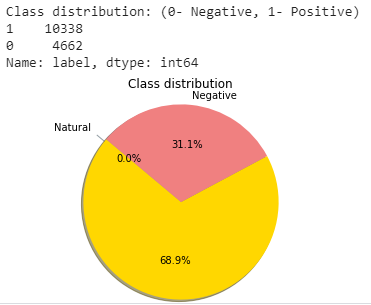
Testing accuracy: 0.7891

We can see that the model is a bit overfit.

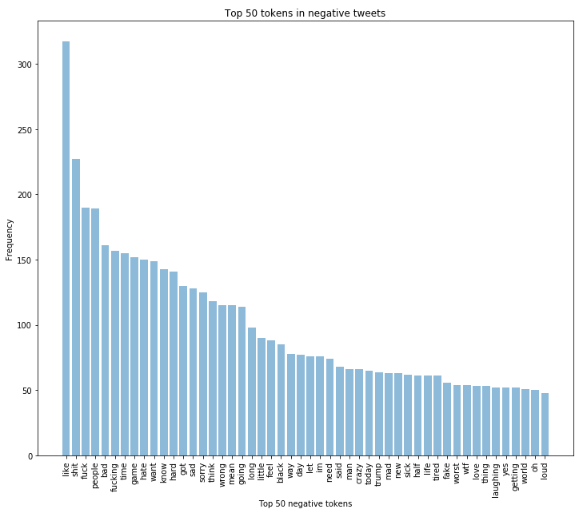
**Question 3:**

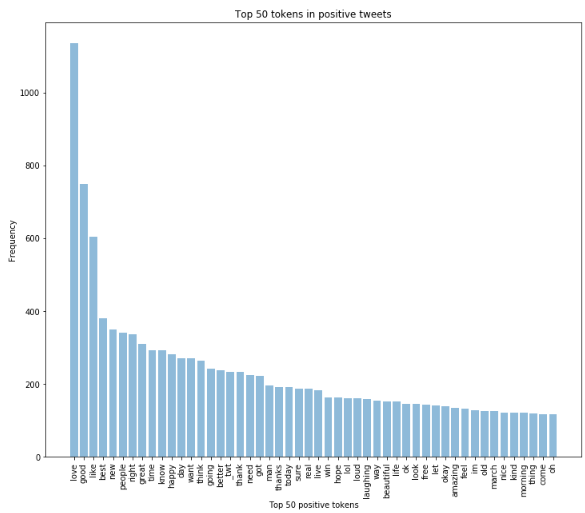
We had trouble distinguishing tweets written in English. So, we used a small dictionary of the most frequently used words within English tweets, to include only those on our mined dataset.

We decided to stream 30,000 tweets at first and then take 15,000 tweets that contain positive and negative sentiment, since we assumed that there will be tweets with neutral sentiment.

Here the class distribution is not equal:

Most popular terms for each sentiment:





We decided to label the tweets using a known library, and after some research we found textBlob library (depended on nltk library). We changed the emojis to be represented as words, but we found out that this library is not helping with the words we chose.

Therefore, we summed the positive emojis by ranking we thought would fit (happy – 0.1, playful – 0.05, and the negative smileys representation were managed by the library correctly), and added to the ranking we got from textBlob library for the sentences from the tweet.

In addition, we thought that the timing, length and many other properties, such as number of exclamation marks, number of dots, number of capital letters, etc, will help us to determine the tweet’s sentiment, but unfortunately, that did not help.

We left the code we tried in comment in the notebook.

Comparing those terms frequencies with the train we can see that most of the words in the train set for negative sentiments are words with a bad context. i.e.: miss, sad, sorry, hate, sick, etc.

While in the test set, we can find more cursing and slang tokens. i.e.: shit, fuck, crazy, wtf, etc.

In the positive sentiment, the words are mostly the same: love, happy, going, morning, amazing, awesome and many more appearing in both the train and the test set. We did not find any unique tokens besides ‘twt’, which appears in the train set as twitter.

In addition, the word ‘like’ is resented in the top 5 of both negative and positive sentiment (in the train and test sets). We can understand that this word is used as an adjective and adverb and not only as a verb.

From that we can conclude that while the samples were taken at different times, there is some resemblance between the given dataset and the data we have been mining from Twitter. Let it be noted that there is a difference in the distribution of negative-positive tweets, but that is to be expected as:

1. The tweets we have mined were randomly sampled, while the given dataset has been artificially engineered to have a uniform distribution.
2. The mined tweets have been labeled using a different algorithm than the labeling done on the given dataset.

**Question 4:**

Attached the output of our prediction according to the model we chose in Question 2 (CNN).

We compared the prediction from this section to the labels calculated in the 3rd section and got accuracy of: 0.60261

As in Question three, we expected the performance to degrade, as the labeling has been done differently in respect to the dataset the model has been trained on.

Let it be noted that upon calculating the accuracy, we rely on the fact that the labels determined in the tweet labeling (Question 3) are indeed correct, and we would like to compare the prediction performance in this section to the labels given in Question three.