

Fake News Detection Using Machine Learning

Final report

Submitted for the BSc in

Computer Science

April 2020

by

Idrees Malik

Word count: 16295

Abstract

This paper explores various natural language processing techniques available within the machine learning domain in order to find and detect fake news. Using a suitable dataset that is found on the website Kaggle, we apply various feature extraction techniques including Term-Frequency Inverse Document Frequency along with Bag of Words at various n-gram levels on a corpus 20,000 large. We use various classification techniques which include, Multinomial Naïve Bayes, Logistic Regression, Linear Support Vector Machine, Random Forrest and Voting Ensembles. We find various findings including Linear-SVM achieving 95% accuracy at word level TF-IDF.

Acknowledgements

I would like to acknowledge my supervisor Ashley Williamson for giving me the direction and reassurance during the project from start to finish. Without his guidance, I would not have been able to fully complete the project, as due to his insights and advice I was able to prevent myself from falling into any pitfalls.

Further acknowledge goes out to my friends and family who helped me stay motivated and pushed me to learn, and through their utmost support and encouragement through this journey I was able to complete my dissertation and maximise my time at university.

Contents

Abstract	i
Acknowledgements	ii
1 Introduction	4
1.1 Background	4
1.2 Design Brief	5
1.3 Understanding Fake News	5
1.3.1 Characterization	6
1.3.2 Feature Extraction	6
1.3.3 News Content Models	7
1.4 Development Methodology	7
1.4.1 Cross-Industry Process for Data Mining	7
1.4.2 Agile Development	8
.....	9
1.5 Aims and objectives	9
1.5.1 Project Objectives	10
1.5.2 Project Deliverables	11
1.6 Report Structure	11
2 Literature Review	12
2.1 Introduction	12
2.2 Supervised Learning for Fake News Detection	12
2.3 Detecting Fake News using Machine Learning and Deep Learning Algorithms	13
2.4 Fake News Detection Using Stacked Ensemble of Classifiers	14
2.5 Comparison of Various Machine Learning Models for Accurate Detection of Fake News ..	15
2.6 Multiclass Fake News Detection Using Ensemble Machine Learning	16
2.7 Conclusion	17
3 Design & Methodology	18
3.1 Outline	18
3.2 Data Source	19
3.2.1 Fake News Dataset Understanding	19
3.3 Data Preparation	20
3.3.1 Feature Selection & Minimization	20
3.3.2 Data Cleaning and Consistency	20
3.3.3 Removing Non-English Words	21
3.3.4 Removing Punctuation	21

3.3.5	Removing Stop Words, Tags and Special Characters	22
3.3.6	Lemmatization.....	22
3.4	Feature Extraction Techniques	22
3.4.1	Term-Frequency Inverse Document Frequency.....	23
3.4.2	Bag of Words Approach	23
3.4.3	N-grams Analysis	24
3.5	Modelling.....	24
3.5.1	Multinomial Naïve Bayes	24
3.5.2	Logistic Regression	25
3.5.3	Linear Support Vector Machines.....	25
3.5.4	Random Forrest Classifier	26
3.5.5	Ensemble Modelling.....	26
3.6	Model Evaluation Techniques.....	27
3.6.1	K-Fold Cross Validation	28
3.6.2	Confusion Matrix & Key Terms	28
3.7	Final Remarks.....	30
4	Implementation & Justification	31
4.1	Software Development Language.....	31
4.1.1	C++.....	31
4.1.2	Python.....	31
4.2	Integrated Development Environment	32
4.2.1	Jupyter Notebook.....	32
4.2.2	PyCharm.....	32
4.3	Key Libraries.....	32
4.3.1	NumPy	32
4.3.2	NLTK.....	33
4.3.3	Scikit-Learn	33
5	Results & Evaluation	34
5.1.1	Experimental Reiteration.....	34
5.2	Experiment 1: Results & Evaluation	34
5.2.1	Word Level Analysis.....	34
5.2.2	Bigram Level Analysis	35
5.2.3	Character Level Analysis	36
5.3	Comparing obtained Results Against Published Works	37
5.3.1	Word-Level Comparative Analysis.....	37
5.3.2	Bigram Level Comparative Analysis.....	37

5.4	Experiment 2: Results & Evaluation	38
5.4.1	World-Level Analysis	38
5.4.2	Bigram-Level Analysis	39
5.4.3	Character – Level Analysis	39
5.5	Comparing obtained Results Against Published Works	40
5.5.1	Unigram & Bigram Level Analysis.....	40
5.6	Experiment 3: Results & Evaluation	41
5.7	Overall Evaluation	42
6	Conclusion.....	43
7	References.....	44

1 Introduction

This document will analyse and outline the gradual process of development, as well as providing a detailed description and discussion of the background material in order to accurately outline an approach for detection of fake news within the online news domain. This report will detail all the technical development required of the project, with each stage being discussed and analysed, in regards to finding the most appropriate feature extraction required and also finding and choosing the most appropriate models in order to find and evaluate the given corpus against the performing classification algorithms. An evaluation at the end will be performed in order to consider how well the architecture meets the objectives created and specified in the PID. After the evaluation, pointers on what would be done in order to improve the accuracy and general performance of the system will be drawn up and explained if certain restrictions and resources were available and in place.

1.1 Background

In recent years fake news, also known as junk news or pseudo-news, is a type of yellow journalism that consists of deliberate dis-information in the hopes of the readers believing this false information in order to spread their own propaganda. In the age of the internet where 62% of US citizens get their news from online media (Jeffrey Gottfried, 2016), fake news is now being spread more than ever, blurring the lines between fact and fiction. Due to recent political and socioeconomic world events there has been a significant rise in fake news causing it to become one of the biggest threats to many industries. As a result, there have been many attempts to detect and stop the spread of fake news. This problem has proved trickier to solve than originally thought, with various different methods being used, from manually being able to detect fake news to now using state of the art artificial intelligence techniques in order to detect fake news.

Fake news was around before the age of the internet as publishers used false or misleading information to further their own interests. As time has moved along and as technology has vastly advanced the non-ambiguousness of fake news in the media became more and more transparent. Following the advent of the web the number of consumers being exposed to such misinformation over online platforms has increased. The combination of ease of access to media platforms over the internet and the convenience of being able to access such information so easily meant people were more accepting of the content they read online. As a result of this, the approaches required to detect fake news had to develop in order to stop this spread of misinformation.

Developing an efficient and accurate solution for a supervised machine learning program in order to detect the fake-ness within a news articles in itself is a technically challenging task. Manually checking whether a given article is fake or not itself is a flawed approach as humans we a very subjective allowing our opinions and thoughts to cloud our better judgement. Therefore, being able to find the exactitude of a news article which is fake is a complex and tiresome task, even for human experts in this given field (University of Michigan, 2018).

Machine learning is a branch of artificial intelligence which offers hopes in improving the detectability of fake news through supervised learning. Machine learning techniques analyse the dataset it's being trained on in order to find recurring lexical and textual patterns and linguistic cues in order to detect whether the given corpus is considered fake. This approach is different to fact checking algorithms which cross-reference an article with other pieces to see if there are any inconsistencies within the text. This approach to detecting fake news is already considered to be better than humans (Tristan Greene, 2018), with some machine learning models obtaining >95% accuracy (Aaron Edell, 2018).

Economically, fake news may exert devastating effects not only on individual people and the society they live in but also industries within that society. With so many readers so willing to believe what they read on the internet and take it for fact. Take for example news that spread in a Vietnamese newspaper in 2007, the news article stated there was a connection between eating grapefruits and having cancer. This article led to a significant drop in the prices of grapefruits to only 10% of the original price and as a result impacted not only the economy of the country but also to the lives of all the farmers as they were not able to sell their grapefruits for what they originally were (Kinh Doanh, 2020). As a result of all the damage that was done, the related newspaper agencies that were responsible for spreading such news were charged with spreading the fake news.

In this report I experiment the possibility to detect web based fake news using textual and lexical analysis by applying traditional machine learning techniques, as well as evaluating each different machine learning technique in order to find which approach work the best. The outcome of this project should determine how much can be achieved by analysing patterns contained within text using various lexical and textual analysis.

1.2 Design Brief

The original project has been re-structured as various aspects of the project specification changed during the research and development stage, as a result the final specification of the project reads as follows:

Data mining allows for deeper analysis and decisions to be made based on patterns and relationships discovered during the data mining process. Data mining allows for specific patterns to be found from the given data source. Depending on the strength of the given data source and how relevant the given data is to the specific problem domain, more accurate and specific patterns can be discovered. This project intends to utilise a given data source (relevant to the given domain) in order to accurately predict the reliability of news articles, in order to combat fake news. Appropriate machine learning techniques will be deployed upon the given cleaned data source. The purpose of cleaning the given data source is to ensure there are no inconsistencies within the dataset and prevent false conclusions. This benefits the modelling stage as it allows relevant patterns to be found, which will then be tested and evaluated. Various evaluation techniques will be put to use to ensure the reliability and accuracy of the given approach. The intended outcome of the project will be to accurately detect the reliability of news articles in order to then predict the likelihood of the given news article being fake.

The given project will be developed entirely in Python due to how well suited the given language is to the machine learning and artificial intelligence domain, with the access of various libraries and tools available to allow full implementation of various approaches used in natural language processing. As well as taking into consideration the given timeframe, Python is one of only few languages to realistically allow the full completion of the project.

1.3 Understanding Fake News

In order to work on fake news, it's important to understand the problem domain a little further, therefore it's important to understand what fake news is and how it's characterized. The following understanding of characterization of fake news was based of Fake News Detection on Social Media: A Data Mining Perspective (Fake News Detection on social media: A data mining Perspective, Kai Shu).

We will split problem domain into two sections in order to gain a better understanding of each individual part. The first section is in regard to characterization of fake news, the second is detection of fake news. In order to build a machine learning model there needs to be some sort of characterization, as we cannot solve a problem without understanding the problem itself.

1.3.1 Characterization

One such definition of fake news are articles that are intentionally and verifiably false and could mislead readers (Hunt, 2017). There are two aspects of this definition that need to be considered: authenticity and intent. The first aspect of this definition, authenticity means that fake news contains false information that can be verified as such. Secondly, intent means that the misleading information has been written as such with the intention of purposely spreading false news for various personal reasons.

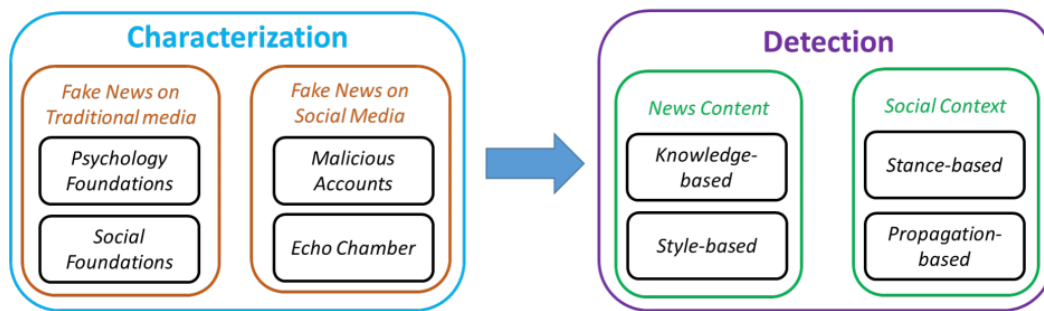


Figure 1 Fake News Online: Characterization to Detection

1.3.2 Feature Extraction

Now there is a better understanding of what fake news is, with a basic definition set and defined, we now need to analyse what features are able to be used in order to classify fake news logically. As the problem domain that will mainly be focussing on is online news content due to the plethora of fake news available, there are four main component of online news content:

- **Source:** Where does the given news article come from, who is responsible for writing the article, and whether the given source is reliable or not.
- **Headline:** A short overview of the news content, which are worded to entice the reader in order to click on the given article.
- **Body/Text:** The actual article containing all the information.
- **Images/Videos:** This is usually combined with the body of text in order to further visualise what the text is trying to say.

Features will be extracted from the first three basic components of a news article, with the main feature being the body of text allowing for linguistic-based and textual based analysis. Using the previous definition fake news contains an aspect of false intent and un-authenticity in order to mislead the consumer, in order to achieve this, specific language is used i.e. formality of the article. On top of this, lexical analysis of the linguistic feature in tandem can be used to analyse word-frequency and the occurrence of the same words, as well as the sentiment of the given text.

The reason as to why visual features are not going to be able to be considered is because they are not within the given domain as in order to be able to analyse such a feature more complex technologies such as deep learning or CNN's would need to be considered on top of machine learning. Yet generally speaking the fact that images/videos are present along with the piece of text would generally back up the authors statements and would carry more weight, meaning the given article is less likely to be false if there is visual proof to back it up.

1.3.3 News Content Models

Various different types of models can be implemented using the features explained previously. One such example is related to news content and is based on knowledge. Based on the information provided in the given model the given model will check the truthfulness of the news content. This can be achieved in a news content model in three different ways:

- **Expert – Orientated model** – This approach relies on experts, such as scientist (people of the respective fields), to assess the truthfulness of the news
- **Crowdsourcing – Orientated Model** – this approach essentially says if a large enough number of persons say something is either true or false, then they must be true.
- **Computationally – Orientated Model** – this approach relies on computational fact-checking of the news content in order to find if it's truthful or not.

These such methods have their own advantages and disadvantages. In the case of expert-orientated hiring such experts is a good approach to ensure truthful articles, but it can potentially be costly. Crowdsourcing can be easily debunked as the large group of persons could all be wrong about something; they might perceive to be correct.

1.4 Development Methodology

During the project a mix of the data mining standard process CRISP-DM along with an Agile development approach will be used when developing the given architecture for the project. The purpose for combining both development approaches together is because during the given project there was a lot of information learned, meaning that previous implementations within the given code were replaced for more appropriate approaches allowing for better results to be obtained.

1.4.1 Cross-Industry Process for Data Mining

In order to ensure quality results, a standard process model will be implemented which is followed throughout the whole data mining process. It's important to select the most appropriate process model, as the process model helps plan, organize and structure the whole data mining project. As well as this process models allow for easier repeatability of results as each data mining project follows a specific methodology. It was found the most suitable process model for the given data mining project was Cross-Industry Process for Data Mining (CRISP-DM).

CRISP-DM provides a structured, industry-proven approach for planning a data mining project. This process model provides a hierarchical overview of the data mining life cycle. The CRISP-DM process model aims to make large data mining projects, less costly, more reliable, more repeatable, more manageable, and faster (H Wiemer, 2019). The CRISP-DM model contains 6 stages:

- **Business Understanding** – This stage of the process model is concerned with an in-depth analysis of the objects the data mining project needs to define.
- **Data Understanding** – Includes collecting data, and then interpreting the collected data. Once data collection is done; the properties of the given data need to be examined in order to assess the quality of the given data. This is important as the data may need to be refined in order to meet the data quality objectives.
- **Preparation of data** – This stage is often one of the most time-consuming aspects of data mining. This stage involves selecting the data that will be used in the modelling stage (usually based off a criterion that usually links the data to the data mining goals). Data preparation also involves raising the data quality to allow for accurate analysis in the further stages, this is done through data cleaning (CRISP-DM, 2020).

- **Modelling** – The objective of the modelling stage is to select a model(s) that can then be trained with the data in order to a valid outcome to be achieved and assessed. It is common for several models to be run on the single dataset in order to build up a comparison to then find the model of choice.
- **Evaluation** – The outcomes generated for each model is then assessed against the original objectives. This can then lead to new objective as new information and patterns have been discovered.
- **Deployment** – The information and data discovered during the previous steps are then presented to stakeholders for a strategy to be determined for deployment, in order to integrate the findings from the data mining project into the real world.

The main reason for CRISP-DM being favoured over all other standard process models such as KDD or SEMMA, was due to CRISP-DM being cross industry standard; meaning that this methodology can be implemented in any DS project notwithstanding its domain or destination. Other key reasons for choosing CRISP-DM include:

- **Flexibility** – At the beginning of the project a lack of domain knowledge and ineffective data, this leads to many pitfalls and mistakes in the beginning. Due to CRISP-DM being flexible, this allows the models and processes at the beginning to be imperfect. Allowing over time for each stage to improve gradually as domain knowledge and data quality increases.
- **Long-Term Strategy** – At the beginning the model cycle could be small and simple, but due to further iterations the model can develop and become stronger. This allows for the model to develop along with the data scientists allowing for both to gain an understanding over time.

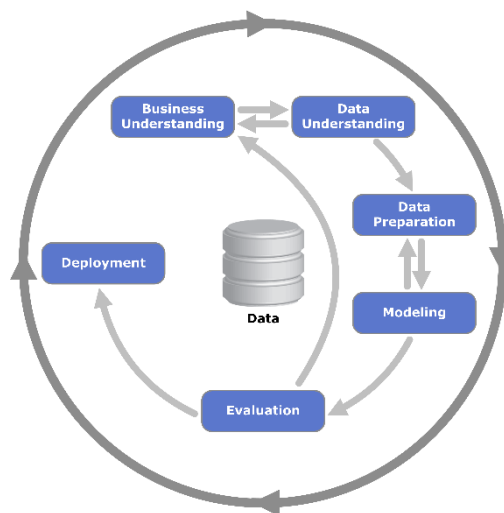


Figure 2 CRISP-DM Methodology

1.4.2 Agile Development

As well as implementing the CRISP-DM standard process for the reasons stated above there will also be an agile methodology aspect to this project as well. The main reason for implementing an agile development approach is because it favours a rapidly changing environment allowing for the architecture to change with ease based on the new requirements. This allows for there to be changes to the project in order to better the original specification of the project to allow it to keep up to date with new approaches as the authors domain knowledge in the given field increases.

Agile development currently is industry relevant, as in the real world it's common practice for customers to change their requirements for a system during the development process in order to keep with the times. This could potentially be case with the given project or due to project specification changes. This would make an agile approach more desirable in the given circumstances.

The agile approach was favoured over other development methodologies because:

- **Waterfall Approach** - The reason for not using the waterfall approach was due to the fact that such a development process is not flexible enough in terms of changes to the project after the requirements have been set and development has begun. This is something which could potentially happen during this project so such an approach could be detrimental to the success of the project.
- **Spiral** - The spiral approach is a combination of the waterfall and iterative approach; it was considered for the project but was not used mainly due to the fact that the spiral approach favours larger projects.
- **Prototyping** - The prototyping model was considered mainly due to the fact that many prototypes of the artefact will be built. This allows for a feel of the given artefact. Although the other two previous development approaches chosen against it in the end mainly due to the fact that prototyping is a time-consuming approach and given that the project has only a limited amount of time, this could potentially jeopardize the outcome of the artefact.

The success of the given methodologies can be measured in various ways such as; if the artefact is meeting the specification scope and requirements, and if the artefact is research relevant as using out of date approaches wouldn't contribute much to the research field.

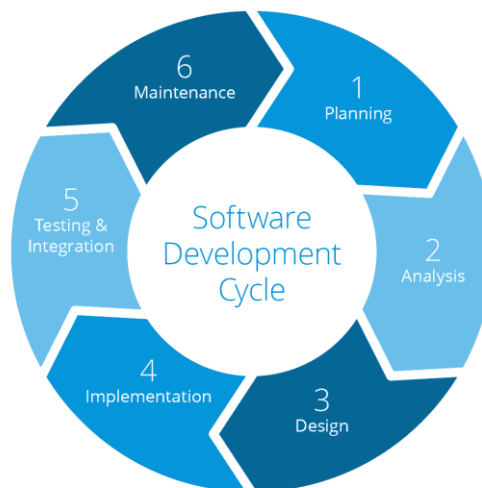


Figure 3 Agile Development Cycle

1.5 Aims and objectives

The original specification that was developed had been created with developing the architecture in C# using statistical methods in order to find and detect fake news. However, after doing much research and having a discussion with my primary supervisor, the project had changed its focus from developing the architecture in C# using statistical methods and shifted towards using machine learning in python. This approach means that the project will no longer require an interactive user interface due to the shift from the project being software orientated to become more research orientated and as such merits the reason to remove such functionality. However certain elements from the original

specification will be carried over to the new specification. Further information of the new specification will be detailed below.

1.5.1 Project Objectives

1.5.1.1 *Objective 1 – Obtain a Relevant Data Source*

In order to execute any data mining tasks a data source is required as creating a new primary data source from scratch would take far too long considering the given time constraints. As a result of the time constraint, an appropriate data source for the domain of fake news will be sourced and utilised instead. The dataset will need to contain at the minimum, English news articles which are clearly labelled as fake or not. A large enough amount of data is also required for the data mining task.

As with many secondary data sources a cleaning phase is required to ensure the quality of the data source and data within is up to standard to ensure the data is correct and consistent, as well as ensuring any unsuitable data columns within the data source are removed before the modelling stage occurs.

1.5.1.2 *Objective 2 – Find and Implement Appropriate Feature Extraction Techniques*

In order to properly prepare the given data within the data source the corpus of the news articles needs to be converted into a set of real numbers/vectors. This process of vectorization allows the given machine learning algorithms to extract useful features from the given corpus thus allowing the given machine learning classifier the ability to predict whether a given news article is reliable or not. The purpose of this objective is to research various feature extraction techniques available in the given NLP domain to then chose the most appropriate techniques which will then be implemented on the given corpus. This aspect of comparing feature extraction techniques allows us to compare and analyse which vectorization process is more appropriate for the given domain. As implementing a potentially unsuitable vectorization process may lead to substandard results during the modelling phase of the given project.

1.5.1.3 *Objective 3 – Implement Appropriate Modelling Algorithms*

In order to properly obtain suitable and accurate results there is a strong importance to use the correct classifier. As with any data mining task if the suitable classifier isn't chosen then the final conclusion achieved will not be either reliable or accurate. Given the problem-domain for fake news deals with mainly text, certain algorithms are more preferable due to such algorithms favouring that kind of data. During this objective it's important to research various sources in order to understand and acquire important information about which algorithms are more appropriate and suitable for the problem than others using sources such as research papers or trial and error.

1.5.1.4 *Objective 4 – Use Appropriate Evaluation Techniques*

Evaluating the performance of such models is an integral component of understanding whether the model is merely memorizing the data it's being fed or if it's predictions can be trusted. During this process the degree to which the model meets the given objectives that were set will be determined. There is a great importance on evaluating the performance of the given models as repeatability of results is important. The main purpose of this objective is to determine whether the given results obtained during the modelling stage is consistently accurate and reliable using the most suitable evaluation techniques available.

1.5.2 Project Deliverables

- Provide an architecture which is able to evaluate given data using up-to-date feature extraction techniques and appropriate modelling algorithms to effectively determine the truthfulness of an article using linguistic and lexical analysis.
- Thesis report document
 -

1.6 Report Structure

The remainder of the thesis is structured into three following parts:

- Second Chapter - Provides the literature review and background research conducted during the duration of the project.
- Third Chapter – Provides the technical development behind the data mining processes, modelling and evaluation techniques that were chosen.
- Fourth Chapter – Provides brief overview of the Implementation of the given artefact, with justification.
- Fifth Chapter – Provides discussion and critical evaluation of the results obtained with given consideration of the results.
- Sixth Chapter - This chapter includes the conclusions and recommendations as to evaluate the whole methodology and find if the overall project was considered a success based on the aims and objectives.

2 Literature Review

2.1 Introduction

In this chapter, there will be critical analysis of methodology and evaluation of related works that are strongly linked to the given thesis domain. The scope of this background research will vary, as various different approaches within Artificial Intelligence have been utilised in order to combat the Fake News problem domain. This chapter will document some of the related works previously carried out within the given field; each piece of related work will have some aspect which is linked to the given project in some sense from feature extraction to modelling techniques to evaluation techniques.

2.2 Supervised Learning for Fake News Detection

Reis et al (2019) used various state-of-the-art machine learning classifiers on a Buzzfeed dataset related to the 2016 United States presidential election. Reis et al (2019) evaluates the machine learning algorithms, K-Nearest Neighbour, Multinomial Naïve Bayes, Random Forrest, Support Vector Machine (SVM) with RBF Kernel and XGBoost. The corpus used for the modelling on the previously stated classifiers had various feature extraction techniques at different levels such as:

- Language Features – Sentence level features including bag of words (BOW), n-grams, and part of speech were all explored in the feature extraction technique, including evaluating the writer's style of writing as a potential indicator for detecting fake news.
- Lexical Features – Including character and word length level, analysing the unique words within the given corpus and the frequency of the unique words, furthermore first-person pronouns, demonstrative pronouns, verbs and punctuation were all also considered in a lexical aspect on the given corpus.
- Psycholinguistic Features – Psycholinguistic features were also analysed as dictionary-based text mining software were utilised in understanding and finding fake news within the given corpus.
- Semantic Features – Including creating a toxic score using a google API.

Various other feature extraction techniques were utilised with the aforementioned feature extraction techniques being the main techniques used.

The results are shown below in **figure 4**

The given results show that XGBoost is more preferred for detecting fake news on text that needs to be hand-verified, meaning that the text that are classified as not-fake, are indeed not-fake. This allows for the amount of text that needs to be checked manually to be reduced.

Table 1. Results obtained for different classifiers w.r.t AUC and F1 score.

Classifier	AUC	F1
KNN	0.80±0.009	0.75±0.008
NB	0.72±0.009	0.75±0.001
RF	0.85±0.007	0.81±0.008
SVM	0.79±0.030	0.76±0.019
XGB	0.86±0.006	0.81±0.011

RF and XGB performed best.

Figure 4 Results for Supervised ML Approach

Reis et al (2019) shows that not only can you can you implement various aspects of machine learning in order to detect fake news successfully, but also achieve very high scores for the given dataset as Reis et al (2019) was able to achieve relatively good results across the board for all the machine learning classifiers chosen with the given Buzzfeed dataset.

2.3 Detecting Fake News using Machine Learning and Deep Learning Algorithms

Tanvir et al (2019), use the social media platform Twitter due to the plethora of information in order to compare the performance between the machine learning algorithms – Support Vector Machine (SVM), Multinomial Naïve Bayes, Logistic Regression and a Recurrent Neural Network (RNN) in order to determine the best approach required to detect fake news.

The dataset uses the social media platform Twitter, more specifically, this data includes information from the Chile earthquake in 2010, due to twitter playing a significant role during this time in both co-ordination and misinformation. The dataset contained 20,360 twitter articles which for the given context and training is enough to create a well performing Machine Learning and Recurrent Neural Network.

In order to ensure the given corpus is ready for modelling the dataset had the various feature extraction techniques applied to itself in order to ensure the dataset was in vector form and algorithm recognisable. These feature extraction techniques involved:

- Count Vectors
- Term-Frequency Inverse-Document-Frequency (TF-IDF) at various n-gram-analysis levels
- Word Embedding

(Further details of the given feature extraction techniques can be found in section 3.4, Design & Methodology)

The results showed that including all modelling techniques from Supervised Machine Learning, LSTM Deep Learning and Recurrent Neural Network, it was shown that out of the Machine Learning Algorithms SVM model performed the best in both vectorization feature extraction techniques with Multinomial Naïve Bayes considerably close. Regarding the Deep Learning Algorithms LSTM and RNN both achieved similar results (default and with kernel initialization) with the given results being considerably less accurate compared the machine learning counter parts. Logistic Regression performed the worst of all modelling techniques with the reason being was due to the model becoming too overconfident as it was trying to predict outcomes based on a set of independent variables.

ble 1: Accuracy comparison after cross validation


Classifiers	ACCURACY	
	Count Vector	TF-IDF
Logistic Regression	62.47	69.47
Naive Bayes	84.56	89.06
 SVM	89.34	89.34

Figure 5 Supervised Machine Learning Results


Classifiers	ACCURACY	
	Default	With Kernel initialization
Long Short-Term Memory (LSTM)	73%	76%
 Recurrent Neural Network (RNN)	74%	73%

Figure 6 Deep Learning Modelling Results

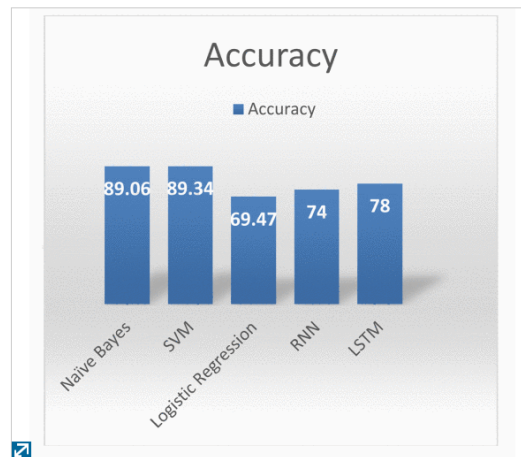


Figure 4:
Overall accuracy comparison of machine learning and deep learning algorithms

Figure 7 Supervised ML vs Deep Learning Accuracy Results

Various aspects of the project overlapped with the thesis project, as aspects such as the feature extraction methodologies chosen in order to ensure the corpus in vectorization form is most appropriate for the modelling phase and the given machine learning classifiers chosen by Tanvir et al (2019). Tanvir et al (2019) was able to show that TF-IDF was equal to or always considerably better than Count Vectorization, this is also an aspect of the given research project to be explored within the thesis project.

2.4 Fake News Detection Using Stacked Ensemble of Classifiers

Thorne et al decided to combat the *Fake News Challenge* by proposing the idea of using an ensemble comprised of five individual systems. Stacking was involved in the ensemble technique as it had previously been applied to a number of other successful tasks, with the stacked ensemble score above the accuracy of the individual classifiers. The given individual classifier used in the stacked ensemble in this particular research paper used: a multilayer perceptron with a relu activation on the average of the Word2Vec for the headlines and article body, with Term-Frequency Inverse-Document-Frequency (TF-IDF) at the analysis level of bigrams, unigrams on the particular article body, along with this Logistic Regression with L2 regularization and concatenation of Word2Vec for the headlines and article body with MLP and dropout. The given stacked ensemble that utilises these particular classifiers is set up in a two-layer classification architecture that leverages predictions from weaker slave classifiers as given features in a stronger classifier. This whole process is simplified for better understandability below:

- Classifier 1 – Concatenates the average Word2Vec vectors for the given headline and article body, using TF-IDF on the cosine-similarity between the headline and article body. This information is fed into an MLP with a Relu activation function.
- Classifier 2 – 4-way classification using MLP with a Relu activation function with the average Word2Vec embedding for the given headline and article excluding stop words and indicator features for punctuation.
- Classifier 3 – 4-way classification using one-vs-all logistic regression with L2 regularization over words including unigram and bigram word level TF-IDF vectorization.
- Classifier 4 – Utilises Word2Vec embedding on headlines and article body on an MLP classifier with a dropout probability of : 0.5, 0.3, 0.1, between layers with the given MLP classifier using a Relu activation function.
- Classifier 5 – Gradient Boosted decision tree classifier using the feature values predicted from the slave classifiers.

System	Dev %	Test %
Official Baseline ³	79.53	75.20
SOLAT in the SWEN ⁴	-	82.02
Athene ⁵	-	81.97
UCL Machine Reading ⁶	-	81.72
C1	88.09	75.77
C2	86.68	75.08
C3	87.48	77.99
C4	87.36	58.69
C5	79.25	75.22
Our Ensemble (CM)	90.05	78.04
CM Upper Limit	97.25	90.89

Figure 8 Results for Slave Classifiers and Stacked Ensemble

From the results shown in figure 8, each individual slave classifier performed well with the best of all the modelling techniques being the stacked ensemble. The results for the test dataset show that the models did not bode well with the Thorne et al stating the difficulty level between the train and test datasets being too much which resulted in the models not performing at their optimal levels. As a results the outcome of the results were disappointing but eye opening, with Thorne et al stating including temporal splits could help the classifiers perform better in a future paper on the given topic, as the models were not able to generalize the overlapping headlines between the training and testing datasets.

2.5 Comparison of Various Machine Learning Models for Accurate Detection of Fake News

Poddar et al compares various probabilistic and geometric machine learning models in order to find the best computational machine learning model in order to tackle the fake news problem domain. The various models that were compared include Multinomial Naïve Bayes, SVM, Logistic Regression and Decision Trees in order to predict the fake news. The feature extraction techniques utilised in order to vectorize the given text within the dataset are Count Vectorization and TF-IDF. Before any extraction of the text was done various steps were taken to ensure that Poddar et al had the text cleaned which included stages of cleaning which involved removing stop words and punctuation.

The dataset that Poddar et al decided to use was a dataset found of Kaggle, the dataset contained 13000 fake news articles that were taken from 244 websites which were all associated with fake news. Such articles within the dataset were taken mainly between the year 2016 -2017.

Poddar et al showed that the Support Vector Machine along with the feature extraction TF-IDF yielded the best results compared to the same modelling approach with Count Vectorization. Multinomial Naïve Bayes algorithm tended to attain a greater accuracy result when being tested on shorter length article where both Logistic Regression and Support Vector Machines on the whole achieving better results when a larger test news article. Throughout the whole experiment it was found that the Decision Tree consistently performed poorly with both TF-IDF Vectorization and Count Vectorization on various corpus lengths.

Poddar et al utilises various techniques and approaches in order to detect fake news with a high level of accuracy across the board for all given models. Such various approaches have been incorporated into this thesis from removing stop words to using both feature extraction techniques, Count vectorizer and TF-IDF used within the given research papers and two of the four Machine Learning classifier also implemented. This further circumstantiates the approach and decision that were incorporated into the given thesis paper as it reinforces the point that the given approaches used are more preferred and yield better results when combating fake news in the machine learning domain.

TABLE I Comparison of the accuracy of fake news classification using different classifiers


	Naive Bayes	SVM	Log. Reg.	Dec. Tree	NN
Count Vectorizer	0.863	0.891	0.916	0.825	0.499
 F-IDF Vectorizer	0.854	0.928	0.910	0.816	0.499

Figure 9 Results of ML Modelling Comparison on Various Vectorization Methods

2.6 Multiclass Fake News Detection Using Ensemble Machine Learning

Kaliyar et al sets out to use non-binary classification in order to see if it's possible to detect fake news using other prediction methods available to the given problem-domain. In this research project, experiments were conducted using a tree-based Ensemble Machine Learning Framework (Gradient Boosting) with Kaliyar et al choosing various parameters in order to optimize them as well as combining content and content-level features in order to increase the accuracy of detecting fake news. The given ensemble is then compared against various industry benchmarked classifiers commonly used within the given domain in order to compare whether the given ensemble developed out-performs the various other classification methods.

Kaliyar et al decided to perform extensive experiments using a multiclass dataset (FNC-based fake news dataset) which was sourced from Kaggle. The given dataset contained 49,972 instances of fake news within the given dataset containing the columns: Headline, Stance of News Article, Discuss, Disagree. During the experiments most classification models were training using a train/test split of 70/30 of which 70% of the given dataset was dedicated to just training whilst the 30% remaining was dedicated to testing the given classification models.

Kaliyar et al took the following feature extraction techniques into consideration in order to compose the corpus in order for the modelling stage, such feature extraction techniques included:

- Term-Frequency Inverse-Document Frequency (TF-IDF)
- Cosine Similarity Features
- Hand Selected Features
- Word Overlap Features
- Polarity Features using TextBlob
- Refuting Features

Kaliyar et al compared the developed Gradient Boosting Ensemble Classifier against various machine learning benchmark performers, such as: Random Forrest, Multinomial Naïve Bayes, Gradient Boosting, Decision Tree, Logistic Regression, Linear Support Vector Machine as shown in the figures below. From the given experiments it was found that the gradient boosting provides state-of-the-art

results as it was able to achieve 86% accuracy compared to the various other benchmark classifiers. As well as this the Gradient Boosting algorithm was also able to achieve the highest confidence score with TF-IDF feature extraction.

Aspects of Kaliyar et al incorporated various techniques such as using a gradient boosted ensemble in order to detect fake news and feature extraction techniques such as TF-IDF which has also has been incorporated into the given thesis. As Kaliyar et al is able to achieve state of the art results using such approaches this further demonstrates the reasons for these such methods to also be incorporated into the thesis project such as an ensemble using gradient boosting.

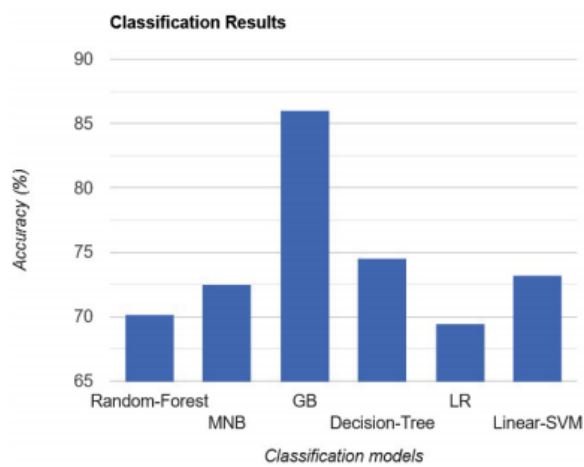


Figure 11 Classification Results using various Machine Learning Algorithms

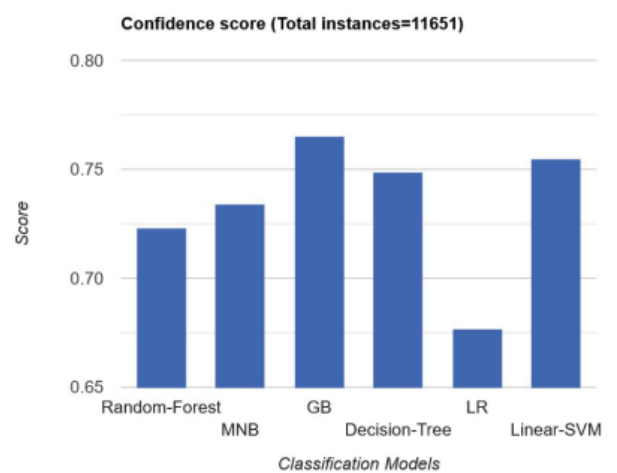


Figure 10 Confidence Score using Machine Learning Algorithms

2.7 Conclusion

This chapter presented a review and evaluation of the literature that investigated the problem of fake news relating to the given domain of machine learning, with various other non-ML approaches also being reviewed as certain characteristics within these particular pieces of literature overlaps with the given thesis project. This allows us to highlight our approach (described) in the next chapter with greater justification and assurance as approaches such as feature selection and classification algorithm choices having already been implemented in previous works covered in the literature review have shown the potential they have within similar or the same domain.

3 Design & Methodology

In this chapter, we will focus on the methodologies used in the natural language processing in order to help achieve our objective. In order to understand the approach used we need to analyse and understand the key individual aspects of the project in a linear fashion. Through-out this chapter there will be a key focus on how the Agile-CRISP-DM approach (mentioned in section 1.4) played a significant role in allowing the flexibility and adjustability required in order to successfully create this project.

Below is a diagram which clearly outlines how the design and methodology of the given thesis will be implemented. The CRISP-DM/Agile approach will be followed strictly with further details mentioned in latter part of chapter 3.

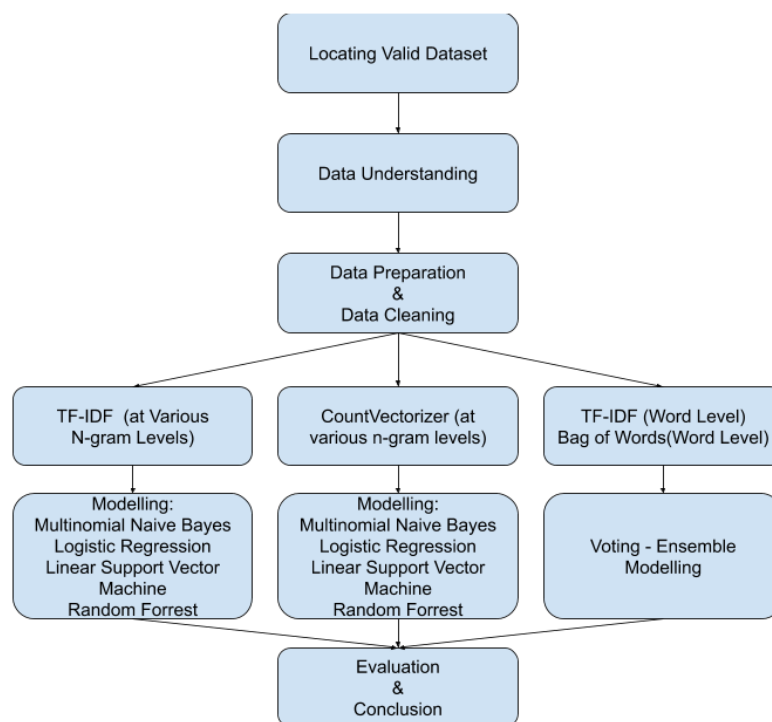


Figure 12 Methodology Outline with Given Approaches Used

3.1 Outline

In this section we will outline and highlight the proposed approach used in order to identify fake news. Our approach will be trained and tested on a given dataset (details of which are below), the project as a whole will contain 4 main aspects: (1) data pre-processing, in which the given dataset is thoroughly cleaned and raised to an appropriate level, (2) feature extraction techniques, In which we utilise the most appropriate vectorization techniques available on the given corpus, (3) feature selection, in which the most suitable columns for the given problem-domain will be selected, and finally (4) classification and evaluation, in which we are able to detect if a given news articles if reliable or not with appropriate evaluation metrics in place in order to measure the robustness of the classifications used.

Within the literature review within section 2 it can be deduced that various approaches can be adopted in order to combat and detect fake news with relatively good results all round. An aspect of

this project which we want to find is which individual approach is most appropriate for this given problem-domain of fake news. In order to do this when it comes to step 2 of the feature extraction techniques there will be 2 main comparisons made between the feature extraction techniques Term-Frequency Inverse Document Frequency and Bag of Words, the same classifications and evaluation techniques will be deployed on the different feature extraction techniques, with the aim of this aspect of the project is to find the most suitable matrix vectorization representation for the given corpus.

After doing much research and reading various research papers there was an interest in seeing how the accuracy of the given classification models varied when using a given ensembling approach instead of an individual classification algorithm. So in order to find and test whether an ensembling approach would be a more ideal approach, as-well as implementing the previous experiment another approach was added where a voting ensemble was to be used with the given classification models on both feature extraction techniques Term-Frequency Inverse-Documents Frequency as-well as Bag of Words in order to find which approach of all was more appropriate.

3.2 Data Source

When initially starting the project using the current specification the lack of labelled fake news datasets became a significant bottleneck as in order to start any sort of CRISP-DM methodology a relevant and suitable dataset was crucial. This bottleneck was due to the given domain as there was a lot of labelled twitter datasets available, but not many labelled news articles datasets. This led to attempting to collect enough labelled relevant data using web-scraping tools in python, this soon became unfeasible due to the amount of data required for the data mining problem as well as pushing the time constraints. After some time searching, an appropriate dataset was located on Kaggle. This given dataset met the given requirements as specified in section 1.5.1.1. More details about the given dataset will be specified below

3.2.1 Fake News Dataset Understanding

The dataset picked was from Kaggle which classified real or fake news. The given dataset contained of around 20380 rows, with 4 different columns, each individual column contained: ID, Title, Author, Text and Label, with the label column containing 0 – representing unreliable or 1 – representing reliable. Only the given text column was analysed in the given project. When searching for a dataset It was important to ensure the given dataset had an equal distribution of reliable and unreliable news articles. This is an important aspect that needs to be considered in the data mining project because in order for the predictive algorithms to accurately and reliably learn from the data sources there needs to be an equal and adequate amount of data within the dataset otherwise the modelling algorithms will become proficient in one type of labelled data over another. In order to prevent this from happening it's vital for the given dataset to have an equal distribution of reliable and unreliable labelled data.

Another key aspect of the dataset that was considered when verifying whether the given Kaggle dataset was suitable or not was the amount of relevant data within the given dataset. The Kaggle dataset originally contained around 20380 but after thorough cleaning and feature selection the amount of useful data for the given domain was roughly around 17000. This amount of data was still satisfactory; but the reason for there being a strong concern on the given dataset containing enough data is because the amount of data the predictive algorithms require depends on certain factors, these include:

- The Complexity of the problem which is being answered - nominally the unknown underlying function that best relates the input variables to the output variable.

- The Complexity and Suitability of the Chosen Learning Algorithm – How well the chosen classifying algorithm used to inductively learn the unknown-underlying mapping function from the given examples during training.

Other reasons for deciding to choose the given dataset also include the word-length as most research papers that were evaluated in section 2 of the literature review mainly focused on social media and more specifically twitter. As twitter only allow 140 per tweet the amount of information available is quite small in comparison to a news article. This aspect of the dataset plays a significant role in the when it comes to modelling as in order for the model to achieve ideal accuracy results the text-length in which the model will be training upon ideally should be long in order to ensure the given classification algorithm is able to learn as much as possible.

3.3 Data Preparation

The data preparation phase is a fundamental stage within the CRISP-DM methodology due to the impact and importance it could potentially have on the analysis techniques chosen during the modelling phase. A key objective of the given project is to accurately detect fake news, this objective could be argued relies on the cleaning process, as without high quality cleaned data the analysis techniques will not be able to clearly find useful patterns which could potentially lead to low performance and poor quality outputs (NA Karur, 2015).

Data preparation is a key step in the CRISP-DM methodology as well as reducing noise before any machine learning can be done. In this aspect of the methodology our main task is to raise the quality of the data to the level required by the analysis techniques that have been selected. The dataset chosen (stated in section 3.1.1) was found to contain many errors which needed to be amended before the process of analysis and predictions being made. Some of the errors that were contained within the original dataset include various empty or null inputs within certain columns, as well as certain features within the dataset that wouldn't add any benefit to being used in the analysis techniques. The whole process of preparing the dataset was extensive because as well as removing any nulls or imputes the dataset was conformed to the given requirements of the project, this includes removing any non-English news articles, as well as removing text articles with a considerably short length and much more. The whole cleaning process will be further detailed below.

3.3.1 Feature Selection & Minimization

This aspect of cleaning dealt with certain features within the dataset that contributed to any of the useful data that was going to be used in the analysis techniques. These given un-useable features only ever increase the search space for the given analysis techniques and as a result it was decided to remove them for the dataset entirely. There were a few attributes within the given dataset that fell under this category such as the ID attribute as well as the Author attribute. The reason for not including the author within the analysis stage was because the authors name doesn't contribute any lexical or linguistic analysis to the news article that they wrote in helping the analysis techniques find useful patterns for whether it's reliable or not. This not only reduced the search space allowing for useful patterns to be found but also improves the efficiency and computational intensity when it came to execute the analysis techniques.

3.3.2 Data Cleaning and Consistency

As previously mentioned after feature selection we are left with the desired attributes within the dataset that benefit and contribute to useful patterns when executing the analysis techniques. But in order to fully utilise these particular attributes they need to be at a high data quality beforehand. The data within these attributes contain many errors, so the given dataset needs to be thoroughly checked to see if any abnormalities exist within the given dataset.

Missing values within the attribute columns were removed completely, as with the given attribute columns the missing data cannot be assumed in the same way numeric data can sometimes be dealt with.

After removing any null or missing values within the attribute columns within the dataset the next step of action was to lowercase all the text within the attributes. This is common practice within most Natural Language Processing tasks and helps the text to all have the same consistency. More importantly the reason for having all the words within the attributes the same case it in order to ensure that all the same words are mapped correctly onto the given matrix during vectorization (further details explained below). If the given text doesn't contain the same consistency throughout the same two words with slightly different variation of capital letters will be mapped completely differently to one another leading to different outputs which will ultimately results in a reduced accuracy overall. In order to ensure and avoid such an occurrence the given text within the dataset all have the same consistency throughout.

3.3.3 Removing Non-English Words

As previously mentioned in section 1.5.1.1 the data source should only contain English news articles so in order to ensure the dataset is meeting the requirements all non-English words are to be removed in order to ensure consistency and allow for words remaining which has meaning within the given text attribute. Another major reason as to why non-English are being removed from the corpus is because the words carry no meaning; as they increase the search space making it more difficult to find beneficial patterns causing it to reduce the accuracy during the analysis phase of the given project. This cleaning process massively benefits the given analysis techniques used as it increases the interpretability of the given corpus allowing for the classifiers to understand the text a lot more quickly in a numerical sense.

As previously stated above (section 1.5) the preferred development language used is python due to various reasons. During this data cleaning stage there were a few routes in order to remove any non-English words but ideally the most preferred method would be the least computationally expensive and efficient. This leads to using the library langdetect as it searches the given body of text in order to find any text within the news articles which didn't meet the English language. Once located the given news article which mainly contained non-English words were removed. This approach was not only preferred due to the previously stated reasons but was also time effective in getting the task completed.

3.3.4 Removing Punctuation

The main reason for removing punctuation is because it is simply not part of the English language. As the classification methods will be analysing a vectorized version of the given corpus maintaining the given punctuation would only server to hinder the progress. This is because the given classifier methods would read two of the same sentences completely differently if one sentence contained some punctation over another. The only NLP domain in which punctuation could serve to aid the progress and textual understanding would be in sentiment analysis in which punctuation could help aid the feeling the writer is trying to give off through their text (Gagandeep Singh, 2019), but as no sentiment analysis will be conducted during this project the punctuation will be removed due to the aforementioned reasons stated above. Another reason for removing punctuation is because it allows us to reduce the training data leading to reduced computational intensity as-well as punctuation not adding any extra information.

When implementing the given code in order to ensure the given punctuation was removed correctly, it was found the given library that was chosen didn't achieve the desired output after the given corpus

had the punctuation removed, this is because whenever a punctuation mark was removed there was a space left, this space when it came to vectorization was interpreted as an individual word itself which created some confusion during the analysis phase. In order to combat this error I created a given method in python which used the library `nlk.tokenize.word_tokenize` to tokenize the given corpus in which the punctuation was then removed once this had happened the words were combined together as originally desired. This method is longer than previous library attempts but achieves the desired output which is a key factor to consider.

3.3.5 Removing Stop Words, Tags and Special Characters

A stop word is a commonly used word that is mainly used as a connector word in order to aid the flow of the given sentence, stop words do not add any value to the given news article in order to help the given analysis techniques in order to identify whether a given article is reliable or not (Usman Malik, 2020). The main reason as for removing stop words as-well as the previously stated reasons is because various vectorization techniques weight each given word based on their occurrence within the given corpus, due to stop words being extremely common within a given corpus these such words would hold a greater weight against an actual meaningful word; even though they are considered as connector words and don't carry any meaning. As well as removing stop words, the decision was also taken to remove tags and any special characters also due to such words not contributing anything to the analysis phase.

3.3.6 Lemmatization

Lemmatization essentially analyses the given vocabulary of each word within the given corpus in order to find the given words morphology linguistic makeup. The objective of lemmatization is to remove the form off the endings of the given words in order to return the base or root origin of the words. This aspect of finding the root of the word is known as lemma. Due to lemmatization being a more dictionary orientated, it requires a greater corpus size and greater computational linguistic power to achieve better resolutions.

Stemming is also an approach attempts to achieve the same outcome as lemmatization as it reduces the inflectional form of each word into a common base route/stem. Stemming however goes about it a different way as to lemmatization. In this approach the stemming algorithm works by cutting off the end or the beginning of the word, taking only into account a list of common prefixes and suffixes which all words can share.

The reason for choosing lemmatization over stemming is due to, stemming algorithm doesn't necessarily work all the time as some word do not have a give root leading to overstemming and understemming (Hunter Heidenreich, 2018). As a result of this lemmatization was chosen as it always resolves the words to their respective dictionary form.

3.4 Feature Extraction Techniques

In this section of chapter 3, we will focus on the more traditional feature extraction technique methods chosen in natural language processing such as Term Frequency-Inverse Document Frequency and Count Vectorization at various N-grams. These given techniques will serve as the basis for the analysis phase, as in order for the for any machine learning to be executed on the given corpus the text in its raw form would not be valid. In order to present the given text in a valid form a mathematical representation of the text is required.

3.4.1 Term-Frequency Inverse Document Frequency

As explained previously, the given text needs to be represented in a manner that allows it to be interpretable and meaningful to the machine learning algorithm. There are various techniques that can be adopted in order to go about this, one such way is TF-IDF.

TF-IDF is a statistical measure that evaluates how relevant a given word is to a document, in a collection of documents. The importance of the given word increases proportionally to the number of instances the given word appears in the document but is offset by the number of instances of the word in the given corpus. TF-IDF can be separated into two parts, the first part regarding term frequency and the second part regarding inverse document frequency.

In a mathematical sense TF-IDF arranges the documents into a multidimensional area which is directly proportional to the vocabulary size. However, in TF-IDF the documents are represented as feature vectors, in which the given elements are the product of TF and IDF. Therefore, each document d corresponds respectively to a vector in the given space, where each vector in d is the weighted frequency of a given word in the document which is then normalized by the frequency of the given word in the dataset. This explanation is broken down below to help understanding.

- **Term Frequency** – This aspect of the vectorization technique measures how frequently a given term occurs in a document. Since each corpus has a different length it's probable that the greater the length of the document the greater the frequency of use of the given word. In order to account for this the Term Frequency is often divided by the given document length in order to normalise the varying lengths of the documents (Jocelyn D'Souza, 2018).

$$TF(t) = \frac{\text{Number of times term } t \text{ appears in a document}}{\text{Total number of terms in the document}}$$

- **Inverse Document Frequency** – This aspect measures the importance of the term. During Term Frequency all terms are as equally important, but during this stage of TF-IDF a weight is given to each term. This allows us to know what words in the given corpus are more important in the given text compared to others. (Jocelyn D'Souza, 2018)

$$IDF(t) = \log_e \left(\frac{\text{Total number of documents}}{\text{Number of documents with term } t \text{ in it}} \right)$$

Finally, the TF-IDF score is calculated by:

$$TF - IDF \text{ score} = TF * IDF$$

This text embedding approach was the more preferred of the two embedding techniques due to the fact it takes into consideration the relevance of the given terms throughout the given document as well as how efficient TF-IDF performs.

3.4.2 Bag of Words Approach

Another more simplistic approach available which will be used in the given project is the Bag of Words (BoW) approach. This approach is a more simplistic method for extracting features from a given text than the previously mentioned TF-IDF approach.

The bag of words techniques is a simple and flexible technique used for processing text documents in machine learning. In this approach a sparse matrix is created which is directly proportional to the number of terms within the given document. This matrix is sparse due to many of the vectors being 0's within the given matrix. The bag of words model discards all the other information in the given word and only considers the occurrence of that word in the given document, meaning no spatial information of relevance weighting of the word is either calculated or taken into consideration. The intuition behind this is that if the documents have similar content, they are therefore similar to each other.

In the given code of the project the Bag of Words approach implements the Count Vectorizer. The Count Vectorizer generates an encoded normalised vector that contains the length of the entire vocabulary combined with the frequency of each word in the given document.

3.4.3 N-grams Analysis

N-grams analysis is the analysis of contiguous sequence of N items within the given document. The N refers to the number of various different combinations of items, such as phonemes, syllables, letters, words, character, byte etc. Most commonly used N-gram levels in NLP are character level and word level.

In this thesis we apply varying levels of character level, bi-gram level and word level N-gram analysis during both TF-IDF and Bag of Words feature extraction in the approach to detect fake news. The reason for this is because at different N-gram levels characters and words take up a new meaning within a more or less restricted context. This could potentially have an impact on the how well the given models are able to detect fake news. As using n-grams analysis allows us to see the actual relationship various words have with one-another whether it's examining words that occur after one-another, or words that tend to co-occur within the same document. N-gram analysis adds an extra layer of textual understanding during the feature extraction technique.(Text Classification Using the N-Gram Graph Representation Model Over High Frequency Data Streams, John Violos et al, 2018)

3.5 Modelling

The models that have been chosen to classify the given text represented in the TF-IDF and BoW vectorization techniques at various N-grams include, Multinomial Naïve Bayes, Logistic Regression, Linear Support Vector Machine, Random Forrest and Voting Ensemble. These given classifiers will be detailed below with a brief introduction into the given classifiers and why they are considered suitable for the given domain.

Before doing any sort of action is taken, it's always a good idea to also shuffle the data the reason for shuffling the given data is to ensure the given data has a reduced variance over the given dataset, as well as this it also allows an equal dispersion of results; this spread allows for the given dataset to contain an equal distribution of results resulting in no bias.

3.5.1 Multinomial Naïve Bayes

Naïve Bayes is a simple classifier that classifies the given data based on probabilities of events. The main domain of implementation of MNB is within the NLP and more specifically text-classification. Though MNB is considered a simple algorithm, it performs relatively well within the given domain as well as being more efficient and less computationally intense.(Multinomial Naïve Bayes for Text Categorization Revisited, Ashraf M. Kibriya et al)

Naïve Bayes assumes, naively the independence of features which may or may not necessarily be true based on the given domain. In a Naïve Bayes classifier, the probabilities of the individual features are multiplied together in order to obtain an overestimation of the probability. This overestimated result decides the class of the given input. Due to this overestimation of the given features you would assume that the MNB algorithm would not perform well as many inputs especially in the NLP domain are dependent upon one-another, however it may perform well as even with strongly dependant inputs as the dependant inputs within the algorithm cancel each other out(Comparison of Various Machine Learning Models for Accurate Detection of Fake News, Karishnu Poddar et al, 2019 I-PACT). The figure below represents a mathematical formula what was previously mentioned.

Linear classifiers such as Naïve Bayes are very simple as well as being highly efficient and computationally inexpensive. The fundamental concept of Naïve Bayes is based off Bayes theorem and is only called Naïve due to the algorithm assuming the given features within the dataset are not dependent upon on-another. Even still Naïve Bayes algorithms tend to perform well under such assumptions as previously seen as part of the results during the literature review in section 2.

3.5.2 Logistic Regression

Logistic Regression is a Machine Learning algorithm which is used mainly in classification domains, Logistic Regression is a predictive algorithm which is based on the concept of probability. Logistic Regression is similar to Linear Regression but utilises a more complex sigmoid function as well as mainly working with dichotomous dependant variables. The given sigmoid function is limited to between 0-1. The given classifier works with binary data, where either the event the given outcome is either fake or not fake for our given domain.

Simplified, Logistic Regression is similar to Linear Regression, but instead of fitting a straight line or hyperplane, the Logistic Regression classifier uses the cost function to normalize the output to between 0-1. (Ayush Pant, 2019)

Once the given inputs have been normalized between 0-1 using the sigmoid function a threshold value can be calculated which dictates to which binary class the given inputs belong to. This process apprehends a vector of variables and evaluates coefficients or weights for each input variable and then predicts the class of the stated news article with the threshold value in mind also.

The reason for using Logistic Regression is because it has multiple benefits such as features within the given text are able to be dependent upon each other (unlike Multinomial Naïve Bayes). As-well as this Logistic Regression performs better when using a larger dataset. This favours the given project as with over 20,000 news articles within the original dataset there is a lot of data to be analysed. Finally, various research papers within the given domain used the Logistic Regression algorithm within their projects and it was found they consistently performed well due to various reasons.

3.5.3 Linear Support Vector Machines

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for classification and regression domain analysis. The SVM algorithm works by transforming your data then based on the given transformations it finds an optimal boundary known as a hyperplane between each potential class output. This technique is considerably more complex and intense than the previously mentioned Logistic Regression and Multinomial Naïve Bayes.

The support vector machine (SVM) used projects each feature into a given vector space, where given vectors are used to determine the distance between the separation between the binary classes. The purpose of this is to generate a hyperplane that is used to separate the training data as far as possible. This hyperplane allows the documents from various classes to be separated as far as possible.

The reason for deciding to use a Support Vector Machine was because of its ability to perform and generalise well given high dimensional spaces as well as being highly accurate and also getting trained in a lesser time compared to other complex algorithms available.

3.5.4 Random Forrest Classifier

A Random Forrest is a machine learning algorithm which is of an additive in nature meaning that It makes predictions by combining decisions from a sequence of base models (Turi Machine Learning Platform User Guide). This can be written in a more formal manner such as:

$$g(x) = f_0(x) + f_1(x) + f_2(x) + \dots$$

Figure 13 Mathematical Representation of the additive sum of all the slave predictors

The mathematical equation depicts the technique used in Random Forrest where the algorithm utilises each base classifier (Decision Trees) in order to obtain better predictive performance. This aspect of the Random Forrest algorithm is called model ensembling. This approach allows the weak learning models within the Random Forest to make a prediction then aggregating such predictions together in a process known as wisdom of the crowd allows the Random Forrest to produce a greater performing output. This is because a large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models. This is because the decision trees protect each other from their individual errors (as long as the decision trees don't all error in the same direction). This is because even though some trees could potentially be wrong, many other trees will be correct allowing for the Random Forrest algorithm to perform well. (Understanding Random Forrest, Tony Yiu, 2019)

Using Random Forrest for predictions within this problem domain has many advantages such as its immunity to overfit. As well its ability to handle large datasets with a considerably high dimensionality. Even though the given algorithm is the most computationally intense due to the large number of weak classifiers (Decision Trees) taking some time to compute and output the classifier on the whole achieves in other NLP domains a relatively high accuracy due to the previously states reasons.(Understanding Random Forests Classifiers in Python, Avinash Navlani, 2018)

3.5.5 Ensemble Modelling

An Ensemble analysis method is a machine learning technique that combines several base models in order to produce one optimal and strong predictive model. An ensemble is a meta-algorithm that combines several weaker machine learning techniques in the hopes of decreasing variance (bagging), bias (boosting), or to improve the predictions (stacking) . Within the Ensemble machine learning technique there can be a further sub-divide as there are different types of Ensemble techniques such as (Ensemble Learning to Improve Machine Learning Results, Vadim Smolyakov, 2017):

- Sequential Ensemble (aka Boosting) – This Ensemble technique is where the base learners are generated sequentially. The purpose of this is to exploit the dependence between the base learners. Sequential Ensemble methods can boost their performance by weighting previously mislabelled examples with a greater weight.
- Parallel Ensemble (aka Bagging) – This Ensemble technique is where the base learners are generated in a parallel manner (in some sense similar to the Random Forrest Technique mentioned above in section 3.4.4). The purpose of this is to exploit the independence between the base learners and also showcase the individual performance of the slave models; this can reduce the overall error of the slave models as the average can be taken.

- Voting (aka stacking) – This Ensemble technique involves building multiple models (usually heterogenous) and the calculating the given average/mode in order to then combine the given predictors.

With any Ensemble Machine Learning technique, it is possible to use both homogenous and heterogenous slave models in order to make up the actual Ensemble technique. But regardless in order for the given Ensemble algorithm to work the slave learners must need to be as accurate as possible in order to maximize the accuracy overall.

Within the given thesis project, it was decided the author would use a voting classifier technique within the given Ensemble Machine Learning algorithm. The voting classifier allows for predictions from various heterogenous which then utilises a voting system to combine the individual slave predictors in order to deduce a final outcome. The diagram below helps further understanding.

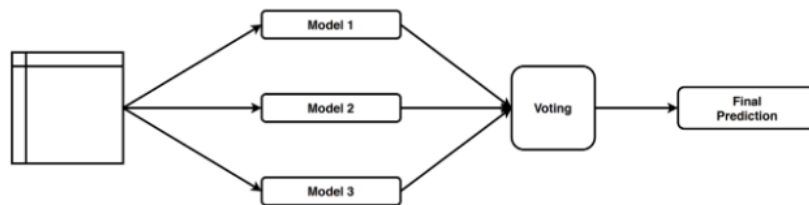


Figure 14 Ensemble Using a Voting Classifier

During the process when developing the Ensemble Machine Learning classifier various other predictions alternatives were considered, with the two main approaches considered within the Ensemble Machine Learning algorithm being previously mentioned which were:

- Sequential Ensemble Approach (Boosting)
- Parallel Ensemble Approach (Bagging)

The reason for using the given voting approach was mainly due to the ability to utilise various individual slave predictors not only to analyse the variance within the results but to also increase the given overall accuracy of the performing ensemble classifier. Within voting the performance of this technique in all competitions is massive and remarkable. (Stacking – A Super Learning Technique, Guru Charan, 2017)

3.6 Model Evaluation Techniques

In order to assess the effectiveness of the given classification models over the unseen data and numerically assess their generalization ability, the evaluation techniques are used. These techniques are used in order to help find whether or not the given models used do achieve a relatively good score and if so, how reliable the given results are. As well as this evaluating the given models helps ensure we know if the given models are either overfitting or underfitting. Overfitting occurs if the given predictive model fits the training results too well, and underfitting occurs when the given predictive model is not able to find any trend patterns over the training data at all meaning it doesn't train well. Both of these outcomes result in a poor effectiveness of the given model when applied to unseen data. In this section we will describe what evaluation techniques were used as well as understanding of the given approaches and justification.

3.6.1 K-Fold Cross Validation

K-Fold cross validation is an evaluation technique used in which the dataset is randomly split into n mutually exclusive subsets of approximately equal size. The given predictive model is trained and tested n times, with each cycle the predictive model is trained and tested a different subset of the given dataset is used until the whole of the given dataset is both trained and tested. Within the given stratified K-fold cross-validation the folds are stratified in order that they contain approximately the same proportions of labels as the original dataset. This is important as stratification with K-Fold seeks to ensure its representative of all strata of the data, reducing any bias within the evaluation stage and leading to a more representative and accurate outcome.

Configuration of K during this process is an important aspect when setting up the K-fold Cross Validation as the value of K must be chosen carefully as otherwise a mis-representative value of K could potentially lead to a mis-understanding of the skill of the model, such as a score with a high-variance or potentially high bias. Usually the value of K is between 5-10 but there is no formal rule of how high or low you can go but bear in mind as the value of K increases, the difference in size between the training set and resampling subsets becomes smaller. As this difference decreases, the bias of the technique becomes smaller (A Gentle Introduction to K-Fold Cross-Validation, Jason Brownlee, 2018). Implementation of K-Fold Cross Validation is a computationally intensive process as the given algorithm has to be rerun constantly K number of times but the benefits of implementing such an approach is that there is a reduced bias overall for the given model as well as this every data point gets tested once and is used in training $K-1$ times. Also, as K increases the variance of the resulting estimate is reduced.

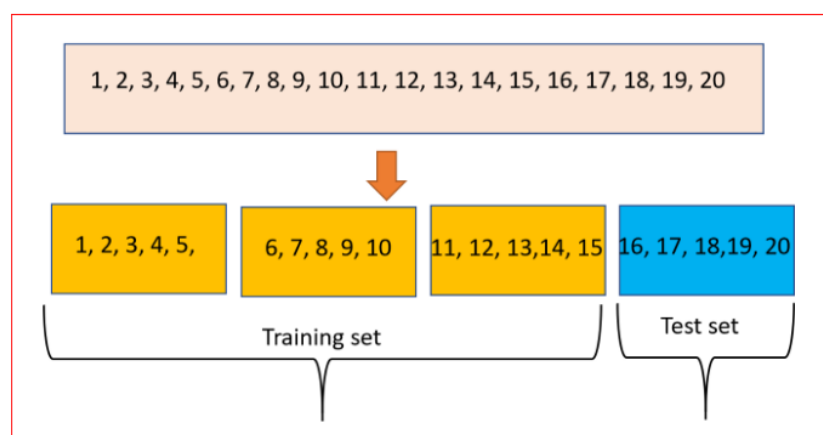


Figure 15 K-Fold Cross Validation

3.6.2 Confusion Matrix & Key Terms

A confusion matrix is a common technique used for summarizing the performance of the given classification algorithm. Classification accuracy can sometimes be very misleading especially if you have an unequal number of observations within each class or if your dataset contains more than just binary outcomes. The purpose of a confusion matrix is that it's able to achieve a better idea of the overall performance of the given classifiers and in particular quantify what the given algorithm is getting right and what it's getting wrong.

A confusion matrix table consists of four outcomes:

- True Positive – The True Positive represents the proportion of cases that were correctly identified for the given classifier.
- False Positive – The False Positive rate is the proportion of negative cases that were incorrectly classified as positive for the given classifier.
- False Negative – The False Negative is the proportion of positive cases that were incorrectly classified as negative for the given classifier.
- True Negative – The True Negative is the proportion of negative cases that were classified correctly for the given classifier.

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

Figure 16 Abstract Confusion Matrix

(Understanding Confusion Matrix, Sarang Narkhede, 2018)

Recall/Sensitivity

Recall is also known as the True Positive rate and is the measure of all the positive classes that were predicted correctly by the given classifier. Recall is also referred to in some texts as the completeness of the given model. (Understanding Confusion Matrix, Sarang Narkhede, 2018)

$$\text{Recall} = \frac{TP}{TP + FN}$$

Precision

Precision is essentially the proportion of the predicted positive cases that were correct, this is calculated using the given equation below. (Understanding Confusion Matrix, Sarang Narkhede, 2018)

$$\text{Precision} = \frac{TP}{TP + FP}$$

F1-Score

F1 score is defined as the harmonic mean between the precision and the recall (as stated above). It is used as a statistical measure to rate the performance of the given models. Put simply the F1 score is

a judgement for the given classifier's performance with the Precision and Recall being the two considered factors. (Understanding Confusion Matrix, Sarang Narkhede, 2018)

$$F - measure = \frac{2 * Recall * Precision}{Recall + Precision}$$

Specificity

Specificity also referred to as the True Negative Rate is the proportion of actual negatives that were correctly identified within the given classier.

$$Specificity = TN / (TN + FP)$$

3.7 Final Remarks

In this chapter, we presented the methodology used in the thesis in order to detect fake news with theoretical understanding and reasoning for the approaches chosen. We explained in a linear CRISP-DM fashion, the process required from the dataset chosen and cleaning required to the feature selection and feature extraction techniques such as Term Frequency – Inverse Document Frequency and Bag of Words Count-Vectorization. By using various N-gram analysis on the given features we utilised the given classification algorithms chosen which include Multinomial Naïve Bayes, Logistic Regression, Linear Support Vector Machine, Random Forest and finally Ensembling.

4 Implementation & Justification

4.1 Software Development Language

This section will outline the two main languages considered for the given project with justification for the final choice:

4.1.1 C++

When deciding on a potential development language to choose from C++ was a seriously considered due to various reasons. With 44% of data scientists and machine learning developers using C++, coming in second (Timo Rohner, 2018). Apart from the author feeling confident in their ability to complete the given proposed project within the given time-frame available due to their prior experience developing various software project within the given language, C++ is a development language which allows for a greater emphasis and prioritization upon efficiency and due to the given project dealing with potentially large sets of data, well within the tens of thousands and combined with the limited computational capacity on which the given project is to be developed; developing the given project in C++ would allow for faster run-time code due to the given language being more compact and compiled as a language.

Furthermore, machine learning orientated libraries such as MeTA (MeTA, 2020), MITEE, text2vec (text2vec, 2016) and many more are libraries commonly used within Natural Language Processing allowing for the facilitation of computational linguistic and lexical analysis. Such libraries would most definitely make the given thesis project feasible within C++ as-well as allowing for the run-time code to maintain a high-level of efficiency and less computationally intense.

4.1.2 Python

Another key player when considering the development language to choose from was python - with over 57% of data scientists and machine learning developers using it and 33% preferring it over other languages for development (Medium, Himani Bansal, 2019).

The main reason for considering Python for the given project was due to how well supported the development language is for the thesis project, with over 57% of data scientists and machine learning developers using it (Medium, Himani Bansal, 2019). Due to the backing python has within the data science domain and also within the machine learning domain with the various different libraries available. Libraries such as NLTK and spacey that were key and useful through-out the given thesis will be detailed later on in the chapter. Another aspect of the given language which was considered was how popular the given language is, allowing for any changes if needed in the unpredictable future. Choosing such a language could potentially future proof the given thesis regarding whether any objectives change.

The author had not had much experience with python before delving into the given project but due to how concise and readable the given language is which the simplicity surrounding such complex implementations the author was confident that the project would be complete and meet all the aforementioned objectives.

After weighing up various aspects of the pros and cons of both development languages as already stated and mentioned above the author came to the decision to pick python development language over C++, due to the previously stated reasons but also due to the fact that it would incorporate significantly better with the IDE chosen to develop the artefact within. Further details about the IDE chosen will be stated below.

Efficiency is bs in this report and in C++ its more long anyways. Python uses a lot of c therefore not that much slower realistically.

4.2 Integrated Development Environment

Once it had been decided to choose Python as the preferred development language to create the artefact the next software decision to consider was the IDE. Due to not having much experience in Python the author did not have clear IDE in mind. When researching and considering various IDE suitable for the given domain the two main IDE that were suitable for the given thesis considering the timeframe and usability were:

4.2.1 Jupyter Notebook

Jupyter is an interactive web-tool which is essentially a computational notebook. This computational notebook allows for the combination of various aspects of a research project. Jupyter allows for researchers to combine software code, computational output and explanatory text and visualisation sources within a single file. Jupyter notebook has exploded in popularity recently especially within the data science field as according to the code sharing site GitHub counted 2 million public Jupyter Notebooks from September 2018, up from 200,000 in 2015 (Jeffery M.Perkel, 2018).

A key aspect of Jupyter Notebook which would help benefit with the thesis is the ability Jupyter Notebook has to explore and understand the given data and code in a very convenient and concise blocks, Jupyter allows users to view the results of the code without the dependency of the other parts of the code. Another high point for Jupyter is the data visualisation, Jupyter Notebook supports visualizations and includes rendering of some of the data sets like graphics and charts.

4.2.2 PyCharm

PyCharm was another considered IDE for the given project. PyCharm is a powerful IDE which is used for mainly Python applications development. PyCharm has various key features which would definitely enhance the development aspect of the given artefact including an Intelligent Code Editor, this feature ensures that high quality, error free code is written to ensure an easier coding experience is had by the user. As well as Jupyter Notebook PyCharm also allows you to visualize data in a graphic form allowing for meaningful data understanding to be translated more easily over to the reader.

After comparing the two IDE's against one-another the author decided to choose Jupyter as the main IDE for the thesis due to the aforementioned reasons above, but for also having more experience within Jupyter Notebook allowing for there to be more confidence and less adjustment time required.

4.3 Key Libraries

One key aspect of developing the given artefact with Python was to utilize the plethora of libraries available within the NLP and Machine Learning domain. Access to libraries ensures standards to be maintained and reduce time spent completing tasks. In this sub-section we will mention a few key libraries that were used when developing the given artefact.

4.3.1 NumPy

NumPy (NumPy, 2019) is a fundamental library within Python which is mainly focused on scientific computing. NumPy focuses on aspects such as a powerful N-dimensional arrays object and matrices, as-well as adding a large collection of high-level mathematical functions to operate. As-well as computing various mathematical functions the given NumPy library calculates the given complex operations with a greater level of efficiency, allowing the library to utilize a smaller memory footprint compared to Python's own methods. The NumPy library was used throughout the given project at

various aspects of the CRISP-DM methodology from cleaning, feature extractions and various other places.

4.3.2 NLTK

NLTK (NLTK 3.5 Documentation, 2020) is one of the leading libraries used within Python to work within Natural Language Processing. It provides easy access to over 50 corpora and lexical analysis resources such as WordNet, and Tokenization and Stemming and Lemmatization and various more within the given toolkit. Such features within NLTK are considered key aspects of various NLP orientated projects including this given thesis.

NLTK has been considered “a wonderful tool for teaching and working in computational linguistics using Python” (Ambika Choudhry, 2019). Overall, the NLTK library is considered a key library which without would have made completing the given artefact for the project potentially a lot more difficult and inefficient.

4.3.3 Scikit-Learn

Scikit-learn is another important library within Python, which mainly focuses on machine-learning with access to a broad-range of clustering, regression, and other-various classification algorithms. When utilizing the given Scikit-Learn library the author not only used the library for machine learning, but for also implementing the two chosen feature extraction techniques being compared against one-another: TF-IDF and Bag of Words approaches (Scikit-learn, 2007). Both techniques were implemented using the given Scikit-Learn library. Scikit-Learn is built on-top of various pre-existing libraries within the python; Such integration into various common libraries allows for much easier integration between all the libraries

5 Results & Evaluation

In this chapter, we detail the results and experimental evaluation we conducted in the given artefact, with detail discussion about the given results. Our main goal of this chapter is to understand the results and why the given results achieved the outcome they did, as well as assessing how effective the given classifiers are at distinguishing reliable and unreliable news from each other.

5.1.1 Experimental Reiteration

The given setup of the given project was split into 3 main experiments in which various different approaches were employed in order to find the most appropriate approach available for the given domain. More details of reasoning of the experimental outline can be found in section 3.1.

A brief explanation of the given experimental set up will show the given thesis split into 3 separate experiments in which **experiment 1** will focus on Term Frequency – Inverse Document Frequency feature extraction, **Experiment 2** will focus on Bag of Words feature extraction and **Experiment 3** will focus on both these feature extraction techniques but using a voting ensemble. This experimental set up means that there will be 15 individual experiments which will be conducted based on the methodology outline which can be found in section 3.1

5.2 Experiment 1: Results & Evaluation

5.2.1 Word Level Analysis

Table 1 demonstrates the performance of the given classifiers with the given feature extraction technique chosen.

Table 1 Results Obtained from TF-IDF feature Extraction at Word-Level Corpus Analysis

Classification Type	Accuracy (%)	K-Fold Accuracy (%)	Precision %	Recall %	F1-Score %
Multinomial Naïve Bayes	76.1	73.3	83	76	73
Logistic Regression	94.0	93.3	94	94	94
SVM	95.4	94.7	95	95	95
Random Forrest	83.9	82.3	85	84	83

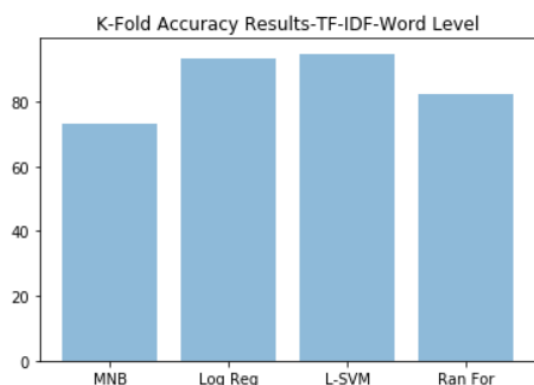


Figure 18 K-Fold Accuracy, TF-IDF

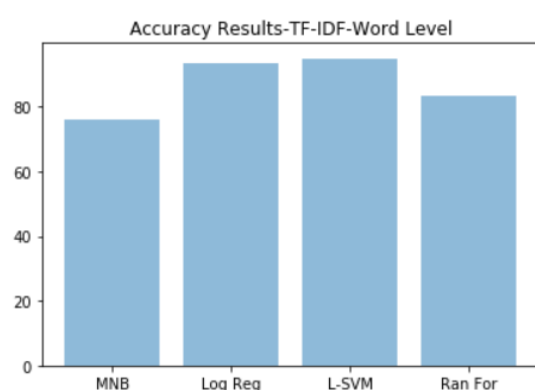


Figure 17 Accuracy Results, TF-IDF, Word-Level

The results from implementing this feature extraction technique upon the given corpus shows that the baseline classification algorithms Multinomial Naïve Bayes performed overall the least impression with an overall accuracy of 76.1% on an individual run as well as 73.3% when run under 5-fold cross validation. Then the Logistic Regression algorithm performed the second best overall obtaining 94% accuracy on individual runs as well as achieving 93.3% on 5-fold cross validation. Thirdly Random Forrest algorithm performed with the recorded output of achieving 82.3% on individual runs as-well as obtaining 82.3% on 5-fold cross-validation. SVM was able to perform the best achieving 95.4 individual accuracy and 94.7 K-Fold average.

An interesting point of observation to be made, the given results at least for both SVM and Logistic Regression classification the given models achieves noticeable better results compared to the various other algorithms chosen, at-least for the N-gram analysis at word-level obtaining at least 14% greater accuracy on individual runs and at least 15.% greater results during K-Fold validation. As well as the accuracy, SVM and Logistic Regression achieves greater Precision, Recall and F1-Scores across the board pitting it to be the given classification algorithms of this domain for the given specifications chosen. Overall SVM is the favoured algorithm with Logistic Regression closely following in 2nd

5.2.2 Bigram Level Analysis

Table 3 below shows the results with the TF-IDF feature extraction technique chosen at Bigram-Level analysis.

Table 2 Results Obtained from TF-IDF feature Extraction at Bigram-Level Corpus Analysis

Classification Type	Accuracy (%)	K-Fold Accuracy (%)	Precision	Recall	F1-Score
Multinomial Naïve Bayes	82.1	89.8	86	82	81
Logistic Regression	86.8	82.2	89	87	86
Support-Vector Machine	94.4	93.6	95	94	94
Random Forrest	87.7	87.2	88	88	88

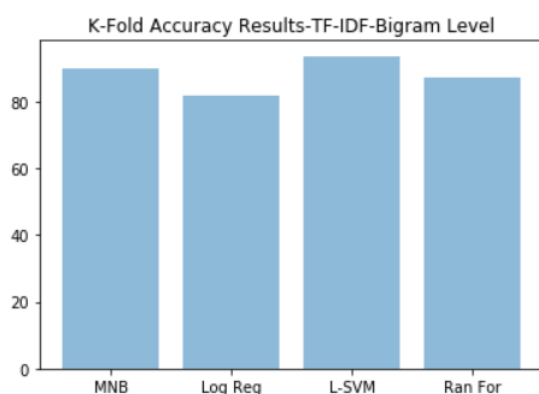


Figure 20 K-Fold Accuracy TF-IDF, Bigram Level

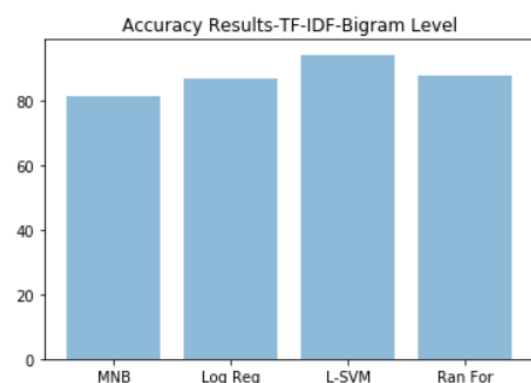


Figure 19 Accuracy TF-IDF, Bigram Level

The figures shown within table 2 above show that as at the Bigram-level SVM obtains the greatest Individual accuracy at 94.4%, and 93.6% during the K-Fold cross validation placing it first above all the other classifiers, with Logistic Regression achieving 86.8% accuracy on individual runs, with 82.2% accuracy during K-Fold Cross-Validation. Multinomial Naïve Bayes achieved the third place as it obtained 82.1% on individual runs, and 63.8% on K-Fold cross validation. Random Forest achieves 87.7 individual accuracy with 87.2 K-Fold average.

When observing the given results, it's shown that there seems to be a noticeable disparity between the individual runs compared against the K-Fold within MNB. This disparity could be potentially due to various reasons such as the given train/test split chosen might not favour the given MNB algorithm. Due to the reliable nature of K-Fold and repeatedly running the given test numerous times across various different sections within the given dataset, K-Fold is more-likely to be accurate and reliable with their overall outcomes of the given classifiers.

5.2.3 Character Level Analysis

Table 3 Results Obtained from TF-IDF feature Extraction at Character-Level Corpus Analysis

Classification Type	Accuracy (%)	K-Fold Accuracy (%)	Precision	Recall	F1-Score
Multinomial Naïve Bayes	84.2	83.7	85	84	84
Logistic Regression	92.3	91.6	92	92	92
Support Vector Machine	93.3	93.0	93	93	93
Random Forrest	85.2	83.9	85	85	85

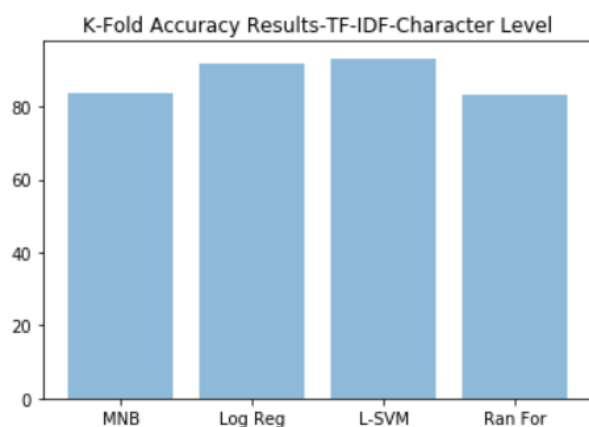


Figure 21 K-Fold Results, Character Level

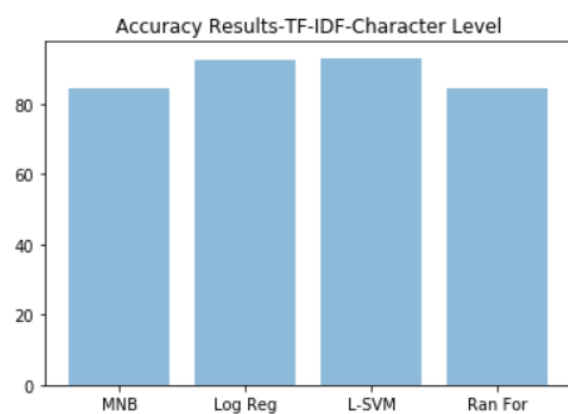


Figure 22 Individual Accuracy, Character Level

The results shown in table 3 seem to show that there is generally a good range of accuracy and K-fold results across the board from all chosen classification algorithms. MNB algorithm achieves 84.2% accuracy with 83.7% K-Fold, Logistic Regression achieves 92.3% accuracy and 91.6 with K-Fold and Random Forrest achieves 85.2% accuracy with 83.9 in K-Fold. Overall, the preferred classification algorithm would be SVM due to the given results achieved for the given dataset with 93.3% individual accuracy with 93.0% K-Fold average.

5.3 Comparing obtained Results Against Published Works

5.3.1 Word-Level Comparative Analysis

Source of Results	Multinomial Naïve Bayes (%)	Logistic Regression (%)	SVM (%)
Tanvir et al (2019)	89.06	69.47	89.34
Given-Thesis Results	73.3	93.3	95.4
Poddar et al	85.4	91.0	92.8

Table 5 Comparative Results for Similar Published Experiments from Poddar et al

If we compare these results to various other research papers within the similar domain such as Tanvir et al (2019), as well as Poddar et al (results are displayed in the table). The given research results presented within the given domain shows the outcomes and results achieved within very similar domains, at Word Level TF-IDF the accuracy of the MNB algorithms tends to perform the word comparatively to the other results but the Logistic Regression seems to out-perform the given other approaches.

Tanvir et al (2019) approach the reason for the logistic regression approach not performing as well was due to Logistic Regression approach is vulnerable to overconfidence. As each data point needs to be independent of on-another. The reason for the given Logistic Regression performing the best for our give approach is due to the majority of data-points being independent of one-another due to the number of various different news articles covering various news aspects within the chosen dataset.

Poddar et al's results seemed to achieve respectable results across the board with the main reason for achieving this was due to the dataset mainly as the given dataset mainly focused on various news articles released between the years 2016-2017 from the top 15 American publications which were generated using articles to analyse the connections between common political affiliations. Setting up such a dataset drastically reduces the scope of fake news observed as the given classification system achieves greater results within a more restricted domain.

5.3.2 Bigram Level Comparative Analysis

Source of Results	Features	Multinomial Naïve Bayes (%)	Random Forrest (%)	SVM (%)
Gilda et al (2017)	TF-IDF – Word Level Analysis	67.9	67.6	76.2
Given Thesis Results	TF-IDF – Word Level Analysis	63.8	87.2	93.6

Table 4 TF-IDF at Bigram Level

Table 4 below shows the given results achieved for Gilda et al in-which there was a high focus on N-gram analysis at bi-gram level. Comparing the comparative results straight off the bat it's clear to see that the Random Forrest and SVM classifiers noticeably outperforming any given model on either the given thesis or Gilda et al. When comparing the MNB performance it's important to note that the

methodology to set up the given classifier varied significantly as during the MNB comparison Gilda et al set up a model that randomly selects a classification for each article as either reliable or unreliable based on various posterior probabilities. Such an approach with varying approaches could significantly alter the results leading to a large difference between our given results and Gilda et al.

5.4 Experiment 2: Results & Evaluation

5.4.1 World-Level Analysis

Table 5 Bag of Words feature Extraction on Word-Level N-gram analysis

Classification Type	Accuracy (%)	K-Fold Accuracy (%)	Precision %	Recall %	F1-Score %
Multinomial Naïve Bayes	90.4	90.6	90	90	90
Logistic Regression	94.6	90.8	95	95	95
SVM	92.9	93	93	93	93
Random Forrest	85	84	86	85	85

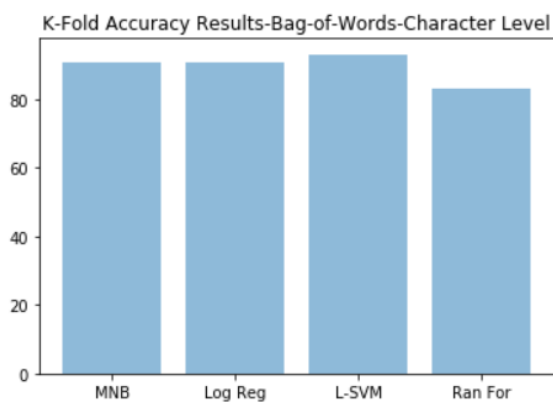


Figure 23 K-Fold Cross Validation - Word-Level Analysis

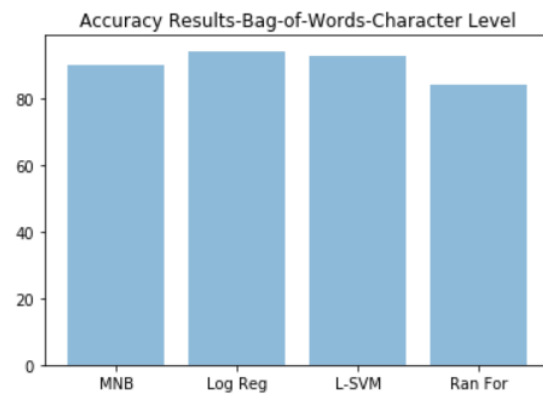


Figure 24 Accuracy Bag of Words, Word Level

From the given results shown in table 5 and figure 23 and 24 its clear the least favourable given model is Logistic Regression achieving 94.6% accuracy and 90.8% K-Fold Cross Validation with greater precision, recall and F1-Score. MNB and SVM both achieve very similar results with SVM achieving a greater overall score; with an accuracy of 92.9% and K-Fold validation score of 93% compared to MNB with 90.4% accuracy with 90.6% K-Fold Score. The slight discrepancy between the Logistic Regression accuracy results against K-Fold results could potentially be due to the favourable train/test split of the given run causing there to be a noticeable difference between the two results. As observed from table 5 overall all classifiers seem to achieve respectable results across the board for this given approach. Random Forest achieves the worst as it obtains 85% accuracy with 84% K-Fold average.

5.4.2 Bigram-Level Analysis

Table 6 Bag of Words feature Extraction on Bigram-Level N-gram analysis Results

Classification Type	Accuracy (%)	K-Fold Accuracy (%)	Precision %	Recall %	F1-Score %
Multinomial Naïve Bayes	92.4	89.8	93	92	92
Logistic Regression	91.3	90.8	92	91	91
SVM	89.6	88.9	90	90	90
Random Forrest	88.4	87.6	89	89	89

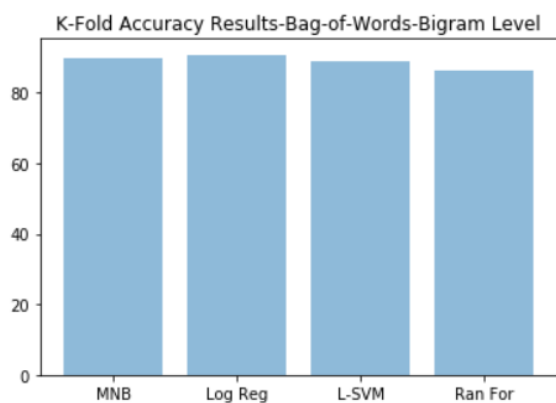


Figure 25 K-Fold Cross Validation, Bigram-Level

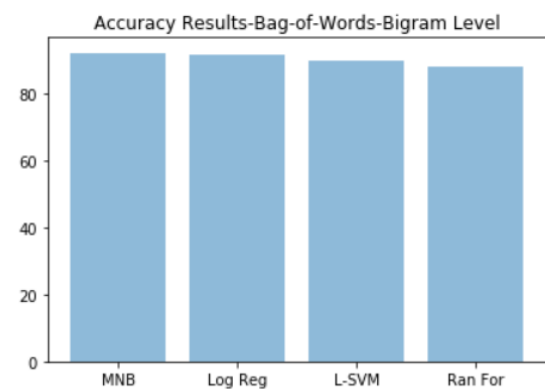


Figure 26 Individual Accuracy Score, Bigram-Level

The results and graphical information presented within table 6 and figure 25 and 26 show potentially conflicting results. Beginning with Random Forest which came last with 84% Accuracy and 87.6% K-Fold Cross Validation then came SVM with 89.6% individual accuracy and 88.9% K-Fold average. Then the Accuracy metrics between MNB and Logistic Regression was quite skewed as MNB achieved the highest Accuracy metric with 92.4% which is 1.1% greater than Logistic Regression but achieved 89.8% on K-Fold Cross validation, whereas Logistic Regression attained 90.8% K-Fold Cross validation. Due to MNB achieving a contrasting outcome between accuracy and K-Fold, Logistic Regression would be considered the favourable classification algorithm for the given experiment.

5.4.3 Character – Level Analysis

Table 7 Bag of Words feature Extraction on Character-Level N-gram analysis Results

Classification Type	Accuracy (%)	K-Fold Accuracy (%)	Precision %	Recall %	F1-Score %
Multinomial Naïve Bayes	64.2	63.8	64.0	64.0	64.0
Logistic Regression	71.0	70.6	71.0	71.0	71.0
SVM	N/A	N/A	N/A	N/A	N/A
Random Forrest	69.0	68.9	69.0	69.0	68.0

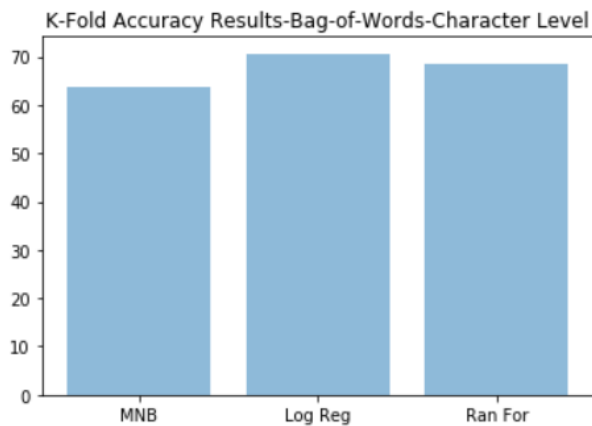


Figure 28 Bag of Words, Character-Level Graphical Results

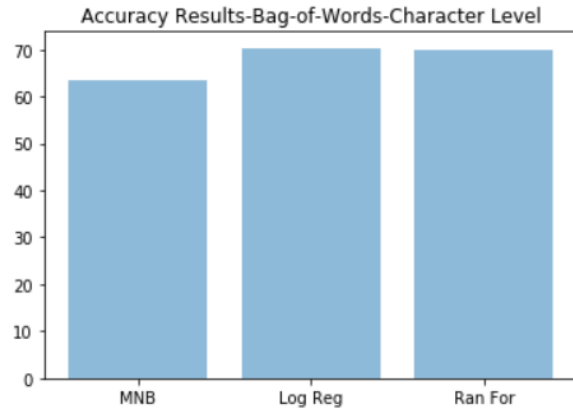


Figure 27 Bag of Words Character Level Results Accuracy Score

The final individual aspect of Experiment 2 is analysing the character Level Bag of Words approach. The character Level didn't reach the higher accuracy and K-Fold scores as achieved by the Bigram and Unigram Level experiments that were carried out within the bag of words approach but seemed to still achieve respectable results, nonetheless. From the given graphical and tabular results above MNB achieved 64.2% accuracy with a K-Fold score of 63.8%, whereas Random forest fared slightly better achieving 69.0% accuracy with 68.9% K-Fold score. Again, the classification algorithm that achieved the greatest was Logistic Regression achieving 71% accuracy with 70.6% K-Fold score.

One observation made is the nearness both Logistic Regression and Random Forest seemed to perform to one-another as both classification algorithms consistently achieved extremely close results to one-another and in some cases less than 1.0% to each other in both accuracy and K-Fold score respectively.

The reason as for why SVM could obtain any results it due to the computational intensity of the given algorithm cause the run-time for the given algorithm taking too long preventing any results from being obtained for the given algorithm, causing the algorithm to be omitted from the character level analysis for experiment 2.

5.5 Comparing obtained Results Against Published Works

5.5.1 Unigram & Bigram Level Analysis

Table 8 Comparison between Kharde et al 2016 research paper on Sentiment Analysis within twitter against the Bag of Words Results obtained within the thesis

Source of Results	N-gram Analysis	Multinomial Bayes (%)	Naïve	SVM (%)
Kharde et al 2016	Unigram	73.1		75.4
Kharde et al 2016	Bigram	74.6		76.1%
Given Thesis	Unigram	90		93.0
Given Thesis	Bigram	92		88.9

When comparing Kharde et al (2016) it is immediately obvious as to the large difference between the results obtained, as even though the domain is slightly different between the two sources the approach adopted by both papers are largely quite similar just being applied to a different domain. A possible reason for Kharde et al (2016) not achieving as good with the MNB algorithm could be due to the fact that one aspect of the given data pre-processing that was focused upon is removal of stop words within an already short corpus due to the domain being twitter related. Removing stop words from a tweet could be drastic in how much valuable information could be lost in a limited corpus leading to potentially poor accuracy during modelling. This could explain the reason for the large difference with the results obtained between the two papers

5.6 Experiment 3: Results & Evaluation

Table 9 Voting Ensemble -10-Fold Cross Validation Results between TF-IDF & BoW

Feature Extraction Technique	K-Fold Accuracy (%)
TF-IDF	83.7
Bag of Words	83.9

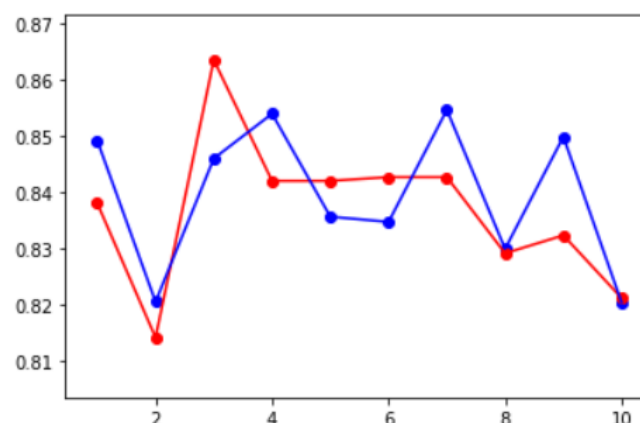


Figure 29 Result for Each Result During K-Fold Execution, Bag of Word = Blue, TF-IDF = Red

When observing the results and evaluating which approach between the Bag of Words compared to the TF-IDF Ensemble classifier what was found was that both voting classifiers consistently performed the same or extremely similar to one-another. These results were repeatedly run, with K-Fold Cross Validation results where K = 10 consistently finding the results to be between 0.5% of each other. An observation to note is that the TF-IDF Ensemble's individual K-Fold results commonly had a greater spread compared to the Bag of Words approach which could cause the given Ensemble to perform 0.2% worse than its Bag of Words counterpart.

5.7 Overall Evaluation

When analysing the results and evaluations of the given experiments conducted and comparing experiment 1 against experiment 2, we find that at word level and Bigram Experiment 1 achieves the greatest individual classifier but experiment 2 achieves the greatest overall accuracy across all performing classifiers. That is when considering all performing classifiers experiment 2 seems to have to greatest mean average score compared to experiment 1. When analysing character level analysis, it is found that experiment 1 is far superior to experiment 2 as its able to not only have a far greater mean average accuracy compared to experiment 2 but also contains the single best performing classifier from the two experiments.

It's therefore difficult to conclusively say that either experiment is better than the other as from the data the two feature extraction techniques perform so close to one another. Out of all the combinations it through both TF-IDF and Bag of Words it was noted how well linear SVM performed on both experiments. The given classifier consistently within most individual experiments within the project was always up there with its accuracy and K-Fold score, along with Logistic Regression too. These algorithms performed well for both vectorization techniques and hence are good choices for combating the problem-domain at least within the machine learning aspect.

6 Conclusion

In this paper we analyse and implement the most suitable techniques within machine learning today in order to detect fake news within a news domain with respectable accuracy. In order to achieve such, we split up given favoured approaches in order to find the most suitable approach at various N-gram levels and also find the most appropriate classifiers for each individual feature extraction technique too. When laying out these given approaches there was a particularly strong focus on whether these approaches which were commonly implemented within the twitter domain to find whether tweets are reliable or not were now being implemented in order to justify and detect whether the same could be said for online news sources. Not only did we find that we could comfortably implement such approaches within the news domain, but in some cases our results achieved greater accuracy then our twitter counterparts showing there is potential with the given approach in this problem domain.

Analysing whether the given project as whole is considered a success or not based on whether I believe all the objectives that were laid out at the beginning of the project, and based on the results obtained and the methodology used and the evaluation techniques I believe all the objective that were initially set have all been achieved.

7 References

- News Use Across Social Media Platforms 2016. (2020) Pew Research Centre's Journalism Project. Available online: <https://www.journalism.org/2016/05/26/news-use-across-social-media-platforms-2016/> [Accessed 10/4/2020].
- Fake news detector algorithm works better than a human. (2020) Techxplore.com. Available online: https://techxplore.com/news/2018-08-fake-news-detector-algorithm-human.html?utm_source=TrendMD&utm_medium=cpc&utm_campaign=TechXplore.com_TrendMD_1 [Accessed 10/4/2020].
- Greene, T. (2020) *This fake news detection algorithm outperforms humans*. The Next Web. Available online: <https://thenextweb.com/artificial-intelligence/2018/08/22/this-fake-news-detection-algorithm-outperforms-humans/> [Accessed 4/4/2020].
- I trained fake news detection AI with >95% accuracy, and almost went crazy. (2020) Medium. Available online: <https://towardsdatascience.com/i-trained-fake-news-detection-ai-with-95-accuracy-and-almost-went-crazy-d10589aa57c> [Accessed 10/4/2020].
- ONLINE, T. (2020) *Tin đồn ăn bưởi bị ung thư làm thiệt hại hàng trăm tỉ*. TUOI TRE ONLINE. Available online: <https://tuoitre.vn/tin-don-an-buoi-bi-ung-thu-lam-thiet-hai-hang-tram-ti-216359.htm> [Accessed 28/3/2020].
- Shu, K., Sliva, A., Wang, S., Tang, J. & Liu, H. (2017) Fake News Detection on Social Media. *ACM SIGKDD Explorations Newsletter*, 19(1), 22-36.
- Allcott, H. & Gentzkow, M. (2017) Social Media and Fake News in the 2016 Election. *Journal of Economic Perspectives*, 31(2), 211-236.
- Wiemer, H., Drowatzky, L. & Ihlenfeldt, S. (2019) Data Mining Methodology for Engineering Applications (DMME)—A Holistic Extension to the CRISP-DM Model. *Applied Sciences*, 9(12), 2407.
- What is the CRISP-DM methodology?. (2020) Smart Vision - Europe. Available online: <https://www.sv-europe.com/crisp-dm-methodology/#businessunderstanding> [Accessed 4/6/2020].
- Reis, J., Correia, A., Murai, F., Veloso, A., Benevenuto, F. & Cambria, E. (2019) Supervised Learning for Fake News Detection. *IEEE Intelligent Systems*, 34(2), 76-81.
- Khalil, A., Hajdiab, H. & Al-Qirim, N. (2017) Detecting Fake Followers in Twitter: A Machine Learning Approach. *International Journal of Machine Learning and Computing*, 7(6), 198-202.
- Khalil, A., Hajdiab, H. & Al-Qirim, N. (2017) Detecting Fake Followers in Twitter: A Machine Learning Approach. *International Journal of Machine Learning and Computing*, 7(6), 198-202.
- Abdullah-All-Tanvir, Ehasas Mia Mahir, Saima Akhter, Mohammed Rezwanul Haq – Detecting Fake News Using Machine Learning and Deep Learning. In proceedings of 2019 7th International Conference on Smart Computing and Communication
- James Thorne, Mingjie Chen, Giorgos Myrianthous, Jiashu Pu, Xiaoxuan Wang, and Andreas Vlachos. Fake news stance detection using stacked ensemble of classifiers. In Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism, pages 80–83, 2017.
- Karishnu Poddar, Geraldine Bessie Amali D, Umadevi K S. Comparison of Various Machine Learning Models for Accurate Detection of Fake News, 2019 Innovations in Power and Advanced Computing Technology (i-PACT)
- Rohit Kumar Kaliyar, Anurag Goswami, Pratik Narang - Multiclass Fake News Detection Using Ensemble Machine Learning, 9th International Conference on Advanced Computing (IACC)
- NA Karur, S. (2015) Data mining and other Data base techniques for PhD thesis preparation. *Transactions on Machine Learning and Artificial Intelligence*, 3(3).

Updated Text Preprocessing techniques for Sentiment Analysis. (2020) Medium. Available online: <https://towardsdatascience.com/updated-text-preprocessing-techniques-for-sentiment-analysis-549af7fe412a> [Accessed 4/6/2020].

Removing Stop Words from Strings in Python. (2020) Stack Abuse. Available online: <https://stackabuse.com/removing-stop-words-from-strings-in-python/> [Accessed 4/6/2020].

Stemming? Lemmatization? What?. (2020) Medium. Available online: <https://towardsdatascience.com/stemming-lemmatization-what-ba782b7c0bd8> [Accessed 4/6/2020].

An Introduction to Bag-of-Words in NLP. (2020) Medium. Available online: <https://medium.com/greyatom/an-introduction-to-bag-of-words-in-nlp-ac967d43b428> [Accessed 4/6/2020].

Webb, G. & Yu, X. (2004) *AI 2004*. Berlin: Springer.

Introduction to Logistic Regression. (2020) Medium. Available online: <https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148> [Accessed 4/6/2020].

Random Forest Regression | Turi Machine Learning Platform User Guide. (2020) Turi.com. Available online: https://turi.com/learn/userguide/supervised-learning/random_forest_regression.html [Accessed 4/6/2020].

Understanding Random Forest. (2020) Medium. Available online: <https://towardsdatascience.com/understanding-random-forest-58381e0602d2> [Accessed 4/6/2020].

Random Forests Classifiers in Python. (2020) DataCamp Community. Available online: <https://www.datacamp.com/community/tutorials/random-forests-classifier-python#advantages> [Accessed 4/6/2020].

Ensemble Learning to Improve Machine Learning Results. (2020) Medium. Available online: <https://blog.statsbot.co/ensemble-learning-d1dcd548e936> [Accessed 4/6/2020].

Ensemble Learning to Improve Machine Learning Results. (2020) Medium. Available online: <https://blog.statsbot.co/ensemble-learning-d1dcd548e936> [Accessed 4/6/2020].

Brownlee, J. (2020) *A Gentle Introduction to k-fold Cross-Validation*. Machine Learning Mastery. Available online: <https://machinelearningmastery.com/k-fold-cross-validation/> [Accessed 4/6/2020].

Understanding Confusion Matrix. (2020) Medium. Available online: <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62> [Accessed 4/6/2020].

Understanding Confusion Matrix. (2020) Medium. Available online: <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62> [Accessed 4/6/2020].

Understanding Confusion Matrix. (2020) Medium. Available online: <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62> [Accessed 4/6/2020].

Python Vs. C++ for Machine Learning – Language Comparison. (2020) Netguru.com. Available online: <https://www.netguru.com/blog/cpp-vs-python> [Accessed 4/6/2020].

Best Languages For Machine Learning in 2020!. (2020) Medium. Available online: <https://becominghuman.ai/best-languages-for-machine-learning-in-2020-6034732dd242> [Accessed 4/6/2020].

Choudhury, A. (2020) *Top 10 Python NLP Libraries For 2019*. Analytics India Magazine. Available online: <https://analyticsindiamag.com/top-10-python-nlp-libraries-for-2019/> [Accessed 4/6/2020].

DevDocs — scikit-learn documentation. (2020) Devdocs.io. Available online: https://devdocs.io/scikit_learn/ [Accessed 4/6/2020].

MeTA: ModErn Text Analysis : MeTA. (2020) Meta-toolkit.org. Available online: <https://meta-toolkit.org/> [Accessed 4/6/2020].

Natural Language Toolkit — NLTK 3.5 documentation. (2020) Nltk.org. Available online: <https://www.nltk.org/> [Accessed 4/6/2020].

Selivanov, D. (2020) *text2vec*. Text2vec.org. Available online: <http://text2vec.org/> [Accessed 4/6/2020].

Why Jupyter is data scientists' computational notebook of choice. (2020) Nature.com. Available online: <https://www.nature.com/articles/d41586-018-07196-1> [Accessed 4/6/2020].

Gilda, S. (2017) Evaluating Machine Learning Algorithms for Fake News Detection. *2017 IEEE 15th Student Conference on Research and Development (SCoReD)*