COMP3222 Coursework MediaEval-2015 Report

William Mayhew

1 Introduction

The project centers on the critical task of classifying tweets based on their content to determine their authenticity. The primary objective is to create and implement two distinct algorithms capable of accurately categorising tweets as 'real' or 'fake'. Combining precision and recall, F1 scores will be used as the benchmark, aiming for a high accuracy to validate the efficacy of our machine learning models.

We are provided with two datasets, namely *mediaeval-2015-trainingset.txt* and *mediaeval-2015-testset.txt*, which encompass tweet-specific fields tweetId, tweetText, userId, imageId(s), timestamp, and label (classifier).

2 Data Analysis

Under examination, the dataset comprises 14,277 entries. Primarily focusing on the 'tweetText' field, a wide array of content encompassing various languages, hashtags, URLs, user mentions, emojis, and special characters, each with differing frequencies are apparent.

Figure 1 graphs the entries. Evidently, a disparity exists between 'fake' (including humour) and 'real' tweets, with 9,356 instances marked as fake and 4,921 as real. This imbalanced distribution could potentially introduce bias into our models, favoring the majority class.

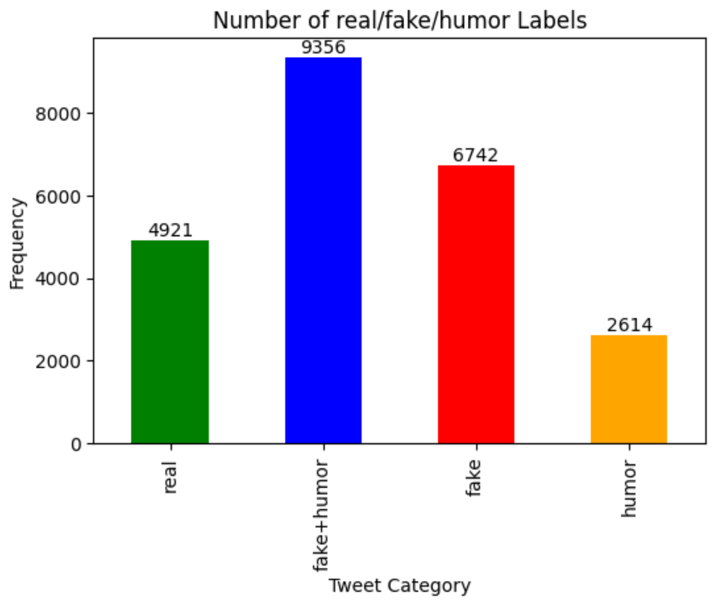

Figure 1: Label Frequency

To identify the predominant languages in the dataset, the langdetect library was utilised. Whilst inconsistent, it revealed the top three prevalent languages as English (en) at 76.8% , Spanish (es) at 9.1% , Tagalog (tl) at 2.2% as seen in Figure 2. The other 40 languages were categorised as 'Other'. These include those which were not as significant, as well as those which were unidentifiable.

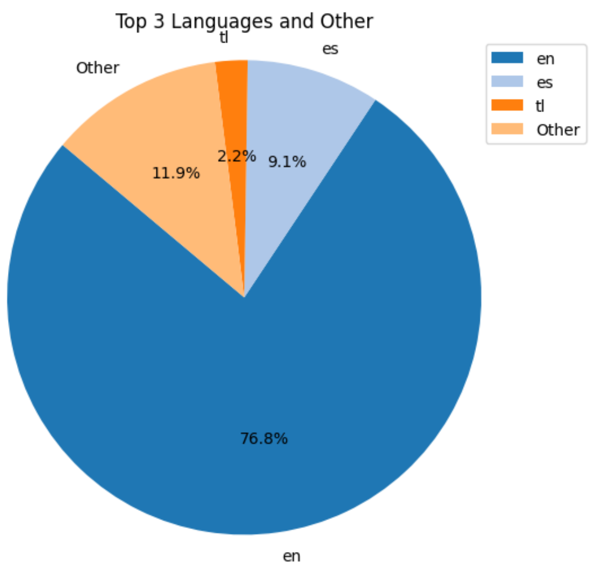

Figure 2: Language Frequency Pie Chart

Delving into the presence of hashtags, URLs, user mentions, emojis, and special characters, frequency analysis was conducted to discover any correlation between real and fake tweets. Figure 3 and Table 1 depict the frequency of these features on a per tweet basis. These highlight a notable distinction in the usage frequency of user mentions and emojis, with real tweets more inclined towards user mentions and fake tweets leaning towards emojis.

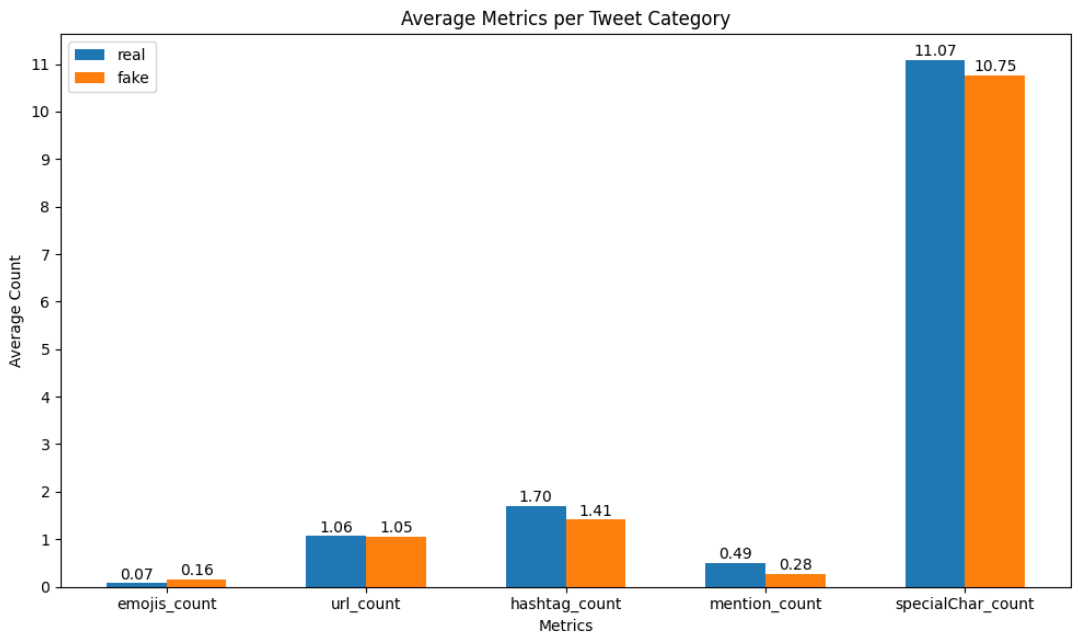

Figure 3: Frequency per tweet for categories in Real and Fake tweets

Category	Frequency per Tweet		Percentage Difference
	Real Tweets	Fake Tweets	
Emojis	0.07	0.16	78.26%
URLs	1.06	1.05	0.95%
Hashtags	1.7	1.41	18.65%
Mentions	0.49	0.28	54.55%
Special Characters	11.07	10.75	2.93%

Table 1: Frequency per Tweet for Categories in Real and Fake tweets

As seen in Figure 4, tweet character lengths revealed 135 instances of tweets surpassing the 140-character limit set in 2015. Manual inspection revealed formatting irregularities in some cases. For example, tweetId '262987209144160000' contained additional tweet IDs contributing to inflated character counts. This meant that some tweets had excessively large character counts up to the 7000s. Additionally, tweetId '263143193200173000' also breached the limit due to character encoding issues. Characters such as '>' were being encoded as their HTML encoding to '&gt;'.

Considering these anomalies, tweets exceeding 160 characters will be excluded from analysis due to inconsistencies. 160 was chosen as the upper boundary because between 140 and 160 characters, 117 tweets were found, with the 118th appearing at 190 characters. This enables flexibility and the inclusion of tweets with slight character encoding issues.

Figure 5 shows the character count distribution between real and fake tweets. No distinct correlation between the classifications and character count emerged.

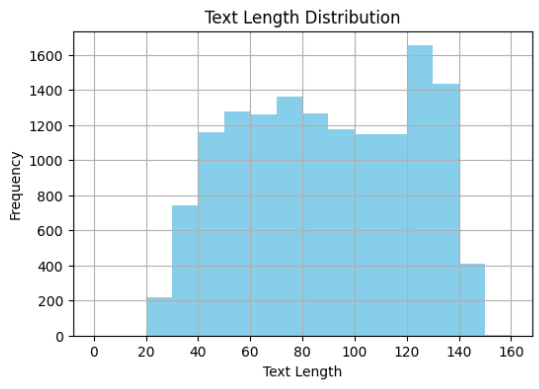

Figure 4: Text Length Distribution

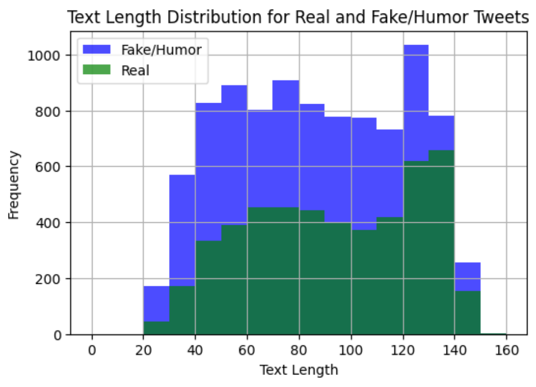

Figure 5: Text Length Distribution Real and Fake

3 Similar Problems

Several similar issues have been tackled in the past, showcasing diverse methodologies and approaches. The strategies adopted vary from incorporating both user-centric and content-based features [1], [2] to focusing solely on content-based features [3].

Nikam and Dalvi [1] addressed the identification of fake news articles by utlising the article body and publisher's name. Their methodology integrated Term Frequency - Inverse

Document Frequency (TF-IDF) alongside Naive Bayes (NB) and Passive Aggressive classifiers. Preprocessing involved text cleaning steps like stop words removal, tokenization, and text stemming.

Krishnan and Chen [2] took a more expansive approach by incorporating an array of user-based features such as friend count, tweet count, and verification, alongside in-depth content-based features like URL credibility scores and sentiment scores. They experimented using J48 Decision Trees and Support Vector Machines (SVM), emphasising SVM as the baseline classifier for binary classification.

Ahmed et al [3] conducted experiments aimed at detecting fake reviews and news. Their methodology encompassed a wide spectrum of classifiers, including Stochastic Gradient Descent (SGD), SVM, Linear SVM (LSVM), K-Nearest Neighbor (KNN), Logistic Regression (LR), and Decision Trees (DT). Employing an N-gram model to represent the data, they found that unigrams and bigrams outperformed trigrams and quadgrams. Their findings highlighted the superiority of linear-based classifiers (LSVM, SGD, and LR) over non-linear ones, despite some non-linear classifiers achieving high accuracies. Their preprocessing steps involved a thorough data cleaning process encompassing stop words removal, tokenization, stemming, text normalisation to lowercase, sentence segmentation, and punctuation elimination.

4 Selecting an algorithm

Choosing the right machine learning algorithm requires a thoughtful consideration of their respective strengths and weaknesses. In this section, the suitability of Naive Bayes (NB), Support Vector Machine (SVM), and Decision Trees (DT) will be explored.

Naive Bayes (NB) functions on probabilities and statistics. It is known for its swift learning capabilities and doesn't demand a massive dataset to perform well [4], [5]. However, its vulnerability to irrelevant attributes might necessitate robust preprocessing, potentially leading to subpar estimations [5].

Linear Support Vector Machine (LSVM) stands out as a highly accurate and advanced learning method, especially proficient in classification problems [4], [5]. Yet, its slower learning pace with larger datasets and reduced accuracy with imbalanced datasets [6] might limit its suitability, considering our dataset's bias towards fake and humor labels.

Decision Trees (DT) represent a simple, logic-based algorithm [5], known for their speed and ability to handle large datasets with high accuracy [4], [7]. Their flexibility with data preprocessing, along with their tolerance for irrelevant attributes, makes them an appealing choice [4], [5].

Considering the prior insights, the focus will be on deploying Linear Support Vector Machines (LSVM) and Decision Trees (DT) due to their proven efficacy and accuracy in similar problem domains. Leveraging N-grams and TF-IDF for feature selection, coupled with preprocessing techniques such as stop words removal, tokenization, and stemming, aligns with the successful methodologies previously employed.

5 Pipeline Design

The outlined steps for the algorithms are as follows:

1. Preprocessing
2. Feature extraction
3. Classification

Initially, the datasets underwent initial data cleansing steps:

- Entries labeled as 'humor' were relabeled as 'fake'
- 'Bad' tweets exceeding 160 characters were removed
- A reformatting process for tweet texts was executed

Similar to the methodologies in [1], [3], our preprocessing included:

- **Lowercasing**
- **Removal of emojis, URLs, Mentions, and Special Characters** using regular expressions from the 're' module.
- **Stop Words Removal** utilising the 'NLTK' library, which excludes common English language words.
- **Stemming** using the Porter Stemmer class from the 'NLTK' library to reduce words to their root form.
- **Lemmatization** employing WordNet's Lemmatizer from the 'NLTK' library to convert words to their base or dictionary form.

5.1 Algorithm 1 - Random Forest Classifier

In this algorithm, TF-IDF with N-grams alongside the Random Forest Classifier (RFC) has been applied. Utilising TF-IDF allowed us to weigh the importance of terms within documents considering the context provided by both unigrams and bigrams.

All aforementioned preprocessing techniques were applied.

GridSearchCV hyperparameter tuning revealed the optimal 'random_state' value as 45 for the Random Forest Model. Other parameters would've also been thoroughly tested, but the processing time outweighed the potential benefits as each computation was taking well over a minute each. Overall, this parameter yielded the best performance based on the F1 score metric of 0.907.

5.2 Algorithm 2 - Linear Support Vector Classifier

This algorithm Utilised Count Vectorizer with only unigrams, coupled with the Support Vector Classifier using the linear kernel.

Not all preprocessing techniques were applied as it was observed that the accuracy improved without word stemming.

GridSearchCV hyperparameter tuning revealed the optimal 'C' value for the SVM model as 0.17. However, this lowered the calculated F1 score from 0.875 to 0.818. This new 'C' value was tested with cross-validation, providing better generalisation.

As mentioned in the previous literature, SVM typically decreases in accuracy when the datasets are imbalanced. Balancing the datasets with class weights further improved the F1 score from 0.818 to 0.832.

5.3 N-grams and Hyperparameter Tuning

The F1 scores varied based on the algorithm, feature extraction method, N-gram size, and hyperparameters.

As seen in Table 2, unigrams and bigrams outperformed trigrams and quadgrams, aligning with similar findings in Ahmed et al.'s experiments.

The adjustments to the classifiers' parameters aimed at improving generalisation, particularly in LSVM, even though the new F1 score was slightly lower. These new parameters provide better generalisation by preventing overfitting through the use of cross-validation. Hyperparameter tuning resulted in an accuracy of 0.907 for RFC and 0.818 for LSVM.

	Unigrams	**Bigrams**	**Unigrams & Bigrams**	**Trigrams & Quadgrams**
RFC with TF-IFD	0.761	0.805	0.9	0.808
RFC with Count	0.772	0.805	0.796	0.808
RFC with TF-IFD 'random_state = 45'			0.907	
LSVM with TF-IFD	0.813	0.795	0.824	0.808
LSVM with Count	0.875	0.807	0.724	0.808
LSVM with Count 'C = 0.17'	0.818			

Table 2: F1 Scores with different N-grams

5.4 Data Balancing

Initially, an imbalance in the labels of the training data was evident, comprising of 4,921 'real tweets (34.47%) and 9,356 'fake/humor' tweets (65.53%). To mitigate potential bias and enhance training accuracy, weighting strategies were implemented for both algorithms. However, this adjustment led to a reduction in the accuracy of our Random Forest implementation. On the other hand, the Linear Support Vector Machine showed a slight improvement. Reduced accuracy with imbalanced datasets were indicated, aligning with the findings of [6]. The impact of balancing the data is summarized in Table 3.

	Unbalanced	**Balanced**
RFC with TF-IFD	0.907	0.881
LSVM with Count	0.818	0.832

Table 3: Balanced vs Unbalanced data

6 Evaluation

Considering dimensionality reduction techniques, Principal Component Analysis (PCA) emerged as a potential approach to reduce dataset dimensionality while preserving most of its variance. Unfortunately, hardware limitations hindered the implementation of PCA. The computation requirements and memory limitations posed significant constraints, making it infeasible within the current setup.

In the pursuit of further model optimisation, hyperparameter tuning using GridSearchCV was utilised. This approach facilitated an exhaustive search across a parameter set, employing cross-validation to evaluate model performance. However, the default parameters often delivered satisfactory results. Further iterations of tuning required extensive efforts without guaranteeing substantial benefits, potentially outweighing the benefits.

Overall, two highly accurate machine learning classification models have been developed. As can be seen in the confusion matrices, Algorithm 1 has higher true positives at 2468 compared to Algorithm 2 with 2057 and lower false positives at 78 compared to 489. This indicates a better ability to correctly identify true positives with fewer false alarms. Algorithm 1 also has a higher precision compared to algorithm 2. This implies that algorithm 1's positive predictions are more accurate. However, algorithm 2 has slightly higher recall indicating that it captures a slightly higher proportion of actual positives.

Algorithm 1: Figure 6

- Utilized `TfidfVectorizer(ngram_range=(1,2))` with `RandomForestClassifier(random_state=45)` to achieve an F1 score of 0.907.

Algorithm 2: Figure 7

- Employed `CountVectorizer()` with `SVC(kernel='linear', C=0.17, class_weight=`*`class_weights`*`)` to achieve an F1 score of 0.832.
  - *`class_weights`* `= {'fake': total_samples / (2 * total_fakes), 'real': total_samples / (2 * total_reals)}`

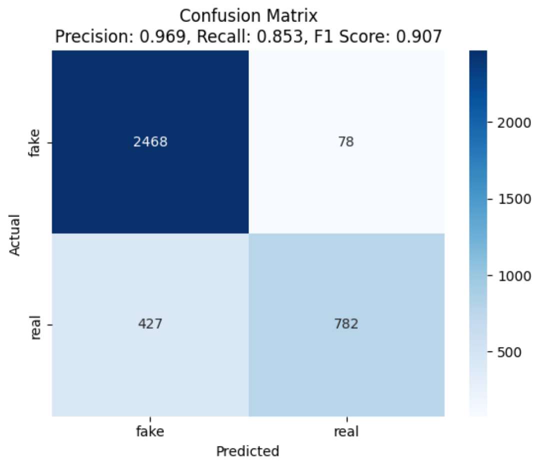

Figure 6: Algorithm 1 Confusion Matrix

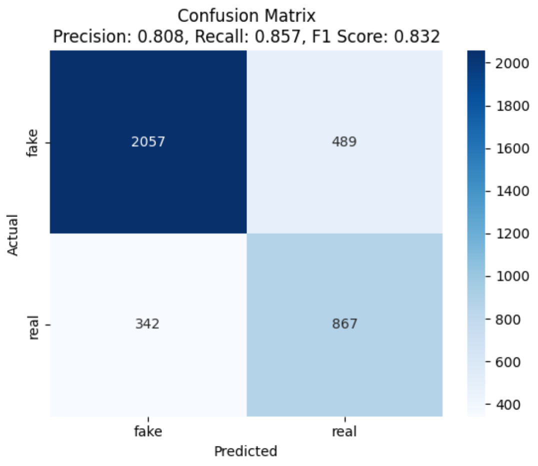

Figure 7: Algorithm 2 Confusion Matrix

7 Conclusion

The tweet classification process involved data analysis, preprocessing, feature extraction, and algorithm selection. Initial analysis revealed a notable imbalance between 'fake/humor' and 'real' tweets. Addressing this imbalance involved exploring class weights to mitigate bias and address potential underfitting. Additionally, the textual features of tweets provided insights into any relations associated with tweet authenticity.

Further exploration was prevented by computational limitations - notably, memory constraints - when considering dimensionality reduction techniques like PCA.

Evaluation of machine learning algorithms, particularly Random Forest and Linear Support Vector Machines, showcased the effects of hyperparameter tuning in optimizing model performance. Despite the default values from the Scikit library yielding satisfactory results, the use of GridSearchCV offered marginal improvements.

Future iterations may benefit from incorporating additional features like tweet locations and leveraging image IDs. Enhanced preprocessing, such as translating foreign tweets into English, could ensure a more uniform training set. Despite limitations, these models achieved success and accuracy in predictions.

References

[1] S. S. Nikam and R. Dalvi, “Machine learning algorithm based model for classification of fake news on twitter,” in *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India: IEEE, Oct. 7, 2020, pp. 1–4, ISBN: 978-1-72815-464-0. DOI: 10.1109/I-SMAC49090.2020.9243385. [Online]. Available: https://ieeexplore.ieee.org/document/9243385/ (visited on 12/23/2023).

[2] S. Krishnan and M. Chen, “Identifying tweets with fake news,” pp. 460–464, Jul. 2018. DOI: 10.1109/IRI.2018.00073. [Online]. Available: https://ieeexplore.ieee.org/document/8424744/ (visited on 12/24/2023).

[3] H. Ahmed, I. Traore, and S. Saad, “Detecting opinion spams and fake news using text classification,” *SECURITY AND PRIVACY*, vol. 1, 2018, ISSN: 2475-6725. DOI: 10.1002/spy2.9. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spy2.9 (visited on 12/23/2023).

[4] S. B. Kotsiantis, “Supervised machine learning: A review of classification techniques,” (visited on 12/24/2023).

[5] P. C. Sen, M. Hajra, and M. Ghosh, “Supervised classification algorithms in machine learning: A survey and review,” Advances in Intelligent Systems and Computing, J. K. Mandal and D. Bhattacharya, Eds., pp. 99–111, 2020. DOI: 10.1007/978-981-13-7403-6_11. (visited on 12/24/2023).

[6] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, “A comprehensive survey on support vector machine classification: Applications, challenges and trends,” *Neurocomputing*, vol. 408, pp. 189–215, Sep. 30, 2020, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2019.10.118. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231220307153 (visited on 12/24/2023).

[7] “Comparative study of k-NN, naive bayes and decision tree classification techniques,” *International Journal of Science and Research (IJSR)*, vol. 5, no. 1, pp. 1842–1845, Jan. 5, 2016, ISSN: 23197064. DOI: 10.21275/v5i1.NOV153131. [Online]. Available: https://www.ijsr.net/archive/v5i1/NOV153131.pdf (visited on 12/24/2023).