COMP3222 Assignment: MediaEval

Signe Rebassoo [ID: 29554896]
sr2u17@soton.ac.uk

January 10, 2020

1 Introduction and Data Analysis

After a high impact event, a lot of controversial information goes viral on social media. In order to verify their sources and be true to their code of faithfulness to reality and objectivity, news professionals would have to debunk it all and decide whether the shared multimedia represents real and accurate information. Since there is lack of publicly accessible tools for doing so, the task at hand is to come up with a machine learning algorithm to classify viral social media content propagating fake images or presenting real images in a false context.

1.1 Problem Characterization

The problem defined identifies posts as fake when they are:

a) reposts of real multimedia, such as real photos from the past re-posted as being associated to a current event;

b) digitally manipulated;

c) synthetic, such as artworks or snapshots presented as real imagery.

However, the problem given for this assignment does not apply machine learning on images themselves. In the context of this problem, we apply machine learning on the textual tweet data in order to improve the identification of fake multimedia posts. The approach could then perhaps be used in conjunction with machine learning algorithms that process image data to tackle the problem more thoroughly. Though, this extension is out of the scope of the algorithm to be implemented for this instance.

Therefore, the features we can work with will be built around natural language processing using different forms of word count vectorisation.

The implementation of the algorithm wouldn't need to have any requirements for the computational speed as we are dealing with static data and the results of the algorithm will not be reflected by the customer in any online context.

The main performance measure used for evaluating the different approaches to this algorithm is the F1 score. Some additional insight into the performance will also be provided by the use of confusion matrices.

1.2 Data Characterization

The given MediaEval 2015 data set consists of labelled offline data from Twitter, where the multimedia posts have been deemed either 'fake', 'real', or 'humor'. For this algorithm, 'humor' will be treated as 'fake', so the original 3 classes will be narrowed down to 2 - 'fake' and 'real'. The actual multimedia is present in the dataset as a link to the source within the tweetText field, but the main data for analysis is represented by text and metadata.

The dataset has already been broken down into a training and a testing set, both of which are formatted in a text file (.txt) with each row on a new line and columns separated by a tab character. The obtained column fields from the dataset are tweetId, tweetText, userId, imageId(s), username, timestamp, and label.

The training set has 14,277 posts and test set 3,755 so the entire data set consists of 18,032 Twitter posts. As for event coverage, the training set has 11 events/phenomena: terrorist attack at the Boston Marathon (boston), Chibok schoolgirls kidnapping (bringback), chemical plant explosion in Columbia (columbianChemicals), elephant shaped rock (elephant), the Livr social network (livr), missing Malaysia Airlines flight MH370 (malaysia), a father left stranded in South Korea after his child had drawn on the passport (passport), discovery of pigfish in Arkansas (pigFish), Hurricane Sandy (sandyA, sandyB), Sochi olympic games (sochi), and and underwater bedroom in Fiji (underwater).

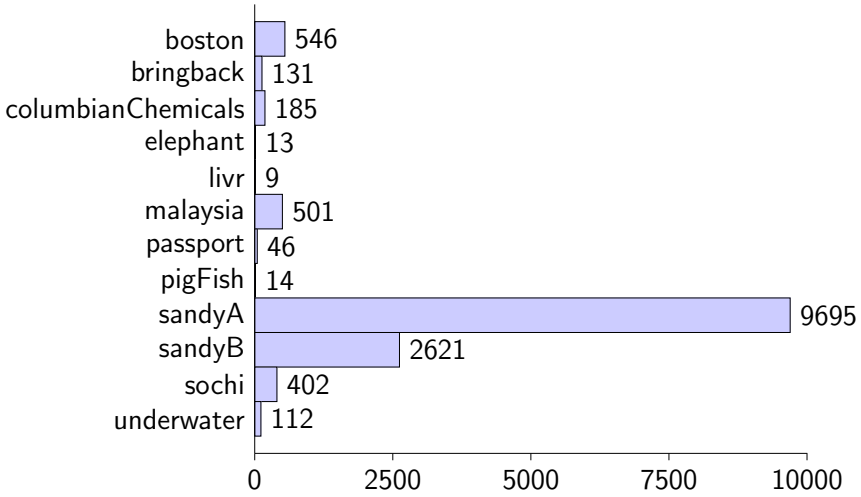


Figure 1: Number of posts per image/event in training data

The graph above indicates a significant bias towards posts about Hurricane Sandy with a total of 12,318 posts about it in the training data, which means repeating words in these posts such as 'hurricane' and 'Sandy' would be learned as features which would not apply to any other events.

As for the testing set, it covers 6 events: the solar eclipse (eclipse), shooting at Garissa University in Kenya (garissa), an earthquake in Nepal (nepal), a paranormal samurai ghost (samurai), a Syrian boy rescuing a girl in a shootout (syrianboy), and a satire video on Varoufakis (varoufakis).

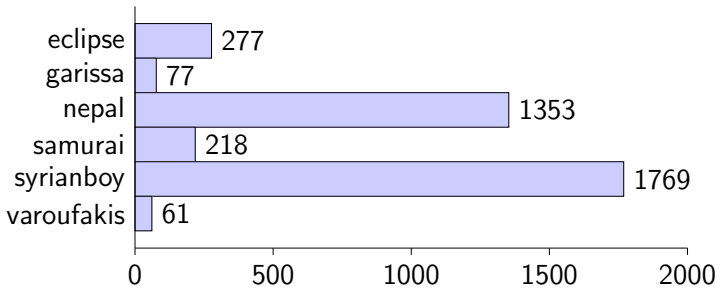


Figure 2: Number of posts per image/event in testing data

As can be seen, the testing data has a considerably higher coverage of the Syrian boy and the earthquake in Nepal.

The majority of these posts are in English - 10,968 and 2,781 in the training and testing data respectively, but there are many other languages included in the dataset (see Appendix A, B). Of course, the accuracy of these reported numbers is dependent on that of the language detection of Google's langdetect. Taking into account that many tweets contain slang, misspelled words, abbreviations, and emojis, the language detection is likely not very accurate due to this noise in the data. Furthermore, some tweets contain random appearances of 'amp;' and '\n'. The effect of this noise is also apparent from the fact that Google's language detection was unable to identify the language of the following tweet full of emojis and misspelled words:

It is also worth noting that some events may be covered majorly in a foreign language. For example, the posts about the Varoufakis satire video in the testing set are all in German, which means the algorithm would have to be flexible in terms of language for better results.

2 Algorithm Design

2.1 Preprocessing

The first step taken was to change the 'humour' labels to 'fake' as the task is to implement a binary classifier that labels posts as either 'fake' or 'real'.

Due to a heavy bias in the dataset towards certain events, most of which is caused by numerous retweets of the same posts, removing those repetitive posts from the dataset seemed a reasonable move to avoid overfitting the model to specific features that may originate from Hurricane Sandy posts as the algorithm needs to be flexible for twitter posts related to any event. In order to catch all such tweets, a set of regular expressions was used that took into account retweets, reposts, and modified tweets.

On the other hand, keeping retweets, reposts and modified posts may be beneficial for the algorithm since they were brought out as characteristics of fake posts in Section 1.1 Problem Characterization. In problem characterization, those aspects are mentioned from an image

manipulation point of view, but it is also reasonable to assume that the same characteristics would apply to tweets. However, at the end it turned out removing retweets had a better effect on the performance of the classifiers.

In order to deal with foreign languages in the training set, translation was attempted using googletrans, but it appears Google allows for a limited number of requests within an hour, which means translating the tweets would require too much time and thus this approach was left out. The non-English tweets in the training set can provide useful features for predicting the label for non-English tweets in the testing data since almost all of the languages in the testing data are covered in the training data. Additionally, since some events are solely covered in one non-English language as mentioned in Section 1.2, those languages would have to be included in the training process as the classification of such events in the testing data would depend on the algorithm having some flexibility in terms of language.

As mentioned earlier, the actual multimedia is represented within the tweet text as a link. Such links would not give much value to the classification algorithm and thus all links were removed from the tweet text.

Another popular thing to consider within tweets is emojis. Removing them would reduce symbolic noise within the data and add more quality to the textual contents of the tweet as emojis themselves don't give too much information.

In order to reduce meaningless symbols, further cleaning was done by removing 'amp;' and '\n' within the text as they appeared in quite a few tweets as identified in data characterization.

The removal of the leftover username mentions (@username) after removing all retweets was also implemented since they don't convey much information and could be used by both fake and real posts.

After these mentioned steps, a quick whitespace correction was performed before proceeding since removing symbols and contents within the tweet can leave behind excessive whitespace.

For the final two steps of preprocessing, two separate columns were added to the DataFrame in order to make it easier to compare the effect of those two steps on classifier performance. The first of the two steps was removing stopwords. The set of stopwords was also extended with punctuation before removal. The result was directed into a new column with the title 'filteredTweet'. The data within 'filteredTweet' was then lemmatised and directed to 'lemmatisedTweet'.

Having analysed the dataset, some preprocessing steps seemed not necessary. One of those being spelling correction as by the looks of it, incorrect spelling pointed towards the post being fake. Thus the original spelling was kept such that the algorithm could have better prediction. Moreover, POS and NER tagging was not attempted due to not being able to implement translation of non-English tweets and the complexity of applying such tagging accurately on foreign languages.

2.2 Training Models

In order to find the best vectorisation approach and training model, various different combinations were considered and tested with the application of the preprocessing steps discussed

in the previous section.

Multinomial Naive Bayes is often used for text categorisation problems like this due to its computational efficiency and relatively good performance (Frank and Bouckaert 2006). Multinomial Naive Bayes is a unigram language model with integer word counts. However, there is another popular Naive Bayes model for text classification - a multi-variate Bernoulli Naive Bayes with no dependencies between words and binary word features. (Mccallum and Nigam 2001)

The Bernoulli model does perform well with small vocabulary sizes, but Multinomial Naive Bayes usually performs even better at larger vocabulary sizes. It provides on average a 27% decrease in error over Bernoulli at any vocabulary size. (Mccallum and Nigam 2001)

Both Naive Bayes classifiers were tested, but a higher F1 score was therefore already to be expected from Multinomial Naive Bayes.

Additionally, a Passive Aggressive Classifier was tested due to claims of it being ideal for classifying massive streams of data. Such as data from Twitter, which is precisely the case with this problem at hand. However, Passive Aggressive does have a downside - it does not provide global guarantees like the SVM (Support-Vector Machine). (Lavrenko 2014)

Thus, in order to compare Naive Bayes and Passive Aggressive to a classifier that fits a linear SVM, an SGD (Stochastic Gradient Descent) classifier was tried out, which does precisely that by default while using SGD learning.

2.3 Feature Processing

For feature extraction, three different vectorisation approaches were tried out and analysed. The first approach was a simple Bag-of-Words method, where the occurrence of words was counted. In combination with Passive Aggressive Classifier and Stochastic Gradient Descent Classifier, this form of feature extraction provided good results.

One potential downside of the Bag-of-Words extraction is that it forms a collection of unigrams. Therefore, it is disregarding any word order dependence. As an alternative approach, N-Grams vectorisation was also tested with n = 2, where occurrences of pairs of consecutive words were counted. (Scikit-learn.org 2020a) However, better results weren't expected due to the nature of the tweets as most of them consist of hashtags, which were assumed to be better analysed as standalone features rather than considered pairwise.

Finally, in attempt to improve the Bag-of-Words vectorisation, TF-IDF (Term Frequency–Inverse Document Frequency) vectoriser was implemented. The TF-IDF vectoriser is equivalent to combining CountVectoriser (Bag-of-Words) with TF-IDF transformer. The goal of using TF-IDF instead of the raw word counts in a given text is to scale down the impact of words that occur very frequently and that are hence empirically less informative than features that occur in a small fraction of the training data (Scikit-learn.org 2020b). The best tested max_df argument value, which ignores words that have a frequency strictly higher than the given value, was 0.2.

3 Evaluation

The tested algorithms were evaluated based on their confusion matrix and the F1 score calculated from the confusion matrix values. In this context, a True Positive is a correctly classified 'fake' tweet since the task is to identify fake posts. Therefore, a False Positive is a 'real' tweet that has been classified 'fake', a True Negative is a correctly classified 'real' and a False Negative is a 'fake' that has been classified 'real'.

The F1 score for each classifier was determined following the formula:

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{1}$$

Where precision and recall are calculated as follows:

$$precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{2}$$

$$recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{3}$$

As mentioned in the previous section, we evaluate the Multinomial Naive Bayes, Bernoulli Naive Bayes, Passive Aggressive, and Stochastic Gradient Descent classifiers in combination with three feature extraction methods - Bag-of-Words, N-Grams and TF-IDF.

3.1 First iteration - Bag-of-Words vectorisation

The first iteration was a classic Bag-of-Words feature extraction with the application of all the preprocessing steps discussed.

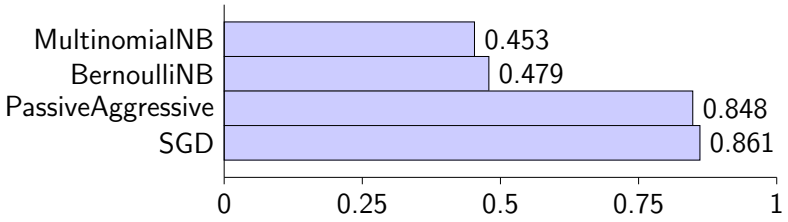


Figure 3: Classifier F1 scores with Bag-of-Words feature extraction

The performance of the Naive Bayes models in this setting was poor, whereas Passive Aggressive and SGD provided good F1 scores.

3.2 Second iteration - N-Grams with n = 2

Even though the hypothesis was that N-Grams wouldn't help much with tweets that majorly consist of hashtags, the approach was still tested out to confirm that assumption.

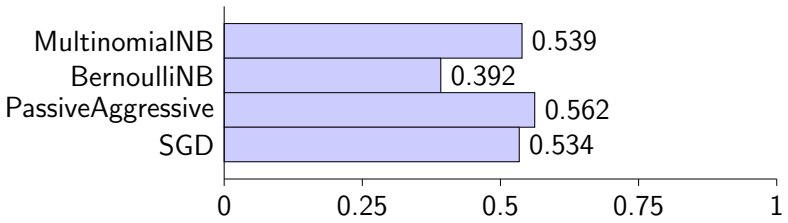


Figure 4: Classifier F1 scores with TF-IDF feature extraction

As can be seen from the results of this iteration (see Figure ?), N-Grams with n = 2 performed very poorly compared to the previous iteration. The only classifier that saw an increase in the F1 score was Multinomial Naive Bayes. Thus, that approach was not worth pursuing much further.

3.3 Third iteration - TF-IDF vectorisation

The third attempt of making feature extraction better was to use TF-IDF vectorisation, which extends Bag-of-Words with the TF-IDF transformer.

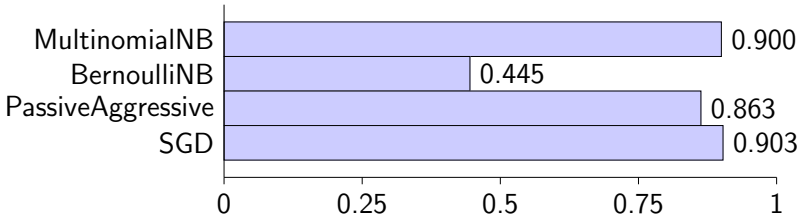


Figure 5: Classifier F1 scores with TF-IDF feature extraction

This iteration introduced significant improvement in the performance of Multinomial Naive Bayes as well as slight improvement for Passive Aggressive and SGD compared to iteration 1 (see Figure 3, 5), which in addition now made TF-IDF feature extraction with SGD the best-performing classifier. However, the F1 score of SGD does fluctuate between values such as 0.894, 0.897, 0.903, etc. Whereas the score of MultinomialNB remains stable at 0.900.

That was the case for all iterations: the F1 scores achieved by Multinomial and Bernoulli Naive Bayes stayed static over multiple runs of the same combination, whereas the scores reported by Passive Aggressive and SGD varied a bit every time. Overall, Passive Aggressive and SGD performed very similarly to each other for every iteration and Bernoulli Naive Bayes was the worst-performing of the four in all of them. Multinomial Naive Bayes was the only classifier of the four that was so drastically affected by introducing TF-IDF transformation.

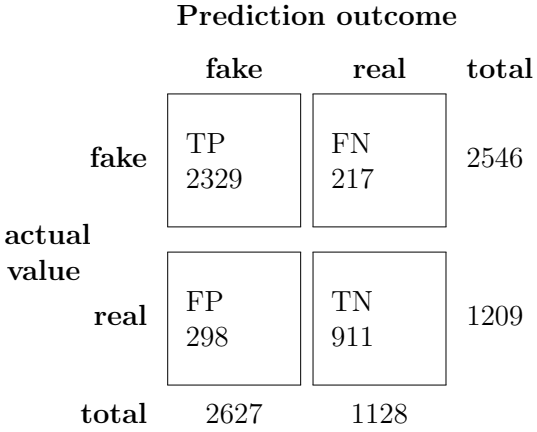


Figure 6: Confusion matrix of the MultinomialNB classifier with TF-IDF vectorisation

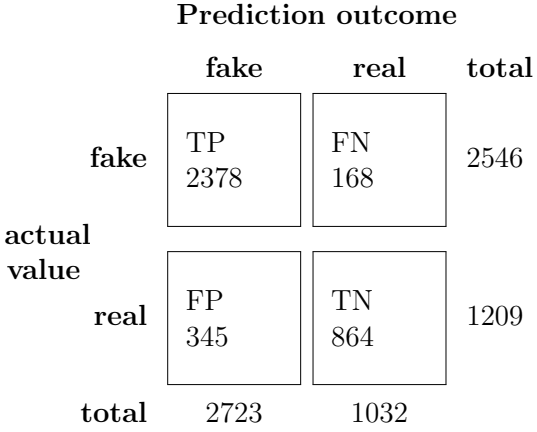


Figure 7: Confusion matrix of the SGD classifier with TF-IDF vectorisation

As can be seen from the confusion matrices (see Figure 6, 7), both algorithms tend to predict more of the posts as fake than there actually exists and SGD does so even more than Multinomial Naive Bayes. Therefore, they miss some of the real posts in the data.

We can safely say that the differences between the performances of the classifiers are not random chance since they were all trained on the same set of training data and tested on the same set of testing data. Moreover, the training data for all of them went through the same steps of preprocessing and each algorithm combination was run several times.

4 Conclusion

The task at hand was to identify fake posts, which was achieved with an F1 score of 0.9 using Multinomial Naive Bayes classification with Term Frequency–Inverse Document Frequency feature vectorisation. An even higher F1 of 0.903 was achieved with Stochastic Gradient

Descent classification using the same type of vectoriser. However, due to the slight variance in the F1 score of the SGD classifier over multiple runs, with values also falling under the performance of Multinomial Naive Bayes, we can consider MNB the better approach due to its stable score.

However, since SVM-based classifiers are expected to have higher performance than Naive Bayes in literature and we obtained proof that SGD can outperform Multinomial Naive Bayes, with further data processing it is possible the F1 score of SGD can be pushed up even more.

One of possible additions could be translating the non-English tweets, which was attempted, but given up on due to limitations on translation requests set by Google. One workaround to save time could be to use a combination of multiple translators, but that would then create inconsistency in the quality of the translation.

Additionally, after successful translation, it would be interesting to investitgate the effect an approach that includes POS and NER tagging as NER tags can give even more information about the text and its underlying meaning.

Further analysis of the algorithm performance showed that the two best-performing models overpredict posts as fake and since classifying them as fake is the more important task, then it is a better outcome than overpredicting them as real which could lead to news professionals taking fake posts into account as real ones. Moreover, an F1 score above 0.9 can definitely be considered a success.

References

Frank, Eibe and Remco R. Bouckaert (2006). "Naive Bayes for Text Classification with Unbalanced Classes". In: *Lecture Notes in Computer Science*, pp. 503–510.

Lavrenko, Victor (2014). *Text Classification 3: Passive Aggressive Algorithm.* URL: `https://www.youtube.com/watch?v=TJU8NfDdqNQ` (visited on 01/07/2020).

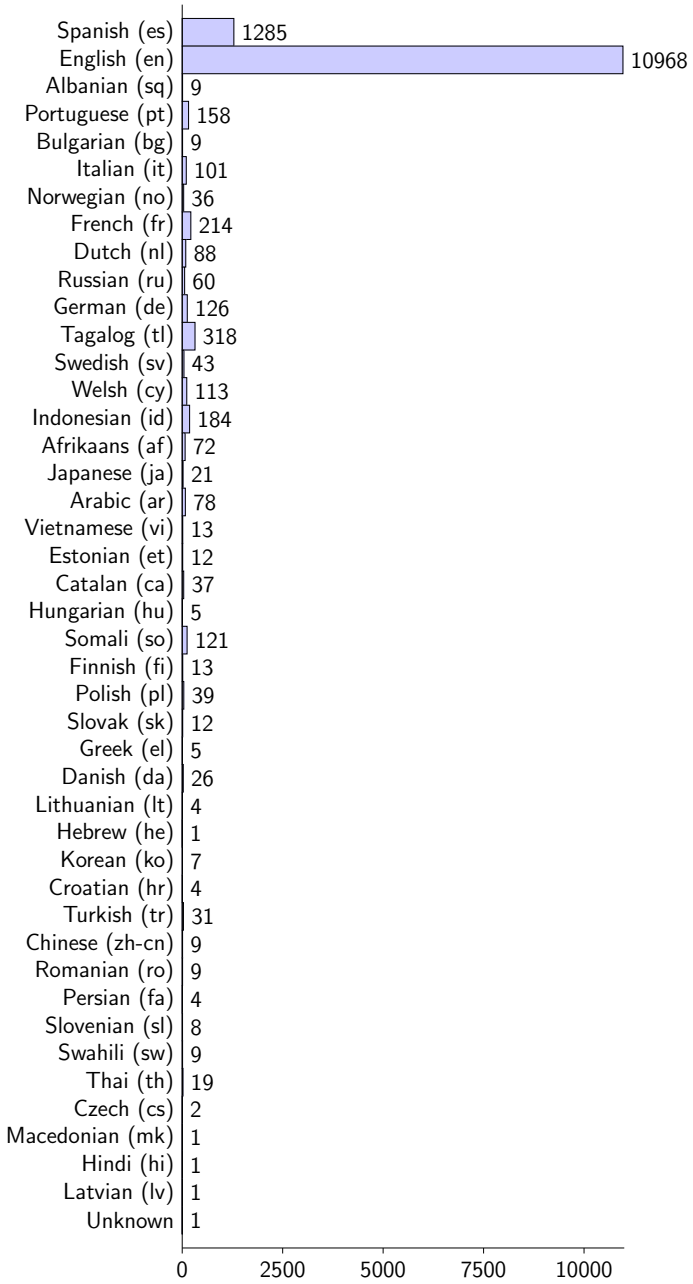
Mccallum, Andrew and Kamal Nigam (2001). "A Comparison of Event Models for Naive Bayes Text Classification". In: *Work Learn Text Categ* 752.

Scikit-learn.org (2020a). *6.2. Feature Extraction — scikit-learn 0.22.1 documentation.* URL: `https://scikit-learn.org/stable/modules/feature_extraction.html` (visited on 01/07/2020).

— (2020b). $sklearn.feature_extraction.text.TfidfTransformer|scikit-learn0.22.1documentation.$ URL: `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#sklearn.feature_extraction.text.TfidfTransformer` (visited on 01/07/2020).

Appendices

A Number of posts per language in training data



B Number of posts per language in testing data