Applied Data Mining Coursework 2

Dimitrios Priovolos, 36086817
School of Computing and Communications
MSc Data Science

***Abstract*—This report studies how easy it is to predict an article's category from its text, using different classification models. More than thirty thousand articles from CNN are used to create three different models that can predict the category of an article. The data set used in the training and testing processes is the CNN News Articles from 2011 to 2022 data set, which contains many features for CNN articles, like the Author, the URL and the actual text of the article. The results show that with the right pre-processing of the data, it is possible to predict the category of an article with high accuracy. The model with the highest accuracy is the Support Vector Classification (SVC), though the other two models performed very well too. The overall performance will be judged by other metrics as well, like precision, recall and more.**

I. Introduction

Natural Language Processing (NLP) is a well-known domain in the Machine Learning field. It assists the computer with comprehending and interpreting human natural language variances, such as text and speech. One of its most popular applications is the classification of a text in a category that summarises the content of the text. Text classification is used to organise many different documents in a set of classes, that share commonalities on various dimensions such as language, field, and so forth. This is important because records of value can be separated from records of little or no value. Nowadays some of its most known uses are spam detection in emails, or search engines content management.[1] Text classification is also used in research papers, the reason being to capture, interpret, and discuss changes in research practices.[2] It can also be used to define the field of activity of authors, which is useful for journal publishers who are responsible for categorising their journals and presenting them on their websites in a compact manner.[3] All these applications, make the ability to accurately predict the category of a text very important.

In this report, the data is pre-processed so it is 'cleaned' for the model training, using different methods like Tokenization, Lemmatization and stemming, which are taken directly from the NLTK library. After that, three different popular models for text classification are trained and used to predict the category of many articles, in order to compare their performance. The data set used in this report is the CNN News Articles from 2011 to 2022 data set[4], which consists of approximately thirty-seven thousand articles. For each article, the data set contains eleven features, with the most useful one and the one used in model training being the article text.

The first classifier that was implemented, is the Logistic Regression classifier. Logistic Regression (LR) is one of the most important statistical and data mining techniques employed by statisticians and researchers for the analysis and classification of both binary and multi-class classification problems.[5] Logistic Regression is generally perceived as an effective model and a great starter algorithm for text data classification. This model performed very well in categorising the CNN articles. While its accuracy is almost the same as the LSVC model, it surpasses the latter in almost every other aspect as seen in the results section.

The second classifier that was implemented, is the Linear Support Vector Classification (LSVC) classifier. In multi-class classification, LSVC implements the "one vs the rest" multi-class strategy. That means that the LSVC trains the same number of models as the number of different classes, contrary to the normal SVC, which is explained next. This model is preferred when dealing with a big amount of samples because it provides more flexibility in the choice of the loss functions and the penalties.[6] This model performed the worst out of the three, but it still had good results, as shown in the results.

The third classifier that was implemented, is the normal Support Vector Classification (SVC) classifier. SVCs do not require parameter tweaking, since they can automatically identify optimal parameter settings.[7] Contrary to the linear form of this classifier, in multi-class classification, SVC implements the 'one versus one' approach. This means that $nclasses * (nclasses - 1)/2$ classifiers are constructed and each one trains data from two classes.[6] Like LSVC, this model needs the data to be transformed into a vector. So after 'cleaning', the data is transformed to the Tfidf vector, which will be analysed in the Methods section. This model performed the best out of the three but had a big drawback, which will be analysed in the Results section.

II. Related Work

With the number of complicated documents and texts increasing exponentially in recent years, the need for a deeper understanding of machine learning technologies and methods has increased. This has motivated many researchers to try to build new machine learning algorithms, that can produce high-quality results.

Pre-processing the data is one of the most important steps in text classification. Therefore, many researchers have studied its impact on text classification in terms of various aspects, like accuracy, dimension reduction etc. One of the most in-depth researches on the influence of pre-processing tasks on text classification is done by Alper Kursat Uysal and Serkan Gunal[11]. They investigate the impact of many methods, like tokenization, stop-word removal and lowercase conversion on

different text domains and with many different combinations between them.

The process of text classification is a very complicated process involving lots of different steps. Those steps have been enumerated and studied by K. Aas and L. Eikvil[8]. They describe the steps of the process, present different feature selection methods and finally summarise and compare some of the most known binary and multi-class machine learning classifiers.[10]

In 2012, Aggarwal and Zhai[9], greatly influenced by the previously mentioned research, provided more examples and in-depth information about the steps of the classification process. They introduce some new feature selection methods, like gini index and information gain, and they describe some text classification algorithms, like decision trees, naive Bayes, neural networks and more.

Lastly, deep learning models have emerged in the last years. They are used for sentiment analysis, document categorization and more. There are many deep learning models for text classification, many of which are presented in the Shervin Minaee et al[12]. This research includes more than a hundred fifty deep learning models, using more than sixty popular data sets for text classification and finally provides an analysis of the performance of the most popular models.

III. Methodology

A. Data Sourcing

The data used in this experiment is the CNN News Articles from 2011 to 2022 data set, and Google Colab was used as the platform for the algorithm creation. To import the data in Colab, the data was uploaded to Google Drive and using the drive library from Google Colab, the drive was mounted to the Colab file. After that, the pandas library was used to load the data from the drive.

B. Data Pre-Processing

Pre-processing is a vital part of the text classification process. The general steps that are taken are presented in Fig.1.

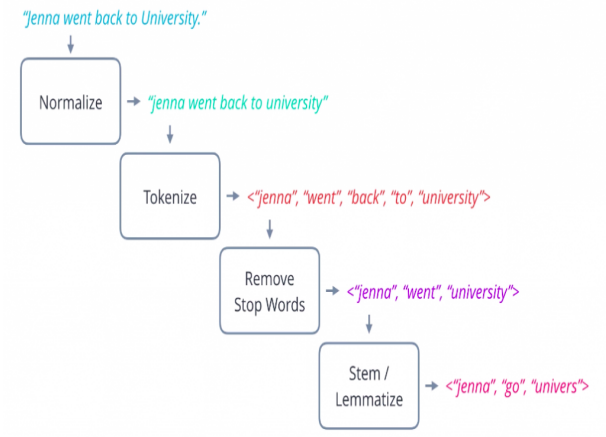

Fig. 1. Text Pre-Processing Steps

During the pre-processing, the first thing done is to change the data types of all the features, from object, which is the way they were read using the pandas library, to each feature's respective data type.

The next step is to search the data for NULL values. This is easy to do by using the pandas library again and seeing the info of all the features. In this case, there were NULL values detected, so they had to be removed. To do that, all rows that had one or more features with the NULL value were dropped from the data set. The next issue is the existence of duplicate articles, which was resolved by simply dropping all the duplicate articles and keeping only the first occurrence of each article.

The text that is used for the model training, might contain the article's category as a string. This exposes the model to the issue of data leakage, which is a common problem in machine learning when developing predictive models. Data leakage is the issue when information from outside the training data set is utilised to generate the model. This new data might allow the model to learn or know something it wouldn't have known otherwise, invalidating the predicted performance of the model being built.[13] The 'Author' and 'Keywords' features are the two features most likely to leak this information. To solve that issue, a function that removes all the occurrences of the category names from those features. For that to happen, each sample is tokenized, with each of the resulting tokens compared to the flagged words. Finally, the function removes all non-letter characters, and joins all tokens, split by ','. This function is called for all the samples of the data and replaces the original data with the new one.

The next thing that should be done is the 'cleaning' of the text for all the relevant features. The 'cleaning' process starts with the lower-casing of the text, which helps in parsing, at the later stages of the NLP application. After that, the text is tokenized and using the num2words library, all the numbers are changed to words. They are not dropped here, because they might prove a useful feature for information extraction. Next, all the special characters are removed, so only characters [a-z] remain in the text. Using the nltk library, a set of stopwords for the English language is created. Then, a new text that is composed of the original text minus all the words contained in the stopwords list is made and replaces the original. After that, the text is tokenized once more, just in case something changed during the previous process, and undergoes lemmatization. Lemmatization returns the base or dictionary form of a word, which is known as the lemma and replaces all the words with their lemma. It is preferred over Stemming because even if it is harder to do and takes more time, it provides better results. In this case, both of them are used. So after the lemmatization of the text, it finally undergoes the stemming process and the final product is returned.

The data is clean now. But what is needed now is to check if the data is imbalanced. Imbalanced refers to the problem where the number of examples in the training data set for each class label is not balanced. To figure out if this issue applies here, the number of articles for each category is plotted in a

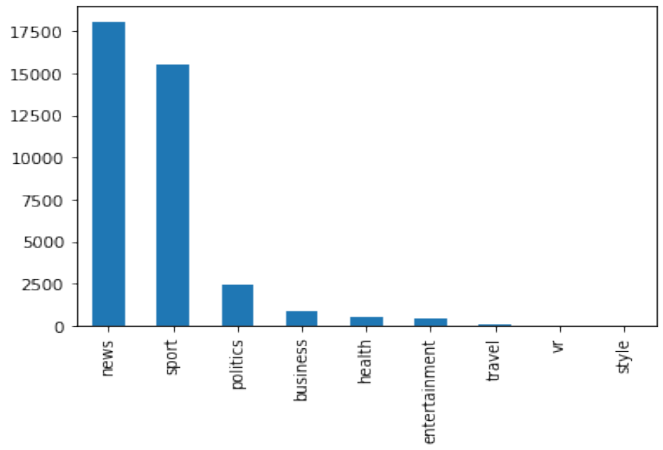

Fig. 2. Number of articles per category

bar chart, in order to make the comparison easier.

It is clear from Fig. 2, that the 'vr' and 'style' articles have close to no articles, compared to the other categories. So in order to fix the imbalance issue, the rows which had those two as a category are removed from the data set. The classification process is hindered by the big volume of features, many of which are sometimes irrelevant. By reducing the number of those features by 'cleaning' our data, the performance of the model rises. The last thing was done before the model training is the addition of two columns in the data set, which represent the 'Category' and 'Section' features as categorical values, which are made using the pandas library.

For the models to be able to receive the text data, the text needs to be transformed into a matrix of numbers, that will contain information about the given text. One very popular way of doing that is through TfidfVectorizer from the sklearn library. This method, will tokenize the text, learn its vocabulary and assign weights to words that are more interesting and frequent in a document, but not across all documents. There are many parameters for this method. Some of those that were tweaked for this experiment, are the ngram range, which was set to (1,2), which allows both unigram and bigram features to be extracted, the lowercase, which was assigned as False, because the text has already been lowercased, the max and min df, which are the top and bottom limits of how many times or in what percentage of the articles should a word be present for it to be ignored as a feature, and the max features, which was put to 350, meaning that no more than 350 words or phrases can be used as weighed features for the Tfidf Vector.

With that the pre-processing of the data is complete. There are more things that could be done, like removing extra white spaces, dropping more features, or dropping some samples from the more dominant categories to balance the data set more, but the pre-processing done lead to great results, so it was decided that it sufficed.

C. Model Training

Before the training process, the data set has to be split into the training set and the test set. It is split randomly, with 80% of the data assigned to train the models and 20% of the data used to test their performance. After that, the sklearn library was used to create the three models, using the default parameters. Finally, each model was trained using the sklearn library and tested with the test data set. The results of the experiments are shown in the next section.

IV. Experimental Results

After training all the models, it is time to test them and evaluate their performance. The models will be compared using the Precision, Recall, F1-score and accuracy metrics.

A. Logistic Regression Classifier

The classification report, which contains all the metrics for the model's performance is displayed in Table 1.

Category	Precision	Recall	F1-score
0	0.76	0.40	0.53
1	0.90	0.32	0.47
2	0.77	0.28	0.40
3	0.89	0.96	0.93
4	0.80	0.70	0.75
5	0.97	0.98	0.98

TABLE I
Classification Report Logistic Regression

The accuracy of this model is 92.27%, and its training took 11.34 seconds. In Fig. 3, the confusion matrix of the Logistic Regression classifier results is displayed, where it is easy to see which categories had more wrong predictions.

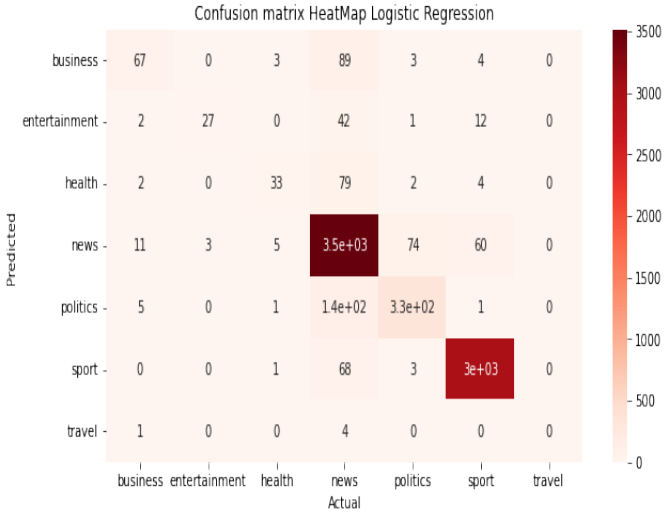

Fig. 3. Heat Map of Confusion matrix for Logistic Regression

B. Linear Support Vector Classification

Next are the results of the LSVC. The classification report can be seen in Table 2.

The accuracy of this model is 92.28%, and its training took 1.69 seconds. In Fig. 4, the confusion matrix of this model is displayed. It can be observed both here, but on the other

Category	Precision	Recall	F1-score
0	0.75	0.42	0.53
1	0.82	0.33	0.47
2	0.71	0.24	0.36
3	0.89	0.95	0.92
4	0.79	0.69	0.74
5	0.97	0.98	0.98

TABLE II
CLASSIFICATION REPORT LINEAR SUPPORT VECTOR CLASSIFICATION

confusion matrices are well, that most of the model mistake is between the category of the 'news' and other categories, which is understandable because an article can be 'news' but also can belong to another category.

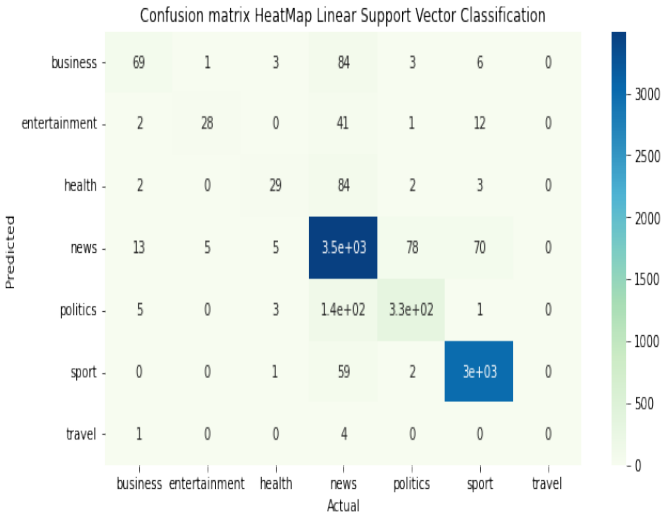

Fig. 4. Heat Map of Confusion matrix for Linear Support Vector Classification

C. Support Vector Classification

Finally, the results of the LSVC are presented. The classification report can be seen in Table 3.

Category	Precision	Recall	F1-score
0	0.88	0.46	0.60
1	0.87	0.24	0.37
2	0.85	0.29	0.43
3	0.90	0.97	0.93
4	0.86	0.73	0.79
5	0.98	0.98	0.98

TABLE III
CLASSIFICATION REPORT SUPPORT VECTOR CLASSIFICATION

The accuracy of this model is 97.07% and its training took 268.47 seconds. In Fig. 5, the confusion matrix of this model is displayed.

It is clear, that the SVC model performs better than the other two at almost everything. It has higher accuracy than the other two, and better Precision, Recall and F1-score values for the most part. That is mostly to be expected, since SVC works well with unstructured and semi-structured data like

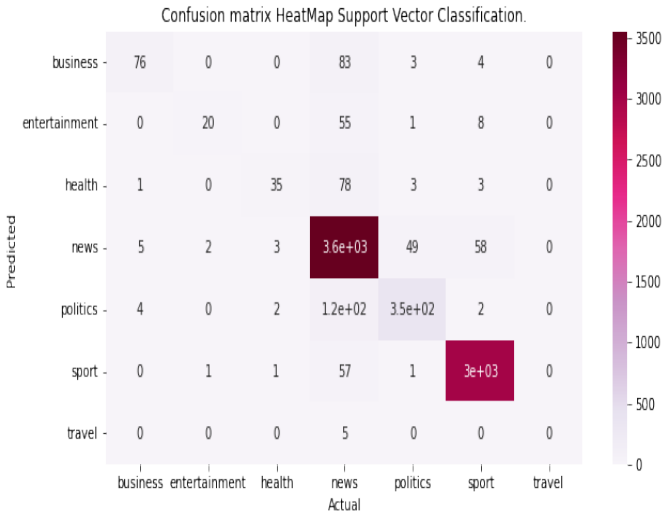

Fig. 5. Heat Map of Confusion matrix for Support Vector Classification

text and images, while logistic regression works with already identified independent variables.[14] For the LSVC and SVC, the latter performed better, most likely because the data could not easily be separated with a hyperplane by drawing a straight line, so a generic SVC is better in this case. Another possible reason why SVC performs better than the other two models is that SVC handles outliers better than linear models. While all models do a good job at classifying with high accuracy and precision, the SVC surpasses the rest.

V. CONCLUSIONS AND FUTURE WORK

In this report, the steps for an effective way of pre-processing text data are shown, and three models are trained and evaluated, using the product of the pre-processing. The models were created using the sklearn library with all their parameters set to default. The pre-processing included most of the known steps for text cleaning, like tokenization, use of stopwords and lemmatization. The whole process, from the data import to the results visualisation, took 41.6 minutes, with most of the time being taken by the data cleaning and specifically by the lemmatization process. Since the results of the presented models are more than acceptable and the run time of the algorithm is already big, there was no need to try implementing more complex text classification models, like deep neural networks. It would be interesting to split the categories with more articles ('News' and 'Sports') into sub-categories using their sections, in order to further balance the data set. It would also be interesting to try more combinations of pre-processing methods, like skipping the lemmatization process or the lowercasing process to see what effect would it have on the final results. Finally, more classification models like Naive Bayes or K-Nearest Neighbors could also be included and trained both by the TfidfVectorizer transformed text and without it to see how much does vectorizing the text using frequency weights impacts the results.

Acknowledgements

The code for each model, the data needed to reproduce the results and the already trained models are all uploaded in GoogleDrive. You can gain access to them through the public share link here: Dimitrios Priovolos CW2 Files. Some ideas and parts of code were taken from ”CNN articles text classification”.

REFERENCES

[1] Agarwal, B., Mittal, N. (2014). Text Classification Using Machine Learning Methods-A Survey. In: , et al. Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012. Advances in Intelligent Systems and Computing, vol 236. Springer, New Delhi. https://doi.org/10.1007/978-81-322-1602-5_75

[2] Rivest M, Vignola-Gagné E, Archambault É. Article-level classification of scientific publications: A comparison of deep learning, direct citation and bibliographic coupling. PLoS One. 2021 May 11

[3] Ioannidis JPA, Baas J, Klavans R, Boyack KW (2019) A standardized citation metrics author database annotated for scientific field. PLoS Biol 17(8): e3000384. https://doi.org/10.1371/journal.pbio.3000384

[4] https://www.kaggle.com/datasets/hadasu92/cnn-articles-after-basic-cleaning

[5] Maalouf, Maher. “Logistic regression in data analysis: an overview.” Int. J. Data Anal. Tech. Strateg. 3 (2011): 281-299.

[6] https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.htmlsklearn.svm.LinearSVC

[7] Joachims, T. (1998). Text categorization with Support Vector Machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds) Machine Learning: ECML-98. ECML 1998. Lecture Notes in Computer Science, vol 1398. Springer, Berlin, Heidelberg. https://doi.org/10.1007/BFb0026683

[8] Aas, Eikvil, 1999, Text categorisation: A survey, K. Aas and L. Eikvil, Technical report, Norwegian Computing Center (1999)

[9] Aggarwal, Zhai, 2012, A survey of text classification algorithms

[10] Marcin Michał Mirończuk, Jarosław Protasiewicz, A recent overview of the state-of-the-art elements of text classification, Expert Systems with Applications, Volume 106, 2018, Pages 36-54, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2018.03.058.

[11] Alper Kursat Uysal, Serkan Gunal, The impact of preprocessing on text classification, Information Processing Management, Volume 50, Issue 1, 2014, Pages 104-112, ISSN 0306-4573, https://doi.org/10.1016/j.ipm.2013.08.006.

[12] Shervin Minaee, Nal Kalchbrenner et al, Deep Learning–based Text Classification: A Comprehensive Review, ACM Computing Surveys, Volume 54, Issue 3, April 2022

[13] https://machinelearningmastery.com/data-leakage-machine-learning/

[14] https://medium.com/axum-labs/logistic-regression-vs-support-vector-machines-svm-c335610a3d16: :text=Difference%20between%20SVM%20and%20Logistic%20Regressiontext=SVM%20works%20well%20with%20unstructured,is%20based%20on%20statistical%20approaches.