Report

Project Name

Bangla Sentiment Analysis

Pattern Recognition Laboratory

Section B

Group Members

Himel Mazumder (011 161 045)

Kazi Faria Oishee (011 153 060)

Abstract:

Now a days Natural Language Processing (NLP) has become very popular as field of application for NLP is growing day by day. From searching on internet to reviewing a product at amazon, we are generating a huge number of data in the text form every day. Which can be analyzed to generate various predictions.

NLP is vastly used in different fields like Text Classification, Sentiment Analysis, Personality Analysis, Smart Assistance, Search Engine, Text Prediction etc. there are 3 different methods to implement NLP. These are Heuristic method, Machine Learning method, and Deep Learning method. Advantages and Disadvantages are there for all three approaches.

Sentiment Analysis is a Text Classification task that requires supervised learning models. In Sentiment Analysis, A piece of text is taken and then analyzed to predict the sentiment the text contains. Along with Heuristic approaches there are several Machine Learning and Deep Learning models for Text Classification tasks.

In this project, we've tried to build a Sentiment Analyzer for Bangla language. We have used both Machine Learning and Deep Learning methods for our analyzer.

In our project, we've implemented Naïve Bayes and Logistic Regressing for Machine Learning tuned with K-Fold Cross validation and Grid Search technique. For Deep Learning we've used LSTM model. At the end of the project, we've compared the performances in terms of Accuracy scores of these models.

Introduction:

For text classification tasks, Naïve Bayes is better performing models among all other machine learning models. It is widely used model in this domain of NLP. Along with Naïve Bayes, some other machine learning models such as Logistic Regression, Support Vector Machine (SVM) are also used. Logistic Regression is quite popular. For our project, we've implemented Naïve Bayes and Logistic Regression. To make our models more exposed to various data, both models have been trained with K-Fold Cross Validation techniques. Only Logistic Regression model has been hyper tuned with Grid Search technique. Grid Search can't be applied to Naïve Bayes as this model doesn't take any parameter.

Among Deep Learning model, RNN has been used traditionally. But due to Gradient Vanishing problem RNN suffers Short Term Memory. In order to solve this, another model Long Short-Term Memory (LSTM) is used to eliminate this problem. And to this date, LSTM is the best performing Deep Learning model for Text Classification.

We've implemented LSTM model for our Bangla Sentiment Analysis project.

Methodologies:

Bangla Sentiment Analysis project building involves 5 steps namely as Data Acquisition, Data Preprocessing, Feature Engineering, Model Implementation, and Result Analysis.

These steps are explained in details below.

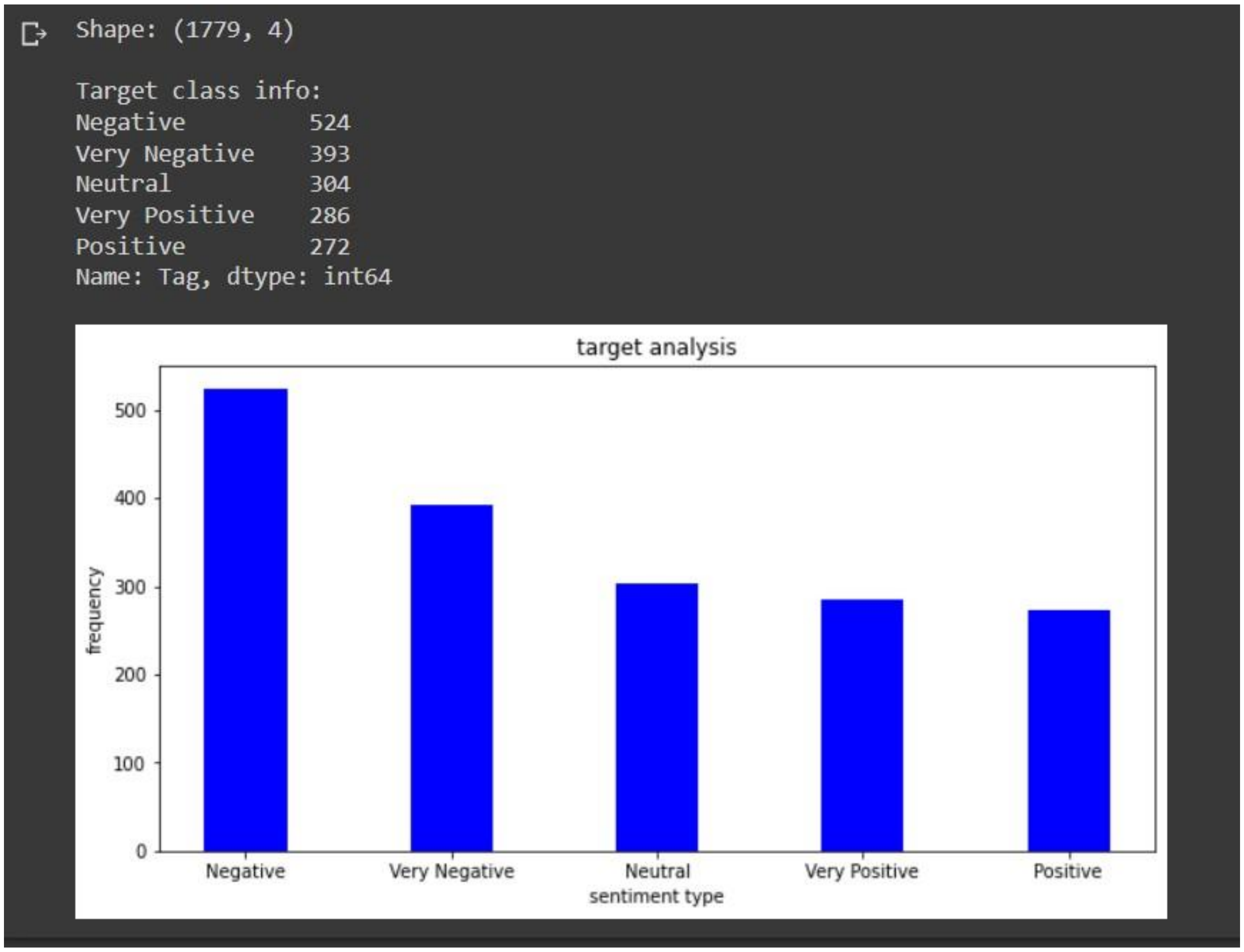
Data Acquisition:

The Dataset used in our Bangla Sentiment Analysis project has been collected from Kaggle. The URL for the dataset is

https://www.kaggle.com/datasets/mobassir/bengali-news-comments-sentiment?resource=download.

The dataset has 1779 instances and 2 columns. First column contains comments which is the feature for our project and the other one contains sentiments for the comments. Second column is the target column for our project. There are 5 unique values for our target column. These are Negative, Very Negative, Neutral, Positive, Very Positive.

The dataset is in Bangla only. But we have changed a few instances to check some of our text preprocessing functions performing tasks such as converting emojis to text, English special character and number removal, and hyperlink removal.



```
Shape: (1779, 4)

Target class info:
Negative         524
Very Negative    393
Neutral          304
Very Positive    286
Positive         272
Name: Tag, dtype: int64
```

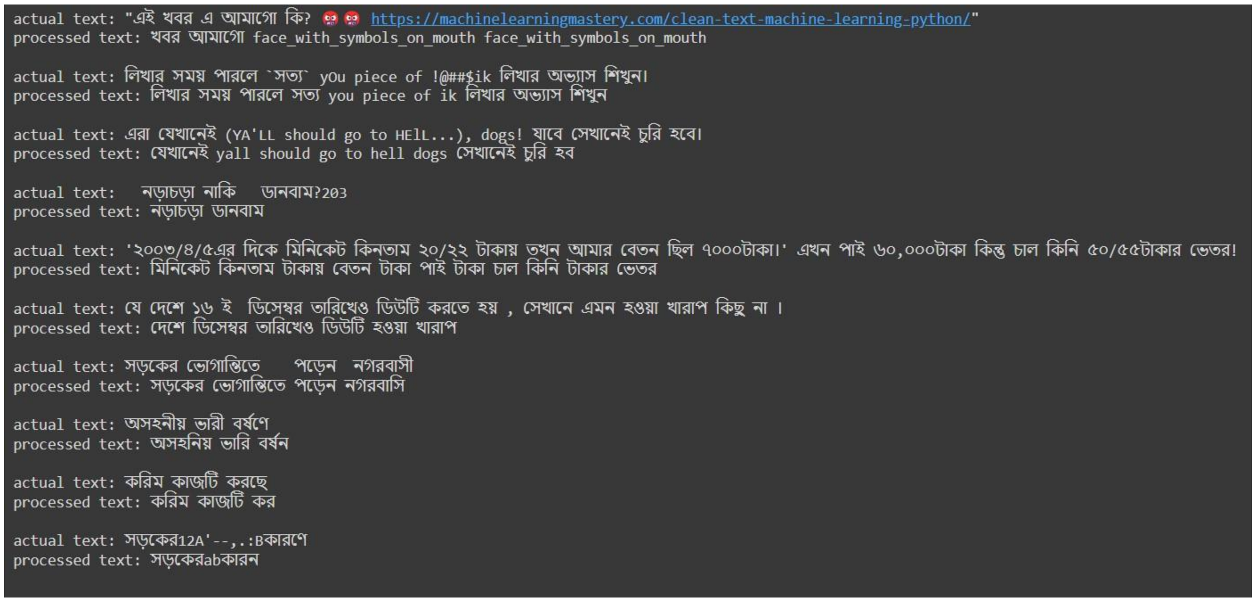
Data Preprocessing:

In the data preprocessing steps we've completed several tasks to process the raw data. These are listed below:

1. **Converting emojis to text:**
   We didn't remove emojis considering that these will contain sentimental value of the text.
2. **Removing URL:**
   We've considered that an URL holds no significance for sentiment analysis of a text. Hence, we've removed them.
3. **Removing English special characters except '_':**

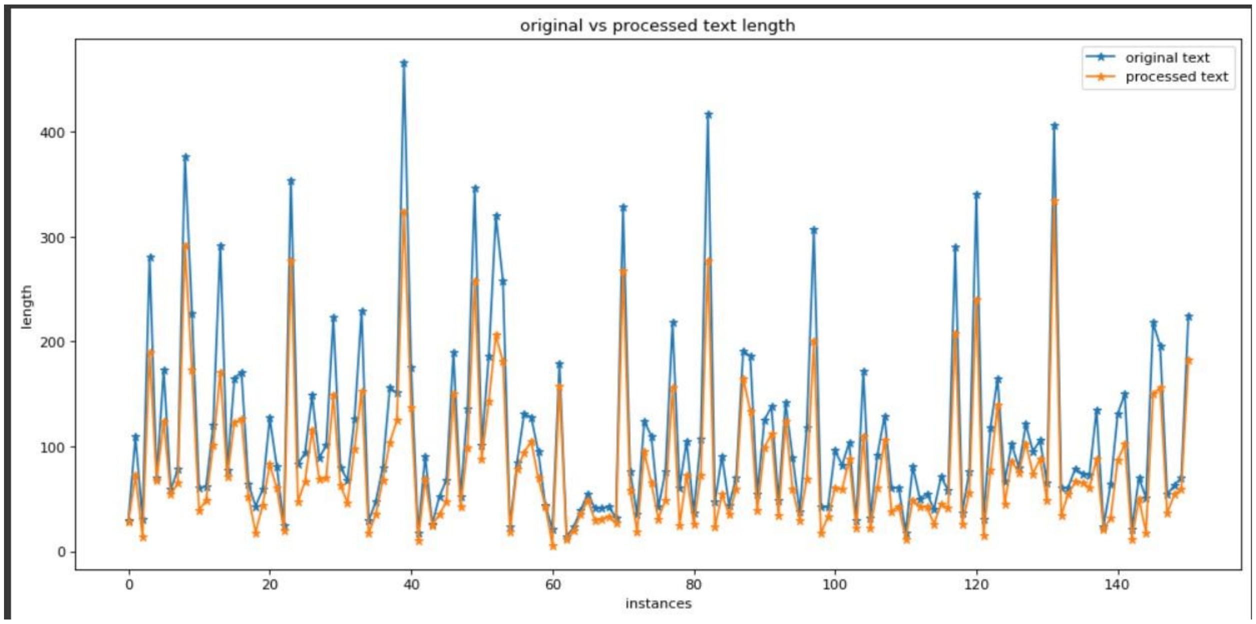
'_' has been used for converting emojis to text. Because names we've used to replace emojis have '_' in them instead of SPACE.

4. **<u>Removing Bangla special characters and numbers:</u>**
5. **<u>Removing stop words for both Bangla and English:</u>** Stop words has no sentimental significance hence removed.
6. **<u>Stemming for Bangla words:</u>** Stemming converts a word to it's root form.
   Input: ['কবিরগুলিকে', 'আমাকে', 'নামাবার']
   Output: ['কবির', 'আমা', 'নামা']
7. **<u>Normalization for Bangla words:</u>** Normalization is a technique that uses same vowel for all the same instances of Bangla words. It stops the same word being considered as different due to vowel formation.
   Input: নগরবাসী
   Output: নগরবাসি
8. **<u>Data cleaning:</u>**
   For data cleaning, we've tried to detect 2 things.
   1. **Duplicated value:** Duplicated instances will be removed keeping only the first instance.
   2. **Instances with missing value:** if there's a missing value in any column for an instance, that instance will be dropped.



```
actual text: "এই খবর এ আমাগো কি? 🤬🤬 https://machinelearningmastery.com/clean-text-machine-learning-python/"
processed text: খবর আমাগো face_with_symbols_on_mouth face_with_symbols_on_mouth

actual text: লিখার সময় পারলে `সত্য` yOu piece of !@##$ik লিখার অভ্যাস শিখুন।
processed text: লিখার সময় পারলে সত্য you piece of ik লিখার অভ্যাস শিখুন

actual text: এরা যেখানেই (YA'LL should go to HElL...), dogs! যাবে সেখানেই চুরি হবে।
processed text: যেখানেই yall should go to hell dogs সেখানেই চুরি হব

actual text:   নড়াচড়া নাকি   ডানবাম?203
processed text: নড়াচড়া ডানবাম

actual text: '২০০৩/৪/৫এর দিকে মিনিকেট কিনতাম ২০/২২ টাকায় তখন আমার বেতন ছিল ৭০০০টাকা।' এখন পাই ৬০,০০০টাকা কিন্তু চাল কিনি ৫০/৫৫টাকার ভেতর!
processed text: মিনিকেট কিনতাম টাকায় বেতন টাকা পাই টাকা চাল কিনি টাকার ভেতর

actual text: যে দেশে ১৬ ই  ডিসেম্বর তারিখেও ডিউটি করতে হয় , সেখানে এমন হওয়া খারাপ কিছু না ।
processed text: দেশে ডিসেম্বর তারিখেও ডিউটি হওয়া খারাপ

actual text: সড়কের ভোগান্তিতে    পড়েন  নগরবাসী
processed text: সড়কের ভোগান্তিতে পড়েন নগরবাসি

actual text: অসহনীয় ভারী বর্ষণে
processed text: অসহনিয় ভারি বর্ষন

actual text: করিম কাজটি করছে
processed text: করিম কাজটি কর

actual text: সড়কের12A'--,.:Bকারণে
processed text: সড়কেরabকারন
```

Length comparison between raw and processed comments in the dataset:

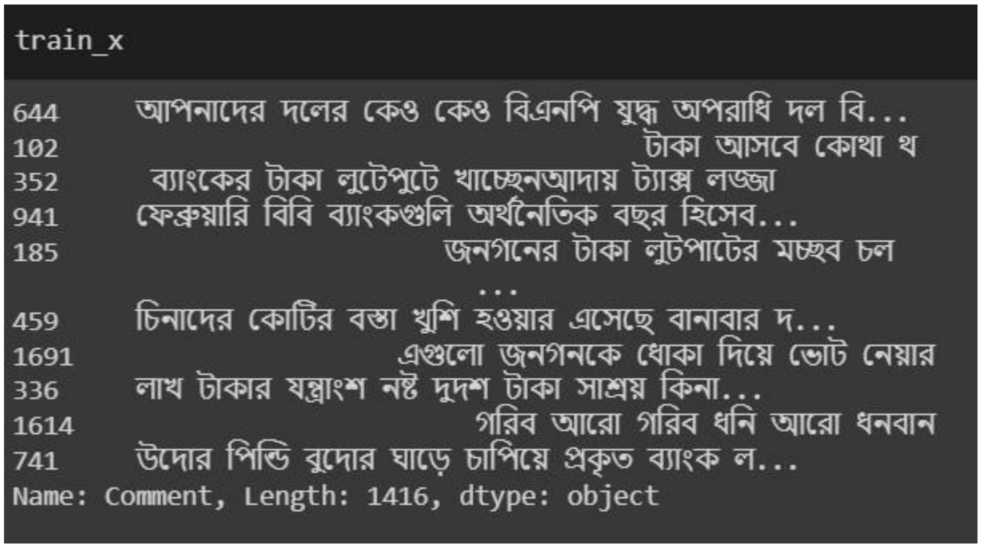


Feature Engineering:

In NLP feature engineering means converting words in to number vectors. In our feature engineering part, we've used TF-IDF vectorization technique. TF-IDF is a better method as it not only stores the frequency of a word but also the significance of that specific word is also taken into consideration. This way we can remove words with least significance. So, size of our dataset doesn't get unnecessarily large.

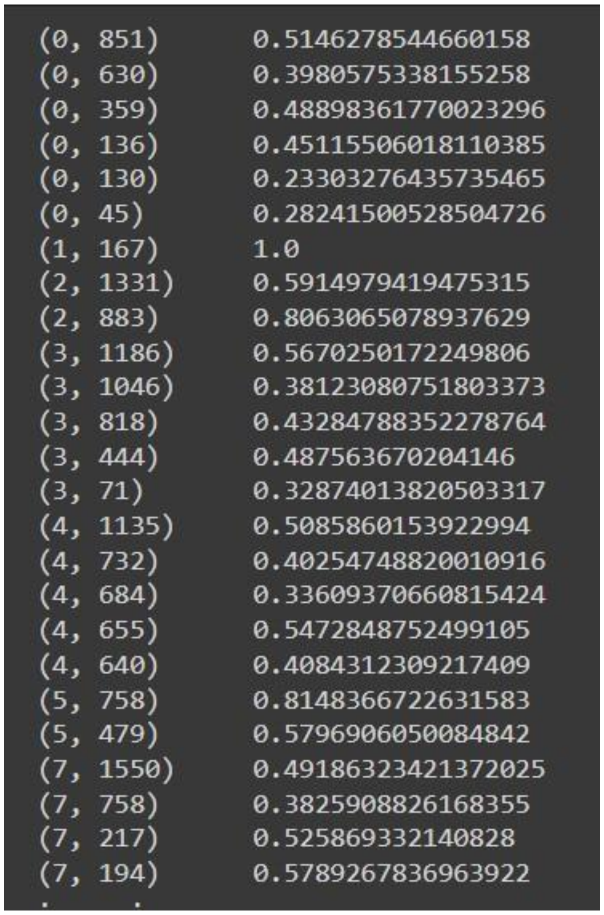
Feature engineering for deep learning is done by the model automatically.

Training feature before vectorization:



```
train_x
644      আপনাদের দলের কেও কেও বিএনপি যুদ্ধ অপরাধি দল বি...
102                                   টাকা আসবে কোথা থ
352       ব্যাংকের টাকা লুটেপুটে খাচ্ছেনআদায় ট্যাক্স লজ্জা
941      ফেব্রুয়ারি বিবি ব্যাংকগুলি অর্থনৈতিক বছর হিসেব...
185                           জনগনের টাকা লুটপাটের মচ্ছব চল
                                   ...
459      চিনাদের কোটির বস্তা খুশি হওয়ার এসেছে বানাবার দ...
1691                     এগুলো জনগনকে ধোকা দিয়ে ভোট নেয়ার
336      লাখ টাকার যন্ত্রাংশ নষ্ট দুদশ টাকা সাশ্রয় কিনা...
1614                         গরিব আরো গরিব ধনি আরো ধনবান
741      উদোর পিন্ডি বুদোর ঘাড়ে চাপিয়ে প্রকৃত ব্যাংক ল...
Name: Comment, Length: 1416, dtype: object
```

Training feature after vectorization:



```
  (0, 851)      0.5146278544660158
  (0, 630)      0.3980575338155258
  (0, 359)      0.48898361770023296
  (0, 136)      0.45115506018110385
  (0, 130)      0.23303276435735465
  (0, 45)       0.28241500528504726
  (1, 167)      1.0
  (2, 1331)     0.5914979419475315
  (2, 883)      0.8063065078937629
  (3, 1186)     0.5670250172249806
  (3, 1046)     0.38123080751803373
  (3, 818)      0.43284788352278764
  (3, 444)      0.487563670204146
  (3, 71)       0.32874013820503317
  (4, 1135)     0.5085860153922994
  (4, 732)      0.40254748820010916
  (4, 684)      0.33609370660815424
  (4, 655)      0.5472848752499105
  (4, 640)      0.4084312309217409
  (5, 758)      0.8148366722631583
  (5, 479)      0.5796906050084842
  (7, 1550)     0.49186323421372025
  (7, 758)      0.3825908826168355
  (7, 217)      0.525869332140828
  (7, 194)      0.5789267836963922
  :     :
```

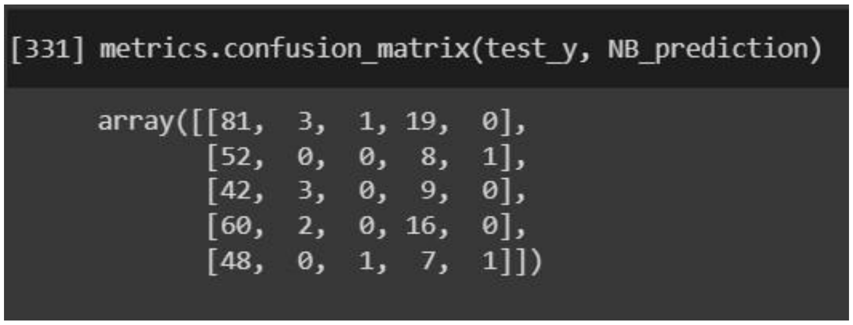
Model Implementation:

Models we've implemented are:

Machine Learning:

Naïve Bayes: It's a normal Naïve Bayes model. That splits data in to train and test. Then model is trained with train data and its performance is measured using test data. The model is never trained with testing data.

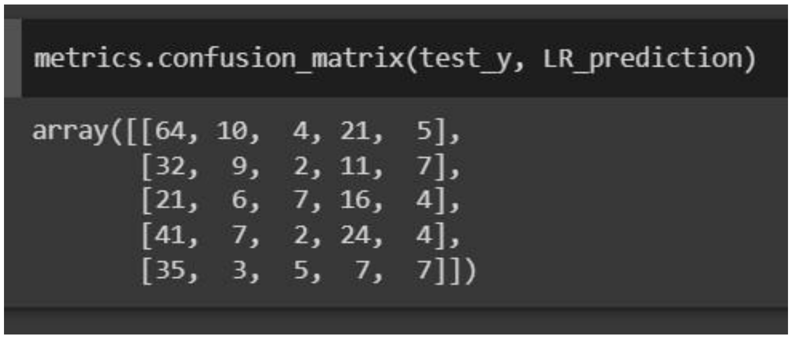
Confusion matrix



```
[331] metrics.confusion_matrix(test_y, NB_prediction)

     array([[81,  3,  1, 19,  0],
            [52,  0,  0,  8,  1],
            [42,  3,  0,  9,  0],
            [60,  2,  0, 16,  0],
            [48,  0,  1,  7,  1]])
```

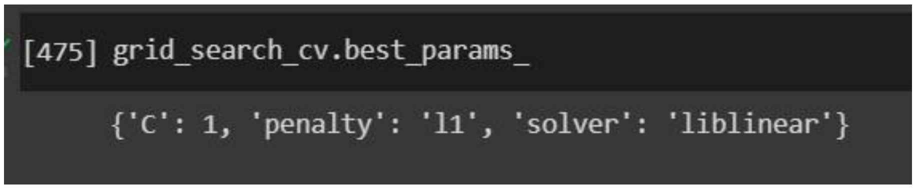
Naïve Bayes with K-Fold Cross Validation: The model is trained and tested with all the data instances eventually. We've tested for different values of k. these are 3, 5 and 10. Among them model with k = 3 has the higher accuracy. We've used stratified K-Fold.

Logistic regression: Normal Logistic Regression model.



```
metrics.confusion_matrix(test_y, LR_prediction)

array([[64, 10,  4, 21,  5],
       [32,  9,  2, 11,  7],
       [21,  6,  7, 16,  4],
       [41,  7,  2, 24,  4],
       [35,  3,  5,  7,  7]])
```

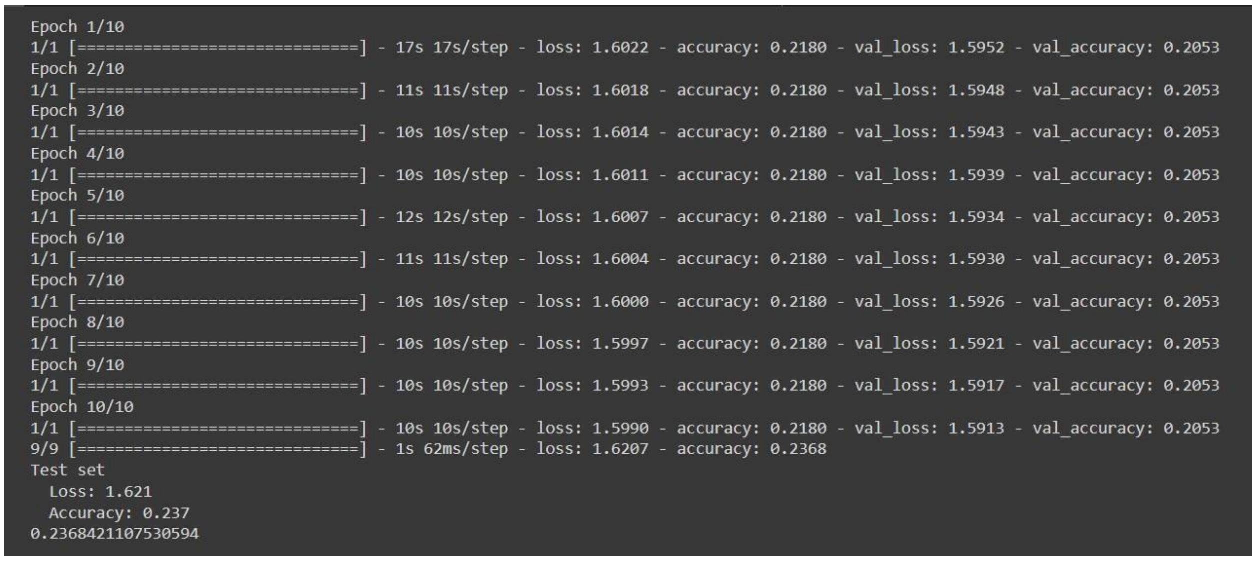
Logistic regression with Grid Search: Grid Search is used for hyper tuning. If a model takes different parameters, then using Grid Search we can find out which combination of parameter values gives the model the best accuracy. Grid search uses stratified K-Fold cross validation automatically. In grid search we used 3 for our k value. For our project, the best combination of parameters is



```
[475] grid_search_cv.best_params_

      {'C': 1, 'penalty': 'l1', 'solver': 'liblinear'}
```

Deep Learning:

LSTM: We have used LSTM deep learning model for our project. The model run using Batch = 1500 and epoch = 10 values. Maximum number of words in the dataset is set to 5000 and maximum number of words in each instance is set to 300.



```
Epoch 1/10
1/1 [==============================] - 17s 17s/step - loss: 1.6022 - accuracy: 0.2180 - val_loss: 1.5952 - val_accuracy: 0.2053
Epoch 2/10
1/1 [==============================] - 11s 11s/step - loss: 1.6018 - accuracy: 0.2180 - val_loss: 1.5948 - val_accuracy: 0.2053
Epoch 3/10
1/1 [==============================] - 10s 10s/step - loss: 1.6014 - accuracy: 0.2180 - val_loss: 1.5943 - val_accuracy: 0.2053
Epoch 4/10
1/1 [==============================] - 10s 10s/step - loss: 1.6011 - accuracy: 0.2180 - val_loss: 1.5939 - val_accuracy: 0.2053
Epoch 5/10
1/1 [==============================] - 12s 12s/step - loss: 1.6007 - accuracy: 0.2180 - val_loss: 1.5934 - val_accuracy: 0.2053
Epoch 6/10
1/1 [==============================] - 11s 11s/step - loss: 1.6004 - accuracy: 0.2180 - val_loss: 1.5930 - val_accuracy: 0.2053
Epoch 7/10
1/1 [==============================] - 10s 10s/step - loss: 1.6000 - accuracy: 0.2180 - val_loss: 1.5926 - val_accuracy: 0.2053
Epoch 8/10
1/1 [==============================] - 10s 10s/step - loss: 1.5997 - accuracy: 0.2180 - val_loss: 1.5921 - val_accuracy: 0.2053
Epoch 9/10
1/1 [==============================] - 10s 10s/step - loss: 1.5993 - accuracy: 0.2180 - val_loss: 1.5917 - val_accuracy: 0.2053
Epoch 10/10
1/1 [==============================] - 10s 10s/step - loss: 1.5990 - accuracy: 0.2180 - val_loss: 1.5913 - val_accuracy: 0.2053
9/9 [==============================] - 1s 62ms/step - loss: 1.6207 - accuracy: 0.2368
Test set
  Loss: 1.621
  Accuracy: 0.237
0.2368421107530594
```

Result Analysis:

Naïve Bayes (NB): 0.276

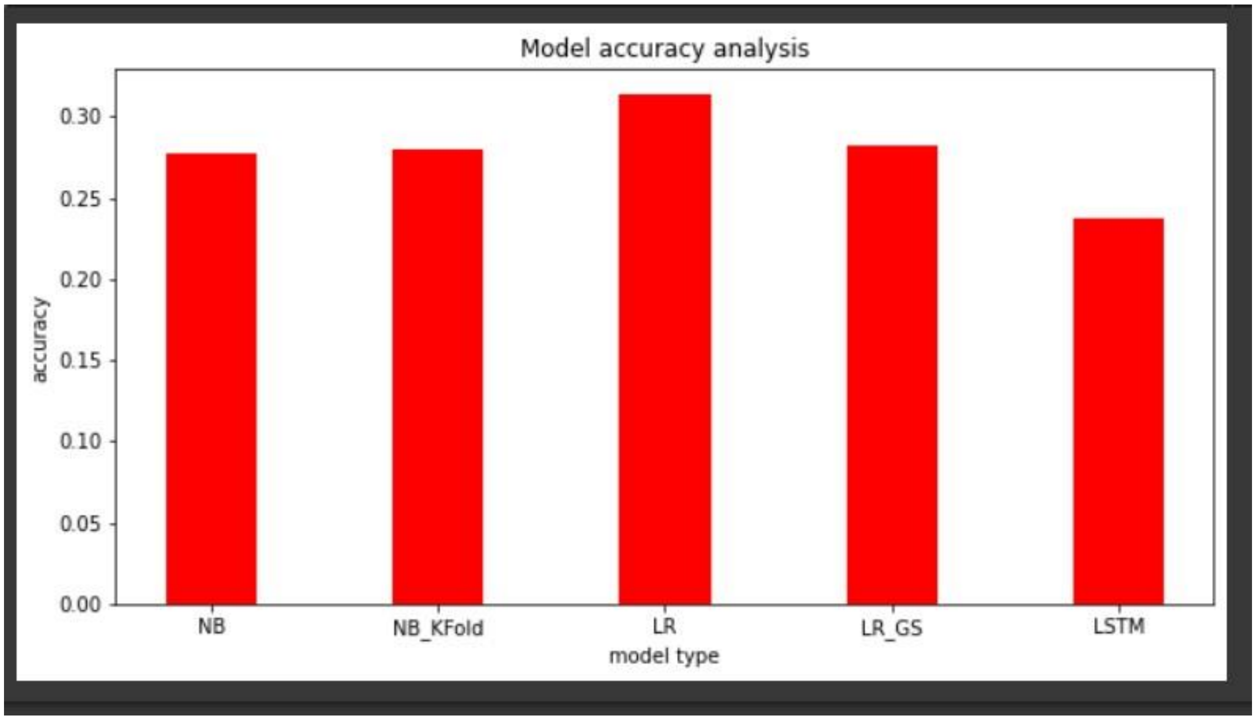
Naïve Bayes with K-Fold (NB_KFold): 0.279

Logistic Regression (LR): 0.313

Logistic Regression with Grid Search (LR_GR): 0.281

LSTM: 0.237

Model Accuracy Comparison



From the result analysis we can see that Logistic Regression model gives the best accuracy score on the given dataset for our Sentiment Analysis project.

Conclusion:

It is clearly visible that accuracy is very low. One of the prime reason for this can be dataset with low amount of instances. If these models were trained with much bigger dataset accuracy might improve.