Climate Change Misinformation Detection

Hardik Sawhney
(1042848)
Masters of Data Science
University of Melbourne

Abstract

The objective of the assignment was to build a system that was able to detect whether a document contains climate misinformation or not. The system is developed in a way that it reads the data from json files, pre-processes it (tokenization , stop words and punctuations removal , lemmatization), performs some feature engineering on it (count vectorizer, modal terms count , punctuation count) and feeds the features to the a Stochastic Gradient Decent model to check for misinformation. The accuracy of the model is 85% and the F1 score on the positive class is 0.854 on dev set.

1 Introduction

1.1 Problem Statement:

The problem in hand was to detect whether a given news article is a climate misinformation or not. For a news article to classify as a climate information, it should first of all talk about climate change and should contain ideologically driven misinformation about the climate change. By ideologically driven misinformation, we mean that it should provide wrong facts or refute the science behind the climate change. The basic idea behind solving this problem was simple, all the news articles that were not talking about the climate change at all, or news articles that provided genuine facts about the climate changes were not considered as misinformation while the articles that promoted personal opinions or created unnecessary fear were considered as misinformation.

1.2 System Design:

The system that is built to detect the climate misinformation has following 3 main stages: **(i) Data Cleansing:** The raw data from various json files is read into the code and is cleaned and pre-processed using various python libraries like NLTK and Spacy in order to make it ready for the next step. **(ii) Feature Extraction :** Then, the data is passed on to perform feature extraction on it. Various features like count vectorizer, modal terms count , climate words count, and punctuations count are extracted from the cleaned data. These features are then passed on to various classification models. **(iii) Model Selection:** This includes different approaches that are tried for this classification problem. The approaches are as follow: • One-Class Classification: Considering only the positive labels of the training data and trying to train a model using that data to classify the document as misinformation or not a misinformation. • Binary Classification: Training the model using both positive and negative labels in order to perform the classification task.

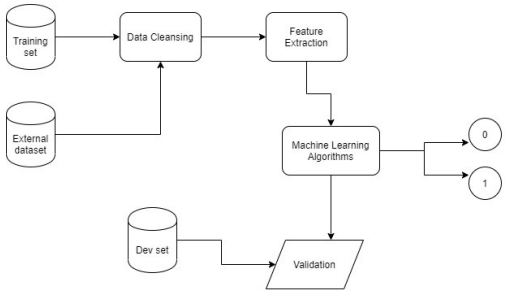

Figure 1: :System Design

2 Our System

2.1 Dataset:

The given dataset is divided into train dataset and dev dataset. The training set only has the news articles with positive labels meaning that all the new articles in training set are misinformation and has 1167 documents. Some external negative news articles (343) were also crawled from various websites in order to enhance the training set .The dev dataset has both the positive and negative articles and is used to evaluate the various machine learning

algorithms.There are 100 documents in dev dataset.

2.2 Data Cleansing :

This is the first step that is performed on the training data in order to clean the data before doing any further processing. This is because the unfiltered data may lead to many inconsistencies that can affect the accuracy of the model. We are reading the raw data from the json file using the python library and converting it into the dictionary. Then, the document and label of each document is getting separated and being stored in two different lists. Each document is then converted into lower case in order to reduce the possibility of two same words being counted as two different ones. By using NLTK package, each lowered document is tokenized, and all the punctuations and stopwords are removed. After all the punctuations are removed, the filtered document is lemmatized using Spacy in order to group the same inflected forms of the different words. Spacy was preferred over nltk because the lemmatization was being done without considering the POS-tags and results of NLTK in this case were not accurate.

2.3 Feature Extraction:

Features extraction plays an important role in text classification as it can exploit meaningful information from the text. It also helps in converting the text in a format that is supported by various machine leaning algorithms. In our case, we concentrated on the attribute-based features of the text which included quantity (Count Vectorizer, TF-IDF Vectorizer), uncertainty (quantifiers, tentative terms, modal terms), climate-related word counts and punctuation counts.(Li et al., 2019)

Count Vectorizer and TF-IDF Vectorizer : We have used a python library (Sklearn) to build the Count Vectorizer and TF-IDF vectorizer. Count Vectorizer converts the collection of text documents into the sparse representations of token counts(i.e. the vocabulary for the collection of texts). The pre-processed text was passed through this vectorizer to get the word count for each document in our training dataset. The parameters of the vectorizer were set in a way that the vocabulary was being created by ignoring the terms that were present in more than 95% of the documents. Moreover, the terms occurring in less than or equal to 3 documents were also ignored. TF-IDF or term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. TF-IDF scores were calculated on the pre-processed dataset using TF-IDF vectorizer by keeping the same setting as described above in order to make a meaningful vocabulary. Also, different values of n-gram range were tried in order to help the model to understand the context of the text. The n-gram range was varied from unigram to trigram in order to include words like 'global-warming', 'climate-change', 'seasonal temperature fluctuations' in the vocabulary. Table 1 shows the vocabulary size for each n-gram setting that was tried.

N-Gram Range	Vocab-Size
(1,1)	9024
(1,2)	26351
(1,3)	32319
(2,2)	17327
(2,3)	23295
(3,3)	5968

Table 1: Vocabulary Size

Uncertainty : It is often understood that usage of modal words like 'must', 'might', 'will', 'would' etc and quantifiers like 'greatly', 'enough', 'mostly' etc indicate modality that is likelihood of something. The degree of uncertainty increases with increase in frequency of such words. Since, most of these words were removed in data cleansing, their count was separately taken for each document in order to check the degree of certainty in each document. It was thought that the document with more frequency of modal terms would have more chance of being a misinformation.

Climate word count: In order to give more weights to the climate words, the words like temperature, climate and warming were counted in each document to learn whether the document was talking about climate or not.

Punctuation count : The number of '!' and '?' were also counted in each document in order to check whether the article was a personal opinion or a news article with facts and evidence.

2.4 Model Selection:

The results from the feature extraction are used to train various machine learning algorithms. Two approaches are followed in order to the train the model i.e. One-class classification and Binary Classification

One-class Classification: Since, the given dataset had only one type of labels, it was decided to follow one-class classification approach. In this approach, the model tries to learn the characteristics of the objects of only one class and tries to distinguish between them and potential outliers. Two models mainly One-class SVM and Isolation Forest were used for this. Initially, both the models were fitted using only TF-IDF feature and then the features were gradually increased. The accuracy and the F1 score of the models decreased with gradual increase in the features as shown in the results section. This is because the model was not able to find the dividing boundary with more features (Manevitz and Yousef, 2001).

Binary Classification: After seeing the poor performance of both the one-class models, it was decided to go with a different approach. The external dataset i.e. news articles with negative labels were crawled form various news websites in json form.(Gua).The news articles were selected in a way that some articles were not talking about the climate change while other were discussing genuine climate- change facts with evidence.(Kag) Three models : Stochastic Gradient Decent, Two-Class SVM and Logistic regression are trained using the various features discussed in the previous section and evaluated on dev dataset. Count-vectorizer was performing better than TF-IDF vectorizer as the weights for TF-IDF vectorizer were not scaling as well as for Count-Vectorizer.Therefore, it is used as base feature and other features are added on top of it. The results of the same are discussed in the Result section.(Akhter et al., 2020)

3 Result:

In this section , we talk about the results of various models that were trained. Figure 2 and Figure 3 shows the F1 score of positive class for One-class SVM and Isolation forest with respect to different feature.

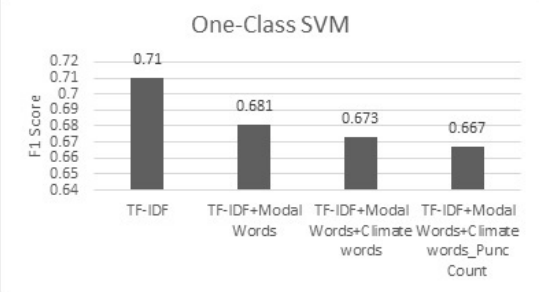

Figure 2: :One-class SVM

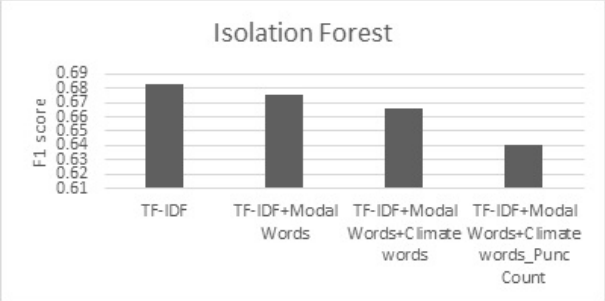

Figure 3: :Isolation Forest

Figure 4 shows the performance of TF-IDF vectorizer with respect to Count-Vectorizer while Figure 5 shows the F1 score and Precision of the positive class for the various binary classification models with respect to the different features.

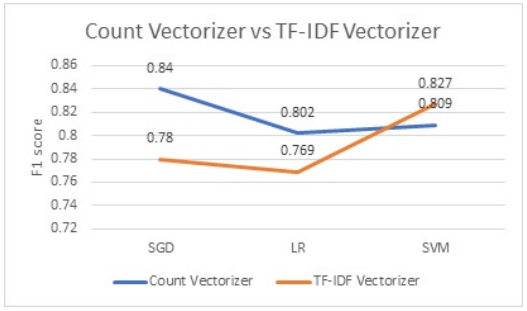

Figure 4: :Count Vectorizer vs TF-IDF

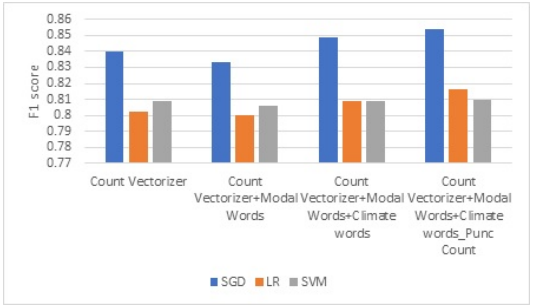

Figure 5: :Binary Classification

Its evident from Figure 5 that Stochastic Gradient Decent (SGD) is performing best among all the models. It was selected as the final model and its hyperparameters were tuned. Table 2 and Table 3 shows F1 score for SGD for different values of alpha and n-gram range of Count-Vectorizer.

The model with ngram range (1,3) and alpha=0.001 is selected as the final model.

4 Error Analysis:

We have carefully analysed the performance of each model and listed out the possible errors that each one of them were making.

N-Gram Range	F1-score
(1,1)	0.808
(1,2)	0.844
(1,3)	0.854
(2,2)	0.844
(2,3)	0.735
(3,3)	0.667

Table 2: F1-score

Alpha	F1-score
0.0001	0.846
0.001	0.854
0.01	0.810
0.1	0.790
1	0.760

Table 3: Alpha Values

4.1 Stochastic Gradient Decent:

By seeing the metric report (Figure 6), we can clearly see that the model performs somewhat poorly on the negative labels. This means that the news articles that had no misinformation were being classified as misinformation. The model is giving over predictions on the positive class as the model is trained on imbalanced dataset with positive class having 1167 instances while the negative class has only 343 instances.

	precision	recall	f1-score	support
0	0.87	0.82	0.85	50
1	0.83	0.88	0.85	50
accuracy			0.85	100
macro avg	0.85	0.85	0.85	100
weighted avg	0.85	0.85	0.85	100

Figure 6: :Metric Report

The ROC curve(Figure 7) shows that the model is giving a decent performance, but it can be improved further.

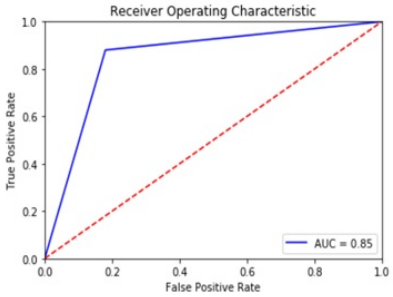

Figure 7: :ROC-Curve

4.2 Logistic Regression and SVM:

Both Logistic Regression and SVM model were performing poorly on the positive class. They were classifying the articles with misinformation as not a misinformation which was highly affecting the accuracy of both the models. The precision and recall for negative of both the models were around 0.79 and 0.82 indicating that both the models did not converge well. Also, SGD was performing much better on both the classes as compared to these two models.

5 Future Improvements:

We can improve the system by building word embeddings before passing the pre-processed data for feature extraction. This will be useful as the words that have the same meaning will have a similar representation in feature space. This will also help the model to learn well-scaled weights and will lead to better performance of the whole system. Also, we can consider the structure-based features in order increase the accuracy of the model. We can extract semantic and discourse information from the text, such as part-of-speech taggers, polarity, subjectivity etc in order to build a more efficient model. Finally, the code can be optimized to extract the POS-tags in less time as discussed in Data Cleansing section which will help in better lemmatization of each document.

6 Conclusion:

The F1-score of the model in dev set is 0.854 with the precision of 83% on the positive class.The F1-score of the model is 0.64 on positive class for the test dataset. It is a moderate improvement from the baseline model but can be improved further if the ideas from section 5 are followed.

References

https://www.theguardian.com/au.

https://www.https://www.kaggle.com/.

M. P. Akhter, Z. Jiangbin, I. R. Naqvi, M. Abdelmajeed, A. Mehmood, and M. T. Sadiq. 2020. Document-level text classification using single-layer multisize filters convolutional neural network. *IEEE Access*, 8:42689–42707.

Quanzhi Li, Qiong Zhang, Luo Si, and Yingchi Liu. 2019. Rumor detection on social media: Datasets,

methods and opportunities. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 66–75, Hong Kong, China. Association for Computational Linguistics.

Larry M Manevitz and Malik Yousef. 2001. One-class svms for document classification. *Journal of machine Learning research*, 2(Dec):139–154.