Amarandei Alexandru, Mosor Andrei, Pelin Ioana, Smău Adrian

Team-Leader: *Cușmuliuc Ciprian Gabriel*

April 2021

CLEF2021 - CheckThat! Lab

Week 2 Report

Introduction

This team was assigned with the task of implementing an algorithm for detecting Fake News in Python. During the first week, the team's work consisted mostly of research and analysis of basic classifiers and encoding methods. This week's run had two other goals, improving performance on the current solutions and trying a BERT-based approach.

Methodology

Overview

There were two main goals involved, one being exploring hyper-parameter tuning, along with a more in-depth result analysis, and one being creating a rudimentary BERT (Bidirectional Encoder Representations from Transformers). As a result, the team split into two sub-teams focusing on each of the individual goals.

The reasoning behind the first goal was that by tuning the hyper-parameters of the models used and by trying out different representations of the data (e.g. Bag-of-Words, Doc-Term, Frequency and TFIDF (Term-Frequency Inverse Document Frequency)) the team may find the best-performing combination and thus obtain an optimal set of parameters and representations.

The reasoning behind the second goal was to extend the areas covered by the team and explore new technologies that may yield better results than the common approaches.

Hyper-Parameter Tuning and Result Analysis

Data Preparation

The Data Preparation stage consisted of all of the steps taken in the last week's run, involving:

- removal of the **title** and **public_id** columns from the CSV file received
- removal of punctuation signs
- removal of stop-words
- removal of dashes and underscores
- lowercasing the text
- lemmatization of text

After "cleaning" the text, the sub-team created four matrix-encodings, namely, Bag-of-Words, Doc-Term, Frequency and TFIDF. The **our rating** column was also converted into number format as follows:

- *False*: 0
- *True*: 1
- *Partially-False*: 2

Randomized Search Cross Validation

Using RandomizedSearchCV the sub-team tried to find the best hyper-parameters for the <u>DT</u> (Decision Tree), <u>SVM</u> (Support Vector Machine), <u>NB</u> (Naive-Bayes) and <u>KNN</u> (K Nearest Neighbours) classifiers.

After finding the parameters needed, the models were trained and tested, using all of the above-mentioned representations.

BERT

Data Preparation

The steps taken by the sub-team responsible for this approach differed from the ones discussed above, as the BERT model has distinct requirements for the representation of the data. They are as follows:

- removal of the **title** and **public_id** columns from the CSV file received
- data was balanced out, as the distribution of the *True*, *False* and *Partially-False* labels was significantly uneven
- data was split into training and testing sets in a 50% / 50% ratio
- data was converted into a list of dictionaries with **text** and **our rating** keys
- a list of tuples was generated from the dictionaries
- tokens and respective token-ids were generated from the tuples (and the text was also lowercased in the process)

Model Building

After defining the required *initialization* and *forward* methods, training and testing token tensors were generated, the last step involving the preparation of the data loaders.

Fine Tuning

The sub-team used the *Adam* optimizer in order to minimize the Binary Cross Entropy loss. The training was done using a Batch Size of 1 for 1 Epoch.

After the final step above, the model was evaluated. The results are discussed in the section below.

Results

Hyper-Parameter Tuning and Result Analysis

Doc Term Representation

	precision	recall	f1-score	support
0	0.50	0.50	0.50	2
1	0.25	0.33	0.29	3
2	0.75	0.60	0.67	5
accuracy			0.50	10
macro avg	0.50	0.48	0.48	10
weighted avg	0.55	0.50	0.52	10

Decision Tree

	precision	recall	f1-score	support
0	0.60	0.60	0.60	5
1	0.00	0.00	0.00	1
2	0.50	0.50	0.50	4
accuracy			0.50	10
macro avg	0.37	0.37	0.37	10
weighted avg	0.50	0.50	0.50	10

Support Vector Machine

	precision	recall	f1-score	support
0	1.00	0.20	0.33	5
1	0.25	1.00	0.40	1
2	0.20	0.25	0.22	4
accuracy			0.30	10
macro avg	0.48	0.48	0.32	10
weighted avg	0.60	0.30	0.30	10

Naive Bayes

	precision	recall	f1-score	support
0	0.50	1.00	0.67	5
1	0.00	0.00	0.00	1
2	0.00	0.00	0.00	4
accuracy			0.50	10
macro avg	0.17	0.33	0.22	10
weighted avg	0.25	0.50	0.33	10

K Nearest Neighbours

Bag of Words Representation

	precision	recall	f1-score	support
0	0.50	0.67	0.57	3
1	0.33	0.50	0.40	2
2	0.33	0.20	0.25	5
accuracy			0.40	10
macro avg	0.39	0.46	0.41	10
weighted avg	0.38	0.40	0.38	10

Decision Tree

	precision	recall	f1-score	support
0	0.30	1.00	0.46	3
1	0.00	0.00	0.00	4
2	0.00	0.00	0.00	3
accuracy			0.30	10
macro avg	0.10	0.33	0.15	10
weighted avg	0.09	0.30	0.14	10

Support Vector Machine

	precision	recall	f1-score	support
0	1.00	0.33	0.50	3
1	0.60	0.75	0.67	4
2	0.50	0.67	0.57	3
accuracy			0.60	10
macro avg	0.70	0.58	0.58	10
weighted avg	0.69	0.60	0.59	10

Naive Bayes

	precision	recall	f1-score	support
0	0.30	1.00	0.46	3
1	0.00	0.00	0.00	4
2	0.00	0.00	0.00	3
accuracy			0.30	10
macro avg	0.10	0.33	0.15	10
weighted avg	0.09	0.30	0.14	10

K Nearest Neighbours

Frequency Representation

	precision	recall	f1-score	support
0	0.50	1.00	0.67	4
1	0.00	0.00	0.00	3
2	1.00	0.67	0.80	3
accuracy			0.60	10
macro avg	0.50	0.56	0.49	10
weighted avg	0.50	0.60	0.51	10

Decision Tree

	precision	recall	f1-score	support
0	0.50	1.00	0.67	5
1	0.00	0.00	0.00	2
2	0.00	0.00	0.00	3
accuracy			0.50	10
macro avg	0.17	0.33	0.22	10
weighted avg	0.25	0.50	0.33	10

Support Vector Machine

	precision	recall	f1-score	support
0	1.00	0.50	0.67	6
1	0.00	0.00	0.00	1
2	0.00	0.00	0.00	3
accuracy			0.30	10
macro avg	0.33	0.17	0.22	10
weighted avg	0.60	0.30	0.40	10

Naive Bayes

	precision	recall	f1-score	support
0	1.00	0.67	0.80	6
1	0.00	0.00	0.00	1
2	0.40	0.67	0.50	3
accuracy			0.60	10
macro avg	0.47	0.44	0.43	10
weighted avg	0.72	0.60	0.63	10

K Nearest Neighbours

TFIDF Representation

	precision	recall	f1-score	support
0	1.00	0.25	0.40	4
1	0.50	0.50	0.50	4
2	0.00	0.00	0.00	2
accuracy			0.30	10
macro avg	0.50	0.25	0.30	10
weighted avg	0.60	0.30	0.36	10

Decision Tree

	precision	recall	f1-score	support
0	0.43	0.75	0.55	4
1	0.00	0.00	0.00	3
2	0.33	0.33	0.33	3
accuracy			0.40	10
macro avg	0.25	0.36	0.29	10
weighted avg	0.27	0.40	0.32	10

Support Vector Machine

	precision	recall	f1-score	support
0	1.00	0.50	0.67	6
1	0.00	0.00	0.00	1
2	0.00	0.00	0.00	3
accuracy			0.30	10
macro avg	0.33	0.17	0.22	10
weighted avg	0.60	0.30	0.40	10

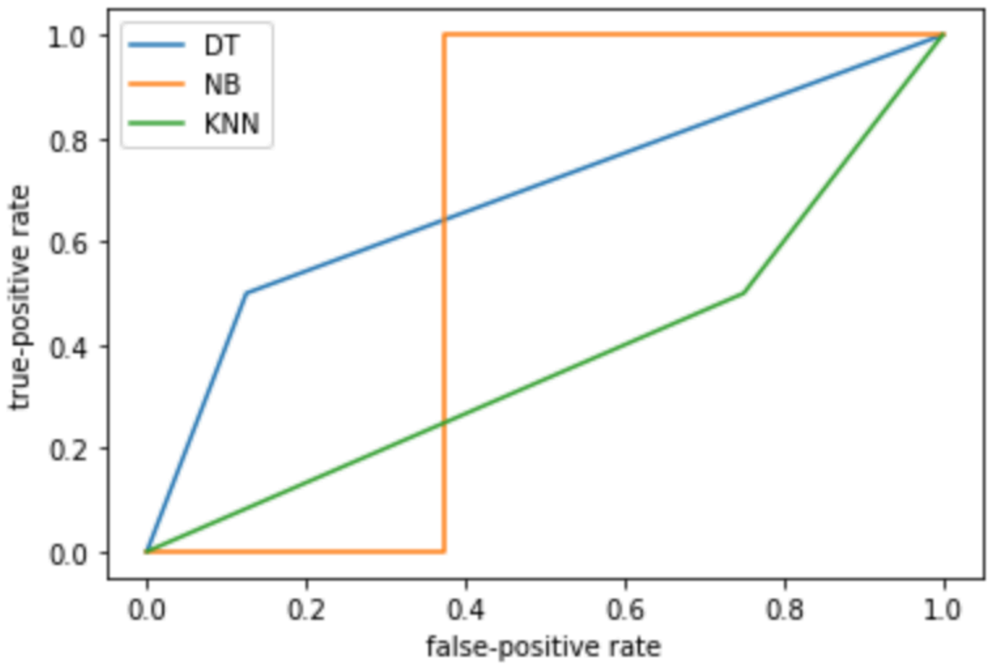
Naive Bayes

	precision	recall	f1-score	support
0	0.40	1.00	0.57	4
1	0.00	0.00	0.00	3
2	0.00	0.00	0.00	3
accuracy			0.40	10
macro avg	0.13	0.33	0.19	10
weighted avg	0.16	0.40	0.23	10

K Nearest Neighbours

ROC / AUC

~ Mention: this representation was done on a binary output (merging the false-labeled and partially-false-labeled subsets, resulting in only *True* and *False*)




```
AUC for DT: 0.687500
AUC for NB: 0.625000
AUC for KNN: 0.375000
```

BERT

	precision	recall	f1-score	support
False	0.69	1.00	0.81	176
True	0.00	0.00	0.00	80
micro avg	0.69	0.69	0.69	256
macro avg	0.34	0.50	0.41	256
weighted avg	0.47	0.69	0.56	256

Discussion

Hyper-Parameter Tuning and Result Analysis

The results were diverse, but the true purpose of this trial was to discover whether or not hyper-parameter tuning could influence the performance of the models to a significant degree. The answer is yes and thus, hyper-parameter tuning will be a tool that will be taken advantage of during future runs.

The ROC / AUC evaluation system would have been useful in a different context, but considering the fact that there are three labels, this system will be discarded in future runs.

BERT

The performance of the model has exceeded the expectations of the team, as there are many ways in which the model could be improved, performing very well for a first-try run.

The results are, consequently, encouraging and are enough of a reason for the sub-team to continue pursuing this approach.

Future Approaches

Data Preparation

The **title** and **text** columns will be merged and considered as a whole when analyzing and encoding texts.

All non-alphabetical characters will be removed from the text.

POS Tagging

POS (Part of Speech) Tagging will be tested on the unmodified text in order to reveal if the syntactic analysis of the text can provide relevant statistics regarding the truthfulness of the tweets.

Sentiment Analysis

Sentiment analysis will also be considered during future runs as the topics of discussion or categories of words in a tweet may be relevant in determining its proper labelling.

BERT Improvements

The BERT model will continue to be further studied and improved on during future runs, as this week's results are promising.