Challenges with Detecting Sarcasm in Natural Language (text)

Harsha Vardhan, Khurdula.

Suchir Santosh, Naik.

Advanced Topics in Natural Language Processing.

Dr. Russert.

Introduction

One of the most important challenges that we had while attempting to develop an Artificial Intelligence, or a Machine Learning Model was the fact that *NONE* of the trained models for one particular domain would work well when attempting to test for detecting sarcastic text from other domains.

This report attempts to justify a work around that issue. The problem at hand with trained instances or models of one domain say Reddit or Twitter is that all of the text on each of these platforms are unique. The way they are written and the way sarcasm is conveyed. If we attempt to just fit a model over one large corpora of text from say Reddit, then it does not work well while trying to detect Sarcasm on Twitter. One way around this would be Data Engineering and Building. This seems to be the solution.

Methodology

The proposed methodology is to combine text from each of the platform under consideration (Twitter, Reddit, & News Headlines) which would give us the chance to cover much ground and establish a more widened definition of Sarcasm, in-order to call a text as Sarcastic in nature.

The key to fitting decent models for this task is data engineering or building, the following tasks were performed by me to create the datasets that would be used for model fitting.

1. Normalization:

Normalization involves convergence the syntax of text within a specific format that makes the text less noisy and more distinct. The following steps were carried out for text normalization:

- Case Conversion: The entire text is transformed into lower case for case consistency.
- Apostrophe Deletion: Words like "*can't"*, "*won't"* are converted into "*cant", etc.*
- Word Extraction: Within a sentence only words with alphabets are extracted, rest are discarded, for example: punctuation and numbers are discarded during this process.

This process of Normalization has been automated by a module called *Normalizer()* which performs all the above by just a simple method call and a lot more.

2. Text Representation:

Usually text cannot be passed as in its original form to a Machine Learning or Artificial Intelligence model that expects text as an input. There has to be a way in which we can actually pass the text input, by representing those words in some way. For the current problem at hand we have used the following Text Representations:

- TF-IDF: TD-IDF stands for Term Frequency Inverse Document Frequency, which basically tells us how important is word is to a document (paragraph) within a collection of documents (collection of statements or paragraphs.). The TF part gives the frequency of a word within document, and IDF part reduces the weight for heavily used words across all documents. It generates an ***n*m matrix*** of large size, giving the ***tf-idf matrix***.
- Word Embeddings: They are one of the most popularly used word representation that allows words with similar meaning to be identified with similar representation. We specifically use ***Word2Vec*** method for generating these word embeddings. These are great to understand some ***context between lines of text***.

3. Fitting a Model:

Fitting a Machine Learning Model/ a Deep Neural Network is quite an intriguing task for this task. Each model/NN, is fitted on one of these datasets:

- Twitter.csv: This dataset contains text scrapped on twitter threads. The scrapped tweets mostly have **#sarcasm** tag associated with them, such threads are available within this dataset.
- Reddit.csv: This dataset is a subset of the original dataset available on **Kaggle** called the *Reddit_Sarcasm* dataset, which contains a corpora of user comments and their reply to a comment. This dataset was scraped using tagging (<s> hello </s>) that would determine the intention of the author of the message/comment that they are trying to be Sarcastic.
- News Headlines.csv: This is by far the most promising dataset, as the headlines dataset are more complete than the above, the headlines cover more context and have lesser external context requirement, as the headlines are written in such a way that the

reader should be able to understand it, without or without knowing what the author might be talking about.

- Twitter-News.csv: As the name suggest this dataset contains a mix of samples from both twitter and news headlines.
- Combined-Set.csv: This basically a global set containing text from all three domains/social platforms.

Notes:

- Each of these datasets are balanced. (They contain equal no. of sarcastic and non-sarcastic samples.)
- Each of the original datasets: Twitter, Reddit and News Headlines have ***20,000*** samples within them, and all of these sets are balanced.
- The Twitter-News.csv contains ***40,000*** samples with ***20,000*** samples being sarcastic and ***20,000*** being non-sarcastic in nature.
- The remaining samples that are discarded during production of these curated datasets can be used at later stages for more rigorous testing of the hypothesis that combining different styles of sarcastic texts would help us better represent/detect Sarcasm within models.
- Each of the Learning Algorithm is fitted with each of the above mentioned datasets, and their performances are keenly noted in due to prove our hypothesis.
- The testing data is held out from the training data using hold out method maintaining balance within the testing samples as well, and shuffling them for test.
- The model metrics are then noted to give their accuracy, precision, f1-score, recall, confusion matric, roc_auc curve.
- Lastly this model and its corresponding input Vectorizer are saved locally.

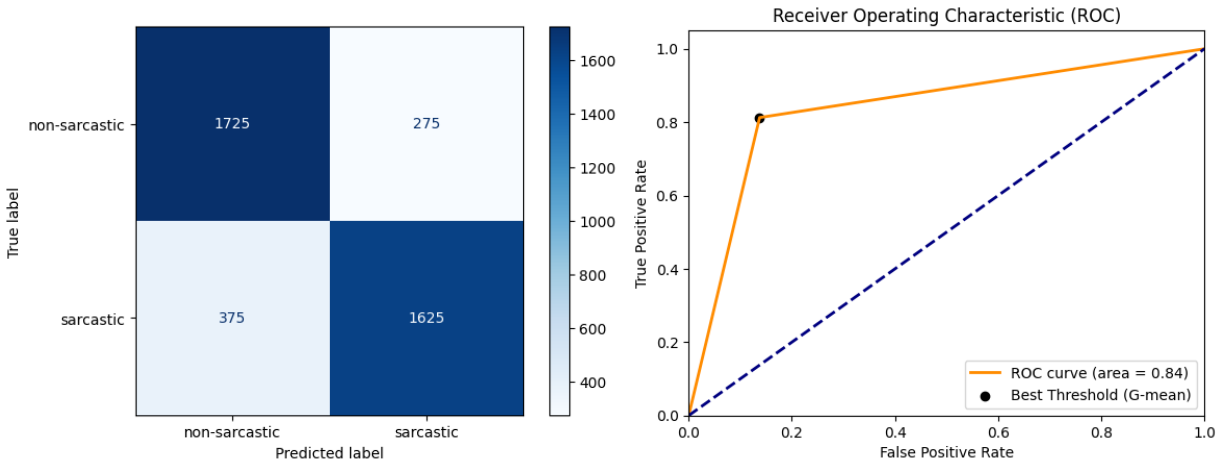
Results

Naïve Bayes

1. Naïve Bayes over News Headlines dataset:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.82**	**0.86**	**0.84**	**2000**
Sarcastic	**0.86**	**0.81**	**0.83**	**2000**
Average	**0.84**	**0.84**	**0.84**	**4000**
Accuracy:			**0.840**	

Looking at the above metric table we can say that it did a decent job as expected over our best set at the moment.



The above image represents the Confusion Matrix and ROC_AUC for Naïve Bayes over News headlines.

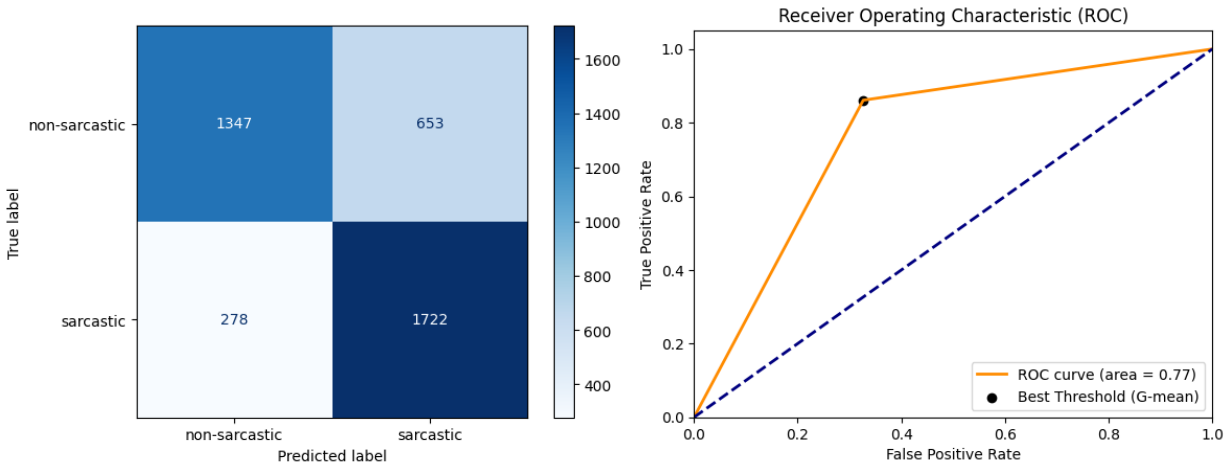
Additionally we also performed Stratified K-fold cross validation to validate the model and its metrics to avoid under-fitting or over-fit on one type of samples.

Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.8255**
Fold II	**0.8327**
Fold III	**0.8297**
Fold IV	**0.8312**
Fold V	**0.8400**
Mean Accuracy	**0.8318**

2. Naïve Bayes over Twitter Dataset:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.83**	**0.67**	**0.74**	**2000**
Sarcastic	**0.73**	**0.86**	**0.79**	**2000**
Average	**0.78**	**0.77**	**0.77**	**4000**
Accuracy:			**0.770**	

The above are the observed metrics for Naïve Bayes over Twitter Set.



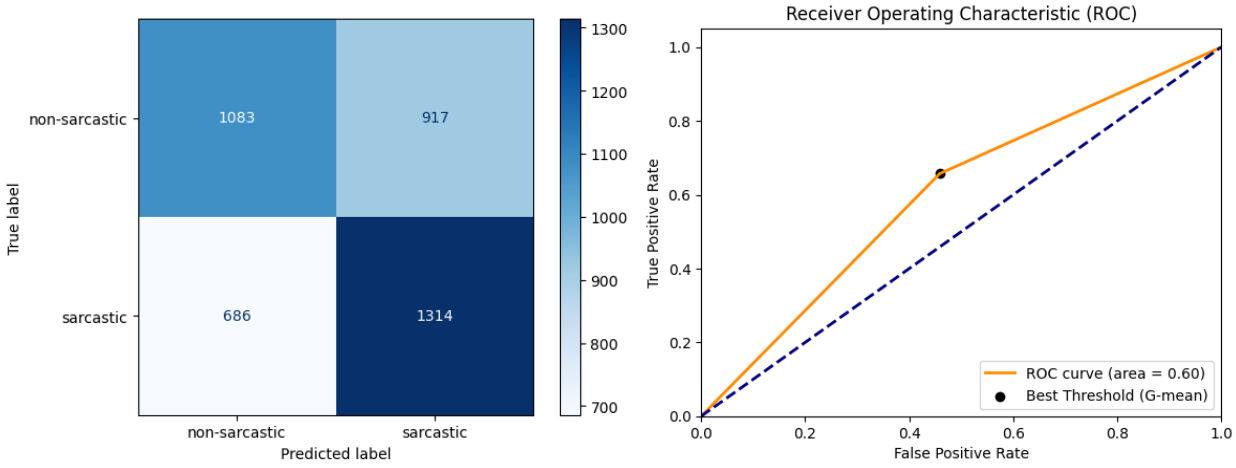
Confusion Matrix and ROC_AUC for Naïve Bayes over Twitter Dataset.

Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.7687**
Fold II	**0.7505**
Fold III	**0.7437**
Fold IV	**0.7547**
Fold V	**0.7500**
Mean Accuracy	**0.753**

3. Naïve Bayes for Reddit Data:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.61**	**0.54**	**0.57**	**2000**
Sarcastic	**0.59**	**0.66**	**0.62**	**2000**
Average	**0.60**	**0.60**	**0.60**	**4000**
Accuracy:			**0.60**	

The above metrics are observed for Naïve bayes over Reddit Dataset



Confusion Matrix and ROC_AUC curve for Naïve Bayes over Reddit Dataset.

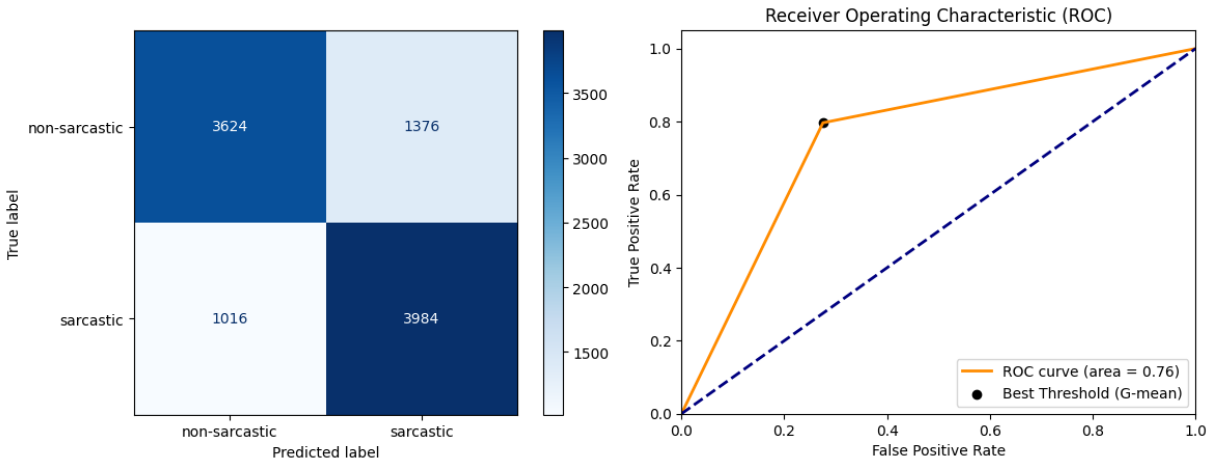
Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.6015**
Fold II	**0.5915**
Fold III	**0.6037**
Fold IV	**0.6110**
Fold V	**0.6025**
Mean Accuracy	**0.6020**

The above are the metrics cross validation metrics for Naïve Bayes over Reddit Data

4. Naïve Bayes over Twitter-News dataset:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.78**	**0.72**	**0.75**	**5000**
Sarcastic	**0.74**	**0.80**	**0.77**	**5000**
Average	**0.76**	**0.76**	**0.76**	**10000**
Accuracy:			**0.76**	

The above are metrics observed for Naïve Bayes over Twitter-News Dataset



Confusion Matrix and ROC_AUC curve for Naïve Bayes over Twitter-News dataset.

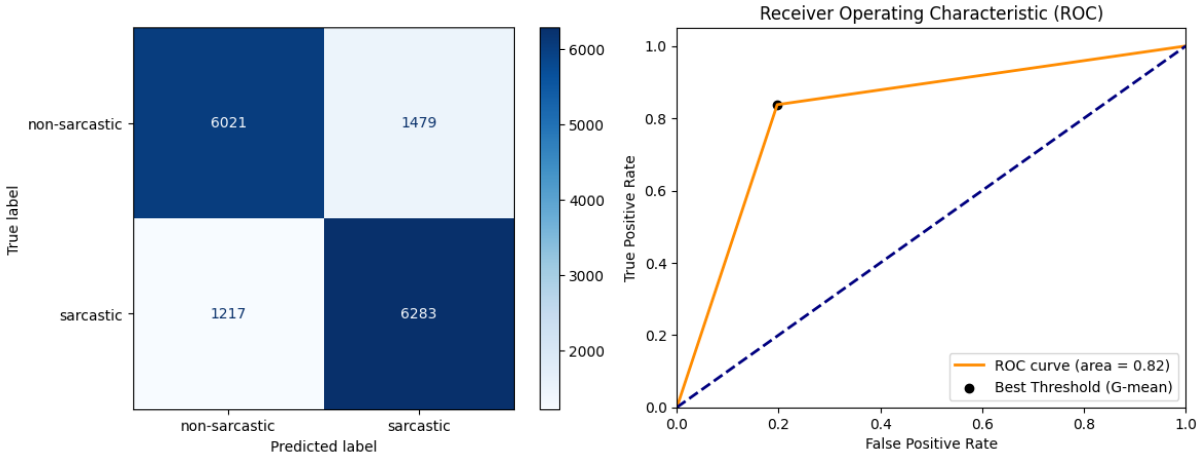
Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.7598**
Fold II	**0.7591**
Fold III	**0.7578**
Fold IV	**0.7593**
Fold V	**0.7621**
Mean Accuracy	**0.7596**

The above are Cross Validation scores for Naïve Bayes over Twitter-News dataset.

5. Naïve Bayes over Combined Dataset:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.83**	**0.80**	**0.82**	**7500**
Sarcastic	**0.81**	**0.84**	**0.82**	**7500**
Average	**0.82**	**0.82**	**0.82**	**15000**
Accuracy:			**0.82**	

The above table denotes the metrics observed for Naïve Bayes over Combined Set.



Confusion Matrix and ROC_AUC curve for Naïve Bayes over Combined Entire set.

Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.8186**
Fold II	**0.8245**
Fold III	**0.8242**
Fold IV	**0.8168**
Fold V	**0.8195**
Mean Accuracy	**0.8207**

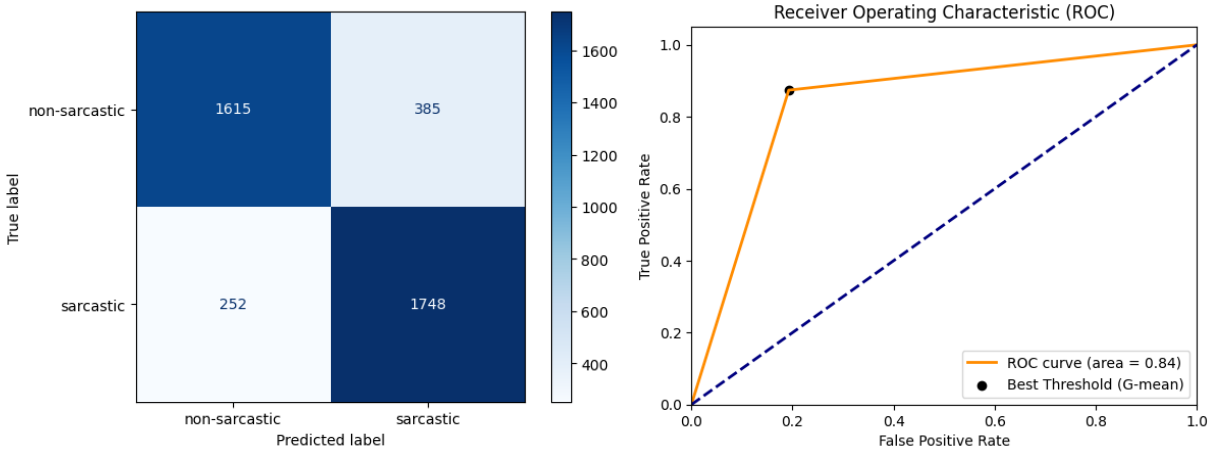
The above scores are observed for cross validation for Naïve Bayes over Combined Dataset.

Logistic Regression

1. Logistic Regression over News Headlines:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.87**	**0.81**	**0.84**	**2000**
Sarcastic	**0.82**	**0.87**	**0.85**	**2000**
Average	**0.84**	**0.84**	**0.84**	**4000**
Accuracy:			**0.84**	

The above metrics are observed for Logistic Regression over News Headlines Dataset.



Confusion Matrix and ROC_AUC curve for Logistic Regression over News Headlines Dataset.

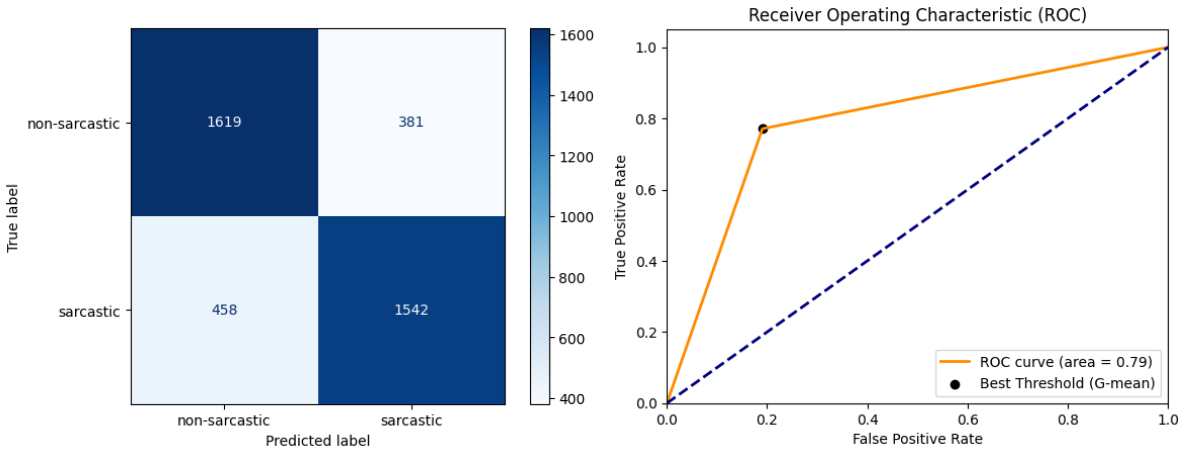
Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.8377**
Fold II	**0.8330**
Fold III	**0.8315**
Fold IV	**0.8300**
Fold V	**0.8427**
Mean Accuracy	**0.835**

These are the cross validation scores for LR over News Headlines Dataset.

2. Logistic Regression over Twitter Dataset:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.78**	**0.81**	**0.79**	**2000**
Sarcastic	**0.80**	**0.77**	**0.79**	**2000**
Average	**0.79**	**0.79**	**0.79**	**4000**
Accuracy:			**0.79**	

The above metrics are for Logistic Regression over Twitter Dataset.



Confusion Matrix and ROC_AUC curve for LR over Twitter Set.

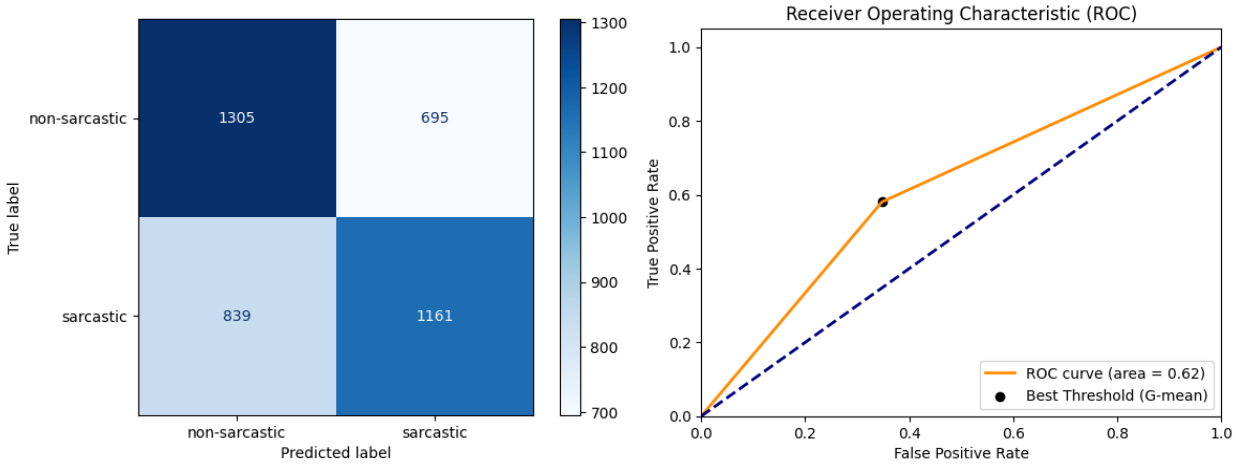
Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.7902**
Fold II	**0.7700**
Fold III	**0.7832**
Fold IV	**0.7665**
Fold V	**0.7815**
Mean Accuracy	**0.7782**

These are the Cross Validation Scores for Logistic Regression over Twitter Dataset.

3. Logistic Regression over Reddit Data.

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.61**	**0.65**	**0.63**	**2000**
Sarcastic	**0.63**	**0.58**	**0.60**	**2000**
Average	**0.62**	**0.62**	**0.62**	**4000**
Accuracy:			**0.62**	

These are the metrics observed for Logistic Regression over Reddit.



Confusion Matrix and ROC_AUC curve for Logistic Regression over Reddit Data.

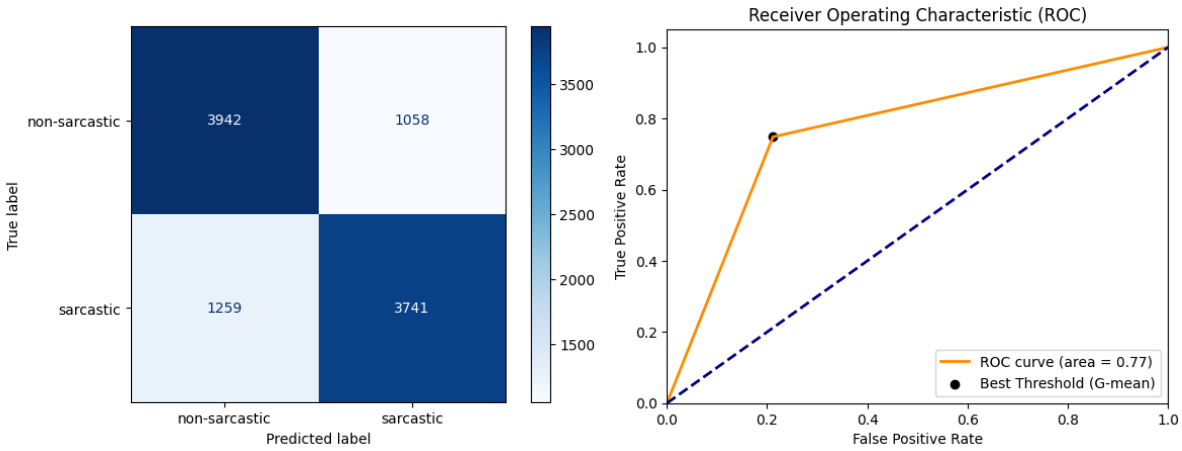
Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.6085**
Fold II	**0.6187**
Fold III	**0.6250**
Fold IV	**0.6315**
Fold V	**0.6218**
Mean Accuracy	**0.6211**

5-Fold Cross Validation for Logistic Regression over Reddit

4. Logistic Regression Over Twitter-News Dataset:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.76**	**0.79**	**0.77**	**5000**
Sarcastic	**0.78**	**0.75**	**0.76**	**5000**
Average	**0.77**	**0.77**	**0.77**	**10000**
Accuracy:			**0.77**	

These are metrics observed for Logistic Regression over Twitter News Data.



Confusion Matrix and ROC_AUC curve for Logistic Regression over Twitter-News Set.

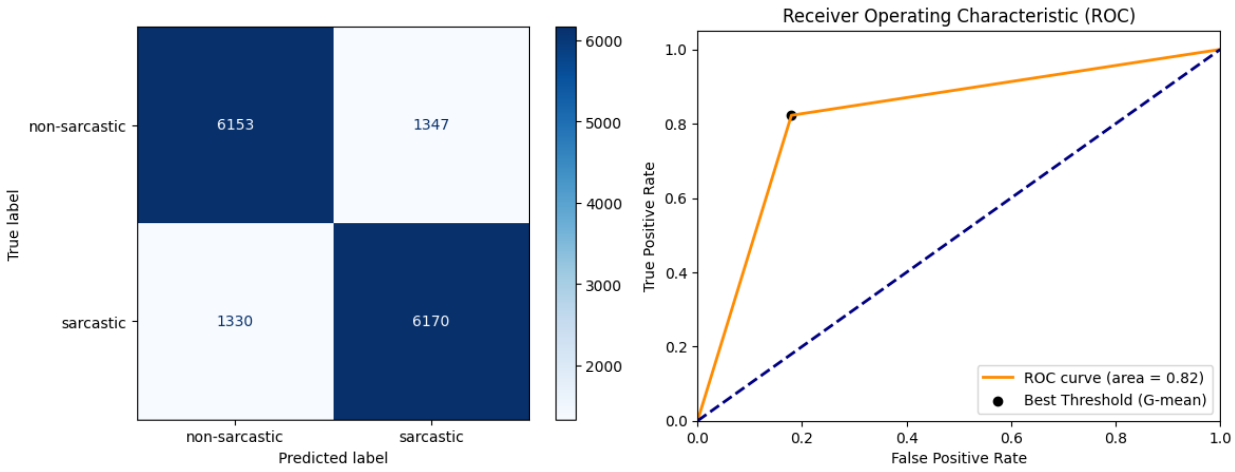
Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.7693**
Fold II	**0.77776**
Fold III	**0.7657**
Fold IV	**0.7796**
Fold V	**0.7751**
Mean Accuracy	**0.7350**

5-Fold Cross Validation scores for Logistic Regression over Twitter-News set.

5. Logistic Regression over Combined Dataset:

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.82**	**0.82**	**0.82**	**7500**
Sarcastic	**0.82**	**0.82**	**0.82**	**7500**
Average	**0.82**	**0.82**	**0.82**	**15000**
Accuracy:			**0.82**	

Logistic Regression Metrics over Combined Dataset.



Confusion Matrix and ROC_AUC curve for Logistic Regression Combined Dataset.

Stratified 5-Fold Cross Validation for Naïve Bayes	
Fold I	**0.825**
Fold II	**0.835**
Fold III	**0.831**
Fold IV	**0.820**
Fold V	**0.822**
Mean Accuracy	**0.83**

5-Fold Cross validation Scores for Logistic Regression over Combined Dataset.

Multi-Layer Perceptron

1. Multi-Layer Perceptron Over News Headlines:

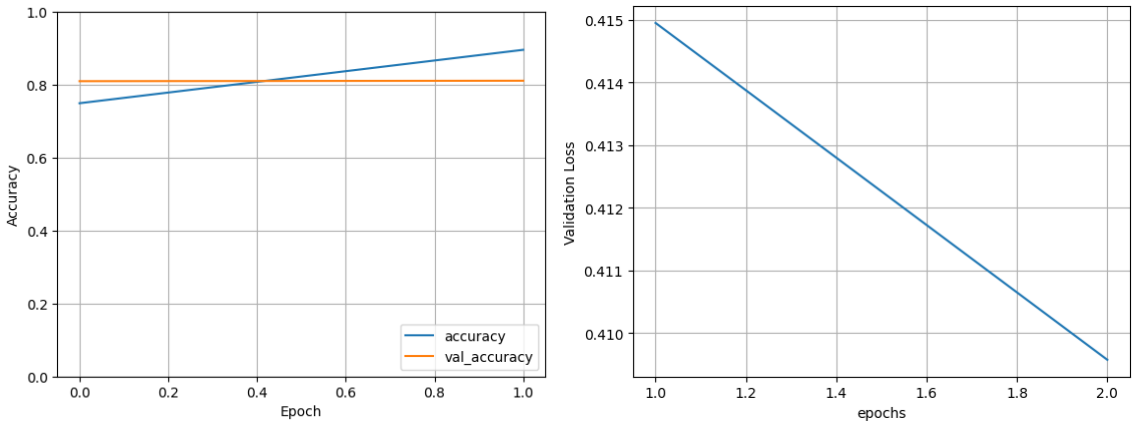
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 256)               5790464

 dense_1 (Dense)             (None, 128)               32896

 dense_2 (Dense)             (None, 128)               16512

 dense_3 (Dense)             (None, 1)                 129

=================================================================
Total params: 5840001 (22.28 MB)
Trainable params: 5840001 (22.28 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

This is the Multi-Layer Perceptron Architecture for News Headlines Dataset.

Training Constants	
Train Set size	**70%**
Validation Set size	**10%**
Test Set size	**20%**

Epochs	2
Batch Size	128
Mean Accuracy	0.83

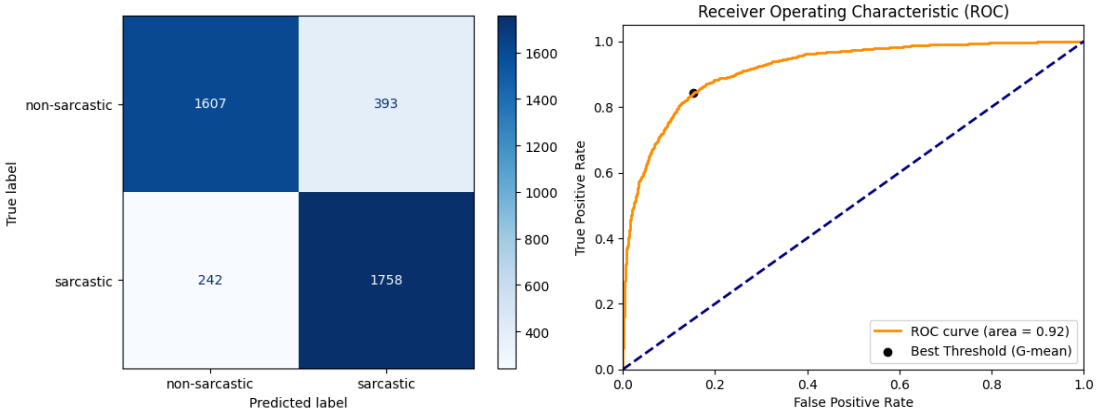
These constants have been used to Fit the MLP.



Training and Losses Comparison for Train and Validation set.

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	0.87	0.80	0.84	2000
Sarcastic	0.82	0.88	0.85	2000
Average	0.84	0.84	0.84	4000
Accuracy:			0.84	

These metrics are observed for a Multi-Layer Perceptron over News Headlines Test set.



Confusion Matrix and ROC_AUC Curve for MLP over News Headlines Dataset.

Stratified 5-Fold Cross Validation for Multi-Layer Perceptron	
Fold I	**0.8372**
Fold II	**0.8324**
Fold III	**0.8327**
Fold IV	**0.8280**
Fold V	**0.8427**
Mean Accuracy	**0.83**

These are the Cross Validation scores for MLP over News Headlines Dataset

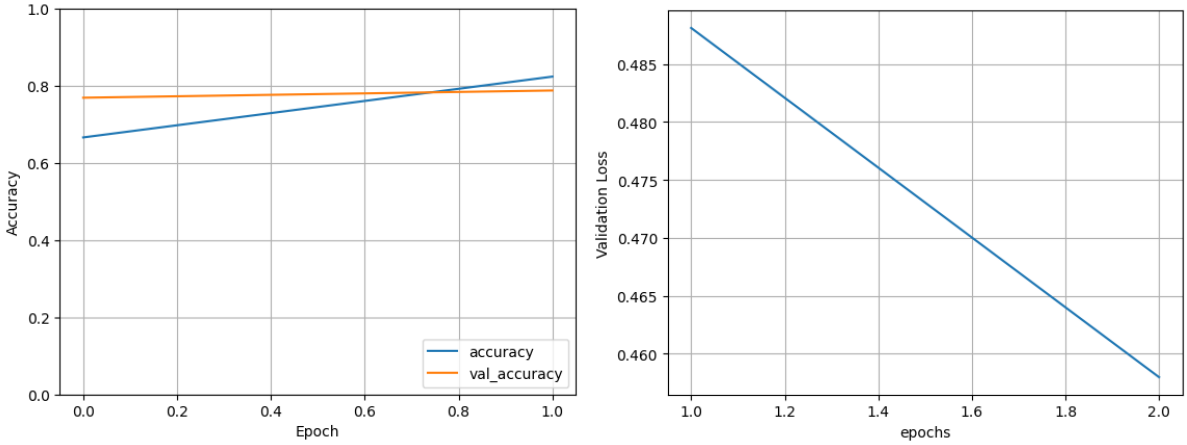
2. Multi Layer Perceptron Over Twitter Dataset:

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_24 (Dense)            (None, 256)               5681152

 dense_25 (Dense)            (None, 128)               32896

 dense_26 (Dense)            (None, 128)               16512

 dense_27 (Dense)            (None, 1)                 129

=================================================================
Total params: 5730689 (21.86 MB)
Trainable params: 5730689 (21.86 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Training Constants	
Train Set size	**70%**

Validation Set size	**10%**
Test Set size	**20%**
Epochs	**2**
Batch Size	**128**

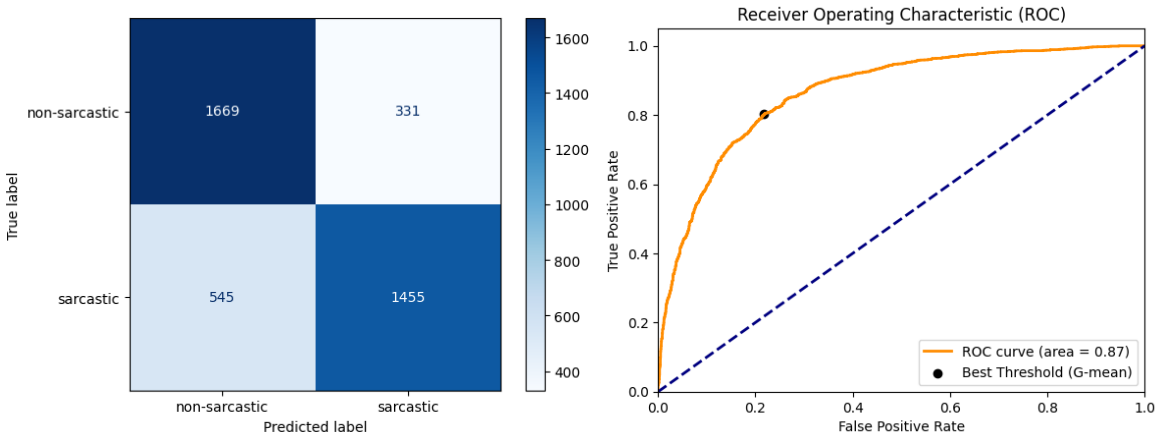
These are the constants used for fitting Multilayer Perceptron Over Twitter Set



Training + Validation Accuracies and Losses over Epochs

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.75**	**0.83**	**0.79**	**2000**
Sarcastic	**0.81**	**0.72**	**0.77**	**2000**
Average	**0.78**	**0.78**	**0.78**	**4000**
Accuracy:			**0.78**	

These are the test results for Twitter Set



Confusion Matrix and ROC_AUC curve for MultiLayer Perceptron Over Twitter Set.

3. MultiLayer Perceptron over Reddit Data:

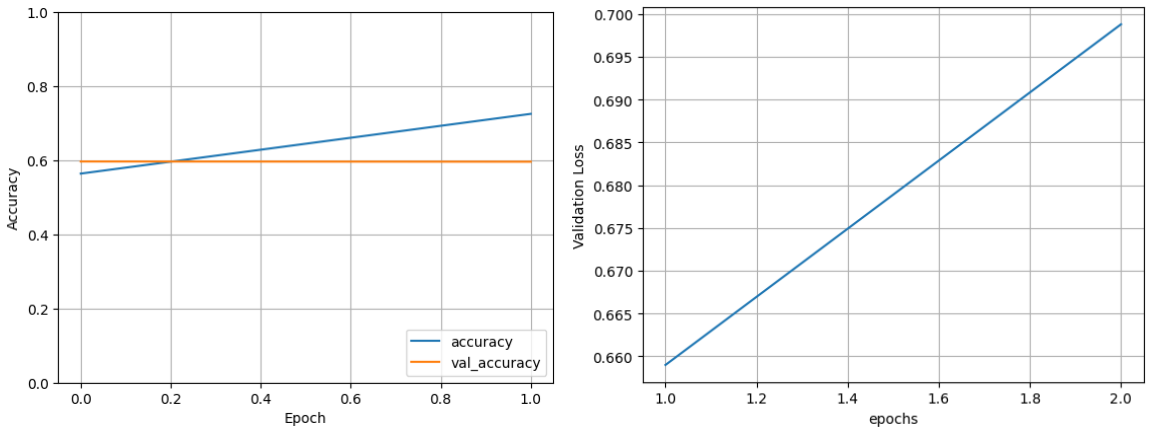
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_48 (Dense)            (None, 256)               10724864

 dense_49 (Dense)            (None, 128)               32896

 dense_50 (Dense)            (None, 128)               16512

 dense_51 (Dense)            (None, 1)                 129

=================================================================
Total params: 10774401 (41.10 MB)
Trainable params: 10774401 (41.10 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

This is the model architecture for MLP over Reddit Data.

Training Constants	
Train Set size	**70%**
Validation Set size	**10%**
Test Set size	**20%**

Epochs	2
Batch Size	64

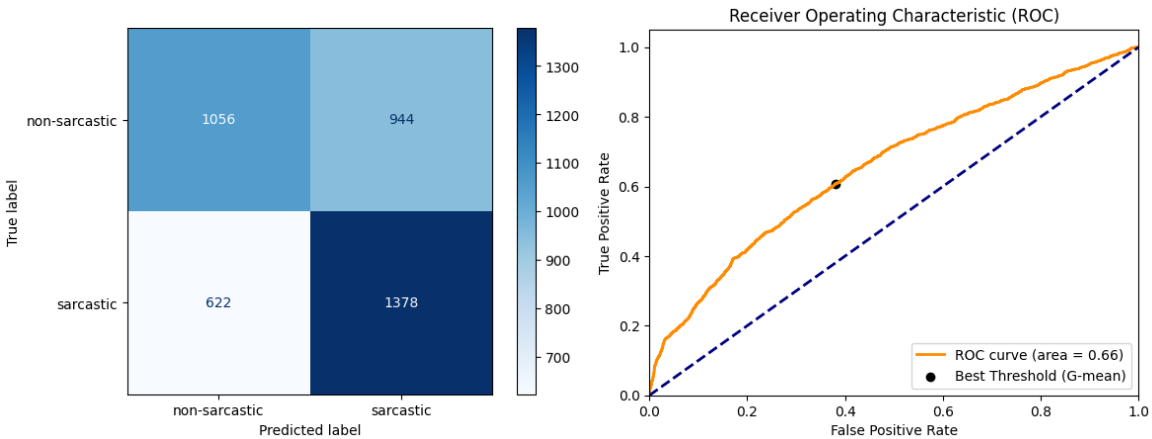
These are the constants used to Fit the MLP over Reddit Data



Training, Validation: Accuracies and Losses for MLP over Reddit Dataset.

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	0.63	0.53	0.57	2000
Sarcastic	0.59	0.69	0.64	2000
Average	0.61	0.61	0.61	4000
Accuracy:			0.61	

These are the results obtained for Test over MLP for Reddit Data.



Confusion Matrix and ROC_AUC curve for MLP over Reddit Data.

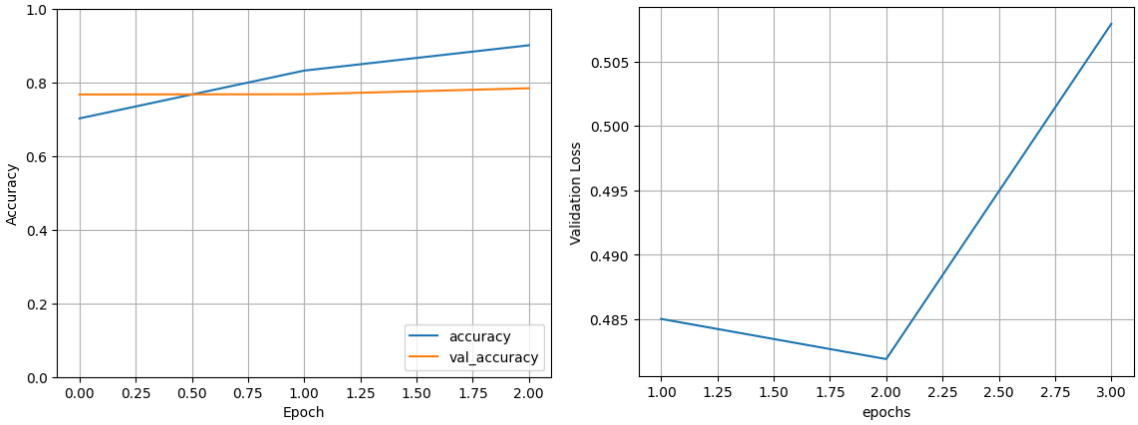
4. Multi Layer Perceptron over TwitterNewsDataset:

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_12 (Dense)            (None, 256)               8995328

 dense_13 (Dense)            (None, 128)               32896

 dense_14 (Dense)            (None, 128)               16512

 dense_15 (Dense)            (None, 1)                 129

=================================================================
Total params: 9044865 (34.50 MB)
Trainable params: 9044865 (34.50 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

This is the model architecture for MLP over Twitter+News Headings Dataset

Training Constants	
Train Set size	**70%**
Validation Set size	**10%**
Test Set size	**20%**
Epochs	**3**
Batch Size	**128**

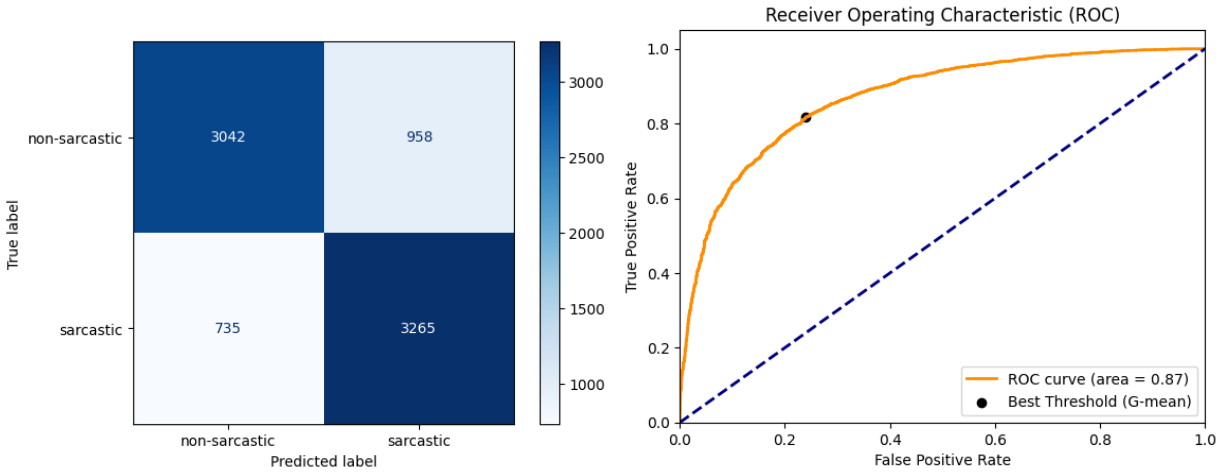
These were the training constraints that were used for MLP over Twitter+News Data.



Training, Validation: Accuracies and Losses for MLP over Twitter+news dataset.

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.63**	**0.53**	**0.57**	**2000**
Sarcastic	**0.59**	**0.69**	**0.64**	**2000**
Average	**0.61**	**0.61**	**0.61**	**4000**
Accuracy:			**0.61**	

These were the results that were obtained by MLP.



Confusion Matrix and ROC_auc for MultiLayer Perceptron over Twitter+News Dataset

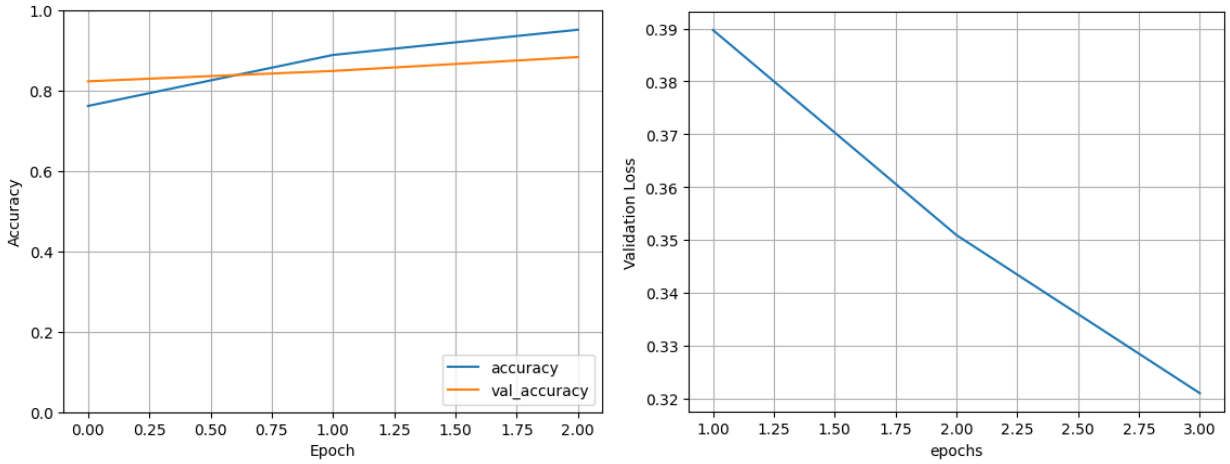
5. MultiLayer Perceptron Over Combined Dataset:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 256)               8995328

 dense_1 (Dense)             (None, 128)               32896

 dense_2 (Dense)             (None, 128)               16512

 dense_3 (Dense)             (None, 1)                 129

=================================================================
Total params: 9044865 (34.50 MB)
Trainable params: 9044865 (34.50 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

This is the MLP architecture over Combined Dataset.

Training Constants	
Train Set size	**70%**
Validation Set size	**10%**
Test Set size	**20%**
Epochs	**3**
Batch Size	**128**

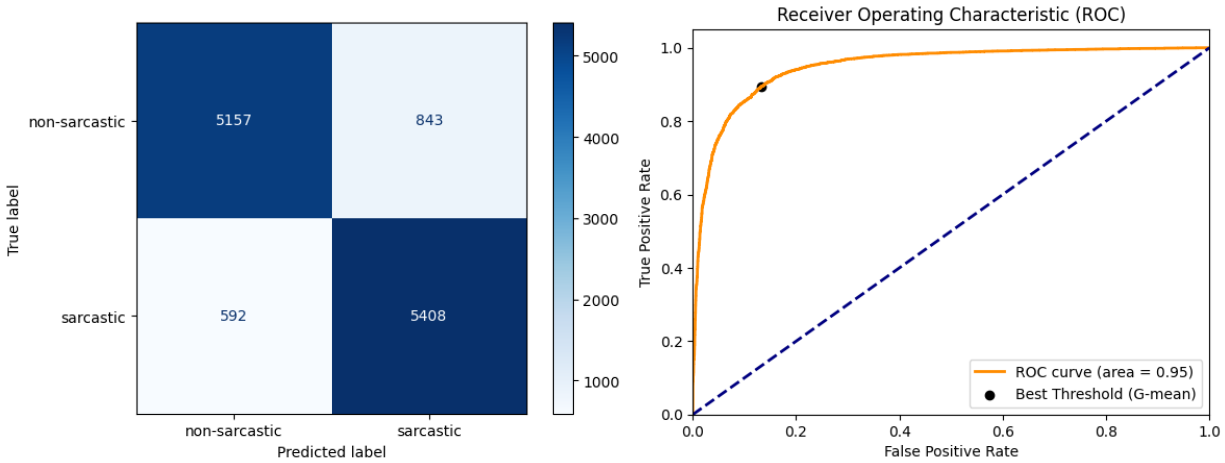
These Training Constraints Were used to fit the MultiLayer Perceptron over Combined Data.



Training and Validation: Accuracies and Losses for MLP.

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.90**	**0.86**	**0.88**	**6000**
Sarcastic	**0.87**	**0.90**	**0.88**	**6000**
Average	**0.88**	**0.88**	**0.88**	**12000**
Accuracy:			**0.88**	

These were the results that are obtained for mlp over combined set



Confusion Matric and ROC_AUC curve for Multi Layer Perceptron over Combined Set.

Bi-Directional LSTM

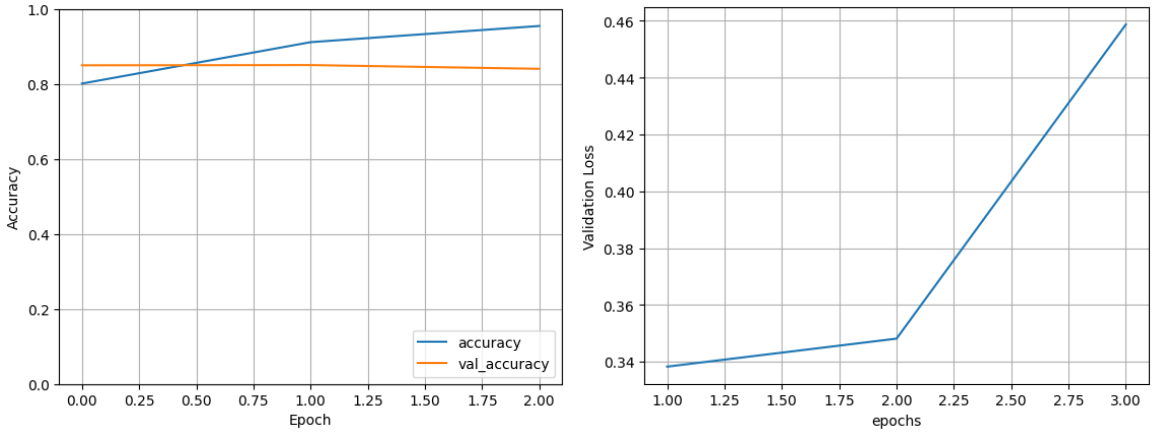
1. Bi-LSTM for News Headlines:

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, 35, 128)           1280000

 bidirectional_2 (Bidirecti  (None, 128)               98816
 onal)

 dense_2 (Dense)             (None, 1)                 129

=================================================================
Total params: 1378945 (5.26 MB)
Trainable params: 1378945 (5.26 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

This is the model architecture for the Bi-LSTM over News dataset.

Training Constants	
Train Set size	**70%**
Validation Set size	**10%**
Test Set size	**20%**
Epochs	**3**
Batch Size	**128**

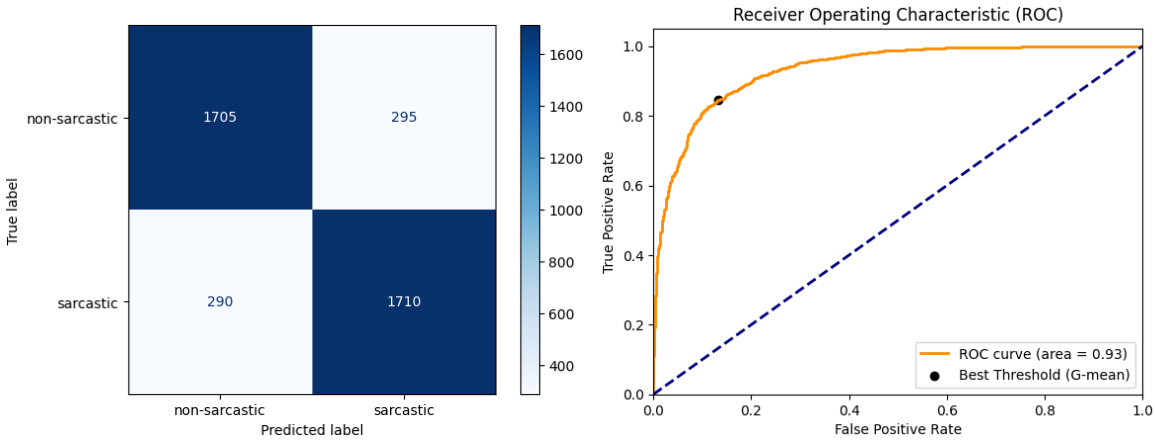
These are the training constraints on which the model was trained.



Training, Validation: Accuracies and Losses over epochs for News Data.

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.85**	**0.85**	**0.85**	**2000**
Sarcastic	**0.85**	**0.85**	**0.85**	**2000**
Average	**0.85**	**0.85**	**0.85**	**4000**
Accuracy:			**0.85**	

These are the test results for Bi-LSTM over News data



Confusion Matrix and ROC_AUC curve for Bi-LSTM over News Headlines Dataset.

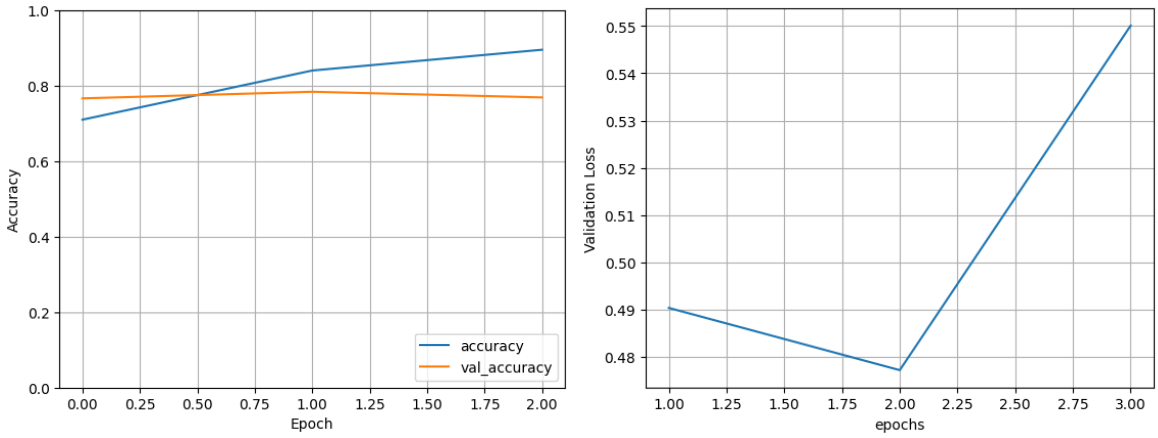
2. Bi-LSTM for Twitter Dataset:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 41, 128)           1280000

 bidirectional (Bidirection  (None, 128)               98816
 al)

 dense (Dense)               (None, 1)                 129

=================================================================
Total params: 1378945 (5.26 MB)
Trainable params: 1378945 (5.26 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

This is the model architecture for Bi-LSTM over Twitter Dataset

Training Constants	
Train Set size	**70%**
Validation Set size	**10%**
Test Set size	**20%**
Epochs	**3**
Batch Size	**32**

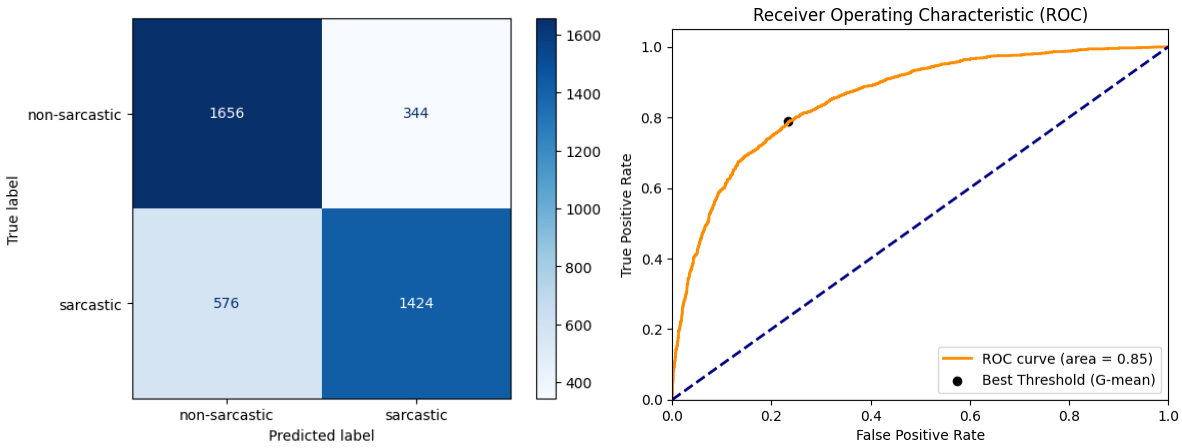
These were the training constraints that were followed for Bi-LSTM



Training, Validation: Accuracies and Lossed over EPOCHs for Twitter Data.

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	0.74	0.83	0.78	2000
Sarcastic	0.81	0.71	0.76	2000
Average	0.77	0.77	0.77	4000
Accuracy:			0.77	

These were the results obtained by using Bi-LSTM over Twitter Data



Confusion Matrix and ROC_AUC curve for Bi-LSTM over Twitter Data.

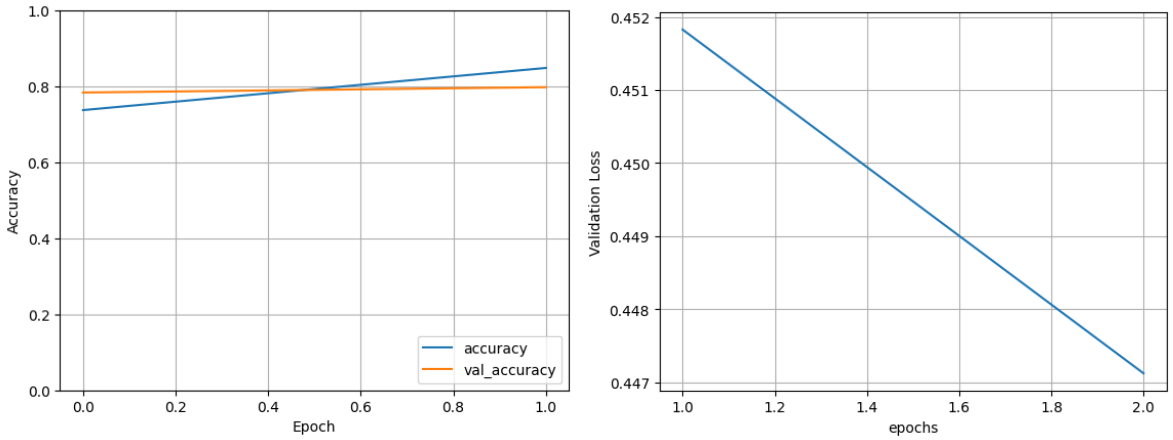
3. Bi-LSTM over Twitter+News Data:

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 41, 128)           1280000

 bidirectional_1 (Bidirecti  (None, 128)               98816
 onal)

 dense_1 (Dense)             (None, 1)                 129

=================================================================
Total params: 1378945 (5.26 MB)
Trainable params: 1378945 (5.26 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

This is the Model Architecture for Bi-LSTM over Twitter + News dataset.

Training Constants	
Train Set size	**70%**
Validation Set size	**10%**
Test Set size	**20%**
Epochs	**2**
Batch Size	**32**

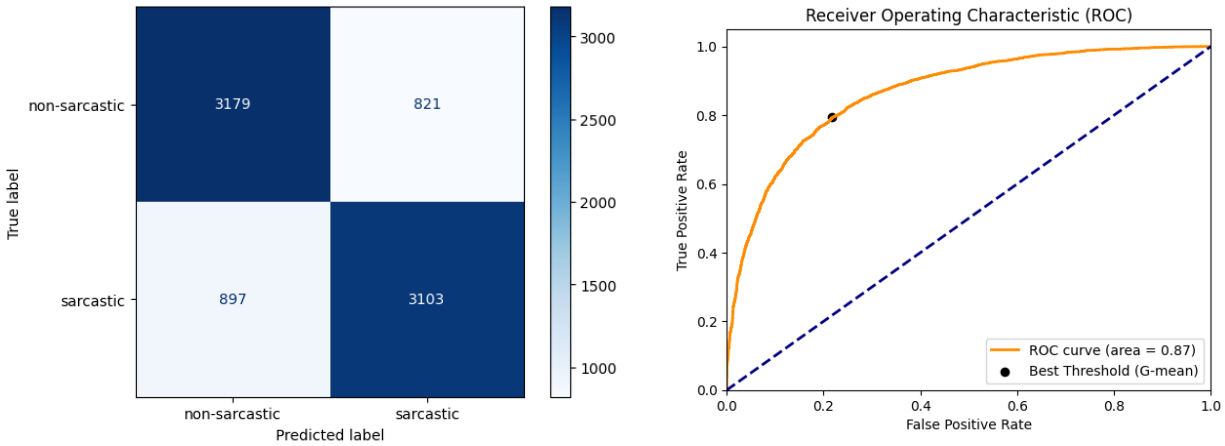
These are the model training Constraints



Training, Validation: Accuracies and Losses over EPOCHs

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	**0.78**	**0.79**	**0.79**	**4000**
Sarcastic	**0.79**	**0.78**	**0.78**	**4000**
Average	**0.79**	**0.79**	**0.79**	**8000**
Accuracy:			**0.79**	

These are the test Results obtained by using Bi-LSTM over Twitter+News Headlines.



Confusion Matrix and ROC_AUC curve for Bi-LSTM over Twitter + News Dataset.

4. Bi-LSTM for Combined Dataset:

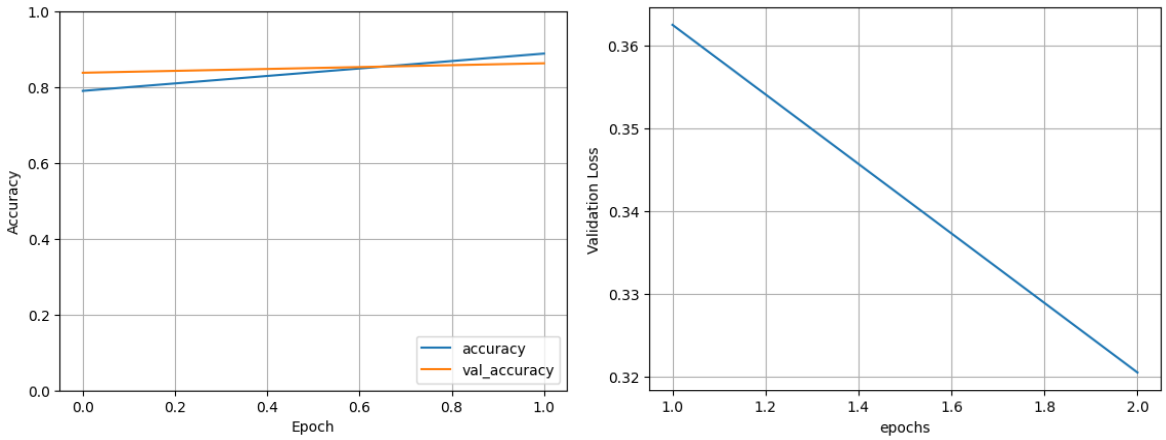
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 41, 128)           1280000

 bidirectional (Bidirection  (None, 128)               98816
 al)

 dense (Dense)               (None, 1)                 129

=================================================================
Total params: 1378945 (5.26 MB)
Trainable params: 1378945 (5.26 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

This is the model Architecture for Bi-LSTM over Combined Dataset

Training Constants	
Train Set size	**70%**
Validation Set size	**10%**
Test Set size	**20%**
Epochs	**2**

Batch Size	32

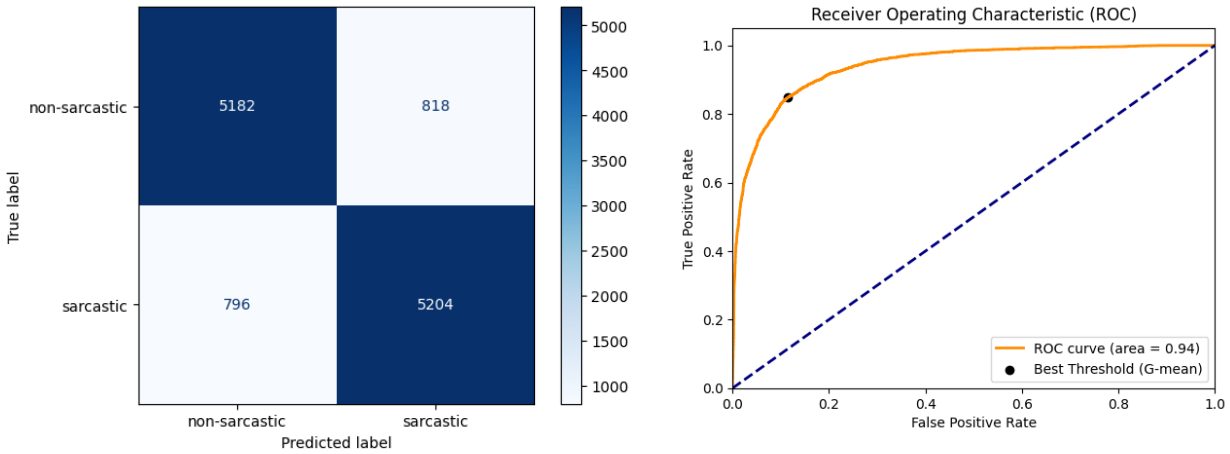
These are the training constraints that were used while fitting the model



Training, Validation: Accuracies and Lossed over EPOCHs.

Class	Precision	Recall	F1-Score	Support
Non-Sarcastic	0.87	0.86	0.87	6000
Sarcastic	0.86	0.87	0.87	6000
Average	0.87	0.87	0.87	12000
Accuracy:			0.87	

These are the test Results for BiLSTM over Combined Data



Confusion Matrix and ROC_AUC curve for BiLSTM over Comined Data.