NANYANG
TECHNOLOGICAL
UNIVERSITY

A Study on Rumour Detection on Online Social Networks

Submitted in Partial Fulfilment of the Requirements for the
Degree in Bachelor of Engineering (Computer Science) of
the Nanyang Technological University

by

Cheng Gibson

School of Computer Science and Engineering

2017

Abstract

This study seeks to identify the key traits of rumours on online social networks such as Twitter and Facebook. The importance of automating the identification of rumours is growing ever-increasingly important, given the rise of the internet's popularity as a source of news, and the ever-growing amount of information on the internet.

A set of qualitative and quantitative metrics were developed to better understand the characteristics of each search query and the resultant dataset that it generates. The quantitative metrics indicate the size of the dataset, and the qualitative metrics evaluate the News/Rumour Purity and Contextual Purity of a dataset. The metrics will indicate how much preprocessing effort a dataset requires to dissect the different contexts from a dataset, and to make it more useful for further analysis.

Leveraging on existing literature from both Computer Science and from the Social Sciences, three experiments were formulated:

1. What are the general sentiment profiles of the datasets?
2. How well can rumours and non-rumours be separated in rumour-centric datasets?
3. How well can rumours and non-rumours be separated using all datasets?

The findings from the experiments indicate the following trends:

1. Features generated from sentiment analysis libraries such as SentiWordNet and AFINN can be as reliable as features generated from a tf-idf model, in terms of resultant classifier performance.
2. Tweets with a high proportion of neutral-sentiment words and a high proportion of punctuations are more likely to be related to the key contexts of their respective datasets.
3. Rumours and non-rumours can be separated with a high degree of accuracy, in the case of having two predefined but significantly different types of datasets (ie. One is rumour-centric, the other is news-centric)

Through the course of this project, several custom software was developed. An Android information harvester was developed to automate the task of collecting tweets. A tweet processing and analysis software were developed to automate the testing for the experiments. Lastly, A web user interface for data visualisation was developed to easily gain insights from the experiment results.

Acknowledgements

I would like to thank my project supervisor, Associate Professor Yeo Chai Kiat, for her guidance throughout the project and for her professionalism and kindness throughout her supervision.

I would also like to thank PhD Candidate Chen Weiling for her technical guidance and suggestions on how to progress through the various stages of the project.

Finally, I would like to thank Nanyang Technological University and the School of Computer Science and Engineering for the opportunity to pursue this Final Year Project.

Table of Contents

Abstract 2

Acknowledgements 3

List of Tables & Figures 7

1. Introduction 10

1.1. Objectives 10

1.2. Background 10

1.3. Project Scope 11

1.4. Project Resources 13

1.5. Report Structure 15

1.6. Project Planning 16

1.7. Project Schedule 17

2. Literature Review 19

2.1. Literature Review in Computer Science 19

2.2. Literature Review in Social Sciences 19

3. Information Harvester Development 21

3.1. Platform Considerations 21

3.2. Android Information Harvester Features 22

3.3. Worker Architecture 24

3.4. Data Storage Structure 25

3.5. JSON Configuration Files 25

3.6. Challenges 25

4. Dataset Selection & Characteristics 27

4.1. Selection Criteria 27

4.2. General Characteristics 28

4.3. Detailed Characteristics 29

4.3.1. ‘#Sickhillary’ 29

4.3.2. ‘Trump cabinet’ 30

4.3.3. 'US Economic Policy' 30

4.3.4. 'Mosul battle' 31

4.3.5. 'Baghdadi dead' 31

4.3.6. 'Death hoax' 32

4.4. Challenges 32

4.5. Insights 33

5. Data Workflow: Collection, Import & Retrieval 35

5.1. Overview 35

5.2. Collection: Information Harvester 36

5.3. Import: Database 36

5.4. Retrieval: Database 37

6. Feature Selection & Engineering 39

6.1. Feature Selection 39

6.2. Feature Engineering 39

6.2.1. Overview 39

6.2.2. Word Counts (Tf-idf) 39

6.2.3. Manual Labeling 40

6.2.3.1. Workflow 40

6.2.3.2. Manual Labeling Schemes 40

6.2.3.2.1. Labeling Scheme for Rumour-Centric Datasets 41

6.2.3.2.2. Labeling Scheme for News-Centric Datasets 42

7. Sentiment Analysis using Machine Learning Techniques 43

7.1. Sentiment Analysis Techniques 43

7.2. Real-Time Considerations 43

7.3. Machine Learning Techniques 45

7.3.1. Model Training 45

7.3.2. Model Testing 46

8. Full System Integration 47

8.1. Back-End Development 47

8.2. Front-End Development 47

8.2.1. Visualization of Datasets 47

8.2.2. Visualizations of Testing 49

9. Testing, Results & Analysis 50

9.1. Experiments 50

9.2. Testing Methodology 51

9.3. Results & Analysis 52

10. Discussion & Conclusion 63

10.1. Discussion 63

10.2. Conclusion 64

10.3. Recommendations for Future Work 64

References 65

Appendix 67

List of Tables & Figures

Table 1: Hardware Resources Utilized for Project 13

Table 2: Software Resources Utilised for Project 14

Table 3: Final Project Schedule 17

Table 4: Initial Project Schedule 18

Table 5: Comparison of Platforms for Android Information Harvester 21

Table 6: Additional Metadata Stored per Request by Tweet Collection Application 24

Table 7: Table of Quantitative Metrics of Datasets 28

Table 8: Table of Qualitative Metrics of Datasets 28

Table 9: Response Format for Twitter API Search Result 36

Table 10: Labeling Scheme for Datasets 40

Table 11: Example Tweets for Labeling Scheme for Rumour-centric Datasets 41

Table 12: Example Tweets for Labeling Scheme for News-centric Datasets 42

Table 13:Machine Learning Classifiers and Initialization Parameters Used 45

Table 14: Web Endpoints for Web Deployment 47

Table 15: Experiment 1 – 'trump cabinet' Feature Vector Performance Overview 52

Table 16: Experiment 1 – LR Coefficients for Support Label in 'trump_cabinet' Dataset 53

Table 17: Experiment 1 – LR Coefficients for POS in 'trump_cabinet' Dataset 53

Table 18: Experiment 1 – ‘sickhillary’ Feature Vector Performance Overview 54

Table 19: Experiment 1 – Coefficients for Support Label in 'sickhillary' Dataset 54

Table 20: Experiment 1 – ‘US Economic Policy’ Feature Vector Performance Overview 55

Table 21: Experiment 1 – LR Coefficients for Support Label in 'US Economic Policy' Dataset 55

Table 22: Experiment 1 – ‘baghdadi dead’ Feature Vector Performance Overview: 56

Table 23: Experiment 2 – ‘sickhillary’ Feature Vector Performance Overview 57

Table 24: Experiment 2 – ‘baghdadi dead’ Feature Vector Performance Overview 58

Table 25: Experiment 2 – ‘death hoax’ Feature Vector Performance Overview 59

Table 26: Experiment 3 – Feature Vector Performance Overview 60

Table 27: Experiment 3 – LR Coefficients for POS 61

Table 28: Experiment 3 – LR Coefficients for Support Label 62

Table 29: Detailed Explanation of Main Configuration Object 67

Table 30: Detailed Explanation of Base Configuration Object 68

Table 31: Detailed Explanation of Worker Configuration Object 69

Table 32: Detailed Explanation of SubWorker Configuration Object 69

Table 33: Tweet Type to Integer Enumeration Mapping 70

Figure 1: Android Information Harvester GUI 22
Figure 2: Android Information Harvester Web Worker Notifications 22
Figure 3: Web Worker Architecture for Android Information Harvester 24
Figure 4: Data Storage Structure for Android Information Harvester 25
Figure 5: Bar Chart of Quantitative Metrics of Datasets 28
Figure 6: Bubble Chart of Selected Datasets 29
Figure 7: General Data Workflow of Tweets........ 35
Figure 8: Sample Usage of MongoDB Compass 38
Figure 9: Overview of Full Data Pipeline 39
Figure 10: Drop-down Box for Dataset Analysis 47
Figure 11: 'sickhillary' Pie Chart of Tweets by Type 48
Figure 12: 'sickhillary' Pie Chart of Tweets Labelled 48
Figure 13: 'sickhillary' Pie Chart of Tweets Labelled (Unique) 48
Figure 14: 'sickhillary' Pie Chart of Tweets Retweet Distribution 49
Figure 15:Drop-down Box for Experiment Result Analysis 49
Figure 16: Experiment 1 – 'trump cabinet' AFINN + SWN Classifier Performance 52
Figure 17: Experiment 1 – 'trump cabinet' POS Classifier Performance........ 53
Figure 18: Experiment 1 – ‘sickhillary’ AFINN + SWN Classifier Performance 54
Figure 19: Experiment 1 – ‘US Economic Policy’ AFINN + SWN Classifier Performance........ 55
Figure 20: Experiment 1 – ‘baghdadi dead’ AFINN + SWN Classifier Performance 56
Figure 21: Experiment 2 – ‘sickhillary’ Tf-idf Classifier Performance 57
Figure 22: Experiment 2 – ‘baghdadi dead’ Tf-idf Classifier Performance 58
Figure 23: Experiment 2 – ‘death hoax’ POS Classifier Performance 59
Figure 24: Experiment 3 – Tf-idf Classifier Performance 60
Figure 25: Experiment 3 – POS Classifier Performance 61
Figure 26: Experiment 3 – SWN Classifier Performance 62
Figure 27: Class Diagram for Main Configuration Object 67
Figure 28: Class Diagram for Base Configuration Object 67
Figure 29: Class Diagram for Worker Configuration Object 68
Figure 30: Class Diagram for SubWorker Configuration Object 69

Equation 1: Basic Law of Rumour 19
Equation 2: Formula for Min-Max Scaling Normalization 44

Equation 3: Formula for Normal Distribution Normalization 44
Equation 4: Formula for Log Normalization 44
Equation 5: Formula for Variable-Factor Normalization 44
Equation 6: Formula for Classifier Accuracy 46
Equation 7: Relative Scoring of Classifier Performance 51

1. Introduction

1.1. Objectives

This project aims to investigate the characteristics of rumours found on online social networks, such as the following: Size and frequency of messages, message propagation through the social network, and sentence structure of the messages.

This project encompasses studying literature related to rumours, collecting rumour and non-rumour messages, generating new features through feature engineering & selection, manual labelling of message sentiments, and the usage of sentiment analysis and machine learning techniques to automate the detection of rumours.

1.2. Background

The relatively recent development of the Internet in the history of humanity has resulted in a Copernican revolution in the way that information is distributed between humans in society. With the explosion of popularity of social media platforms such as Facebook and Instagram into the mainstream, social media platforms and other complementary digital news channels have increasingly encroached upon traditional print media. According to the Media Consumption Report: Q3 2014 [1], GlobalWebIndex found that people spent 5 to 6 hours online, compared to a paltry 2 to 3 hours spent on the next highest medium, Traditional TV. They also found that out of the 32 countries surveyed, peoples of 26 countries spent more time online than on traditional media.

Major news streams such as BBC, Al Jazeera, and New York Times have also augmented their presence and reach by utilising social media platforms, in response to the general trend that people are increasingly relying on digital mediums as their source of news. Sources of information on the internet include not only traditional news streams with digital avenues, but also from online discussion platforms such as Quora and Reddit, and ad-hoc discussions platforms such as Facebook and Twitter. These trends are also in line with the general evolution of the internet towards Web 2.0 – an internet that is characterised by user-generated content.

The increasing pervasiveness and influence of digital mediums have only exacerbated the importance of being able to ascertain the veracity of information on the internet. In recent history, the internet's role in spreading information of questionable veracity has been brought to the attention of the key stakeholders of society, such as governments, organisations, companies, and community leaders, recognising the risks of prevalent false information in society. Examples of such incidents include the spreading of false ideologies of Islam by ISIS, and the endemic spreading of rumours and false allegations by people of political allegiances in the recent US Presidential Election in 2016 [2].

Given the ever-increasing growth of information on the internet, devising semi-automated or automated means of verifying the veracity of information presents itself as an increasingly viable way of comprehensively addressing the issue.

1.3. Project Scope

The scope of each section of the project is as follows:

Project Planning

The project plan must comprehensively account for all tasks required to be completed for the project, accounting for the research direction of the project and the dependencies between tasks.

Information Harvester Development

The information harvester must be able to collect tweets from Twitter in automated and consistent manner.

Literature Review

Literature review will be undertaken for both academic fields of Computer Science and Social Sciences, to gain a mix of insights of how rumours are detected.

Feature Selection & Engineering

This project will be exploring datasets gathered through the collection of tweets from the Twitter API. Investigative work will be performed to engineer additional features based on existing tweet data, such as tweet type and tweet text. This section will also include manual labelling of tweets to indicate the tweet's sentiment (eg. is news, is rumour), which will be used as the target label for classification purposes.

Sentiment Analysis using Machine Learning Techniques

Work will be performed to engineer more features via the usage of sentiment libraries. Lastly, Machine Learning classifiers will be used to detect key trends in the dataset.

Testing, Results, and Discussion

The testing phase will report characteristics of the datasets collected and elaborate on the impact of the findings generated.

Full System Integration

The full system integration seeks to provide an easy-to-use web user interface for the user to easily discover insights from the datasets and experiment results generated.

Report Writing and Presentation

The key deliverables of the project - namely the Interim Report, Final Report, and Oral Presentation – will be completed as per requirements.

1.4. Project Resources

The following resources are utilized for the project:

Hardware	**Purpose**	**Cost**
MSI GL62 6QD	Development and Testing platform for all software tools utilized in this project. .	Free
LG L9 P760 Android Phone	Android phone for testing and deployment of Android Information Harvester.	Free

Table 1: Hardware Resources Utilized for Project

Software	**Purpose**	**Cost**
Development Platforms		
Intellij IDEA 2016	Integrated Development Environment (IDE) for all Python, HTML, JavaScript, and CSS work.	Free
Android Studio v2.1.3	Integrated Development Environment (IDE) for Android development.	Free
Android Software Development Kit (SDK)	Libraries for Android development.	Free
Python v2.7.11	Libraries for Python development.	Free
Enthought Canopy v1.7.4	Scientific Python development platform.	Free
Web Frameworks and Tools		
Django Web Framework v1.10.5	Web framework for Python.	Free
Bootstrap v3.3.7	Front-end framework for Web development.	Free
jQuery v1.12.4	JavaScript library for DOM manipulation and AJAX-handling.	Free
D3.js v4.7.2	JavaScript-based data visualization tool.	Free
Sentiment Analysis and Machine Learning Tools		
NLTK v3.2.2	Natural Language Processing library for Python.	Free
AFINN Sentiment Library v0.1 [3]	Sentiment Library used to score sentiments of sentences.	Free
NPS Chat Corpus [4]	Corpus containing posts from online chat services.	Free
scikit-learn v0.18.1	Machine Learning library for Python.	Free
pandas v0.19.2	Data manipulation and analysis library for Python.	Free
Database and Database Tools		

MongoDB v3.4.1	NoSQL Object-based database.	Free
MongoDB Compass v1.6.0	MongoDB Client for preliminary exploration of data in database.	Free
Productivity and Diagramming Tools		
Microsoft Word 2016	Used to write reports.	Student License
Microsoft Office 2016	Used to create diagrams.	Student License
StarUML v2.8.0	Used to create UML diagrams.	Free

Table 2: Software Resources Utilised for Project

1.5. Report Structure

The report is organized as follows:

Introduction

This section covers the broad objectives of the project, reasons for the importance of automated rumour detection, the scope of the project, resources utilised in the project, the structure of the report, and the planning and schedule of the project.

Literature Review

The literature review phase outlines the key learning points and methodologies gleaned from the study of papers from both Computer Science and Social Sciences. This section also includes how the literature review has influenced the general direction of the project.

Information Harvester Development

The information harvester development section includes the purpose of the software, the design considerations surrounding the software, and details the structure and features of the software.

Dataset Selection & Characteristics

This section outlines the qualitative and quantitative metrics used to evaluate the nature of the datasets collected.

Data Workflow: Collection, Import & Retrieval

This section describes the entire flow of how tweet data is collected, processed, and retrieved for further analysis.

Feature Selection & Engineering

This section describes the techniques and considerations behind selecting and generating features, how manual labelling of tweets has been performed, and how the feature vectors used in this project were generated.

Sentiment Analysis using Machine Learning Techniques

This section outlines the sentiment analysis libraries used to generate features for further testing, and describes the specifics of how machine learning classifiers have been used and tested in this project.

Full System Integration

This section describes the development and details of the web user interface used for discovering insights from the datasets and from the experiment results.

Testing, Results, and Analysis

This section outlines the three experiments and their objectives, testing methodology used in this project, and detailed breakdowns of the experiment results.

Conclusion

This section will highlight key findings from the experiments, conclude the study, and recommend potential improvements and research directions for the project.

1.6. Project Planning

The initial planning phase consists of understanding the exact specifications and expectations from the project supervisor. The key deliverables of this phase are the Project Schedule, and the understanding of the general research direction for this project. Attention was also paid as to the order of how the tasks in the project are undertaken, to avoid halts in progress due to task prerequisites or external scheduling limitations.

One time-saving measure undertaken was to complete the information harvester development at the early stages of the project, so that the incremental collection of tweets could happen in tandem with the remaining tasks.

1.7. Project Schedule

The following Gantt chart details the final project schedule.

	SEMESTER 1 2016/2017																			SEMESTER 2 2016/2017																
Date **Task**	AUG				SEP				OCT				NOV			DEC			JAN				FEB				MAR				APR			MAY		
Project Planning	▨	▨	▨												EXAM																		EXAM			
Information Harvester Development				▨	▨	▨																														
Literature Review							▨	▨	▨																											
Feature Selection & Engineering										▨	▨																									
Feature Selection R&D										▨																										
Feature Engineering R&D											▨																									
Sentiment Analysis using Machine Learning Techniques												▨	▨	▨		▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨								
Sentiment Analysis Techniques R&D													▨	▨		▨							▨	▨	▨	▨	▨	▨								
Machine Learning Techniques R&D																	▨	▨	▨				▨	▨	▨	▨	▨	▨								
Result Testing and Verification																	▨	▨	▨	▨			▨	▨	▨	▨	▨	▨								
Interim Report Writing																					▨	▨														
Full System Integration																											▨	▨	▨							
Final Report Writing																												▨	▨							
Oral Presentation																														▨	▨	▨				

Table 3: Final Project Schedule

The differences in the project schedule are attributed to the full system integration portion requiring lesser time than expected; the extra time was spent into performing more analyses and tests for the project.

The following Gantt chart details the initial project schedule.

	SEMESTER 1 2016/2017																			SEMESTER 2 2016/2017																
Date **Task**	AUG				SEP				OCT				NOV			DEC			JAN				FEB				MAR				APR			MAY		
Project Planning	▨	▨	▨												E X A M																		E X A M			
Information Harvester Development				▨	▨	▨																														
Literature Review							▨	▨	▨																											
Feature Selection & Engineering										▨	▨																									
Feature Selection R&D										▨																										
Feature Engineering R&D											▨																									
Sentiment Analysis using Machine Learning Techniques												▨	▨	▨		▨	▨	▨	▨	▨	▨	▨														
Sentiment Analysis Techniques R&D													▨	▨		▨																				
Machine Learning Techniques R&D																	▨	▨	▨																	
Result Testing and Verification																	▨	▨	▨	▨																
Interim Report Writing																					▨	▨														
Full System Integration																							▨	▨	▨	▨	▨	▨								
Final Report Writing																													▨	▨	▨	▨				
Oral Presentation																																		▨		

Table 4: Initial Project Schedule

2. Literature Review

The following findings were yielded after performing literature review for papers related to rumours in both academic fields of Computer Science and Social Sciences.

2.1. Literature Review in Computer Science

Following the 4-way typology outlined by Zhang et al. [5], the general direction of this project was decided to be focused on investigating Content-based methods of detecting rumours; content-based methods generate features based on the text and entities of a given message. As manually labelling was not feasible for large datasets, the search for automated sentiment analysis tools yielded several options, such as ANEW, AFINN, and LWIC. The ANEW tool showed good performance in a study about tweets related to Michael Jackson's death, as it could detect differences between a normal corpus and a corpus containing the tweets related to Michael Jackson's death [6]. Due to time constraints related to obtaining the ANEW dataset, AFINN was selected as one of the tools used for further analysis [3].

2.2. Literature Review in Social Sciences

According to Allport and Postman, the Basic Law of Rumour [7] consists of the following equation:

$$R \approx ia$$

Equation 1: Basic Law of Rumour

Where *R* represents the strength of a rumour, *i* represents the importance of the content of the message to a given individual, and *a* represents the ambiguity of evidence of the content. While the foregoing assertion alone may not have the academic rigor for it to be credible, the formula would broadly account for the pervasiveness of 'fake news' during the period of the US Presidential Election in 2016. The research findings from Heath et al. , which suggest that rumours and urban legends thrive on information and emotion selection [8], could account for many of the rumours during the 2016 US Presidential Elections, where key trending topics included tweets which were crafted to invoke disgust against their political opponents; examples include invoking feelings of disgust against Hillary Clinton due to her suspected corruption, or against Donald Trump due to his inexperience in political office or against his sexist stances. Additionally, many of the tweets related to the elections reflected the strong tribality exhibited by both Democrats and Republicans, which resulted in tweets from a given faction supporting the general narratives of the given faction whilst casting scathing remarks upon their opponents; this social phenomenon supports the findings of Wilson et al. [9] that gossip which is not self-serving usually reflected better on themselves and more harshly on their opponents.

The research findings from Rosnow et al. [10] may also account for the fact that there were many Dread Rumours in the form of tweets that were circulating during the Presidential Elections. Many people on both sides of the political spectrum, including people from both inside and outside of the US, projected their fears and disappointments on the future of the USA should either Hillary Clinton or Donald Trump get elected as president.

All in all, the foregoing studies lend further credence that the emotional aspect of a rumour is an integral component of a rumour. Therefore, this project's investigative work in rumour detection will be largely focusing on helping computers automatically detect the sentiment of a given message and determine whether it is a rumour.

3. Information Harvester Development

As collecting tweets is a key requirement for this project, software must be developed to collect tweets in a reliable and robust manner. The following sections elaborate on the various considerations made for choosing a suitable platform for the software, and elaborates on the features of the software.

3.1. Platform Considerations

Table 1 describes the various factors taken into consideration upon choosing a suitable platform for the development of the Information Harvester.

	Android Phone	**Single-board Computer**	**Desktop/Laptop**
Power Consumption	●●●	●●●	●○○
Languages Supported	●○○	●●○	●●●
One-time Platform Cost	●●○	●●●	●○○
Legend: More dots means better			

Table 5: Comparison of Platforms for Android Information Harvester

The platforms considered were desktops/laptops, single-board computers such as Raspberry Pi and CHIP, and Android phones. The factors used to compare the platforms were power consumption, languages supported, and the one-time platform cost. An ideal platform would be one that consumes the least power, supports many programming languages, and has a low one-time platform cost; this is to allow for the lowest costs related to running the information harvester, having the most flexibility of leveraging upon multiple programming languages if necessary, and being cheap to obtain.

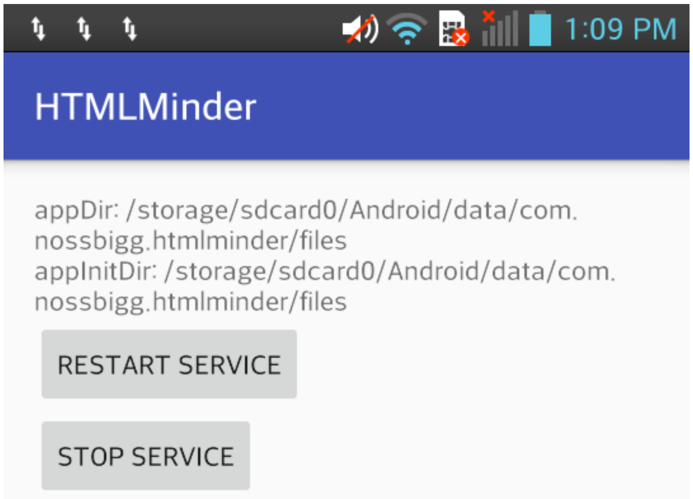
All things being equal, the single-board computer makes for the best platform, as it consumes a low amount of power, supports a reasonable number of languages, and has the lowest one-time platform cost of the three options. However, as I already have numerous unused Android phones, I can obtain an Android phone at zero cost. Thus, I have elected to use the Android phone platform for the development of the information harvester. Additionally, the Android phone option is much more portable than the other two options, as it does not require wall power and is a self-contained package.

3.2. Android Information Harvester Features

The software has the following features:

1. Android GUI Interface for starting and stopping tweet worker services

The application has two buttons to allow the user to start and stop the tweet worker services. Restarting the tweet worker services would be necessary to apply the changes to the configuration files. The GUI also displays the local directories chosen by the software, for ease of knowing where the tweet archives are stored at.

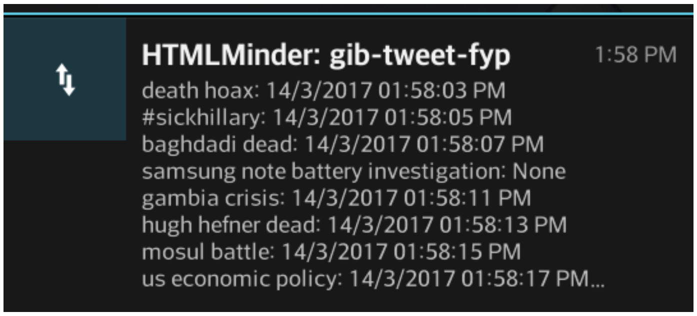


appDir: /storage/sdcard0/Android/data/com.nossbigg.htmlminder/files
appInitDir: /storage/sdcard0/Android/data/com.nossbigg.htmlminder/files

Figure 1: Android Information Harvester GUI

2. Live Worker Information via Notifications

The application will display a detailed breakdown of each worker service; every web worker will have a dedicated notification. The notifications will also display the timestamp of the last time each search query has retrieved a new tweet.



HTMLMinder: gib-tweet-fyp 1:58 PM

```
death hoax: 14/3/2017 01:58:03 PM
#sickhillary: 14/3/2017 01:58:05 PM
baghdadi dead: 14/3/2017 01:58:07 PM
samsung note battery investigation: None
gambia crisis: 14/3/2017 01:58:11 PM
hugh hefner dead: 14/3/2017 01:58:13 PM
mosul battle: 14/3/2017 01:58:15 PM
us economic policy: 14/3/2017 01:58:17 PM...
```

Figure 2: Android Information Harvester Web Worker Notifications

3. JSON-based configuration for easy configuration of software

The application's configuration parameters are modifiable through its JSON-based configuration files; details of the configuration files can be found in Appendix A: Android Information Harvester.

4. Automated collection and compilation of tweets into user-configurable folder names

The software will automatically collect tweets based on the parameters of the worker configuration files and store the tweets into text archives. Additional details of the automated collection and archival of tweets can be found in Section 5.2: Collection: Information Harvester.

5. Duplicate tweet reduction and network bandwidth optimisations via persistence of last collected tweet ID

The application will persist the last tweet ID that it has retrieved from a given worker, which leverages on the Twitter API's functionality to specify the last retrieved tweet ID, so that every API request will only return tweets that have been published after the specified tweet ID. This reduces the number of duplicate tweets retrieved, and reduces the overall network bandwidth used by the application. Additionally, users can also specify which fields should the worker check for contents before storing the response returned by the request; this ensures that only responses with non-empty datasets are stored, thereby saving storage space for the user.

6. Gzip compression for tweet archives for reduction of disk usage

Applying gzip compression on the collection tweet archives helps to dramatically reduce the total storage requirements of the application. Gzip compression has been found to have an average compression ratio of 2.99, thus yielding on average 70% in space savings [11]. In this application, there have been instances of 90% in space savings.

7. Additional metadata embedding functionality (eg. received timestamp, request parameters) into tweet archives

The application also adds additional metadata upon storing each successful request made by each web worker. The additional metadata persists important information made during the crafting of each request to the Twitter API; the additional metadata would be of interest if one would need to investigate and recreate previous search requests.

Field	**Purpose**
Parent element: htmlminder_custom_metadata	
timestamp_received_epoch	Timestamp of when the response of the request was received, in since epoch time format.
query_url	The URL endpoint to which the request was sent to.
querry_method	The method of the request.
parameters	The parameters of the request.

Table 6: Additional Metadata Stored per Request by Tweet Collection Application

3.3. Worker Architecture

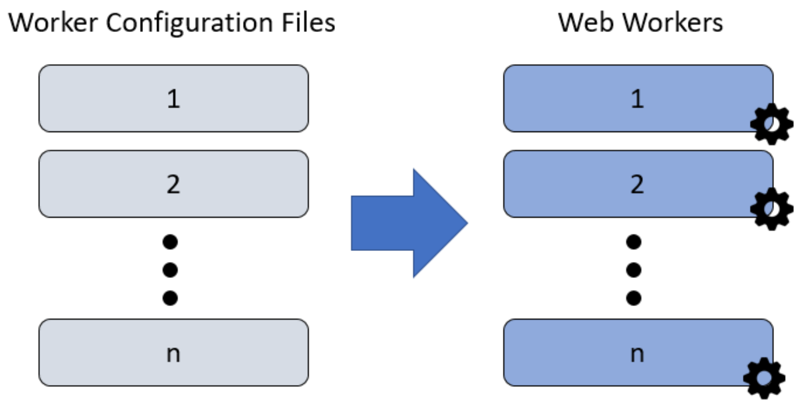
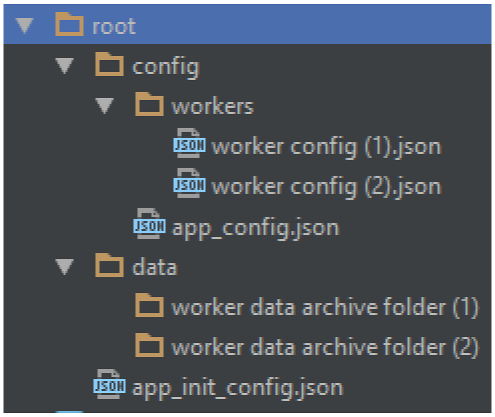


Figure 3: Web Worker Architecture for Android Information Harvester

The application automatically creates web workers based on the web worker configuration files. This allows users to easily create as many workers as they require to fit their needs.

3.4. Data Storage Structure



```
▼ root
    ▼ config
        ▼ workers
              worker config (1).json
              worker config (2).json
          app_config.json
    ▼ data
          worker data archive folder (1)
          worker data archive folder (2)
      app_init_config.json
```

Figure 4: Data Storage Structure for Android Information Harvester

The data storage is split into two sections.

1. config Directory

The config directory stores the key configuration files, which are detailed in the next section.

2. data Directory

The data directory stores tweets into individual folders for every web worker defined by the user.

3.5. JSON Configuration Files

There are three types of JSON configuration files in the application: The main configuration file, the base configuration file, and the worker file. Details of the configuration files can be found in Appendix A: Android Information Harvester.

3.6. Challenges

The challenges faced in this phase are mostly domain-specific with regards to developing software on the Android platform.

1. Inconsistent Storage Implementation across Android Versions

Different manufacturers of Android phone implement storage options for applications differently. For example, phones may or may not have external SD storage. This has direct implications to where the application will choose to store the tweet archives, as external SD storage is preferred so as not to use disk space on the phone.

The solution to this problem is to have two configuration files. The first configuration file resides in the Internal Storage, a directory that is guaranteed to exist, and it stores the path to the actual storage location that the app needs to access. The second configuration file, which resides in the External Storage, contains the user's configuration files. In the event where there is an absence of external SD storage, the application will then default to internal storage.

2. Tackling Android's Automated Task-killing

Android will automatically kill applications which it deems to be idle, and this poses a problem for the information harvester as it needs to collect tweets in a consistent and reliable manner.

This problem is addressed by using Persistent Services, where the service which is used to instantiate the web requests to retrieve tweets, is attached to a notification. However, if there are more than 3 persistent services, Android will kill enough persistent services till there are only three running persistent services; this can be addressed by collating web services together.

3. Unreliable MTP File Access

The MTP File Access implementation for Windows is unreliable as it exhibits erratic behaviour with regards to displaying and retrieving files. For example, newly created tweet archives may not be guaranteed to appear in MTP. Additionally, JSON files retrieved from the android device via MTP may be incomplete, resulting in errors when syncing JSON files back to the device after editing. This issue can be addressed by using the Android Debug Bridge (ADB) to pull and push files from and to the device. The ADB option, however, is not as user-friendly as the MTP option.

4. Dataset Selection & Characteristics

4.1. Selection Criteria

This project will be studying on datasets consisting of tweets gathered through the Twitter API.

The criteria for selecting datasets has been divided into two types of metrics:

Quantitative Metrics

1. Number of Tweets

The number of tweets must be more than 100. Having an insufficient number of tweets will weaken the conclusions derived from the findings made from the dataset.

2. Number of Unique Tweets

The number of unique must be more than 50. Having an insufficient number of unique tweets will weaken the conclusions derived from the findings made from the dataset.

Qualitative Metrics

1. News/Rumour Purity

The News/Rumour Purity metric measures how much of a dataset is either news or rumours, which varies depending on the search query used to build the dataset. A High measure indicates that the dataset consists of mostly news.

The three levels of ranks for this metric are as follows: Low, Medium, and High.

2. Contextual Purity

The Contextual Purity metric measures the number of contexts in a dataset, which varies depending on the search query used to build the dataset. A High measure indicates that the dataset consists of a low number of contexts and that the dataset's dominant context matches with the data that is sought by the researcher.

Using the foregoing metrics, we can select a mix of datasets with varying characteristics with regards to the metrics to test hypotheses and gain new insights from the datasets.

The three levels of ranks for this metric are as follows: Low, Medium, and High.

4.2. General Characteristics

The following tables and figures describe the datasets selected and their quantitative and qualitative scores.

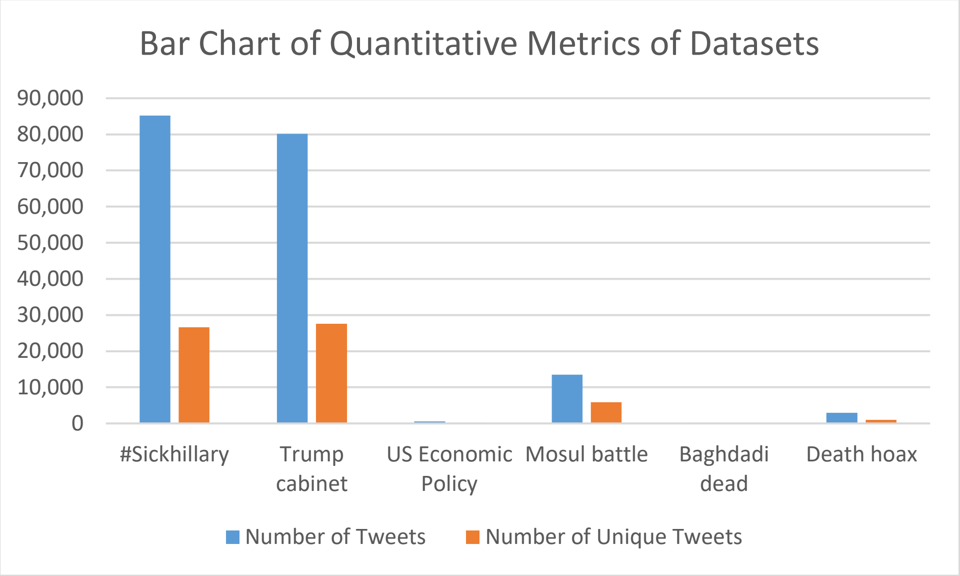


Figure 5: Bar Chart of Quantitative Metrics of Datasets

Dataset	Number of Tweets	Number of Unique Tweets
#Sickhillary	85,194	26,642
Trump cabinet	80,145	27,621
US Economic Policy	587	216
Mosul battle	13,483	5,886
Baghdadi dead	192	89
Death hoax	2,934	1,035

Table 7: Table of Quantitative Metrics of Datasets

Dataset	News/Rumour Purity	Contextual Purity
#Sickhillary	Low	Low
Trump cabinet	Low	Medium
US Economic Policy	Medium	High
Mosul battle	High	High
Baghdadi dead	Low	High
Death hoax	Medium	Low

Table 8: Table of Qualitative Metrics of Datasets

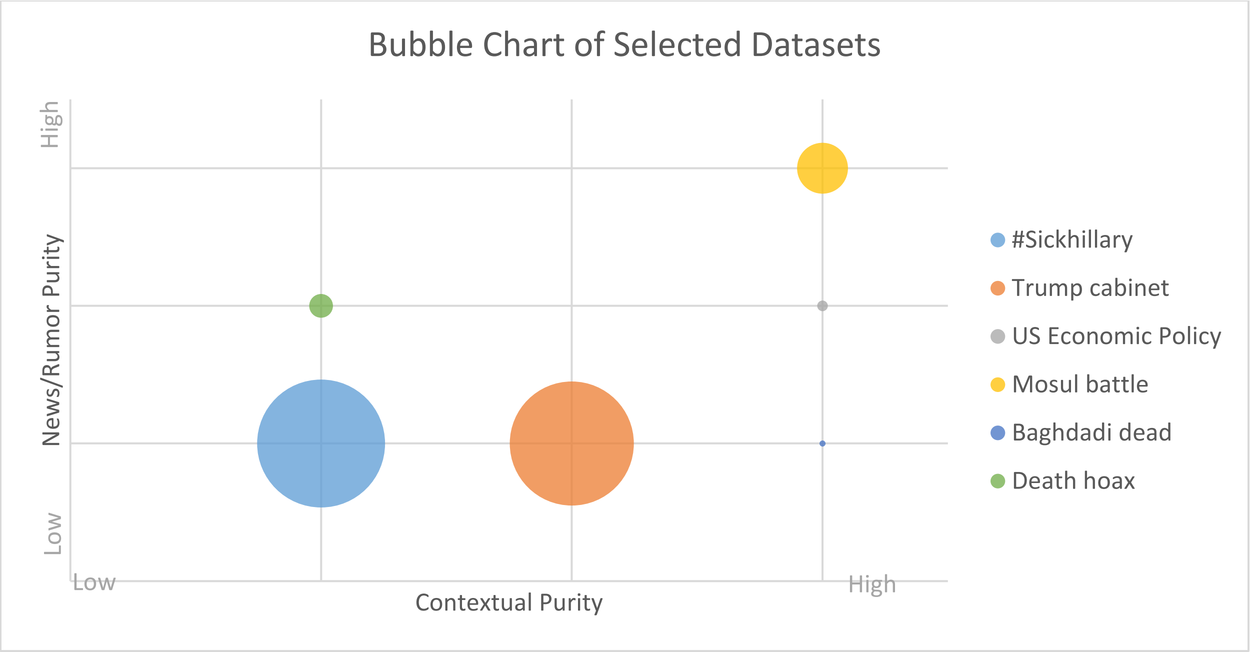


Figure 6: Bubble Chart of Selected Datasets

4.3. Detailed Characteristics

The following section describes each dataset in greater detail, explaining the contextual characteristics of each dataset and the rationale for each qualitative ranking.

4.3.1. '#Sickhillary'

Contexts

The main context of interest was to identify rumours which questioned the health of US Presidential Election candidate Hillary Clinton. The bulk of the rumours arose after a medical fiasco which happened at the 9/11 memorial event in 2016 at New York City, where she fell over as she was entering her vehicle and had to be held up by her bodyguards to enter her vehicle.

However, the study of the dataset collected with the dataset yielded two additional major contexts, in addition to the main context described above. The first additional major context were Twitter users who used the #sickhillary hashtag as a noun to describe Hillary Clinton, which resulted in many subcontexts that were ultimately unrelated to the main context. The second additional major context were tweets that promoted rumours about Hillary Clinton being corrupted, for example by casting aspersions on her character by suggesting that she was involved in corruption during her time in the US government and in The Clinton Foundation.

It is also useful to note of the strong sampling bias exhibited in this dataset: since the hashtag is manifestly anti-Hillary, only users who are expressly anti-Hillary would use this hashtag. Thus, it is expected that most of the tweets would have a negative stance towards Hillary Clinton.

News/Rumour Purity

The news/rumour purity of this dataset is Low, as there are large amounts of non-news content; the dataset is dominated by the opinions of Twitter users on Hillary Clinton.

Contextual Purity

The contextual purity of this dataset is Low, as there are multiple contexts within the dataset, with the first additional context exhibiting significant contextual differences vis-à-vis the main context of interest.

4.3.2. ‘Trump cabinet’

Contexts

The main context of interest was to identify news about Donald Trump’s cabinet, such as the cabinet’s policy announcements or the inauguration of new cabinet members. This information was particularly important given the immense amount of political uncertainty that came with the election of Donald Trump as President of the United States.

However, most the tweets collected were the opinions of Twitter users expressing their displeasure about the nominees being put up for Trump’s cabinet, such as Betsy DeVos and Steve Bannon.

Nevertheless, there were tweets about factual news on the developments surrounding Trump’s cabinet and policy decisions.

News/Rumour Purity

The news/rumour purity of this dataset is Low, as most of the dataset is about people’s negative opinions about Trump’s cabinet, and opinions cannot be considered as news, in a categorical sense.

Contextual Purity

The contextual purity of this dataset is Medium. Although there was a common context of Donald Trump, there were differences as to what they were exactly speaking about Trump, which in this instance exists in the form of two distinct contexts.

4.3.3. ‘US Economic Policy’

Contexts

The main context of interest was to identify news regarding the US economic policy.

The dataset of tweets was found to have an even mix of two contexts. The first context dealt with news surrounding updates in the US economic policy, which is in line with the main context of interest. The second context was about people's opinions and how they felt about the US economic policies and its implications.

News/Rumour Purity

The news/rumour purity of this dataset is Medium, as the dataset has a good mix of news in the form of news of US economic policy, and of opinions in the form of people expressing their thoughts on the US economic policy.

Contextual Purity

The contextual purity of this dataset is High, as most the tweets were centered around the topic of US economic policy.

4.3.4. 'Mosul battle'

Contexts

The main context of interest was to identify news reports of the ongoing Battle of Mosul taking place in Iraq.

The dataset was dominated by the main context of interest: news and updates on the progress of the ongoing war in Mosul. There were very few tweets that represented the opinions of people.

News/Rumour Purity

The news/rumour purity of this dataset is High, as most the tweets were about the ongoing war in Mosul, which easily constitutes as news.

Contextual Purity

The contextual purity of this dataset is High, as most tweets were about the main context of interest.

4.3.5. 'Baghdadi dead'

Contexts

The main context of interest was to identify rumours surrounding the status of Abu Bakr al-Baghdadi, who is the leader of the Islamic State of Iraq and the Levant, also known as ISIS.

The dataset showed a good mix of tweets that expressed agreement, disagreement, and questioned the rumour. There were also tweets that were completely unrelated to the main context of interest.

News/Rumour Purity

The news/rumour purity of this dataset is Low, as most of the tweets were opinions and not news.

Contextual Purity

The contextual purity of this dataset is High, as most of the tweets were talking about Baghdadi.

4.3.6. ‘Death hoax’

Contexts

The main context of interest was to identify death hoaxes circulating on Twitter.

The dataset exhibited two major contexts. The first major context was around tweets that disproved death hoaxes, which is closely related to the main context of interest. The second major context was tweets that were completely unrelated to the main context of interest.

News/Rumour Purity

The news/rumour purity of this dataset is Medium, as the first major context constitutes as news. However, the second major context consists of unrelated content, which is not news.

Contextual Purity

The contextual purity of this dataset is Low, as although the first major context is pure by itself, the second major context consists of unrelated content with many contexts that differ from the main context of interest.

4.4. Challenges

The following challenges were encountered whilst analyzing the datasets:

1. Uncertainty in Identifying Major Contexts in Datasets

The task of identifying major contexts in each dataset has a considerable amount of uncertainty, due to the qualitative nature of the task which leverages upon the domain knowledge of the labeler, and the possibility of human error during the process. This risk may be mitigated by Topic Modeling techniques such as Latent Dirichlet Allocation (LDA) and Latent Sentiment Analysis (LSA), or by having more human coders and ensuring that the Kappa measure between the human coders is low.

2. Handling Large Amounts of Data

It is infeasible to manually analyze every tweet, especially for large datasets due to time constraints. Therefore, prior to the manual analysis, tweets were sorted in descending order per the number of retweets for each tweet. By analyzing the most popular retweets, which are representative of the major contexts for a given dataset, we could quickly evaluate the dataset vis-à-vis the metrics without spending copious amounts of time analyzing every single tweet.

4.5. Insights

The key insights from the analysis of the datasets are as follows:

1. Using the Qualitative Measures in Tandem: News/Rumour Purity and Contextual Purity

The qualitative measures, News/Rumour Purity and Contextual Purity helps one easily identify how to treat a given dataset.

For example, a dataset with high News/Rumour Purity requires less data cleaning and can be more readily used as a training corpus for a classifier to identify news. Consequently, a dataset with a low News/Rumour Purity requires more preprocessing to separate the news tweets from the rumour tweets before it can be meaningfully used. However, an exception would be a dataset that consists mostly of rumours – such a dataset requires little data cleaning and can be used as a corpus for a classifier to identify rumours.

As for Contextual Purity, a dataset with high Contextual Purity would require less effort in having to separate the dataset into the different contexts identified; the reverse holds true.

Therefore, researchers can use this as a tool to can quickly classify the characteristics of datasets and design the appropriate strategies to process and utilize the dataset. Additionally, researchers can also use this to classify existing datasets to see whether their findings in one type of dataset can be generalized to similar and/or dissimilar datasets.

2. Accounting for Ambiguity in Search Queries

Search queries to the Twitter API may return tweets that are completely unrelated to what is the main context of interest identified, due to possibly ambiguous words used in the search queries. Therefore, researchers must be cognizant of such an issue when designing a search query and avoid it if possible. Possible mitigation strategies include using platform-specific search query modifiers (eg. “” in Google

Search query to specify exact match), or by using keywords unique to a concept. Researchers can also search online for a list of ambiguous words to aid their efforts in avoiding ambiguous words when constructing search queries.

5. Data Workflow: Collection, Import & Retrieval

5.1. Overview

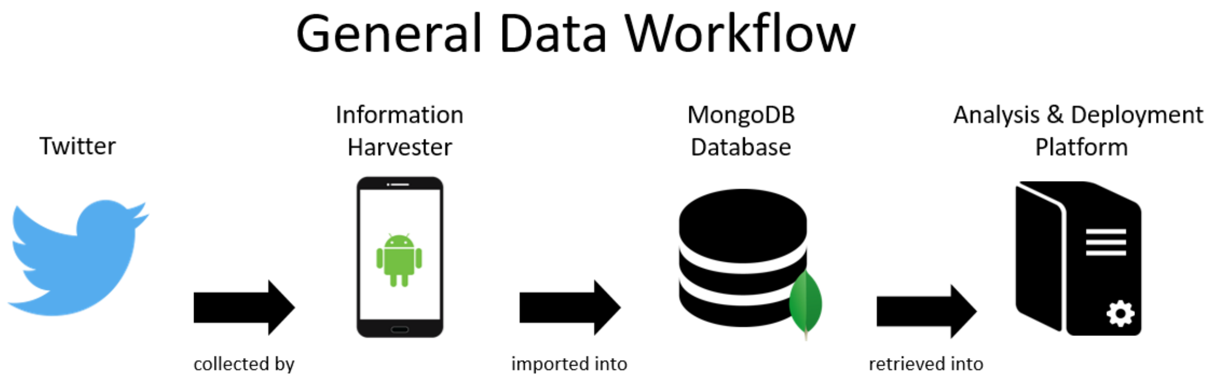


Figure 7: General Data Workflow of Tweets

The general data workflow consists of 4 elements with 3 stages of data transfer between the elements. The 4 elements are the following:

1. Twitter

Twitter is a social network platform where participants can make posts and interact with fellow participants using hashtags, quote retweets, retweets, and comments. The datasets used in this project are based on tweets collected from Twitter.

2. Information Harvester

The Information Harvester collects tweets from Twitter based on search queries by the user. More details of the information harvester can be found in Section 3: Information Harvester Development.

3. MongoDB Database

The MongoDB Database stores tweets from the information harvester. Tweets are put through a data cleaning process and are imported into the MongoDB Database.

4. Analysis & Development Platform

The Analysis & Development Platform is where all further in-depth analysis and work described in Sections 6, 7, and 9 are performed.

The three 3 stages of data transfer are described in the rest of this chapter.

5.2. Collection: Information Harvester

The Information Harvester collects tweets from the Twitter API using the following process:

1. Fully Receive Response from Request

The response for the search query from Twitter is fully received by the Information Harvester. More details about the response format can be found at the Twitter API website.

As of writing, the response format is as follows:

Field	**Description**
statuses	Contains matching tweets for the search query
search_metadata	Contains additional metadata for the search query

Table 9: Response Format for Twitter API Search Result

2. Add Custom Metadata to Request

Additional custom metadata is appended to the response data; details of the custom metadata can be found in Section 0: Android Information Harvester Features.

3. Append Request to Text File

The web worker saves the response data by appending it as a single line to the corresponding text file associated with the day that the response is received.

4. Compress Old Text Files with Gzip

The web workers will compress any past text archives with the gzip compression scheme. This helps to save storage space on the Android device.

More details of the information harvester with regards to data storage structure and gzip compression can be respectively found in Section 3.4: Data Storage Structure and Section 0: Android Information Harvester Features.

5.3. Import: Database

The tweet import process involves the decompression of tweet archives from the Information Harvester and generates two MongoDB collections per dataset: one for storing tweets, the another for storing relationships between tweets.

The tweet import process includes the following steps:

1. Remove Tweet Duplicates

A tweet duplicate is defined as a tweet that has the same tweet ID as an existing tweet. All duplicates are excluded from the import process.

2. Label Tweet Type of Tweets

All tweets are labelled by their type with the following algorithm:

- If tweet has ‘retweet_status’ field, tweet type is Retweet.
- If tweet ‘is_quote_status’ field is True, tweet type is Quote Retweet.
- Else, tweet type is Normal.

More details about this can be found at Appendix B-1: Tweet Type to Integer Enumeration Mapping.

3. Analyze and Persist Tweet Relationship Information

From the tweet types of each tweet, the relationship information between tweets are generated. This step also helps us identify the set of unique tweets in each dataset, which will be leveraged up to speed up all subsequent labelling processes and help with reducing sampling bias when training machine learning classifiers.

4. Generate Tweet Sentiment of Tweets

Tweet sentiments are generated from the tweets. More information about how these sentiments are generated can be found in Section 7.1: Sentiment Analysis Techniques.

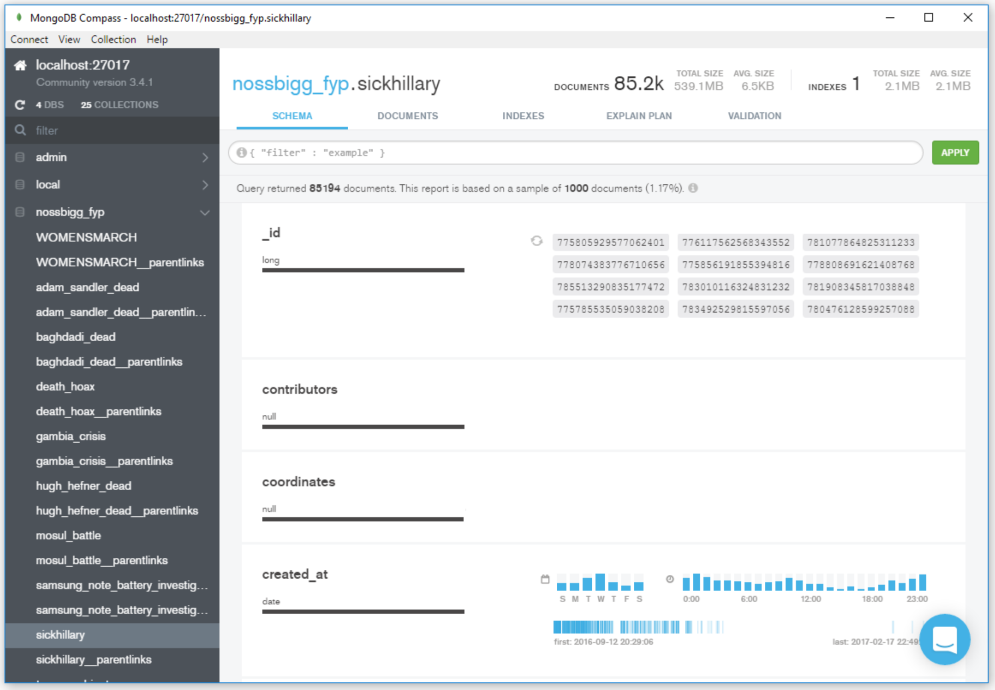
5.4. Retrieval: Database

Tweets from the database are retrieved by two applications and for the following purposes:

1. Analysis & Deployment Platform

The Analysis & Deployment Platform retrieves tweets from the database using the PyMongo client, taking full advantage of the search and filter functionalities of MongoDB.

2. MongoDB Compass



MongoDB Compass - localhost:27017/nossbigg_fyp.sickhillary

nossbigg_fyp.sickhillary

DOCUMENTS 85.2k TOTAL SIZE 539.1MB AVG. SIZE 6.5KB

INDEXES 1 TOTAL SIZE 2.1MB AVG. SIZE 2.1MB

{ "filter" : "example" }

Query returned **85194** documents. This report is based on a sample of **1000** documents (1.17%).

_id
long

775805929577062401 776117562568343552 781077864825311233
778074383776710656 775856191855394816 778808691621408768
785513290835177472 783010116324831232 781908345817038848
775785535059038208 783492529815597056 780476128599257088

contributors
null

coordinates
null

created_at
date

first: 2016-09-12 20:29:06

last: 2017-02-17 22:49

Figure 8: Sample Usage of MongoDB Compass

MongoDB Compass is a software that allows users to easily connect to a MongoDB database and to analyze the general characteristics of the collections available in the database.

6. Feature Selection & Engineering

6.1. Feature Selection

The key feature used for further analysis is the text field of the tweet, which is used for generating new features, as described in detail in Section 7.1: Sentiment Analysis Techniques.

6.2. Feature Engineering

6.2.1. Overview

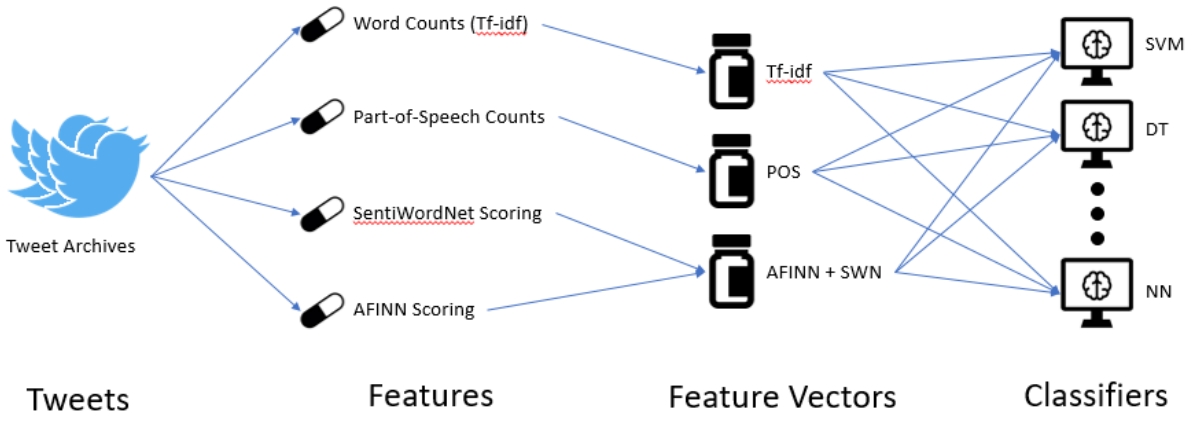


Figure 9: Overview of Full Data Pipeline

The purpose of feature engineering is to generate new features from existing data, so that these new features can be used in tandem with existing features for classifiers to interpret patterns from.

Apart from the features generated in Section 7.1: Sentiment Analysis Techniques, two extra features have been generated – Bag-of-Words Model with Tf-idf, and Manual Labeling

6.2.2. Word Counts (Tf-idf)

Using a Bag-of-Words model, tf-idf values for each term are calculated using scikit-learn's TfidfVectorizer model. However, prior to inserting the tweets into the vectorizer, the tweets are preprocessed using the following steps:

1. Remove HTTP links
2. Remove User References (eg. @realDonaldTrump)
3. Remove Hashtags
4. Remove Stopwords

5. Lemmatize Tokens
6. Remove tokens that are shorter than 3 characters

6.2.3. Manual Labeling

Manual labeling is performed to help to determine whether a certain tweet is congruent to the main context of interest of the dataset.

6.2.3.1. Workflow

The workflow of the manual labeling process is as follows:

1. Generate CSV Document for Labeling

The CSV document contains a collection of unique tweets sorted by descending order per the number of retweets for each tweet. This step speeds up the subsequent labeling process by helping the labeler identify the most popular tweets.

2. Label CSV Document

Only the most popular tweets are labeled, as the most popular tweets in a dataset are representative of the major contexts in a dataset. The labeler will then manually analyze each popular tweet and add the label category to the tweet. The labeling scheme is defined in the following section.

3. Import CSV Labeling Document

The labels in the CSV document are imported into the database.

6.2.3.2. Manual Labeling Schemes

There are two schemes that are used for the manual labeling process: one for rumour-centric datasets and another for news-centric datasets.

The following table details the specific scheme used for each dataset.

Rumour-centric Datasets		
#Sickhillary	Baghdadi dead	Death hoax
News-centric Datasets		
Mosul Battle	US Economic Policy	Trump cabinet

Table 10: Labeling Scheme for Datasets

6.2.3.2.1. Labeling Scheme for Rumour-Centric Datasets

The scheme for rumour-centric datasets labels tweets in 4 distinct categories:

Support (s)

The Support label is for tweets that support a rumour explicitly or implicitly (eg. using rumour to augment another opinion)

Deny (d)

The Deny label is for tweets that deny or disproves a rumour.

Neutral (n)

The Neutral label is for tweets that neither supports nor rejects a rumour. Such rumours may also occasionally express doubt about the rumour.

Unrelated (u)

The Unrelated label is for tweets that are unrelated to the rumour.

The following table details examples for each label.

Dataset	**Tweet ID**	**Label**	**Tweet Text**
Sickhillary	778806373312176128	Support	RT @ThePatriot143: Kimmel: #SickHillary Conspiracy Theories Would Be Harder to Believe If They Didn't Actually Come True #ThanksObama https…
Death hoax	827268557868199937	Deny	Hoax Exposed: Muslim Man Blamed Trump For Iraqi Mother's Death » Alex Jones' Infowars: There's a war on for your... https://t.co/5bOrJ8daCD
Baghdadi dead	831538243795558403	Neutral	Isis commanders killed in Iraqi air strikes targeting Baghdadi, but leader&amp;apos;s fate remains unknown https://t.co/oqlyJks1e5
Death hoax	827401608690438144	Unrelated	We have a new record coming. We would love for you to pre-order with us at @Bandcamp . Thank you https://t.co/2M9cdJ54xC

Table 11: Example Tweets for Labeling Scheme for Rumour-centric Datasets

6.2.3.2.2. Labeling Scheme for News-Centric Datasets

The scheme for news-centric datasets labels tweets in 3 distinct categories:

Support (s)

The Support label is for tweets that constitute as news.

Deny (d)

The Deny label is for tweets that are not news (eg. opinions of users).

Unrelated (u)

The Unrelated label is for tweets that are unrelated to the context of interest.

The following table details examples for each label.

Dataset	Tweet ID	Label	Tweet Text
Mosul battle	824194119413223424	Support	Drone of Iraqi militants strike Abrams main battle tank in Mosulhttps://t.co/Od4p6XkT4N https://t.co/VSZsuz0Po8
US Economic Policy	836282821454942209	Deny	RT @Chellaney: Just as US policy created the jihadist scourge, it created a rules-violating monster by aiding China's economic rise https:/…
Trump cabinet	831774593962672128	Unrelated	RT @HamiltonElector: If Trump's colluded with Russia he, his VP, and his cabinet are all illegitimate. There'd have to be a unity govt til…

Table 12: Example Tweets for Labeling Scheme for News-centric Datasets

7. Sentiment Analysis using Machine Learning Techniques

7.1. Sentiment Analysis Techniques

Sentiment Analysis Techniques is a specific subset of Feature Engineering which seeks to generate features based on key information from a given text.

The following techniques were utilized in this project:

1. NLTK Part-of-Speech Tagging

Part-of-Speech (POS) tagging involves identifying the syntactic functions of the words in a sentence. The reference corpus utilized for this task is the NPS Chat Corpus [4], which consists of 10,576 posts from several online chat services. As the corpus is derived from an online source, it is a suitable corpus for the task, as it is more likely to share syntactic similarities with the collected tweets than other corpuses based off non-online origins.

The NPS Chat Corpus utilizes the Penn Treebank POS tagset [12]. However, in the interest of simplicity, the Universal POS tagset outlined in the NLTK book [13] is preferred for this project's usage. Due to a lack of a common approach to map the Penn Treebank POS tagset to the Universal POS tagset, a mapping was devised for this purpose. The two tagsets and the mapping can be found in Appendix A: Sentiment Analysis Techniques.

2. NLTK SentiWordNet Scoring

The SentiWordNet 3.0 [14] is a sentiment analysis tool that generates scores based on the sentiment of the word. There are three types of scores: Positivity, Negativity, and Objectivity (Neutrality).

As the SentiWordNet contains multiple scorings per word, only the first scoring of a given word is considered in this project.

3. AFINN Scoring

The AFINN [3] library is a sentiment analysis tool that generates scores based on the sentiment of a given word or series of words. AFINN is particularly useful in this project, as AFINN was designed for sentiment analysis in microblogs, and tweets constitute as microblog material.

7.2. Real-Time Considerations

In terms of feature normalization for continuous variables, there are a few methods available:

- Min-Max Scaling Normalization: Normalize the feature space into [0 1] bounds using the following formula:

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)}$$

Equation 2: Formula for Min-Max Scaling Normalization

Where x is the dataset, x_i is the value of the data, and z_i represents the i^{th} normalized data.

- Normal Distribution Normalization: Normalize the feature space by translating all values to the Z-scores of the values using the following formula:

$$z_i = \frac{x_i - \mu}{\sigma}$$

Equation 3: Formula for Normal Distribution Normalization

Where x is the dataset, z_i represents the Z-score of the i^{th} data, x_i is the value of the data, μ is the mean of the dataset, and σ is the standard deviation of the dataset.

- Log Normalization: Normalize the feature space by translating all values through a log transformation using the following formula:

$$z_i = log(x_i)$$

Equation 4: Formula for Log Normalization

Where x is the dataset, z_i represents the log-normalized value of the i^{th} data, x_i is the value of the data, and log is the logarithm function.

- Variable-Factor Normalization: Normalize the feature space based on a variable factor that differs from one observation to another. The function is as follows:

$$z_i = \frac{x_i}{len(t_i)}$$

Equation 5: Formula for Variable-Factor Normalization

Where x is the dataset, z_i represents the normalized value of the i^{th} data, x_i is the value of the data, $len(t_i)$ represents the number of tokens derived from a tweet's text field.

In the context of a real-time classifier, utilizing a min-max scaler or a normal distribution normalization would be much more computationally expensive as compared to log normalization or variable-factor normalization. In the case of normal distribution normalization, all values would need to be recomputed in the event of a new training observation. In the case of the min-max scaler, if a new training observation is not within the current minimum and maximum bounds, all values would need to be recomputed.

However, while log transformation does not require re-computing of all past values when a new observation appears, it may not be suitable for some cases as it 'flattens' the data dramatically. Thus, it is only appropriate to use it if the feature has a large range of values.

For this project, the Variable-Factor Normalization technique is used to normalize the scores generated in Section 7.1: Sentiment Analysis Techniques.

7.3. Machine Learning Techniques

The machine learning classifiers utilized in this project are provided by the scikit-learn library. The following classifiers and the corresponding initialization parameters were used:

Classifier	Initialization Parameters
Support Vector Machine	Random initialization value
Decision Tree	Random initialization value
Linear Regression	Random initialization value
Naïve Bayes (Bernoulli)	Default
Random Forest Regressor	Default
Gradient Boosting Regressor	Default

Neural Network	Random initialization value
	Solver = 'lbfgs'
	Alpha = 1e-5
	Hidden_layer_sizes=15

Table 13:Machine Learning Classifiers and Initialization Parameters Used

7.3.1. Model Training

The models were trained using an 80/20 train/test ratio. Each classifier is trained using the train dataset first, and then tested for its accuracy with the test dataset. Stratified sampling was not used, in order to simulate real-world conditions where tweets may not always be created or sampled with fixed distribution percentages per label class.

The target variable is the manual label field (eg. Support, Deny, Neutral, Unrelated).

Three different feature vectors were constructed using the features generated from the feature engineering and sentiment analysis phases; the three feature vectors are as follows:

1. Tf-idf Feature Vector

The 'tf-idf' feature vector consists of tf-idf vector space for each tweet.

It is important to note that the vectorizer must be trained only with the train dataset, and not together with the test dataset; doing so would result in an incorrectly high accuracy for the classifier.

2. POS Feature Vector

The 'POS' feature vector consists of values that were generated during the NLTK Part-of-Speech Tagging process.

3. AFINN + SWN Feature Vector

The 'AFINN + SWN' feature vector consists of a straightforward concatenation of values generated from the AFINN scoring and NLTK SentiWordNet scoring processes for each tweet.

7.3.2. Model Testing

The accuracy of a classifier is as follows:

$$ACC = (TP + TN)/(P + N)$$

Equation 6: Formula for Classifier Accuracy

Where ACC is accuracy, TP is true positives, TN is true negatives, P is the number of positive observations, and N is the number of negative observations.

For this project, only the accuracy metric is used to evaluate classifier performance.

8. Full System Integration

8.1. Back-End Development

The back-end is developed using the Django Python Web Framework, which serves data to the front end for further visualization. The following endpoints were created:

URL	**Purpose**
/fyp/	Homepage for project. Displays user interface for users to explore datasets and test results.
/fyp/api/datasets_info	Web service endpoint that serves key details about each dataset.
/fyp/api/ml_tests_info	Web service endpoint that serves ML test details.

Table 14: Web Endpoints for Web Deployment

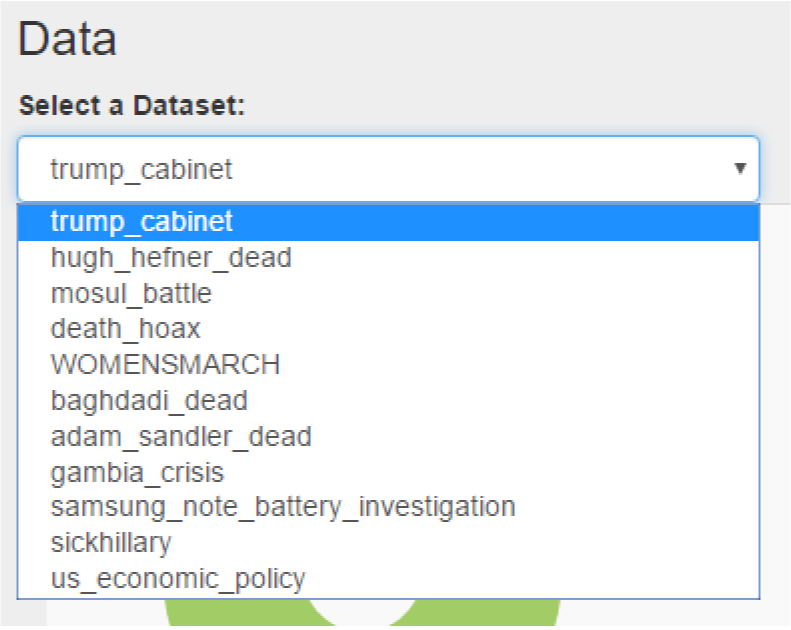
8.2. Front-End Development

Bootstrap was used as the front-end framework, jQuery was used to retrieve and process data from web service endpoints, and d3.js was used for the drawing of graphs and charts to illustrate key details for the datasets and testing results.

8.2.1. Visualization of Datasets

The combination of pie charts and bar charts helps users easily evaluate the various characteristics of each dataset.

Users are able analyze the dataset by selecting on the dataset name from the drop-down box.



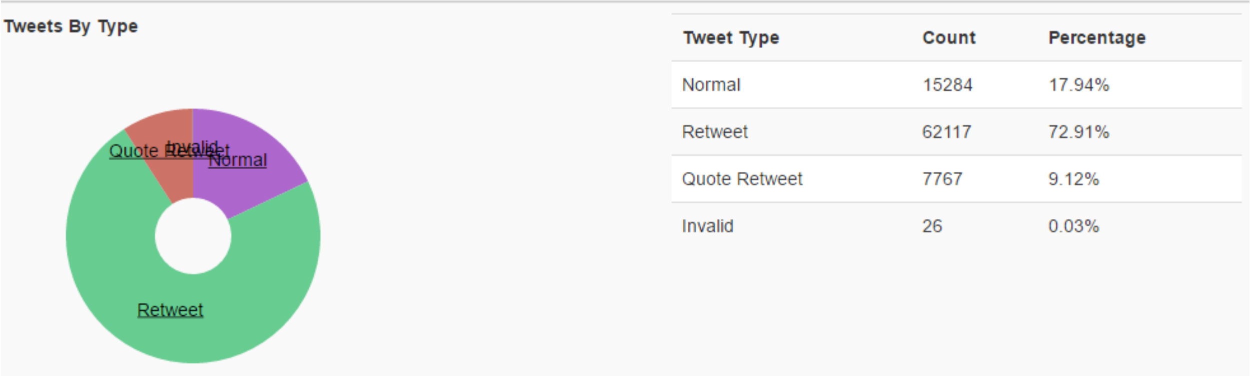
Data

Select a Dataset: trump_cabinet

Figure 10: Drop-down Box for Dataset Analysis

The following examples illustrate the various types of data visualizations used.

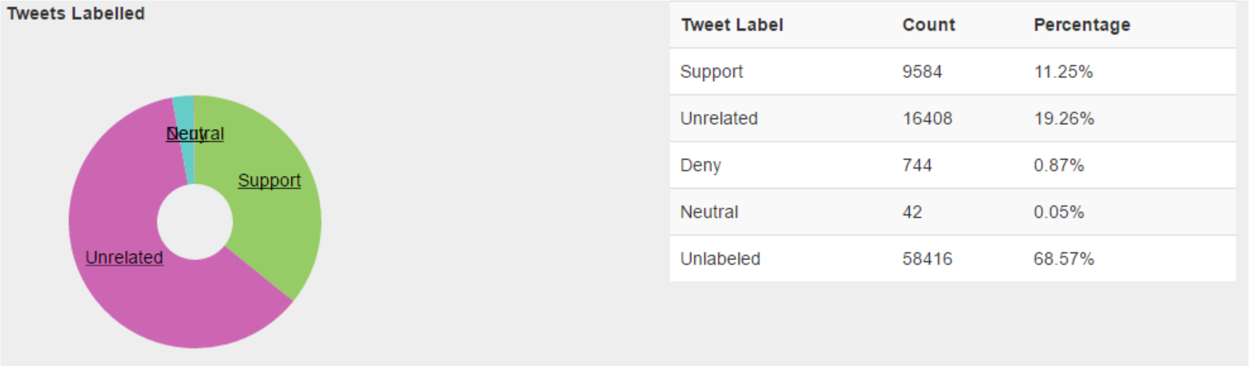
- Tweets by Type


Tweets By Type

Tweet Type	Count	Percentage
Normal	15284	17.94%
Retweet	62117	72.91%
Quote Retweet	7767	9.12%
Invalid	26	0.03%

Figure 11: 'sickhillary' Pie Chart of Tweets by Type

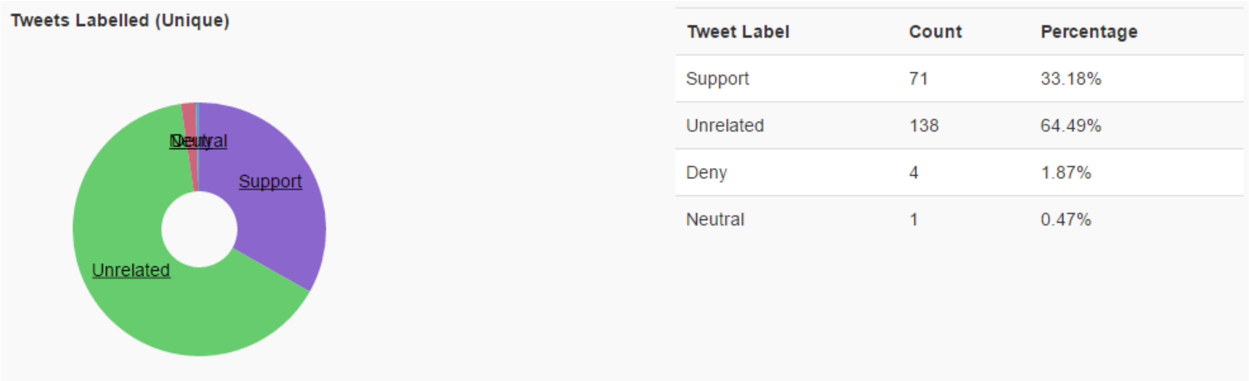
- Tweets Labelled


Tweets Labelled

Tweet Label	Count	Percentage
Support	9584	11.25%
Unrelated	16408	19.26%
Deny	744	0.87%
Neutral	42	0.05%
Unlabeled	58416	68.57%

Figure 12: 'sickhillary' Pie Chart of Tweets Labelled

- Tweets Labelled (Unique)


Tweets Labelled (Unique)

Tweet Label	Count	Percentage
Support	71	33.18%
Unrelated	138	64.49%
Deny	4	1.87%
Neutral	1	0.47%

Figure 13: 'sickhillary' Pie Chart of Tweets Labelled (Unique)

- Tweet Retweet Distribution

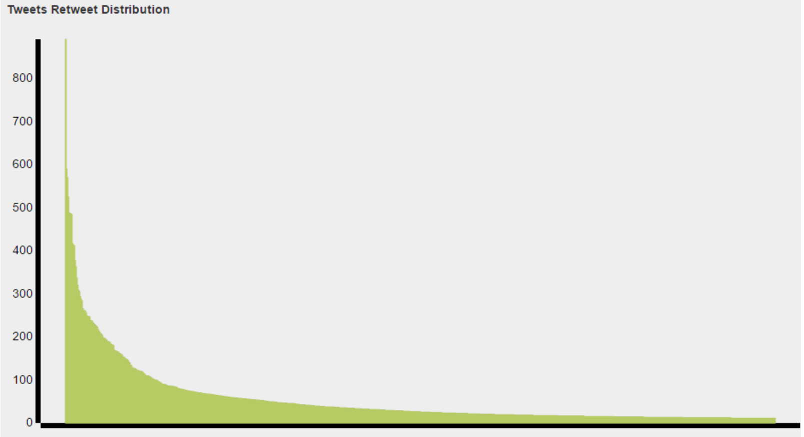
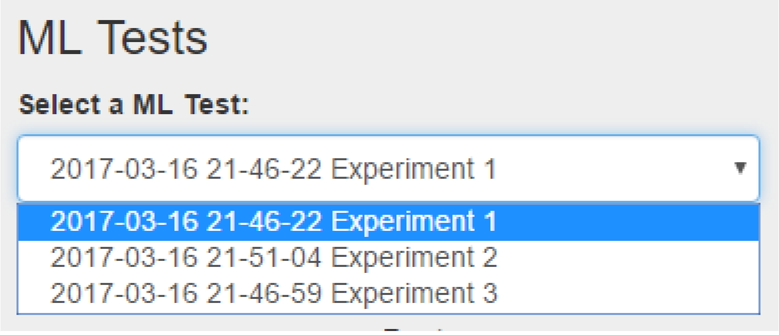


Figure 14: 'sickhillary' Pie Chart of Tweets Retweet Distribution

8.2.2. Visualizations of Testing

Line charts have been used extensively to evaluate the relative performance of classifiers against one another in each test set.

Users are able analyze experiment results by selecting on the experiment name from the drop-down box.



ML Tests

Select a ML Test:

2017-03-16 21-46-22 Experiment 1 ▼

2017-03-16 21-46-22 Experiment 1
2017-03-16 21-51-04 Experiment 2
2017-03-16 21-46-59 Experiment 3

Figure 15:Drop-down Box for Experiment Result Analysis

The multiline graph visualizations and tables illustrated to visualize experiment results can be found in Section 0: Results & Analysis.

9. Testing, Results & Analysis

9.1. Experiments

Experiment 1: What are the general sentiment profiles of the datasets?

This experiment seeks to investigate if tweets are linearly separable by Logistic Regression based on the Sentiment Scores of each tweet. By examining the weights in a Logistic Regression classifier, we can discover what are the most pertinent features for each dataset.

This experiment will only use the AFINN + SWN and POS feature vectors.

Experiment 2: How well can rumours and non-rumours be separated in rumour-centric datasets?

This experiment seeks to investigate if rumours and non-rumours can be separated from one another within rumour-centric datasets.

Experiment 3: How well can rumours and non-rumours be separated using all datasets?

This experiment seeks to identify how well can rumours and non-rumours be separated from one another, using all 6 datasets in unison.

9.2. Testing Methodology

Each experiment consists of 100 iterations each, where each train/test mix will be randomized per iteration. In each iteration, each train/test dataset will be used to train and test the 7 machine learning classifiers. After the 100 iterations, statistical methods are used to generate the following metrics for each type of classifier:

- Minimum accuracy
- Maximum accuracy
- Mean accuracy
- Median accuracy
- Standard deviation of accuracy scores
- 25th percentile of accuracy scores
- 75th percentile of accuracy scores

To choose the best classifier for each feature vector, the classifier with the highest score is selected; the scoring scheme for each classifier is as follows:

$$score(c)_i = w_{min}\mathrm{r}ank\_min(c)_i + \mathrm{rank\_median}(c)_i w_{median} + \mathrm{rank\_stddev}(c)_i w_{stddev}$$

Equation 7: Relative Scoring of Classifier Performance

Where c is the classifier,and i is the score for the classifier. w are the weights for each of the factors considered. The rank_min, rank_median, and rank_stddev functions calculate the rank positon of each classifier based on each classifier's minimum accuracy, median accuracy and standard deviation of accuracy scores. For example, if the classifier has the best minimum accuracy, it will have rank score of 6.

After the best classifier is selected for each feature vector, the feature vector with the best classifier accuracy is selected as the best feature vector for the experiment.

9.3. Results & Analysis

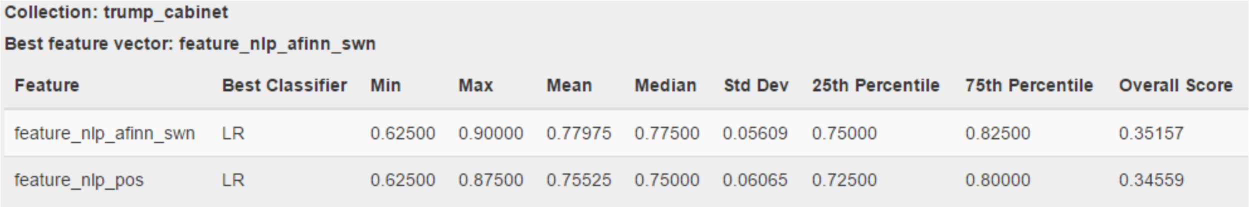
Only datasets which have at least two distinct label classes with an acceptable number of observations for each class are examined. This is to ensure that the classifier has enough observations from each class to learn patterns from.

Due to the verbosity of test results, only the significant findings have been outlined.

Experiment 1: What are the general sentiment profiles of the datasets?

Due to the nature of Logistic Regression (LR), there is no R-squared score to perform the goodness-of-fit evaluation. Hence, only the accuracy and coefficients will be interpreted.

For the ‘trump_cabinet’ dataset, both feature vectors showed good accuracy.



Collection: trump_cabinet

Best feature vector: feature_nlp_afinn_swn

Feature	Best Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
feature_nlp_afinn_swn	LR	0.62500	0.90000	0.77975	0.77500	0.05609	0.75000	0.82500	0.35157
feature_nlp_pos	LR	0.62500	0.87500	0.75525	0.75000	0.06065	0.72500	0.80000	0.34559

Table 15: Experiment 1 – 'trump cabinet' Feature Vector Performance Overview

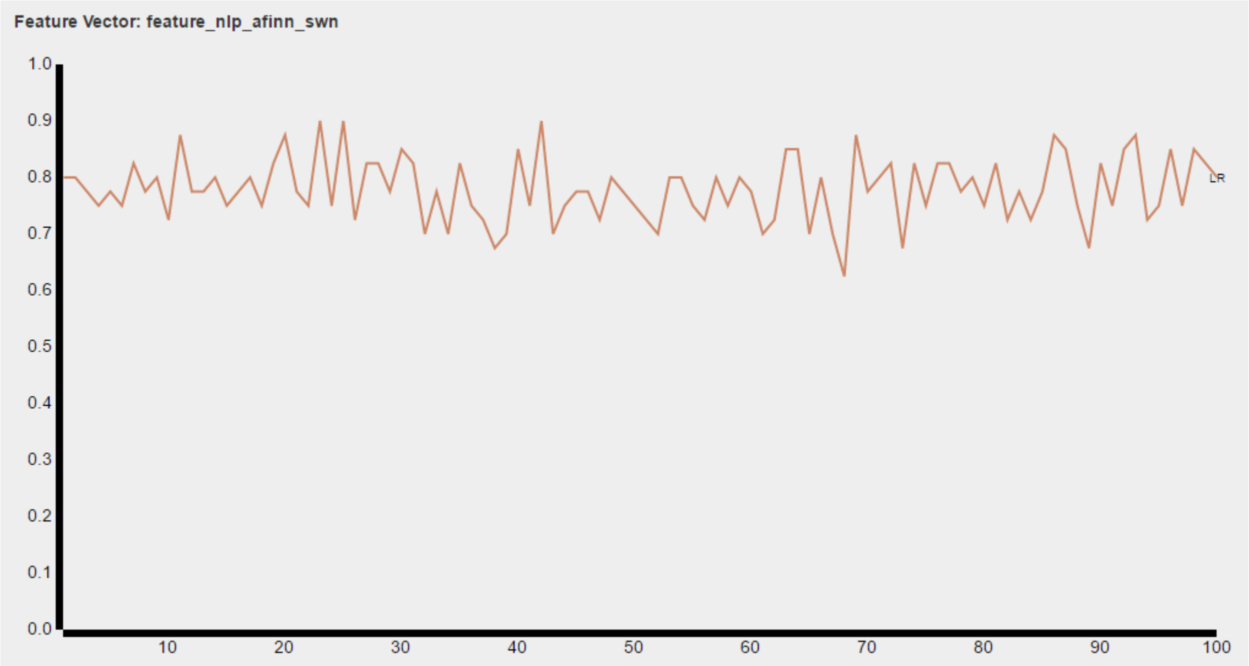


Figure 16: Experiment 1 – 'trump cabinet' AFINN + SWN Classifier Performance

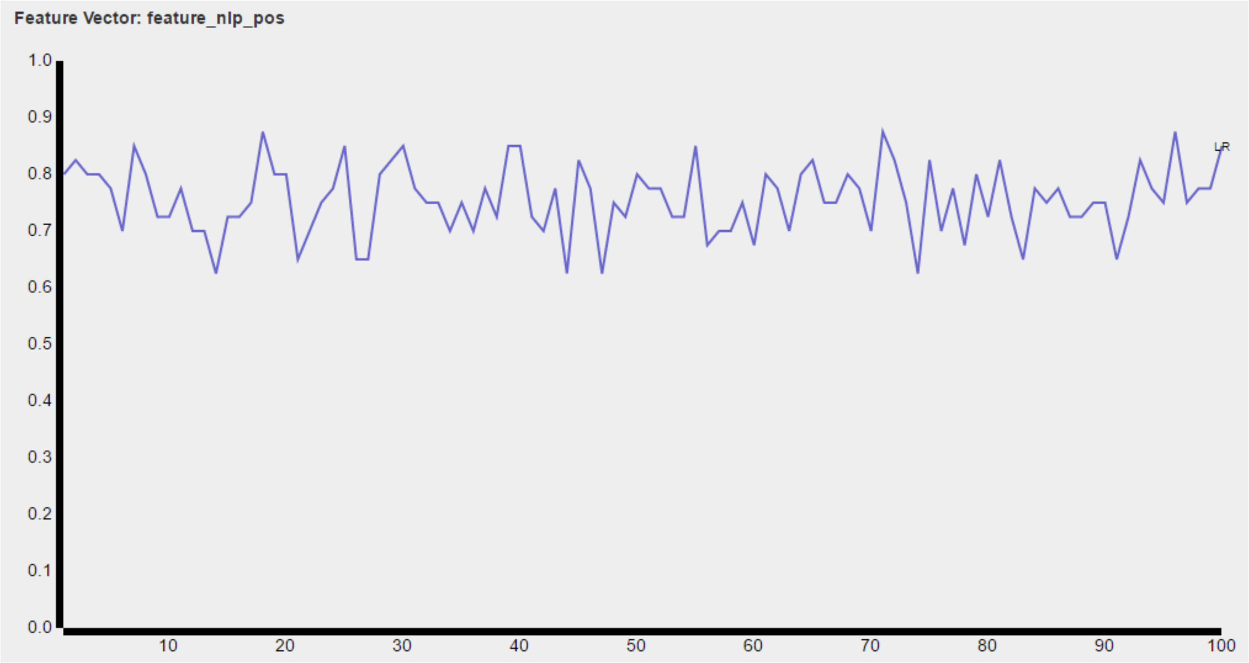


Figure 17: Experiment 1 – 'trump cabinet' POS Classifier Performance

The following tables are the coefficients gathered from the best models of both feature vectors.

Feature (Ft)	**AFINN**	**Pos**	**Neg**	**Obj**
Weight (Wt)	-0.103	0.059	0.065	0.430

Table 16: Experiment 1 – LR Coefficients for Support Label in 'trump_cabinet' Dataset

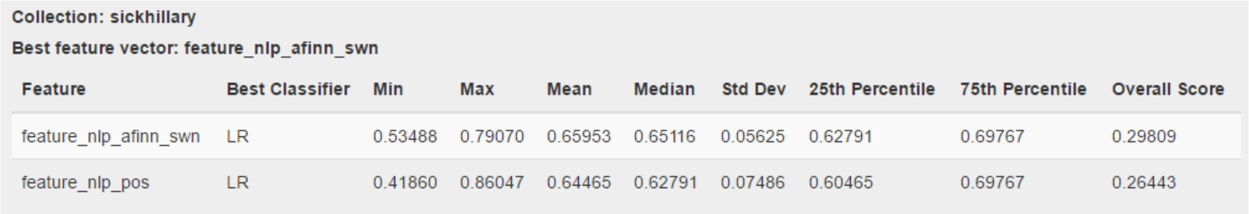
The LR coefficients for the AFINN + SWN feature vector indicate that the Obj feature is the strongest determinant of a tweet's label. Therefore, tweets with words with the high objectivity ratings are more likely to be of 's' label, ie. News.

Ft	**ADJ**	**ADP**	**ADV**	**CONJ**	**DET**	**NOUN**	**NUM**	**PRT**	**PRON**	**VERB**	**.**	**X**
Wt	0.032	- 0.025	0.061	0.142	0.063	0.104	0.066	0.048	- 0.003	0.082	- 0.531	0.066

Table 17: Experiment 1 – LR Coefficients for POS in 'trump_cabinet' Dataset

The '.' coefficient is the strongest determinant in the LR model for the POS feature vector. Therefore, tweets with more punctuations would be more likely not to be classified as a news tweet.

For the ‘sickhillary’ dataset, both feature vectors showed poorer accuracy, with performance levels that are only marginally better than a random classifier.

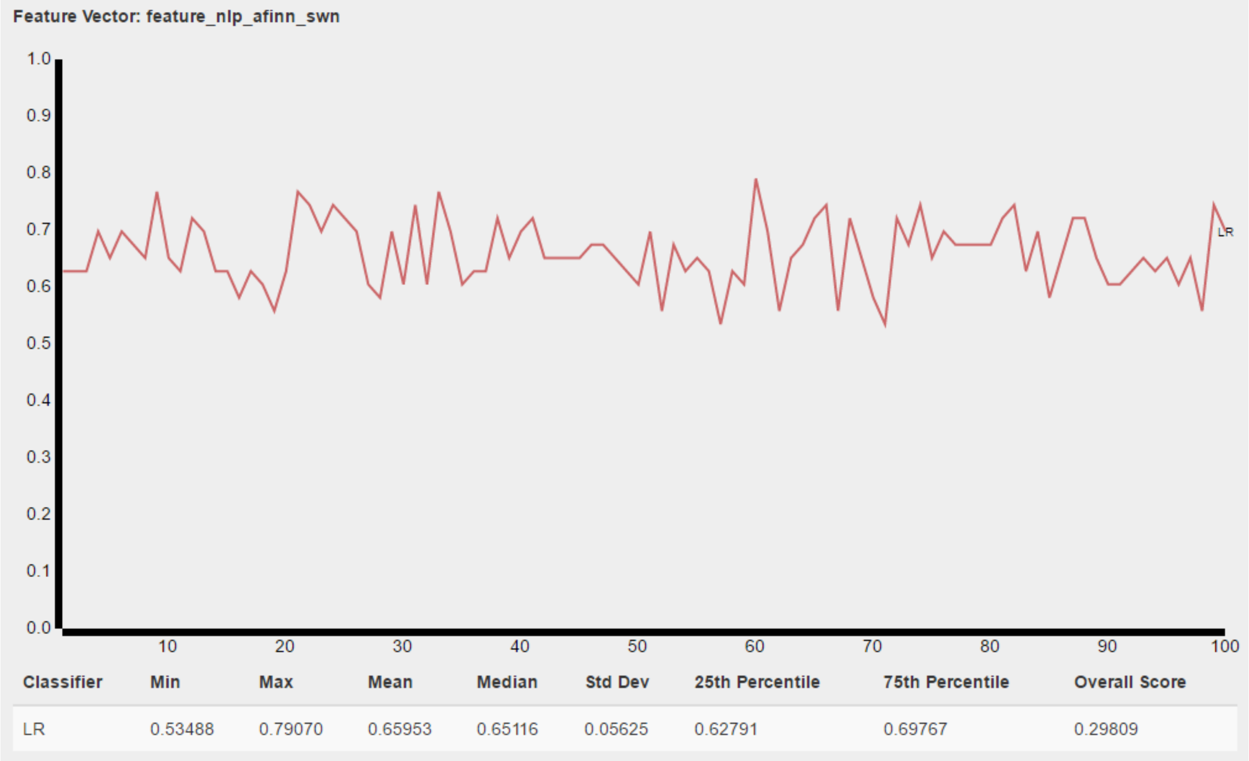


Collection: sickhillary

Best feature vector: feature_nlp_afinn_swn

Feature	Best Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
feature_nlp_afinn_swn	LR	0.53488	0.79070	0.65953	0.65116	0.05625	0.62791	0.69767	0.29809
feature_nlp_pos	LR	0.41860	0.86047	0.64465	0.62791	0.07486	0.60465	0.69767	0.26443

Table 18: Experiment 1 – ‘sickhillary’ Feature Vector Performance Overview



Feature Vector: feature_nlp_afinn_swn

Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
LR	0.53488	0.79070	0.65953	0.65116	0.05625	0.62791	0.69767	0.29809

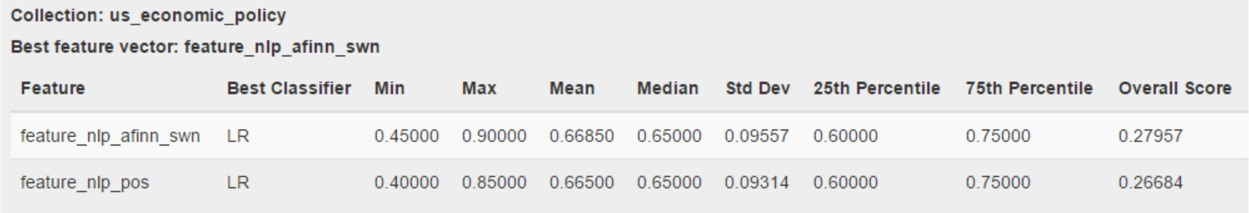
Figure 18: Experiment 1 – ‘sickhillary’ AFINN + SWN Classifier Performance

Feature (Ft)	**AFINN**	**Pos**	**Neg**	**Obj**
Weight (Wt)	-0.026	0.060	0.054	0.258

Table 19: Experiment 1 – Coefficients for Support Label in 'sickhillary' Dataset

The LR coefficients for the AFINN + SWN feature vector indicate that the Obj feature is the strongest determinant of a tweet’s label. Therefore, tweets with words with the high objectivity ratings are more likely to be of ‘s’ label, ie. Rumour.

For the ‘US Economic Policy’ dataset, both feature vectors showed good accuracy.

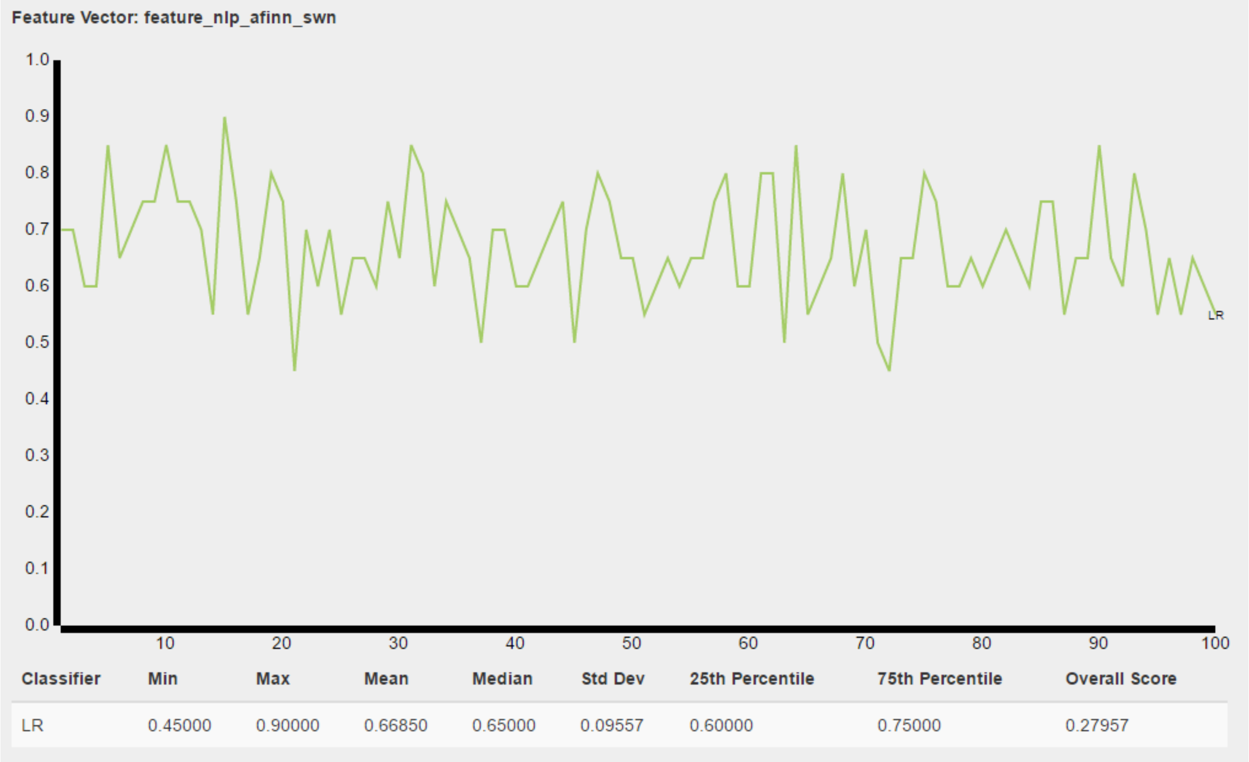


Collection: us_economic_policy

Best feature vector: feature_nlp_afinn_swn

Feature	Best Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
feature_nlp_afinn_swn	LR	0.45000	0.90000	0.66850	0.65000	0.09557	0.60000	0.75000	0.27957
feature_nlp_pos	LR	0.40000	0.85000	0.66500	0.65000	0.09314	0.60000	0.75000	0.26684

Table 20: Experiment 1 – ‘US Economic Policy’ Feature Vector Performance Overview



Feature Vector: feature_nlp_afinn_swn

Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
LR	0.45000	0.90000	0.66850	0.65000	0.09557	0.60000	0.75000	0.27957

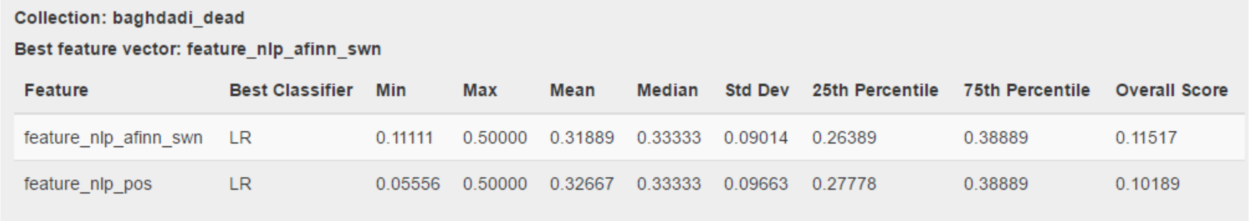
Figure 19: Experiment 1 – ‘US Economic Policy’ AFINN + SWN Classifier Performance

Feature (Ft)	**AFINN**	**Pos**	**Neg**	**Obj**
Weight (Wt)	-0.004	-0.024	-0.017	0.224

Table 21: Experiment 1 – LR Coefficients for Support Label in 'US Economic Policy' Dataset

The LR coefficients for the AFINN + SWN feature vector indicate that the Obj feature is the strongest determinant of a tweet’s label. Therefore, tweets with words with the high objectivity ratings are more likely to be of ‘s’ label, ie. News.

For the ‘baghdadi_dead’ dataset, both feature vectors showed poor accuracy. This means that the sentiment scores alone are not good enough features for classifiers to reliably recognize the patterns of the tweets in this dataset. Thus, there is no necessity to interpret the LR model.

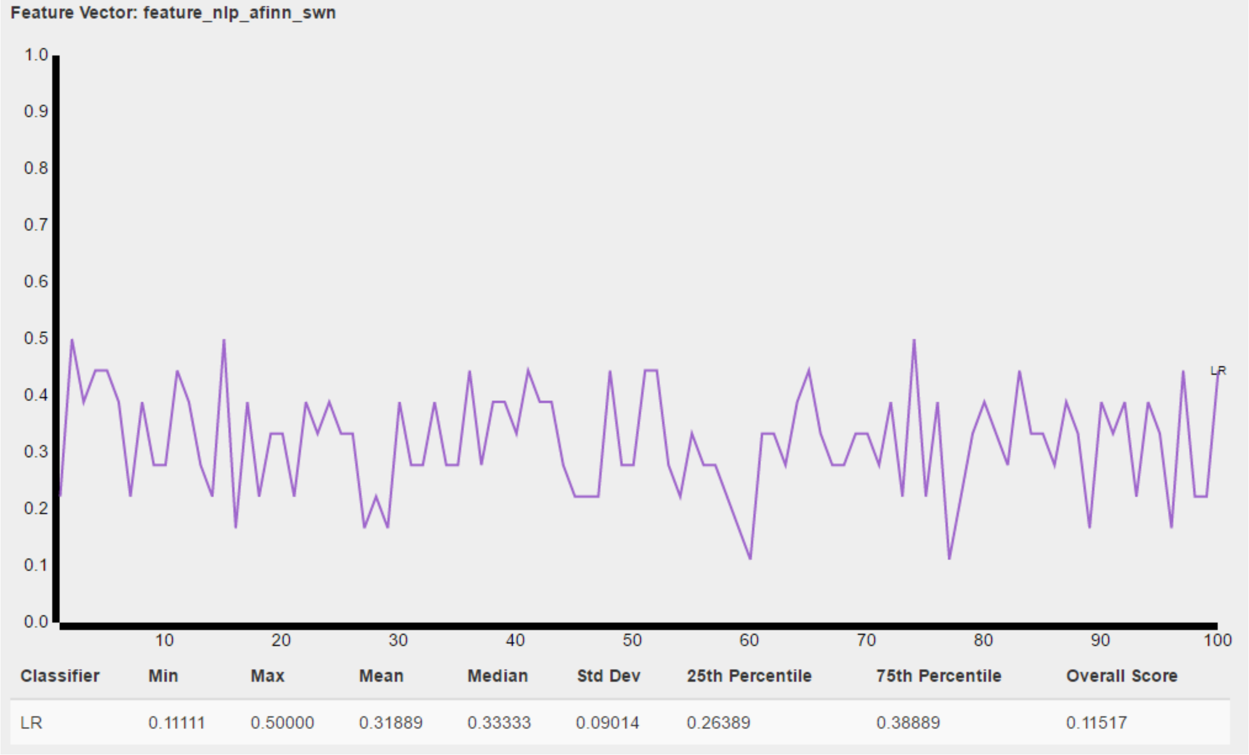


Collection: baghdadi_dead

Best feature vector: feature_nlp_afinn_swn

Feature	Best Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
feature_nlp_afinn_swn	LR	0.11111	0.50000	0.31889	0.33333	0.09014	0.26389	0.38889	0.11517
feature_nlp_pos	LR	0.05556	0.50000	0.32667	0.33333	0.09663	0.27778	0.38889	0.10189

Table 22: Experiment 1 – ‘baghdadi dead’ Feature Vector Performance Overview:



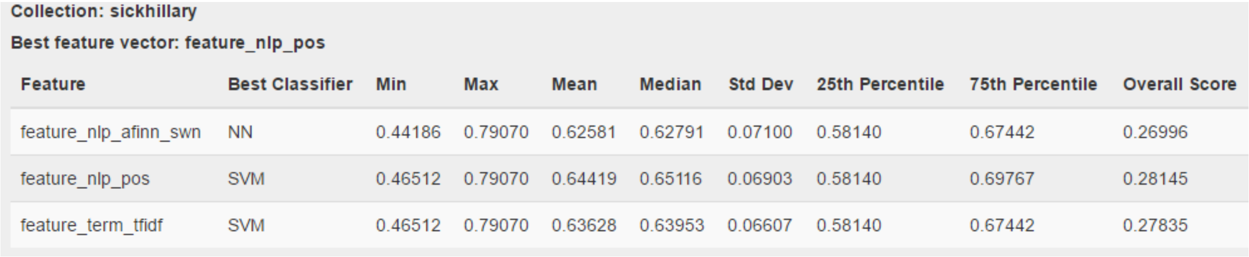
Feature Vector: feature_nlp_afinn_swn

Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
LR	0.11111	0.50000	0.31889	0.33333	0.09014	0.26389	0.38889	0.11517

Figure 20: Experiment 1 – ‘baghdadi dead’ AFINN + SWN Classifier Performance

Experiment 2: How well can rumours and non-rumours be separated in rumour-centric datasets?

For the ‘sickhillary’ dataset, the POS feature vector rendered the best classifier results in general, but otherwise all three feature vectors performed to a similar level. This is a notable finding and indicates potential usage of the features generated by sentiment analysis libraries.

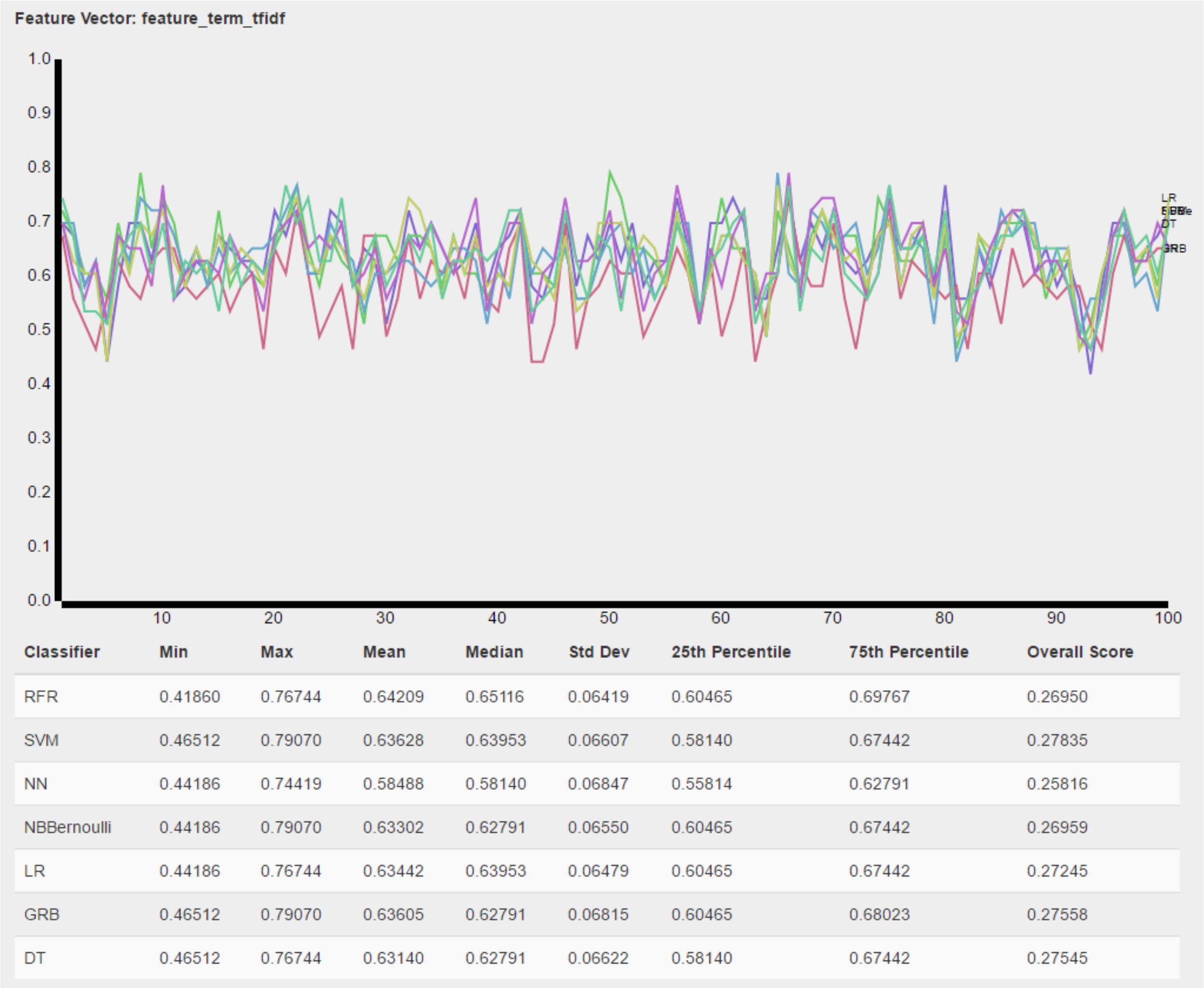


Collection: sickhillary

Best feature vector: feature_nlp_pos

Feature	Best Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
feature_nlp_afinn_swn	NN	0.44186	0.79070	0.62581	0.62791	0.07100	0.58140	0.67442	0.26996
feature_nlp_pos	SVM	0.46512	0.79070	0.64419	0.65116	0.06903	0.58140	0.69767	0.28145
feature_term_tfidf	SVM	0.46512	0.79070	0.63628	0.63953	0.06607	0.58140	0.67442	0.27835

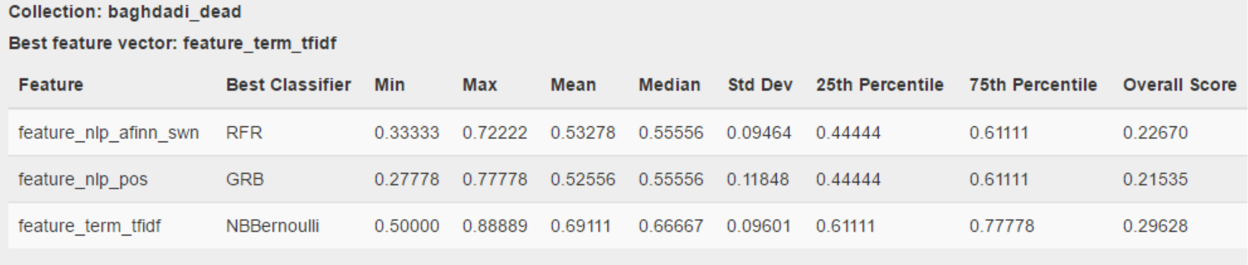
Table 23: Experiment 2 – ‘sickhillary’ Feature Vector Performance Overview



Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
RFR	0.41860	0.76744	0.64209	0.65116	0.06419	0.60465	0.69767	0.26950
SVM	0.46512	0.79070	0.63628	0.63953	0.06607	0.58140	0.67442	0.27835
NN	0.44186	0.74419	0.58488	0.58140	0.06847	0.55814	0.62791	0.25816
NBBernoulli	0.44186	0.79070	0.63302	0.62791	0.06550	0.60465	0.67442	0.26959
LR	0.44186	0.76744	0.63442	0.63953	0.06479	0.60465	0.67442	0.27245
GRB	0.46512	0.79070	0.63605	0.62791	0.06815	0.60465	0.68023	0.27558
DT	0.46512	0.76744	0.63140	0.62791	0.06622	0.58140	0.67442	0.27545

Figure 21: Experiment 2 – ‘sickhillary’ Tf-idf Classifier Performance

For the ‘baghdadi_dead’ dataset, all feature vectors resulted in large standard deviations in the classifiers tested. Notably, the SVM classifier performed significantly worse than the other classifiers. The poor overall performance may be attributed to the low amount of observations for both label classes.

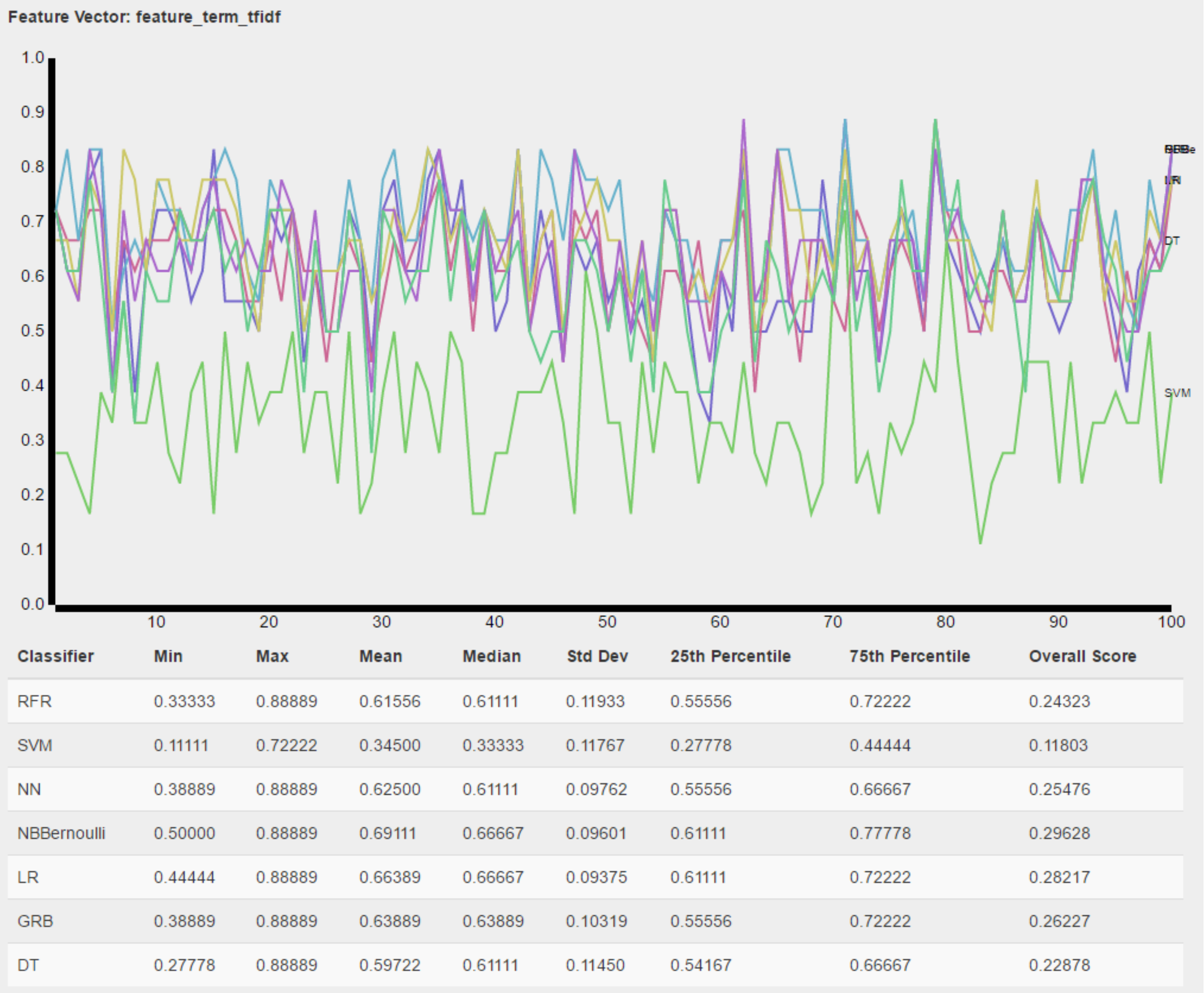


Collection: baghdadi_dead

Best feature vector: feature_term_tfidf

Feature	Best Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
feature_nlp_afinn_swn	RFR	0.33333	0.72222	0.53278	0.55556	0.09464	0.44444	0.61111	0.22670
feature_nlp_pos	GRB	0.27778	0.77778	0.52556	0.55556	0.11848	0.44444	0.61111	0.21535
feature_term_tfidf	NBBernoulli	0.50000	0.88889	0.69111	0.66667	0.09601	0.61111	0.77778	0.29628

Table 24: Experiment 2 – ‘baghdadi dead’ Feature Vector Performance Overview

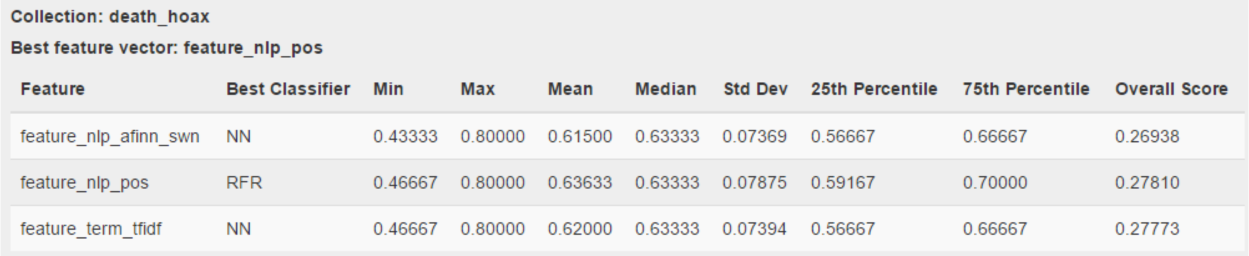


Feature Vector: feature_term_tfidf

Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
RFR	0.33333	0.88889	0.61556	0.61111	0.11933	0.55556	0.72222	0.24323
SVM	0.11111	0.72222	0.34500	0.33333	0.11767	0.27778	0.44444	0.11803
NN	0.38889	0.88889	0.62500	0.61111	0.09762	0.55556	0.66667	0.25476
NBBernoulli	0.50000	0.88889	0.69111	0.66667	0.09601	0.61111	0.77778	0.29628
LR	0.44444	0.88889	0.66389	0.66667	0.09375	0.61111	0.72222	0.28217
GRB	0.38889	0.88889	0.63889	0.63889	0.10319	0.55556	0.72222	0.26227
DT	0.27778	0.88889	0.59722	0.61111	0.11450	0.54167	0.66667	0.22878

Figure 22: Experiment 2 – ‘baghdadi dead’ Tf-idf Classifier Performance

For the ‘death_hoax’ dataset, all feature vectors rendered better-than-average performance.

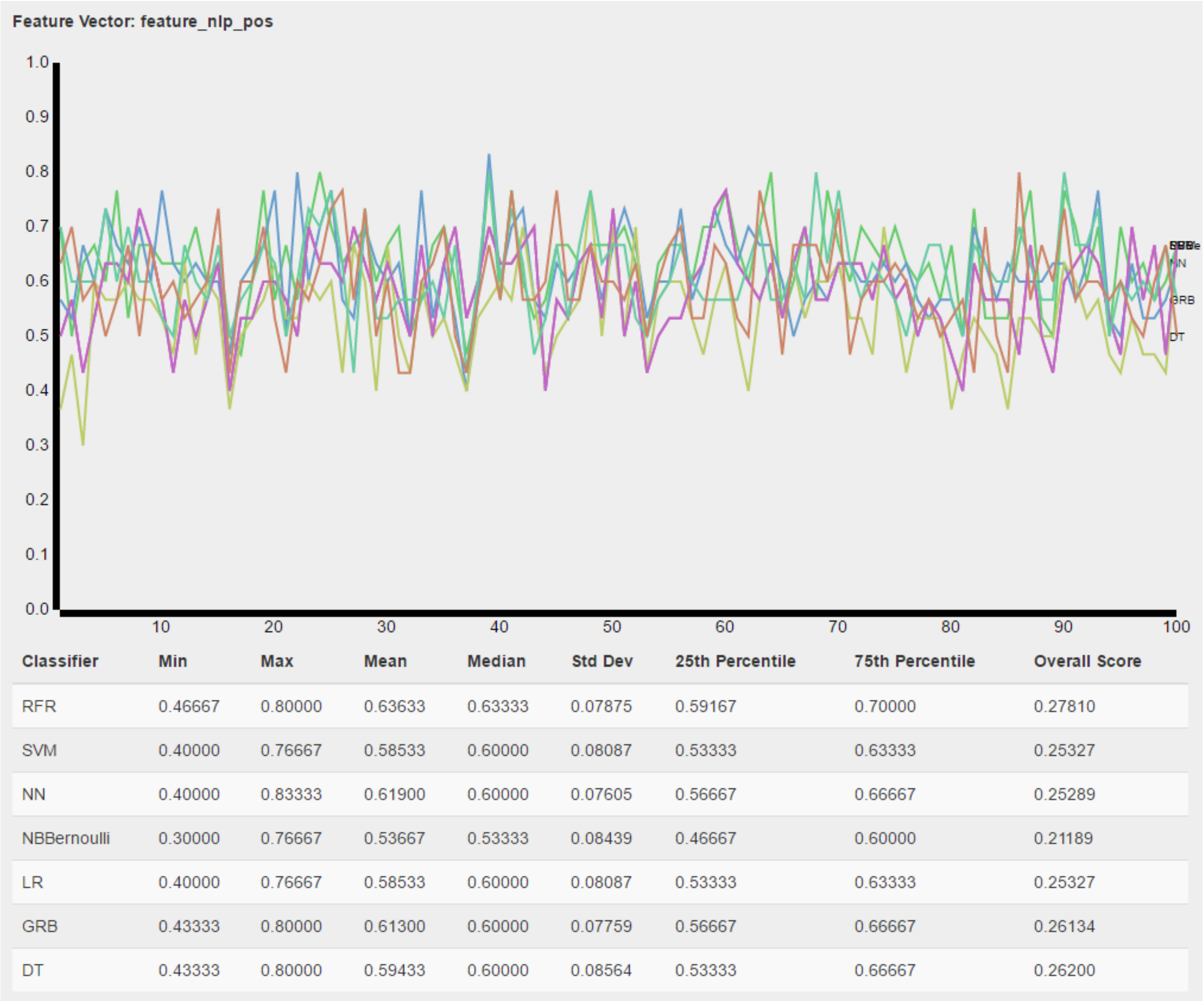


Collection: death_hoax

Best feature vector: feature_nlp_pos

Feature	Best Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
feature_nlp_afinn_swn	NN	0.43333	0.80000	0.61500	0.63333	0.07369	0.56667	0.66667	0.26938
feature_nlp_pos	RFR	0.46667	0.80000	0.63633	0.63333	0.07875	0.59167	0.70000	0.27810
feature_term_tfidf	NN	0.46667	0.80000	0.62000	0.63333	0.07394	0.56667	0.66667	0.27773

Table 25: Experiment 2 – ‘death hoax’ Feature Vector Performance Overview



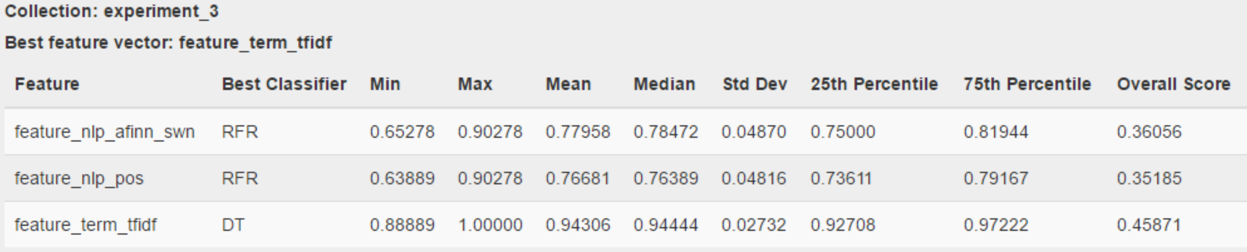
Feature Vector: feature_nlp_pos

Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
RFR	0.46667	0.80000	0.63633	0.63333	0.07875	0.59167	0.70000	0.27810
SVM	0.40000	0.76667	0.58533	0.60000	0.08087	0.53333	0.63333	0.25327
NN	0.40000	0.83333	0.61900	0.60000	0.07605	0.56667	0.66667	0.25289
NBBernoulli	0.30000	0.76667	0.53667	0.53333	0.08439	0.46667	0.60000	0.21189
LR	0.40000	0.76667	0.58533	0.60000	0.08087	0.53333	0.63333	0.25327
GRB	0.43333	0.80000	0.61300	0.60000	0.07759	0.56667	0.66667	0.26134
DT	0.43333	0.80000	0.59433	0.60000	0.08564	0.53333	0.66667	0.26200

Figure 23: Experiment 2 – ‘death hoax’ POS Classifier Performance

Experiment 3: How well can rumours and non-rumours be separated using all datasets?

All the feature vectors showed good-to-excellent performance levels, with the tf-idf model being significantly better than the other two. This indicates that the keyword-based approach of the tf-idf model is good at separating rumours from non-rumours, at least for the observed topics in the 6 datasets. More notably, the AFINN + SWN and POS feature vectors performed very well, indicating that there are significant sentiment differences between rumour and non-rumour datasets.

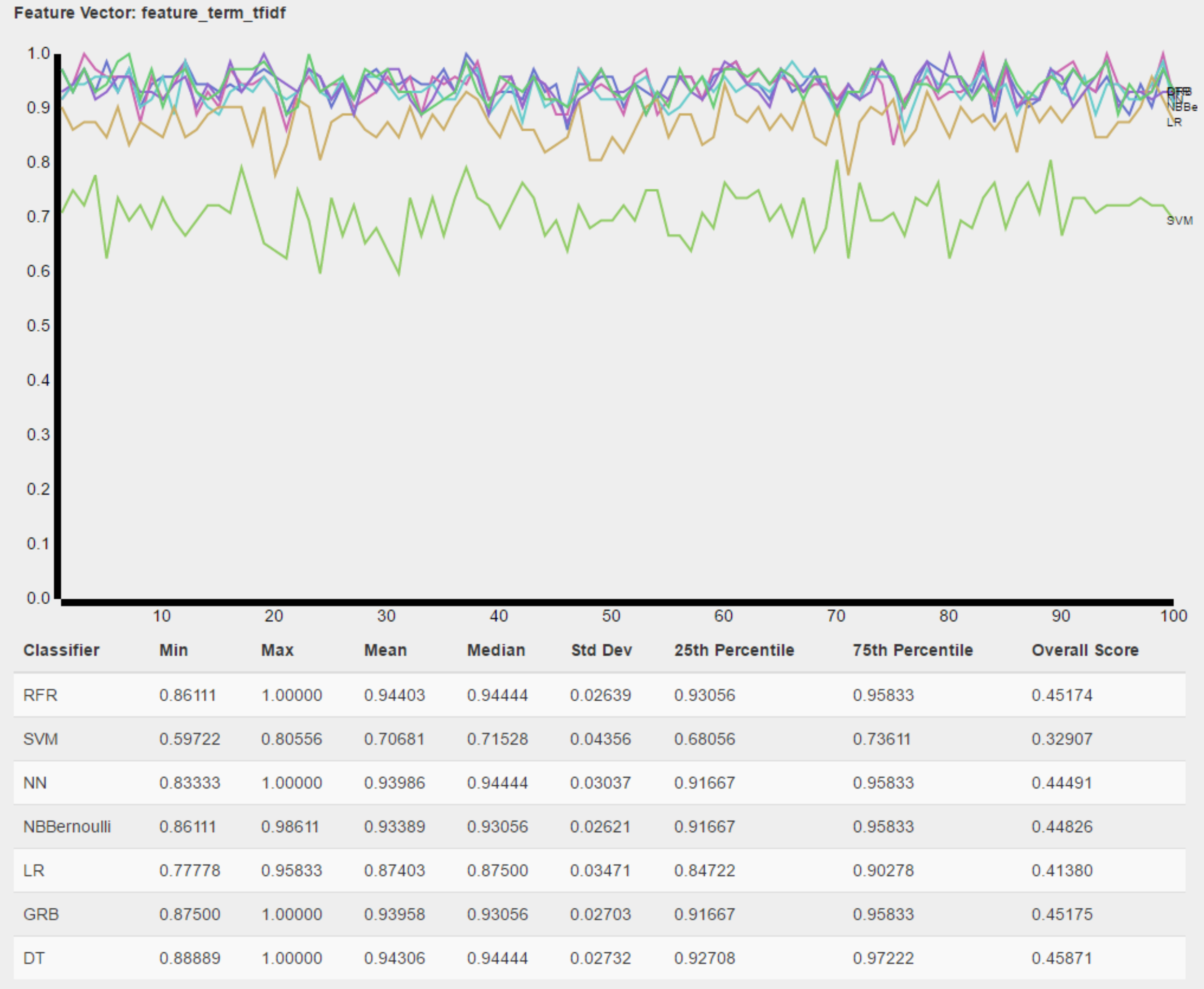


Collection: experiment_3

Best feature vector: feature_term_tfidf

Feature	Best Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
feature_nlp_afinn_swn	RFR	0.65278	0.90278	0.77958	0.78472	0.04870	0.75000	0.81944	0.36056
feature_nlp_pos	RFR	0.63889	0.90278	0.76681	0.76389	0.04816	0.73611	0.79167	0.35185
feature_term_tfidf	DT	0.88889	1.00000	0.94306	0.94444	0.02732	0.92708	0.97222	0.45871

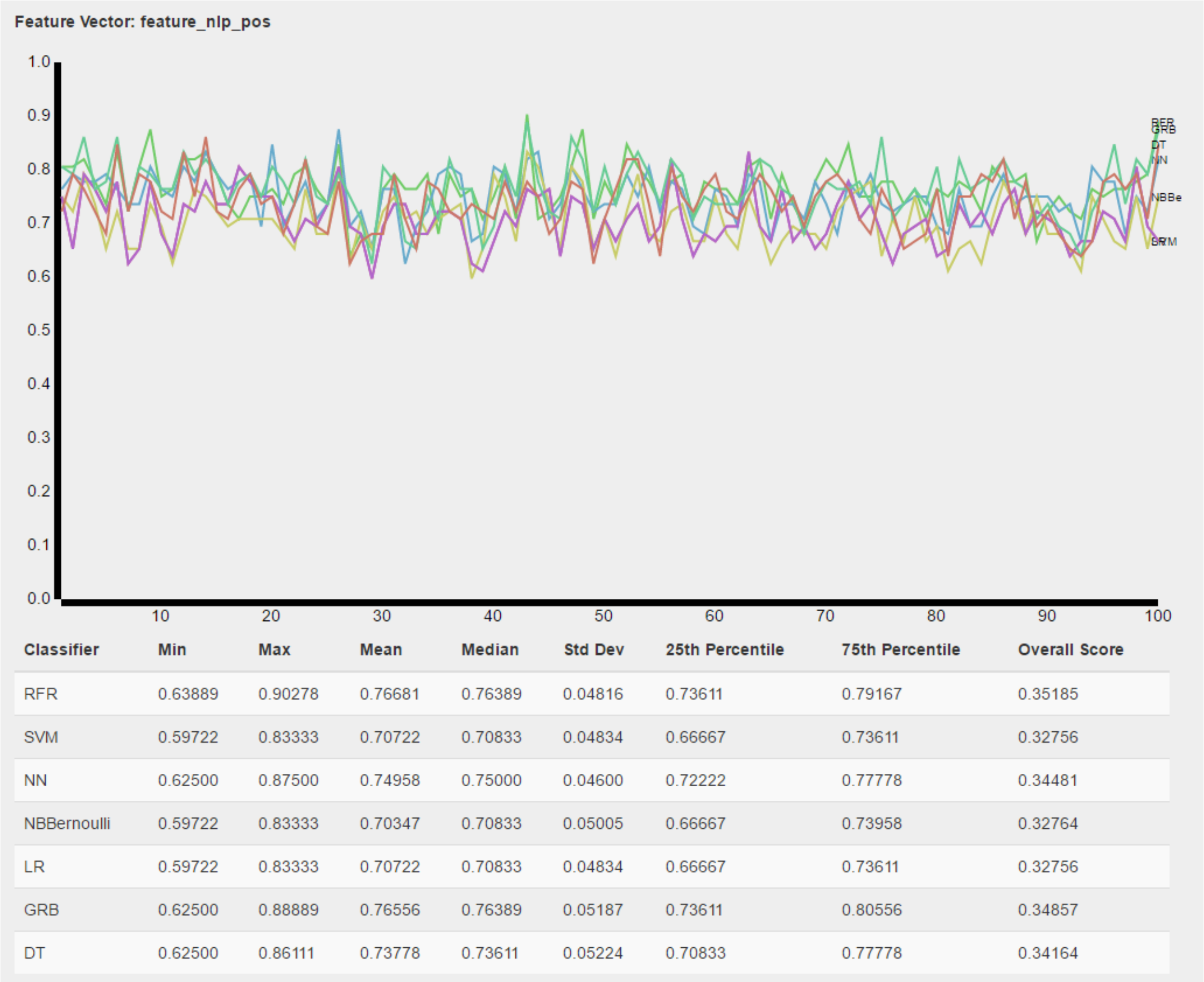
Table 26: Experiment 3 – Feature Vector Performance Overview



Feature Vector: feature_term_tfidf

Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
RFR	0.86111	1.00000	0.94403	0.94444	0.02639	0.93056	0.95833	0.45174
SVM	0.59722	0.80556	0.70681	0.71528	0.04356	0.68056	0.73611	0.32907
NN	0.83333	1.00000	0.93986	0.94444	0.03037	0.91667	0.95833	0.44491
NBBernoulli	0.86111	0.98611	0.93389	0.93056	0.02621	0.91667	0.95833	0.44826
LR	0.77778	0.95833	0.87403	0.87500	0.03471	0.84722	0.90278	0.41380
GRB	0.87500	1.00000	0.93958	0.93056	0.02703	0.91667	0.95833	0.45175
DT	0.88889	1.00000	0.94306	0.94444	0.02732	0.92708	0.97222	0.45871

Figure 24: Experiment 3 – Tf-idf Classifier Performance



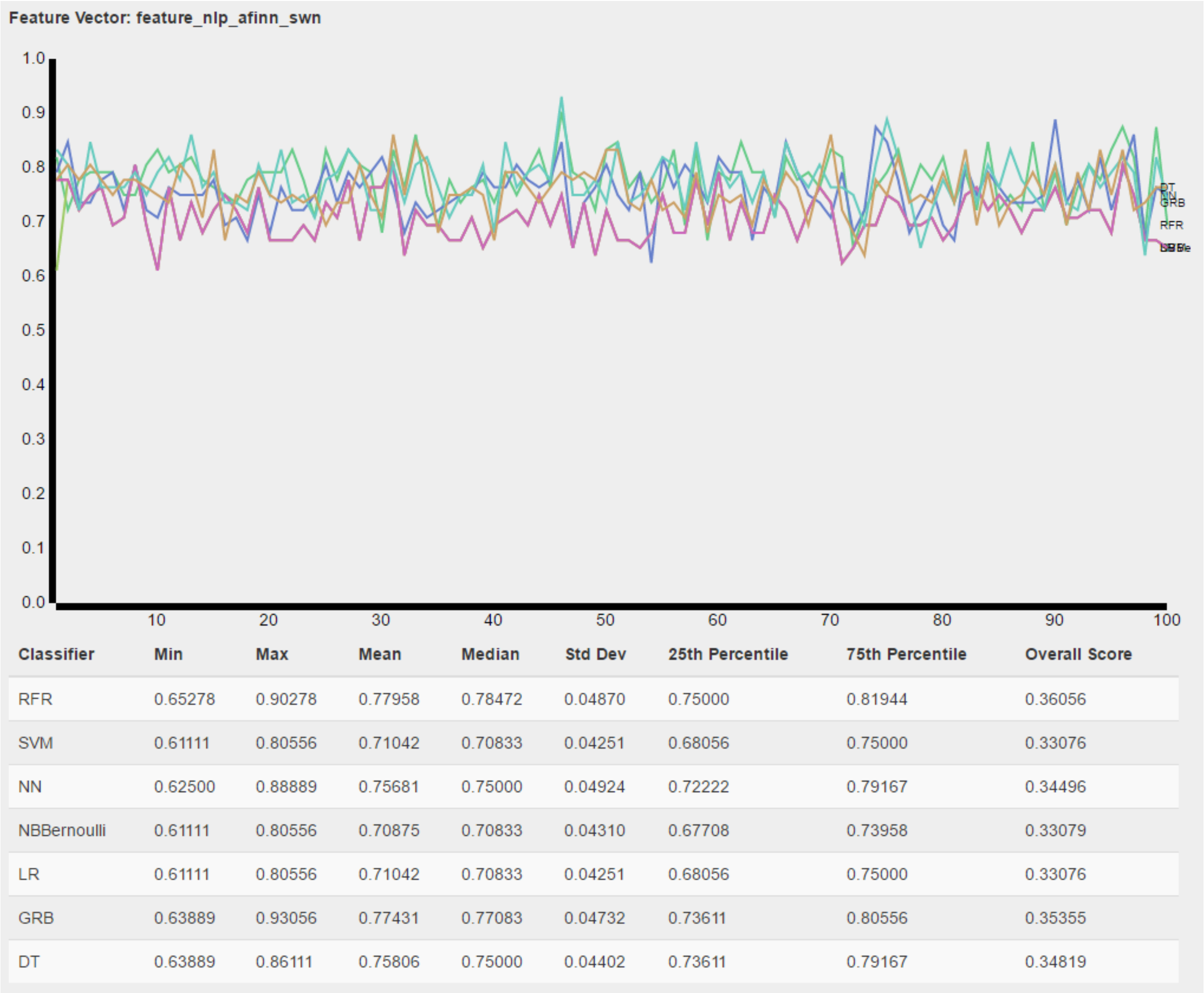
Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
RFR	0.63889	0.90278	0.76681	0.76389	0.04816	0.73611	0.79167	0.35185
SVM	0.59722	0.83333	0.70722	0.70833	0.04834	0.66667	0.73611	0.32756
NN	0.62500	0.87500	0.74958	0.75000	0.04600	0.72222	0.77778	0.34481
NBBernoulli	0.59722	0.83333	0.70347	0.70833	0.05005	0.66667	0.73958	0.32764
LR	0.59722	0.83333	0.70722	0.70833	0.04834	0.66667	0.73611	0.32756
GRB	0.62500	0.88889	0.76556	0.76389	0.05187	0.73611	0.80556	0.34857
DT	0.62500	0.86111	0.73778	0.73611	0.05224	0.70833	0.77778	0.34164

Figure 25: Experiment 3 – POS Classifier Performance

Ft	**ADJ**	**ADP**	**ADV**	**CONJ**	**DET**	**NOUN**	**NUM**	**PRT**	**PRON**	**VERB**	**.**	**X**
Wt	- 0.126	0.001	0.028	-0.24	- 0.131	0.307	- 0.028	- 0.057	- 0.090	0.138	1.018	- 0.022

Table 27: Experiment 3 – LR Coefficients for POS

The '.' coefficient is the strongest determinant in the LR model for the POS feature vector. Therefore, tweets with more punctuations would be more likely to be classified as a rumour tweet.


Feature Vector: feature_nlp_afinn_swn

Classifier	Min	Max	Mean	Median	Std Dev	25th Percentile	75th Percentile	Overall Score
RFR	0.65278	0.90278	0.77958	0.78472	0.04870	0.75000	0.81944	0.36056
SVM	0.61111	0.80556	0.71042	0.70833	0.04251	0.68056	0.75000	0.33076
NN	0.62500	0.88889	0.75681	0.75000	0.04924	0.72222	0.79167	0.34496
NBBernoulli	0.61111	0.80556	0.70875	0.70833	0.04310	0.67708	0.73958	0.33079
LR	0.61111	0.80556	0.71042	0.70833	0.04251	0.68056	0.75000	0.33076
GRB	0.63889	0.93056	0.77431	0.77083	0.04732	0.73611	0.80556	0.35355
DT	0.63889	0.86111	0.75806	0.75000	0.04402	0.73611	0.79167	0.34819

Figure 26: Experiment 3 – SWN Classifier Performance

Feature (Ft)	**AFINN**	**Pos**	**Neg**	**Obj**
Weight (Wt)	0.515	-0.137	-0.166	0.827

Table 28: Experiment 3 – LR Coefficients for Support Label

The LR coefficients for the AFINN + SWN feature vector indicate that the Obj feature is the strongest determinant of a tweet’s label. Therefore, tweets with words with the high objectivity ratings are more likely to be of ‘s’ label, ie. Rumour.

10. Discussion & Conclusion

10.1. Discussion

Experiment 1: What are the general sentiment profiles of the datasets?

The general trend of the analysis reveals that tweets with a high proportion of punctuations and tweets with a high proportion of words with high Objectivity ratings are more likely to be relevant to the main context of the dataset, and thus will be predicted to be of the Support label. This also necessarily indicates that tweets that are labelled Support share a common profile of sentiment scores.

Experiment 2: How well can rumours and non-rumours be separated in rumour-centric datasets?

The general trend showed that all three feature-vectors performed at a similar level for each of the three datasets. This suggests that sentiment scores may be used reliably as tf-idf features to train classifiers, as they provide a close level of performance to the tf-idf model.

The only exception is in the case of the 'baghdadi dead' dataset, where all three features vectors performed weakly, possibly due to the small number of observations for the dataset (192 tweets, 89 unique tweets). In this instance, the Tf-idf feature vector performed much weaker as compared to the other two feature vectors.

Experiment 3: How well can rumours and non-rumours be separated using all datasets?

The results indicate that rumours and non-rumours can be separated to a high degree of accuracy, at least for the dataset observations. While a high level of performance from the tf-idf feature vector is to be expected, it is notable that the AFINN + SWN and POS feature vectors have also performed very well. Thus, it suggests that rumour and non-rumour datasets may have significant differences in sentiment scores, and by extension, sentiments in the texts. The high level of performance from the sentiment scores-based feature vectors also lends credence to the potential of sentiment analysis libraries.

The advantage of using sentiment scoring techniques is that they exhibit less bias, as opposed to the tf-idf method, as the latter method relies of having to learn the word weights based upon an existing corpus. The tf-idf method is better for identifying if an unseen observation is similar to the data it was trained with, in terms of exact words, whereas sentiment scoring methods are better for identifying the proportions of sentiments in a sentence.

10.2. Conclusion

In conclusion, this study illustrates the various considerations one needs to take in designing and evaluating search queries and the resultant datasets from it.

This study also illustrates the potential of using sentiment analysis libraries such as SentiWordNet and AFINN in identifying news or rumours, depending on the main context of interest of a dataset. The main advantage of using sentiment analysis libraries for feature engineering as opposed to the Tf-idf model is that the former does not require an existing corpus to generate features against, thus making it more suitable to semi-supervised machine learning contexts.

10.3. Recommendations for Future Work

As this is only a preliminary and broad study on rumours on online social networks, improvements can be broadly groups into two categories:

1. Implementation

The existing workflow can be enhanced in the following ways:

- Leveraging on GPU acceleration to speed up calculations
- Utilizing a distributed database for greater scale-up capability
- Real-time importing and visualization of data
- Web UI tweet labelling capability

2. Investigative

Deeper investigative work can be performed on the following:

- Addressing sampling biases
- Evaluation of existing models on public datasets (eg. Rumour/News datasets)
- Evaluation of existing models on other types of texts (eg. Articles)
- Testing other sentiment analysis libraries (eg. LWIC, SentiStrength, ANEW)
- Testing of different neural network structures (eg. Convolutional Neural Network, Recurrent Neural Network)
- Testing of other classifiers (eg. Conditional Random Field)

References

[1] GlobalWebIndex, "Media Consuption Insight Report: Q3 2014," 2014.

[2] The Guardian, "Facebook’s failure: did fake news and polarized politics get Trump elected?," 10 November 2016. [Online]. Available: https://www.theguardian.com/technology/2016/nov/10/facebook-fake-news-election-conspiracy-theories. [Accessed 13 March 2017].

[3] F. A. Nielsen, "A new ANEW: Evaluation of a word list for sentiment analysis in microblogs," in *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*, Heraklion, Crete, 2011.

[4] E. N. Forsyth and C. H. Martell, "Lexical and Discourse Analysis of Online Chat Dialog," *Proceedings of the First IEEE International Conference on Semantic Computing,* pp. 19-26, 2007.

[5] Q. Zhang, S. Zhang, J. Dong, J. Xiong and X. Cheng, "Automatic Detection of Rumor on Social Network," *Natural Language Processing and Chinese Computing Lecture Notes in Computer Science,* pp. 113-122, 2015.

[6] K. Elsa, G. Sam, E. J. Michael and G. Erhardt, "Detecting Sadness in 140 Characters: Sentiment Analysis and Mourning Michael Jackson on Twitter," Web Ecology Project, Boston, 2009.

[7] G. W. Allport and L. Postman, "An Analysis of Rumor," *The Public Opinion Quarterly,* vol. 10, pp. 501-517, 1946.

[8] C. Heath, C. Bell and E. Sternbern, "Emotional Selection in Memes: The Case of Urban Legends," *Journal of Personality and Social Psychology,* vol. 81, pp. 1028-1041, 2001.

[9] D. S. Wilson, C. Wilczynski, A. Wells and L. Weiser, "The Evolution of Cognition," *The MIT Press,* 2000.

[10] R. L. Rosnow, J. H. Yost and J. L. Esposito, "Belief in Rumor and Likelihood of Rumor Transmission," *Language & Communication,* pp. 189-194, 1986.

[11] A. Carus and A. Mesut, "A new compression algorithm for fast text search," *Turkish Journal Of Electrical Engineering & Computer Sciences,* vol. 24, pp. 4355-4367, 2016.

[12] B. Santorini, "Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision)," *ScholarlyCommons,* 1990.

[13] S. Bird, E. Klein and E. Loper, Natural Language Processing with Python, O'Reilly Media, 2009.

[14] . S. Baccianella, A. Esuli and F. Sebas, "SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining," *Seventh Conference on International Language Resources and Evaluation,* 2010.

Appendix

A. Android Information Harvester

A-1. Main Configuration File

The main configuration file, named ‘app_config.json’, defines the location the configuration files and tweet archives.

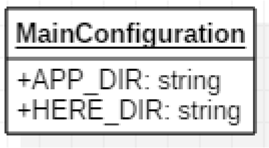


Figure 27: Class Diagram for Main Configuration Object

Field	Type	Purpose
APP_DIR	String	Stores the directory of the main configuration file (ie. Itself)
HERE_DIR	String	Stores the current directory of the main configuration file.

Table 29: Detailed Explanation of Main Configuration Object

A-2. Base Configuration File

The base configuration file defines where the main configuration file is stored. This second configuration directs the program to the correct location (eg. SD storage or Internal Storage) for it to read the main configuration file. The base configuration file will be created in the Internal Storage, and it is called ‘app_init_config.json’. Should the base configuration file be deleted or in the event of the first launch of the application, the program will search for a suitable storage location (eg. SD storage or Internal Storage), store the path to the location, and create the main configuration file at the location.

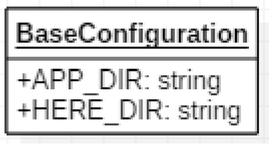


Figure 28: Class Diagram for Base Configuration Object

Field	Type	Purpose
APP_DIR	String	Stores the directory of the main configuration file.
HERE_DIR	String	Stores the current directory of the base configuration file.

Table 30: Detailed Explanation of Base Configuration Object

A-3. Worker Configuration File(s)

The worker configuration files define which URL endpoints to collect information from and the related parameters for the requests. The application will read the JSON web worker configuration files from the /files/config/workers directory. The application can handle multiple web worker configuration files and will create web workers based on the configuration files.

The following figures and tables elaborate on the types and purposes of each variable in the web worker configuration object.

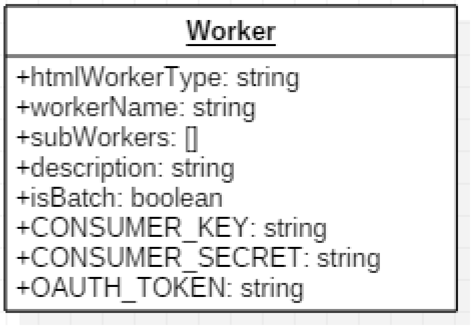


Figure 29: Class Diagram for Worker Configuration Object

Field	Type	Purpose
htmlWorkerType	string	Defines type of worker. PLAIN: A plain worker, collects from endpoints with no authentication. TWEET: A tweet worker, collects data via the Twitter API
workerName	string	Custom name of worker, and determines directory name to be stored.
subWorkers	List	Stores a list of subworkers.
description	String	Custom description for user.
isBatch	Boolean	If true, subworkers will be run in the order that they are declared in the subWorkers list.
CONSUMER_KEY	String	Stores the CONSUMER_KEY required for authentication purposes for Twitter API

CONSUMER_SECRET	String	Stores the CONSUMER_SECRET required for authentication purposes for Twitter API
OAUTH_TOKEN	String	Stores the OAUTH_TOKEN required for authentication purposes for Twitter API

Table 31: Detailed Explanation of Worker Configuration Object

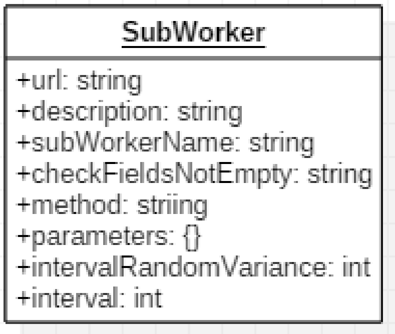


Figure 30: Class Diagram for SubWorker Configuration Object

Field	**Type**	**Purpose**
url	String	Determines the endpoint to make the request to and to collect data from.
description	String	Custom description for user.
subWorkerName	String	Custom name of subworker, and determines directory name to be stored under the directory of the parent worker directory.
checkFieldsNotEmpty	String	If defined, worker will check fields and if fields are empty, worker will not store the response data of the given query.
Method	String	Defines the HTTP method of the request.
Parameters	{}	Defines the additional parameters of the request.
intervalRandomVariance	Int	Defines the time interval variance between requests.
internal	Int	Defines the fixed time interval (ie. Frequency) between requests

Table 32: Detailed Explanation of SubWorker Configuration Object

B. Data Workflow: Collection, Import & Retrieval

B-1. Tweet Type to Integer Enumeration Mapping

Tweet Type	Integer Enumeration Mapping
Normal Tweet	0
Retweet	1
Quote Retweet	2

Table 33: Tweet Type to Integer Enumeration Mapping

C. Sentiment Analysis Techniques

C-1. Penn Treebank Part-of-Speech Tagset

Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential *there*
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun

Tag	Description
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Reference: [12]

C-2. Universal Part-of-Speech Tagset

Tag	**Meaning**	**English Examples**
ADJ	adjective	new, good, high, special, big, local
ADP	adposition	on, of, at, with, by, into, under
ADV	adverb	really, already, still, early, now
CONJ	conjunction	and, or, but, if, while, although
DET	determiner, article	the, a, some, most, every, no, which
NOUN	noun	year, home, costs, time, Africa
NUM	numeral	twenty-four, fourth, 1991, 14:24
PRT	particle	at, on, out, over per, that, up, with
PRON	pronoun	he, their, her, its, my, I, us
VERB	verb	is, say, told, given, playing, would
.	punctuation marks	. , ; !
X	other	ersatz, esprit, dunno, gr8, univeristy

Reference: [13]

C-3. Penn Treebank to Universal POS Tagset Mappings

PTB	Universal
''	.
(	.
,	.
.	.
:	.
BES	VERB
CC	CONJ
CD	NUM
DT	DET
EX	ADV
FW	X
GW	X
IN	ADP
JJ	ADJ
JJR	ADJ
JJS	ADJ
MD	VERB
NN	NOUN
NNP	PRON
NNPS	PRON
NNS	NOUN
PRP	PRON
PRP$	ADJ
RB	ADV
RBR	ADV
RBS	ADV
RP	PRT
SYM	.
TO	PRT
UH	X
VB	VERB
VBD	VERB
VBG	VERB
VBN	VERB
VBP	VERB
VBZ	VERB
WDT	ADJ
WP	NOUN
WRB	ADJ
X	X
^CC	CONJ
^IN	ADP
^JJ	ADJ
^NN	NOUN
^NNP	NOUN
^NNS	NOUN
^PRP	PRON
^PRP^VBP	ADJ
^RB	ADV
^UH	X
^VB	VERB
^VBD	VERB
^VBG	VERB
^VBN	VERB
^VBP	VERB
^VBZ	VERB
^WP	ADJ
^WRB	ADV