A Minor Project Report

On

**Fake news detection**

**using**

**Natural language processing**

SUBMITTED IN PARTIAL FULFILLMENT FOR THE AWARD OF DEGREE OF

**Bachelor of Technology**

**IN**

**Electronics and Communication Engineering**



**Submitted By:** **Under the Guidance Of**

SHIWALI SINGH (9917102145) MRS. JYOTI VYAS

PRASENJIT KATHURIA (9917102259)

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA (U.P.)**

**MAY, 2020**

**CERTIFICATE**

This is to certify that the minor project report entitled, “**FAKE NEWS DETECTION USING NATURAL LANGUAGE PROCESSING**” submitted by **“SHIWALI SINGH AND PRASENJIT KATHURIA ”** in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in **Electronics and Communication Engineering** of the Jaypee Institute of Information Technology, Noida is an authentic work carried out by them under my supervision and guidance. The matter embodied in this report is original and has not been submitted for the award of any other degree.

**Signature of Supervisor:**

**Name of the Supervisor: MRS. JYOTI VYAS**

**ECE Department,**

**JIIT, Sec-128,**

**Noida-201304**

**Dated:**

**DECLARATION**

We hereby declare that this written submission represents our own ideas in our own words and where others ideas or words have been included have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission.

Place: **NOIDA**

Date:

Name: **Shiwali Singh**

Enrollment: **9917102145**

Name: **Prasenjit Kathuria**

Enrollment: **9917102259**

**CHAPTER 1: ABSTRACT**

Fake news has become an important topic of research in a variety of disciplines including linguistics and computer science. Writing fake comments and news on social media is easy for users. The main challenge is to determine the difference between real and fake news. The problem is approached from the perspective of natural language processing, with the goal of building a system to automatically detect misinformation in news.

The main challenge in this line of research is collecting quality data, i.e., instances of fake and real news articles on a balanced distribution of topics. This project attempts to distinguish between “fake news” and “absurd news” by building a classification model trained on Subreddit posts from TheOnion and nottheOnion.

It's imperative to maintain a high level of accuracy when creating a fake news detector--where all fake news gets filtered out and authentic news will not be affected by a filter. We also perform a topic modelling experiment to elaborate on the gaps and sources of imbalance in currently available datasets to guide future effort.

**ACKNOWLEDGEMENTS**

Education along with the process of gaining knowledge and stronghold of subject is a continuous and ongoing process. It is an appropriate blend of mind-set, learnt skills, experience and knowledge gained from various resources. This project would not have been possible without the support of many people.

We would like to express our gratitude and indebtedness to MRS. JYOTI VYAS and ECE project lab coordinators for their kind and valuable guidance that made the meaningful completion of project possible. New ideas and Direction from them made it possible for us to sail through various areas of our Natural Language Processing and Python Language which were new to us. We would also thank our institution and faculty members without whom this project would have been a distant reality.

**TABLE OF CONTENTS**

**CHAPTER 1: INTRODUCTION PAGE NO:**

* 1. Introduction 1

1.2 How we Planned our model 2

**CHAPTER 2: LITRETURE SURVEY**  3

**CHAPTER 3: METHODOLGY**

3.1 Motivation for the dataset 4

3.2 Dataset Description 4

3.3 Project Workflow 5-6

**CHAPTER 4: WEB SCRAPPING AND EDA**

**4.1 Web Scrapping 7**

4.1.1 The basics of Web scrapping 7

4.1.2 Subreddits 8

4.1.3 PushshiftAPI 8-9

**4.2 What is EDA? 10**

4.2.1 Approach 10

4.2.2 Techniques 11

4.2.3 EDA on subreddit data 11

**CHAPTER 5: NLP AND MODELS**

**5.1 Natural Language Processing (NLP) 12**

5.1.1 NLP on subreddit Data 12

**5.2 Modelling 12**

5.2.1 Pipeline and GridSearch 13

5.2.2 Models 14

**5.3 Models Analysis 15**

**CHAPTER 6: RESULTS AND CONCLUSION**

6.1 Results 16

6.2 Model Evaluation 17

6.3 Conclusions 19

**CHAPTER 7: REFERENCES** 20

**CHAPTER 1: INTRODUCTION**

**1.1 Introduction**

There was a time when if anyone needed any news, he or she would wait for the next-day newspaper. However, with the growth of online newspapers who update news almost instantly, people have found a better and faster way to be informed of the matter of his/her interest. Nowadays social-networking systems, online news portals, and other online media have become the main sources of news through which interesting and breaking news are shared at a rapid pace. However, many news portals serve special interest by feeding with distorted, partially correct, and sometimes imaginary news that is likely to attract the attention of a target group of people. Fake news has become a major concern for being destructive sometimes spreading confusion and deliberate disinformation among the people. The term fake news has become a buzz word these days. However, an agreed definition of the term “fake news is still to be found. It can be defined as a type of yellow journalism or propaganda that consists of deliberate misinformation or hoaxes spread via traditional print and broadcast news media or online social media. These are published usually with the intent to mislead in order to damage a community or person, create chaos, and gain financially or politically. Since people are often unable to spend enough time to cross-check reference and be sure of the credibility of news, automated detection of fake news is indispensable. Therefore, it is receiving great attention from the research community.

Recent progress in natural language generation has raised dual-use concerns. While applications like summarization and translation are positive, the underlying technology also might enable adversaries to generate neural fake news. Early in 2019, WhatsApp announced it was deleting over 2 million accounts a month to prevent the rampant spread of fake news. Their decision came after incidents of violent attacks in India were triggered by the spread of fake news on the messaging platform.

We were curious about how WhatsApp created their fake news filter, so we used open source data from Reddit to make one ourselves. We scraped around 30,000 posts from subreddits TheOnion[4] and nottheonion[5] and built a classification model that could distinguish between fake news from TheOnion and absurd news from nottheonion.

**1.2 How we Plan to build our Model**

While building this model, we optimized for accuracy. That is, we wanted to have the highest possible outcomes of True Negatives and True Positives, and least number of False Positives and False Negatives. The worse scenario is deleting an account or post that shares authentic news, mistaking it for fake news. After cleaning, analysing, and performing NLP functions to the data, Using Pipeline and GridSearch, we try to find the best combination of vectorizers and models to achieve the best accuracy when designating a post as either from TheOnion or nottheonion.



Figure1.1: Pictorial representation of the problem and brief introduction of our proposed dataset

**CHAPTER 2: LITERATURE SERVEY**

**1. https://journals.sagepub.com/doi/pdf/10.1177/2053951719843310[1]**

This paper explains how the problem is approached from the perspective of natural language processing, with the goal of building a system to automatically detect misinformation in news. The main challenge in this line of research is collecting quality data, i.e., instances of fake and real news articles on a balanced distribution of topics. We review available datasets and introduce the MisInfoText repository as a contribution of our lab to the community. We also perform a topic modelling experiment to elaborate on the gaps and sources of imbalance in currently available datasets to guide future efforts. We appeal to the community to collect more data and to make it available for research purposes.

**2. https://www.datacamp.com/community/tutorials/web-scraping-python-nlp[2]**

Learn to scrape novels from the web and plot word frequency distributions; You will gain experience with Python packages requests, BeautifulSoup and NLTK. In this, we used some basic Natural Language Processing to plot the most frequently occurring words in the novel *Moby Dick*. In doing so, we also see the efficacy of thinking in terms of the following Data Science pipeline with a constant regard for process:

**3. https://arxiv.org/pdf/1911.03854v2.pdf[3]**

According to a Pew Research Centre report, 50% of Americans view fake news as a critical problem, placing it above violent crime. In addition, the report found that 68% of Americans view fake news as having a significant impact on their confidence of the government and 54% viewed it as having a large impact in their trust in one another. As such, research in the area of fake news detection is of high importance for society. We overcome these limitations posed by conventional datasets through the dataset we propose: Fakeddit. Our dataset will expand fake news research into the multimodal space and allow researchers to develop stronger, more generalized, fine-grained fake news detection systems.

**CHAPTER3: METHODOLOGY**

**3.1 Motivation for The Dataset**

There are many types of false news, ranging from satire to propaganda. In this project, we focus on text-only documents formatted as news articles: stories and their corresponding metadata that contain purposefully false information. In order to build a text classification system to detect false from true content based on linguistic cues, we need news articles assessed individually and labelled with respect to their level of veracity. This type of data collection is labour-intensive and requires hours if hard work.

**3.2 Dataset Description**

We have extracted ~30000 news articles using pushshiftAPI [6] from subreddits: TheOnion [4] and nottheonion [5]. The Onion is a website that publishes made-up stories that are by their topics ridiculous, but reasonable enough that they seem plausible. nottheonion on the other hand is a subreddit that concentrates on real news that seem so out of this world that one would easily mistake them for parody news. We will try distinguish articles from these 2 sources through text classification techniques Based on Natural Language Processing.

From each subreddit I scraped the following information:

1. **title:** Title of subreddit post
2. **Sort**- sort direction of result (“asc” or desc)
3. **Sort**\_**type** – Parameter used for sort
4. **After**- restrict results to those made after this epoch time
5. **Before**- restrict results to those made before this epoch time
6. **Subreddit**: Which subreddit the post belongs to
7. **Number of comments**: The number of comments for a post
8. **author:** Author's username of a post
9. **Subreddit subscribers**: Number of subscribers for that subreddit
10. **score:** The score received on reddit
11. **domain:** The domain referenced in the post
12. **Created time and date**: Date and Time the post was created

All the data after scapping is stored in csv file and we access it through the pandas library in python.

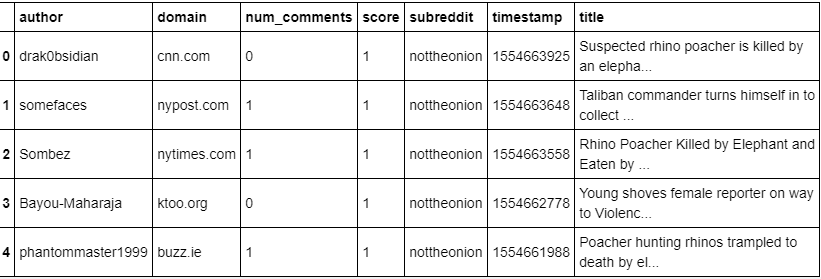


Figure3.1: A sample snapshot of the dataset after web scrapping.

**3.3 Project Workflow**

After collecting our data, we move on to data cleaning where we look for Null values, outliers and duplicates that could hamper our accuracy of the model and we drop all these rows. We also remove numbers and convert all the data into lowercase. After this, our data is ready.

Then we move onto Exploratory Data analysis where we create data visualizations to observe trends about the data so that we can decide a best-fit approach for our data. We also try to find distinguishing characteristics about each subreddit. Then we our Prepare text for modelling. We Count Vectorize the data and analyse trends among posts.

After that we try to find the best combination of vectorizers and models to achieve the best accuracy when designating a post as either from TheOnion [4] or notheonion [5]. We test our model on 4 different combinations of vectorizers and algorithms and to get the best possible accuracy in our project.

|  |  |
| --- | --- |
| **PROJECT WORKFLOW DESCRIPTION** | |
| Workflow | Description |
| Data Acquisition | Scrape ~15k posts from TheOnion & nottheonion, total of ~30k posts. Used pushshift.io API wrapper to acquire data. Clean the data. |
| Exploratory data analysis | Create data visualizations to observe trends about the data. What are distinguishing characteristics about each subreddit? |
| Natural Language Processing | Prepare text for modelling. Count Vectorize the data and analyse trends among posts. |
| Modelling | Using Pipeline and GridSearch, find the best combination of vectorizers and models to achieve the best accuracy when designating a post as either from TheOnion or nottheonion. |

TABLE 3.1: Workflow Description to build a classification model to distinguish between Subreddit posts from TheOnion and nottheonion

**CHAPTER 4- WEB SCRAPING AND EDA**

**4.1 Web Scraping**

Web scraping [2], also known as web data extraction, is the process of retrieving or “scraping” data from a website. Unlike the mundane, mind-numbing process of manually extracting data, web scraping uses intelligent automation to retrieve hundreds, millions, or even billions of data points from the internet’s seemingly endless frontier.

**4.1.1 The basics of web scraping**

It’s extremely simple, in truth, and works by way of two parts: a web crawler and a web scraper. The web crawler is the horse, and the scraper is the chariot. The crawler leads the scraper, as if by the hand, through the internet, where it extracts the data requested.

**1)The crawler**

A web crawler, which we generally call a “spider,” is an artificial intelligence that browses the internet to index and search for content by following links and exploring, like a person with too much time on their hands.

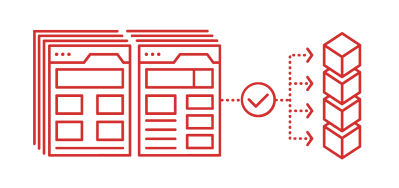


Figure 4.1The web crawler

**2) The scraper**

A web scraper is a specialized tool designed to accurately and quickly extract data from a web page. Web scrapers vary widely in design and complexity, depending on the project.

**4.1.2 Subreddits**

We have used Pushshift.io API Wrapper and scraped approximately 30,000 posts from the Subreddits TheOnion and nottheonion. Posts on the TheOnion feature satirical news from www.theonion.com or other similar parody sites. Posts on nottheonion feature absurd current events reported on by credible news outlets.

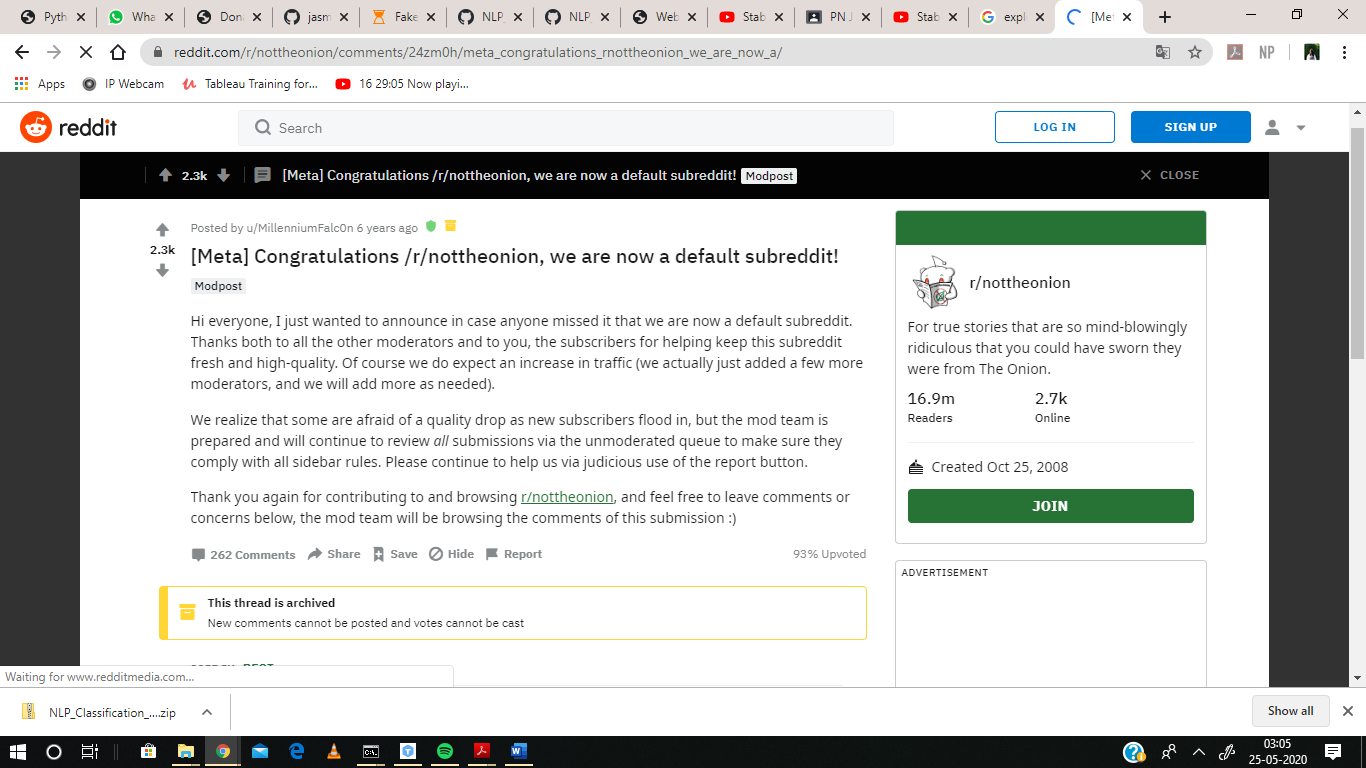
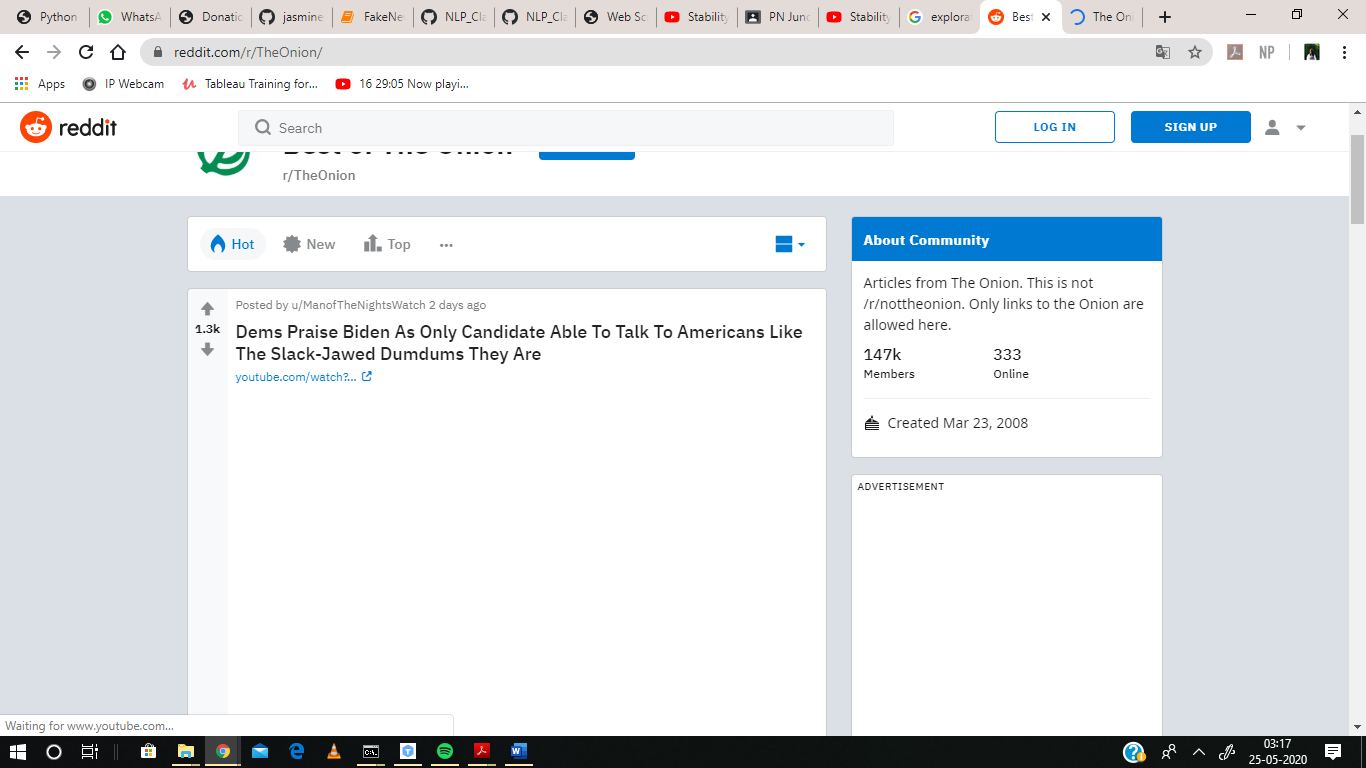
 

Figure 4.2 TheOnion has 147k subscribers while nottheonion has 16.9 million subscribers.

**4.1.3 Pushshift API**

It is an important open data initiative, which ingest data from various social media companies such as Twitter and Reddit and make that data available for researchers and academic institutions.

**List of Endpoints: -**

There are three main endpoints for the API to get information on comments, submissions and subreddits. The main endpoints are:

* /reddit/comment/search
* /reddit/submission/search
* /reddit/subreddit/search

We chose /reddit/subreddit/search to see how well we could distinguish between fake news and absurd news. There are total 100 Pushshift[6] Reddit specific parameters. From each subbreddit we scraped 8 parameters. These contain the following information:

* **Title: -** title of subreddit post
* **Subreddit: -** which subreddit the post belongs to
* **Num\_comments: -** the number of comments for a post
* **Author: -** author's username of a post
* **Subreddit** **subscribers: -** number of subscribers for that subreddit
* **Score: -** the score received on reddit
* **Domain:** - the domain referenced in the post
* **Created\_utc:** - date and time the post was created

In total, 30k posts were scraped: 15k from nottheonion and 15k from TheOnion.

We observed that the number of posts shared by an author in notheonion with 16.9 million subscribers, only one authors shared over 100 posts and more than 900 posts are deleted, while TheOnion with 147k subscribers had 13 authors who shared over 100 posts, the most being 4446 posts. One of the biggest problems with fake news is not necessarily that it gets written, but rather that it gets spread. The activity of TheOnion’s authors mimics the core qualities of the fake news phenomenon.

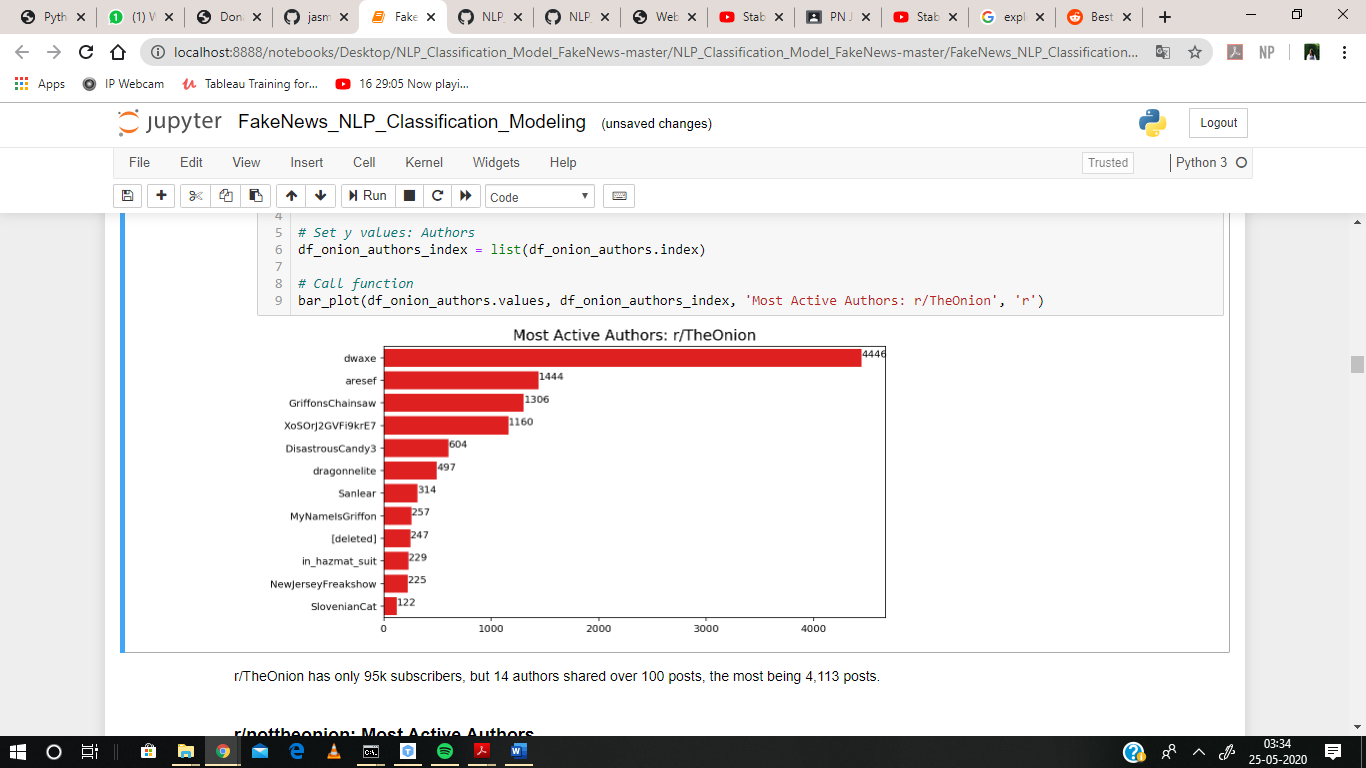
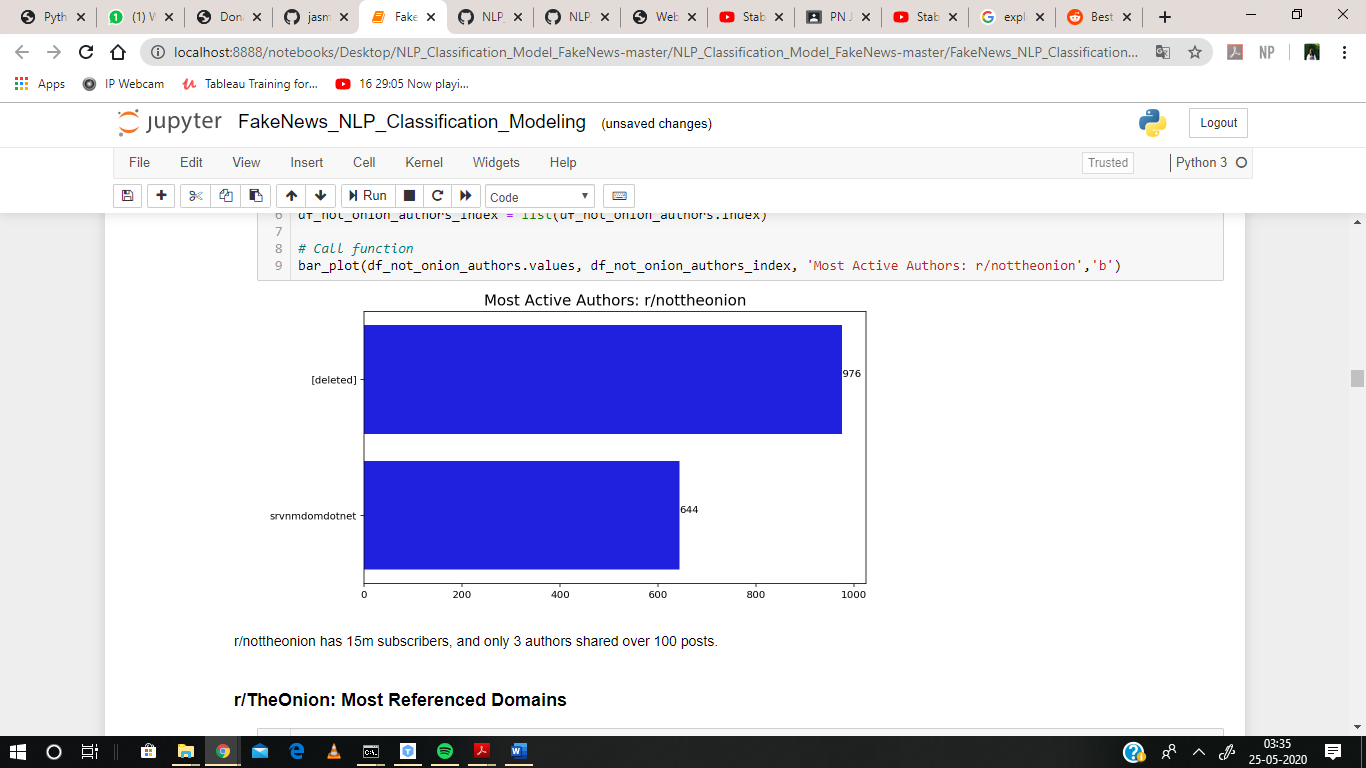
 

Figure 4.3 Most Active Authors in nottheonion & and TheOnion.

To keep data clean and concise, we chose to make predictor variable (X) the title of a post and target variables (y) be 1 to represent TheOnion and 0 to represent nottheonion. To clean data, we have created a data cleaning function that dropped duplicate rows in a DataFrame, removed punctuation and numbers from all text, removed excessive spacing, and converted all text to lowercase. After the Subreddit dataset was nice and clean it was ready to conduct an exploratory data analysis (EDA).

**4.2 What is EDA?**

In [statistics](https://en.wikipedia.org/wiki/Statistics), exploratory data analysis (EDA)[12] is an approach to [analysing](https://en.wikipedia.org/wiki/Data_analysis) [data sets](https://en.wikipedia.org/wiki/Data_set) to summarize their main characteristics, often with visual methods. A [statistical model](https://en.wikipedia.org/wiki/Statistical_model) can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modelling or hypothesis testing task.

* + 1. **Approach**

Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis that employs a variety of techniques (mostly graphical) to

* Maximize insight into a data set
* Uncover underlying structure
* Extract important variables
* Detect outliers and anomalies
* Test underlying assumptions
* Develop parsimonious models and determine optimal factor settings.
  + 1. **Techniques**

The particular graphical techniques employed in EDA are often quite simple, consisting of various techniques of:

1. Plotting the raw data (such as [data traces](https://www.itl.nist.gov/div898/handbook/eda/section3/runseqpl.htm), [histograms](https://www.itl.nist.gov/div898/handbook/eda/section3/histogra.htm), bi-histograms, [probability plots](https://www.itl.nist.gov/div898/handbook/eda/section3/probplot.htm), [lag plots](https://www.itl.nist.gov/div898/handbook/eda/section3/lagplot.htm), [block plots](https://www.itl.nist.gov/div898/handbook/eda/section3/blockplo.htm), and [Youden plots](https://www.itl.nist.gov/div898/handbook/eda/section3/youdplot.htm).
2. Plotting simple statistics such as [mean plots](https://www.itl.nist.gov/div898/handbook/eda/section3/meanplot.htm), [standard deviation plots](https://www.itl.nist.gov/div898/handbook/eda/section3/sdplot.htm), [box plots](https://www.itl.nist.gov/div898/handbook/eda/section3/boxplot.htm), and main effects plots of the raw data.
3. Positioning such plots so as to maximize our natural pattern-recognition abilities, such as using multiple plots per page.
   * 1. **EDA on subreddit data**

We have plotted box plot to analyse the most active author of both the subreddits and the results obtain is shown is figure 4.c

After that we have plotted the most used website from which these subreddits post news content. The outputs are as followed:

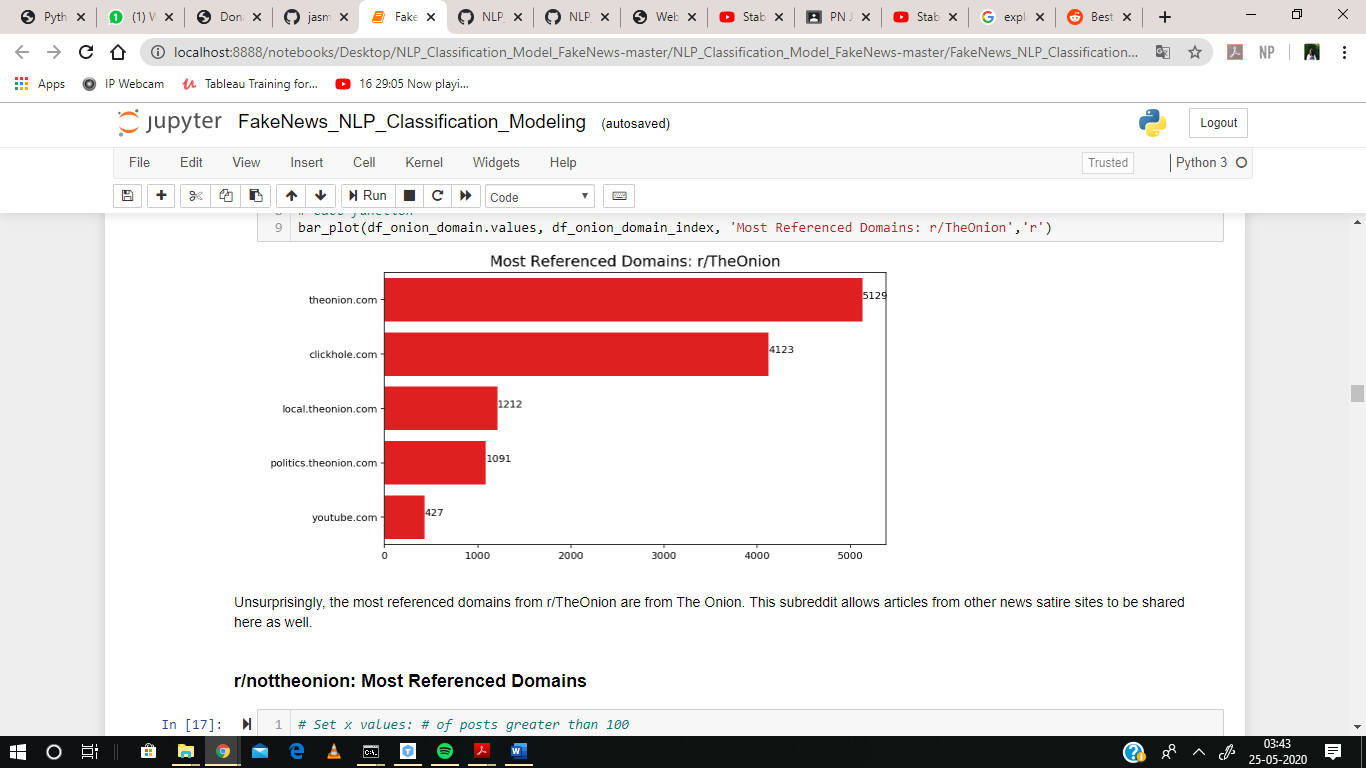


Figure 4.4 Most referenced Domain of TheOnion

Unsurprisingly, the most referenced domains from TheOnion are from The Onion. This subreddit allows articles from other news satire sites to be shared here as well.

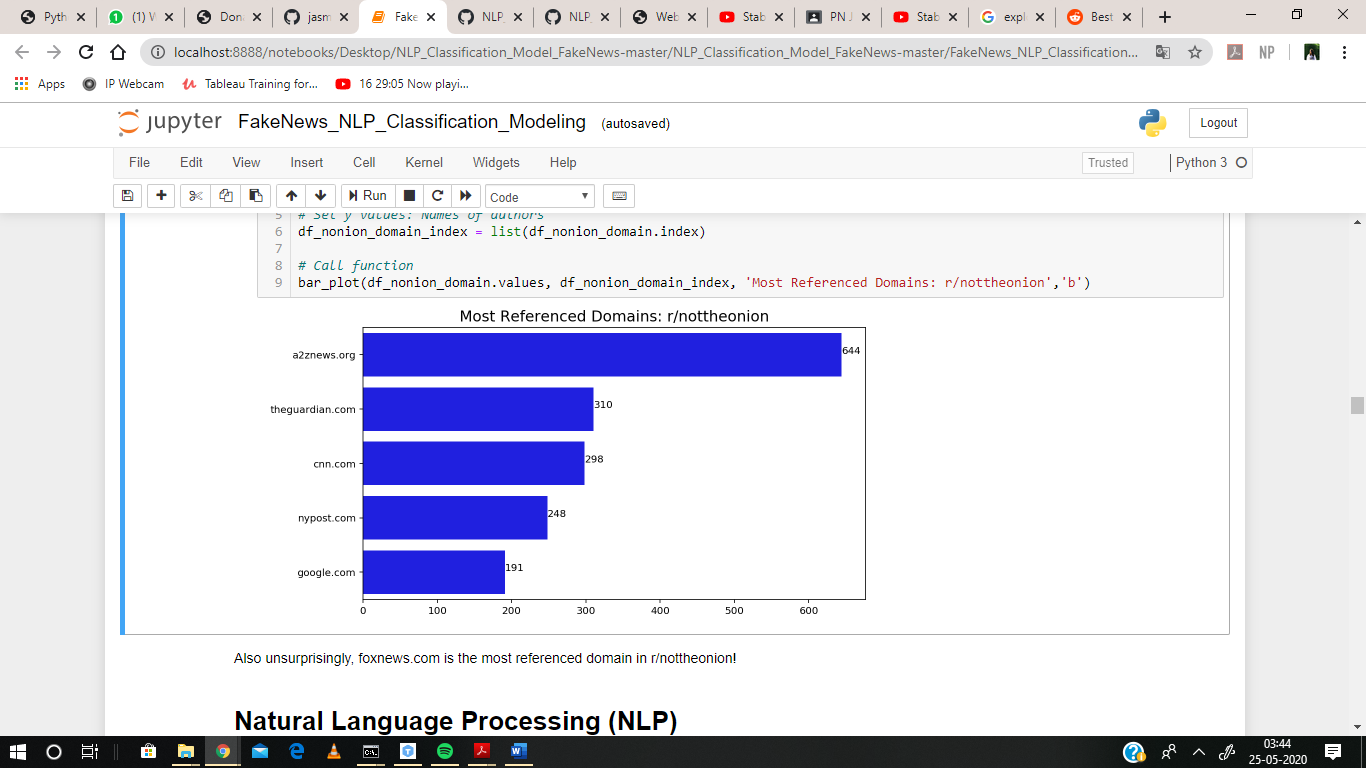


Figure 4.5 Most referenced Domain of nottheonion

Also, unsurprisingly, a2znews.org is the most referenced domain in nottheonion

Another interesting thing discovered during EDA of the data was the most referenced domains in each Subreddit. Of course, the majority of the domains referenced in TheOnion were from theonion.com and other parody news sites. However, the most referenced domains in nottheonion gave me a kick. The top five most referenced domains on nottheonion were a2znews.org, theguardian.com, google.com, cnn.com, and nypost.com.

**CHAPTER 5- NLP AND MODELS**

**5.1 Natural Language Processing (NLP)**

Natural language processing [11] involves the reading and understanding of spoken or written language through the medium of a computer. This includes, for example, the automatic translation of one language into another, but also spoken word recognition, or the automatic answering of questions.

**5.1.1 NLP on subreddit data**

**a) Count Vectorizer-** Count Vectorize is used in DataFrame where subreddit is `0**`** and subreddit is `1**`** with ngram\_range = (1,1). The top 5 most used words are then plotted.

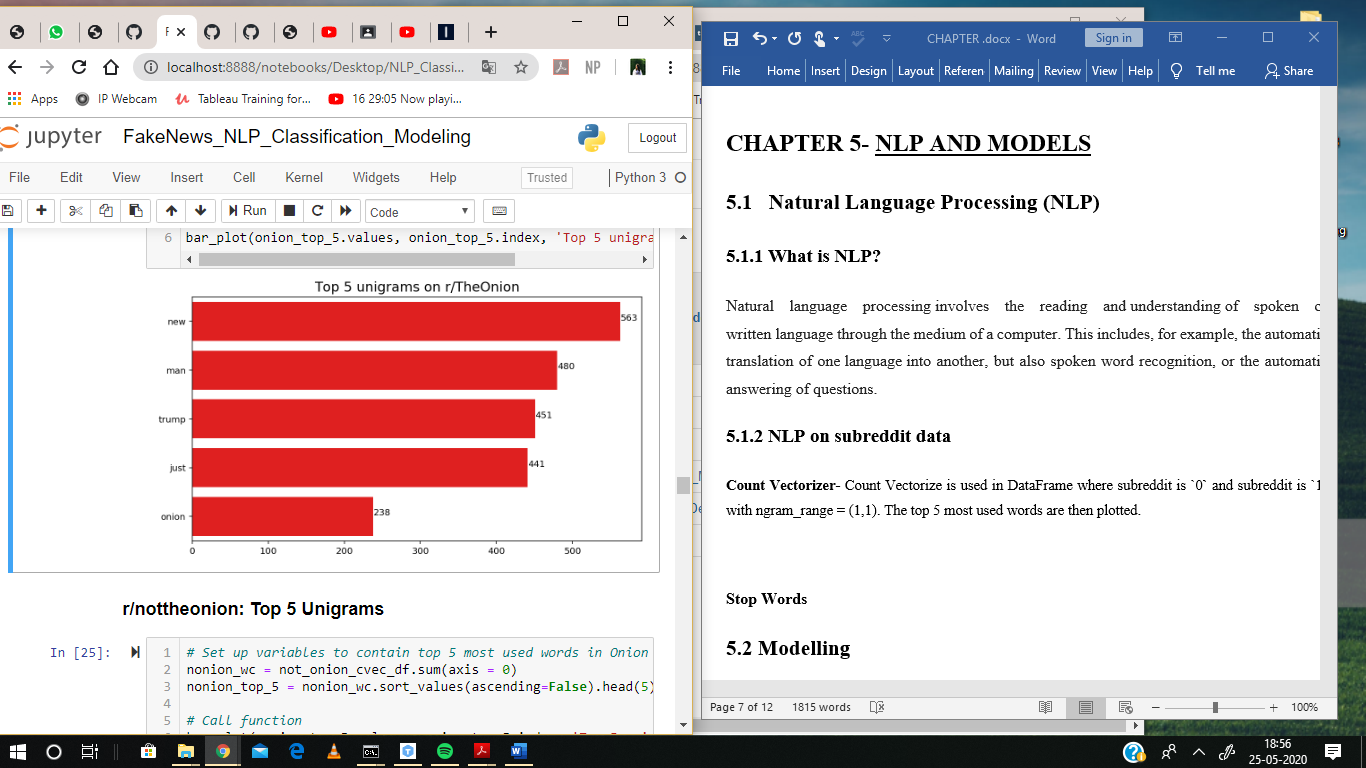


Figure 5.1 Unigram on TheOnion

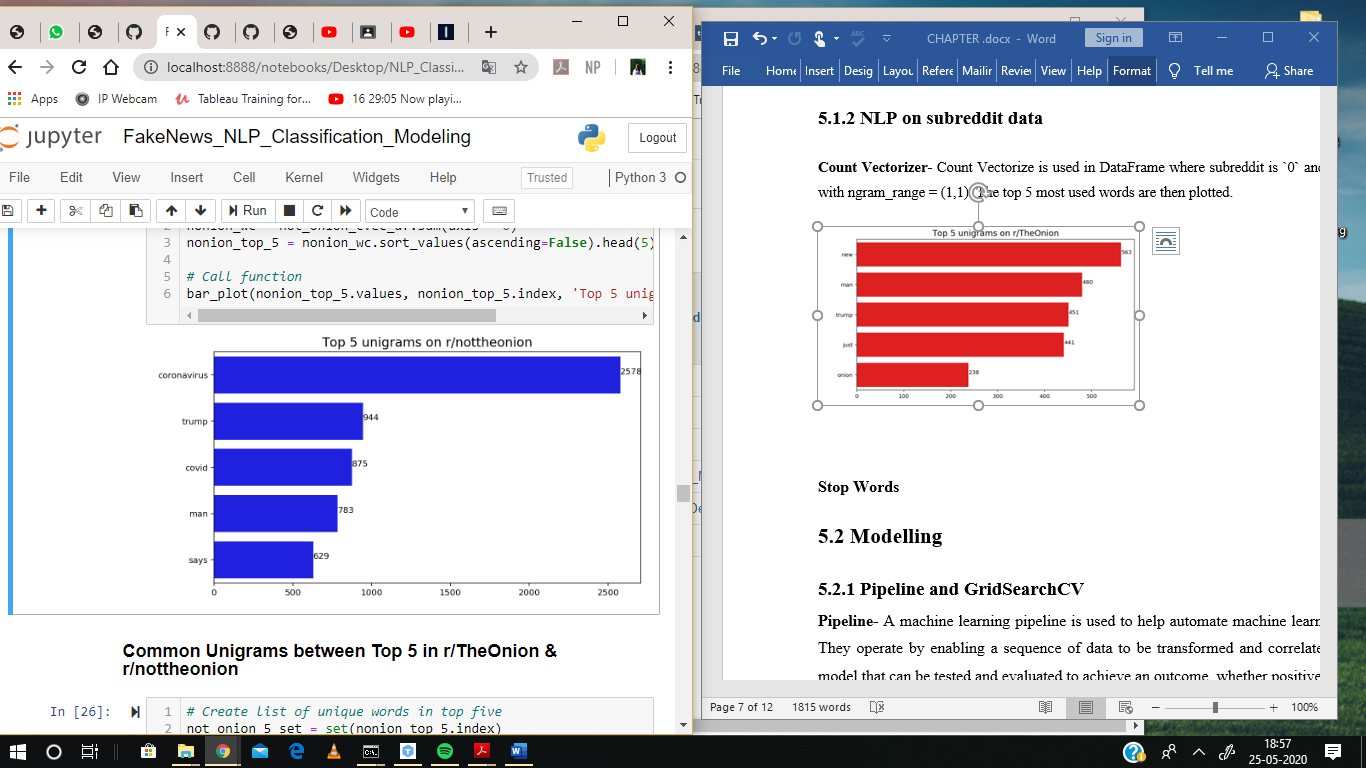


Figure 5.2 Unigram on notheonion

**b) Stop Words-** Take out the most appeared words from the dataset when modelling, since these words occur frequently in both subreddits. E.g. New, man, trump, etc.

**5.2 Modelling**

**5.2.1 Pipeline and GridSearchCV**

We have conducted model-prep and use Pipeline and GridSearchCV[14] to determine the most optimal model to run data on.

**Pipeline-** A machine learning pipeline is used to help automate machine learning workflows. They operate by enabling a sequence of data to be transformed and correlated together in a model that can be tested and evaluated to achieve an outcome, whether positive or negative.

**GridSearchCV-** Grid search is the process of performing hyper parameter tuning in order to determine the optimal values for a given model. This is significant as the performance of the entire model is based on the hyper parameter values specified.

Pipeline and GridSearchCV together are one of the most optimise pipeline method in Machine learning.

* + 1. **Models**

1. **Count Vectorizer -** Count Vectorization[10] involves counting the number of occurrences each word appears in a document (i.e. distinct text such as an article, book, even a paragraph!). Python’s Sci-kit learn library has a tool called CountVectorizer to accomplish this. There are three parameters of CountVectorizer.

* **max\_df-** Tell how many features in this case words, you want CountVectorizer to count. Setting this parameter can be incredibly useful when dealing with a large number of documents.
* **ngram\_range.** This deals with contiguous sequences of words.
* **stop\_words***-* If a list, that list is assumed to contain stop words, all of which will be removed from the resulting tokens.

1. **Logistic Regression-**It is a statistical model that in its basic form uses a logistic[13] function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).
2. **MultinomialNB-** The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. However, in practice, fractional counts such as tf-idf may also work.

**Parameters**

* **alpha:** float, optional (default=1.0) Additive (Laplace/Lidstone) smoothing parameter (0 for no smoothing).
* **fit prior:** Boolean. Whether to learn class prior probabilities or not. If false, a uniform prior will be used.
* **class prior:** array-like, size (n\_classes) Prior probabilities of the classes. If specified the priors are not adjusted according to the data.

1. **TfidfVectorizer-** Tf-idf[15] is a very common technique for determining roughly what each document in a set of documents is “about”. It cleverly accomplishes this by looking at two simple metrics: tf (term frequency) and idf (inverse document frequency). Term frequency is the proportion of occurrences of a specific term to total number of terms in a document. Inverse document frequency is the inverse of the proportion of documents that contain that word/phrase.

1. **Model Combinations-** We have combined all the models into 4 different models to get the best results. The combined models are: -

* Model 1: CountVectorizer & Logistic Regression
* Model 2: TfidfVectorize & Logistic Regression
* Model 3: CountVectorizer & MultinomialNB (Best Accuracy Score)
* Model 4: TfidfVectorizer & MultinomialNB
  1. **Models Analysis**

We have concluded the following analysis from the models using the given parameters-

**1.Model 1: CountVectorizer & Logistic Regression**

Best Coefficient Interpretability and the parameter values are –

* ngram\_range: (1, 1)
* stop\_words: None
* C: 1

Throughout the model testing, none of the stop\_word lists were chosen as a best parameter.

So, from here, we omit them from my parameter selection. Additionally, while the model is overfit, we will be optimizing to get the highest accuracy score in the test set.

**2. Model 2: TfidfVectorize & Logistic Regression**

The parameter values are –

* C: 1
* max\_df: 0.75
* min\_df: 5
* ngram\_range: (1, 3)

This model is also overfit. Model 1 performed with a better test score when comparing

Logistic Regression models.

**3. Model 3: CountVectorizer & MultinomialNB**

Best Accuracy Score and the parameter values are –

* ngram\_range: (1, 3)
* alpha: 0.6

The model is overfit, but as mentioned, we are optimizing for accuracy. All posts from

TheOnion must be classified as being from the subreddit TheOnion, and all posts from

nottheonion must be classified as being from the subreddit nottheonion. This model gave

the best test accuracy score.

**4.Model 4: TfidfVectorizer & MultinomialNB**

The parameter values are –

* alpha: 0.1
* max\_df: 0.75
* min\_df: 4
* ngram\_range: (1, 2)

This model is overfit. When comparing test scores of my MultinomialNB models, Model 3 performs better.

**CHAPTER 6- RESULTS AND CONCLUSION**

**6.1 Results**

Through Pipeline and Gridsearch, we tested four sets of models with unique parameters. The best model in terms of highest accuracy score was Model 3. The best model in terms of coefficient interpretability was Model 1.  
**Model 1 | Best Coefficient Interpretability:**  
CountVectorizer (stop\_words=None, ngram\_range= (1,1))  
LogisticRegression (C=1, solver='liblinear')

* Train score 0.96
* Test score 0.87

**Model 2:**  
TfidfVectorizer (max\_df=0.75, min\_df=3, n\_gram=(1,3))  
LogisticRegression(C=1, solver='liblinear')

* Train score 0.90
* Test score 0.86

**Model 3 | Best Accuracy Score:**  
CountVectorizer (stop\_words=None, ngram\_range=(1,3))  
MultinomialNB(alpha=0.36)

* Train score 0.995
* Test score 0.89

**Model 4:**  
TfidfVectorizer (max\_df = .75, min\_df = 4, ngram\_range=(1,2))  
MultinomialNB(alpha=0.1)

* Train score 0.92
* Test score 0.86

**6.2 Model Evaluation**

The is a visualization of my confusion matrix for model 3, which reveals the values of True Positives, True Negatives, False Positives (Type I error), and False Negatives (Type II errors). I optimized my model to achieve the highest Accuracy score, and also measure the Positive Predictive Rate, True Positive Rate, True Negative Rate, and Misclassification Rate.

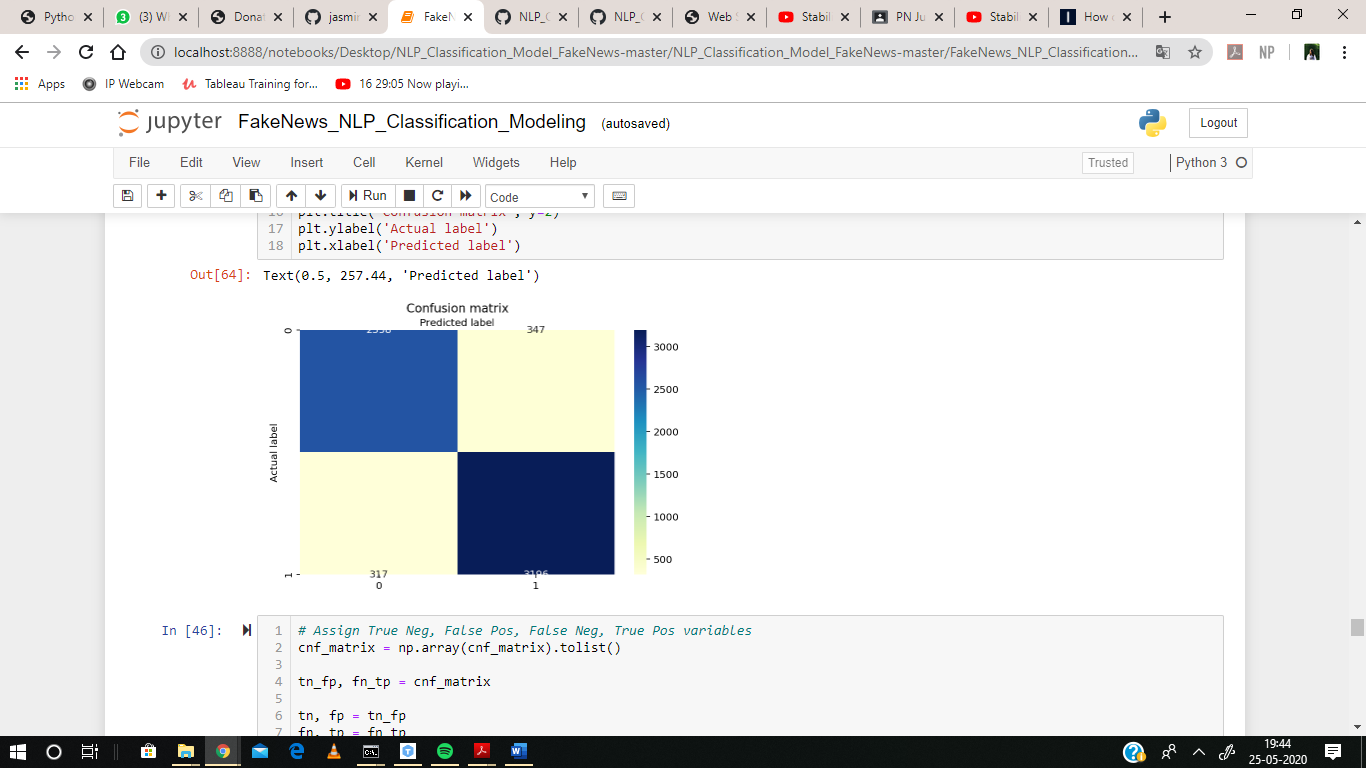


Figure 6.1 Confusion matrix of Model 3

Then we calculated the following parameter to analyse the output of confusion matrix

* Accuracy: 89.65 %
* Precision: 90.21 %
* Recall: 90.98 %
* Specificity: 88.06 %
* Misclassification Rate: 11.49 %

Additionally, we used model 1 to help us interpret estimate values for my coefficients. We determined which words had the highest positive impact on determining whether a post would be classified as coming from TheOnion or nottheonion and calculated the exponential of the coefficients to determine the likelihood of how a post gets classified.

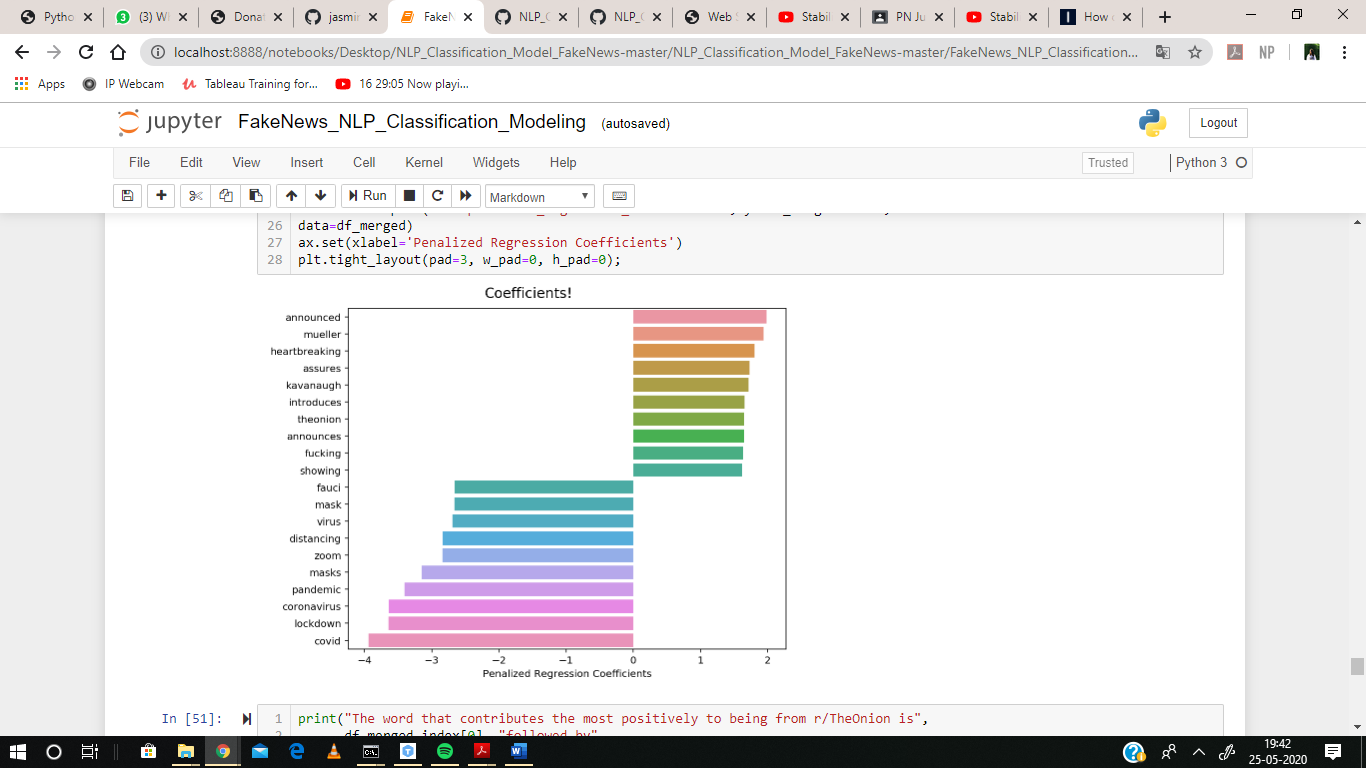


Figure 6.2 Penalized regression coefficients

**Output of regression coefficients:**

1. The word that contributes the most positively to being from TheOnion is announced followed by Mueller and heart-breaking.
2. The word that contributes the most positively to being from nottheonion is *covid*  followed by lockdown and coronavirus.

Next, we have plotted a DataFrame representing which word is more likely to be classified amongst which news.

|  |  |
| --- | --- |
| **Words** | **Exponential value** |
| announced | 7.235504 |
| mueller | 6.930972 |
| heartbreaking | 6.086054 |
| assures | 5.640654 |
| kavanaugh | 5.562235 |
| introduces | 5.227849 |
| theonion | 5.186194 |
| announces | 5.181010 |
| fucking | 5.114093 |
| showing | 5.037954 |

Table 6.1 Coefficients that affect TheOnion

As occurrences of announced increase by 1 in a title, that title is 7.24 times as likely to be

classified as TheOnion.

|  |  |
| --- | --- |
| **Words** | **Exponential value** |
| covid- | 51.676338 |
| lockdown | 38.244509 |
| coronavirus | 38.015729 |
| pandemic | 30.144425 |
| masks | 23.312740 |
| zoom | 17.167190 |
| distancing | 16.962415 |
| virus | 14.805519 |
| mask | 14.310593 |
| fauci | 14.210768 |

Table 6.2 Coefficients that affect nottheonion

As occurrences of **covid** increase by 1 in a title, that title is 51.68 times as likely to be

classified as nottheonion.

**6.3 Conclusions**

The goal of this project has been to comprehensively and extensively review, summarize, compare and evaluate the current research on fake news, which includes the qualitative and quantitative analysis of fake news, as well as detection and intervention strategies for fake news from four perspectives: the false knowledge fake news communicates, its writing style, its propagation patterns, and its credibility; main fake news characteristics (authenticity, intention, and being news) that allow distinguishing it from other related concepts (e.g., misinformation, disinformation, or rumours); various news-related (e.g., headline, body-text, creator, and publisher) and social-related (e.g., comments, propagation paths and spreaders) information that can exploited to study fake news across its lifespan (being created, published, or propagated); feature-based and relation-based techniques for studying fake news; and available resources, e.g., fundamental theories, websites, tools, and platforms, to support fake news studies.

**REFERENCES**

1) https://journals.sagepub.com/doi/pdf/10.1177/2053951719843310

2) https://www.datacamp.com/community/tutorials/web-scraping-python-nlp

3) https://arxiv.org/pdf/1911.03854v2.pdf

4) https://www.reddit.com/r/TheOnion/

5) https://www.reddit.com/r/nottheonion/

6) https://pushshift.io/

7) https://arxiv.org/pdf/1905.12616v2.pdf

8)https://arxiv.org/pdf/1907.07347v1.pdf

9)https://medium.com/datadriveninvestor/a-gentle-introduction-to-naive-bayes-classifier-9d7c4256c999

10)<https://scikitlearn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html>

11) <https://machinelearningmastery.com/natural-language-processing/>

12) <https://www.itl.nist.gov/div898/handbook/eda/section1/eda11.htm>

13) <http://www.appstate.edu/~whiteheadjc/service/logit/intro.htm>

14) <https://www.kaggle.com/evanmiller/pipelines-gridsearch-awesome-ml-pipelines>

15) <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html>