AUTOMATIC STANCE DETECTION

A PROJECT REPORT

Submitted in the partial fulfilment of the requirements
For the award of the degree of
BACHELOR OF TECHNOLOGY
IN
COMPUTER ENGINEERING

Under The Supervision of
Mr. Mohammad Zeeshan Ansari
Assistant Professor

Submitted by
Shaz Akhtar (15BCS0039)
Abhinav Kumar Jha(15BCS0002)

DEPARTMENT OF COMPUTER ENGINEERING
FACULTY OF ENGINEERING AND TECHNOLOGY
JAMIA MILLIA ISLAMIA , NEW DELHI-110025

CERTIFICATE

This is to certify that the dissertation / project report (Course Code) entitled "Automatic Stance Detection" , is an authentic work carried out by
Mr. Shaz Akhtar and Mr. Abhinav Kumar Jha.

The work is submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Engineering under my guidance.

Date : 22/12/2018

Mr. Mohammad Zeeshan Ansari
Assistant Professor
Dept. of Computer Engg.
Jamia Millia Islamia

ACKNOWLEDGEMENT

We would like to thank our mentor Mr. Mohammad Zeeshan Ansari (Assistant Professor Dept. of Computer Engg.) for giving us the opportunity to undertake the project. We thank him for his immense guidance, and appreciate his timely engagement.

We would like to expand special gratitude to the assistants and lab coordinators of the department for providing us the infrastructural facilities necessary to sustain the project.

CONTENTS

Introduction

In context of news, a claim is made in a news headline, as well as in the piece of text in an article body.
Quite often, the headline of a news article is created so that it is attractive to the readers, even though the body of the article may be about a different subject/may have another claim than the headline.

Stance Detection involves estimating the relative perspective (or stance), of two pieces of text relative, i.e. do the two pieces agree, disagree, discuss or are unrelated to one another. Your task

- Stance detection has various potential uses, from search engines (matching queries to documents) to recommending similar articles.

- For this project, we will focus on the specific problem of ‘matching’ an article’s headline with its text. Quite often, news articles may have an ‘attractive’ headline that do not necessarily match the content of its body, which may be about a different subject or have a different claim than the headline.

- The goal of this project is, given an article’s headline and body text, predict whether that body’s text agrees with the headline, disagrees with the headline, discusses the same topics/claim as the headline, or is unrelated to the headline.

TASK DEFINITION & CLASSIFICATION

- The goal in stance detection is to detect whether the headline and the body of an article have the same claim.
- The stance can be categorized as one of the four labels: “agree”, “disagree”, “discuss” and “unrelated”.

Formal definitions of the four stances are as:

1. “agree” – the body text agrees with the headline
2. “disagree” – the body text disagrees with the headline
3. “discuss” – the body text discusses the same claim as the .. . headline but does not take a position
4. “unrelated" – the body text discusses a different claim but not that in the headline.

DATASET OVERVIEW

- The dataset for this project comes from a publicly-available challenge, Fake News Challenge or FNC-1.

- This dataset is divided into a training set and a testing set. The ratio of training data over testing data is about 2:1.

- This dataset provides a total of 75385 articles (headline-body pairs) (split into 49972 articles in the training dataset and 25413 articles in the test dataset).

- Each article (or headline-body pair) is labelled with either

1. Unrelated ,
2. Discuss ,
3. Agree , or
4. Disagree.

The data set is downloaded from
“https://github.com/FakeNewsChallenge/fnc-1”

TOOLS USED:-

Programming Language used:**Python**
Programming Environment :Anaconda Jupyter Notebook with Python 3.7 version
Glove Word Embedding :
Please download it from “https://nlp.stanford.edu/projects/glove/”

Associated Libraries:->
Pandas,SciPy,Sklearn,Numpy,NLTK,TensorFlow,Keras

Pandas:for creating the dataframe for text

Trained Glove word embeddings : for mapping the tokenized words with the corresponding numbers in the embedding.

Scipy:For implementing various data science functionalities

NLTK :For tokenizing the text and extracting the features from the text
TensorFlow:For training the Glove word embeddings

Problem Approach

1. First after loading the data in the form of csv format,we will perform **Data Cleaning** practises and remove all the stop words,do stemming,and all the unwanted symbols from the text data.

2. Then we tokenize the text which represents the text into set of words or tokens and represent the whole sentence as a whole.

3. We preprocess the data so as to be used by different word embedding techniques

4. We perform the feature engineering on our preprocessed data and extract three features from it namely

    1.Word vectorization to find the similarity between the two set of text.

    2.KL Divergence method of finding the divergence between the words.

    3.N gram overlap method of finding the number of occurence of the words in the sentence so as to determine the weight of the word in the text.

5.We pass the vectored method through the pretrained word embedding vector called Glove Word embeddings which has been trained on the massive data set from the wikipedia and get the corresponding numerical weight for a particular word.

Modelling techniques

1.We then split this vectored x_label into split and train data set and train the following algorithms for calculating the accuracy:

- Linear regression:
- Logistic regression
- Linear Regression
- Logistic Regression
- K Nearest Neighbour
- Support Vector Machine
- Qudratic Discriminant Analysis
- Random Forest
- Adaboost
- SGD Classifier
- Decision Tree
- XG Boost
- Linear Discriminant Analysis
- Gaussian Naive Bayes

DATASET SPLIT (TRAIN - VALIDATION SPLIT)

- The first step required was to split the training dataset into a train subset (used to train models for feature engineering (e.g. idf) and machine learning) and a validation subset (used to optimise hyper-parameters for these models independently of the test set).

- The training dataset was split 9:1 (90% training subset, 10% validation subset).
- splitting the training dataset to four arrays for each label, splitting each array 9:1 (to train_unrelated, val_unrelated, train_discuss, val_discuss, etc.), combining all the train subsets into one array and validation subsets into one array, then shuffling them.

- After this split, the number of articles, for each label, on the training and validation subsets are:

	Unrelated	Discuss	Agree	Disagree	All
All	36545	8909	3678	840	49972
Train	32890	8018	3310	756	44974
Validation	3655	891	368	84	4998

DATA PRE-PROCESSING

- The next step (before calculating features) is to pre-process the raw text files. This is done via the tokenize(text) function.

- This function converts raw text into a list of words, after stripping the text of all non-alphanumeric characters.
- Additionally, common, meaningless words (‘stop words’) in English (such as ‘the’ or ‘an’) are removed from the returned list of words to reduce noise in the features (e.g. remove meaningless words from term-frequency vectors).

- This is shown to improve the accuracy of the learning model, as if we do not filter out these stop words, the accuracy of our model on the validation set decreases from 0.870 to 0.818.

Feature Engineering

We are extracting three features from the text data set through three methods

1.Word vectorisation

2.KL Divergence Method

3.N gram overlap

and created a n*3 vector by mapping the data values obtained to pretrained Glove word embeddings.

Word Vectorization

I. Computing **TF-IDF**

- Initially, each document (headlines and article bodies) are vectorised into tf-idf form.

- Before this is possible, we need to learn the idf weights of every word in the collection. This is done by going through all documents and building a dictionary of

  (word -> in how many documents does that word appears). This is the document frequency (df) score of each word.
- To get the idf score of each word, its df score is inverted and smoothed with the following function :

$$idf_{word} = \ln\left(\frac{1 + N\ (no.of\ docs\ in\ collection)}{1 +\ df_{word}\ (no.of\ docs\ in\ which\ word\ occurs)}\right) + 1$$

- Then, we can compute the tf-idf values of each word in a document.

- First, the term frequency of the word is counted (e.g. if the word ‘glove’ appears twice in a document d, the tf of glove is 2), then this tf value is multiplied by the idf weight of said word (e.g. if the idf of ‘glove’ is 1.25, then its tf-idf value in d is 2*1.25=2.5).

- This tf-idf representation of each document is saved as a (word->tf-idf) value dictionary (e.g. ‘glove’->2.5).

$$tfidf_{word,doc} = tf_{word,doc}\ (no.of\ times\ word\ occurs\ in\ doc) \times idf_{word}$$

II. GloVe: Glove Vectors for Word Representation

- Then, to make these tf-idf representations comparable, we used GLoVe pre-trained word vectors (https://nlp.stanford.edu/projects/glove/) to convert these tf-idf representations to fixed-length vectors based on the learned 'meaning' of each word.
- GLoVe provides a mapping of six billion English words to a 50-dimensional vector, trained on Wikipedia and Gigaword.

converting a document to a GLoVe vector -:

- To convert a document to a GLoVe vector, the (scalar) tf-idf value of each word in the document is multiplied by the GLoVe vector associated with the word, which is summed together and normalised for document length:

$$glove_{doc} = \frac{\sum\left(glove_{word} \times tfidf_{word,doc}\right)\ (for\ each\ word\ in\ the\ document)}{\sum\left(tfidf_{word,doc}\right)\ (for\ each\ word\ in\ the\ document)}$$

(Note that $glove_{word}$ is a vector, while $tfidf_{word,doc}$ is a scalar value)

III. Computing Cosine Similarity of GloVe vectors for all Heading-Body pairs -:

- The GLoVe vector representation of documents is used as a feature by calculating the cosine similarity of the GLoVe vector representation of each article’s headline and body.

- The cosine similarity of two similar-length vectors are computed with:

$$cosine\_similarity = \frac{H \cdot B}{|H||B|} = \frac{\sum_{i=0}^{len(glove)}(H_i \times B_i)}{\sqrt{\sum_{i=0}^{len(glove)}(H_i^2)} \times \sqrt{\sum_{i=0}^{len(glove)}(B_i^2)}}$$

Where H = GLoVe vector representation of headline,

B = GLoVe vector representation of body, and
len(glove) is the dimensionality of the GloVe
vector representation used (in this case 50).

- This feature returns a real value between -1.0 and 1.0
(higher = words in headline and body are more similar, lower = words in headline and body are more different).

LANGUAGE MODEL REPRESENTATION (KL-DIVERGENCE)

- Another feature that we are using is the KL-Divergence of the language model representations of the headline and the article body.

- This is a measure of how divergent (i.e. different) are the language (i.e. words) being used in the headline and the body.

- To convert a document to a (simple, unigram) language model, we compute the probability of each word occurring in the document, by using the occurrence of the word:

$$LM_{doc}(word) = \frac{tf_{word,doc}\ (no.of\ times\ word\ occurs\ in\ doc) +\ eps}{|doc|\ (no.of\ words\ in\ doc) + eps \times |V|\ (no.of\ unique\ words\ in\ headline\ AND\ body)}$$

Computing the KL-Divergence of language model (LM) representations of the headline and the body

- KL-divergence is a measure of how different two probability distributions are. In this case, KL-divergence is used to measure the divergence between the language model of each article’s headline and body.

- This formula is defined as:

$$kl\_divergence\ = \sum\left(LM_{headline}(word) \times \ln\left(\frac{LM_{headline}(word)}{LM_{body}(word)}\right)\right)\ (for\ each\ word\ in\ V)$$

- This feature returns a positive real value (in practice between 0.0 and

~3.0, but in theory uncapped), where the higher the value is, the more divergent the language model (i.e. words used) in the article’s headline and body are.

N-gram overlap

- N-gram overlap is a measure of how many times n-grams that occur on the article’s headline re-occur on the article’s body.

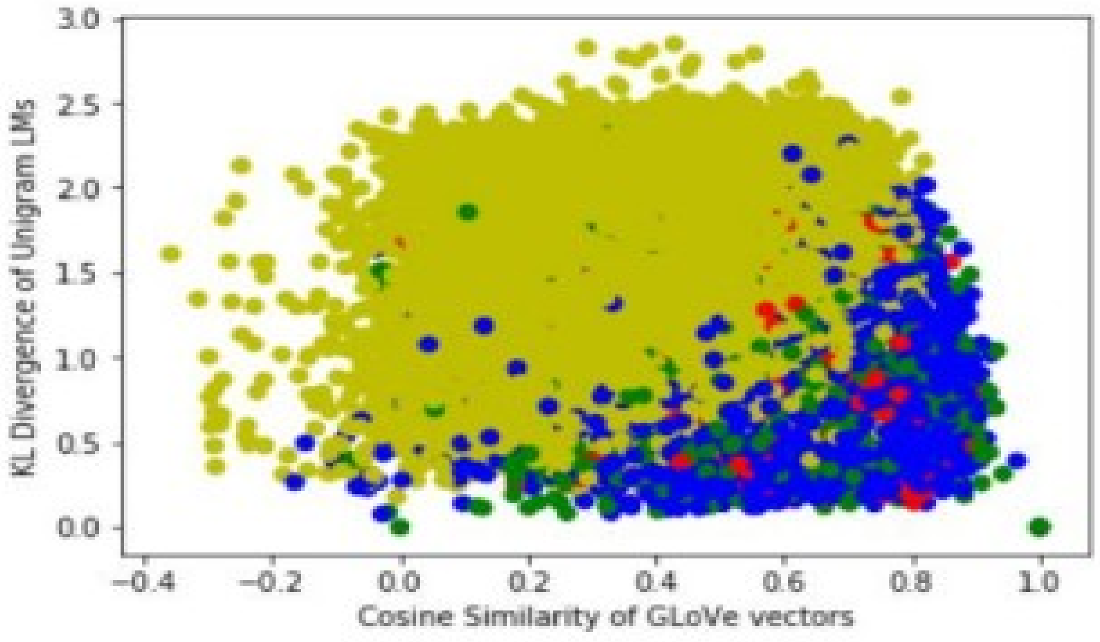
- For each article (headline-body pair), I counted n-gram overlaps up to 3-grams (i.e. count no. of words in headline that re-occur on body + no. of sequence of 2-words in the headline that re-occur in body + no. of sequence of 3-words in the headline that re-occur in body).

$$3gram\_overlap = \left(\frac{no.of\ 1/2/3grams\ in\ headline\ that\ reoccur\ in\ body}{no.of\ words\ in\ body\ (article\ length)}\right)^{\frac{1}{e}}$$

- N-gram overlap returns a real value between 0.0 and 1.0 (higher = words in headline and body are more similar and, lower = words are more different).

DISTANCE DISTRIBUTION PLOT

- To show the distance distribution of the four stances
- we plotted a scatterplot of each article's KL-divergence and cosine similarity scores, alongside its label, using matplotlib:



Where yellow = unrelated, blue = discuss, green = agree, and red = disagree.

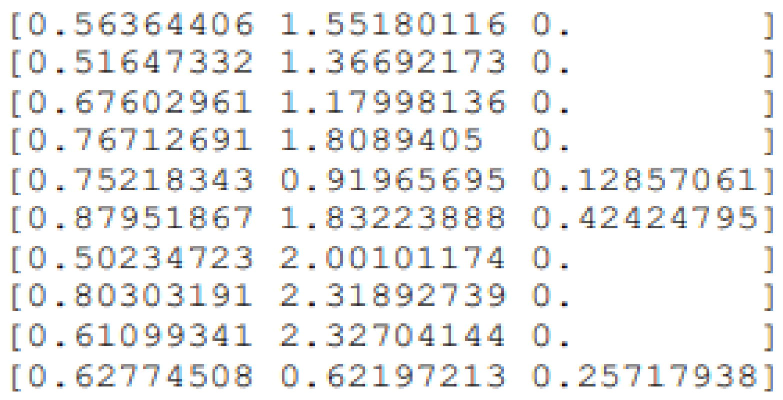
- From this plot, a trend where unrelated articles tend to have higher KL-divergence scores and lower cosine similarity scores between its headline and body can be seen. This would be useful for the task of detecting unrelated articles – the regression model will learn to give higher scores for 'unrelated' where KL-divergence is high and cosine similarity is low, and higher scores for other labels where KL-divergence is low and cosine similarity is high

MACHINE LEARNING   MODELS USED -:

I.Linear Regression -:

- By this point, we have converted each article (headline-body pair) to a feature vector over these features (Cosine similarity of word vectors, KL-divergence of language models, 3-gram overlap).

- This is represented in a matrix X, where e.g. the first ten rows of X (representing the first ten articles or headline-body pairs) are:



```
[0.56364406 1.55180116 0.        ]
[0.51647332 1.36692173 0.        ]
[0.67602961 1.17998136 0.        ]
[0.76712691 1.8089405  0.        ]
[0.75218343 0.91965695 0.12857061]
[0.87951867 1.83223888 0.42424795]
[0.50234723 2.00101174 0.        ]
[0.80303191 2.31892739 0.        ]
[0.61099341 2.32704144 0.        ]
[0.62774508 0.62197213 0.25717938]
```

Where column 1 correspond to cosine similarity of word vectors, column 2 correspond to KL-divergence of language models, and column 3 correspond to 3-gram overlap scores. Each row corresponds to a label in the output vector Y (e.g. the first ten labels in Y):

[3 3 3 3 3 2 3 3 3 2]

Where the expected output for [0.56364406, 1.55180116, 0.], is 3, the expected output for [0.62774508, 0.62197213, 0.25717938] is 2, etc.

(0 = agree, 1 = disagree, 2 = discuss, 3 = unrelated)

A one-vs-all classifier using linear regression models-:

- A one-vs-all classifier using linear regression models was used to train a model to predict the output values of each row in X.

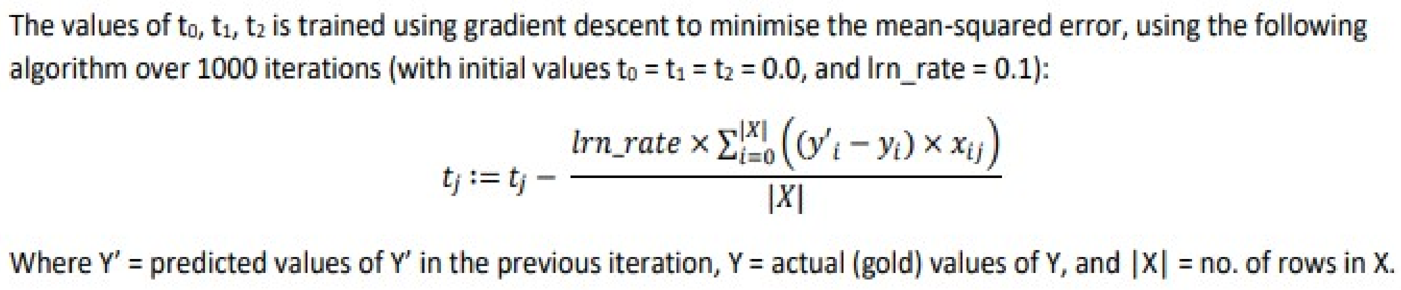
- The One vs All classifier will train four linear regression models (one for each label) by transforming the domain of Y to {0, 1} (0 if yi != the label, 1 if yi = the label) to train each linear regression model.

- Each linear regression model would then predict a real value y’i for each feature vector xi in X, in which y’i corresponds to the likelihood of the feature vector xi having the label associated with that model.

- The linear regression model predicts scores y’i for each feature vector xi, based on the following model:

$$y'\, i = t0\; xi0 + t1\; xi1 + t2\; xi2$$

Learning Rate of Linear Regression-:



The values of t_0, t_1, t_2 is trained using gradient descent to minimise the mean-squared error, using the following algorithm over 1000 iterations (with initial values $t_0 = t_1 = t_2 = 0.0$, and lrn_rate = 0.1):

$$t_j := t_j - \frac{lrn\_rate \times \sum_{i=0}^{|X|}\left((y'_i - y_i) \times x_{ij}\right)}{|X|}$$

Where Y’ = predicted values of Y’ in the previous iteration, Y = actual (gold) values of Y, and |X| = no. of rows in X.

Logistic Regression

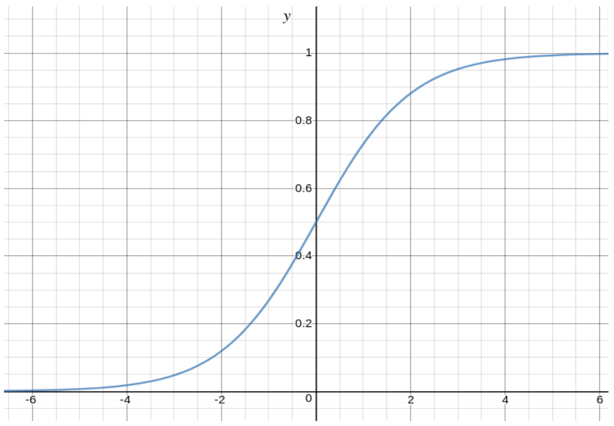
The hypothesis for classification:

$$h(x_i) = g(\beta^T x_i) = \frac{1}{1 + e^{-\beta^T x_i}}$$

where,

$$g(z) = \frac{1}{1 + e^{-z}}$$

is called **logistic function** or the **sigmoid function**.
Here is a plot showing g(z):



We can infer from above graph that:

- g(z) tends towards 1 as $z \to \infty$
- g(z) tends towards 0 as $z \to -\infty$
- g(z) is always bounded between 0 and 1

So, now, we can define conditional probabilities for 2 labels(0 and 1) for i^{th} observation as:

$$P(y_i = 1 | x_i; \beta) = h(x_i)$$
$$P(y_i = 0 | x_i; \beta) = 1 - h(x_i)$$

We can write it more compactly as:

$$P(y_i | x_i; \beta) = (h(x_i))^{y_i} (1 - h(x_i))^{1 - y_i}$$

Now, we define another term, likelihood of parameters as:

$$L(\beta) = \prod_{i=1}^{n} P(y_i|x_i;\beta)$$
$$or$$
$$L(\beta) = \prod_{i=1}^{n} (h(x_i))^{y_i}(1 - h(x_i))^{1-y_i}$$

The **cost function** for logistic regression is proportional to inverse of likelihood of parameters. Hence, we can obtain an expression for cost function, J using log likelihood equation as:

$$J(\beta) = \sum_{i=1}^{n} -y_i log(h(x_i)) - (1 - y_i) log(1 - h(x_i))$$

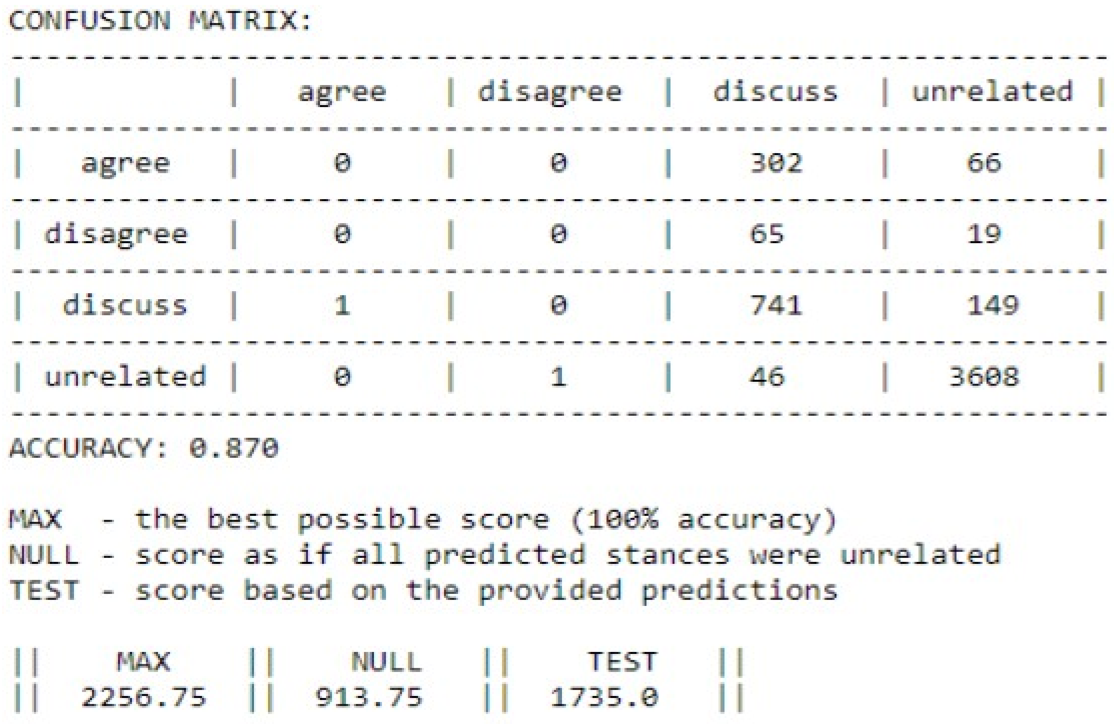
and our aim is to estimate β so that cost function is minimized !!

Similarly we found the accuracy for the following machine learning models:

Linear Regression
Logistic Regression
K Nearest Neighbour
Support Vector Machine
Qudratic Discriminant Analysis
Random Forest
Adaboost
SGD Classifier
Decision Tree
XG Boost
Linear Discriminant Analysis
Gaussian Naive Bayes

Results

1.Score using **Linear Regression** on VALIDATION set -:

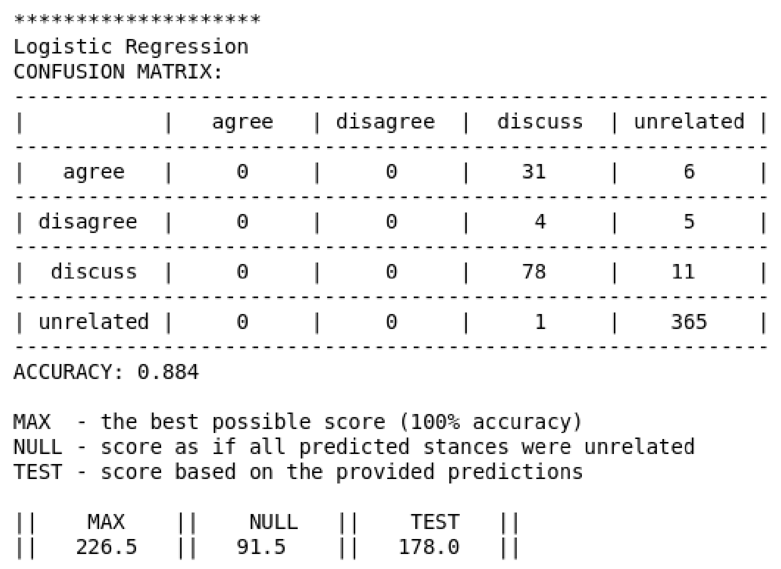


```
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     0     |     0     |    302    |    66     |
-------------------------------------------------------------
| disagree  |     0     |     0     |    65     |    19     |
-------------------------------------------------------------
|  discuss  |     1     |     0     |    741    |    149    |
-------------------------------------------------------------
| unrelated |     0     |     1     |    46     |   3608    |
-------------------------------------------------------------
ACCURACY: 0.870

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL    ||    TEST    ||
||  2256.75  ||  913.75    ||  1735.0    ||
```

2.Score using **Logistic Regression** on VALIDATION set-:

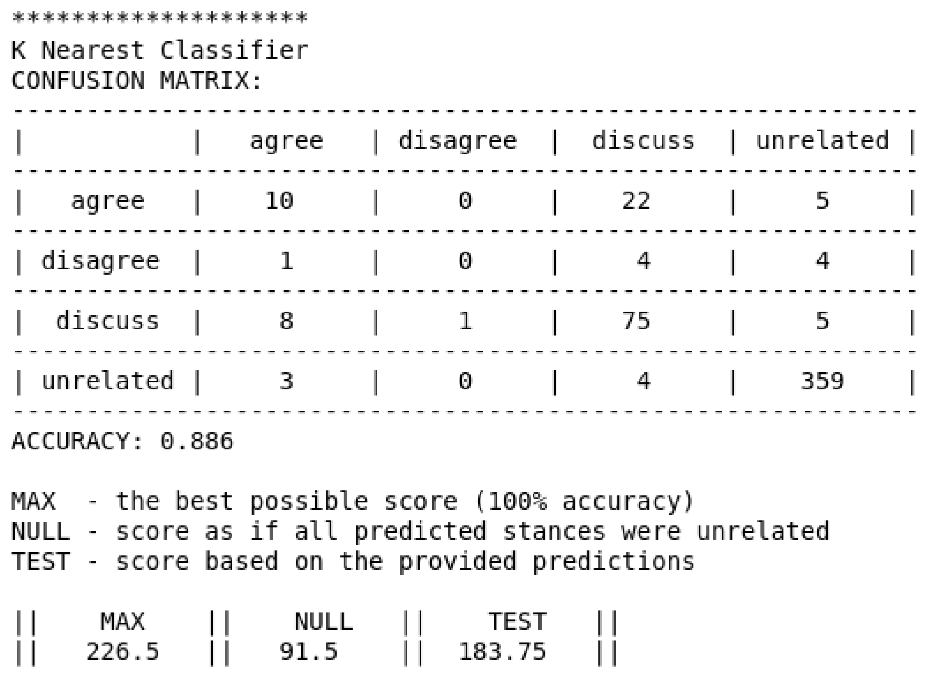


```
********************
Logistic Regression
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     0     |     0     |    31     |     6     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     4     |     5     |
-------------------------------------------------------------
|  discuss  |     0     |     0     |    78     |    11     |
-------------------------------------------------------------
| unrelated |     0     |     0     |     1     |    365    |
-------------------------------------------------------------
ACCURACY: 0.884

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||   178.0   ||
```

3.

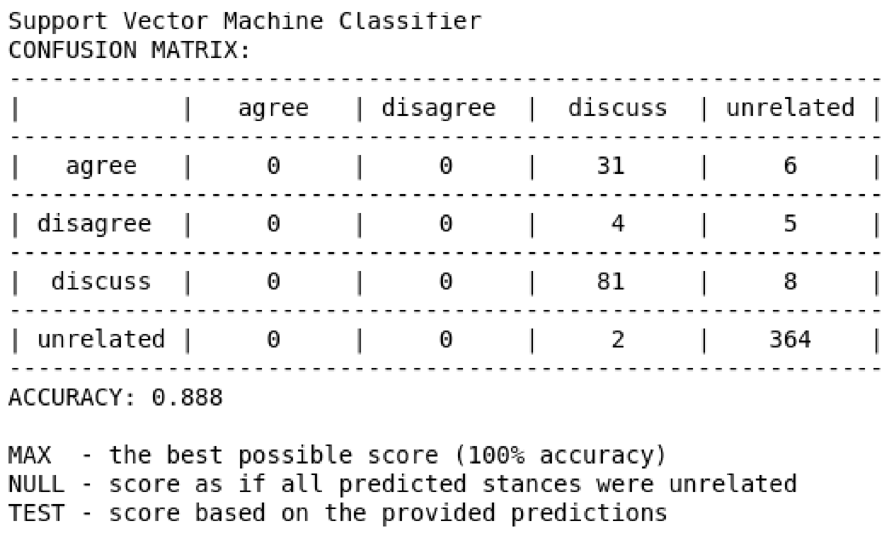


```
********************
K Nearest Classifier
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |    10     |     0     |    22     |     5     |
-------------------------------------------------------------
| disagree  |     1     |     0     |     4     |     4     |
-------------------------------------------------------------
|  discuss  |     8     |     1     |    75     |     5     |
-------------------------------------------------------------
| unrelated |     3     |     0     |     4     |    359    |
-------------------------------------------------------------
ACCURACY: 0.886

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||  183.75   ||
```

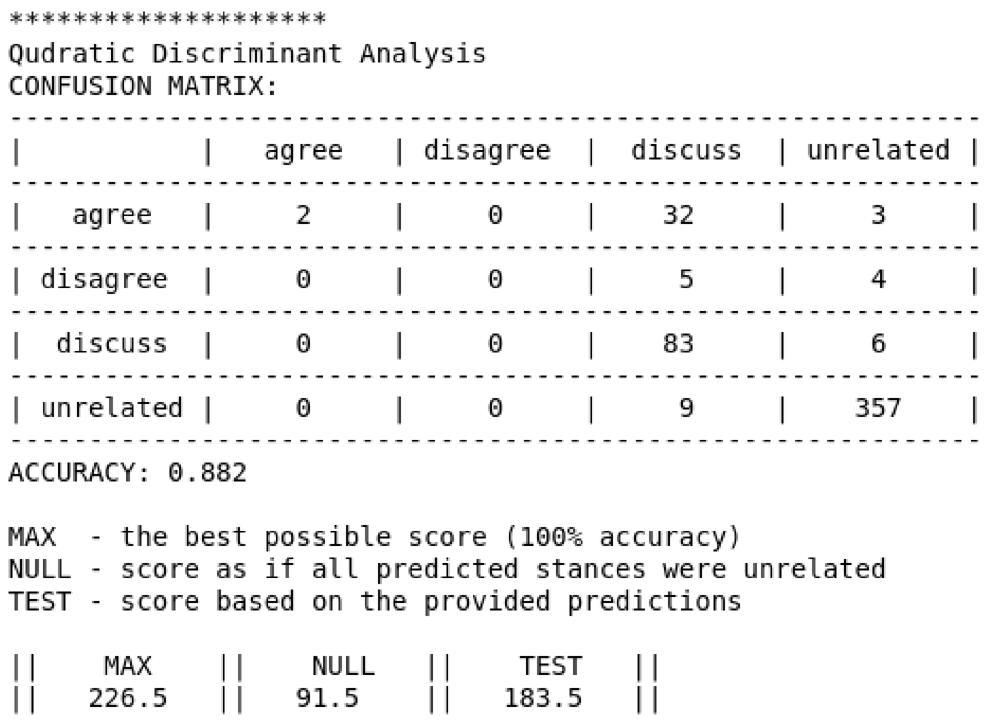
4.



```
Support Vector Machine Classifier
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     0     |     0     |    31     |     6     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     4     |     5     |
-------------------------------------------------------------
|  discuss  |     0     |     0     |    81     |     8     |
-------------------------------------------------------------
| unrelated |     0     |     0     |     2     |    364    |
-------------------------------------------------------------
ACCURACY: 0.888

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions
```

5.



```
********************
Qudratic Discriminant Analysis
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     2     |     0     |    32     |     3     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     5     |     4     |
-------------------------------------------------------------
|  discuss  |     0     |     0     |    83     |     6     |
-------------------------------------------------------------
| unrelated |     0     |     0     |     9     |    357    |
-------------------------------------------------------------
ACCURACY: 0.882

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL    ||    TEST    ||
||   226.5   ||   91.5     ||   183.5    ||
```

6.

```
Random Forest Classifier
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     8     |     1     |    23     |     5     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     5     |     4     |
-------------------------------------------------------------
|  discuss  |     9     |     0     |    72     |     8     |
-------------------------------------------------------------
| unrelated |     5     |     0     |     3     |    358    |
-------------------------------------------------------------
ACCURACY: 0.874

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||   179.0   ||
```

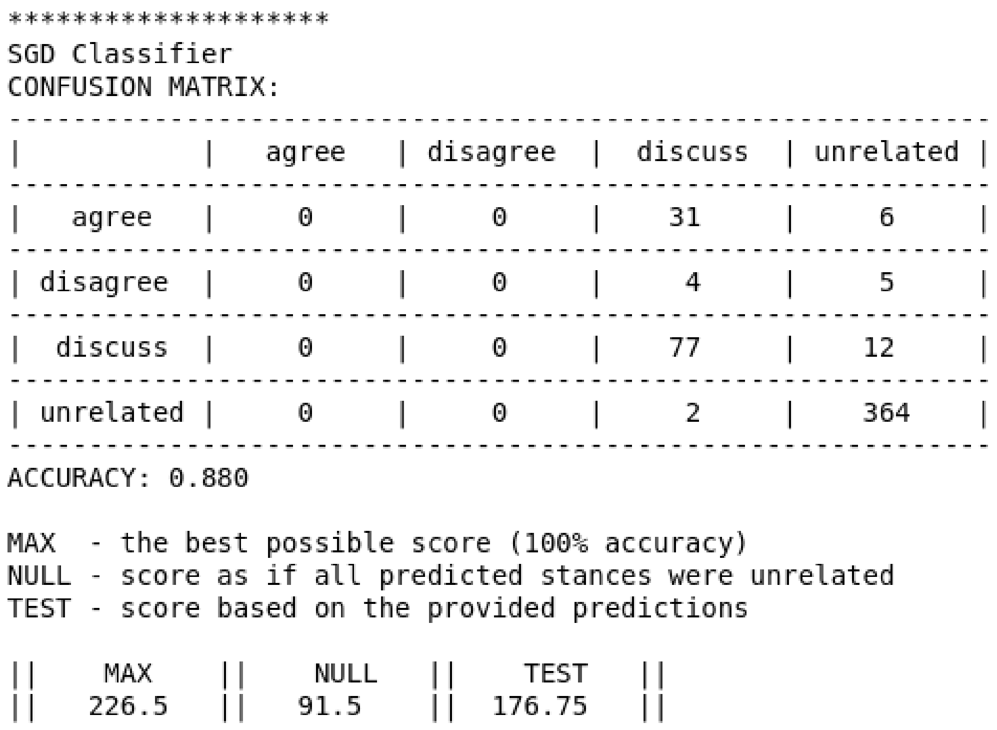
7.

```
********************
Adaboost Classifier
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     1     |     1     |    29     |     6     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     4     |     5     |
-------------------------------------------------------------
|  discuss  |     0     |     1     |    83     |     5     |
-------------------------------------------------------------
| unrelated |     0     |     0     |     9     |    357    |
-------------------------------------------------------------
ACCURACY: 0.880

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||   182.0   ||
```

8.

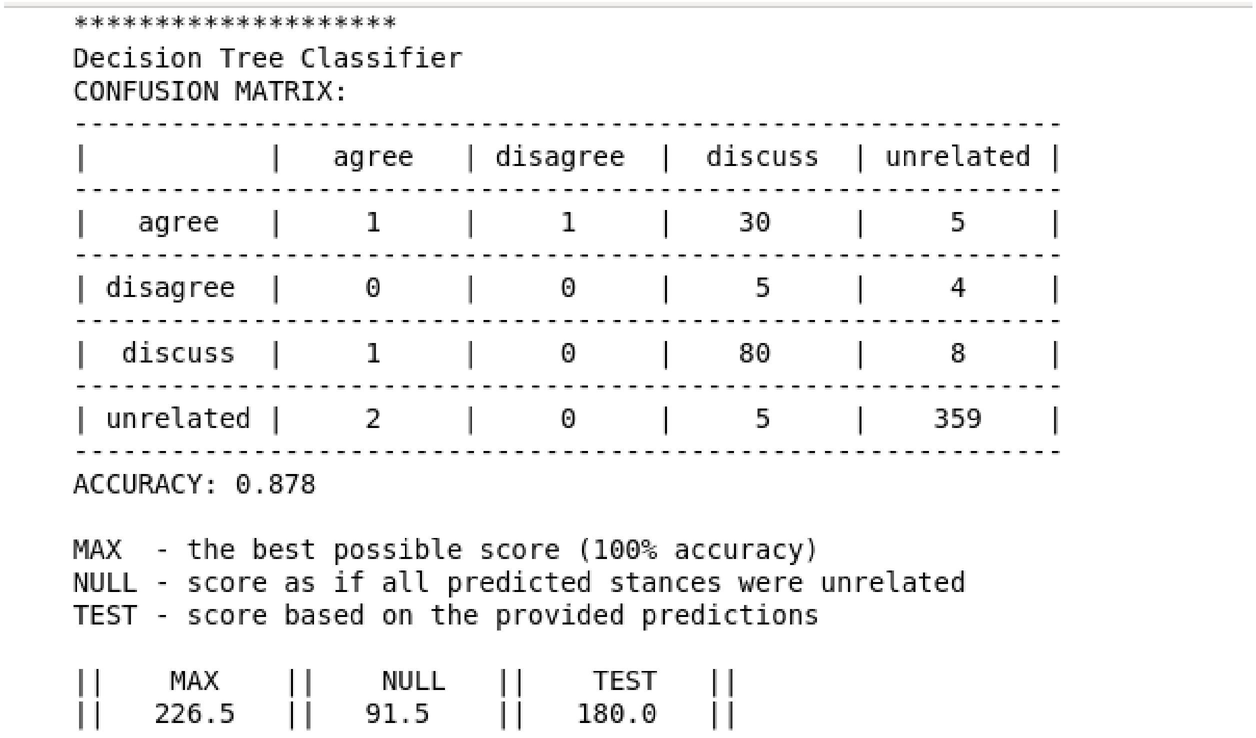


```
********************
SGD Classifier
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     0     |     0     |    31     |     6     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     4     |     5     |
-------------------------------------------------------------
|  discuss  |     0     |     0     |    77     |    12     |
-------------------------------------------------------------
| unrelated |     0     |     0     |     2     |    364    |
-------------------------------------------------------------
ACCURACY: 0.880

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||  176.75   ||
```

9.

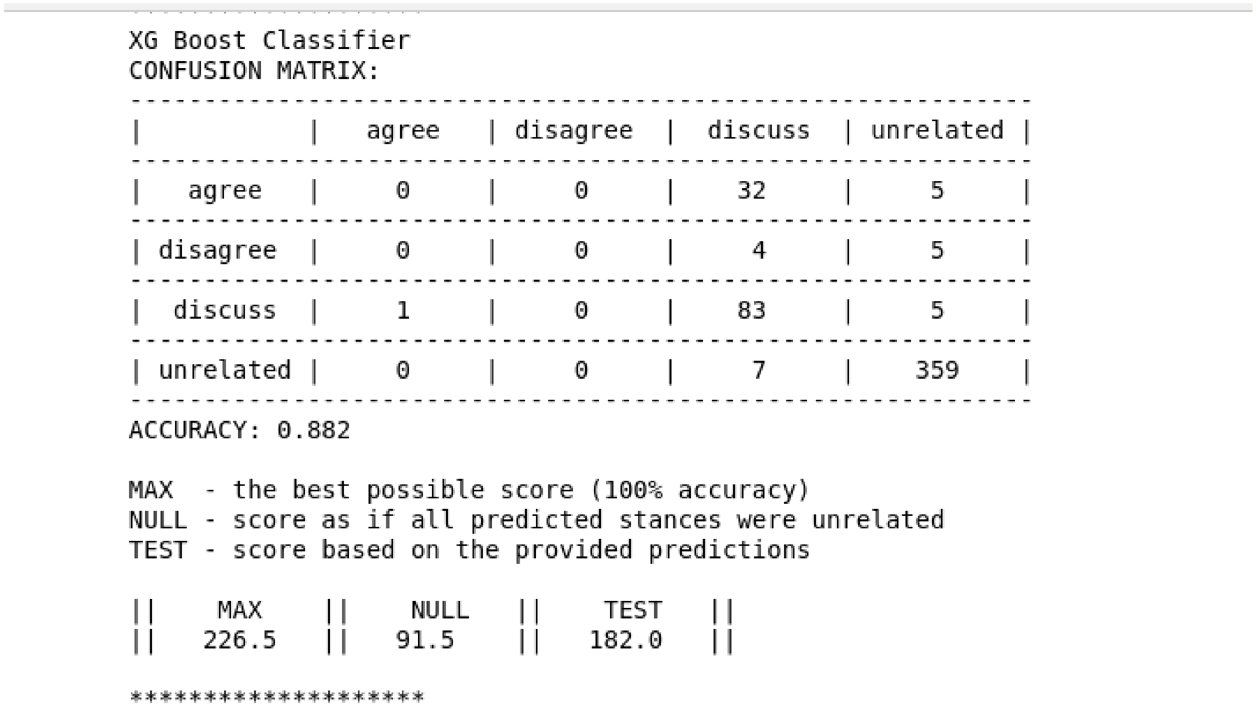


```
********************
Decision Tree Classifier
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     1     |     1     |    30     |     5     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     5     |     4     |
-------------------------------------------------------------
|  discuss  |     1     |     0     |    80     |     8     |
-------------------------------------------------------------
| unrelated |     2     |     0     |     5     |    359    |
-------------------------------------------------------------
ACCURACY: 0.878

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||   180.0   ||
```

10.



```
XG Boost Classifier
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     0     |     0     |    32     |     5     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     4     |     5     |
-------------------------------------------------------------
|  discuss  |     1     |     0     |    83     |     5     |
-------------------------------------------------------------
| unrelated |     0     |     0     |     7     |    359    |
-------------------------------------------------------------
ACCURACY: 0.882

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||   182.0   ||

********************
```

11.

```
********************
Linear Discriminant Analysis
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     0     |     0     |    32     |     5     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     5     |     4     |
-------------------------------------------------------------
|  discuss  |     0     |     0     |    83     |     6     |
-------------------------------------------------------------
| unrelated |     0     |     0     |     7     |    359    |
-------------------------------------------------------------
ACCURACY: 0.882

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||   182.0   ||
```

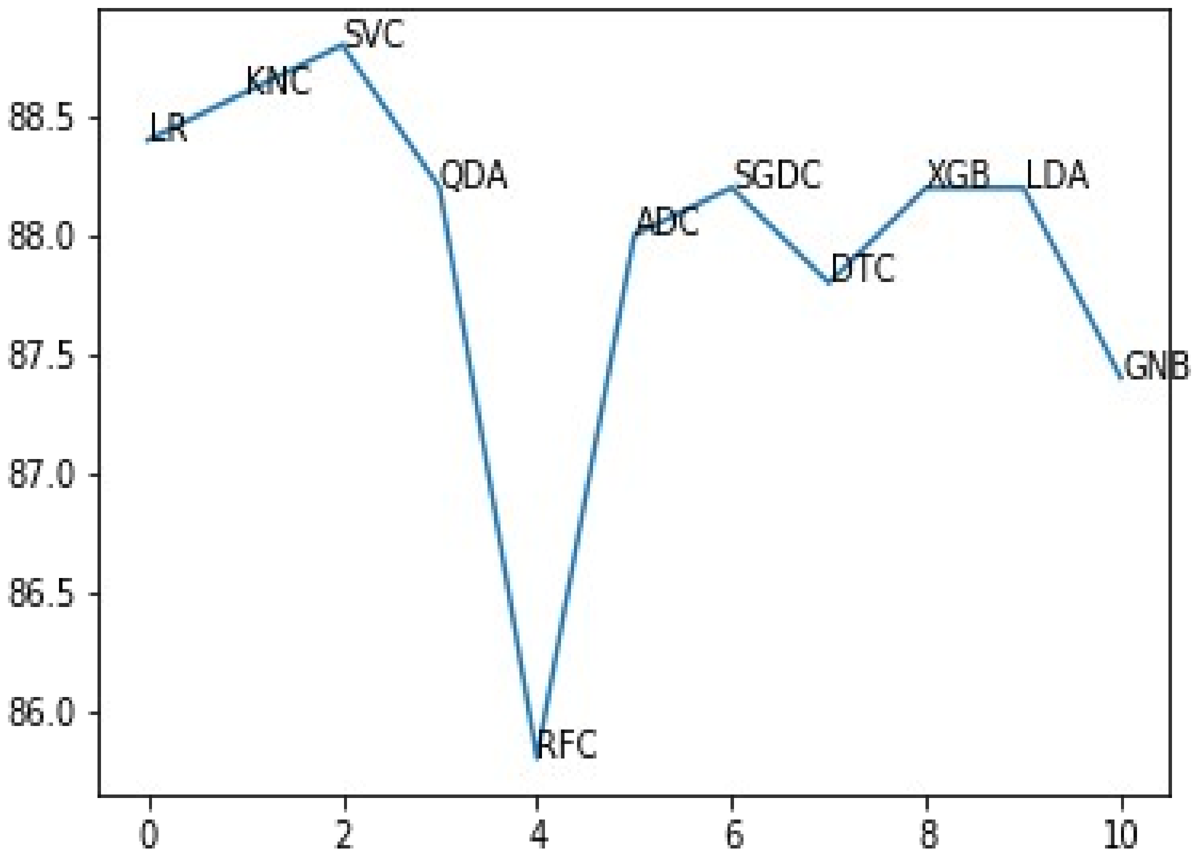
12.

```
********************
Gaussian Naive Bayes
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     1     |     2     |    31     |     3     |
-------------------------------------------------------------
| disagree  |     0     |     0     |     6     |     3     |
-------------------------------------------------------------
|  discuss  |     0     |     0     |    82     |     7     |
-------------------------------------------------------------
| unrelated |     0     |     0     |    11     |    355    |
-------------------------------------------------------------
ACCURACY: 0.874

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
||   226.5   ||   91.5    ||   181.5   ||
```

Comparing the Models:



Problems Faced :-

1.Data is in unorganised form and we had to remove various stop words and noise from the data as the accuracy drastically reduced without doing it,most prominently the Support vector machine classifier gives the poor accuracy.

2.Data set size is also massive and many times gives a memory error while making a high dimensional matrix so we had to sample the data set and apply the algorithms for separate data units.

3.We also had to adjust the number of iterations and learning rate of the data set so as to improve the accuracy.

4.Use of a powerful GPU is needed for training the word embedded model on the large corpora of text.

Conclusion

- As we can see from these results, the model performs well in predicting ‘unrelated’ articles correctly, but is unable to distinguish between ‘agree’, ‘disagree’, and ‘discuss’ labels – in fact, it just predicts every article with a low ‘unrelated’ score as ‘discuss’ as it is the second most common label, and with one or two exceptions do not predict anything as ‘agree’ or ‘disagree’.

- We achieved the highest accuracy in Support Vector Machine Classifier with highes accuracy of 88.8%

- We achieved the poorest accuracy in Random Forest Classifer with accuracy 85.2% and Linear Regression with accuracy 83.2%

- All the other models performed decently but needed the dataset to be cleaned and tuned according the models used to get best results.

Future Works

- Currently, the model developed performs well in predicting ‘unrelated’ labels vs other labels, but is unable to classify between ‘discuss’, ‘agree’, and ‘disagree’. The main reasoning behind this is because the features extracted in this model (cosine similarity of word vectors, KL-divergence of language models, 3-gram overlap) performs well in distinguishing ‘unrelated’ labels vs other labels, but are less useful to distinguish between ‘discuss’, ‘agree’, and ‘disagree’, as shown in the distance distribution plot.

- To distinguish between ‘discuss’, ‘agree’, and ‘disagree’, features that look beyond word-similarity would be required. For example, aspect-based sentiment models could be used to measure the divergence in sentiment between the headline and the article body towards common topics. However, this was infeasible as these sentiment models are quite complex.

- We can use various deep learning models and improve their accuracy and visualise the various parameters that works in improving the accuracy

References

1. Augenstein, Isabelle, et al. "Stance detection with bidirectional conditional encoding." arXiv preprint arXiv:1606.05464, 2016.

2. Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. " machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473, 2014.

3. Bowman, Samuel R., et al. "A large annotated corpus for learning natural language inference." arXiv preprint arXiv:1508.05326, 2015.

4. Ferreira, William, and Andreas Vlachos. "Emergent: a novel data-set for stance classification." Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. ACL, 2016.