Lion Ben Nedava 302680665
Amit Shakarchy 313278889

Assignment 3 - Authorship Attribution

Input text preprocessing:

What data/features were finally used - how the corpus was preprocessed and filtered.

1. Text Normalization:

   The tweets normalization and preprocessing procedure include the following operations:

   a. Conversion of all letters of the string to lowercase
   b. Removal of redundant white spaces
   c. Removal of punctuation
   d. Word tokenization
   e. Construct a representation of the tweet (word embedding)

2. Word Embedding methods:

   We examined several methods for representing the textual data. For all of the examined methods, we used a vector size of 50 dimensions only, to meet the instructions constraints (embeddings file shouldn't be larger than **6**MB).

   a. TF-IDF- Using TfidfVectorizer, to try and highlight interesting words - that are frequent in a tweet but not across all the tweets.
   b. W2V - Using two sets of pre-trained word embeddings (Glove, Gensim [1,2]) to vectorize each tweet. Each tweet then was represented by either
      i. averaging sentence's vector out of all words' vectors
      ii. splitting each sentence into smaller sentences with 14 words each and concatenating their reduced vectors.

   We tested all 5 embeddings (TF-IDF, mean W2V glove, mean W2V gensim, concat W2V glove, concat W2V gensim) across all algorithms.

   Both the TF-IDF and the W2V vectors are very large (vocabulary size and 300 accordingly), so we performed PCA on the vectors to reduce them to 50, the PCA reduces the dimensionality but keeps the similarity between vectors.

3. Since the dataset was quite balanced 70% iPhone and 30% android, we didn't do any upsampling or downsampling.

General training Pipeline:

We trained each model using 5-fold cross-validation. At the end of this procedure, we calculated the average score of several metrics: accuracy, precision, recall, AUC, and f1. We compared the trained models using the average accuracy score, to pick the one with the best performances. We then trained our best model on all of the given trainset, for a better generalization.

Models Description:

1. Logistic Regression - using the default parameters.
2. SVM - We evaluated the performance of SVM with three kernels ('linear', 'rbf').
3. DNN (dense neural network) - constructed out of 2 hidden layers (input,64,32,2).
4. LSTM - 2 stacked LSTM and the 2 fully connected layers (64,16,2).
5. LSTM architecture combined with added features- The model was constructed to examine how a combination of numerical (extracted from the textual data), and textual features could affect the LSTM's performance. The model receives two input vectors treated separately (Tweets embeddings, and the extracted features). The embeddings go through the LSTM architecture, then concatenate with the extracted features vector. The combined vector, which serves as a full representation of a tweet, goes through the sigmoid output layer
   calculated features:
   a. A tweet's number of characters
   b. A tweet's number of words
   c. Average word length
   d. A tweet's number of punctuation characters
   e. A tweet's number of unique words
   f. A tweet's number of uppercase words (e.g., "HELLO", "AMAZING")
   g. A tweet's number of stopwords
   h. A tweet's number of hashtags
   i. A tweet's number of mentions

Parameters:

** Note: The parameters of all three NN are identical.

Parameter	Value
batch size	32
criterion	Cross-Entropy Loss
Optimizer	Adam
Learning rate	0.001
Number of epochs	50
dropout	0.2

Table 1: Parameters used in the deep networks

Algorithms Comparison:

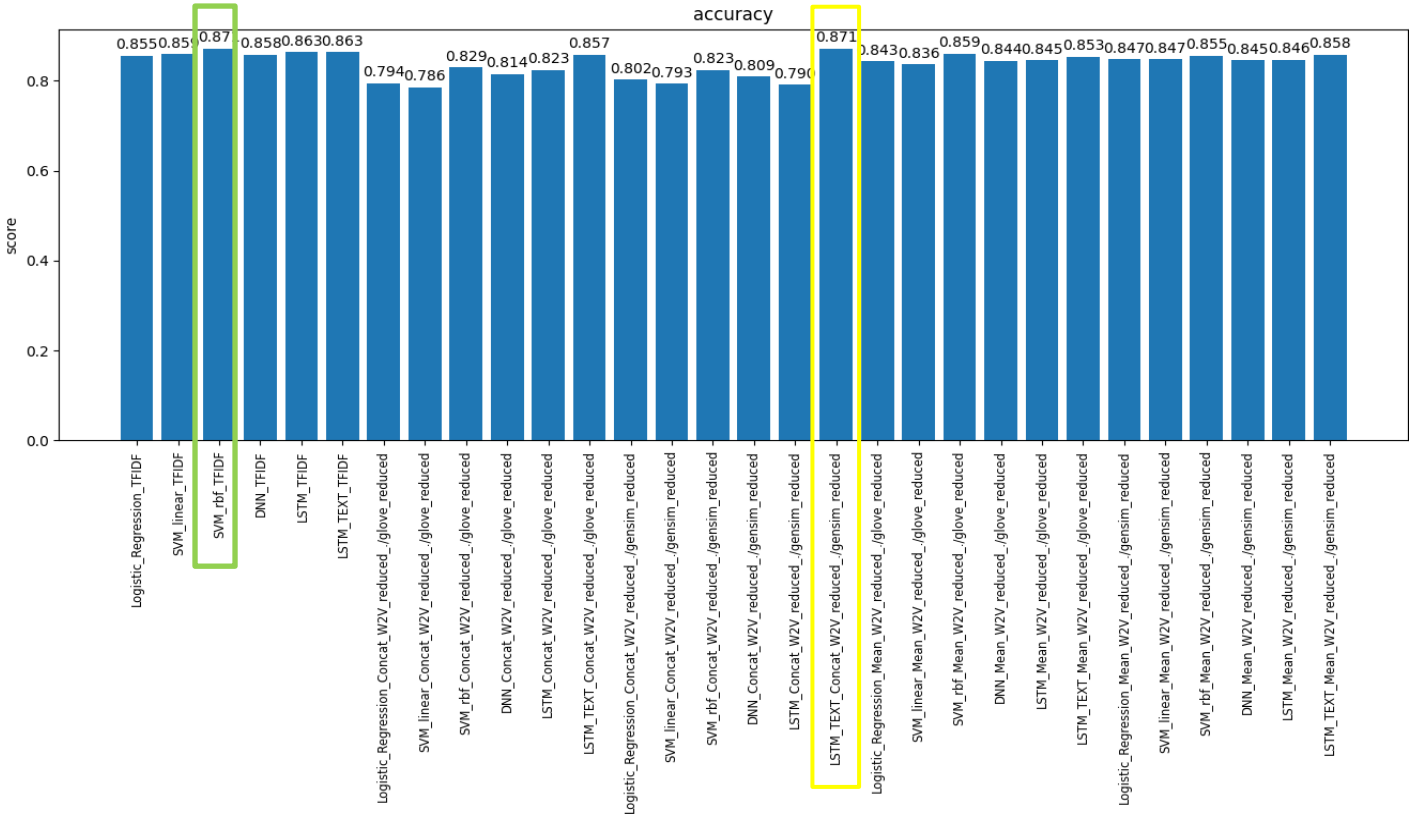

Figure 1: Accuracy score across all algorithms and vectorization methods

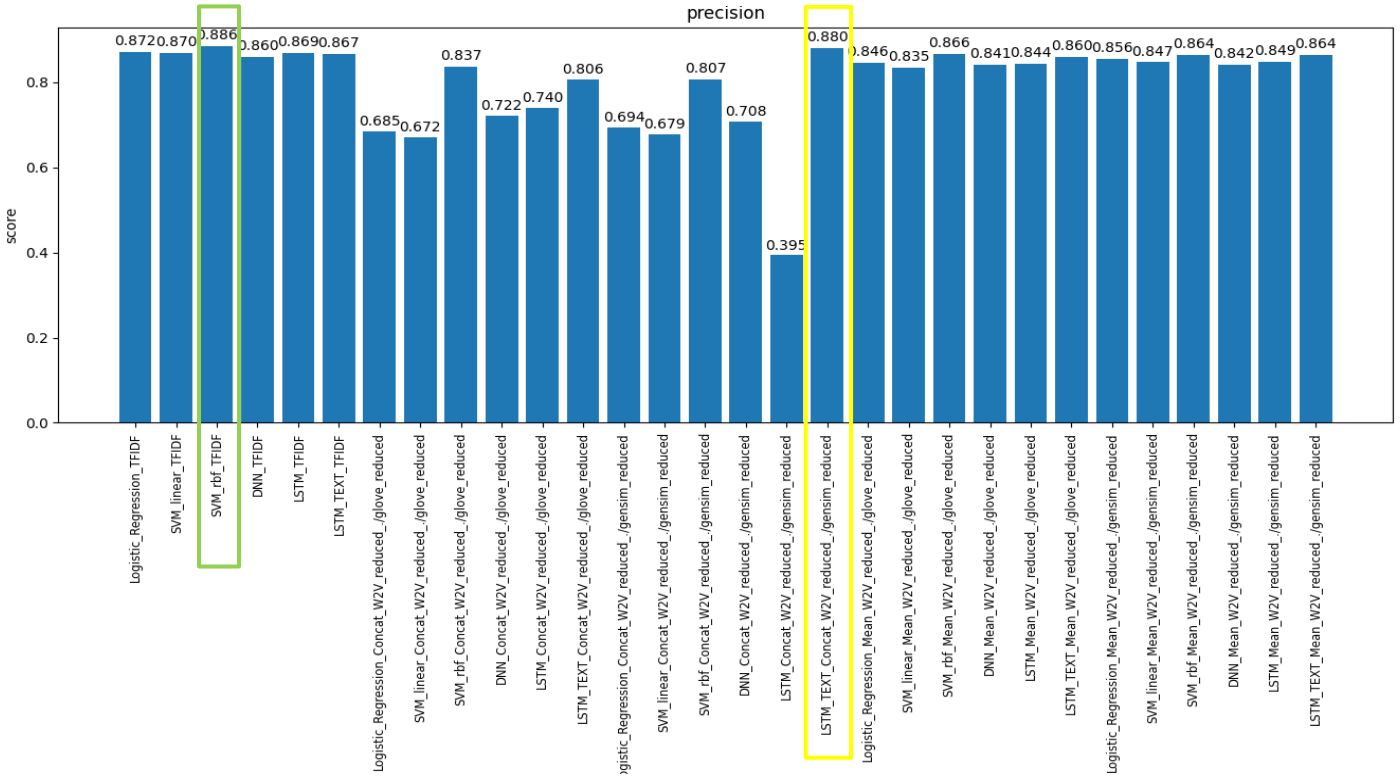

Figure 2: Precision score across all algorithms and vectorization methods

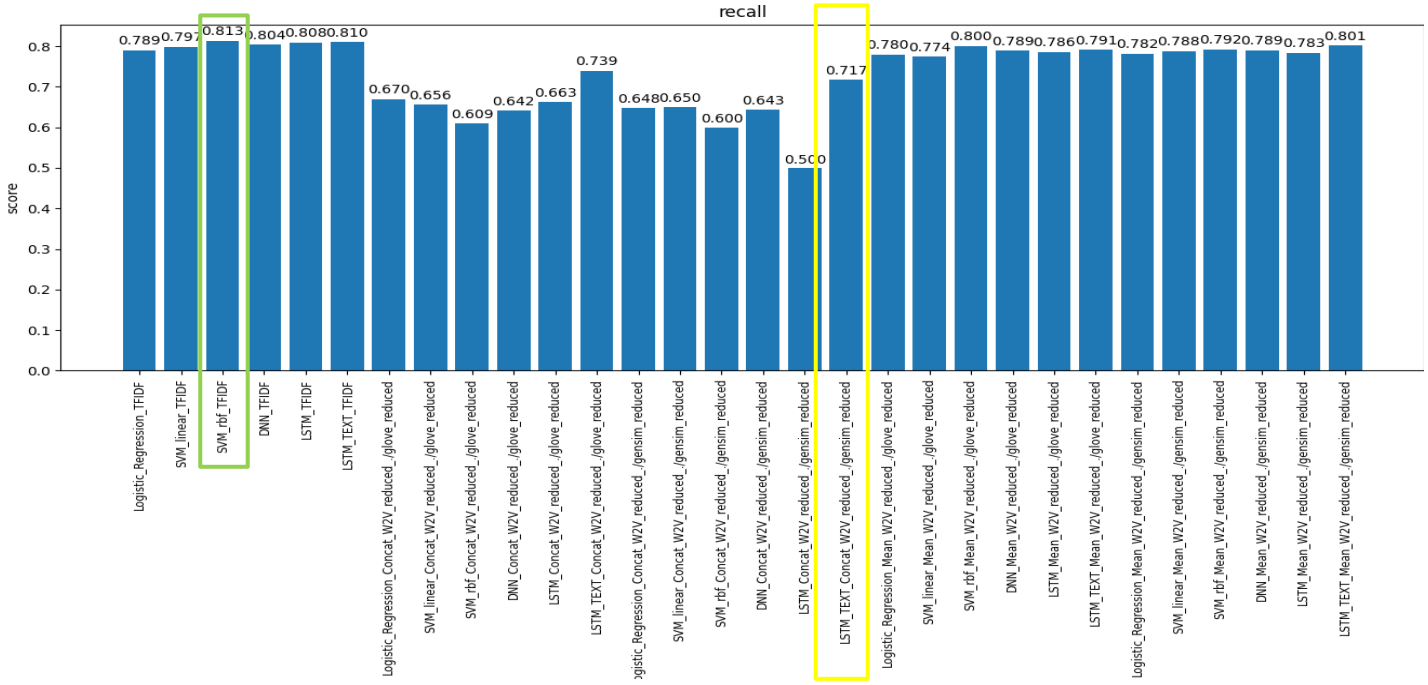

Figure 3: Recall score across all algorithms and vectorization methods

In all three figures, displaying the accuracy, precision, and recall, we can see similar trends. TF-IDF and mean W2V give better results than concatenating W2V vectors. Nevertheless, we expected that the concatenation of vectors would poorly affect all the algorithms except the LSTMs, and there indeed we see similar results to the other vectorization methods.
We can also see that Logistic Regression, the simplest algorithm, has performed less well than the others, but given its simplicity, it is surprisingly not too far behind.
The 2 algorithms that show the best performances are SVM with rbf kernel and the LSTM with the meta-features added to it (marked in green and yellow accordingly).
The SVM has the best results in all the metrics and so it is our best algorithm, the LSTM with meta-features performance is second best, except for the recall. In our task of text classification, both precision and recall are important.

Extra: A verification of the claim that Trump was kept away from his Twitter account during the campaign:

```
true:
[0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0]
prediction:
[0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0,
 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0]
{'accuracy': 0.8387096774193549, 'precision': 0.8628762541806021, 'recall': 0.8387096774193549, 'auc': 0.8387096774193549, 'f1': 0.8148148148148149}
```

Figure 4: Best model predictions vs true label and metrics of 1 month before the 2016 US elections

According to figure 4, we can see that our SVM model with TFIDF predicts that Trump kept tweeting, and in comparison, to the real labels we have 86% precision which says that all the tweets we claimed to trump at least most of them are true.
Also, we have a high recall score, so it means we found most of Trump's tweets.

References:

1. https://nlp.stanford.edu/projects/glove/
2. https://radimrehurek.com/gensim/intro.html