FACULTY OF INFORMATION TECHNOLOGY CTU IN PRAGUE

ASSIGNMENT OF BACHELOR’S THESIS

Title:	Classifying Twitter election news as FakeNews
Student:	Alina Vigriyanova
Supervisor:	Ing. Jaroslav Kuchař, Ph.D.
Study Programme:	Informatics
Study Branch:	Web and Software Engineering
Department:	Department of Software Engineering
Validity:	Until the end of summer semester 2019/20

Instructions

The FakeNews is a deliberate disinformation or propaganda in media. The problem of fighting FakeNews is presenting one of the current challenges data scientists are facing. The aim of this work is to design, implement and evaluate a simple tool enabling exploring the possibilities of classifying news by Data Science techniques.
- Get familiar with the Twitter elections dataset and its properties.
- Obtain a counterpart of the dataset, i.e. not FakeNews.
- Describe the NLP methods overview and preprocessing suitable for the selected dataset.
- Design, implement and evaluate a tool that will be able:
Compare binary classification using the content, metadata and combined.
Perform the topic modeling and analysis.
- Do the analysis of the sources, based on the topic analysis and any other relevant data extracted from the dataset, compare the results with information about the previous retweets, time precedence or ground truth classification.

References

Will be provided by the supervisor.

Ing. Michal Valenta, Ph.D.
Head of Department

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Dean

Prague February 14, 2019

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Bachelor's thesis

Classifying Twitter election news as FakeNews

Alina Vigriyanova

Department of Software Engineering
Supervisor: Ing. Jaroslav Kuchař, Ph.D

May 9, 2019

Acknowledgements

I would like to thank my supervisor Ing. Jaroslav Kuchař, Ph.D. for supporting me with the topic and for all his valuable advice during the creation of the thesis.

Above all, I would like to thank my family for giving me all the opportunities I had and for all the support during my studies.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on May 9, 2019 . . . .

Citation of this thesis

Vigriyanova, Alina. *Classifying Twitter election news as FakeNews.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2019.

Abstrakt

Cílem této bakalářské práce je oblast analýzy dat z aplikace Twitter za účelem výběru spolehlivých zpravodajských a volebních zpráv, analýza metod detekce FakeNews a jejich výsledků, prozkoumání vlivu modelování témat (topic modeling) na výslednou klasifikaci, a implementovat nástroj umožňující klasifikaci zpráv a vyhodnocení různých přístupů. Pro zpracování textu zpráv byly využity metody zpracování přirozeného jazyka jako je rozdělení na menší jednotky (slova), převod do základního tvaru, odstranění nedůležitých slov nebo vektorizace. Pro klasifikaci zpráv a výsledné porovnání bylo použito několik metod jako je: logistická regrese, metoda podpůrných vektorů nebo rozhodovací stromy. Pro modelování témat byla vybrána metoda LDA.

Klíčová slova FakeNews, Zprávy, Twitter, Metody zpracování přirozeného jazyka, Klasifikaci, Modelování témat.

Abstract

This bachelor thesis aims to analyze Twitter archives of potentially state-backed Tweets, find a way of selecting reliable news from Twitter, obtain its counterpart of not Fake News, try different approaches to detect Fake News, analyze the approaches outcome, investigate possibilities of using topic modeling on the problem Fake News classification, and finally implement a tool that can help to classify FakeNews and evaluate the results. For text preprocessing NLP methods such as tokenization, stemming, stop words removal vectorizing were used. Logistic Regression, Linear Support Vector Classification, and Decision Tree classifiers were used to classify and evaluate the data. For topic modeling, Latent Dirichlet allocation was chosen.

Keywords FakeNews, News, Twitter, NLP, Classification, Topic modeling.

Contents

List of Figures

List of Tables

Introduction

The modern society produces and digests gigabytes of information every single day. One type of such information is the news. People hear the news on the radio, watch the news on the TV, and of course read it on the official news provider's web cite, in social media or even in the printed newspaper.

While newspapers' popularity decreasing, more and more people get their news from the social media. It's easy, it's fast and it's just always on hand. Indeed, isn't it the most convenient way to just find out what's going on? In terms of user experience it probably is, however there is a major downside of the social media news. There is a possibility that the news is a fake. The thing is, in social media nobody is responsible to create a quality, fact checked content.

Therefore, there is a need in finding a solution to change this convenient way of getting the news to make it not only convenient but also reliable.

Various attempts have been made by the social media platforms and government to solve this problem. From blocking the Fake News accounts, to the creation of the government regulations that prohibits to post Fake News.Yet, before deleting not reliable information, it has to be found and evaluated first. The existing methods include humans to read and evaluate news, however, considering the amount of information that is posted in the Internet every minute it is not only insufficient for professionals to check everything, but also impossible.

Thus, an automatic way of finding and evaluating Fake News should be established. In this work, an attempt to find such way is presented. More specific, Twitter news and recently published Twitter archives of potentially state-backed Tweets are explored.

Outline

Chapter **State-of-the-Art** describes the main definitions and State-of-the-Art techniques.

In chapter **Dataset Analysis** an analysis of the Twitter archives of potentially state-backed Tweets was performed and on it's basis the dataset of Not Fake reliable News was acquired and analyzed.

In chapter **Experiments and evaluations** various experiments on the obtained data are conducted, evaluated and analyzed for the purpose of finding out which techniques work best.

Finally, the tool for further experiments is developed in chapter **The Tool**.

Objectives

The main goals of this work are:

1. Analyze Twitter archives of potentially state-backed Tweets.
2. Find a way to collect the Not Fake News dataset.
3. Select NFN for the dataset.
4. Create the NLP methods overview.
5. Describe the preprocessing suitable for the dataset.
6. Design, implement and evaluate a tool for comparing classifications on text, metadata and combined. And, perform topic modeling and analysis.
7. Do the analysis of the sources.

Chapter 1

State-of-the-Art

1.1 Approaches definitions

In this section all the theoretical approaches used in the thesis are described.

1.1.1 Data preprocessing approaches

Tokenization [1] is a process of splitting a text sample into its atomic parts, so-called tokens. There is no standardized approach, therefore for every task, it is possible to have different size of a token, i.e. the whole sentence, word, or even a single character. Moreover, even if two tokenizers use words as tokens, they may have a different definition of a word or consider certain idioms as one token.
Tokenization is a vital part of natural language preprocessing because any further steps will require the dataset of these atomic parts of a text.

Stemming [2] is a technique that reduces words to its root form with a purpose of representing different forms of a word as the same word for the algorithm and therefore lower the overall number of distinct words.

This process has it's limitations because of the complexity of human languages, therefore using lemmatization instead of stemming can in theory significantly improve the results.

Lemmatization [2] also reduces words to its root form, therefore reduces the number of word's forms in a text, but unlike stemming it additionally considers word's semantic meaning.

Stop words [3] are words that are frequent in any text and therefore do not bring any information for distinguishing different classes. For example, in English, one of those words is ”the”.

Removing stop words slightly improves classification accuracy, however, another advantage of it is decreasing dimensionality and therefore improving classification performance.

Text data **vectorization** [4] is a process of representing every text sample as a vector of numbers that can be further used in classifications. The most common approaches are bag of words(CountVectorizer) and TF-IDF. Bag of words also named as term frequency is the simplest of all approaches. The main idea is to create a vector where each item represents the quantity of each document-unique word in the sentence.

TF-IDF is a modification of the bag of words that help to reduce the weight of words that are very frequent in every document and therefore less informative. It is defined as follows:

$$tf - idf(t,d) = tf(t,d) \cdot idf(t),$$

and the idf is computed as:

$$idf(t) = \log \frac{n}{df(t)}$$

where n is a total number of documents, and $df(t)$ is the document frequency for the document t.

For both approaches above, the resulting vector is of a size of the number of unique words in the document. This should be considered during the implementation process as the dimension of the vector can significantly affect the classification's performance. One of the solutions for this problem can be limiting maximin and minimum frequency.

1.1.2 Data classification approaches

Logistic Regression [5] is a statistical method for predicting binary classes based on previous observation of data (supervised classifier). Sigmoid hypothesis function is used to calculate the probability of y belonging to a particular class. Sigmoid function:

$$S(x) = \frac{1}{1 + e^{-x}}$$

Linear Support Vector Classification is a Support Vector Machine (SVM) with linear kernel. Support Vector Machine is a supervised classifier defined by a separating hyperplane. In more details SVM is described in [6].

Decision Tree Classifier constructs a decision tree model which is later used for a prediction. Decision tree model uses simple decision rules to predict the target variable. The approach was deeper described in [7].

Latent Dirichlet allocation is a supervised model that can divide a text corpora into a predefined number of topics. After training, for the new entry LDA computes probabilities of how probable the entry refers to each topic. The article by David M. Blei, Andrew Y. NG and Michael I Jordan [8] describes LDA in details.

Sentiment analysis (Opinion Mining) Natural Language Processing (NLP) field dedicated to determine human opinion expressed in text. There exist many types of sentiment analyses based of the desired outcome. For example, sometimes the analysis aims to determine weather a text is positive or negative and sometimes weather the text includes some emotions.

1.2 State-of-the-Art fake news classification methods

Existing approaches for determining fake news can be divided into the following groups:

- Linguistic-based: aimed at the content of text.
- Network-based: aimed at the relations between users.
- Visual-based: aimed at image or video content.
- Metadata-based: aimed at post's or author's metadata, i.e. number of likes or followers.

Among the **Linguistic-based** most used detection approaches are following:

Direct application of classifiers on lexical or syntactic features of text - observation of text features such as characters, words or n-grams frequencies, punctuation or part-of-speech (POS) tagging, number of characters per word, etc. An good example of classifying text using n-grams is the work by Hadeer Ahmed, Issa Traoreand and Sherif Saad [9] where authors showed an effective n-gram model for the fake news and reviews classification.

Deep Syntax - uses Probability Context Free Grammars (PCFG) to analyze deeper language structures to predict deception. The core idea of the method is to use a set of rules with corresponding probabilities of the rule's occurrence. PCFG application in NLP is explained in details in [10]. Even though PCFG approach is not showing the best results in text classification, it can improve models that use TF-IDF features [11].

[Sentiment analysis] algorithms can be divided into three major groups:

- Rule-based - systems that use some set of predefined rules, i.e. uses lexicons to determine polarity of each word. According to [12] the best Sentiment Lexicons in English are SenticNet [13] and SentiWordNet [14].

- Automatic (Corpus-based) - systems that use machine learning. Most of such systems are supervised and requires a training dataset. Among the best implemented algorithms are [15] by Han-Xiao Shi *et al.* and [16] by Erikand Boiy *et al.*

- Hybrid - a combination of rule-based and automatic systems. Mizumoto et al. [17] introduced an approach where the polarity of a small part of words was decided manually, however the polarity of new words was determined automatically.

Stance Detection - determining weather a text has the same topic as it's headline. The Fake News challenge [18] encouraged many researchers to explore this approach in the Fake News classification. For example, the winner team of the challenge used 50/50 weighted average between gradient-boosted decision trees and a deep convolutional neural network. As for the preprocessing, a combination of various features was used, including TF-IDF, SVD, Word2Vec and Sentiment features.[19]

Network-based approaches such as co-occurrence network and friendship network are explored in articles by Natali Ruchansky et al. in [20] and Sejeong Kwon et al. in [21].

One of the versions of **visual-based** techniques is described in [22]

Metadata-based approaches can be divided into Post-level, Author-level, Topic-level and Propagation-level.

Post-level metadata analysis includes any feature that can be extracted from a post. Sometimes linguistic-based approaches are used to create such features.

Author-level metadata can be everything that is possible to extract from a page of the account created a post, such as a number of followers, registration date or number of posts.

Topic-level metadata is the metadata which is common for a group of posts of the same topic.

Concerning the Twitter domain, an example of such group can be Tweets with the same hashtag, then the most obvious feature is the number of Tweets associated with this hashtag. The article by Carlos Castillo, Marcelo Mendoza, and Barbara Poblete [23] explores classification by various types of metadata in details.

For this work combination of Linguistic and Metadata-based approaches were used.

1.3 Existing Tools

Fake News Detector [24] uses users contributions to decide if some content is fake or not. After a user flags some content as Fake News, Click

Baits or Extremely Biased other users can see it. Moreover, the tool's AI is analyzing this data and tries to flag news automatically.

FakerFact [25] is not classifying news into fakes and not fakes directly, but shows the probability of the news being Journalism, Wiki, Satire, Sensational, Opinion or Agenda-driven. This helps users to decide if the news is a fake.

Chapter 2

Dataset analysis

For the classification it is required to obtain two datasets. One is a dataset of fake news (later FN) and another one is a dataset of not fake news (later NFN).

2.1 Fake news dataset

FN dataset is a public version of Twitter archives of potentially state-backed Tweets. More concrete, Internet Research Agency's archive is used because it is the biggest and includes Tweets from different countries.

The dataset can be obtained by downloading it from the Twitter official web pages [26].

2.1.1 Initial preprocessing of FN dataset for the classification

FN dataset contains 32 features that may need preprocessing prior to the classification. Prior to looking at every feature separately the dataset reductions took place:

1. Extract only Tweets which are in English. A Tweet is considered to be in English if *tweet_language* is either "en" or "uk". Despite some Tweets with *tweet_language* equal to 'NaN' may be in English, it was decided to drop them because it is a relatively small group of Tweets compared to the dataset size.

2. Extract only original Tweets, i.e. not retweets or replies.

3. Extract only Tweets which are not polls. A poll does not bring any information to the standard text classifications because consists of different opinions. Moreover, only approximately 0.01% of all the Tweets are polls, which is too small number for separate classification.

4. Following features were deleted:

tweetid is unique, hence does not bring any information to the classification

userid, user_profile_url, user_display_name, user_screen_name are mostly hashed therefore it's not possible to compare it with NFN dataset

account_language, tweet_language, is_retweet, in_reply_to_userid, retweet_tweetid, in_reply_to_tweetid, quoted_tweet_tweetid, retweet_userid do not bring new information because of the previous reductions

latitude, longitude has too small number of not Null values

user_reported_location is actually a text field without any rules so it is possible for the feature to be unique for almost any account

tweet_client_name is impossible to get for the NFN dataset

Further, each feature is described separately in the Table B.1 in the [**Appendix**].

2.1.2 Features analysis

After all, the following features left:

Numerical	**Categorical**	**Textual**
follower_count	hashtags	user_profile_description
following_count	urls	tweet_text
account_creation_date	user_mentions	
tweet_time		
quote_count		
reply_count		
like_count		
retweet_count		
hashtags_count		
url_count		
mentions_count		

At this point analysis is very important because it has direct impact on the NFN dataset assembly.

2.1.2.1 Numerical features

Probably the most important feature for future selecting NFN dataset is *tweet_time*. It is crucial to cover approximately the same time period because this way nearly the same topics will be covered. In fig. 2.1 *tweet_time* distribution is shown and compared with *account_creation_time*.

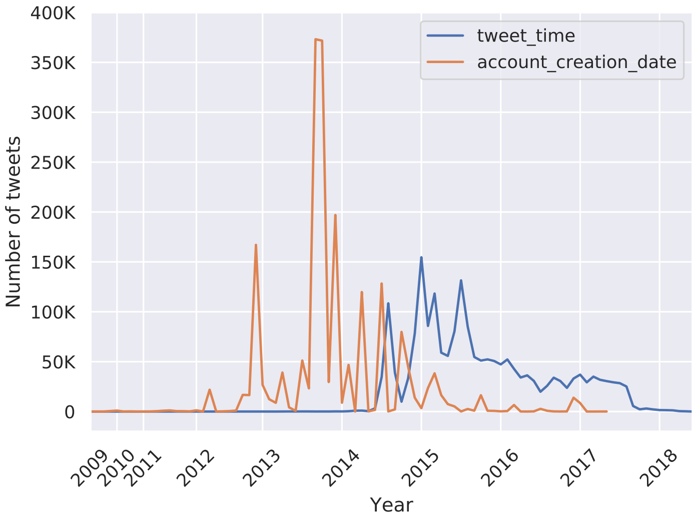


Figure 2.1: Number of Tweets vs. creation date and Number Tweets vs. account's creation date

Another valuable feature is *follower_count*. As it is shown in the figure 2.2, most of the fake news accounts have little to none followers. And even though the mean is around 15000, median is only 7076 followers, which is still more then the Twitter's average of 707 followers [27]. Therefore it is possible to conclude that most of the accounts are media.

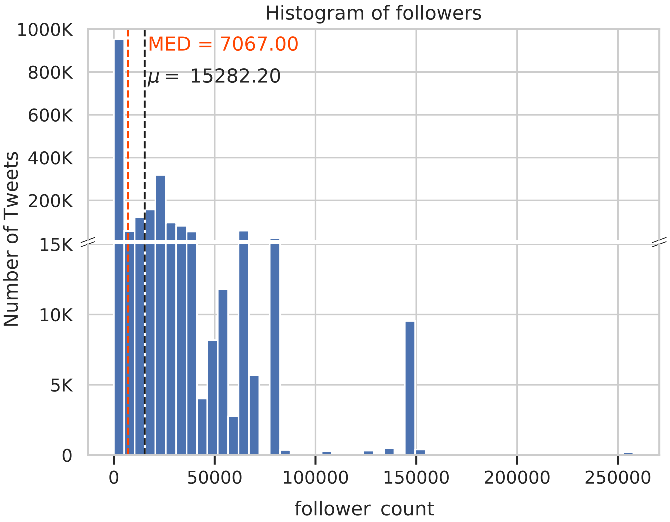


Figure 2.2: Number of FN Tweets with the following number of author's followers

Figure 2.3 shows the similar behavior, however unlike the number of followers at this point of analysis no assumptions can be done.

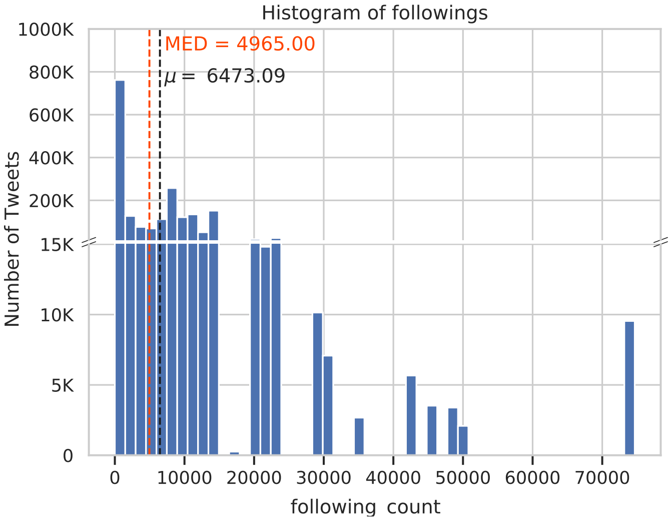


Figure 2.3: Number of FN Tweets with the following number of author's followings

In figs. 2.4 and 2.5 likes, retweets, hashtags, URLs, quotes, replies, and mentions distributions are shown. However those figures are mostly needed to be compared against NFN dataset, it is already possible to make some deductions. For example, that most of the fake news were unnoticed by the Twitter users.

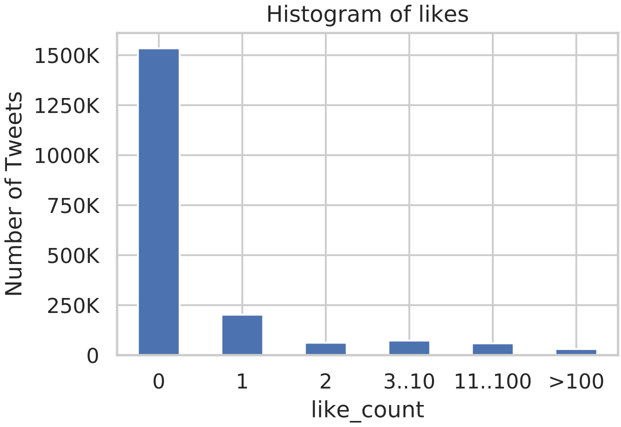


Figure 2.4: Number of FN Tweets with the following number of likes

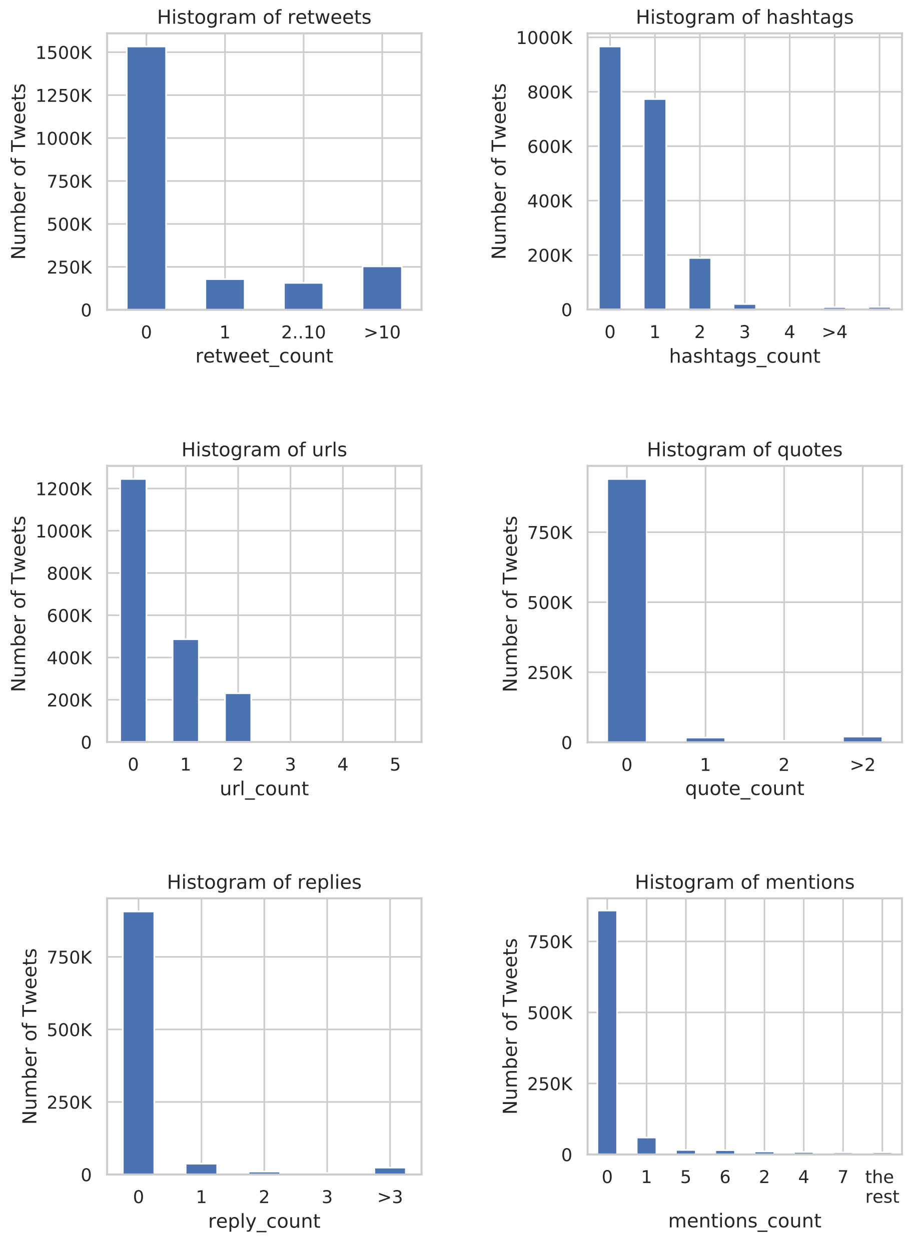


Figure 2.5: FN Tweet's retweets, hashtags, URLs, quotes, replies, mentions distributions

2.1.2.2 Categorical features

Among categorical features ”hashtags” is the most interesting because in the Twitter domain hashtags serve almost a topic purpose. Top 30 hashtags can be seen in fig. 2.6. Most of the Tweets has no hashtags, hashtags ”news”, ”sports”, ”politics”, ”local”, ”world”, ”business” seems logical as the main topic of the dataset is news. Others seems rather random at this point.

Another feature that may be interesting is *urls*. However, as at is shown in fig. 2.7 most of the top URLs are actually URL shortening services so it’s not obvious where those links lead.

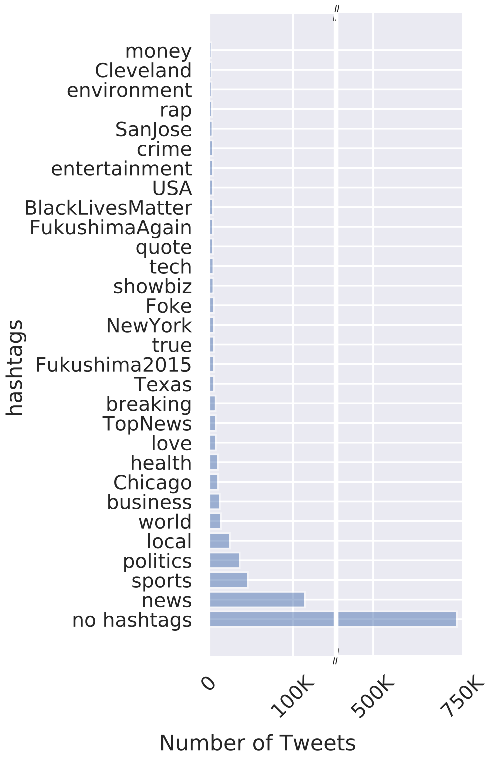


Figure 2.6: Number of FN Tweets with the following hashtags

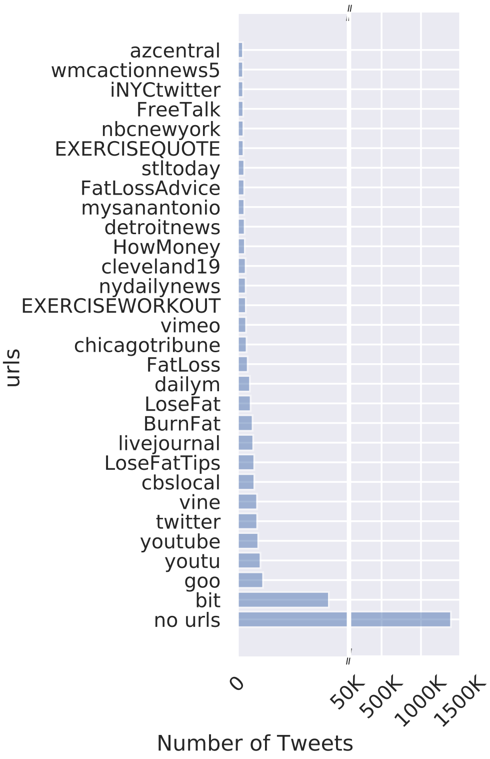


Figure 2.7: Number of FN Tweets with the following URLs

2.1.2.3 Textual features

The one most interesting features is surely the Tweet’s text. To find words that are characteristic to the dataset, [TF-IDF] and [CountVectorizer] approaches were chosen. Moreover, only a sample of the dataset was used because of the high memory usage of the implementation. The sample size and preprocessing is described more deeply in the [Experiments] chapter.

As figs. 2.8 and 2.9 show, the most characteristic words for the FN dataset are ”news”, ”trump” and ”sport”. Which can be explained as the dataset represents news, the most discussed topic of the time frame was the American Elections and sport is just common to the news.

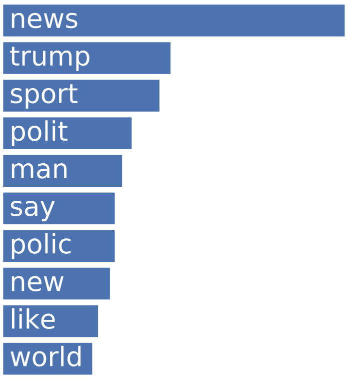


Figure 2.8: Top 10 of FN Tweets with CountVectorizer

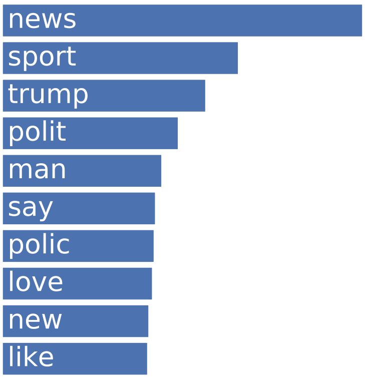


Figure 2.9: Top 10 of FN Tweets with TF-IDF vectorization

2.2 Not fake news dataset

The NFN dataset contains not fake news Tweets scraped directly from Twitter. The reason for not using Twitter API is that it only allows to get Tweets published in the past 7 days, therefore it wouldn't be possible to get Tweets from the required time period.

It was decided to choose a number of trusted Twitter accounts and scrape Tweets from approximately the same time frame as Tweets from FN dataset. With a help of fig. 2.1 01.04.2014 - 31.12.2017 period was chosen.

The above decision implies that first of all it is necessary to select the trusted Twitter news providers. To make this choice as objective as possible a few steps were taken:

1. Search for any big news providers in English.
2. Crosscheck with Forbes article by Paul Glader about reliable news sources [28].
3. Crosscheck with `mediabiasfactcheck.com`.
4. Choose only those with at least high factual reporting and only left-center, right-center or least biased.
5. Check if the news source has a Twitter account.

After performing the steps above following accounts were chosen:

Associated Press (@AP)
Reuters (@Reuters)
PBS NewsHour (@NewsHour)
National Public Radio (@NPR)
Bloomberg News (@business)
BBC (@BBCNews)
Washington Post (@washingtonpost)
ABC News (@ABC)
CBS News (@CBSNews)
The Hill (@thehill)
Denver Post(@denverpost)
NBC News (@NBCNews)
Propublica (@propublica)
USA Today (@USATODAY)
LA Times (@latimes)
The Atlantic (@TheAtlantic)
The Guardian (@guardian)
The Economist (@TheEconomist)
Politico (@politico)
Reason (@reason)
The Fiscal Times (@TheFiscalTimes)

2.2.1 Obtaining NFN dataset

To obtain NFN dataset a nice tool by Jefferson Henrique was chosen [29]. It helps to scrape Tweets from required time period with required account name and much more.

Yet, some improvements had to be made to entirely cover this work's needs.

1. For features as *account_creation_date*, *follower_count*, *following_count* and *user_profile_description* new functionality had to be added. To get those features the tool now not only scans the Tweet's page but also the account's page associated with the Tweet.

2. Added scanning for *urls* feature.

3. Fixed a small bug of getting mentions and hashtags.

4. Added *mentions_count*, *hashtags_count*, *urls_count*. Although, it is possible to do this later in this work's implementation, it was decided to leave it to the GetOldTweets tool.

5. Added *contents* attribute to Tweet object. It is a list of all required features for easy access to the tool's output.

And still, some features is impossible to get with this method. Even though it is possible to search for quotes, there is no way to see the overall number, and *tweet_client_name* is not shown on Twitter web site at all.

The last touch was to rename features to comply with FN dataset.

2.2.2 Features analysis

As the dataset was initially assembled to fit the needs of this work, no extra preprocessing is needed, except for minor type and format conversions. At this point it is important to perform feature analysis and comparison with the FN dataset features.

2.2.2.1 Numerical features

Immediately fig. 2.10 shows a major difference between two datasets. Because of the approach that was chosen to assemble NFN dataset, NFN tweets have much higher number of followers. This implies that unfortunately with this datasets it is better not to use this feature directly for the classification because having these datasets does not mean that news can be not fake if and only if the provider is very popular.

However, fig. 2.12 shows the opposite behavior of the following_count feature and this can actually be explained as the fake news providers tend to spread more aggressively.

Moreover, fig. 2.11 are quite different from the FN's corresponding figures. NFN dataset contains much more Tweets with a larger number of retweets, likes, and replies, which is logical according to the follower_count analysis. On the other hand, NFN Tweets tend to contain less number of hashtags, URLs, and mentions, which can be again linked with the big effort FN providers put into spreading.

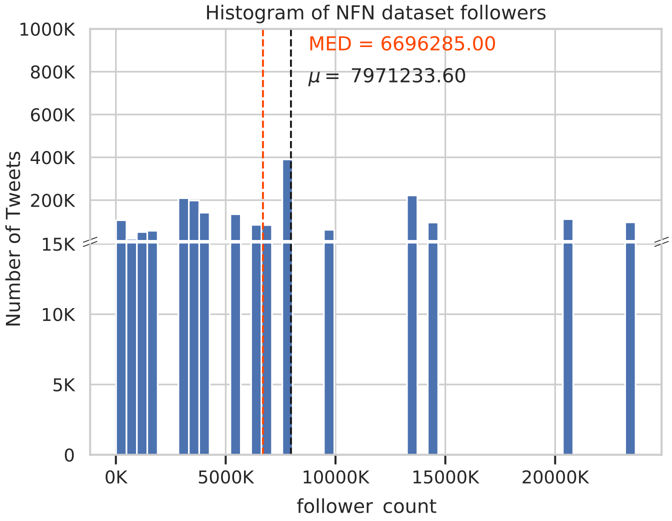


Figure 2.10: Number of NFN Tweets with the following number of author's followers

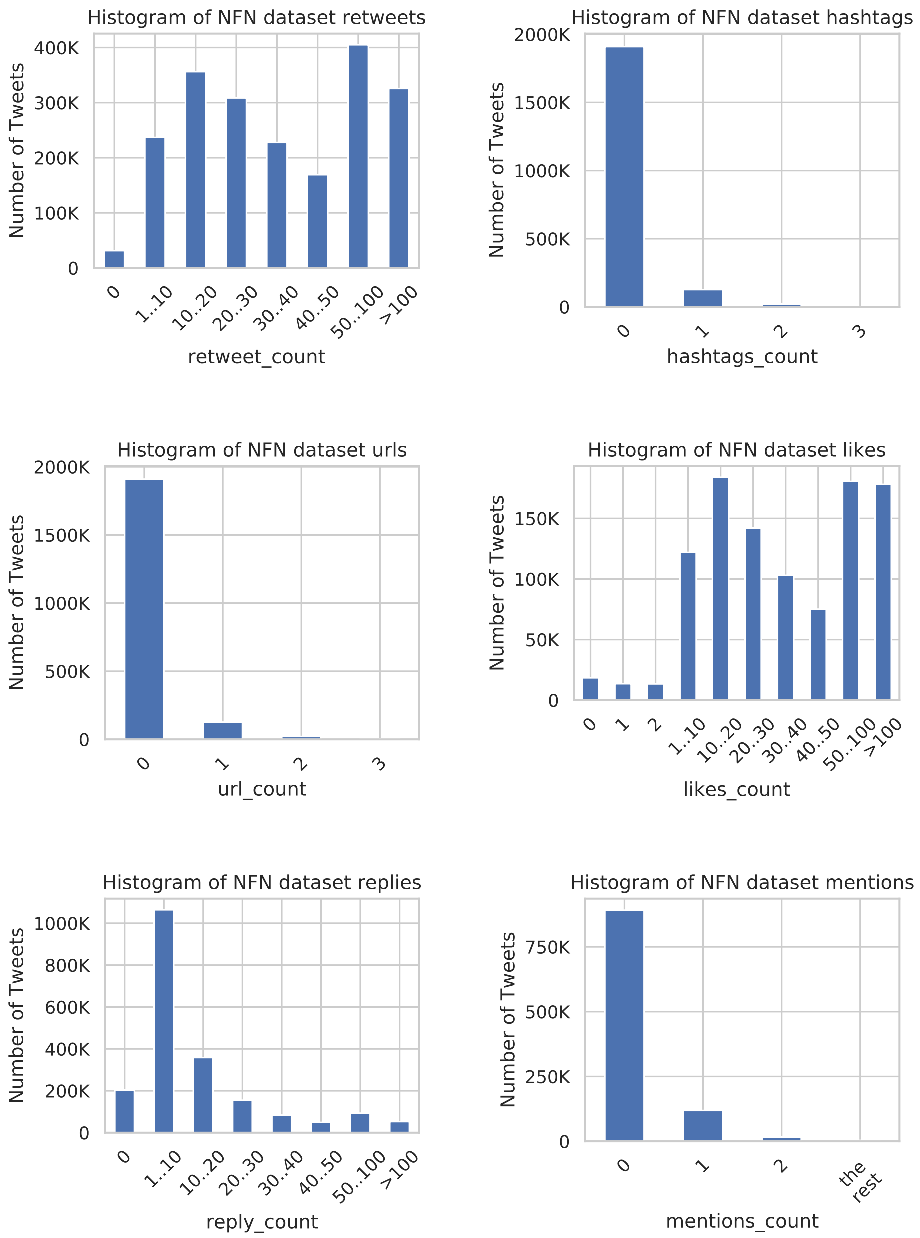


Figure 2.11: NFN Tweet's retweets, hashtags, URLs, quotes, replies, mentions distributions

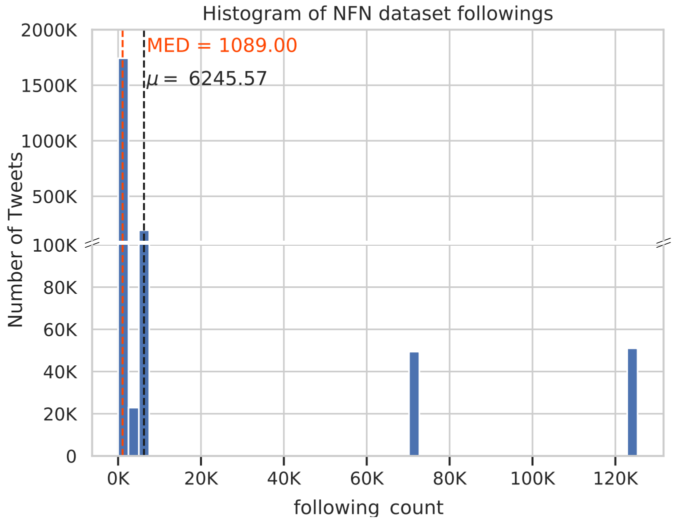


Figure 2.12: Number of NFN Tweets with the following number of author's followings

2.2.2.2 Categorical features

In fig. 2.15 there are the top 30 hashtags of the NFN dataset. Comparing it with the FN top 30 hashtags (fig. 2.6) shows that even larger part of NFN has no hashtags. Next, having the second most usable hashtag ”BREAKING” and not ”news” as in FN dataset can be explained as only news providers were used for the NFN dataset and almost every Tweet of such accounts is news so it makes sense not to use #news every time. However, the use of ”BREAKING” hashtag is justified by the fact that only some news are breaking news. As for the rest of the hashtags, they are more connected to some special events. Overall, FN hashtags are more general comparing to the NFN hashtags.

Concerning the top NFN dataset's URLs (fig. 2.16), what is interesting is that there is no YouTube, Vine or Vimeo among the top URLs. And again, in general, it is more specific then the FN top URLs (fig. 2.7). The reason is probably that the most of the top providers link a Tweet with the whole article at the provider's official web site.

2.2.2.3 Textual features

For the textual features the same preprocessing steps and sample size were used as with the FN dataset. Figures 2.13 and 2.14 shows that ”trump” is still the most discussed figure of the elections period, however unlike FN dataset(figs. 2.8 and 2.9), ”clinton” and ”obama” as well. Moreover, ”news” and ”sport” are not in top ten features at all.

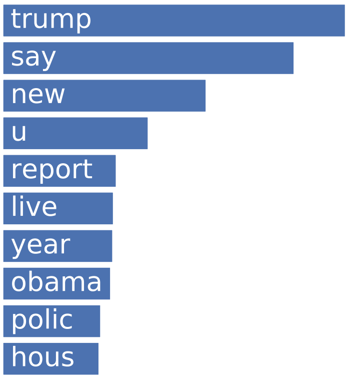


Figure 2.13: Top 10 of NFN Tweets with CountVectorizer

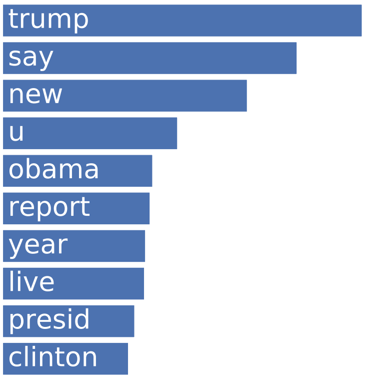


Figure 2.14: Top 10 of NFN Tweets with TF-IDF vectorization

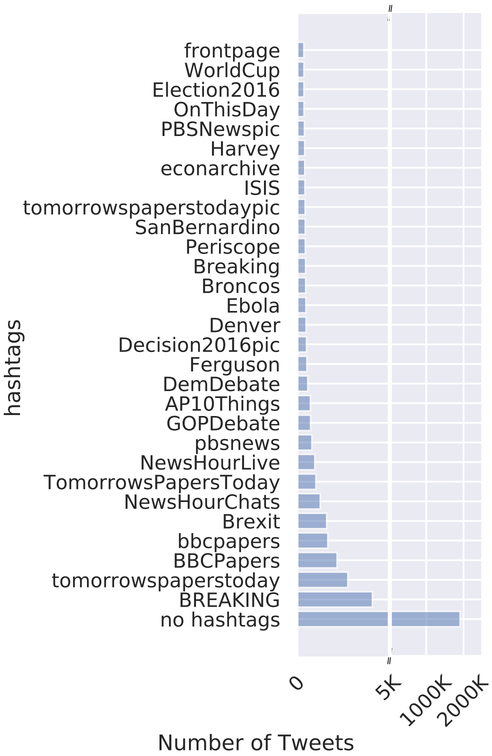


Figure 2.15: Number of NFN Tweets with the following hashtags

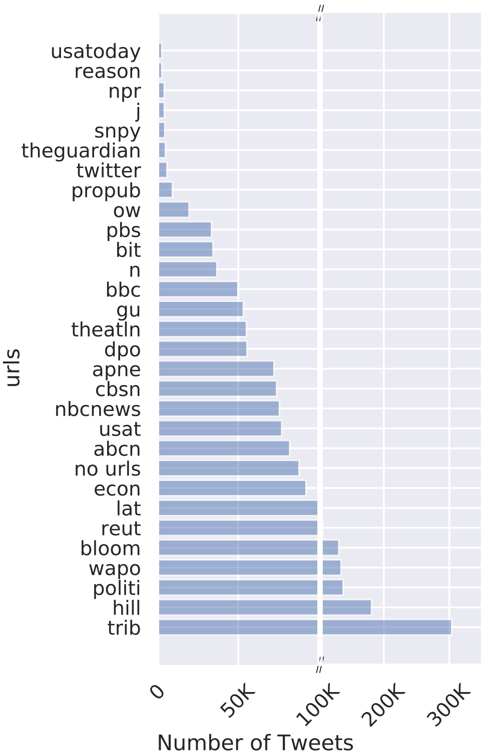


Figure 2.16: Number of NFN Tweets with the following URLs

Chapter 3

Experiments and evaluations

The core idea of the work is not to use Tweet as it is for the classification but to create a new feature vector from every tweet and classify this new feature vector.

In this chapter, first the Baseline Experiment is described and implemented, then each new experiment adds new features to the Baseline Experiment as new columns to the initial vector DataFrame.

3.1 Baseline experiment

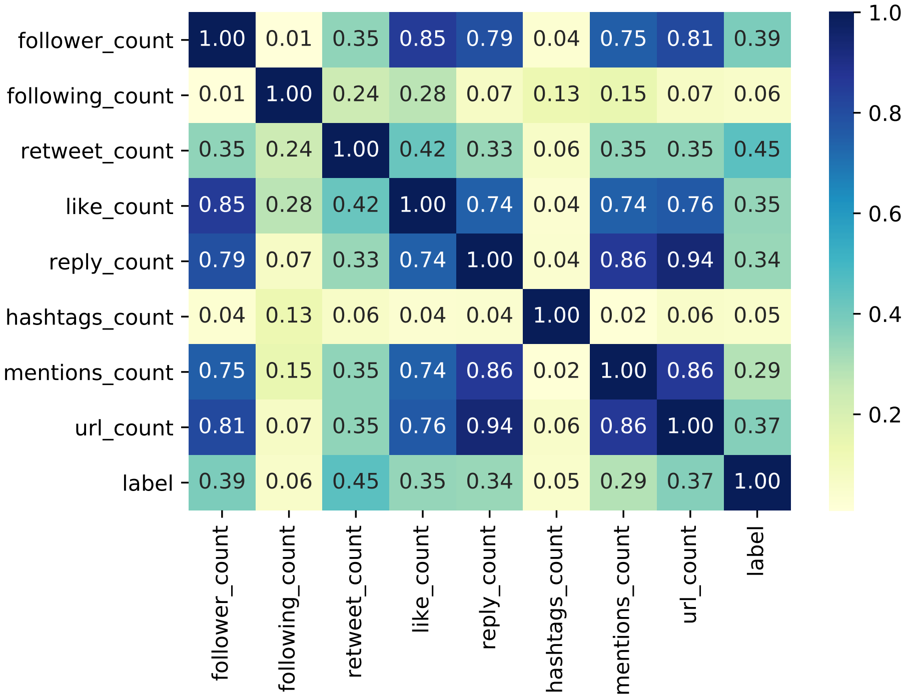
The main point of the baseline approach is to create a basic classification that can be improved in further experiments. This includes choosing a subset of features for the basic feature vector, choosing the sample size that is small enough for reasonably good performance and big enough for the accurate classification, and implementing the basic algorithm for the classification itself.

For the initial vector it is important to take into account that obtained NFN dataset is not perfect as it only includes news from big news providers, however FN dataset was assembled by professionals and includes various accounts. For this reason it is required to be very cautious with *follower_count* feature as it is shown in figs. 2.2 and 2.10. The decision was not to use it for the baseline feature vector.

Moreover, some other features can be dependent on the number of followers. Logically it can be deducted that one of those features can be *like_count* because the more people follow the account the more people see the Tweet. Figure 3.1 shows that indeed *like_count* is very correlated with *follower_count* along with *reply_count*. Interestingly, *retweet_count* is not.

Although *mentions_count* and *url_count* are highly correlated with *follower_count* as well, these features are purely post-level and can not be dependent on the number of followers.

Furthermore, it is better to exclude *following_count* for the same reason as *follower_count*.



	follower_count	following_count	retweet_count	like_count	reply_count	hashtags_count	mentions_count	url_count	label
follower_count	1.00	0.01	0.35	0.85	0.79	0.04	0.75	0.81	0.39
following_count	0.01	1.00	0.24	0.28	0.07	0.13	0.15	0.07	0.06
retweet_count	0.35	0.24	1.00	0.42	0.33	0.06	0.35	0.35	0.45
like_count	0.85	0.28	0.42	1.00	0.74	0.04	0.74	0.76	0.35
reply_count	0.79	0.07	0.33	0.74	1.00	0.04	0.86	0.94	0.34
hashtags_count	0.04	0.13	0.06	0.04	0.04	1.00	0.02	0.06	0.05
mentions_count	0.75	0.15	0.35	0.74	0.86	0.02	1.00	0.86	0.29
url_count	0.81	0.07	0.35	0.76	0.94	0.06	0.86	1.00	0.37
label	0.39	0.06	0.45	0.35	0.34	0.05	0.29	0.37	1.00

Figure 3.1: Correlation matrix

At the end, the following features was chosen:

- TF-IDF features of *tweet_text*
- *retweet_count*
- *hashtags_count*
- *url_count*
- *mentions_count*

As the sampling method, random sampling with a small modification was used. The modification is that it was decided to take some static number of random tweets for each month to be sure that different events throughout the years were covered.

Finally, for the basic classification algorithm three classifiers were used: Logistic Regression, Linear Support Vector Classification (LSVC) and Decision Tree. The choice is justified by the fact that among the most popular classifiers those three are rather different in their logic and implementation so it is possible to see how various experiments works with those different models.

3.1.1 Implementation

First of all it is required to perform sampling because the initial dataset is relatively large and processing such a large dataset can be inefficient. To choose the best sample size it was decided to run a baseline experiment with samples of different size and analyze the results. However, before that it is needed to implement the model.

For the *tweet_text* additional preprocessing was accomplished. The first step was to remove of Twitter specific parts such as user mentions, hashtags, emoji and URLs.

Next, tokenization, stemming, stop words removing and vectorization took place. Tokenization was performed by the TweetTokenizer [30], stemming by the PorterStemmer [31], stop words removal with the help of the NLTK stopwords [31], and vectorization by Scikit-Learn's TfidfVectorizer [32]. TF-IDF approach was chosen because of the best balance of simplicity and quality.

The TweetTokenizer was set up to remove emojies, URL's, hashtags and handles, to make text lowercase and to shorten all the consequently repeated characters. Additionally, everything which is not English letter or number was removed because a lot of extra symbols is used by the Twitter users.

The model was implemented using Scikit-Learn's Pipelins and Feature Unions. The detailed scheme of the model is shown in the Figure 3.7, where "Additional preprocessing" implies format's conversions and merging datasets into one labeled DataFrame, and "Prediction model" implies applying classifiers that were discussed above.

3.1.2 Sample size selection

The results of running a basic model on different sample sizes is shown in Figure 3.2. From this graph it was observed that the best sample size is 400 random samples per month for each class.

3.1.3 Post sampling preprocessing

Because of the size of the dataset it was decided to perform some preprocessing after the sampling. It was required to double check languages because even though language filtering was already done during the initial preprocessing, FN dataset had errors in the *tweet_language* feature. It is important to be sure that only English results are included because having some non-English words in one class of the dataset can significantly influence the classification. To double check languages, first, it was checked if the Tweet has only the English alphabet characters, than with the help of langdetect [33] python library all non-English Tweets were deleted. The first check is required because langdetect results are not perfect and because of the performance reasons.

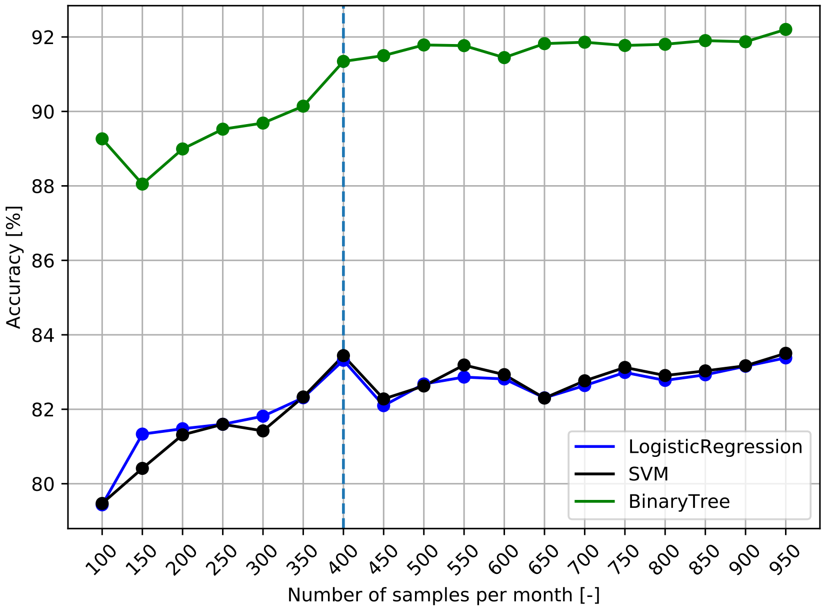


Figure 3.2: Classification results on different numbers samples

3.1.4 Evaluation

The detailed performance of the baseline approach is shown in the Table 3.1.

Table 3.1: Results of the baseline experiment.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	82.26%	79.85%	82.45%	81.13%	82.39%
LSVC	82.11%	80.29%	81.83%	81.05%	82.21%
Decision Tree	91.12%	91.18%	90.18%	90.68%	91.11%

3.2 Experiment I (followers and following)

Experiment I investigates the possibility to normalize *follower_count* by *following_count* feature. New feature *followers_followings_ratio* = *follower_count* / (*following_count* + 1) was created. Adding one suppresses the division by zero.

3.2.1 Evaluation

Table 3.2 shows 99% accuracy for the Decision Tree classifier, which means that the new feature probably inherits the problem of *follower_count* and *following_count*, on the other hand new feature slightly improves the results of the other two classifiers that are not that sensitive to this kind of problems.

Table 3.2: Results of the Experiment I.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	82.45%	81.07%	81.82%	81.44%	82.53%
LSVC	83.11%	82.33%	82.10%	82.21%	83.15%
Decision Tree	99.15%	99.22%	98.98%	99.10%	99.15%

3.3 Experiment II (like and reply count)

This experiment is dedicated to explore features dependent on *follower_count* feature. To normalize such features each of them can be divided by the (*follower_count* + 1). Adding one suppresses the division by zero. Thus, new features *reply_to_follower_count_ratio* and *like_to_follower_count_ratio* were introduced.

3.3.1 Evaluation

Tables 3.3 and 3.4 shows that new features still have the problem of being too different between classes.

Table 3.3: The description of *like_to_follower_count_ratio* feature.

label	FN	NFN
count	27000	27000
mean	0.001748	0.000015
std	0.064953	0.000172
min	0	0
25%	0	0.000002
50%	0	0.000004
75%	0	0.000010
max	8.551259	0.026087

Table 3.4: The description of *reply_to_follower_count_ratio* feature.

label	FN	NFN
count	27000	27000
mean	0.000117	0.000003
std	0.002594	0.000010
min	0	0
25%	0	0
50%	0	0.000001
75%	0	0.000002
max	0.333333	0.000718

Therefore, as expected Table 3.5 shows significant improvement of the results using current dataset.

Table 3.5: Results of the Experiment II.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	88.83%	87.83%	88.48%	88.16%	88.89%
LSVC	91.15%	91.48%	89.86%	90.66%	91.13%
Decision Tree	97.71%	97.81%	97.33%	97.57%	97.70%

3.4 Experiment III (*retweet_count* to *follower_count* ratio)

Another way to introduce the number of followers can be dividing uncorrelated feature such as *retweet_count* by *follower_count* feature. As before, it is required to add one to the divider to avoid division by zero. Moreover, feature *retweet_count* should be dropped.

3.4.1 Evaluation

This experiment's results table 3.6 show similar behavior as in the previous experiments with the Decision Tree classifier, but Logistic Regression and LSVC accuracies don't seem to be that critical. There still is a possibility that the result's improvement only refers to the acquired dataset, however comparing with the Experiment I Decision Tree results improvement is more smoothed.

Table 3.6: Results of the Experiment III.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	84.46%	81.64%	85.39%	83.48%	84.59%
LSVC	84.74%	82.35%	85.36%	83.83%	84.85%
Decision Tree	96.97%	96.92%	96.73%	96.83%	96.97%

3.5 Experiment IV (DateTime features)

In this experiment the DateTime features were explored. In particular new feature *time_diff* was created. This feature shows the number of days from the account creation to the creation of the Tweet. This way fake accounts can be detected because Twitter tries to delete those accounts, therefore fake news providers tend to create new accounts regularly.

3.5.1 Evaluation

Table 3.7 shows almost 100% results. The current NFN dataset provides only news from big and mature news providers, thus the experiment should be explored further with a bigger and more diverse dataset.

Table 3.7: Results of the Experiment IV.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	99.97%	99.96%	99.98%	99.97%	99.97%
LSVC	99.95%	99.96%	99.94%	99.95%	99.95%
Decision Tree	99.95%	99.96%	99.94%	99.95%	99.95%

3.6 Experiment V (topic analysis with LDA)

In this experiment [LDA] implemented by sklearn [32] was used to help the prediction. Because LDA is a supervised model, it is required to use train and test sets, therefore it was decided to add LDA to the pipeline. The transformation by sklearn returns an array of shape [n_samples, n_components], where n_samples is the number of rows in the input DataFrame. The array contains probabilities for each row to be in each component(topic). In this work, this array is transformed to an array of shape [n_samples, 1] which contains a component number with the higher probability for each row. Then, the array is appended to the baseline experiment DataFrame as a new column.

The detailed scheme of the LDA model is presented in fig. 3.8, where "Additional preprocessing" implies format's conversions and merging datasets into one labeled DataFrame, "Transform results" implies transforming LDA results to a list of topics, and "Prediction model" implies applying classifiers that were discussed above.

Moreover, LDA requires the important parameter - number of components (topics). To choose the number of components the LDA model was used with the different number of components. The results are shown in fig. 3.3. In this case there are two approaches to decide the best result, one is to take into account the best result of all classifiers (green dashed line), another one is to look at the mean of the results (red dashed line). For the first approach the best result is acquired with 5 components, for the second one 35 components.

3.6.1 Evaluation

Because two approaches were used, both of them should be evaluated. The best classifier approach results are shown in Table 3.8, and the best mean approach results are shown in Table 3.9. Compared to the baseline experiment outcome (table 3.1), these results are slightly worse in general. This represents that the data does not have any distinct topic division. Either all of the Tweets are too similar, or too different.

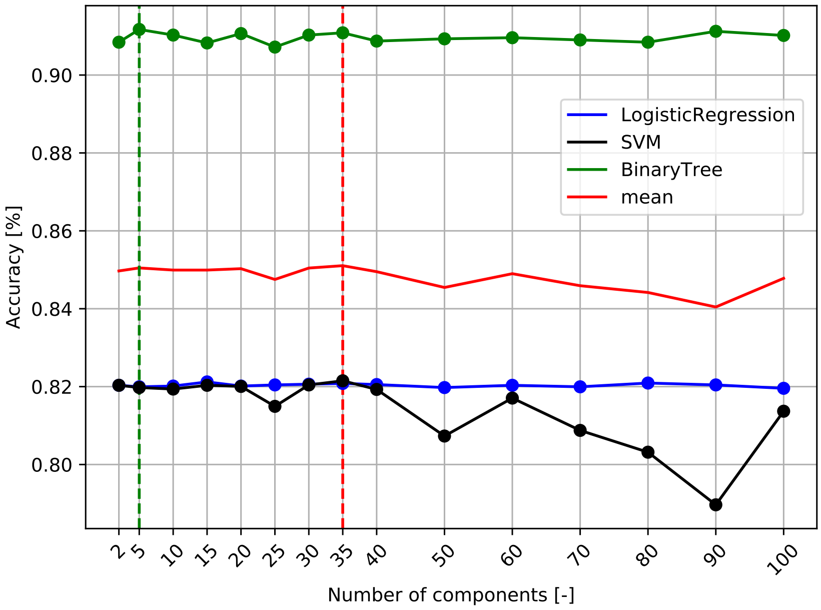


Figure 3.3: Result of the LDA model with different number of components. ”mean” is the mean result of all three classifiers.

Table 3.8: Results of the Experiment V with 35 components.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	81.88%	79.66%	81.80%	80.72%	82.00%
LSVC	82.12%	80.78%	81.41%	81.10%	82.19%
Decision Tree	91.10%	90.58%	90.65%	90.61%	91.12%

Table 3.9: Results of the Experiment V with 5 components.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	81.87%	79.60%	81.84%	80.70%	82.00%
LSVC	81.88%	80.19%	81.42%	80.80%	81.98%
Decision Tree	91.14%	90.51%	90.79%	90.65%	91.17%

It is possible to explore the LDA results a little further. Figure 3.4 agrees with the assumption that the Tweets are too similar between classes for LDA with 5 components because for each topic there is approximately the same number of Tweets. Moreover, looking at the top 10 words it is hard to guess how each topic could be named.

Concerning the LDA with 35 components (fig. 3.5), although most of the components have similar number Tweets per class, some of them show notice-

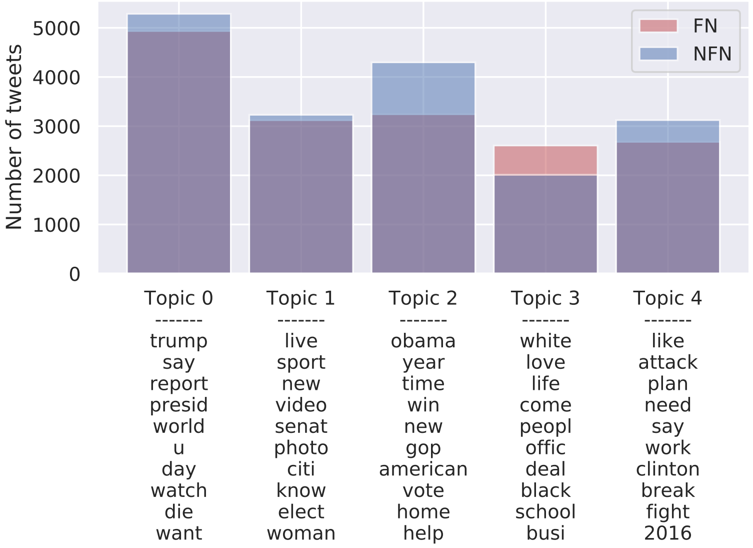


Figure 3.4: Number of Tweets of different classes and top 10 words for each topic using LDA with 5 components.

able difference, mostly ”0”, ”13” and ”25”. Top ten words of those three topics are shown in table 3.10. From this it can be concluded that NFN providers mostly write about Trump, elections and government, but FN providers are more likely to include propagandist phrases or mention children and celebrations. The detailed list of all 35 topics is included in the **Appendix**.

Table 3.10: Top words for topics ”0”, ”13” and ”25” using LDA with 35 components.

Topic Number	Top 10 words
0	want best countri caus studi paul art n child christma
13	presid trump donald use offici open govern court california money
25	trump clinton win polit republican health 2016 hillari big lead

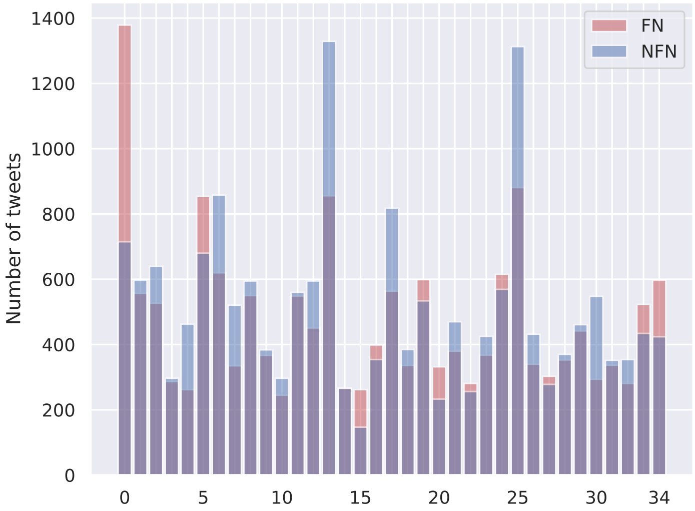


Figure 3.5: Number of Tweets of different classes for each topic using LDA with 35 components.

3.7 Experiment VI (extracting extra features from text)

This experiment is dedicated to explore various features that can be extracted from a Tweet's text. The following new features were proposed:

- Number of characters
- Number of words
- Boolean value that indicates if a text contains a question mark
- Boolean value that indicates if a text contains an exclamation mark
- Boolean value that indicates if a text contains multiple exclamation or question marks
- Number of upper case letters

3.7.1 Evaluation

As shown in Table 3.11, there is a significant improvement compared to the Baseline Experiment (table 3.1).

Table 3.11: Results of the Experiment VI.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	85.26%	84.41%	85.86%	85.13%	85.26%
LSVC	85.10%	84.50%	85.51%	85.01%	85.10%
Decision Tree	93.05%	93.32%	92.81%	93.07%	93.05%

3.8 Experiment VII (sentiment analysis)

This experiment explores the benefits of adding sentiment analysis of the text to the Baseline Experiment features. For the sentiment analysis it was decided to use the TextBlob [34] library. The main advantage of using TextBlob is that training dataset is not required. TextBlob's sentiment analysis provides a rational number between -1 and 1, where -1 is negative, 0 is neutral and 1 is positive.

3.8.1 Evaluation

Figure 3.6 shows the distribution of positive, neutral and negative Tweets among FN and NFN. Although it seems that polarity distribution is similar between classes, combined with other features it can provide some new results.

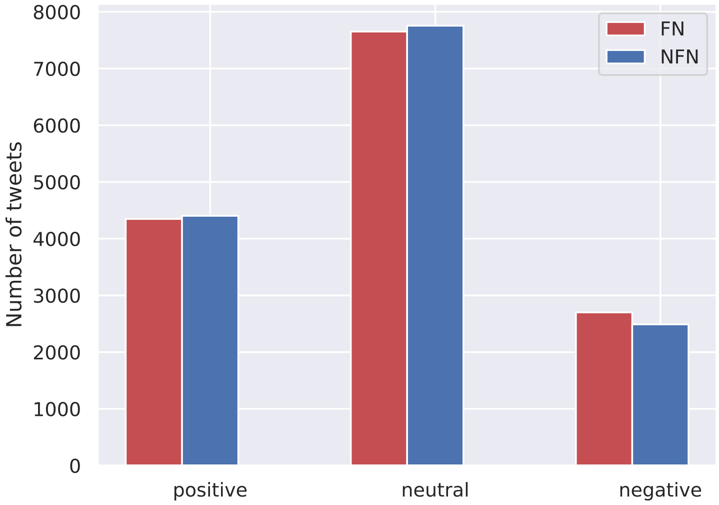


Figure 3.6: Distribution of positive, negative and neutral Tweets. For the visualization positive Tweets are considered Tweets with polarity more then 0, negative - less then 0, and neutral - equal to 0.

Table 3.12 shows that comparing with the Baseline Experiment (table 3.1) sentiment analysis provides a sufficient improvement. Unsurprisingly, many fake news detection techniques uses sentiment analysis.

Table 3.12: Results of the Experiment VII.

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	83.25%	81.78%	84.25%	82.99%	83.25%
LSVC	83.02%	81.91%	83.76%	82.83%	83.02%
Decision Tree	92.12%	92.50%	91.79%	92.15%	92.12%

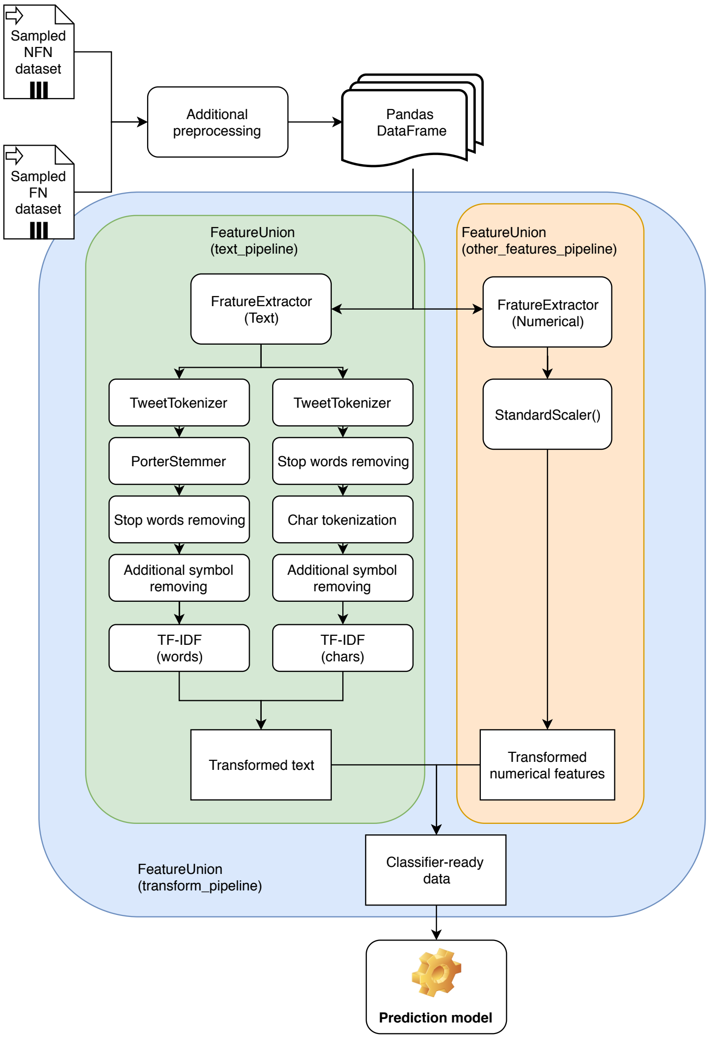


Figure 3.7: Baseline pipelines scheme.

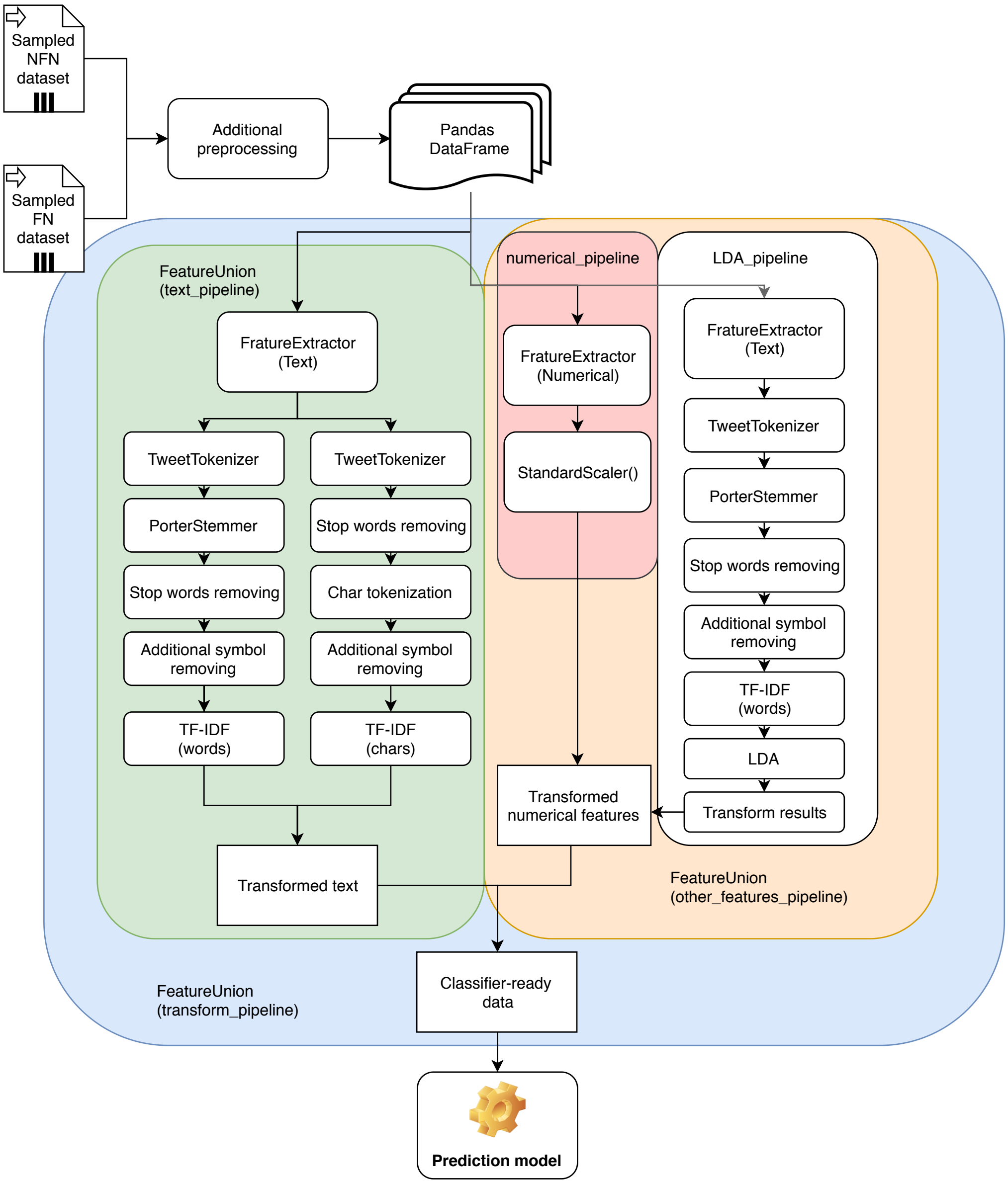


Figure 3.8: Pipelines scheme with LDA.

3.9 Experiments summary

Some of the experiments seem to rely on the provider's popularity more then on other features. Those should be applied with caution because they have a big probability of classifying truthful but small or young news provider as fake news.

Other experiments were not biased by the provider's popularity at all because they only use features extracted from text. Those can be used despite the NFN dataset diversity. Consequently, all the experiments can be divided into three groups:

Easily biased by a provider's popularity:
Experiment I (followers and following)
Experiment II (like and reply count)
Experiment IV (DateTime features)

A little biased by a provider's popularity:
Experiment III (*retweet_count* to *follower_count* ratio)

Not biased by a provider's popularity:
Experiment V (topic analysis with LDA)
Experiment VI (extracting extra features from text)
Experiment VII (sentiment analysis)

It was decided to put Experiment III into the separate group in the middle of easily biased and not biased because, even though Logistic Regression and LSVC results of the second experiment were not influenced that much, in general, this experiment seems to be not that biased comparing to others in the easily biased group.

The overall results are shown in the table 3.13. Among not biased group extracting extra features from text gives the most promising results. Concerning the biased group, it is hard to decide which result is the best because the best accuracy could mean higher bias. Undoubtedly, the most biased feature among all is the *time_diff*.

Table 3.13: Results summary (CLF - Classifier, LR - Logistic Regression, DT - Decision Tree).

CLF	Accuracy of each experiment in percentages								
	BS	I	II	IV	III	V_5	V_35	VI	VII
LR	82.39	82.53	88.89	99.97	84.59	82.00	82.00	85.26	83.25
LSVC	82.21	83.15	91.13	99.95	84.85	81.98	82.19	85.10	83.02
DT	91.11	99.15	97.70	99.95	96.97	91.17	91.12	93.05	92.12

Chapter 4

The Tool

The tool is designed to classify Fake News in a simple way which does not require programming. The aim of the tool is to explore the behavior of different combinations of the experiments with different FN and NFN datasets.

The tool has the following features:

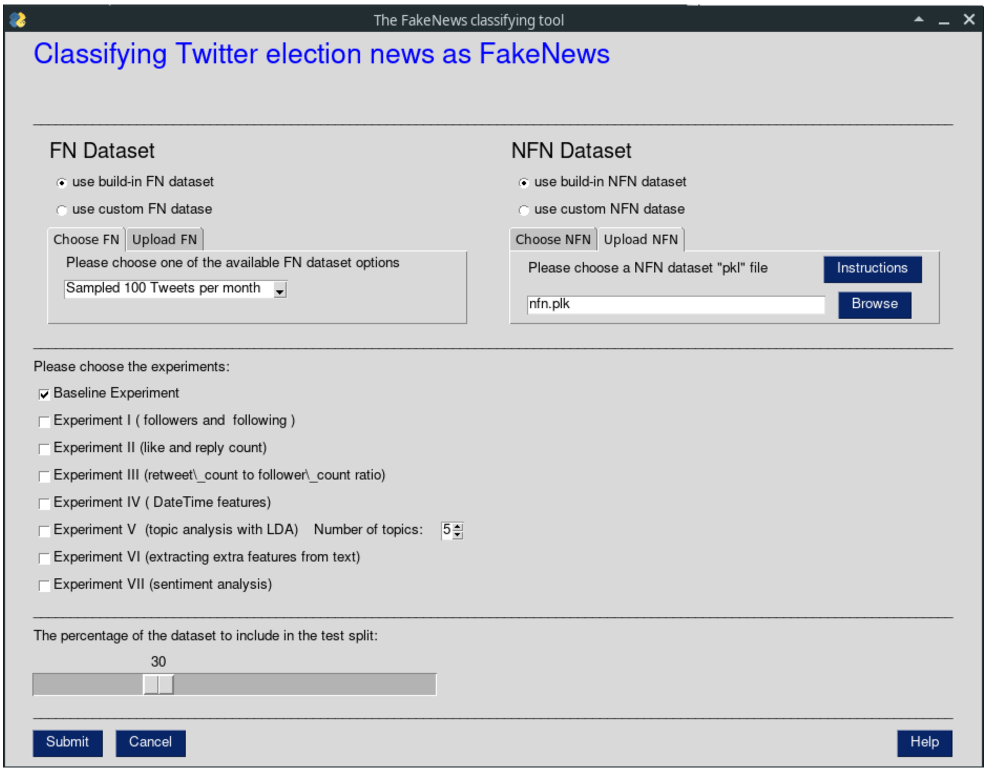
- Choosing one of the 18 preinstalled FN and NFN sampled datasets.
- Uploading custom FN or NFN datasets.
- Choosing any number on experiments to be performed on the selected datasets.
- Choosing a number of components for LDA model.
- Choosing the size of the test split.
- Classifying the datasets with the custom settings.

Also, there are restrictions, some of them are handled by the tool and some should be processed by the user. The list of the restrictions follows:

- Uploaded custom dataset should be in ”pkl” format, upon uploading any other format, user will get an error.
- The integrity and correctness of the uploaded custom dataset is the user responsibility.
- At least one experiment should be selected, otherwise an error will be shown.
- Number of LDA components should not be 0, otherwise an error will be shown.
- If number of LDA components is more then 200, user will get a warning.

4.1 Design and Implementation

The tool is implemented as a Python script. The graphical user interface (GUI) is implemented with a help of PySimpleGUI library[35]. Figure 4.1 shows the main user interface.



The FakeNews classifying tool

Classifying Twitter election news as FakeNews

FN Dataset

● use build-in FN dataset

○ use custom FN datase

Choose FN | Upload FN

Please choose one of the available FN dataset options

Sampled 100 Tweets per month

NFN Dataset

● use build-in NFN dataset

○ use custom NFN datase

Choose NFN | Upload NFN

Please choose a NFN dataset "pkl" file

nfn.plk

Please choose the experiments:

☑ Baseline Experiment

□ Experiment I (followers and following)

□ Experiment II (like and reply count)

□ Experiment III (retweet_count to follower_count ratio)

□ Experiment IV (DateTime features)

□ Experiment V (topic analysis with LDA) Number of topics: 5

□ Experiment VI (extracting extra features from text)

□ Experiment VII (sentiment analysis)

The percentage of the dataset to include in the test split:

30

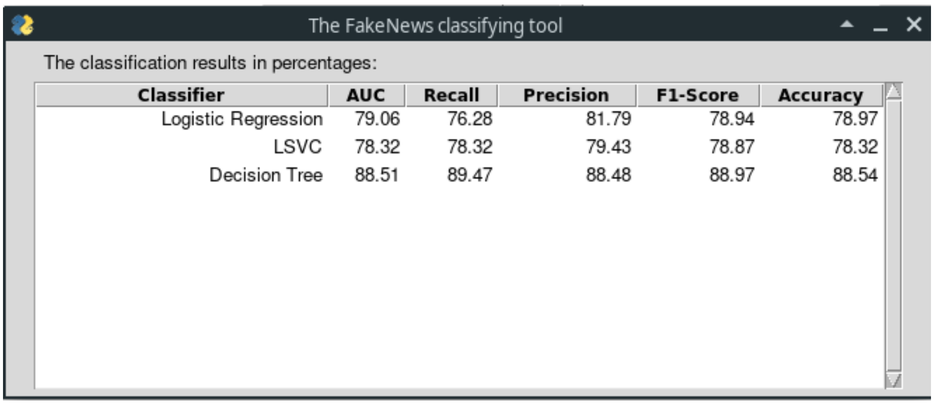
Figure 4.1: Main graphical user interface.

The structure of the program can be described as the main GUI loop that, when the ”Submit” button is pressed, runs a function that starts the algorithm of parsing the input, selecting the right features according to the selected experiment, and runs an appropriate classification pipeline. Feature vector for each experiment is described in the [Experiments] chapter.

There are two major types of pipelines, the Baseline pipeline (fig. 3.7) and the LDA pipeline (fig. 3.8). Moreover, modifications of the pipelines applied according to the selected combination of the experiments. For example, if only the Experiment VI is selected, not the whole Baseline pipeline is used, but only the ”text_pipeline” part.

To choose which FN or NFN dataset is to use, first, it is required to select the mode. Either ”use built-in” or ”upload custom” dataset. Then, the build-in dataset can be chosen via the drop down menu, or uploaded using ”Browse” button in the second tab. The ”Instructions” button is to help user with the format of the dataset to upload.

The classification results are displayed in the new window with the results table as shown in fig. 4.2.



The classification results in percentages:

Classifier	AUC	Recall	Precision	F1-Score	Accuracy
Logistic Regression	79.06	76.28	81.79	78.94	78.97
LSVC	78.32	78.32	79.43	78.87	78.32
Decision Tree	88.51	89.47	88.48	88.97	88.54

Figure 4.2: Results window interface.

Conclusion

The objectives of the work were met in the following ways:

1. The Twitter FN dataset was analyzed in the [**Dataset Analysis**] chapter.

2. Then, GetOldTweets tool [29] was chosen to obtain the NFN dataset.

3. In the same chapter a strategy of selecting Not Fake News was introduced.

4. The NLP methods overview was done in **State-of-the-Art** and explored in chapter **Experiments and evaluations**.

5. The preprocessing was described in the **Experiments and evaluations**.

6. The tool was presented in chapter **The Tool**.

7. And finally, the analysis of the sources was accomplished as a Summary section of the **Experiments and evaluations** chapter.

The main obstacle that was met in this work is the lack of a professionally build NFN dataset that will be diverse enough, big enough and still reliable. On the other hand, it emphasizes the importance of such dataset and the quality data in general. Moreover, this work is another example that good correlation does not always mean good results in total.

Overall, even if no best way of detecting Fake News was selected, the results and more importantly, the resulted tool can help further researchers to finally implement an ultimate news evaluator that will help people to be sure that news from social media can actually be trusted.

For future work, the tool can be improved to use other than Twitter types of news. Another improvement could be to explore URLs and more specifically try to find out what URLs are masked with the shortening services. Also, more advanced algorithms can be implemented, such as word2vec preprocessing or Neural Networks. Finally, more training datasets of higher quality can be constructed.

Bibliography

[1] Habert, B.; Adda, G.; et al. Towards tokenization evaluation. volume 98, 1998: pp. 427–431.

[2] Jivani, A. G.; et al. A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, volume 2, no. 6, 2011: pp. 1930–1938.

[3] Toman, M.; Tesar, R.; et al. Influence of word normalization on text classification. *Proceedings of InSciT*, volume 4, 2006: pp. 354–358.

[4] Manning, C.; Raghavan, P.; et al. Introduction to information retrieval. *Natural Language Engineering*, volume 16, no. 1, 2010: pp. 100–103.

[5] Hosmer Jr, D. W.; Lemeshow, S.; et al. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[6] Joachims, T. Text categorization with support vector machines: Learning with many relevant features. 1998: pp. 137–142.

[7] Breiman, L. *Classification and regression trees.* Routledge, 2017.

[8] Blei, D. M.; Ng, A. Y.; et al. Latent dirichlet allocation. *Journal of machine Learning research*, volume 3, no. Jan, 2003: pp. 993–1022.

[9] Ahmed, H.; Traore, I.; et al. Detecting opinion spams and fake news using text classification. *Security and Privacy*, volume 1, no. 1, 2018: p. e9.

[10] Hale, J. Uncertainty About the Rest of the Sentence. *Cognitive Science*, volume 30, no. 4, 8 2006: pp. 643–672, ISSN 1551-6709, doi:10.1207/s15516709cog0000_64. Available from: `https://doi.org/10.1207/s15516709cog0000_64`

[11] Gilda, S. Evaluating machine learning algorithms for fake news detection. In *2017 IEEE 15th Student Conference on Research and Development (SCOReD)*, IEEE, 2017, pp. 110–115.

[12] Dashtipour, K.; Poria, S.; et al. Multilingual sentiment analysis: state of the art and independent comparison of techniques. *Cognitive computation*, volume 8, no. 4, 2016: pp. 757–771.

[13] Cambria, E.; Speer, R.; et al. Senticnet: A publicly available semantic resource for opinion mining. 2010.

[14] Singh, V.; Piryani, R.; et al. Sentiment analysis of textual reviews; Evaluating machine learning, unsupervised and SentiWordNet approaches. 2013: pp. 122–127.

[15] Shi, H.-X.; Li, X.-J. A sentiment analysis model for hotel reviews based on supervised learning. volume 3, 2011: pp. 950–954.

[16] Boiy, E.; Moens, M.-F. A machine learning approach to sentiment analysis in multilingual Web texts. *Information retrieval*, volume 12, no. 5, 2009: pp. 526–558.

[17] Mizumoto, K.; Yanagimoto, H.; et al. Sentiment analysis of stock market news with semi-supervised learning. 2012: pp. 325–328.

[18] Dean Pomerleau, D. R. The Fake News Challenge. 2017. Available from: `http://www.fakenewschallenge.org/`

[19] Sean Baird, Y. P., Doug Sibley. Talos Targets Disinformation with Fake News Challenge Victory. 2017. Available from: `https://blog.talosintelligence.com/2017/06/talos-fake-news-challenge.html`

[20] Ruchansky, N.; Seo, S.; et al. Csi: A hybrid deep model for fake news detection. 2017: pp. 797–806.

[21] Kwon, S.; Cha, M.; et al. Prominent features of rumor propagation in online social media. 2013: pp. 1103–1108.

[22] Gupta, A.; Lamba, H.; et al. Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of the 22nd international conference on World Wide Web*, ACM, 2013, pp. 729–736.

[23] Castillo, C.; Mendoza, M.; et al. Information credibility on twitter. 2011: pp. 675–684.

[24] Fake News Detector. `https://fakenewsdetector.org/`, accessed: 2019-05-03.

[25] FakerFact. `https://www.fakerfact.org/`, accessed: 2019-05-03.

[26] Twitter, I. Twitter fake news dataset. 2019. Available from: `https://about.twitter.com/en_us/values/elections-integrity.html#data`

[27] MacCarthy, R. The Average Twitter User Now has 707 Followers. 2016. Available from: https://kickfactory.com/blog/average-twitter-followers-updated-2016/

[28] Glader, P. 10 Journalism Brands Where You Find Real Facts Rather Than Alternative Facts. 2017. Available from: www.forbes.com/sites/berlinschoolofcreativeleadership/2017/02/01/10-journalism-brands-where-you-will-find-real-facts-rather-than-alternative-facts/#1c47fbc9e9b5

[29] Henrique, J. GetOldTweets-python. https://github.com/Jefferson-Henrique/GetOldTweets-python, 2016.

[30] Varis, E. Tokenizer. https://github.com/erikavaris/tokenizer, 2017.

[31] Loper, E.; Bird, S. NLTK: The Natural Language Toolkit. 2002.

[32] Pedregosa, F.; Varoquaux, G.; et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, volume 12, 2011: pp. 2825–2830.

[33] Danilak, M. Language detection library ported from Google's language-detection. 2016. Available from: https://pypi.org/project/langdetect/

[34] Loria, S.; Keen, P.; et al. TextBlob: simplified text processing; 2018. 2013.

[35] B., M. PySimpleGUI. 2018. Available from: https://github.com/PySimpleGUI

[36] Twitter, I. Twitter readme. 2019. Available from: https://storage.googleapis.com/twitter-election-integrity/hashed/Twitter_Elections_Integrity_Datasets_hashed_README.txt

Appendix A

Acronyms

FN Fake News

NFN Not Fake News

TF-IDF Term-frequency times inverse document-frequency

LDA Latent Dirichlet allocation

BS Baseline

GUI Graphical user interface

Appendix B

FakeNews dataset description

Table B.1: Table of features. (*) - at the time of suspension (ˆ) - these engagement counts exclude engagements from users who are suspended, deleted or otherwise actioned against by Twitter at the time of this data release. (°) Described in Chapter: [**Theoretical approaches**]

Begin of Table		
Feature name	**Feature description**	**Preprocessing description**
tweetid	tweet identification number	**Attribute initial type:** float64 **Percentage of NaN's:** 0 **Required preprocessing:** Not used because it is not bringing any information to the classification.
userid	user identification number (anonymized for users which had fewer than 5,000 followers at the time of suspension)	**Attribute initial type:** object **Percentage of NaN's:** 0 **Required preprocessing:** Not used because it is not bringing any information to the classification and hashed.
user_display_name	the name of the user (same as userid for anonymized users)	**Attribute initial type:** object **Percentage of NaN's:** 0 **Required preprocessing:** Not used because it is not bringing any information to the classification and hashed.
user_screen_name	the Twitter handle of the user (same as userid for anonymized users)	**Attribute initial type:** object **Percentage of NaN's:** 0 **Required preprocessing:** Not used because it is not bringing any information to the classification and hashed.
user_reported_location	the user's self-reported location (*)	**Attribute initial type:** object **Percentage of NaN's:** 10% **Required preprocessing:** Not used because on the inability to collect this information for the NFN dataset

Continuation of Table B.1		
Feature name	Feature description (taken from Twitter dataset readme[36])	Preprocessing description
user_profile_description	the user's profile description (*)	**Attribute initial type:** object **Percentage of NaN's:** 14% **Required preprocessing:** Described in the Experiments and evaluations chapter.
user_profile_url	the user's profile URL (*)	**Attribute initial type:** object **Percentage of NaN's:** 79% **Required preprocessing:** Not used as useless because of the NaN's percentage
follower_count	the number of accounts following the user (*)	**Attribute initial type:** float64 **Percentage of NaN's:** 0 **Required preprocessing:** The type was changed to int.
following_count	the number of accounts followed by the user (*)	**Attribute initial type:** float64 **Percentage of NaN's:** 0 **Required preprocessing:** The type was changed to int.
account_creation_date	date of user account creation	**Attribute initial type:** object (date format yyyy-mm-dd) **Percentage of NaN's:** 0 **Required preprocessing:** The date format was changed to Datetime.
account_language	the language of the account, as chosen by the user	**Attribute initial type:** object **Percentage of NaN's:** 0 **Required preprocessing:** Not used because can be different from tweet_language.
tweet_language	the language of the tweet	**Attribute initial type:** object **Percentage of NaN's:** 0 **Required preprocessing:** Used for preprocessing, then deleted because only 'en' language tweets are used.
tweet_text	the text of the tweet (mentions of anonymized accounts have been replaced with anonymized userid)	**Attribute initial type:** object **Percentage of NaN's:** 0 **Required preprocessing:** The preprocessing of the Tweet's text is described in the Experiments and evaluations chapter.
tweet_time	the time when the tweet was published (UTC)	**Attribute initial type:** object (date format yyyy-mm-dd hh:mm) **Percentage of NaN's:** 0 **Required preprocessing:** The date format was changed to Datetime.
tweet_client_name	the name of the client app used to publish the tweet	**Attribute initial type:** object **Percentage of NaN's:** 0.01% **Required preprocessing:** Not used because on the inability to collect this information for the NFN dataset.

Continuation of Table B.1

Feature name	Feature description (taken from Twitter dataset readme[36])	Preprocessing description
in_reply_to_tweetid	the tweetid of the original tweet that this tweet is in reply to (for replies only)	**Attribute initial type:** float64 **Percentage of NaN's:** 97% **Required preprocessing:** Not used because only the original tweets was chosen.
in_reply_to_userid	the userid of the original tweet that this tweet is in reply to (for replies only)	**Attribute initial type:** object **Percentage of NaN's:** 95% **Required preprocessing:** Not used because only the original tweets was chosen.
quoted_tweet_tweetid	the tweetid of the original tweet that this tweet is quoting (for quotes only)	**Attribute initial type:** float64 **Percentage of NaN's:** 99% **Required preprocessing:** Not used because of the NaN's percentage.
is_retweet	True/False, is this tweet a retweet	**Attribute initial type:** bool **Percentage of NaN's:** 0 **Required preprocessing:** Used for preprocessing, then deleted because only the original tweets was chosen. Not used because only the original tweets was chosen.
retweet_userid	for retweets, the userid who authored the original tweet	**Attribute initial type:** float64 **Percentage of NaN's:** 0 **Required preprocessing:** Not used because only the original tweets was chosen.
retweet_tweetid	for retweets, the tweetid of the original tweet	**Attribute initial type:** float64 **Percentage of NaN's:** 0 **Required preprocessing:** Not used because only the original tweets was chosen.
latitude	geo-located latitude, if available	**Attribute initial type:** float64 **Percentage of NaN's:** 99.9% **Required preprocessing:** Not used as useless because of the NaN's percentage.
longitude	geo-located longitude, if available	**Attribute initial type:** float64 **Percentage of NaN's:** 99.9% **Required preprocessing:** Not used as useless because of the NaN's percentage.
quote_count	the number of tweets quoting this tweet	**Attribute initial type:** float64 **Percentage of NaN's:** 0 **Required preprocessing:** Not used because on the inability to collect this information for the NFN dataset.
reply_count	the number of tweets replying to this tweet	**Attribute initial type:** float64 **Percentage of NaN's:** 0 **Required preprocessing:** Type was changed to int.
like_count	the number of likes that this tweet received (ˆ)	**Attribute initial type:** object **Percentage of NaN's:** 0 **Required preprocessing:** Type was changed to int.

Continuation of Table B.1

Feature name	Feature description (taken from Twitter dataset readme[36])	Preprocessing description
retweet_count	the number of retweets that this tweet received (ˆ)	**Attribute initial type:** float64 **Percentage of NaN's:** 0 **Required preprocessing:** Type was changed to int.
hashtags	a list of hashtags used in this tweet	**Attribute initial type:** object **Percentage of NaN's:** 13% is NaN and 38% is empty brackets ”[]”. Format was ”[hashtag1, hashtag2, ...]” **Required preprocessing:** ”[]” was replaced with NaN, format was changed to ”hashtag1, hashtag2, ...”
urls	a list of urls used in this tweet	**Attribute initial type:** object **Percentage of NaN's:** 18% is NaN and 46% is empty brackets '[]' **Required preprocessing:** ”[]” was replaced with NaN, format was changed to ”url1, url2, ...”
user_mentions	a list of userids who are mentioned in this tweet (includes anonymized userids)	**Attribute initial type:** object **Percentage of NaN's:** 89% **Required preprocessing:** ”[]” was replaced with NaN, format was changed to ”mention1, mention2, ...”
poll_choices	if a tweet included a poll, this field displays the poll choices separated by |	**Attribute initial type:** object **Percentage of NaN's:** 99.9% **Required preprocessing:** The feature was deleted with all the items where this feature is not null.

End of Table B.1

Appendix C

LDA with 35 components topic analysis

Table C.1: Top words for each topic using LDA with 35 components.

Topic Number	Top 10 words
0	want best countri caus studi paul art n child christma
1	right happen save compani twitter judg free order hear hour
2	peopl photo miss 10 care debat star 2015 month young
3	time fall 4 tv problem stand futur administr past angel
4	ap state meet week china start head chief continu number
5	man kill polic talk arrest shoot offic protest north suspect
6	new hous look leader campaign gop poll set john read
7	russia way 2017 speak brexit justic ahead travel c area
8	watch sport accus test water sign servic team expect dog
9	american work come game war bank florida bad fear son
10	plan good charg speech congress colorado word hand light suicid
11	home local near question person victim wall cop insid risk
12	say video stop said tech mean build climat act internet
13	presid trump donald use offici open govern court california money
14	need know america tri final let stay britain blame god
15	love becom teen beat place alway 7 depart heart fact
16	world rt billion news food reach hack saudi rape deni
17	report obama run isi race hope rise record media lose
18	shot 1 russian bomb driver trade england fed hurrican parent
19	famili end case got texa south deal korea search high
20	black cut market children pre air workout stock w flag
21	elect fight million tax ban join obamacar group sander san
22	day pictur injur bring away step hurt march movi william
23	live make help tell secur feel major challeng wrong border
24	woman dead 3 releas immigr review anoth histori gener issu
25	trump clinton win polit republican health 2016 hillari big lead
26	break latest share law pay kid rais hold target respons
27	citi girl return sinc night updat babi everi fbi pari
28	rule today follow park david price propos thank ceo honor
29	death chang 2 parti power forc realli play unit men
30	u die face hit student uk colleg key cup link
31	white senat nation school crash investig fund counti hospit seek
32	year car stori job better buy alleg crisi old list
33	attack vote democrat busi support thing women leav claim chicago
34	like life think great mani turn gun drug real control

Appendix D

Contents of enclosed CD

- `TheTool` .... the directory with the Tool
  - `README.txt` .... the README file for the program
  - `requirements.txt` .... the requirements for the program
  - `Data` .... the directory of integrated datasets
  - `TheTool` .... the program sources
- `Thesis` .... the thesis text directory
  - `src` .... the directory of LaTeX source codes of the thesis
  - `BT_Vigriyanova.pdf` .... the thesis text in PDF format