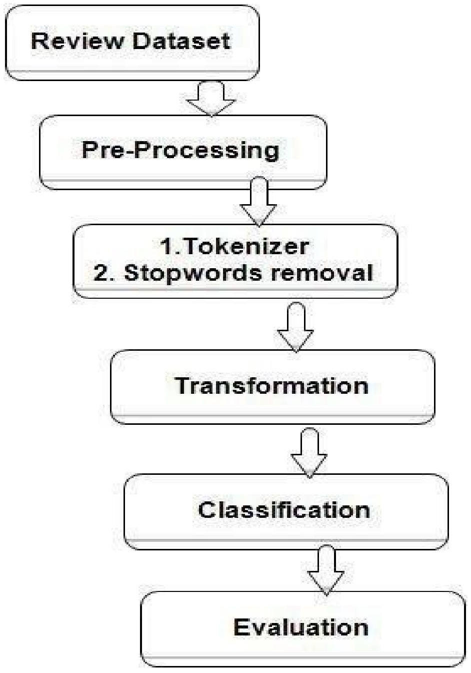
Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

Sentiment Analysis examines the problem of studying texts, like posts and reviews, uploaded by users on microblogging platforms, forums, and electronic businesses, regarding the opinions they have about a product, service, event, person or idea.
Sentiment analysis is becoming more and more popular with availability of alternative datasets such as Twitter etc. Companies like Amazon, Yelp, Uber etc. routinely analyze user reviews to understand user sentiments. With chatbots and conversational agents becoming more and more popular, sentiment analysis is gaining mainstream acceptance.Building a sentiment analysis engine from scratch is a challenge.Traditionally, bow models, word embeddings using glove/word2vec have been popular and the Movie review database is used to illustrate these concepts. In the last few years, Deep Neural networks and RNNs have been tried for sentiment analysis. Many cloudvendors are offering sentiment analysis APIs[2,3,4,5].
As a classification problem, Sentiment Analysis uses the evaluation metrics:

- Precision
- Recall
- F-score
- Accuracy.
- Macro
- Micro
- Weighted F-1



Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

EXPERIMENT 1:

Using the earning call transcripts of 12 companies in Quarter-4 2018, we will review some methods to evaluate sentiment analysis techniques.The goal
of this project is to build a sentiment analysis engine that can be used to understand the sentiments of the earnings call transcripts.
We attempt to build 3 models in the experiment which are as follows:

1. BOW model
2. Word embeddings (GLOVE)
3. RNN

1. BOW model:
The bag-of-words model is one of the simplest language models used in NLP. It makes an unigram model of the text by keeping track of the number of occurences of each word. This can later be used as a feature for Text Classifiers and Sentiment Analysis. In this bag-of-words model we only take individual words into account and give each word a specific subjectivity score.If the total score is negative the text will be classified as negative and if its positive the text will be classified as positive. It is simple to make, but is less accurate because it does not take the word order or grammar into account.

In experiment 1, we created the Bag of Words model. We trained the model on 91 samples and validated that on 11 samples. We got the following results:

```
Instructions for updating:
Use tf.cast instead.
Train on 91 samples, validate on 11 samples
Epoch 1/2
91/91 [==============================] - 0s 3ms/step - loss: 1.1105 - acc: 
0.4615 - val_loss: 1.1106 - val_acc: 0.4545
Epoch 2/2
91/91 [==============================] - 0s 363us/step - loss: 0.8521 - 
acc: 0.6923 - val_loss: 1.0956 - val_acc: 0.4545
```

The test score accuracies are as follows:

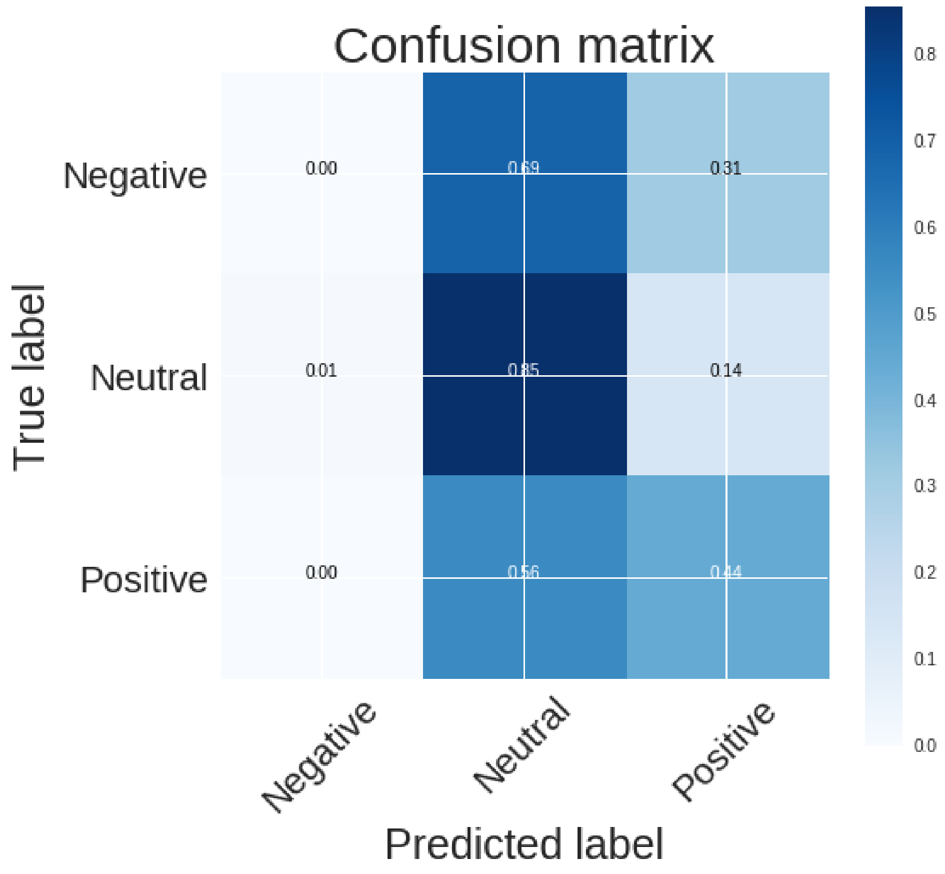
Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

```
26/26 [==============================] - 0s 77us/step
Test score: 1.0748307704925537
Test accuracy: 0.38461539149284363
```

For this model,we created the confusion matrix. The confusion matrix displays the positive,negative and neutral results.

- 44% positive results were predicted accurately
- 85% neutral results were predicted accurately
- 0% negative results were predicted accurately



Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

2. Word embeddings (GLOVE)

The GloVe system is an alternative to gensim. GloVe (“global vectors for word representation”)can be used in several ways. GloVe has several pre-built word embedding files. We used total 400000 distinct words. The dimensions of embedding matrix are 400000x50

For the Glove model we designed the following model:

```
Layer (type)                 Output Shape              Param #
=================================================================
embedding_4 (Embedding)      (None, 300, 8)            80000
_________________________________________________________________
flatten_2 (Flatten)          (None, 2400)              0
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 7203
=================================================================
Total params: 87,203
Trainable params: 87,203
Non-trainable params: 0
_________________________________________________________________
```

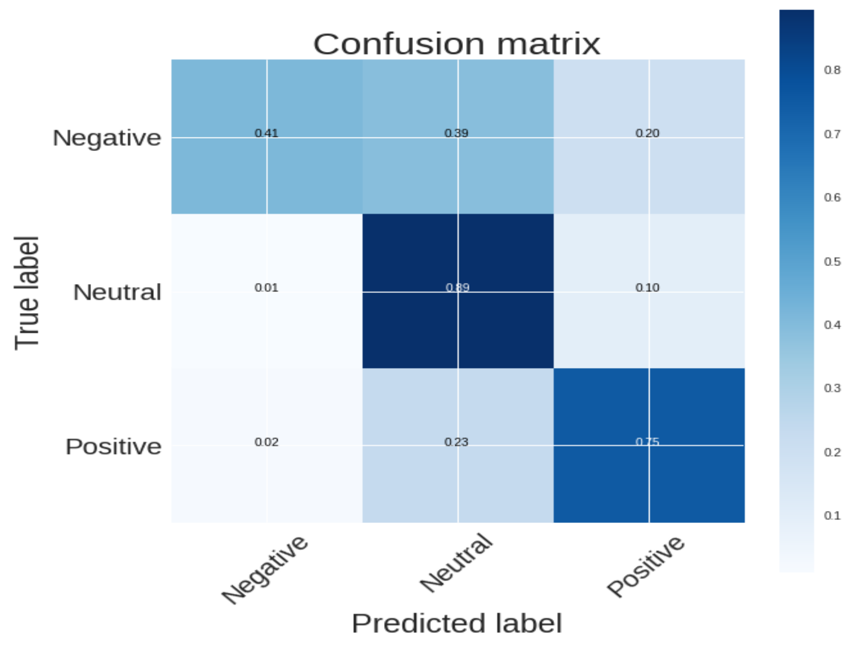
The glove model is trained on 982 samples and the results are validated on 246 samples.

After running the model, the predicted accuracies were :

```
307/307 [==============================] - 0s 99us/step
Test score: 1.2828871815523029
Test accuracy: 0.6449511637516829
```

Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale



For this model,we created the confusion matrix. The confusion matrix displays the positive,negative and neutral results.

- 75% positive results were predicted accurately
- 89% neutral results were predicted accurately
- 41% negative results were predicted accurately

3. RNN

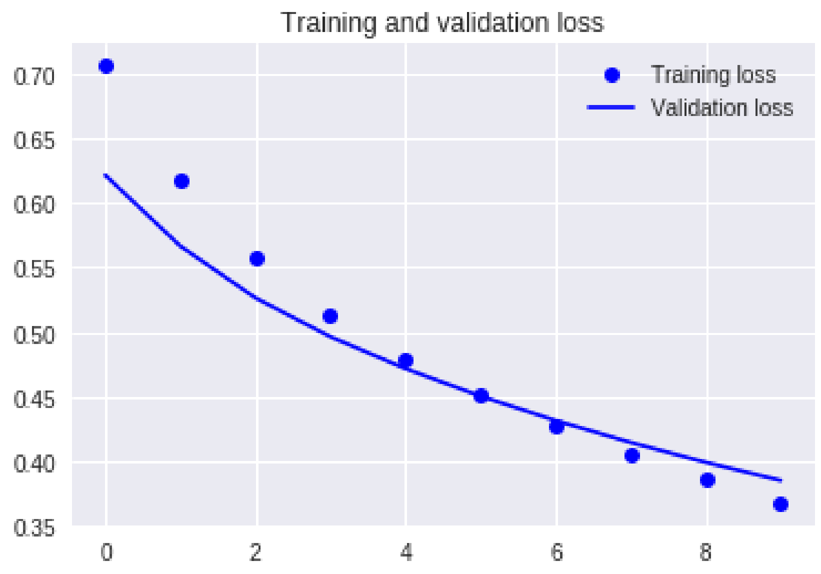
The idea behind RNNs is to make use of sequential information. In a traditional neural network we assume that all inputs (and outputs) are independent of each other. RNNs are called *recurrent* because they perform the same task for every element of a sequence, with the output being depended on the previous computations. Another way to think about RNNs is that they have a “memory” which captures information about what has been calculated so far. In theory RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps.
The Rnn model has predicted the following results:

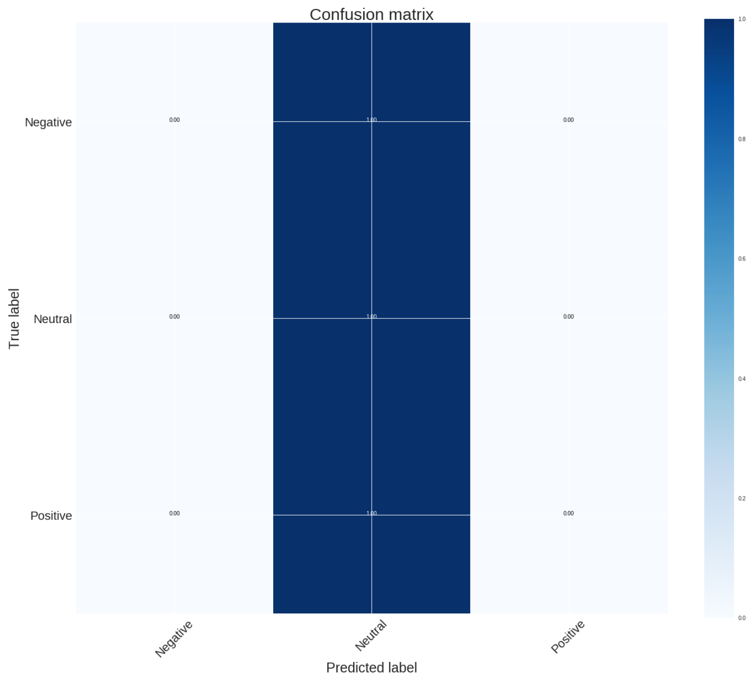
Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

We have used 82 trained samples and 21 validated samples.

```
26/26 [==============================] - 0s 3ms/step
Test score: 0.4825204014778137
Test accuracy: 0.8205128312110901
```





Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

- Based on the model predictions, confusion matrix results and accuracies, the model that performed the best was the GLOVE(word embeddings) with the test accuracy of 64% as compared to the BOW and RNN model.

EXPERIMENT 2:
Transfer Learning

- In this experiment we used Transfer Learning using the IMDB Movie review dataset and built the 3 models:BOW, GLOVE and RNN
- We computed the confusion matrix for each model and used the 3 models to predict the sentiments for the financial dataset.
- The findings of the models are as follows:

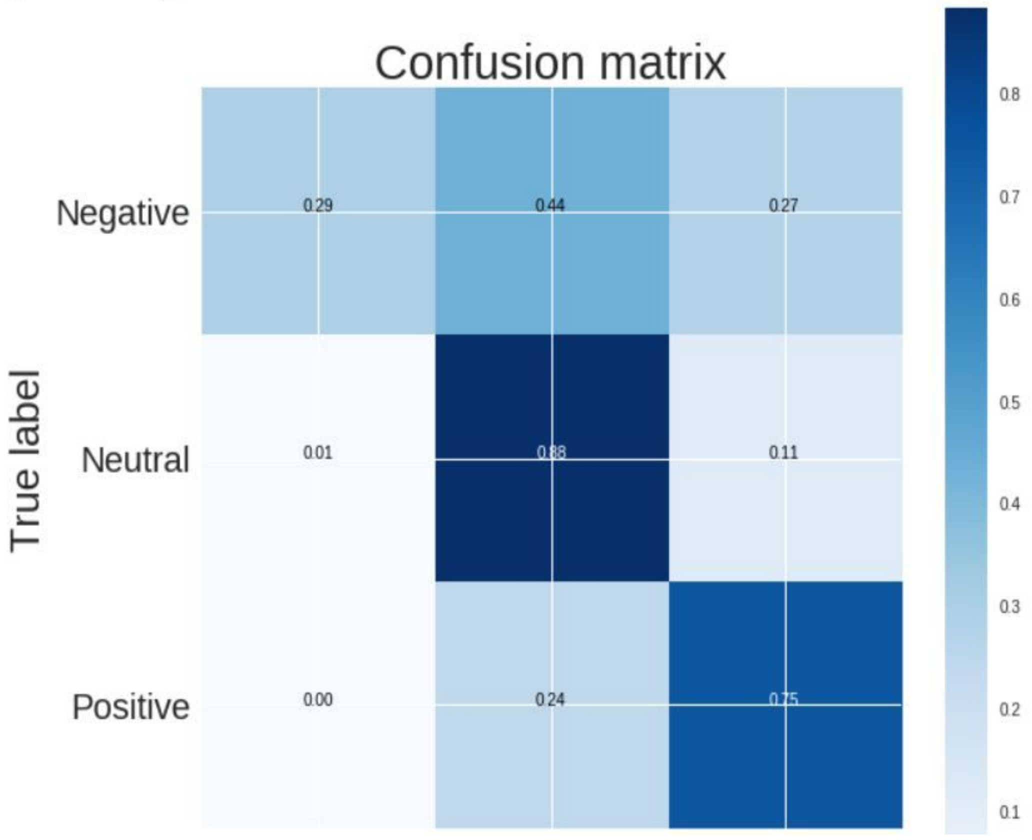
Models	Train size	Test size	Test score	Test accuracy
BOW	608	147	0.2781	0.194
GLOVE	608	147	0.642	0.805
RNN	608	147	0.551	0.805

- After performing transfer learning , we computed that based on the model predictions, confusion matrix results and accuracies, the model that performed the best was the RNN with the test accuracy of 80% as compared to the BOW and RNN model.
- The confusion matrix for the best model RNN is as follows:

Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

```
[[ 38  57  36]
 [  4 580  72]
 [  2 107 332]]
```



EXPERIMENT 3:

Using APIs

- We used the Amazon, Google, Microsoft and Watson APIs [2,3,4,5], obtain the sentiment scores for your entire dataset.
- Then we performed normalization for the scores and took the average normalized score for determining the sentiments
- Finally, we computed the confusion matrix wrt the original dataset and predicted the results

AMAZON: Using Amazon Comprehend we determined the sentiment of a document. We determined if the sentiment is positive, negative, neutral, or mixed. For example, if we can use sentiment analysis to determine the sentiments of comments on a blog posting to determine if thereaders liked the post.

Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

You can use any of the following operations to detect the sentiment of a document or a set of documents.

- DetectSentiment
- BatchDetectSentiment
- StartSentimentDetectionJob

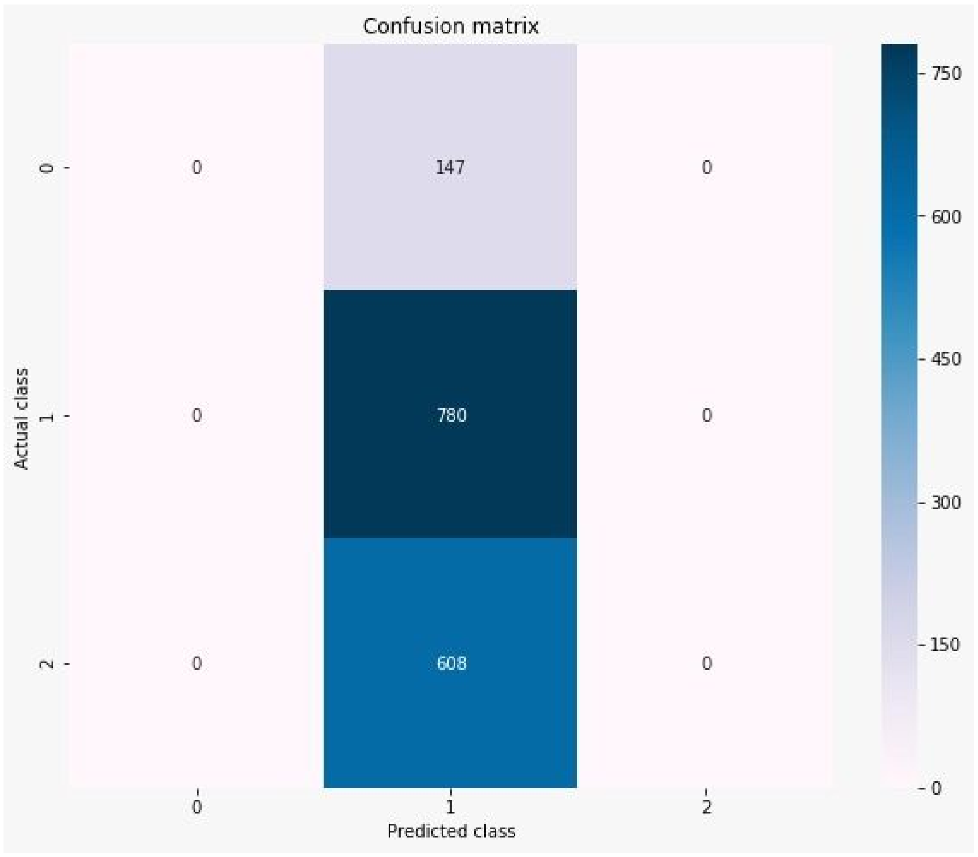
GOOGLE: We analyzed a sentiment document, using an analyzeSentiment request, which performs sentiment analysis on text. Sentiment analysis attempts to determine the overall attitude (positive or negative) and is represented by numerical score and magnitude values.

MICROSOFT :
Sentiment analysis produces a higher quality result when you give it smaller chunks of text to work on. This is opposite from key phrase extraction, which performs better on larger blocks of text. To get the best results from both operations, consider restructuring the inputs accordingly.
We had a JSON document in this format: id, text, language
The steps we followed while using Microsoft API:
Step 1: Structure the request
Step 2: Post the request
Step 3: View results

WATSON: The sentiment and context analysis block uses the IBM Watson Natural Language Understanding to analyze text and execute functions based on the sentiment or emotion of that message. This is done in an after-publish event handler. The sentiment analysis for each message is saved in the PubNub distributed data store. The session sentiment is calculated as an average and is published on an output sentiment channel.

Among all the API's used, IBM Watson performed better since normalized results were aligned with the raw dataset.

Following is the confusion matrix:



EXPERIMENT 4:

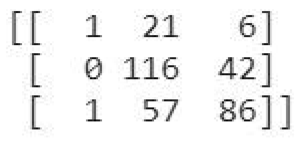
Ensemble learning using AutoML

- In this experiment, we build a model that can map the raw(not normalized) results from the 4 APIs to the outputs labeled. i.e Inputs: Amazon, Google, IBM, Microsoft scores

Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

- In order to find the best model we used TPOT, AutoSKLearn, H2O.ai’s APIs for choosing the best model.
- H2O predicted the accurate values and along with that it handled the null values on its own
- Also, H2O gave positive, negative and neutral results for each data and it took less time for training the data.
- The confusion matrix for the experiment 4 is as follows:



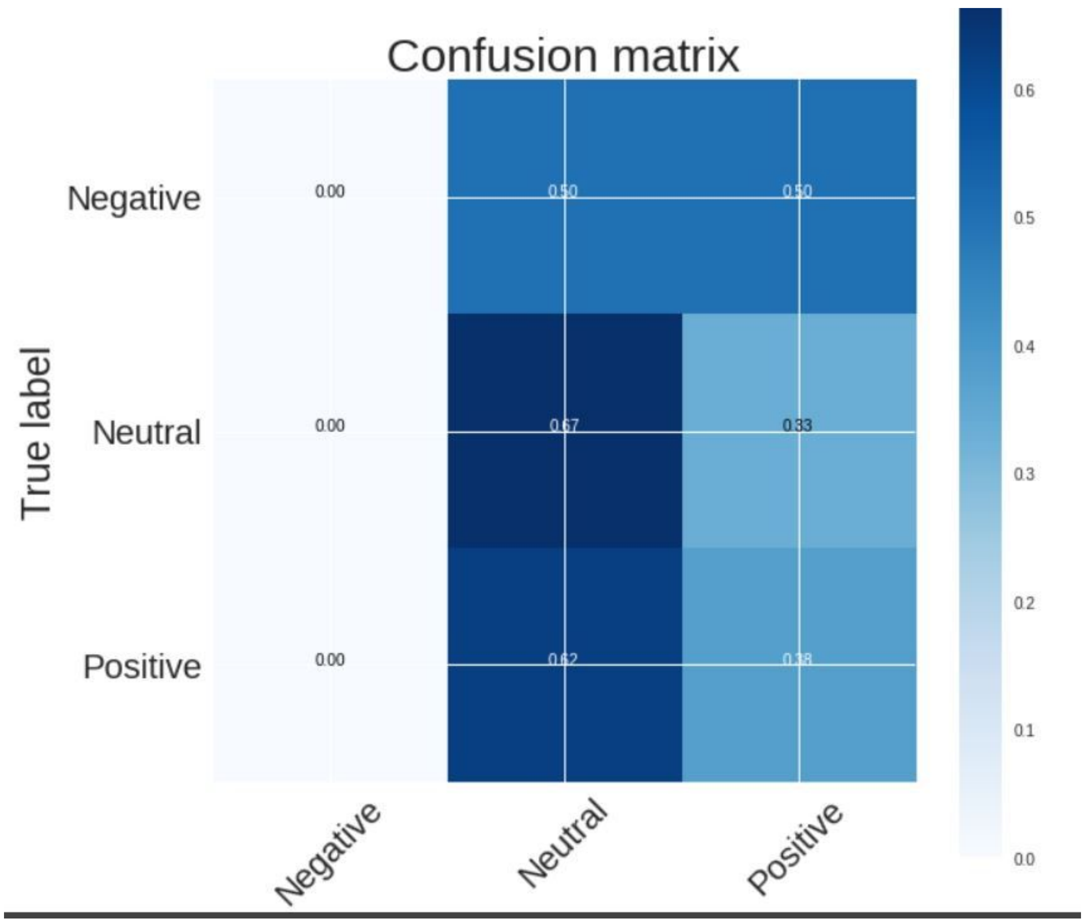
```
[[  1  21   6]
 [  0 116  42]
 [  1  57  86]]
```

FINAL MODEL:

- For the final model, we created the earnings call transcript for GE
- After creating the transcripts we parsed and formatted the file in the json format
- The next step we performed was reading the transcript and labeling the paragraphs with (+/-/neutral) sentiments.
- Then we passed this file to the final model which was (GLOVE)word embeddings. This model we selected gave better accuracy and the model used maximum words from the data. We got an accuracy of nearly 68% on the test dataset.
- The confusion matrix for the (GLOVE)word embeddings :

Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale



From the above results, we researched that the model which gave best results and performed well with the API's and test data was GLOVE(word embeddings) model.Glove, Word2vec worked well "due to certain system design choices and hyperparameter optimizations, rather than the embedding algorithms themselves" Word2vec is that it can give fairly good performance/accuracy with a range of parameters(dimension(100–500), negative sampling(5–10), epochs(5–10), window size(5–10)) across the size of the corpus.

Assignment 3: Uncovering Sentiments using EDGAR Datasets Report

Dhanashri Palodkar Parag Bhingarkar Rajeshree Kale

The experiment 4 can be recommended for the production because in that experiment we can use multiple API’s and get multiple sentiment scores. Also, it works like ensemble learning instead of relying on one model we can get accumulative sentiment scores of four models together provided that the data is normalized.