DETECTION OF ORGANIZED BEHAVIOURS ON TWITTER

by

Erdem Beğenilmiş

B.S., Computer Engineering, Ege University, 2011

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2017

DETECTION OF ORGANIZED BEHAVIOURS ON TWITTER

APPROVED BY:

Prof. Tunga Güngör (Thesis Supervisor)	.....
Suzan Üsküdarlı,Ph.D. (Thesis Co-supervisor)	.....
Assoc. Prof. Haluk Bingöl	.....
Assist. Prof. Murat Can Ganiz	.....

DATE OF APPROVAL: 26.05.2017

ACKNOWLEDGEMENTS

I would first like to thank my thesis advisor, Ph.D. Suzan Üsküdarlı for all kinds of valuable assistance that she provided to me. During this study, anytime I stumbled, her assistance and support helped me. I am in gratitude beyond words.

Also, I am deeply grateful to Prof. Tunga Güngör and SosLab members Ahmet Yıldırım and Onur Güngör for their continuous help during my thesis.

Finally, I must express my very profound gratitude to my parents and to my wife Gamze for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without her. If there is anyone to dedicate this thesis to, it would be her.

ABSTRACT

DETECTION OF ORGANIZED BEHAVIOURS ON TWITTER

Microblogging platforms are widely used to share information, feelings, and ideas about anything. With nearly 320 million users (as of April 2017) Twitter is one of the most popular microblogging platforms making it a lucrative platform for propagating (mis)information through organized activities. Such cases have been observed during election campaigns (2016 United States), disasters (2010 Haiti earthquake), and resistance movements (2011 Occupy Wall Street, 2011 Arab Spring). As a result of this, there exists an increased use of social media to recruit people to illegal organizations and to spread fake news. Recruited users utilize various Twitter entities like hashtags, mentions, URLs to organize and coordinate their efforts towards a specific goal. Besides from recruited users, fake accounts and bots are also frequently used in Twitter. In such cases, users can be manipulated, since users assume that tweets are posted with the free will of individuals without intent of collusion.

This thesis proposes a supervised classification model for distinguishing tweet sets that are "organized" and "organic". A prototype implementation of this model is implemented and experiments with a large tweet sets are conducted. During study, numerous features associated with tweets and posting behavior were examined to identify those that are appropriate for training the model. Analyzed tweets were collected by querying hashtags, since hashtags serve to group tweets. The training data set, which has a size of 1000 records with 299 features, is used as a result of analyzing more than 200 million tweets. Among the applied supervised learning algorithms, Random Forest gave the best results in all data sets with f-measure and accuracy of 0.98.

ÖZET

ORGANİZE DAVRANIŞLARIN TWITTER'DA BELİRLENMESİ

Günümüzde mikroblog platformları bilgi, düşünce, duygu paylaşımı dahil bir çok konuda kullanılmaktadır. Bu platformların içerisinde, 320 milyon kullanıcısı (Nisan 2017) ile Twitter başı çekmektedir. Twitter'ın mikroblog platformları arasındaki bu durumu, onu organize aktivitelerin kullanılması ve bu yollarla kullanıcılara doğru olmayan bilgilerin iletilmesi konusunda cazip kılmakta. Twitter'ın günümüzdeki kullanımına bakıldığında, organize davranışların seçim kampanyaları (Amerika Birleşik Devletleri 2016 Başkanlık Seçimi), doğal afetler (2010 Haiti Depremi), direniş hareketleri (2011 Arap Baharı) gibi bir çok konularda karşımıza çıktığını görmekteyiz. Bunlara ek olarak, son zamanlarda Twitter'da Sahte Haberler yayması için kullanıcıların istihdam edildiklerini, ve otomatik botların kullandıklarını görmekteyiz. Bu durum, sosyal medya kullanıcıları arasındaki güven ilişkisini zedeleyebilmekte, ve toplumun doğru olmayan haberler doğrultusunda manipülatif yönlendirimine olanak vermektedir. Bu bilgiler ışığında, Twitter'daki tweet atma davranışlarını "organize" ve "organik" olarak iki ayrı kategoride değerlendirilebilmekte.

Bu tez, bir tweet kümesindeki davranışların "organize" veya "organik" olduğunu anlayabilen, otomatik olarak kullanılabilecek bir sınıflandırma modeli sunmaktadır. Yapılan çalışmalarda 200 milyon tweet analiz edilmiştir. İlgili tweet set'lerin hepsi ingilizcedir ve çoğunluğu Amerika Birleşik Devletleri kullanıcıları tarafından atılmıştır. Model'i üretmek için kullanılan eğitim seti her biri 299 parametreden oluşan toplam da 800 kaydın olduğu bir eğitim setinden üretilmiştir. Kullanılan algoritmalar arasında, en iyi sonucu "Random Forest" algoritması %98 f-skoru ve doğrulukla vermiştir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . iv
ÖZET . . . v
LIST OF FIGURES . . . x
LIST OF TABLES . . . xiv
LIST OF ACRONYMS/ABBREVIATIONS . . . xvi
1. INTRODUCTION . . . 1
2. RELATED WORK . . . 7
2.1. Comparison of Studies With This Thesis . . . 11
3. BACKGROUND . . . 13
3.1. Twitter . . . 13
3.1.1. Spam . . . 15
3.1.2. Bot and Cyborg Accounts . . . 15
3.1.3. Twitter REST API . . . 16
3.2. Social Network Analysis . . . 16
3.2.1. Centrality . . . 17
3.2.1.1. Degree Centrality . . . 17
3.2.1.2. Closeness Centrality . . . 17
3.3. Natural Language Processing . . . 18
3.3.1. Sentiment Analysis . . . 18
3.3.2. Text Similarity . . . 18
3.4. Classification . . . 18
3.4.1. Resampling . . . 19
3.4.1.1. Ten Fold Cross Validation . . . 20
3.4.1.2. Simple Random Sampling . . . 20
3.4.1.3. Bootstrapping . . . 20
3.4.2. Classification Methods . . . 21
3.4.3. Principal Component Analysis . . . 22

3.4.4. Tools . . . 22
3.4.4.1. Weka . . . 23
3.4.4.2. Apache Spark . . . 23
3.5. NoSQL Databases . . . 23
3.5.1. Document Based NoSQL Databases . . . 23
3.5.2. NoSQL Graph Databases . . . 24
4. APPROACH . . . 25
4.1. Organized Behavior Use Cases . . . 27
4.1.1. Making a Hashtag Top Trend . . . 27
4.1.2. Spam Attacks . . . 28
4.1.3. Charity Behaviors . . . 29
4.1.4. Humiliation . . . 29
4.1.5. Flattery . . . 30
4.1.6. Propaganda . . . 30
4.1.7. Fake News . . . 30
4.2. General Features and Patterns that Can Be The Sign of Organized Behavior . . . 31
4.2.1. Twitter Account Opening Date . . . 31
4.2.2. Closeness Centrality of Users in the Friend-Follower Network . . . 31
4.2.3. Relation Between Communities . . . 31
4.2.4. Tweeting Behaviors – Tweet History . . . 32
4.2.4.1. Retweeting Same Tweets . . . 32
4.2.4.2. Information Flow During The Propagation of Retweets 32
4.2.4.3. Tweets About Same / Similar Topics in Same Time of the Day . . . 32
4.2.5. Checking Community Users : Are There Bot or Cyborg Users ? 33
4.3. Which Features Can Be Used In Which Use Cases ? . . . 33
4.3.1. Hypothesis I - Top Trend Hashtags & Propaganda & Flattery & Fake News . . . 33
4.3.2. Hypothesis II : Spam Attacks & Humiliation & Fake News . . . 34
4.3.3. Hypothesis III : Charity Behaviors . . . 34

5. MODEL 35
5.1. Campaign Creation 35
5.2. Feature Extraction 40
5.3. User Feature Extraction 42
5.4. Hourly Feature Extraction 42
5.5. Classification 50
5.6. Features 51
5.6.1. User Features 51
5.6.1.1. Extracted User Features 52
5.6.2. Extracted Hourly Features 54
5.6.2.1. General Hourly Features 54
5.6.2.2. Hourly Tweet Entity Features 54
5.6.2.3. Hourly Retweet Features 55
5.6.2.4. Hourly Non-Retweet Features 55
5.6.2.5. Hourly Mention Features 56
5.6.2.6. Hourly Retweeted Mention Features 56
5.6.2.7. Hourly Non - Retweeted Mention Features 57
5.6.2.8. Hourly Sentiment Features of Reply Tweets 57
6. PROTOTYPE IMPLEMENTATION 59
6.1. Twitter Connection Pool Implementation 59
6.2. Campaign Creation Module 60
6.2.1. Front End 61
6.2.2. Tweet Retrieval using Twitter Search API 61
6.3. Feature Extraction Module 62
6.3.1. Tweet Retrieval 63
6.3.1.1. Retrieval of Users' Tweets 64
6.3.2. Implemented User Interfaces 67
6.3.3. Breakpoints 67
6.3.3.1. TWEET_COLLECTION_COMPLETED 68
6.3.3.2. USER_ANALYZE_COMPLETED 68
6.3.3.3. COS_SIMILARITY_INIT 68

6.3.3.4. TF_CALCULATION_COMPLETED 68
6.3.3.5. IDF_CALCULATION_COMPLETED 68
6.3.3.6. TF_IDF_CALCULATION_COMPLETED 68
6.3.3.7. SIMILARTY_CALCULATED 68
6.3.3.8. ALL_HOURLY_CALCULATION_DONE 69
6.3.3.9. GENERAL_STATISTIC_CALCULATION_DONE . . . 69
6.3.3.10. STATISTIC_CALCULATED 69
6.3.4. Campaign Analysis 69
6.3.5. Performance 79
6.3.5.1. Parallelism 79
6.3.5.2. Intermediary Tables 79
6.3.6. Training Data Creation & Classification 81
6.3.6.1. Training Data Extractor Module 81
6.3.6.2. Model Generator 83
6.3.7. Features 83
7. EXPERIMENTS AND RESULTS 110
7.1. Labeling Campaigns 110
7.1.1. Labeling Organized Records 110
7.1.2. Labeling Organic Records 111
7.1.3. Same User Analysis 111
7.1.4. Training Data Set Preparation 113
7.2. Results 114
7.3. Effective Features 117
7.3.1. Data Set 1 Features 117
7.3.2. Data Set 3 Features 118
7.4. Unsupervised Approach 118
8. DISCUSSION AND FUTURE WORK 120
9. CONCLUSION 122
REFERENCES 123

LIST OF FIGURES

Figure 1.1. White Nationalist follower growth [1]. 3

Figure 4.1. Example of the synchronization in organized behavior. 25

Figure 4.2. Example of the adoption of shared messages. 26

Figure 5.1. General Model. 36

Figure 5.2. Model Creation. 37

Figure 5.3. Campaign Hashtag ER diagram. 37

Figure 5.4. Campaign creation general flow. 40

Figure 5.5. Campaign Creation Algorithm. 41

Figure 5.6. Campaign Statistics Calculation Algorithm 41

Figure 5.7. Example of extracted features. 41

Figure 5.8. General Feature Extraction Algorithm. 46

Figure 5.9. User Features Extraction Algorithm. 49

Figure 5.10. Hourly Features Extraction Algorithm. 51

Figure 6.1. System Architecture. 59

Figure 6.2. Campaign Creation Interface. 63

Figure 6.3. User interface for displaying all campaigns. 64

Figure 6.4. User interface for displaying percentages of hashtag in campaign. . 70

Figure 6.5. User interface for displaying percentages of campaign words. . . . 71

Figure 6.6. Example Twitter Search API request. 71

Figure 6.7. Example campaign creation request. 72

Figure 6.8. Campaign statistic DB object. 72

Figure 6.9. Campaign analysis request interface. 73

Figure 6.10. Interface for listing all campaign analysis. 74

Figure 6.11. Interface for displaying extracted user features during campaign analysis. 74

Figure 6.12. Campaign hashtag analysis result interface for labor day, thanksgiving, and election related tweets. 82

Figure 6.13. Percentage of #thanksgiving users based on their tweet counts. . . 84

Figure 6.14. Percentage of #draintheswamp users based on their tweet counts. 85

Figure 6.15. Percentage of #thanksgiving users based on daily tweet counts. . . 86

Figure 6.16. Percentage of #draintheswamp users based on daily tweet counts. 87

Figure 6.17. Hourly entity ratios of #thanksgiving tweets. 88

Figure 6.18. Hourly entity ratios of #draintheswamp tweets. 89

Figure 6.19. Hourly retweet ratios of #thanksgiving tweets. 90

Figure 6.20. Hourly retweet ratios of #draintheswamp tweets. 91

Figure 6.21. Count of hourly #thanksgiving tweets and distinct users, who post tweets. 92

Figure 6.22. Count of hourly #draintheswamp tweets and distinct users, who post tweets. 93

Figure 6.23. Count of hourly #thanksgiving retweets and distinct users, who post tweets. 94

Figure 6.24. Count of hourly #draintheswamp retweets and distinct users, who post tweets. 95

Figure 6.25. Count of hourly #thanksgiving non-retweets and distinct users, who post tweets. 96

Figure 6.26. Count of hourly #draintheswamp non-retweets and distinct users, who post tweets. 97

Figure 6.27. Count of hourly total and distinct mentions for #thanksgiving. . . 98

Figure 6.28. Count of hourly total and distinct mentions in #draintheswamp tweets. 99

Figure 6.29. Count of hourly total and distinct mentions in #thanksgiving set of retweets. 100

Figure 6.30. Count of hourly total and distinct mentions in #draintheswamp set of retweets. 101

Figure 6.31. Count of hourly total and distinct mentions in #thanksgiving non-retweets. 102

Figure 6.32. Count of hourly total and distinct mentions in #draintheswamp non-retweets. 103

Figure 6.33. Entity ratios of #thanksgiving users. 104

Figure 6.34. Entity ratios of #draintheswamp users. 105

Figure 6.35. Friend follower ratios of #thanksgiving users. 106

Figure 6.36. Friend follower ratios of #draintheswamp users. 107

Figure 6.37. Tweet and favorited tweet counts of #thanksgiving users. 108

Figure 6.38. Tweet and favorited tweet counts of #draintheswamp users. . . . 109

Figure 7.1. Unsupervised clustering of the Training Data. 119

LIST OF TABLES

Table 2.1. Classification features used in political abuse detection 8

Table 2.2. Classification features used in detection of spam URLs 10

Table 5.1. Parameters of campaign creation. 38

Table 5.2. Description of campaign creation functions. 38

Table 5.3. Description of campaign statistic calculation functions. 39

Table 5.4. Feature extraction parameters. 43

Table 5.5. Description of functions in General Feature Extraction Algorithm. 44

Table 5.6. Description of functions in General Feature Extraction Algorithm (cont). 45

Table 5.7. Description of functions in User Features Extraction Algorithm. . . 47

Table 5.8. Function description of User Features Extraction Algorithm (cont). 48

Table 5.9. Description of functions in Hourly Features Extraction Algorithm. 50

Table 6.1. Description of parameters In Campaign Creation User Interface. . 62

Table 6.2. Description of used parameters for Twitter Search API. 65

Table 6.3. Campaign rough tweet and user numbers. 66

Table 6.4. User Timeline API parameters. 67

Table 6.5. Campaign analysis request parameters. 75

Table 6.6. Campaign analysis request parameters (cont). 76

Table 6.7. Campaign analysis request parameters (cont). 77

Table 6.8. Campaign analysis request parameters (cont). 78

Table 7.1. Example same user analysis result. 112

Table 7.2. Classifier results using all features. 115

Table 7.3. Classifier results using for Data Set 1. 115

Table 7.4. Classifier results for Data Set 2. 115

Table 7.5. Classifier results for Data Set 3. 116

Table 7.6. Classifier results for Data Set 4. 116

LIST OF ACRONYMS/ABBREVIATIONS

RDBMS	Relational Database Management Systems
SNA	Social Network Analysis
U.S.	United States of America
PCA	Principal Component Analysis
ROC	Receiver Operating Characteristic
ER	Entity Relation

1. INTRODUCTION

Microblogging platforms are widely used to share information, feelings, and ideas about anything. In contrast to regular blogs, microblog posts, which are short texts, are posted much more frequently and followed by much larger user bases. With nearly 320 million users (as of April 2017) Twitter is one of the most popular microblogging platforms. As such, Twitter serves as a lucrative platform for propagating (mis)information through organized activities. Such cases have been observed during election campaigns (2012 and 2016 United States), disasters (2010 Haiti earthquake), resistance movements (2011 Occupy Wall Street, 2011 Arab Spring), and marketing (2017 Pınar products in Turkey). During these events some users collaborated in favor or against a candidate/movement with coordinated effort. In some cases this is done with malicious intent as a smear campaign and other times, like natural disasters, it is done for humanitarian reasons. Also, recently there is an increasing incidence of using social media to recruit people to illegal organizations and the spread of fake news like in the 2016 United States and the 2017 France the presidential campaigns.

Recruited users utilize various Twitter entities to organize and coordinate their efforts towards a specific goal. For example, entities like hashtags serve to associate content with a purpose, while user mentions are used for referring other users. Also, external resources such as URIs are used in creating contexts for attempting to manipulate people. Besides from recruited users, fake accounts and bots are also frequently used in Twitter. In such cases, users can be manipulated, since users assume that tweets are posted with the free will of individuals without intent of collusion.

The possibility of manipulating users and targeting users, has attracted attentions of such organizations, companies, communities etc. As a result of this, Twitter has started to be used for advertisement. Since, Twitter gives a unique opportunity for building relations with targeted users, social media advertisement approaches become a crucial part of marketing campaigns. Also, its low cost attracts attention of advertisers.

Advertisement costs in social media is much more cheaper than any form of advertising available today. It is one of the only forms of media that can expose an advertisement to over 1,000 people with $3 of charge [2]. As a result of this, lots of examples and variants of social media advertisements and propagandas can be seen in social media platforms. This situation not only leads increase of Twitter use for advertising and propaganda purposes, but also leads to presence of fake and manipulative posts. In U.S. Presidential Election 2016 and 2017 France Presidential Election, there was an increase in usage of premeditated tweets. For these premeditated and non-premeditated posting behaviors, the terms "organized" and "organic" can be used respectively.

Tweets that refer to the same concepts are considered "organic" when they are spontaneously posted by various individuals who do not know each other. On the other hand, tweets that refer to the same concepts are considered "organized" when users manage the content and the timing of their posts in accordance with an agenda.

While some organized behaviors have innocuous effects, some of them can harm the decentralized and pluralist nature of Twitter platform, since users tend to post tweets based on the desire or demand of a group. For example, in the domain of group propaganda, organized behaviors are examined behind the behaviors of 20,000 pro-ISIS accounts based on what content they post, and which methods they use while spreading ISIS propaganda and recruiting followers around the globe [3].

Furthermore, another incidence of organized behavior is seen in activities of *Major American White Nationalist Movements* account on Twitter. The 600% increase in its followers between 2012 and 2016 is stated as an example of organized social media activism [1](see Figure 1.1).

Recent studies report that in the 2016 United States Presidential Election there were many negative organized campaigns on social media. "Approximately 400,000 bots were engaged in the political discourse during the Presidential Election [4, 5]."

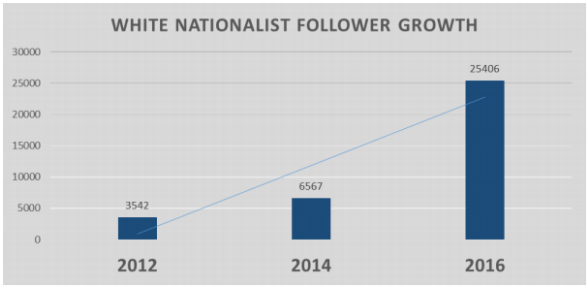


Figure 1.1. White Nationalist follower growth [1].

"These accounts were responsible of roughly 3.8 million tweets, which corresponds to about one-fifth of the entire election related posts. It is discovered that the Twitter traffic on pro-Trump hashtags was roughly double that of the pro-Clinton hashtags. About one third of the pro-Trump Twitter traffic was driven by bots and highly automated accounts in comparison to one quarter of the pro-Clinton twitter traffic."

Fake news [6] may be also considered as organized behavior. Fake news is defined as "a kind of hoax or a deliberate spread of misinformation through traditional news media or social media with the purpose of misleading people to gain some financial or political profit." Fake news is seen in both political and non-political domains. Example titles of fake news [7] are as below,

- "Trump offering free one-way tickets to Africa & Mexico for those who wanna leave America"
- "ISIS leader calls for American Muslim voters to support Hillary Clinton"
- "FBI Agent suspected in Hillary email leaks found fead in apparent murder-suicide"
- "Woman arrested for defecating on boss' desk after winning the lottery"

To protect the users and the microblogging platforms from malicious and manipulative contents, there is a need to identify patterns that may be harmful.

The manner in which Twitter was used during the 2016 U.S. and the 2017 France presidential elections have demonstrated the need for automatically distinguishing among posts that were made with and without premeditated intent. Such automation would enable systems to handle suspect posts and posters like many email clients handle spam and malicious mails. The most common approaches to dealing with manipulative content focuses on detecting bots, fraudulent users, spam content, and spammers [8–12]. However, in this studies the proposed models are aiming detection of single users or contents and these are not solely enough with the emerged use of recruited and bot accounts.

In order to overcome shortages in detection of single users, two studies proposed models, that detects group behaviors and the type of groups rather than individual users [13, 14]. In one of these studies, the sharing behavior of URLs are investigated and a model is proposed in order to detect whether or not the sharing behavior is organic or organized [13]. Furthermore, in the other study, a generic group classification approach is proposed. In this study, the proposed model achieves classification of groups by applying graph matching algorithms and it is not dependent to a specific Twitter entity like hashtag, URL.

This thesis also focuses on automatically identifying tweet sets that are "organized" or "organic". To do this, relevant user features and tweeting behavior characteristics are sought in order to build a classifier model to detect organized behavior in Twitter. In order to achieve this, the proposed model applies supervised learning algorithms for classifying sets of tweets as "organized" or "organic".

Based on these facts, in this thesis, it is intended to disclose the existence of such groups in a given set of tweets. The hypothesis of this thesis was based on the fact that if in a given set of tweets, there exists organized users or bot accounts in respectable amount, patterns that illustrate the existence of their organized behaviors can be detected. Therefore, by building classification models, it is intended to understand whether or not there is a group of organized users.

The proposed approach can help to detect whether tweets contain information about "real" trending topics or not, when a topic become trending. By this way, users can trust the contents and do not need to think about whether a trending topic is really a trending topic or a manipulated topic.The automatic detection of organized activities can also protect social media users from online fraudsters, who conduct unfair trading activities resulting illegal profits. In cases like that, fraud events may be precluded by informing targeted Twitter users about the behaviors of those accounts, who are trying to take advantage of them, are reflecting the characteristics of organized behaviors.

Therefore, our objective in this thesis is to show feasibility and possibility of implementing real time systems that can help users to detect organized behavior patterns with the help of accurate classifiers. Contribution of this thesis is explained below.

(i) Analysis of classification methods, which can be used to detect "organized behaviors" in Twitter
(ii) A valid classifier model that can be used to detect "organized behaviors" in Twitter
(iii) Understanding importance of each feature during detection of "organized behaviors" in Twitter
(iv) Understanding the cost and performance handicaps for each extracted parameters in case of using these features in a real time "Organized Behavior Detection" system
(v) Dataset. During this thesis, more than 180 million tweets are collected.
(vi) A system. The system, implemented during this thesis, collects tweets, analyzes them and extract features from these analysis. Also, the system provides visualizations of extracted features in order to see the characteristics in the data.
(vii) Understanding which technologies are more suitable for implementing a system in order to detect "organized behaviors" in Twitter

The structure of this thesis is as follows. In Chapter 2 related work is described, while Chapter 3 provides background information about the tools and systems used, and Chapter 4 explains the used approach for detecting organized activities. Later on, Chapter 5 presents the proposed model while Chapter 6 explains implemented prototype. Chapter 7 presents the experiments and results. In Chapter 8 the approach and results are discussed along with future directions. The thesis is completed with concluding remarks in Chapter 9.

2. RELATED WORK

The active usage of Twitter has gained importance during events like elections, advertisements, and propaganda. Consequently, the impact of social media has attracted the attention of researchers who have examined the characteristics of users and their behavior patterns on Twitter. The examination of individual-based behavior regarding web link (URL) sharing shows strategic patterns that is useful in identifying membership of an organized groups towards common goal [13]. In this study, it is stated that "analyzing the patterns of signals, which are embedded in URL posting activities, can uncover groups whose members engage in similar behavior." Furthermore, it is illustrated that those "group-level behavioral signals can help to distinguish organic and organized user groups."

In order to achieve this, in the study, a graph is created by connecting users who shared same URLs. Later on, Louvain Method, which is a community detection algorithm, applied on the graph in order to extract dense user groups. Then, URL based and posting time based features of those users are extracted and supervised classifications methods, such as Random Forest, NBTree, SMO and LogitBoost, are applied on manually labeled training data in order to detect organic and organized user groups. The extracted features in the classifier training data set, are as below.

- Average posting count
- URL posting entropy
- Posting interval median
- Posting interval deviation
- User tweet count deviation
- User follower counts deviation
- Registration interval median

For measures of precision, recall, f-measure, and ROC area, it is shown that applied approach gives a performance around 0.8, no matter which method or measure they choose, while Random Forest works best with high F measure (0.836) and ROC area (0.921).

In another study [15], astroturf political campaigns are studied. In this study, astroturf political campaigns are defined as "campaigns, which are run by politically-motivated individuals and organizations that use multiple centrally-controlled accounts to create the appearance of widespread support for a candidate or opinion."

During the study, topological, content-based and crowdsourced features of information diffusion networks on Twitter, which are obtained from tweets collected during 2010 United States midterm elections, are extracted. These features are mainly extracted from composed directed graph whose nodes represent individual user accounts and edges represent the retweet, mention, reply events between these users. The key extracted features from the graph are illustrated in Table 2.1.

Table 2.1. Classification features used in political abuse detection [15].

Feature	**Description**
nodes	Number of nodes
edges	Number of edges
mean k	Mean degree
mean w	Mean edge weight in largest connected component
max k(i,o)	Maximum (in,out) degree
std k(i,o)	Standard deviation of (in,out) degree
skew k(i,o)	Skew of (in,out) degree distribution
mean cc	Mean size of connected components
max cc	Size of largest connected component
entry_nodes	Number of unique injections
sentiment scores	Google-based profile of mood states and sentiment Score

Later on, in order detect the early stages of viral spreading of political misinformation, a machine learning framework is represented. In this framework, classification models are generated by using AdaBoost and SVM classification methods with a hand labeled corpus of training data. In this study, it is presented that their approach gives accuracy results better than 96% in the detection of astroturf content in the run-up to the 2010 U.S. midterm elections.

In a study about detection of spammers on social networks [11], it is said that "even though social networks have raised the attention of researchers, the problem of spam is still not well understood." In order to overcome this problem, a set of honey-profiles are created and activities of these accounts are logged in a period of one year. After that, spam accounts, which sent messages or friend follower requests to these honey profiles are detected and characteristics of those users are extracted. Later on those features are used for detecting spammers in a social network. The key extracted features are illustrated as below.

- Friend follower ratio
- URL ratio
- Message similarity
- Number of sent messages
- Friend numbers

In a different research in spam domain, detection of spam URLs in social media is studied by extracting features based on the behavioral factors of both who is posting the URL and who is clicking on the URL [9]. In this study, the idea behind using these features is based on the fact that these behavioral signals may be more difficult to manipulate than traditional signals.

For click based features, which take part in discovering click patterns, measurements are received from Bitly click API [16]. The key features extracted for given a URL are shown in Table 2.2 .

Table 2.2. Classification features used in detection of spam URLs [9].

Feature	**Description**
Posting Count	Total number of times a URL has been posted
Posting Standard Deviation	Standard deviation of the days which a URL is posted
User Network	Friend follower numbers of user
Total Clicks	Total number of clicks a URL received
Average Clicks	Average clicks per posting for given a URL's total clicks and posting count
Clicking Days	Number of clicking days a URL received clicks
Max Clicks	Maximum daily clicks
Effective Average Clicks	$\frac{totalclicks}{clickingdays}$
Click Standard Deviation	Standard deviation of daily clicks

In this work, Random Forest classification algorithm adopted to the training data which is constructed by using extracted features given in Table 2.2. After experiments are evaluated, it is illustrated that proposed behavioral approach gives satisfied results with the precision (0.86), recall (0.86), and area-under-the-curve (0.92), which suggests the potential for robust behavior-based spam detection.

In another study [14], it is pointed out that within the increase of criminal and anti-social activities in social media, detection of these behaviors are of interest to commercial enterprises, who want to guard themselves from cyber criminals, and the military intelligence analysts who desire to detect and counteract cyber wars waged by adversarial states and organizations. In the study, it is given that in order to detect these behaviors, the theories of group behaviors and interactions must be developed. As a result of this, graph matching algorithms are applied to explore consistent social interactions, which suggest that complex collaborative behaviors can be modeled and detected using a concept of group behavior grammars.

Lastly, in another research, political bot accounts,who take place in Brexit Referendum and play strategic role in referendum conversations, are analyzed [17]. It is found that these bots use excessively the family of hashtags associated with the argument for leaving the European Union, and utilize different levels of automation. Moreover, in the study, it is given that for the Brexit referendum contents in Twitter, these bots, which compose one percent of sampled accounts, generate almost a third of all the messages. However, unlike studies of [9, 11, 13–15], in [17] bot accounts are investigated after simple series of calculations are applied on the traced hashtag dataset, such as counting tweet post counts in order to find users who post more than 50 tweets in a day.

2.1. Comparison of Studies With This Thesis

When studies illustrated above are examined, this thesis differentiate from those studies in below points.

When [13] is compared with this thesis, it will be seen that the main focus of [13] is to detect organized URL behaviors, while in this thesis it is aimed to propose an approach for detecting organized behaviors for all kind of events in Twitter regardless it is a URL or not. Also, the extracted and derived features in this thesis and [13] are different from each other. However, in both study, same classification method approaches are used except in this thesis training data is prepared also by boosting and re-sampling.

Furthermore, when [15] is compared with this thesis, both study aims to detect organized behaviors in social media except this thesis does not only aim to detect organized behaviors in political domain. Besides from this, in the context of feature extraction both studies are completely different since in [15] features are extracted from a constructed graph, while in this thesis features are extracted from users' features and their collective posting behaviors. However, in both study, supervised classification methods are used although different classifiers are used.

Also, approach in [14] differentiate from this thesis. In [14] group behaviors are identified as a result of graph matching algorithms. Whereas this work focuses on organized behavior detection by extracting and deriving features in Twitter as the bases of a suitable classification model.

The work of [11] and [9] introduce an approach for detecting spammers and spam URLs, which are very disruptive and potentially destructive. Such behavior is observed in organized behavior of malicious intent. Their work focuses on identifying individual, whereas this work focuses on detecting the dissemination of groups. Group activity can be more insidious and more difficult to comprehend, since there appears to be an agreement regarding the message that leads it to be more believable. Also, used features in all of these studies are different from each other.

3. BACKGROUND

This chapter provides information about the tools and techniques utilized during the development of the model and implementation of this thesis.

3.1. Twitter

Microblogging platforms provide environments for users to share their posts with short sentences, photos, URLs, videos etc. Twitter is one of the most famous microblogging platform in the world. It allows its members to share short posts called tweets, which are limited to 140 characters. Users can post tweets by using multiple platforms and devices such as mobile, desktop browsers, rest APIs, third party applications etc.

Twitter has 313 million monthly active users, while seventy nine percent of its registered accounts in outside of United States. Furthermore, in Twitter more than forty languages are spoken and one billion unique visits have been made to platform monthly [18].

In Twitter, users can choose whether their post can be public or private. If a user prefers its posts to be public, any user in Twitter platform can see posts of user. On the other hand, if a user prefers to post its tweets in private, only its followers can see the posts. The default settings for Twitter are public which promotes anyone can follow anyone on Twitter.

Another one of the most important feature of Twitter is, tweets are shared in real time. As a result of this, Twitter users can benefit from this social media platform as a news and event source platform.

In Twitter, the friend and follower concepts are used to indicate whom a user is following and who is following that user respectively. Unlike in some social media platforms the friend concept is not reciprocal, thus, it is a directional relationship.

Users are subscribed to the posts of the users they follow. The posts of the users that are followed are displayed on the timeline of the user. The more followers a user has the more visible they may be, since their posts will appear in the timelines of their followers. Users who are interested in promoting something would want to have many followers. They would especially be interested in having followers who have many followers, since the network effect will significantly increase the visibility of their posts.

"A retweet is a re-posting of a tweet. Twitter's retweet feature enables user to quickly share a tweet that was posted by someone else." The more a tweet is retweeted the more visible it may become. The retweet count for every post is shown as it indicates how many people have shared it, giving an idea of how popular a tweet is. Again, this is a feature users who have something to promote are interested in. Highly retweeted posts usually end up in trending topics that are displayed by Twitter. Users can retweet their own tweets as well. "Sometimes people type "RT" at the beginning of a tweet to indicate that they are re-posting someone else's content." This became an emergent convention among early Twitter users, which later became officially supported by Twitter. Also, a tweet can be retweeted with a quote, whereby a user can add some comment to a tweet they are retweeting [19].

Users can refer to another user with a mention, which is the user name preceded by the character (i.e. theRealDonaldTrump). A reply to a tweet is indicated with a mention as the first token in the post, which means that the post is a response to that user. Mentions are used to create notifications to people who get mentioned. This is considered to be a good mechanism for getting attention.

In order to mark a tweet, users can like or favorite it. Twitter enables users to see which users have liked/favorited a tweet by inspecting the user's profile page.

A hashtag is used to create a relationship among tweets related to a topic, event, people, etc. It is some sort of meta-information and is denoted by a string preceded with the character #. For example, to indicate a tweet is about an election, the hashtag #2016UsElection may be used.

Hashtags support the grouping (categorizing) of tweets by means of agreeing to use a specific set of hashtags for a given purpose. It is common for designate hashtags to be announced prior to an event for the purposes of marking posts related to that event. Users search for hashtags to see the posts that contain them [20].

3.1.1. Spam

In general, spam behaviors described as malicious and deceiving actions that impact other users negatively. Examples and characteristics of spam behaviors are listed as below [21].

- Posting harmful links (including links to phishing or malware sites)
- Aggressive following behavior (mass following and mass unfollowing for attention)
- Abusing the reply or mention functions to post unwanted messages to users
- Creating multiple accounts (either manually or using automated tools)
- Posting repeatedly to trending topics to try to grab attention
- Repeatedly posting duplicate updates
- Posting links with unrelated tweets

3.1.2. Bot and Cyborg Accounts

Accounts, which post, retweet, like tweets automatically as a result of machine automated software without any human interaction, are called as bot accounts. Cyborg accounts are similar to bot accounts. However, except that in cyborg accounts human interference is present. So, bots are autonomous, whereas cyborg accounts are semi-autonomous. Such accounts may serve the purpose of providing useful information (i.e. weather and earthquakes) or harbor malicious intent (smear campaigns and fake news).

3.1.3. Twitter REST API

The Twitter REST API methods provide numerous read and write functionalities in Twitter Domain including update timelines, status data, and user information.

The Twitter Search API is part of Twitter's REST API. It provides means for searching tweets based on queries and various criteria such as geographic location and time. It returns historical data by searching through a sampling of recent tweets (published in the latest seven days) that are optimized for relevance. The Search API does not search through complete tweets. It focusing on relevance, not completeness. It is worth noting that the results of work on tweets selected by the search API are subject to the sampling strategies of this API, which are not made available.

The User Timeline API is another API provided by Twitter Rest API. It supports the collection of tweets for a given user. Queries are made on the user screen name.

Twitter APIs impose access limits in order to prevent excessive queries. In order to manage the rate limits, there are functions to examine the remaining call limits for a given time window. The use of these functions are essential when continuous tweet collections is needed. In this work, sets of tweets are collected based on hashtag queries. The queries are called campaigns and are continuous collections, such as #draintheswamp.

3.2. Social Network Analysis

"Social network analysis (SNA) is the process of investigating social structures through the use of networks and graph theory [22]." Social network analysis empowers users to apply numerous functionalities including segmentation of users, community detection, and tracing of information flows in social media platforms.

3.2.1. Centrality

In Social network analysis, centrality metrics aims to answer question "Who is the most important or central person in the given network ? [23]"

However, the importance of a person in a network varies according to what has been looking for in the network. Therefore, there is no certain and single centrality measurement approach in order to find the most important nodes in a network. In this thesis, we benefit from two centrality measures which are degree centrality and closeness centrality.

3.2.1.1. Degree Centrality. Degree centrality is calculated by counting edges of node. For directed graphs, degree centrality is separated as in-degree centrality and out-degree centrality. In-degree centrality of a node is calculated by counting edges which are directed from other nodes to the node. On the other hand, out-degree centrality of a node is calculated by counting edges directed other nodes from the node. In-degree centrality is often interpreted as a form of popularity, while out-degree centrality is interpreted as gregariousness. In Twitter domain, friend count of a user corresponds to out-degree centrality, while follower count of a user corresponds to in-degree centrality [24].

3.2.1.2. Closeness Centrality. In social network analysis, closeness centrality represents how much a user is close to rest of the network and this can display the familiarity of a user to other users in the network, which means how much they know each other. By this way, if a user has the highest closeness centrality score in a network, it represents that this user has a high possibility to influence other users within the network.

In order to calculate a user's closeness centrality in the network, its distance (a.k.a shortest path between two users) to all other users within the network are calculated and average of its result is considered.

3.3. Natural Language Processing

Natural Language Processing is a research area which aims to make possible for computers to understand and interpret spoken and written human languages [25].

3.3.1. Sentiment Analysis

Sentiment analysis is a sub-field of Natural Language Processing. In sentiment analysis, it is aimed to find the sentiment of given text or speech for a given language. While in some analysis, the output of Sentiment Analysis is simple as Positive, Neutral and Negative, in other analysis, sentiment of a text is given in a very large spectrum such as happy, sad, angry, disappointed, surprised, proud, in love, and scared.

In this thesis, Stanford Core NLP tool [26] is used for sentiment analysis. The retrieved results of sentiment consists of five values such as very negative, negative, neutral, positive, and very positive.

3.3.2. Text Similarity

In text similarity, it is aimed to find how close two pieces of text in contexts of meaning and surface closeness. The meaning closeness can be referred as semantic similarity while surface closeness reflects the lexical similarity [27]. In order to determine how two texts are similar to each other, there are numerous of approaches. In this thesis, cosine similarity approach is used for this purpose.

3.4. Classification

In machine learning, classification is the problem of identifying the category or the class of an entity automatically.

In machine learning domain, classification methods are separated into three different approaches [28]. These are,

- Supervised Learning Methods
- Semi Supervised Learning Methods
- Unsupervised Learning Methods

In this thesis, supervised learning methods are used in order to detect organized behavior. In supervised learning methods, a model is trained by training dataset. Note that, training dataset consists of labeled data samples (a.k.a observations) that illustrates which data belongs to which class. Later on, after the model is trained with the help of training data, these classification methods predicts the classes in the dataset, which has unclassified data.

In other words, classification can be seen as a case of pattern recognition. Problems of number recognition, color recognition, spam mail recognition, fraud detection etc... are all subjects to the classification.

3.4.1. Resampling

In statistics, resampling methods cover [29],

(i) "Estimating the precision of sample statistics (median,mean, variance) by using subsets of available data or generating subsets with replacement from data set."
(ii) "Exchanging labels on data points when performing significance tests"
(iii) "Test generated models by using random subsets (bootstrapping, cross validation)"

By using resampling techniques [30], it is aimed to

(i) Train non-biased classifier models for Supervised Learning approaches
(ii) Increase variances in the training data set in order to prevent overfitting of models
(iii) Increase training data samples in case of the actual training data size is small

3.4.1.1. Ten Fold Cross Validation. In supervised learning methods, when there is a small training data set or training data consists of a large number of features, it is likely that generated model is overfitted, which means it gives good accuracy results with training data while giving low accuracy results with test data.

In order to prevent this issue, it is common to use ten fold cross validation during training of model. During usage of ten fold cross validation, whole training data set is divided into ten subsamples. Later on, in training phase, training of model is accomplished with nine subsamples of the training data and the obtained model is tested with tenth subsample of the training data. Moreover, this process is repeated ten times by switching the subsample in training data and test data which aims to test every sub sample with the obtained model which is trained with remaining of the data set. After all ten subsamples are tested with the trained model, the result is averaged to see whether or not the obtained model is good enough [31].

3.4.1.2. Simple Random Sampling. A simple random sample is a subset of data chosen from a larger set . In this approach, data of each k subset is selected randomly from the actual dataset with the same probability in order to have an unbiased dataset.

3.4.1.3. Bootstrapping. In Bootstrapping, sample data sets are generated by resampling of the actual data set. This resampling actions may be with or without replacement of data.

There are numerous of Bootstrapping techniques in the literature. However, the technique, which is closest to approach that is used in this thesis, is block bootstrapping [32, 33] .

In the moving block bootstrap [33], data is divided into $n - b + 1$ overlapping blocks of length b in a data set with size n. Then from these blocks, $\frac{n}{b}$ blocks will be drawn at random with replacement [34].

On the other hand, in another bootstrap approach [32], sub samples are constructed from observations per unit. By this way, it becomes easy to obtain bootstrapped samples. Furthermore, in this approach resampling actions, which do not change the observations in the data set, are only made within the groups [34].

3.4.2. Classification Methods

In this study, three supervised learning algorithms are used. These are "Random Forest", "Support Vector Machine", and "Logistic Regression".

Random Forest is an ensemble classifier that includes multiple decision trees [35]. "The Random Forest approach tries to decorrelate the trees by randomizing the set of variables that each tree is allowed to use. For each individual tree in the ensemble, the random forest method does the following:"

(i) "Draws a bootstrapped sample from the training data"
(ii) "For each sample, grows a decision tree, and at each node of the tree"
- "Randomly draws a subset of variables from the all given total features"
- "Picks the best variable and the best split from variables"
- "Resumes until the tree is fully grown"

As a result, the final ensemble of trees is formed to make predictions.

Support Vector Machine algorithm is mostly used in classification problems while it can be used also for regression problems. In general, "a support vector machine constructs a hyperplane or set of hyperplanes in a high dimensional space." During the construction of hyperplanes it is aimed to have hyperplanes with the largest distance since as the margin getting larger, the classification error gets lower. Furthermore, support vector machines are preferred in case of applying linear classification to a non-linear dataset. This can be achieved by applying kernel functions to non-linear data in order to map the data to a higher dimensional space which make to separate data linearly easier [36, 37].

Logistic Regression is one of the most important member of linear models and also it is a good first choice for binary classification problems [35].

3.4.3. Principal Component Analysis

For classification, it can be hard to detect patterns during training and testing the model in case of the number of features in the data is large. In order to overcome this problem, dimension reduction approaches can be applied on the dataset. Principal Component Analysis (a.k.a PCA) method is one of the dimension reduction functionality used in the domain of machine learning.

In PCA, it is aimed to find smallest number of uncorrelated features, which contains the highest variances in all features. By this way, it is aimed to find the features, which will cause minimal data loss in case of they are not used during training phase [28].

3.4.4. Tools

In this thesis, for machine learning tasks Weka [38] and Apache Spark [39] tools are used.

3.4.4.1. Weka. Weka is a machine learning tool, which can be used in data mining tasks with its support of numerous machine learning algorithms. Also, Weka implements tools for data clustering, feature selection, pre-processing, classification,visualization regression, clustering, association rules [38].

3.4.4.2. Apache Spark. Apache Spark is an open source data processing tool, which is commonly used in the domain of Big Data. For machine learning purposes, Spark provides a library called MLlib [39], which enables usage of different classification, regression, feature selection and dimension reduction algorithms.

3.5. NoSQL Databases

With the evolve of technology and internet, data is created and its format changes more than it has been ever. Furthermore, relational database systems become inadequate and can not answer the needs of this rapid changes happen in data.

NoSQL databases become an option for the evolving data of today's application since in NoSQL databases, unstructured, semi structured and structured data can be stored, while in classical RDBMS systems only structured data can be stored. Furthermore, NoSQL databases are designed to cope with scale and agility challenges of today's rapidly generated and evolving data [40].

In this thesis, NoSQL databases MongoDB and Neo4J are used for data persistence.

3.5.1. Document Based NoSQL Databases

"Document based databases pair each key with a complex data structure known as a document which documents can contain many different key-value pairs, or key-array pairs, or even nested documents [40]. MongoDB is also a document based NoSQL database. "

3.5.2. NoSQL Graph Databases

In Graph databases, network and graph data such as social media connections are stored in contexts of graph nodes and edges. Neo4j is an example of a NoSQL graph database.

4. APPROACH

When organized activities in the domain of fake news, politics, group propaganda, and fraud are examined, it reveals that organized behavior exhibits the characteristics of "a goal", "synchronization between users", and "the adoption of shared messages" [41–43]. These characteristics enable the message that is desired to be disseminated to first become visible due to simultaneous posting. And, thereafter to spread with sharing.

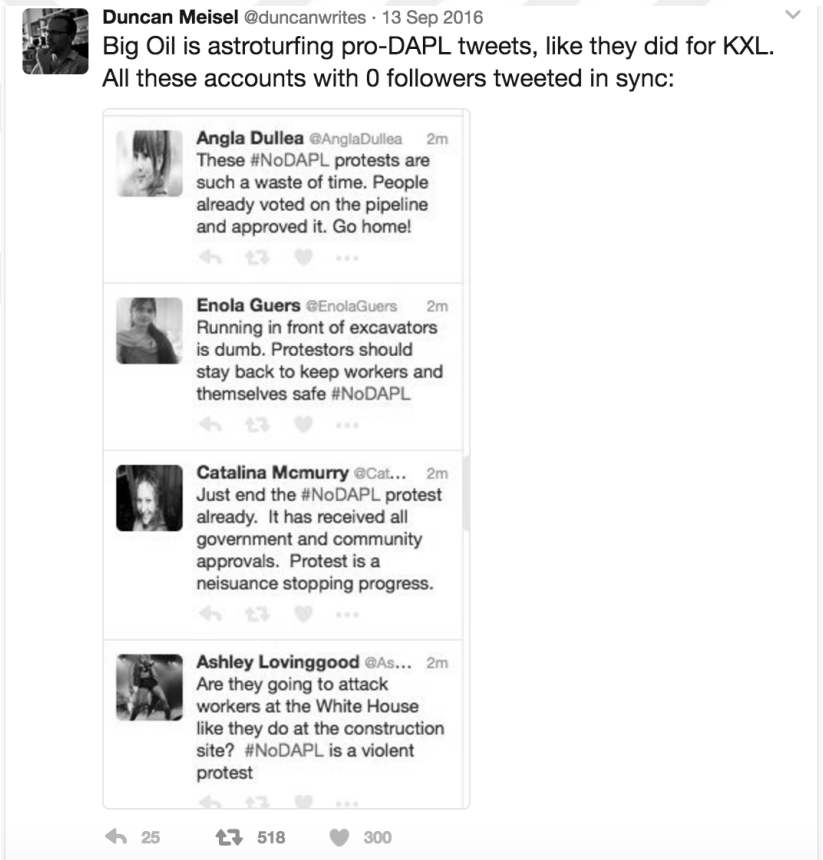


Figure 4.1. Example of the synchronization in organized behavior.

In organized behaviors the following items must be satisfied for influencing users.



Figure 4.2. Example of the adoption of shared messages.

- The visibility of messages increase,
- Messages persists,
- Messages are effectively deployed,
- Messages appeal to those who would facilitate reaching the goal

In Twitter, such coordination may be achieved with:

- Choosing effective hashtags & vocabulary,
- Retweeting,
- Posting similar tweet context,
- The use of multiple accounts to respond to posts to create an impression of numerous supportive participant,
- The use of multiple predefined tweets with the same or similar meaning to convey the same message without being repetitive. This also reduces the chance of being perceived as a bot or spam account.
- Automated management of multiple identities.

4.1. Organized Behavior Use Cases

According to the given organized behavior characteristics, organized behaviors can be seen in below use cases.

4.1.1. Making a Hashtag Top Trend

Top trend concept is one of the most important aspect of Twitter. When an event occurs, people tend to look at Twitter first. When they check out Twitter, they usually first look at top trend topics in order to see what is happening or spoken at this very moment. Because of this reason, top trend mechanism is one of the most manipulated aspect of Twitter. Some authorities and companies hire Twitter users and synchronize them to post a tweet contains same hashtag in same time in order to make the hashtag top trend.

By this way, an intervention, which has happened as a result of an organized manipulation of system, occurs in Twitter, and corrupts the natural and pluralist nature of Twitter. As a result of these organized behaviors, Twitter can lose its up-to-date architecture. Furthermore, its Top Trend functionality can lose its purpose and this may harm Twitter's reputation as an online public news resource.

4.1.2. Spam Attacks

Spam is described as a variety of prohibited behaviors that violate the Twitter Rules [21]. Twitter defines characteristics of a Spam account as below.

- Posting harmful links (including links to phishing or malware sites)
- Aggressive following behavior (mass following and mass unfollowing for attention)
- Abusing the reply or mention functions to post unwanted messages to users
- Creating multiple accounts (either manually or using automated tools)
- Posting repeatedly to trending topics to try to grab attention
- Repeatedly posting duplicate updates
- Posting links with unrelated tweets

However, it is not always the case for users who are reported as Spammers. Since Twitter has a very wide spectrum of users from very different cultures, interests, political opinions, it is possible that some conflicts occur between users. Sometimes, these conflicts between users can ended up by reporting each other as spammers although they have not been committed any of the prohibited behaviors that are announced as spamming by Twitter.

What is more, these spamming notifications are not always reported by solely individuals. Sometimes, users report a user as spammer in an organized way by telling each other to spam her just because they have a conflict with her based on their culture, interest, political view etc.

As a result of this kind of organized spam attack behaviors, since large amount of spam notification is reported in Twitter, Twitter may suspend the user,who is reported as spammer, although she is not spammer at all. This may harm freedom of speech. Furthermore, communities, who are able to behave in an organized way, can dominate the Twitter by eliminating users who do not share their opinions.

4.1.3. Charity Behaviors

Twitter provides a perfect place for charity communities, who work in charity related business and ask for donations [18]. However, user accounts may not always illustrate the real identity of users who ask for donation .

How can a user trust charity users to donate money on their Charity projects? It may be an "Organized Fraud Action" in order to deceive users for stealing their money under the name of charity.

If charity networks are analyzed and users are informed whether those behaviors in the given network reflects characteristics of organized behaviors or not, users can feel more safe and can be more willingly to make donations.

4.1.4. Humiliation

Celebrities and politicians also have accounts on Twitter. Sometimes a humiliation campaign can be conducted on Twitter in order to harm their reputation. Since, these humiliation campaigns can harm a person's reputation in a significant way, it would be better if the characteristics of these behaviors are examined and announced to users whether or not these humiliation behaviors reflects spontaneous responses or they are done as a result of organized behaviors (2016 U.S. Presidential Election).

4.1.5. Flattery

Trust is an important concept in online social networks. Because of this reason, there can be organized behaviors directed to a user, which have remarkably positive statements (2016 U.S. Presidential Election). Since, these kind of "Organized Flattery Behaviors", can raise a "Fake Trust" relationship between users, it would be better if these kind of organized flattery actions are detected and users are informed about it.

4.1.6. Propaganda

United States Presidential Election 2012 and 2016 are illustrated that Twitter is also used for propaganda purposes. Furthermore, [3] emphasized the usage of propaganda in Twitter for recruitment of ISIS.

Therefore, detection of organized behaviors in propagandized tweets can help users to be aware of that the content they are reading is created in a premeditated way by an organized group.

4.1.7. Fake News

Fake news are also seen as an example of organized behaviors which occurs both in political and non-political domains. The definition of fake news is given as a type of hoax or deliberate spread of misinformation with the help of traditional news media or social media, to mislead people in order to gain some financially or politically oriented profit [6]. Fake news was also extremely occurred during U.S. Presidential Election 2016 [7].

4.2. General Features and Patterns that Can Be The Sign of Organized Behavior

Below features can be used to detect organized behaviors on Twitter :

4.2.1. Twitter Account Opening Date

If a remarkable amount of user accounts in a tweet dataset are created in close dates to each other, it can be an indicator of an organized movement. Because, this may show an organized bulk user creation in close dates.

4.2.2. Closeness Centrality of Users in the Friend-Follower Network

As it is stated in Chapter 3, closeness centrality illustrates how much a user is familiar with other users in a network.

When the Friend - Follower networks of users, who posted same hashtag or in same content, are examined, if a remarkable amount of users have high closeness centrality in the network, this illustrated the users are familiar to each other. Thus, it can be a sign of an organized behavior.

4.2.3. Relation Between Communities

In behaviors which aim to harm or humiliate a specific user, analyzing characteristics of the communities,which attacking and attacked user belongs to, can be helpful. In cases like this, reply networks between these communities can be beneficial, if remarkable amount of reply tweets between those communities consist of Negative sentiments. In a condition like this, reply tweets with Negative Sentiment can guide to existence of an organized behavior.

4.2.4. Tweeting Behaviors – Tweet History

Features related with tweet history are explained as below. This three sections corresponds to different characteristics of organized behaviors. Here "Retweeting Same Tweets" refers to characteristic of "Share", while "Information Flow During The Propagation of Retweets" corresponds to "Goal". On the other hand, "Posting Tweets About Same / Similar Topics in Same Time of the Day" reflects the characteristic of "Synchronization" .

4.2.4.1. Retweeting Same Tweets. When tweet history of users,who have tweets with same hashtags or about same content, are examined, retweet patterns and similarities of their tweets can be indicator of organized behaviors. Because after some point, similarity of patterns and tweets can not be coincidence.

4.2.4.2. Information Flow During The Propagation of Retweets. When the tweet history of users, who retweeted same tweets, are analyzed, if repeating patterns exist in the information flows of retweets, it can be a sign of organized behavior. Here, information flow is aimed to be used for propagation of those retweets. For example, suppose that User A tweeted Tweet X, and this tweet is retweeted by User B and later on retweeted by User C etc. After a while, User A tweeted Tweet Y at this time and this tweet also retweeted by User B and then retweeted by User C and so on. Since, both of these Retweeted tweets propagated in similar way, after some point we can not mention about a coincidence, there must be an organized behavior in the spread of those retweets.

4.2.4.3. Tweets About Same / Similar Topics in Same Time of the Day. Users' tweet hours and their topics can be also a sign of organized actions in Twitter. For example, when the tweet histories of the users in a community is checked, if most of those tweets about relevant topics are posted same hour of the day, here we can suspect from an organized behavior.

4.2.5. Checking Community Users : Are There Bot or Cyborg Users ?

Since it is harder to find real users who will take place in organized actions than creating automated bot accounts. So, the ratio of bot & cyborg users in a set of tweet group can tell about whether those tweets are posted as a result of organized behavior or not.

4.3. Which Features Can Be Used In Which Use Cases ?

Our hypothesis is that features examined in Section 4.2 can not be used individually in order to detect organized behaviors. Therefore, these features should be used together. However, when use cases are analyzed, it is seen that some features can be more helpful in some cases, while other features contribute poorly in order to detect organized activity . As a result of this, in order to get results with high accuracy, we grouped features base on use cases.

4.3.1. Hypothesis I - Top Trend Hashtags & Propaganda & Flattery & Fake News

For cases of top trend hashtags and propaganda, flattery and fake news, our hypothesis is that detection of organized behaviors can be done by analyzing below behavior patterns & features of users.

- Twitter account opening date of user
- Closeness centrality of users in the Friend-Follower network of the community
- Tweeting behaviors of users
  - (i) Retweeting same tweets
  - (ii) Information flow during the propagation of retweets
  - (iii) Posting tweets about same / similar topics in same time of the day
- Bot & cyborg analysis of the users

4.3.2. Hypothesis II : Spam Attacks & Humiliation & Fake News

In cases of Spam Attacks & Humiliation & Fake News, our hypothesis is that detection of organized behaviors can be achieved by analyzing below patterns & user features.

- Twitter account opening date of user
- Closeness centrality of users in the Friend-Follower network of the community
- Tweeting behaviors of users
  - (i) Retweeting same tweets
  - (ii) Information flow during the propagation of retweets
  - (iii) Posting tweets about same / similar topics in same time of the day
- Relation between communities (between the communities of attacked user and communities of attacker Users)
- Bot & cyborg analysis of the users

4.3.3. Hypothesis III : Charity Behaviors

Our hypothesis is evaluation of trust for users in the charity community can be done by below analyzing below feature categories.

- Twitter account opening date of user
- Bot & cyborg analysis of the users

5. MODEL

Hashtags provide a mechanism to relate tweets under a metadata, therefore in this thesis datasets were collected based on hashtags. Note that, each of these datasets are called as a campaign [1] .

The main steps of the general flow are as follows (see Figure 5.1):

(i) Collect organic and organized tweet sets based on hashtags (each set is called a campaign),
(ii) Analyze campaign tweets to extract features,
(iii) Creating a training dataset where each row correspond to the features extracted for campaigns, and
(iv) Training a model to detect organized campaigns.

In the general flow, the main modules are Feature Extractor and Classifier. The Feature Extractor module analyzes tweets in a campaign and extracts features from them. The Classifier module trains a model by using the features extracted by Feature Extractor module. The main algorithm is shown in Figure 5.2.

5.1. Campaign Creation

Since hashtags provide a mechanism to relate tweets under a metadata, it is more likely that organized activities can be seen in hashtag usages. Therefore, in this thesis, the generated classifier model is created by analyzing extracted features from tweets with hashtags. “Campaign” is used as a mechanism for grouping the given hashtag’s tweets.Note that, a campaign holds only tweets of a single hashtag and for the same hashtag, different campaigns, which preserves different tweets of the hashtag, can be created (Figure 5.3).

[1] A campaign is dataset of tweets which are collected based on a given hashtag

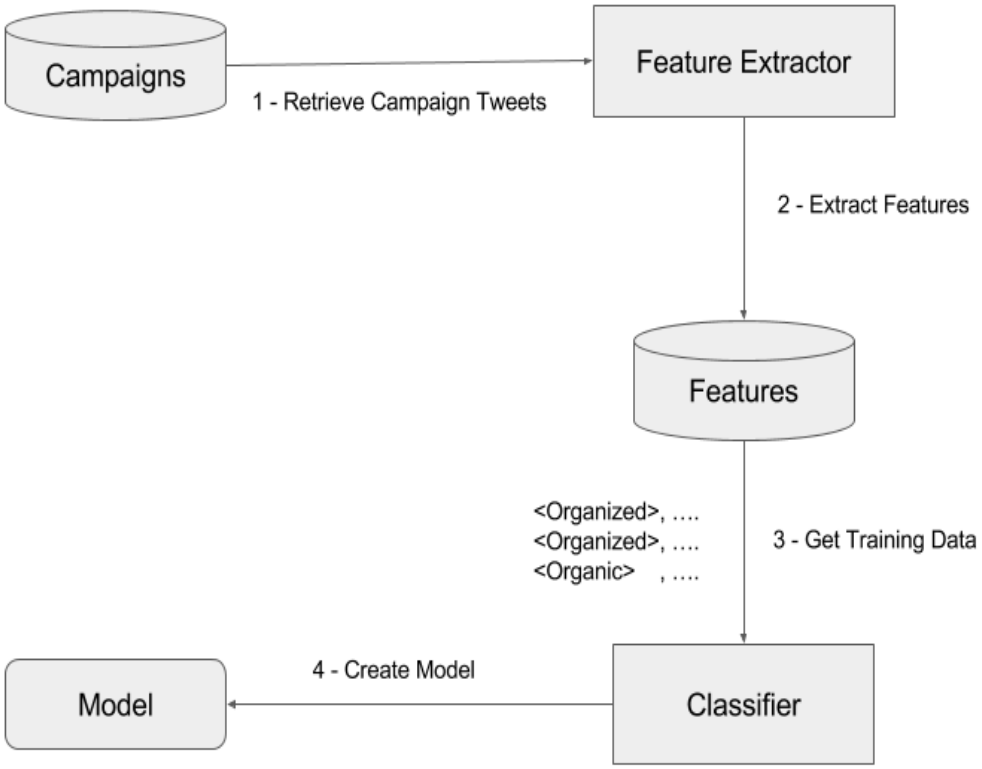


Figure 5.1. General Model.

As it is illustrated in Figure 5.1, campaigns should be created in order to execute Feature Extractor module. Executed steps during campaign creation are as below.

(i) Retrieving tweets of given hashtag, which are posted in a given time interval.
(ii) Calculation of basic statistical values for retrieved hashtag tweets.
(iii) Saving retrieved tweets of given hashtag and its basic statistics under a campaign.

Require *collectedTweets*
CampaignTweets $campaignTweets \leftarrow getCampaignTweets(collectedTweets)$
for all $hashtag, hashtagTweets \in campaignTweets$ **do**
  $allExtractedFeatures \leftarrow extractFeatures(hashtag, hashtagTweets)$
end for
$trainingData \leftarrow getTrainingData(allExtractedFeatures)$
$trainedModel \leftarrow trainClassifierModel(trainingData)$
return *trainedModel*

Figure 5.2. Model Creation Algorithm.



Figure 5.3. Campaign Hashtag ER diagram.

The aim of calculating basic statistical values is to figure out the characteristics of retrieved tweets. This might help during manual labeling to understand whether the behavior patterns in tweets indicate the existence of organized or organic behaviors. These calculated variables are not used in training data. They are just used for observation of campaigns during manual labeling.

General Flow diagram of campaign creation can be seen in Figure 5.4. Also, in Figure 5.5, high level algorithm of campaign creation is given. Meanwhile, explanations of used parameters and functions in Campaign Creation Algorithm (Figure 5.5) are given in Table 5.2 and Table 5.1.

Besides from these, in Figure 5.6, calculations of basic statistical values in campaign tweets are explained in detail. Descriptions of functions used in Figure 5.6, can be found in Table 5.3.

Table 5.1. Parameters of campaign creation functions (Figure 5.5, Figure 5.6).

Parameters	**Description**
campaignId	Unique identifier of campaign, which will be created
tracedHashtag	Hashtag, whose tweets will be retrieved from Twitter
minCampaignTweetDate	The earliest Date, which retrieved tweets should be posted
maxCampaignTweetDate	The latest Date, which retrieved tweets should be posted
allHashtagTweets	Retrieved tweets which contains *tracedHashtag* and posted between minCampaignTweetDate maxCampaignTweetDate dates

Table 5.2. Description of functions in Campaign Creation Algorithm (Figure 5.5).

Functions	**Description**
getTweetsFromTwitter(*tracedHashtag*, *minCampaignTweetDate,* *maxCampaignTweetDate*)	Retrieve tweets from Twitter, that contains the given hashtag and posted between specified dates
calculateCampaignStatistics(*allHashtagTweets*)	Computes all campaign related statistics of $allHashtagTweets$. Detail of calculated statistics can be seen from Figure 5.6.
saveCampaign(*campaignId*, *allHashtagTweets*, *campaignStatistics*)	Saves $allHashtagTweets$ with its calculated statistics

Table 5.3. Description of functions in Campaign Statistics Calculation Algorithm (Figure 5.6).

Use Paramaters	
$allHashtagTweets$	
Functions	**Description**
findWordCount	Find counts of total words in $allHashtagTweets$
findDistinctWordCount	Find counts of distinct words in $allHashtagTweets$
calculateHashtagUsagePercentages	Calculates usage percentages of each hashtag in $allHashtagTweets$
calculateHashtagUsagePercentageVariances	Calculates variances & standard deviation of usage percentages of each hashtags in $allHashtagTweets$
calculateTweetCountPerUser	$\frac{totalTweetCount}{totalUserCount}$
calculateRetweetPercentages	Calculates percentage of retweeted tweets in $allHashtagTweets$
calculateMentionPercentages	Calculates percentage of mention tweets in $allHashtagTweets$
calculateReplyPercentages	Calculates percentage of reply tweets in $allHashtagTweets$

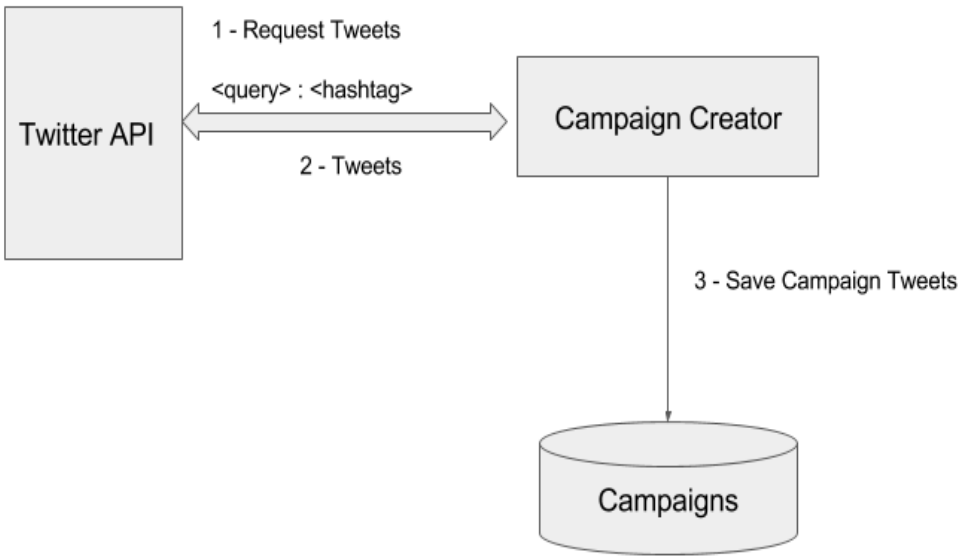


Figure 5.4. Campaign creation general flow.

5.2. Feature Extraction

As it is given in Figure 5.1, during feature extraction, Feature Extractor module retrieves tweets for each campaign and extracts feature in those tweets. It is important to understand that Feature Extractor, extracts features for each campaign as it is illustrated in Figure 5.7.

In Figure 5.8, feature extraction algorithm is explained in detail. Below the applied steps for each campaign can be found.

Require *campaignId*

Require *tracedHashtag*,

Require *minCampaignTweetDate*

Require *maxCampaignTweetDate*

$allHashtagTweets \leftarrow getTweetsFromTwitter(tracedHashtag,$
$minCampaignTweetDate, maxCampaignTweetDate)$

$campaignStatistics \leftarrow calculateCampaignStatistics(allHashtagTweets)$

$saveCampaign(campaignId, allHashtagTweets, campaignStatistics)$

Figure 5.5. Campaign Creation Algorithm.

Require *allHashtagTweets*

$findWordCount(allHashtagTweets)$

$findDistinctWordCount(allHashtagTweets)$

$calculateHashtagUsagePercentages(allHashtagTweets)$

$calculateHashtagUsagePercentageVariances(allHashtagTweets)$

$calculateTweetCountPerUser(allHashtagTweets)$

$calculateRetweetPercentages(allHashtagTweets)$

$calculateMentionPercentages(allHashtagTweets)$

$calculateReplyPercentages(allHashtagTweets)$

return

Figure 5.6. Campaign Statistics Calculation Algorithm.

< Campaign 1 > - < feature 1 > , < feature 2 > , ..., <feature n>, ORGANIC

< Campaign 2 > - < feature 1 > , < feature 2 > , ..., <feature n>, ORGANIZED

⋮

< Campaign (n-1) > - < feature 1 > , < feature 2 > , ..., <feature n>, ORGANIZED

< Campaign (n) > - < feature 1 > , < feature 2 > , ..., <feature n>, ORGANIC

Figure 5.7. Example of extracted features.

(i) Retrieving all collected campaign tweets using given hashtag

(ii) Find users of campaign tweets

(iii) Collect all campaign users' tweets which are posted in $userTweetAnalyzeDuration$[2]

(iv) Extracting user features of all campaign users (see Section 5.3)

(v) Analyzing all users' tweets which are posted in $userTweetAnalyzeDuration$ and extracting hourly features from those tweets (see Section 5.4)

(vi) Finding mean, variance, standard deviation, min and max values of all extracted features

Description of used parameters in algorithms are given in Figure 5.8, Figure 5.10, and Figure 5.9 is explained in Table 5.4. Furthermore, explanation of used functions in General Feature Extraction Algorithm (Figure 5.8) can be found in Table 5.5, and Table 5.6.

5.3. User Feature Extraction

In order to use in training data set, all of $campaignUsers'$ characteristics and features extracted. Algorithm for feature extraction is given in Figure 5.9. In Table 5.7, and Table 5.8, descriptions of functions used in the algorithm can be found. In Chapter 6, extracted features are explained in detail.

5.4. Hourly Feature Extraction

In order to detect organized activities with high accuracy, hourly analysis of all campaign users' tweets, which are posted in $userTweetAnalyzeDuration$, is also done by extracting behavior patterns and features of them (Figure 5.8).

During this phase, hourly intervals are found by extracting each hour between the earliest post date and the latest post date in the all campaign users' tweets.

[2] $userTweetAnalyzeDuration$ refers to time interval of seven days before and after the date when the campaign tweet was posted.

Table 5.4. Description of parameters used in feature extraction algorithms (Figure 5.8, Figure 5.10, Figure 5.9).

Parameters	**Description**
campaignId	Id of campaign whose features will be extracted
tracedHashtag	Hashtag whose tweets will be processed
label	Label info used in training data. Label value can be *Organic* or *Organized*
campaignUsers	Users whose tweets collected during campaign creation
campaignHashtagTweets	Tweets which contains tracedHashtag and retrieved during campaign creation
userFeatures	Extracted Features of users which will be used in training dataset
usersTweets	Tweets of users whose tweets are collected during campaign creation. $usersTweets$ contains user's tweets which are posted between $userTweetAnalyzeDuration$
hourIntervals	Hourly intervals between the latest tweet date and the earliest tweet date of all $usersTweets$. Finding hourly intervals supports making analysis in hourly basis.
tweetsInHourInterval	Tweets posted in given $hourInterval$
hourlyFeatures	Extracted features from tweets in $tweetsInHourInterval$
statisticalValues	mean, variance, standard deviation, minimum and maximum values of all extracted features

Table 5.5. Description of functions in General Feature Extraction Algorithm (Figure 5.8).

Function	**Description**
retrieveTweetsInCampaign(*campaignId,tracedHashtag*)	retrieve collected tweets during campaign creation which belong to given *campaignId* and use *tracedHashtag*
getCampaignUsers(*campaignHashtagTweets*)	extracts users who posted *campaignHashtagTweets*
extractUserFeatures(*campaignUsers,tracedHashtag*)	extracts features of users who used *tracedHashtag* in *campaignHashtagTweets*. This function is explained in detail in Figure 5.9
collectTweetsOfUsers(*campaignUsers*)	collects *campaignUsers′* tweets from Twitter. During collection, tweets which are posted between *userTweetAnalyzeDuration*
getHourlyIntervals(*usersTweets*)	gets hourlyIntervals between the latest tweet date and the earliest tweet date of all *usersTweets*. Finding hourly intervals supports doing analysis in hour basis.

Table 5.6. Description of functions in General Feature Extraction Algorithm (cont). (Figure 5.8)

Function	**Description**
findTweetsInHourInterval(*usersTweets, hourInterval*)	gets users' tweets posted in the given time interval
extractHourlyFeatures(*tweetsInHourInterval*)	extract hourly features of *tweetsInHourInterval* for training data. This function is explained in detail in Figure 5.10
calculateBasicStatisticsForAllExtractedFeatures	finds mean, variance, standard deviation, minimum and maximum values of all extracted features
saveTrainingData(*label,userFeatures, hourlyFeatures, statisticalValues*)	saves all extracted values and features in order to use them during training of model

Require *campaignId*

Require *tracedHashtag*

Require *label*, Behavior Label, "Organic" or "Organized"

$campaignHashtagTweets \leftarrow retrieveTweetsInCampaign(campaignId, tracedHashtag)$

$campaignUsers \leftarrow getCampaignUsers(campaignHashtagTweets)$

$usersTweets \leftarrow collectTweetsOfUsers(campaignUsers)$

$userFeatures \leftarrow extractUserFeatures(campaignUsers, tracedHashtag, usersTweets)$

$hourIntervals \leftarrow getHourlyIntervals(usersTweets)$

for all $hourInterval \in hourIntervals$ **do**

  $tweetsInHourInterval \leftarrow findTweetsInHourInterval(usersTweets, hourInterval)$

  $hourlyFeatures \leftarrow extractHourlyFeatures(tweetsInHourInterval)$

end for

$statisticalValues \leftarrow calculateBasicStatisticsForAllExtractedFeatures()$

$saveTrainingData(label, userFeatures, hourlyFeatures, statisticalValues)$

Figure 5.8. General Feature Extraction Algorithm.

Table 5.7. Description of functions in User Features Extraction Algorithm (Figure 5.9).

Use Parameters	
$campaignUser$	
Function	**Description**
findUserFavoritePostCount	retrieve user's total favorite post count
findUserPostCount	retrieve user's total post count
findUserPostCountWithHashtag (tracedHashtag)	retrieve user's post count which uses $tracedHashtag$ and posted between dates of $userTweetAnalyzeDuration$
calculateUserFriendFollowerRatio	calculates $campaignUser's$ friend follower ratio (see Section 5.6.1.1)
calculateUserUrlRatio	calculates $campaignUser's$ url ratio (see Section 5.6.1.1)
calculateUserMentionRatio	calculates $campaignUser's$ mention ratio (see Section 5.6.1.1)
calculateUserHashtagRatio	calculates $campaignUser's$ (see Section 5.6.1.1)
calculateUserMediaRatio	calculates $campaignUser's$ media ratio (see Section 5.6.1.1)

Table 5.8. Function description User Features Extraction (Figure 5.9) (cont).

Use Parameters	
$campaignUser$	
Function	**Description**
calculateUserDailyAvaragePostCount	calculates $\frac{(user's alltweetcounts)}{user's activedayscountinTwitter}$ (see Section 5.6.1.1)
findAverageDailyHashtagTweetCount(tracedHashtag)	finds average daily count of tweets with $tracedHashtag$ (see Section 5.6.1.1)
findUserDailyPostRatio	finds ratio of daily count of tweets with $tracedHashtag$ to all tweets posted in a day (see Section 5.6.1.1)
extractUserRegistrationDate	extracts $campaignUser's$ Twitter registration date (see Section 5.6.1.1)

Require *campaignUsers*

Require *tracedHashtag*

for all $campaignUser \in campaignUsers$ **do**

$findUserFavoritePostCount(campaignUser)$

$findUserPostCount(campaignUser)$

$findUserPostCountWithHashtag(campaignUser, tracedHashtag)$

$calculateUserFriendFollowerRatio(campaignUser)$

$calculateUserUrlRatio(campaignUser)$

$calculateUserHashtagRatio(campaignUser)$

$calculateUserMentionRatio(campaignUser)$

$calculateUserMediaRatio(campaignUser)$

$calculateUserDailyAvaragePostCount(campaignUser)$

$findAverageDailyHashtagTweetCount(campaignUser, tracedHashtag)$

$findUserDailyPostRatio(campaignUser)$

$extractUserRegistrationDate(campaignUser)$

end for

return

Figure 5.9. User Features Extraction Algorithm.

Extracted features in this phase are given below. Moreover, the algorithm of hourly feature extraction is given in Figure 5.10 with the explanations of function in Table 5.9 (also see Figure 5.8).

Table 5.9. Description of functions in Hourly Features Extraction Algorithm (Figure 5.10).

Use Parameters	
$tweetsInHourlyInterval$	
Function	**Description**
calculateHourlyGeneralFeatures	calculates features illustrated in Section 5.6.2.1 in hourly basis
calculateHourlyRetweetFeatures	calculates features illustrated in Section 5.6.2.3 in hourly basis
calculateHourlyNonRetweetFeatures	calculates features illustrated in Section 5.6.2.4 in hourly basis
calculateHourlyMentionFeatures	calculates features illustrated in Section 5.6.2.5 in hourly basis
calculateHourlyRetweetedMentionFeatures	calculates features illustrated in Section 5.6.2.6 in hourly basis
calculateHourlyNonRetweetedMentionFeatures	calculates features illustrated in Section 5.6.2.7 in hourly basis
calculateHourlySentimentPercentagesInReplies	calculates features illustrated in Section 5.6.2.8 in hourly basis
calculateHourlyTweetEntityFeatures	calculates features illustrated in Section 5.6.2.2 in hourly basis

5.5. Classification

All user features and hourly features of all analyzed campaigns are obtained to create a training data set. During creation of training data set, resampling techniques are applied in order to increment training data size. (see Section 3.4.1)

Require *tweetsInHourlyInterval*
calculateHourlyGeneralFeatures(*tweetsInHourInterval*)
calculateHourlyRetweetFeatures(*tweetsInHourInterval*)
calculateHourlyNonRetweetFeatures(*tweetsInHourInterval*)
calculateHourlyMentionFeatures(*tweetsInHourInterval*)
calculateHourlyRetweetedMentionFeatures(*tweetsInHourInterval*)
calculateHourlyNonRetweetedMentionFeatures(*tweetsInHourInterval*)
calculateHourlySentimentPercentagesInReplies(*tweetsInHourInterval*)
calculateHourlyTweetEntityFeatures(*tweetsInHourInterval*)
return

Figure 5.10. Hourly Features Extraction Algorithm.

After the training data is formed, a classification model is generated by applying Supervised Learning Algorithms (see Section 3.4.2) with usage of 10 fold cross validation (see Section 3.4.1.1).

5.6. Features

In Section 4.2, features and patterns that can be extracted for detection of organized activities are stated. However, the features related to "Closeness Centrality", "Information Flow" and "Posting Tweets About Same / Similar Topics in Same Time of the Day" are not extracted due to performance issues. Furthermore, during experiments below extracted features are used for creating a classifier model which enables detection of organized activities with high accuracy. In Chapter 8, more information can be found about "not extracted features", which are considered in Approach phase (see Section 4.2).

5.6.1. User Features

Note that, in below explanations, *user′sanalyzedtweets* refer to tweets of user which are posted in *userTweetAnalyzeDuration*.

5.6.1.1. Extracted User Features. User features are helpful to illustrate bot and cyborg account characteristics [44]. Since in organized behaviors, bot and highly automated accounts are used (2016 U.S. Presidential Election, [5]), user features are also used during training. This section describes extracted user features.

hashtagPostCount : Remember that, each campaign is created for grouping tweets of $tracedHashtag$. In this feature, count of user's tweets, containing hashtag and posted in $userTweetAnalyzeDuration$ is found, because usage of same hashtag a lot might indicate a premeditated tweeting behavior.

$favoriteCount$: The number of tweets that are favorited by user. This feature is used, because for automated accounts, it is not more likely to favorite a tweet.

postCount : The number of tweets posted by the user. This feature is used, because tweet counts of organized users and normal users might be different and this may help to differentiate organized and organic behaviors.

$friendFollowerRatio$: $\frac{followerCount}{(followerCount+friendCount)}$. $followerCount$ refers to user's follower count while $friendCount$ refers to user's friend count. This may be helpful, since in automated and organized accounts, the aim is to propagate a message, they might not be interested for following other users or vice versa.

hashtagRatio : $\frac{usedHashtagsCount}{allUserTweetsCount}$. $allUserTweetsCount$ refers to count of user's analyzed tweets. $usedHashtagsCount$ refers to the number of the user's analyzed tweets. This may be helpful, since in automated and organized accounts the aim is to propagate a message, they may use hashtag more than organic users .

mentionRatio : $\frac{usedMentionCount}{allUserTweetsCount}$. $allUserTweetsCount$ refers to count of user's analyzed tweets and $usedMentionCount$ refers to count of used mentions in analyzed tweets.

$\underline{urlRatio:}$ $\frac{usedUrlCount}{allUserTweetsCount}$. $allUserTweetsCount$ refers to count of user's analyzed tweets and $usedUrlCount$ refers to count of used URLs in $user'sanalyzedtweets$. This may be helpful, since in automated and organized accounts the aim is to propagate a message, they may use URL more than organic users

$\underline{mediaRatio:}$ $\frac{usedMediaCount}{allUserTweetsCount}$. $allUserTweetsCount$ refers to count of $user'sanalyzedtweets$ while $usedMediaCount$ refers to count of all used videos and images in $user'sanalyzedtweets$. This may be helpful, since in automated and organized accounts the aim is to propagate a message, they may use media entities like image and video more than organic users

$\underline{averageDailyPostCount:}$ $\frac{postCount}{userActiveDayCount}$. $postCount$ refers to count of user's all tweets posted in Twitter while $userActiveDayCount$ refers to count of user's days until its registration to Twitter. This feature is used, because organized users have a goal and they need to propagate their message. In order to do that they may need to post more tweets than normal users.

$\underline{averageDailyPostCountInHashtagDays:}$ $\frac{postCountsInHashtagDays}{countDaysTracedHashtagUsed}$. $postCountsInHashtagDays$ refers to how many tweets are posted in days which user used $tracedHashtag$ of campaign. Furthermore, $countDaysTracedHashtagUsed$ refers to how many days user posted tweets using $tracedHashtag$.

$\underline{averageDailyPostCountRatio:}$ $\frac{averageDailyPostCountInHashtagDays}{averageDailyPostCount}$.

$\underline{userCrationDate:}$ User's Twitter registration date. If a remarkable amount of user accounts in a tweet dataset are created in close dates to each other, it can be an indicator of an organized movement. Because, this may show an organized bulk user creation in close dates.

5.6.2. Extracted Hourly Features

Hourly features might reflect organic and organized posting behaviors in a given tweet set. Therefore, hourly features are also used during training.

In an organized behavior, a user might post multiple tweets in order to propagate a message. Also, users might post messages in a coordinated way to persist and propagate their message as much as possible. Because of this reason,hourly distinct user and tweet counts are calculated and used during training.

5.6.2.1. General Hourly Features. Genaral hourly features are explained as below.

$hourlyDistinctUserCount$: Count of distinct users who posted tweets in given $hourInterval$.

$hourlyTweetCount$: Count of tweets in given $hourInterval$.

$hourlyTweetUserRatio$: $\frac{hourlyTweetCount}{hourlyDistinctUserCount}$

5.6.2.2. Hourly Tweet Entity Features. Use of URLs, hashtags, images, videos, and mentions are more likely to occur in organized activities, since it is possible to explain more things by using them. Therefore, hourly ratios of them might be indicator of an organized behavior existence in a given tweet set.

$hourlyHashtagRatio$: $\frac{hourlyUsedHashtagCounts}{hourlyDistinctUserCount}$. Here $hourlyUsedHashtagCounts$ refers to total count of all used hashtags in tweets posted in given $hourInterval$.

$hourlyUrlRatio$: $\frac{hourlyUsedUrlCounts}{hourlyDistinctUserCount}$. Here $hourlyUsedUrlCounts$ refers to total count of all used URLs in tweets posted in given $hourInterval$.

$\underline{hourlyMentionRatio}$: $\frac{hourlyUsedMentionCounts}{hourlyDistinctUserCount}$. Here $hourlyUsedMentionCounts$ refers to total count of all used mentions in tweets posted in given $hourInterval$.

$\underline{hourlyMediaRatio}$: $\frac{hourlyUsedMediaCounts}{hourlyDistinctUserCount}$. Here $hourlyUsedMediaCounts$ refers to total count of all used images and videos in tweets posted in given $hourInterval$.

5.6.2.3. Hourly Retweet Features. Since "Adoption of Sharing" is a characteristic of organized behaviors, it is more likely that *"retweets"* are used more often in organized activities. Therefore, hourly retweet features are also extracted.

$\underline{hourlyRetweetCount}$: Count of retweets in given $hourInterval$.

$\underline{hourlyDistinctRetweetCount}$: Count of distinct retweets in given $hourInterval$.

$\underline{hourlyDistinctRetweetUserCount}$: Count of distinct users who retweeted in given $hourInterval$.

$\underline{hourlyRetweetUserRatio}$: $\frac{hourlyRetweetCount}{hourlyDistinctRetweetUserCount}$

$\underline{hourlyDistinctRetweetUserRatio}$: $\frac{hourlyDistinctRetweetCount}{hourlyDistinctRetweetUserCount}$

$\underline{hourlyRetweetRatio}$: $\frac{hourlyRetweetCount}{hourlyTweetCount}$

$\underline{hourlyDistinctRetweetTotalTweetCountRatio}$: $\frac{hourlyDistinctRetweetCount}{hourlyTweetCount}$

$\underline{hourlyDistinctRetweetRatio}$: $\frac{hourlyRetweetRatio}{hourlyDistinctRetweetTotalTweetCountRatio}$

$\underline{hourlyDistinctRetweetUserTotalUserRatio}$: $\frac{hourlyDistinctRetweetUserCount}{hourlyDistinctUserCount}$

5.6.2.4. Hourly Non-Retweet Features. In contrast to "retweets", non-retweet tweet counts and distinct users might be indicator of organic activities in a tweet set.

$hourlyNonRetweetCount$: Count of non-retweeted tweets in given $hourInterval$.

$hourlyDistinctNonRetweetUserCount$: Count of distinct users who posted non-retweeted tweets in given $hourInterval$.

$hourlyNonRetweetUserRatio$: $\frac{hourlyNonRetweetCount}{hourlyDistinctNonRetweetUserCount}$

$hourlyNonRetweetTotalTweetCountRatio$: $\frac{hourlyNonRetweetCount}{hourlyTweetCount}$

5.6.2.5. Hourly Mention Features. Compared to entities of URL, hashtag and media, mention entities do not have information about content. They are used for replying to a message, informing and indicating a user. Since there is coordination between users in organized behavior, it is more likely to see *"mentions"* in them. Therefore, hourly mention features are also used during training.

$hourlyTotalMentionCount$: Count of used mentions in tweets posted in given $hourInterval$.

$hourlyDistinctMentionCount$: Count of distinct mentions in tweets posted in given $hourInterval$.

$hourlyTotalDistinctMentionRatio$: $\frac{hourlyDistinctMentionCount}{hourlyTotalMentionCount}$

5.6.2.6. Hourly Retweeted Mention Features. As it is explained earlier, retweets and mentions can be indicator of organized behavior. Therefore, hourly retweeted mention features are also extracted. Hourly retweeted mention features are explained as below.

$hourlyRetweetedTotalMentionCount$: Count of used mentions in retweets posted in given $hourInterval$.

$\underline{hourlyRetweetedDistinctMentionCount}$: Count of distinct mentions in retweets posted in given $hourInterval$.

$\underline{hourlyRetweetedTotalDistinctMentionRatio}$:
$\frac{hourlyRetweetedDistinctMentionCount}{hourlyRetweetedTotalMentionCount}$

5.6.2.7. Hourly Non - Retweeted Mention Features. As it is explained earlier, non-retweets and mentions can be indicator of organized behavior. Therefore, hourly non-retweeted mention features are also extracted. Hourly non retweeted mention features are explained as below.

$\underline{hourlyNonRetweetedTotalMentionCount}$: Count of used mentions in non-retweet tweets posted in given $hourInterval$.

$\underline{hourlyNonRetweetedDistinctMentionCount}$: Count of distinct mentions in non-retweet tweets posted in given $hourInterval$.

$\underline{hourlyNonRetweetedTotalDistinctMentionRatio}$:
$\frac{hourlyNonRetweetedDistinctMentionCount}{hourlyNonRetweetedTotalMentionCount}$

5.6.2.8. Hourly Sentiment Features of Reply Tweets. In behaviors which aim to harm or humiliate a specific user, analyzing characteristics of the communities,which attacking and attacked user belongs to, can be helpful. In cases like this, reply networks between these communities can be used, if remarkable amount of reply tweets between those communities consist of Negative sentiments. In a condition like this, reply tweets with Negative Sentiment can guide to existence of an organized behavior. Therefore, hourly sentiment features of tweets are also used.

$\underline{hourlyVeryNegativeSentimentPercentageInReplyTweets}$: Percentage of reply tweets, whose sentiment is very negative, in all reply tweets posted in given $hourInterval$.

$\underline{hourlyNegativeSentimentPercentageInReplyTweets}$: Percentage of reply tweets, whose sentiment is negative, in all reply tweets posted in given $hourInterval$

$\underline{hourlyNeutralSentimentPercentageInReplyTweets}$: Percentage of reply tweets, whose sentiment is neutral, in all reply tweets posted in given $hourInterval$

$\underline{hourlyPositiveSentimentPercentageInReplyTweets}$: Percentage of reply tweets, whose sentiment is positive, in all reply tweets posted in given $hourInterval$

$\underline{hourlyVeryPositiveSentimentPercentageInReplyTweets}$: Percentage of reply tweets, whose sentiment is very positive, in all reply tweets posted in given $hourInterval$

6. PROTOTYPE IMPLEMENTATION

High level architecture of implemented system is as given in Figure 6.1.

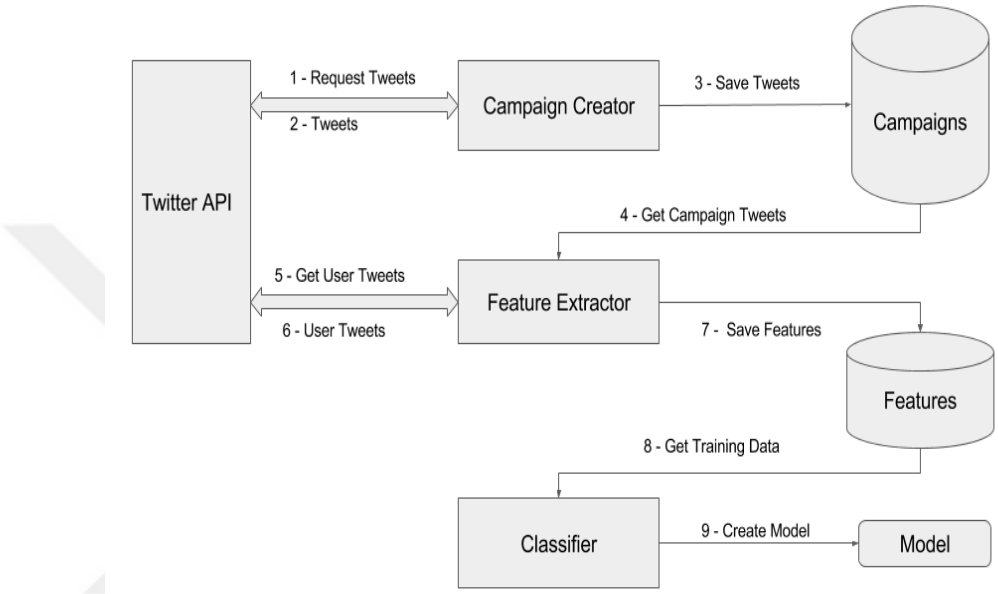


Figure 6.1. System Architecture.

6.1. Twitter Connection Pool Implementation

In this thesis,Twitter Search API and Twitter User Timeline API are used for tweet retrieval (Section 3.1.3). However, in order to prevent excessive usage of these rest APIs, Twitter allows only a certain amount of requests within a time interval of 15 minutes. For Search API, the request limit is 180 while for User Timeline API the limit is 900.

This limitation causes huge delays in both campaign creation and feature extractor phases. To give an example, during this thesis, more than 200 million tweets are collected by User Time Line API.

If those tweets were collected without any interruption by obeying the Twitter limits, collection of these tweets would be completed in 63 month.

In order to overcome this problem, a trivial Twitter Connection pool is implemented allowing round robin usage of Twitter API user accounts in API requests. By this way, the implemented application remains making requests to Twitter API just by switching to another user when the user's request limit is exceeded. Theoretically, the implemented connection pool has no upper limit for user account counts. However, in this thesis, implemented connection pool worked with 9 user accounts in order to prevent possible I/O bottleneck in MongoDB.

As a result of this trivial Twitter Connection Pool, more than 200 million tweets were collected by User Time Line API in a 5 month time interval, which also includes the spent times in feature extraction processes, rather than just collecting in 63 months.

6.2. Campaign Creation Module

During Campaign Creation tweets are retrieved from Twitter API and after campaign's statistics are calculated those tweets are saved into the MongoDB collection for usage of Feature Extractor module (see Figure 5.4 and Figure 5.5). In this part, implementation of these phases will be explained in detail.

In the implemented system, a campaign is created by giving parameters from user interface, which is depicted in Figure 6.2. Descriptions of required parameters for the user interface are explained in Table 6.1.

Later on, after the campaign creation request is made, campaign tweets are collected from Twitter and then statistics of them are calculated and saved into MongoDB collection. An example MongoDB document of campaign creation request is given in Figure 6.7.

6.2.1. Front End

The implemented system provides user interfaces in order to create and analyze the campaigns. These user interfaces can be listed as

(i) User interface for creating a campaign (see Figure 6.2 and Table 6.1)
(ii) User interface for displaying all created campaigns (see Figure 6.3)
(iii) User interface for illustrating campaign statistics
(iv) User interface for illustrating percentages of used hashtags considering all hashtag usages in the campaign (see Figure 6.4)
(v) User interface for displaying percentages of used words considering all word usages in the campaign (see Figure 6.5)

6.2.2. Tweet Retrieval using Twitter Search API

In the implemented system, Twitter Search API is used for collecting tweets which use the given hashtags. Search API request parameters are illustrated in Table 6.2 and with an example request given in Figure 6.6. Recall that, Search API only returns tweets posted in last 7 days.

As it is shown in Figure 6.2, in the campaign creation request, tweets of a campaign are requested by entering minimum date and maximum date values (see Table 6.1). However, Twitter Search API only allows the usage of max date ("until" parameter). Because of this reason, "min date" control is done in the backend for each retrieved tweets from Search API and this results poorly in tweet retrieval performance.

Table 6.1. Description of parameters In Campaign Creation User Interface.

Parameter	**Description**
UserId	user id for authentication of request
Password	user password for authentication of request
CampaignId	id of campaign which corresponds to primary key of campaign The campaign tweets are saved with this campaign id value.
CampaignDefinition	It is a descriptive field. Description of campaign is requested
Hashtag	hashtag(s) whose tweets will be retrieved from Twitter
MaxDate	max post date of hashtag tweets. Tweets posted after this time is not retrieved.
MinDate	min post date of hashtag tweets. Tweets posted before this time is not saved.

6.3. Feature Extraction Module

Recall that, each feature extraction is done as a result of analyzing only one campaign and only one hashtag in the given campaign. As it is explained in Figure 5.8, during feature extraction following steps are executed.

(i) Get campaign tweets according to given $campaignId$ and $tracedHashtag$
(ii) Extract users who have tweets with $tracedHashtag$ in the given campaign
(iii) For each extracted user collect tweets of them which are posted between dates of one week before and one week later than the date of tweet in the campaign.
(iv) Extract features of each extracted user (Section 5.6.1.1)
(v) Extract hourly features (Section 5.6.2)
(vi) Calculate statistical values of all collected campaigns

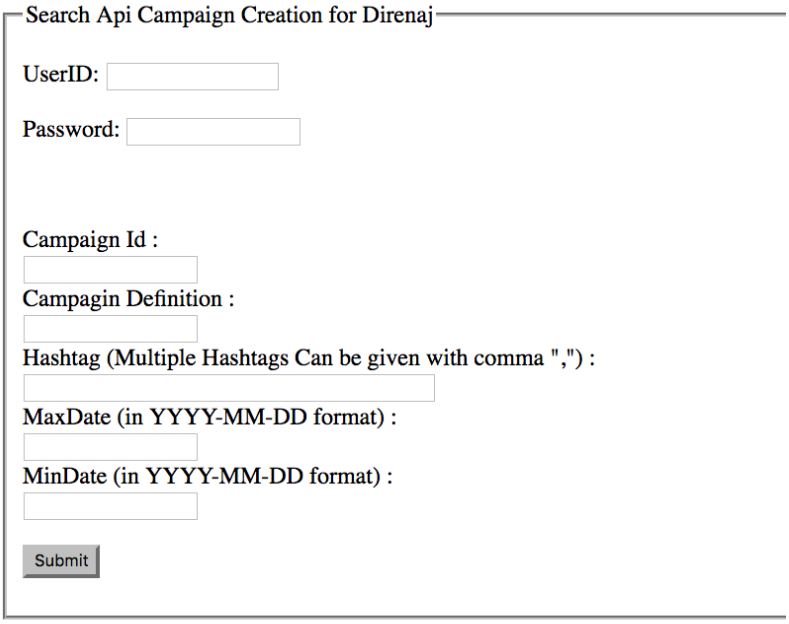


Figure 6.2. Campaign Creation Interface.

As it is stated above, during feature extraction, another tweet collection process is done in addition to tweet retrieval process during Campaign Creation. The tweet collection process executed in the Feature Extraction phase enables analyzing much more tweets than collected tweets during the campaign creation. In below Table 6.3, collected tweet counts for each campaign is given. In Table 6.3, it is seen that there is a huge difference between count of campaign tweets and count of all users' tweets which are posted between times of one week before and one week later than the post date of tweet collected in the campaign.

6.3.1. Tweet Retrieval

All user tweets are collected by using Twitter User Timeline API, which returns a collection of the most recent tweets for a given user. The API returns the tweets that are seen in the user's profile page (see Section 3.1.3). To fetch tweets the parameters shown in Table 6.4 are used.

Toolkit Campaigns

Show 50 entries Search:

Campaign Id	Campaign Type	Campaign Definition	Campaign Query Terms	Campaign Creation Time	Campaign Creation Ended
May1	Search Api	May1 & Date Between : 2017-05-02 & 2017-04-30	#WorkersDay,#LabourDay,#MayDay2017,#MayDay	736818.5785699075	true
ThingsMoreTrustedThanTrump	Search Api	ThingsMoreTrustedThanTrump & Date Between : 2017-04-14 & 2017-04-09	#ThingsMoreTrustedThanTrump	736801.5526828588	true
NationalPetDay	Search Api	NationalPetDay & Date Between : 2017-04-14 & 2017-04-09	#NationalPetDay	736801.552325845	true
NationalSiblingsDay	Search Api	NationalSiblingsDay & Date Between : 2017-04-14 & 2017-04-09	#NationalSiblingsDay	736801.5519746875	true
unitedairlines	Search Api	unitedairlines & Date Between : 2017-04-14 & 2017-04-09	#unitedairlines	736801.5513599307	true
NewUnitedAirlinesMottos	Search Api	NewUnitedAirlinesMottos & Date Between : 2017-04-14 & 2017-04-09	#NewUnitedAirlinesMottos	736801.4543359954	true
saturdayMorning_20170319	Search Api	saturdayMorning & Date Between : 2017-03-19 & 2017-03-16	#saturdayMorning	736773.8174588889	true
sundayMorning_20170319	Search Api	sundayMorning & Date Between : 2017-03-19 & 2017-03-17	#sundayMorning	736773.8171829747	true
DeepLearning	Search Api	DeepLearning_Organic & Date Between : 2017-03-19 & 2017-03-12	#deeplearning	736773.8158633218	true
IoT	Search Api	IoT_Organic & Date Between : 2017-03-19 & 2017-03-12	#IoT	736773.8156229977	true

Figure 6.3. User interface for displaying all campaigns.

6.3.1.1. Retrieval of Users' Tweets. As it is also stated above, for each user in the campaign, users' tweets are collected by Twitter User Timeline API. It is possible to separate these collected tweets of users to below categories.

(i) User's tweets posted in one week earlier than the user's tweet in the campaign
(ii) User's tweets posted in one week later than the user's tweet in the campaign

For the retrieving of earlier tweets, *max_id* is set to *id* of user's tweet in the campaign along with the *user_id*. By this way, earlier tweets are collected until User Timeline API returns tweets earlier than one week.

The same approach is also followed for retrieval of tweets posted in a week later than the tweet in the campaign. At this time, *since_id* is set to *id* of user's tweet in the campaign along with the *user_id*. However, User Timeline API behaves differently in case of *max_id* is not set. When *max_id* is not set, User Timeline API starts to return user's tweets from the user's latest post.

Table 6.2. Description of Twitter Search API parameters.

Parameter	**Description**
q	A UTF-8, URL-encoded search query of 500 characters maximum, including operators. in $allHashtagTweets$
until	Returns tweets created before the given date. Date should be formatted as YYYY-MM-DD.

This can be a problem in case of used campaign belongs to a date which is a long time ago. In a case like this, although the analysis will be done only to tweets posted after in a week later than the tweet in the campaign, User Timeline API returns all of the tweets until the tweet in the campaign.

In order to prevent this, the implemented system searches all tweets in the MongoDB, regardless of their campaign and user, and then gets the *id* of a tweet which is posted one week later than the campaign tweet.

As a result of this, the found tweet id is set to max_id in User Timeline API request. By this way, the Feature Extraction Module completes to retrieval of all users' tweets in a shorter time comparing to retrieval of tweets without setting max_id parameter.

Note that, Twitter assigns Tweet ids incrementally in time basis without taking into account of users. Because of this reason, the id of any tweet, which is posted one week later than the campaign tweet, can be used for max_id parameter in User Timeline API request. Also, since the implemented system has saved more than 200 million tweets for eight months, it is almost possible that system finds a tweet posted in any time of this eight months.

Table 6.3. Campaign rough tweet and user count examples.

Traced Hashtag	**Campaign Tweet Count**	**User Count**	**Collected Tweet Count**
#NeverHillary	44.076	20.626	2.593.598
#StrongerTogether	13.581	8.294	876.343
#NationalPetDay	46.816	39.878	10.571.845
#NationalSiblingsDay	66.359	59.466	15.524.289
#unitedairlines	54.506	42.631	11.837.051
#NewUnitedAirlinesMottos	21.400	13.132	3.992.044
#BigData	10.398	4.348	933.814
#BoycottHamilton	56.523	36.693	2.164.038
#clintoncrimefoundation	800	735	88.341
#benghazi	10.876	7908	1.521.139

Furthermore, another important aspect that should be considered during Retrieval of Users' Tweets is to not saving a saved tweet again. If this case is not taken into consideration, disk consumption of system will increase and most probably the system will be out of free disk space. In order to overcome this problem, during collecting of user's tweets, for each response retrieved from User Timeline API, the latest and earliest dates of tweets are retrieved and a query is sent to database for user to check whether or not the system has user's tweets between these time intervals. If a tweet of user exist, system does not save retrieved tweets and check for other tweets of user from User Timeline API.

Table 6.4. User Timeline API parameters

Parameter	Description
user_id	The ID of the user for whom to return results for.
max_id	Returns results with an ID less than (that is, older than) or equal to the specified ID.
since_id	Returns results with an ID greater than (that is, more recent than) the specified ID.

6.3.2. Implemented User Interfaces

Implemented module has user interfaces as below.

(i) User interface for requesting analysis for a hashtag in a given campaign (see Figure 6.9)
(ii) User interface for displaying all analysis requests with their results and statuses (see Figure 6.10)
(iii) User interface for displaying extracted features of users in an analysis request (see Figure 6.11)
(iv) User interface for displaying extracted hourly features in an analysis request
(v) User interface for visualizing all extracted features in an analysis request

6.3.3. Breakpoints

In the implemented system, Feature Extraction Module execution time, for a hashtag in a campaign, change from one hour to one week. For example, analysis of #NationalSiblingsDay hashtag in campaign "NationalSiblingsDay" is completed in one week. While Feature Extraction Module execution can take that much time, having an exception at the end of the process might cause to execution of whole one week process from the beginning. This problem needed to be handled since in the implemented system exceptions occur from time to time, as it may happen in all systems.

In order to handle this problem, a breakpoint mechanism is implemented in the Feature Extraction Module. After the implementation of breakpoint mechanism, even if the exception is thrown during the execution, it become possible to execute module from the closest breakpoint in the execution flow. This mechanism prevented to run whole process again from the beginning by allowing to resume process from the closest breakpoint. Used breakpoints in the system is explained below.

6.3.3.1. TWEET_COLLECTION_COMPLETED. Pinned breakpoint after all users' tweets are collected by User Timeline API.

6.3.3.2. USER_ANALYZE_COMPLETED. Pinned breakpoint after all users' features are extracted.

6.3.3.3. COS_SIMILARITY_INIT. Pinned breakpoint after cosine similarity calculation of tweets are started for hourly intervals.

6.3.3.4. TF_CALCULATION_COMPLETED. Pinned breakpoint after tf values are calculated for cosine similarity calculation.

6.3.3.5. IDF_CALCULATION_COMPLETED. Pinned breakpoint after idf values are calculated for cosine similarity calculation.

6.3.3.6. TF_IDF_CALCULATION_COMPLETED. Pinned breakpoint after tf, idf values are calculated for cosine similarity calculation.

6.3.3.7. SIMILARTY_CALCULATED. Pinned breakpoint after cosine similarity of tweets are calculated.

6.3.3.8. ALL_HOURLY_CALCULATION_DONE. Pinned breakpoint after all hourly features are extracted.

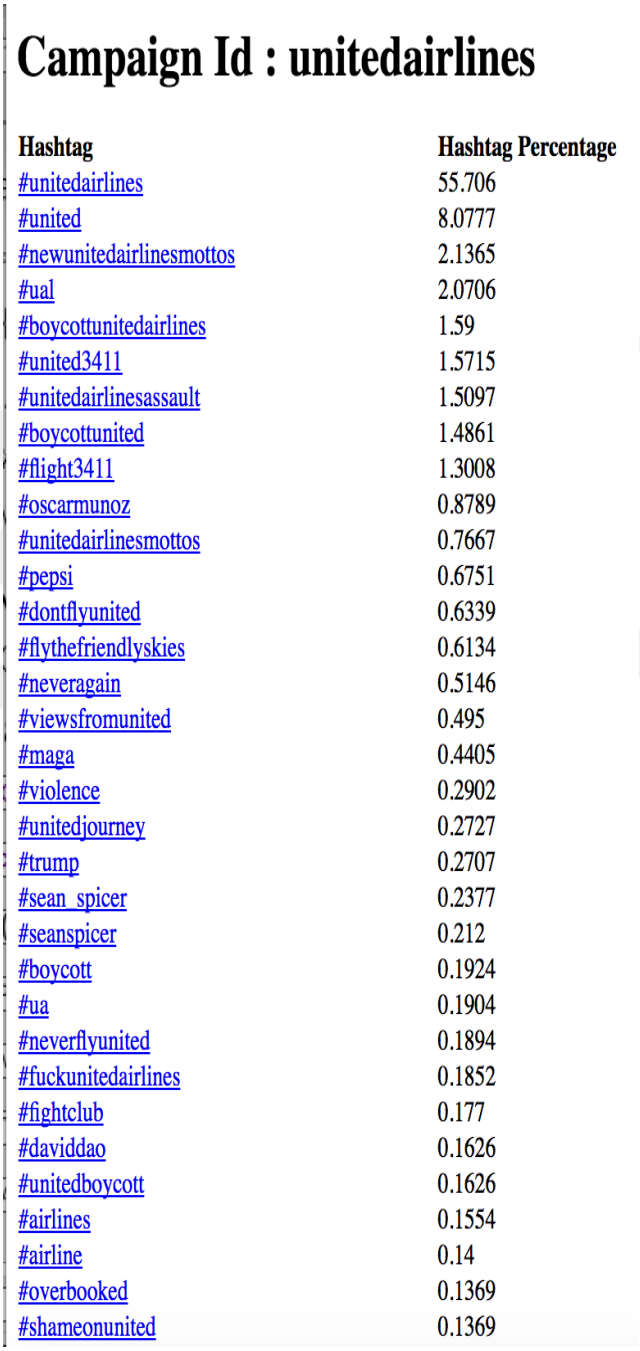
6.3.3.9. GENERAL_STATISTIC_CALCULATION_DONE. Pinned breakpoint after general statistics values are calculated.

6.3.3.10. STATISTIC_CALCULATED. Pinned breakpoint after statistics of all values are calculated.

6.3.4. Campaign Analysis

Each row, which is illustrated in Figure 6.10, corresponds to result of a hashtag analysis in the given campaign. If a campaign has multiple hashtags, which will be analyzed, an analysis can be requested for each hashtag in the campaign.

In the implemented system, an analysis can be requested by using the user interface illustrated in Figure 6.9. In Tables 6.5, 6.6,6.7 and 6.8, parameters of the interface are explained.



Campaign Id : unitedairlines

Hashtag	Hashtag Percentage
#unitedairlines	55.706
#united	8.0777
#newunitedairlinesmottos	2.1365
#ual	2.0706
#boycottunitedairlines	1.59
#united3411	1.5715
#unitedairlinesassault	1.5097
#boycottunited	1.4861
#flight3411	1.3008
#oscarmunoz	0.8789
#unitedairlinesmottos	0.7667
#pepsi	0.6751
#dontflyunited	0.6339
#flythefriendlyskies	0.6134
#neveragain	0.5146
#viewsfromunited	0.495
#maga	0.4405
#violence	0.2902
#unitedjourney	0.2727
#trump	0.2707
#sean_spicer	0.2377
#seanspicer	0.212
#boycott	0.1924
#ua	0.1904
#neverflyunited	0.1894
#fuckunitedairlines	0.1852
#fightclub	0.177
#daviddao	0.1626
#unitedboycott	0.1626
#airlines	0.1554
#airline	0.14
#overbooked	0.1369
#shameonunited	0.1369

Figure 6.4. User interface for displaying percentages of hashtag in campaign.

Campaign Id : unitedairlines

Word	Word Frequency	Word Usage Percentage I Campaign
#unitedairlines	53817.0	%11.6894
...	15352.0	%3.3346
@united	11137.0	%2.419
united	9734.0	%2.1143
#united	7838.0	%1.7025
airlines	7687.0	%1.6697
flight	3829.0	%0.8317
overbooked	3568.0	%0.775
passenger	3318.0	%0.7207
video	3001.0	%0.6518
fly	2885.0	%0.6266
training	2530.0	%0.5495
plane	2217.0	%0.4815
dragged	2078.0	%0.4514
#newunitedairlinesmottos	2076.0	%0.4509
leaked	2035.0	%0.442
@altflyunited	2019.0	%0.4385
#ual	2012.0	%0.437
press	1989.0	%0.432
https://t_co/6innqd0eja	1931.0	%0.4194
passengers	1874.0	%0.407
timing	1719.0	%0.3734
dear	1671.0	%0.363
lawsuit	1652.0	%0.3588
guy	1554.0	%0.3375
#boycottunitedairlines	1543.0	%0.3351
#united3411	1527.0	%0.3317
unfortunate	1519.0	%0.3299
@nihalf	1504.0	%0.3267
https://t_co/lzdlcxtfxf	1500.0	%0.3258
doctor	1483.0	%0.3221
#unitedairlinesassault	1466.0	%0.3184

Figure 6.5. User interface for displaying percentages of campaign words.

```
https://api.twitter.com/1.1/search/tweets.json?q=%23NeverHillary&until=2017-04-28
```

Figure 6.6. Example Twitter Search API request for tweets posted before 2017-04-28 and use the #NeverHillary hashtag.

```
{
  "_id": ObjectId("583bebafe4b08eac9489737c"),
  "campaign_type": "Search Api",
  "description": "Thanksgiving & Date Between : 2016-11-28 & 2016-11-20",
  "created_at": 736662.3561091551,
  "campaign_id": "Thanksgiving",
  "query_terms": "#TurkeyIn4Words,#SurvivalGuideToThanksgiving",
  "tweetCollectionEnded": true,
  "minCampaignDate": "Nov 20, 2016 12:00:00 AM",
  "maxCampaignDate": "Nov 28, 2016 11:59:00 PM"
}
```


Figure 6.7. Example campaign creation request.


```
> db.CampaignStatistics.findOne({"campaign_id":"unitedairlines"})
{
    "_id" : ObjectId("58f3a16fe4b04fe325f029d6"),
    "campaign_id" : "unitedairlines",
    "totalTweetCount" : 54506,
    "retweetedTweetCount" : 38591,
    "campaignRetweetPercentage" : 70.8,
    "replyTweetCount" : 2269,
    "campaignReplyPercentage" : 4.16,
    "mentionTweetCount" : 43182,
    "campaignMentionPercentage" : 79.22,
    "distinctUserCount" : 42631,
    "campaignTweetCountPerUser" : 1.28,
    "totalWordCount" : 460391,
    "totalDistinctWordCount" : 37452,
    "campaignDistinctWordCountPercentage" : 8.13,
    "hashtagVariance" : 0.7544,
    "campaignHastagStandardDeviation" : 0.8686,
    "distinctRetweetUserCount" : 30636,
    "distinctRetweetUserCountPercentage" : 71.8632
}
```


Figure 6.8. Campaign statistic DB object.

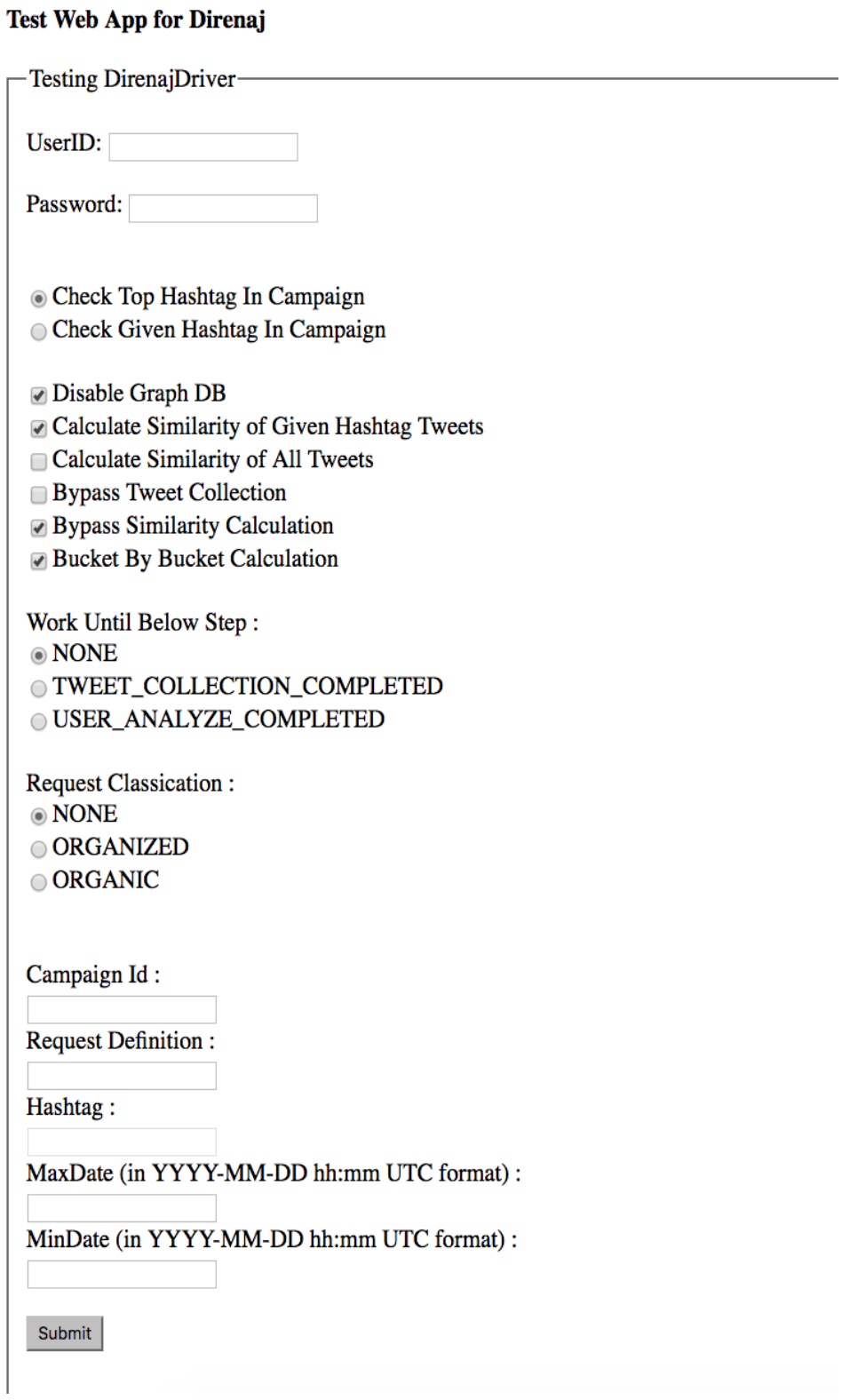


Figure 6.9. Campaign analysis request interface.

Request Id	User Features	Hourly Features	Visual Analysis	Request Definition	Campaign Id	Traced Hashtag	Earliest Tweet Time	Latest Tweet Time	Process Completed	Request Status	Organized Class
20170131161816323 8af0b865-675c-49bc-823f-caf6cd56d00a	User Features	Hourly Features	Visual Analysis	boycotthamilton	BoycottHamilton	boycotthamilton	Sun Nov 13 00:01:32 EET 2016	Tue Nov 29 01:58:13 EET 2016	true	FINISHED	ORGANIZED
20170131161720265a2d8d274-1d42-4086-8127-0cc0a526a495	User Features	Hourly Features	Visual Analysis	imwithher	HillarySupportersElectionDay	imwithher	Thu Oct 27 23:56:13 EEST 2016	Thu Nov 17 01:57:14 EET 2016	true	FINISHED	ORGANIZED
20170130234152643 3f7cf979-8a7d-4a4a-9637-0dd40965 4527	User Features	Hourly Features	Visual Analysis	strongertogether	HillarySupportersElectionDay	strongertogether	Thu Oct 27 23:56:13 EEST 2016	Thu Nov 17 01:57:14 EET 2016	true	FINISHED	ORGANIZED
20170130234123409 6113034-7fe9-4a72-9615-4a805f7be857	User Features	Hourly Features	Visual Analysis	americafirst	HillarySupportersElectionDay	americafirst	Fri Oct 28 00:06:29 EEST 2016	Thu Nov 17 01:35:45 EET 2016	true	FINISHED	ORGANIZED

Figure 6.10. Interface for listing all campaign analysis.

Request Id	User Id	User Screen Name	Hashtag Post Count	Favorite Count	Status Count	Friend Follower Ratio	Hashtag Ratio	Mention Ratio	Url Ratio	Media Ratio	Avarage Daily Post Count	Avarage Daily Post Count In Hashtag Days	Hashtag_Daily_Post_Count_Avarage_Day_Post_Count_Ratio
20170505084347955 7073dbdb-7031-4bea-b1dd-b55501b25d6c	1000417411	JAJJA_Naeem	1	102	6428	0.44	1.32	0.14	0.55	0.09	4	3	0.75
20170505084347955 7073dbdb-7031-4bea-b1dd-b55501b25d6c	100436430	cyrusnahni	1	479	5216	0.58	0.55	1.2	0.05	0.3	1.9441	2	1.029
20170505084347955 7073dbdb-7031-4bea-b1dd-b55501b25d6c	10255262	ReinH	2	5777	44730	0.81	0.07	1.09	0.22	0.15	12.9315	119	9.202
20170505084347955 7073dbdb-7031-4bea-b1dd-b55501b25d6c	102919704	etukenyen	1	324	6789	0.48	0.73	0.67	0.37	0.05	2.5389	24	9.453
20170505084347955 7073dbdb-7031-4bea-b1dd-b55501b25d6c	1045846807	InstaKwara	3	26	1364	0.63	0.51	1.37	0.1	0.06	0.8595	6.5	7.563

Figure 6.11. Interface for displaying extracted user features during campaign analysis.

Table 6.5. Campaign analysis request parameters.

Parameter	**Description**
UserId	Id of registered user in the system. This parameter is requested for authentication of user
Password	Password of registered user in the system. This parameter is requested for authentication of user
Check Top Hashtag In Campaign	If this option is selected, the most used hashtag in the campaign that will be analyzed
Check Given Hashtag In Campaign	If this option is selected, the given hashtag in the campaign will be analyzed
Disable Graph DB	For calculation of closeness centrality, Neo4j graph db is used in the system. Selection of this option, disables closeness centrality calculation.
Calculate Similarity of Given Hashtag Tweets	If this option is selected, similarities between all users' tweets, which contains the analyzed hashtag, is calculated in hourly interval
Calculate Similarity of All Tweets	If this option is selected, similarities between all users' tweets is calculated in hourly interval

Table 6.6. Campaign analysis request parameters (cont).

Parameter	**Description**
Bypass Tweet Collection	If this option is selected, users' tweets will not be collected by Twitter User Timeline API (see Section 6.3.1.1). This option is selected in order to save time if it is known that users' tweets are collected in different campaign analysis or in analysis of different hashtag in the same campaign.
Bypass Similarity Calculation	If this option is selected, similarity calculation is not executed for tweets.
Bucket By Bucket Calculation	If this option is selected, tweets in campaign are analyzed in buckets of 1000 tweets, which means each row in Figure 6.10 corresponds to analysis of a bucket in campaign. For example, if this option is selected for a campaign which have 10.000 tweets for the given hashtag, 10 rows will occur in Figure 6.10.

Table 6.7. Campaign analysis request parameters (cont).

Parameter	Value	Description
Campaign Id	String	Campaign Id of the campaign of interest
Hashtag	A twitter hashtag	Hashtag which will be analyzed in campaign (if "Check Given Hashtag In Campaign" option is selected)
Request Definition	Textual Description	A description of the campaign of interest
Max Date (optional)	Date	Maximum post date of tweets which will be analyzed
Min Date (optional) **Work Until Below Step**	Date NONE	Minimum post date of tweets which will be analyzed If this option is selected, Feature Extraction Module will work until the end of the process
Work Until Below Step	TWEET_COLLECTION_COMPLETED	Feature Extraction Module will work until the breakpoint
Work Until Below Step	USER_ANALYZE_COMPLETED	USER_ANALYZE_COMPLETED Feature Extraction Module will work until the breakpoint

Table 6.8. Campaign analysis request parameters (cont).

Parameter	Value	Description
Request Classification	NONE	Extracted features for the requested analysis will not be used in the training data set
Request Classification	ORGANIZED	The extracted features for the requested analysis will marked as Organized in the training data set
Request Classification	ORGANIC	The extracted features of requested analysis will marked as Organic in the training data set

6.3.5. Performance

During this study, performance problems are also experienced. In this section, performance related studies are explained.

6.3.5.1. Parallelism. In order to improve the performance of the *Feature Extractor* module two parts of the flow are parallelized:

(i) during the retrieval of a user's tweets (see Section 6.3.1.1) and during the extraction of hourly features.
(ii) Parallelism is introduced via the "Cyclic Barrier" functionality in Java with two parallel threads.

For retrieving a user's tweets, two threads are created: one for collecting the user's tweets posted prior to the campaign and one for collecting the tweets posted after the campaign.During extraction of hourly features, two threads are executed in order to analyze two hourly interval at the same time.

6.3.5.2. Intermediary Tables. As it is mentioned above, the implemented system collected more than 165 million tweets of users and all of these tweets are stored in one collection in MongoDB. Even if indexes exist in the collection, performance during campaign analysis was inefficient in some cases because of the I/O bottleneck and querying a huge collection. During execution of Feature Extractor module, in the beginning these performance problems are experienced a lot especially during extraction of hourly features.

The reason of this poor performance is that MongoDB does not allow usage of indexes on multiple fields of array. However, during extraction of hourly features, used queries are based on two array fields which are campaign_id and hashtag fields.

"campaign_id" field is an array field, since the system does not save a user's tweets twice, if it is retrieved in analysis of another campaign (see Section 6.3.1.1). In a case like this, the id of latest analyzed campaign is added to "campaign_id" field of the document.

Also, "hashtag" field is an array field, since Twitter's API response for hashtag entities is in array field as a result of multiple hashtags can be used in a tweet. As a result of this, in querying "OrgBehaviourUserTweets" MongoDB collection, which holds all users' collected tweets, MongoDB could not use index on both fields and these result in a very poor query performance. For instance, in the early phases of implementation, during a campaign analysis, Feature Extractor Module could not finish its job even after one month because of the bottleneck and slow performance of queries in "OrgBehaviourUserTweets" MongoDB collection, which holds all users' collected tweets. In order to prevent this issue, an intermediary small collections with small documents are created. In these documents, for each analysis, the summary of processed tweets in this campaign is saved. Furthermore, after the Feature Extractor Module is finished, all records of the analysis in the intermediary collections are deleted. Note that, saving a document for each analysis solved the problem of array campaign_id field, since an analysis can be done only for a campaign in the implemented system.

Consequently, using these intermediary collections improved the query performances and lessened the I/O bottlenecks in "OrgBehaviourUserTweets" MongoDB collection. Moreover, the analysis, which Feature Extractor Module could not finish in a month in the beginning, was completed in three days.

6.3.6. Training Data Creation & Classification

The implemented modules for training data creation and classification are as below.

(i) Training Data Extractor Module
(ii) Model Generator

6.3.6.1. Training Data Extractor Module. In this module, all found features in the analysis, which are selected in the user interface (see Figure 6.12), are extracted to a file in the system. The values in the "Organized Class" column corresponds to assigned class of the analyzed hashtag in the campaign and during training of classifier, this value is used.



Figure 6.12. Campaign hashtag analysis result interface for labor day, thanksgiving, and election related tweets.

6.3.6.2. Model Generator. For the classification of behaviors in campaign tweets, Java main classes are implemented by using Spark MLlib. Classifiers are generated running these main classes on exported training files.

6.3.7. Features

This section illustrates visualizations of extracted features by comparing organic and organized campaigns. In the figures, the features of #thanksgiving and #draintheswamp hashtag are given. Here, #thanksgiving hashtag represents tweets, which do not have organized activity in it, while #draintheswamp represents tweets, that have organized activity in it. By this way, in this section, it is aimed to show differences between organized and organic activities for each feature. In the implemented system, all visualizations in this section are automatically generated for each analysis.

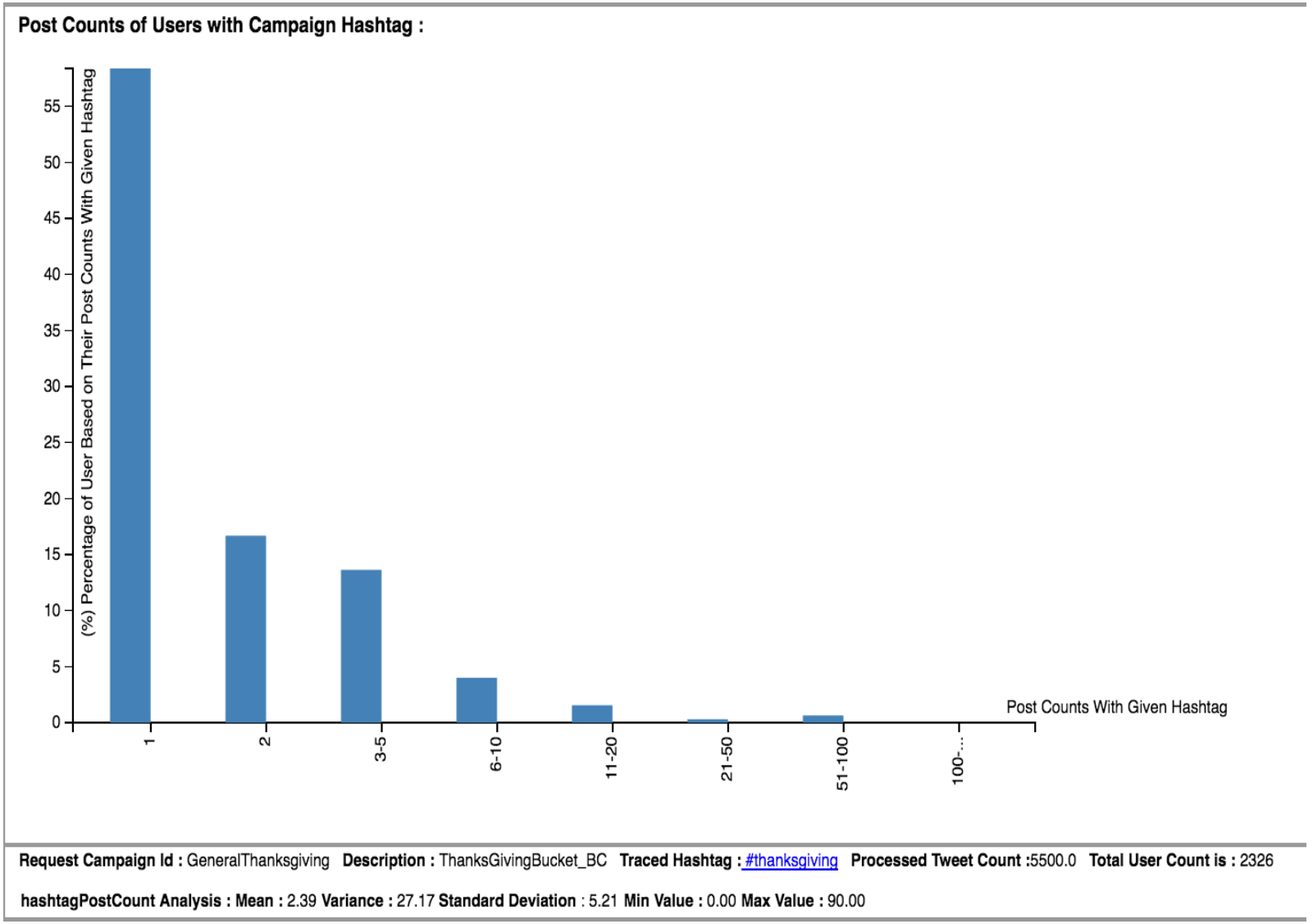


Figure 6.13. Percentage of #thanksgiving users based on their tweet counts.

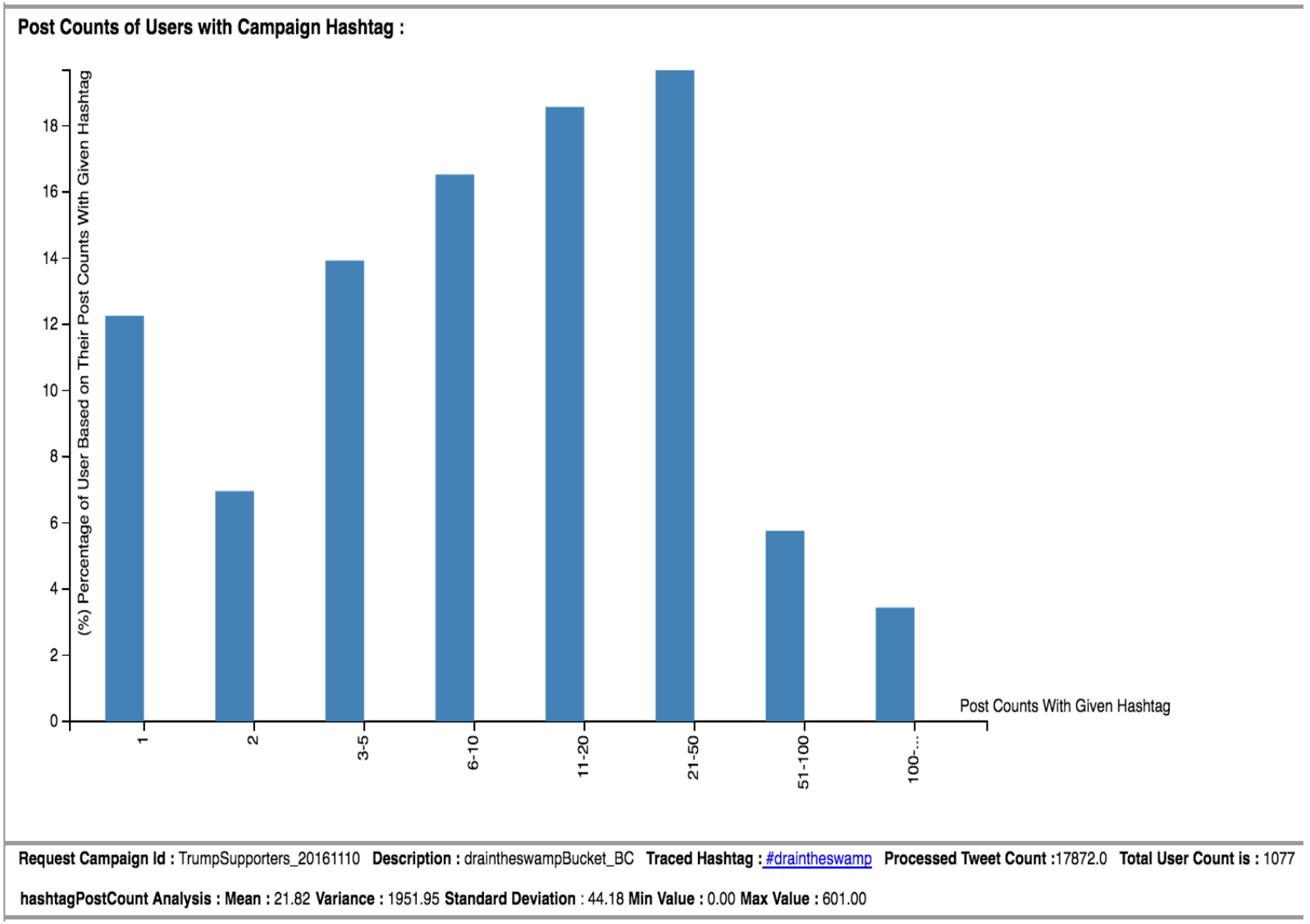


Figure 6.14. Percentage of #draintheswamp users based on their tweet counts.

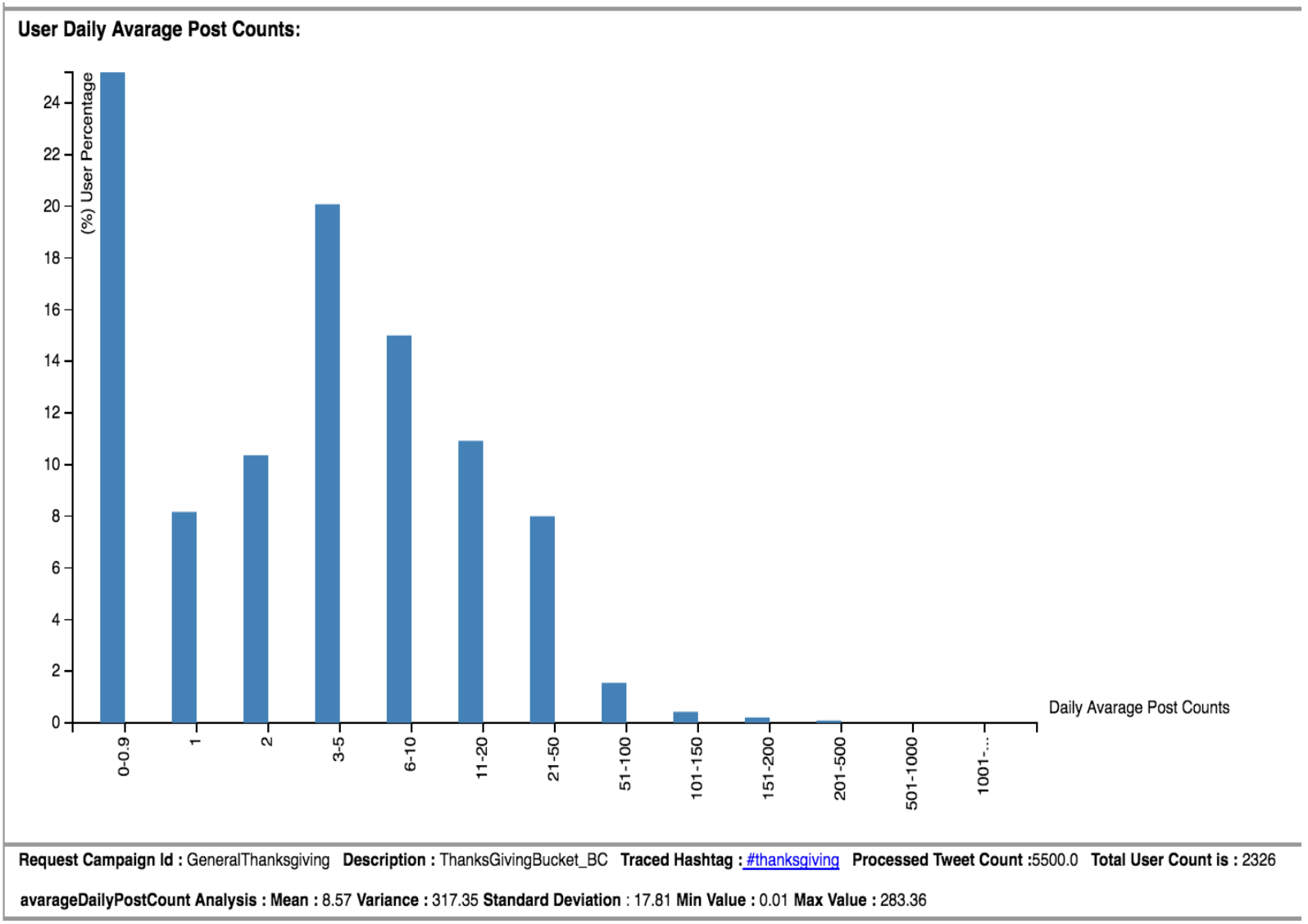


Figure 6.15. Percentage of #thanksgiving users based on daily tweet counts.

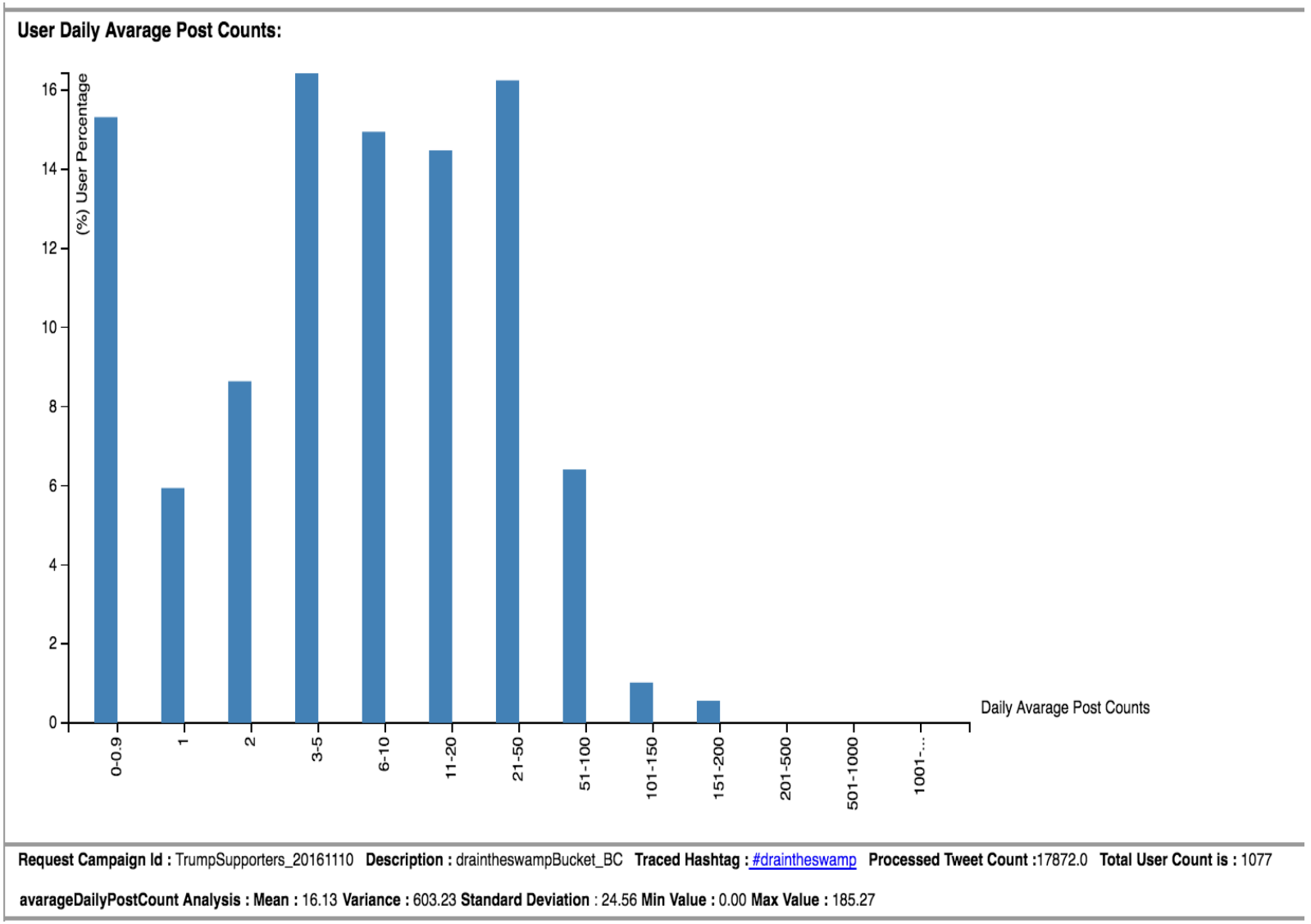


Figure 6.16. Percentage of #draintheswamp users based on daily tweet counts.

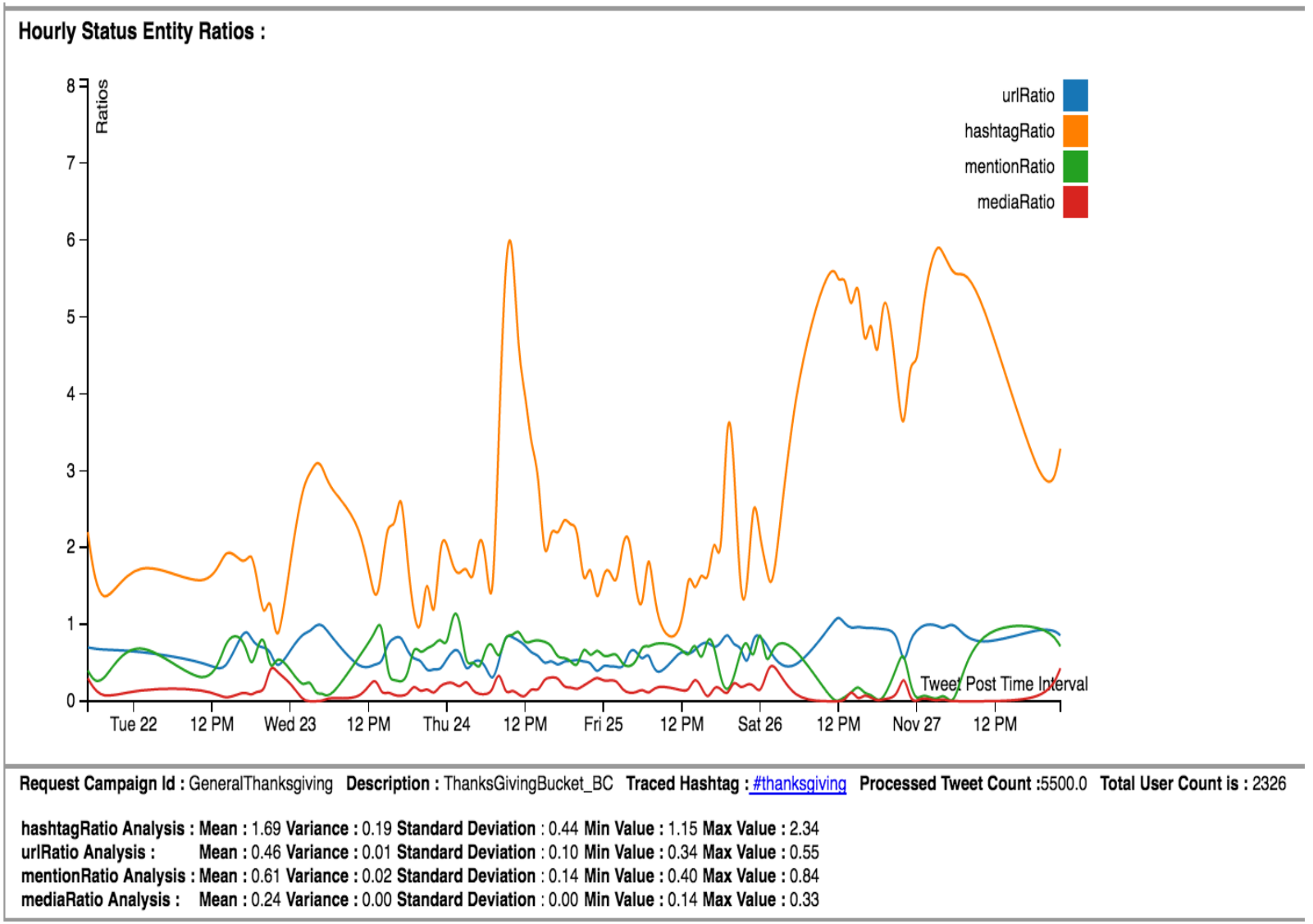


Figure 6.17. Hourly entity ratios of #thanksgiving tweets.

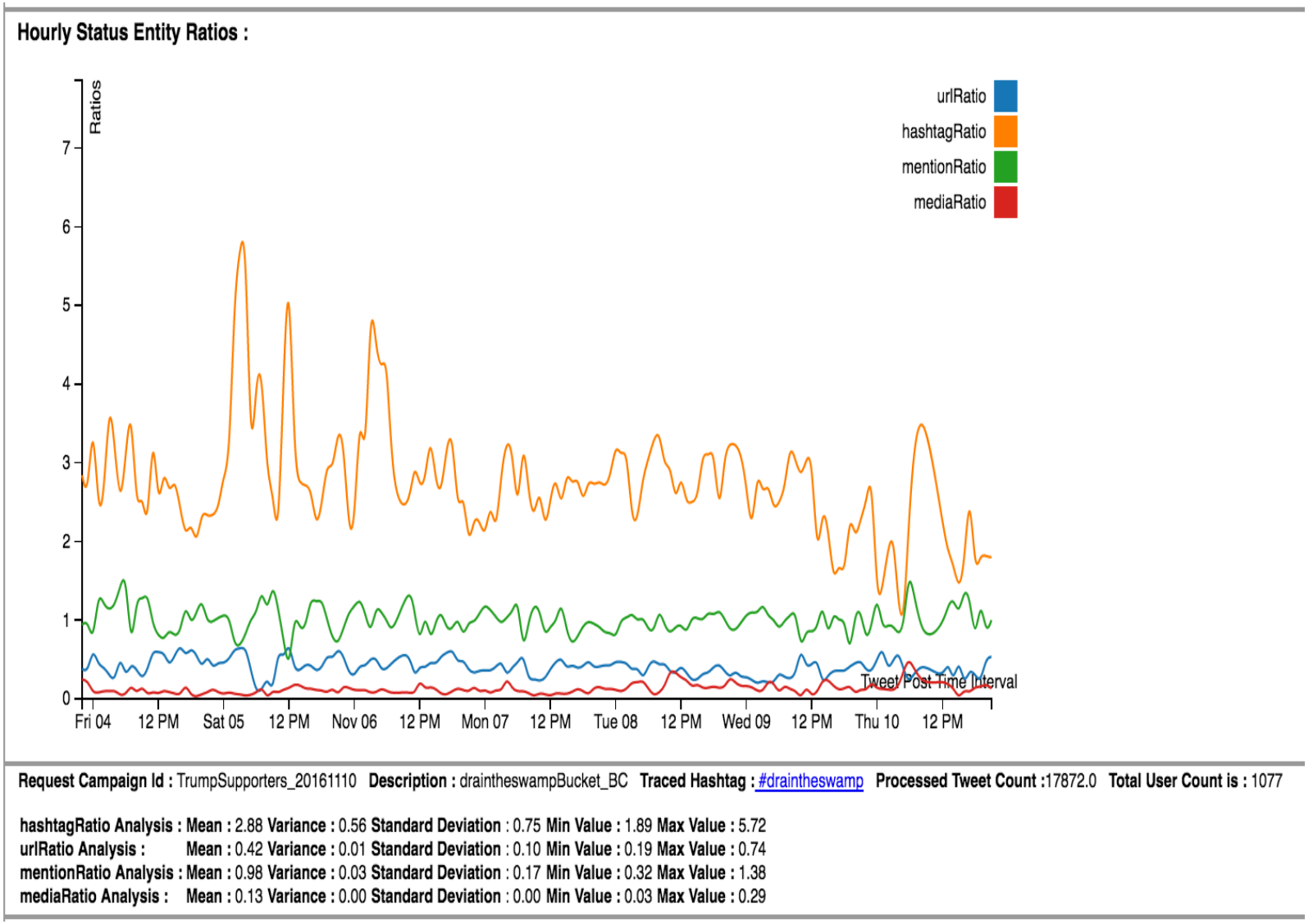


Figure 6.18. Hourly entity ratios of #draintheswamp tweets.

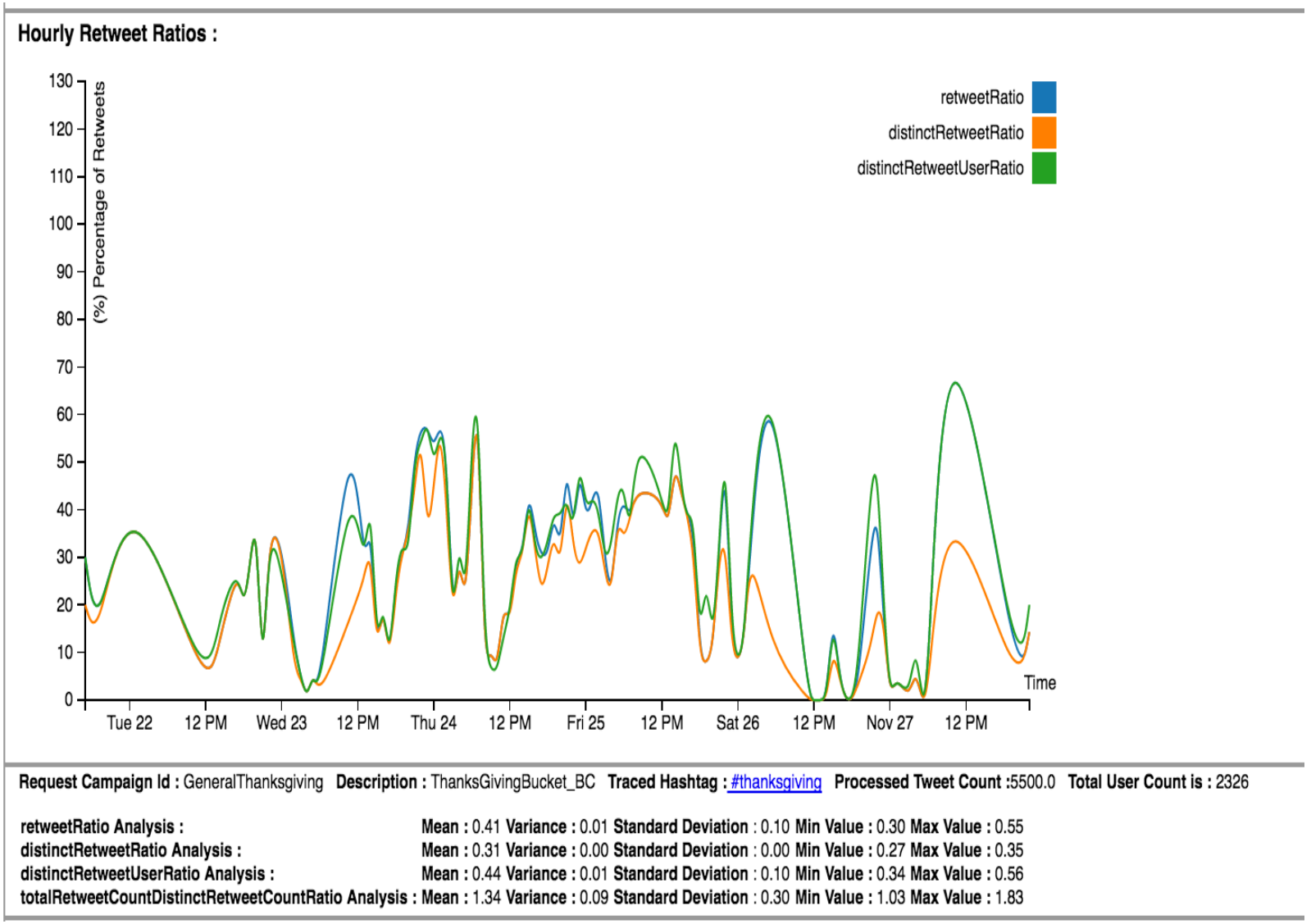


Figure 6.19. Hourly retweet ratios of #thanksgiving tweets.

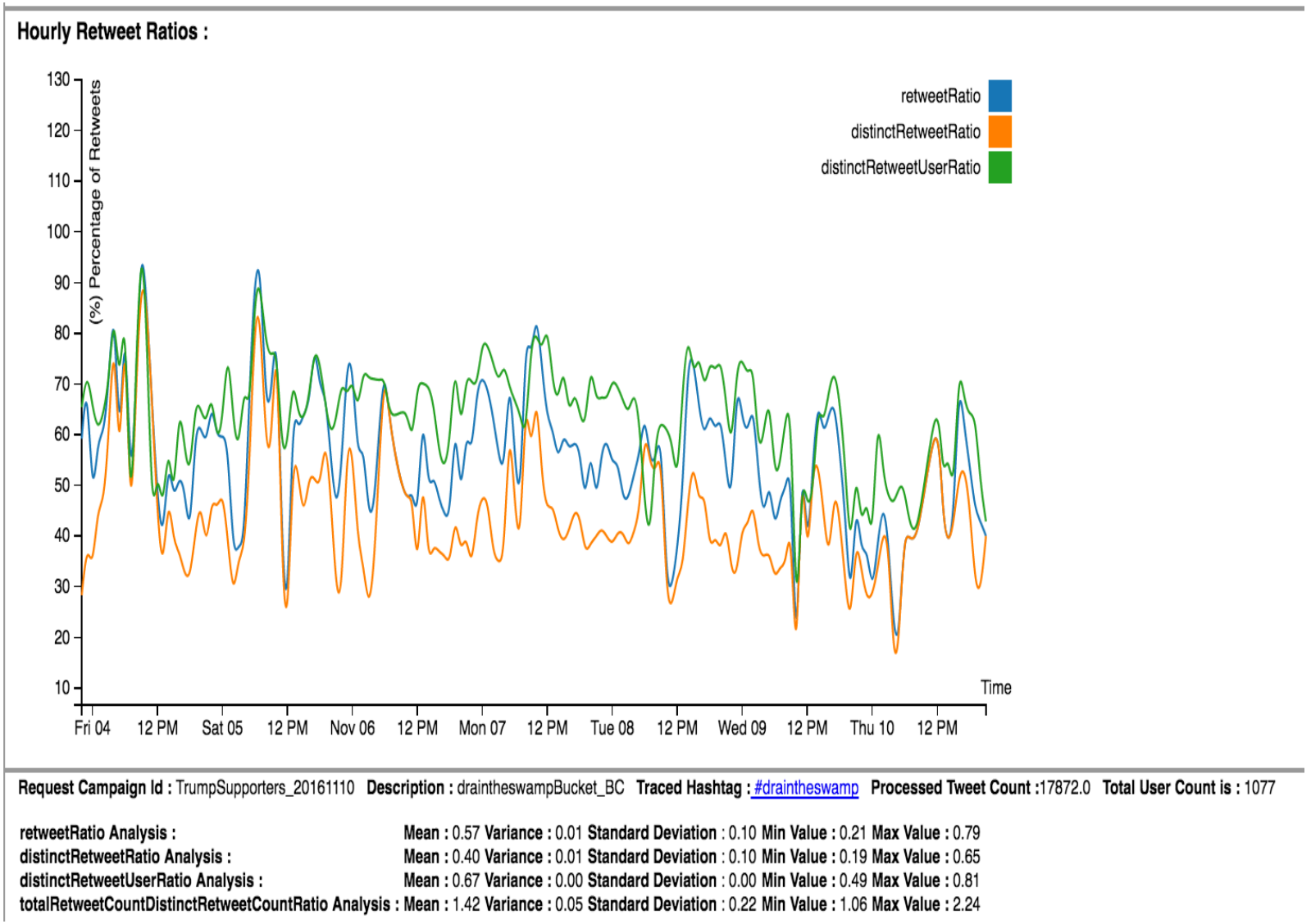


Figure 6.20. Hourly retweet ratios of #draintheswamp tweets.

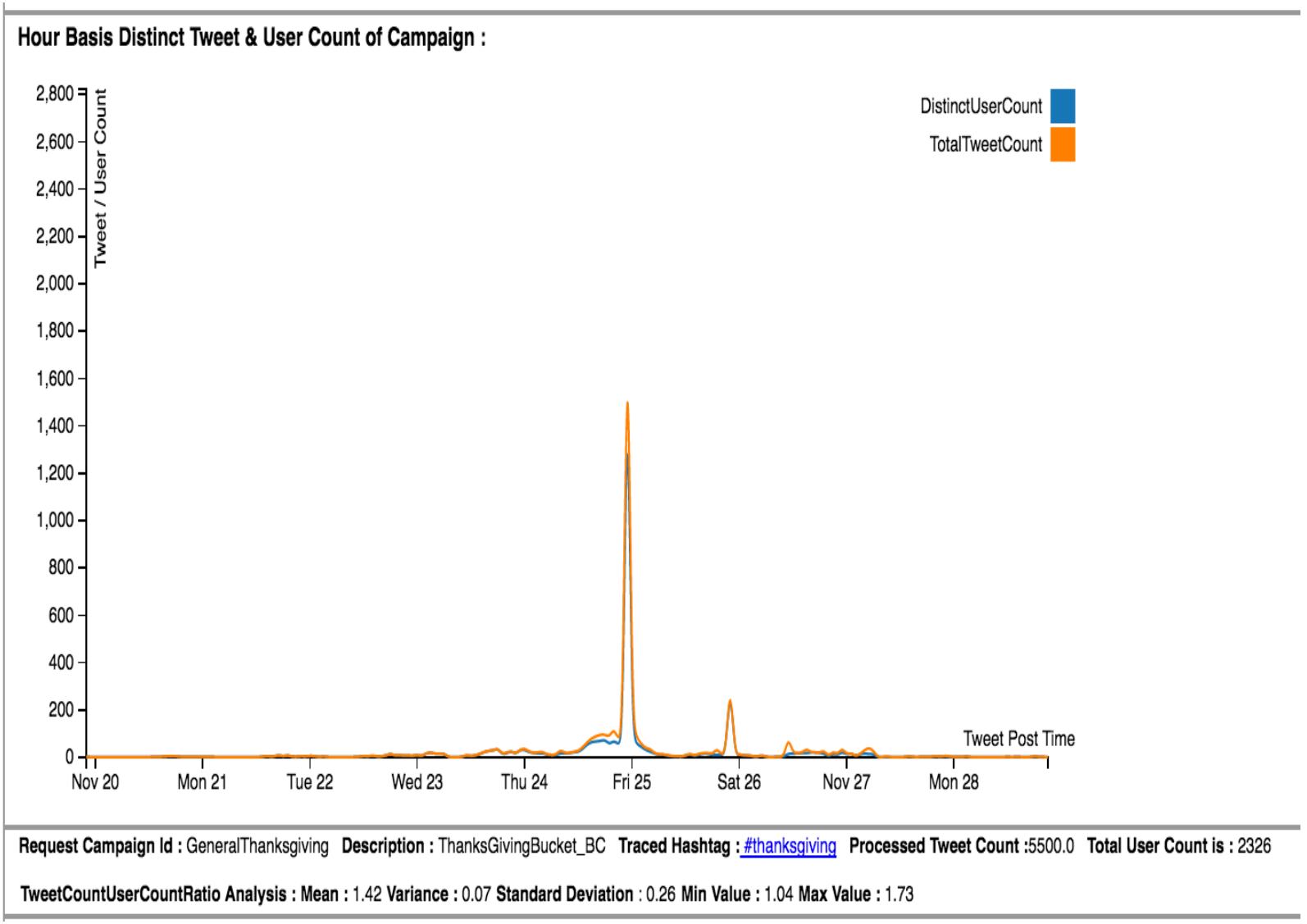


Figure 6.21. Count of hourly #thanksgiving tweets and distinct users, who post tweets.

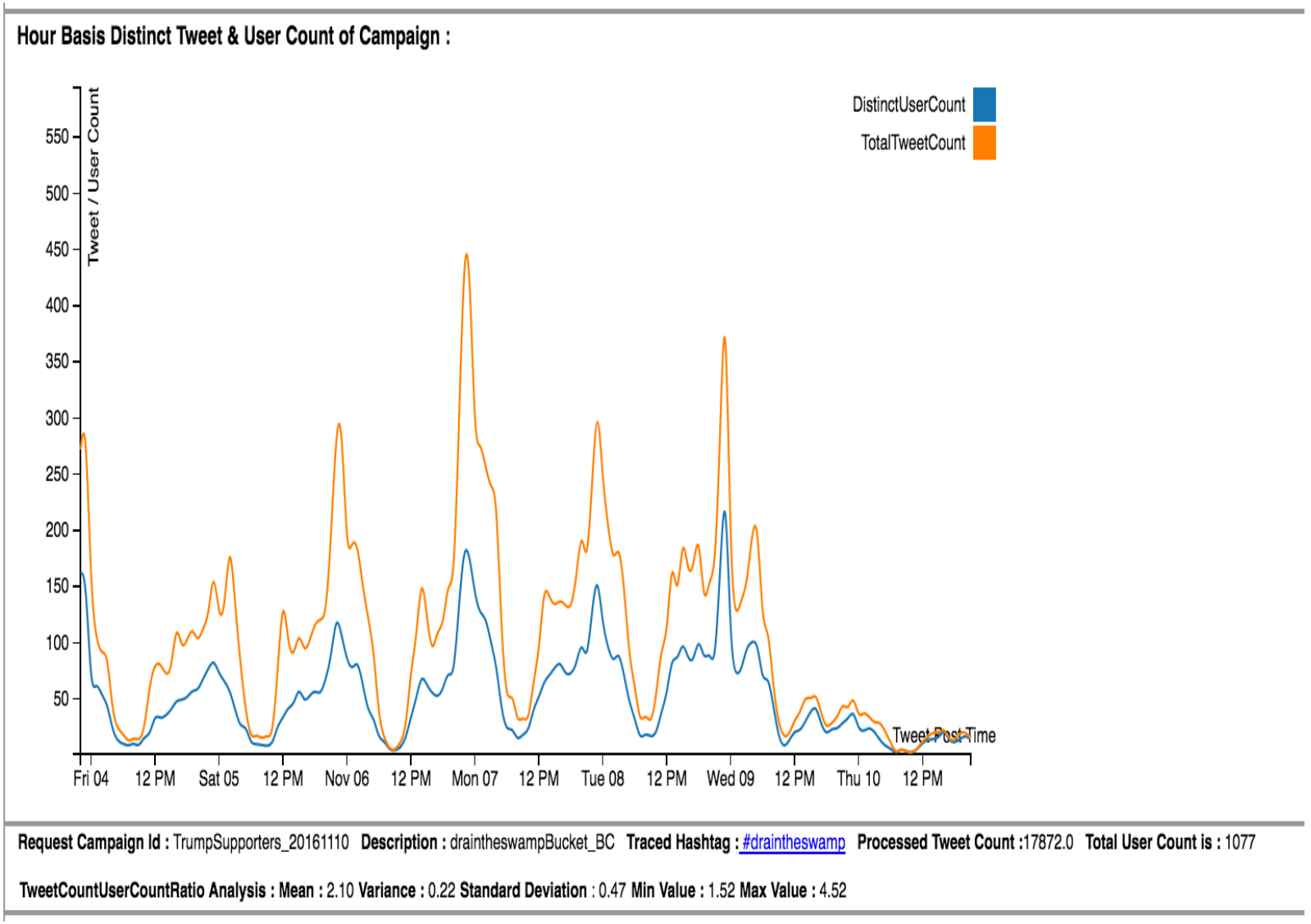


Figure 6.22. Count of hourly #draintheswamp tweets and distinct users, who post tweets.

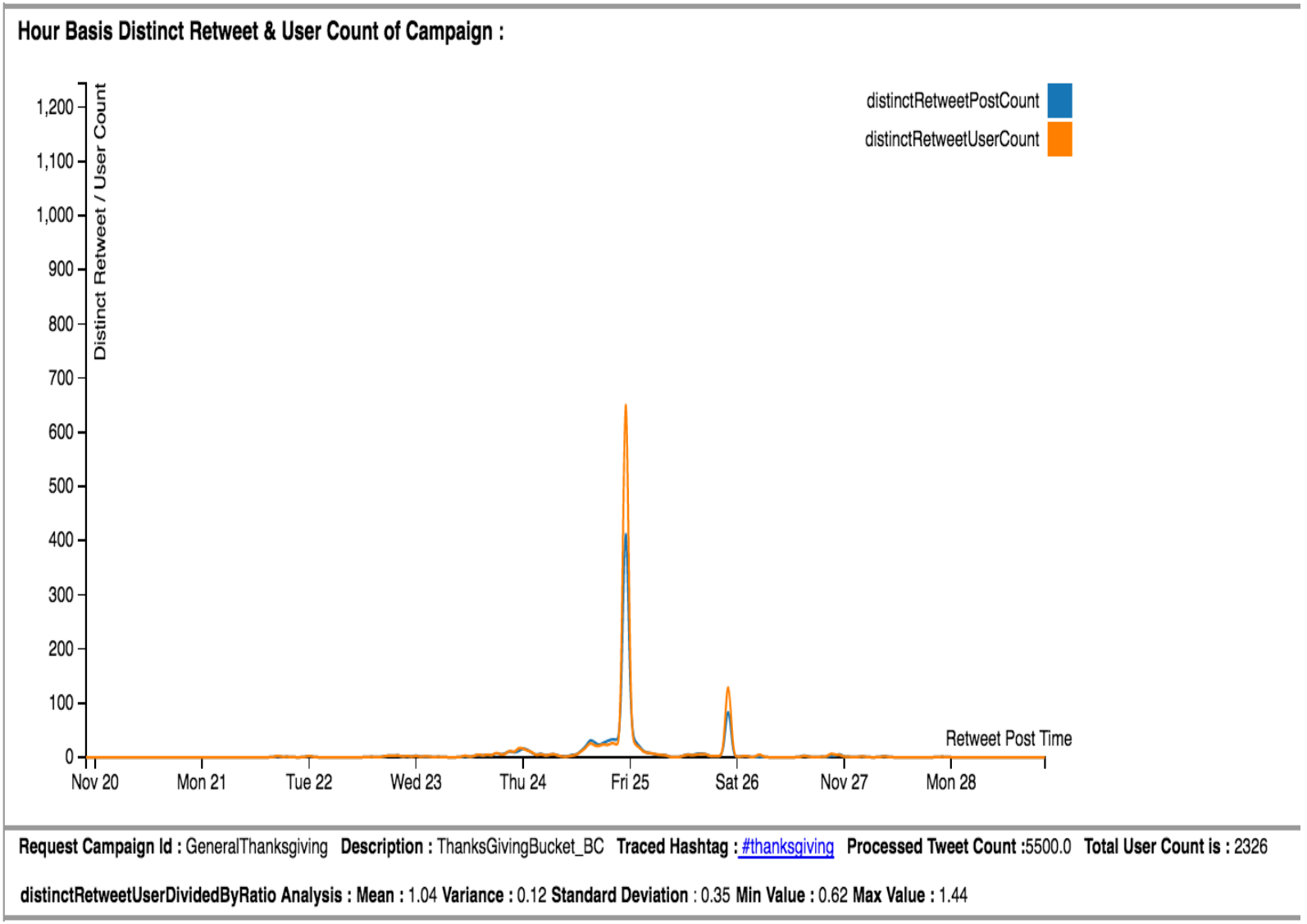


Figure 6.23. Count of hourly #thanksgiving retweets and distinct users, who post tweets.

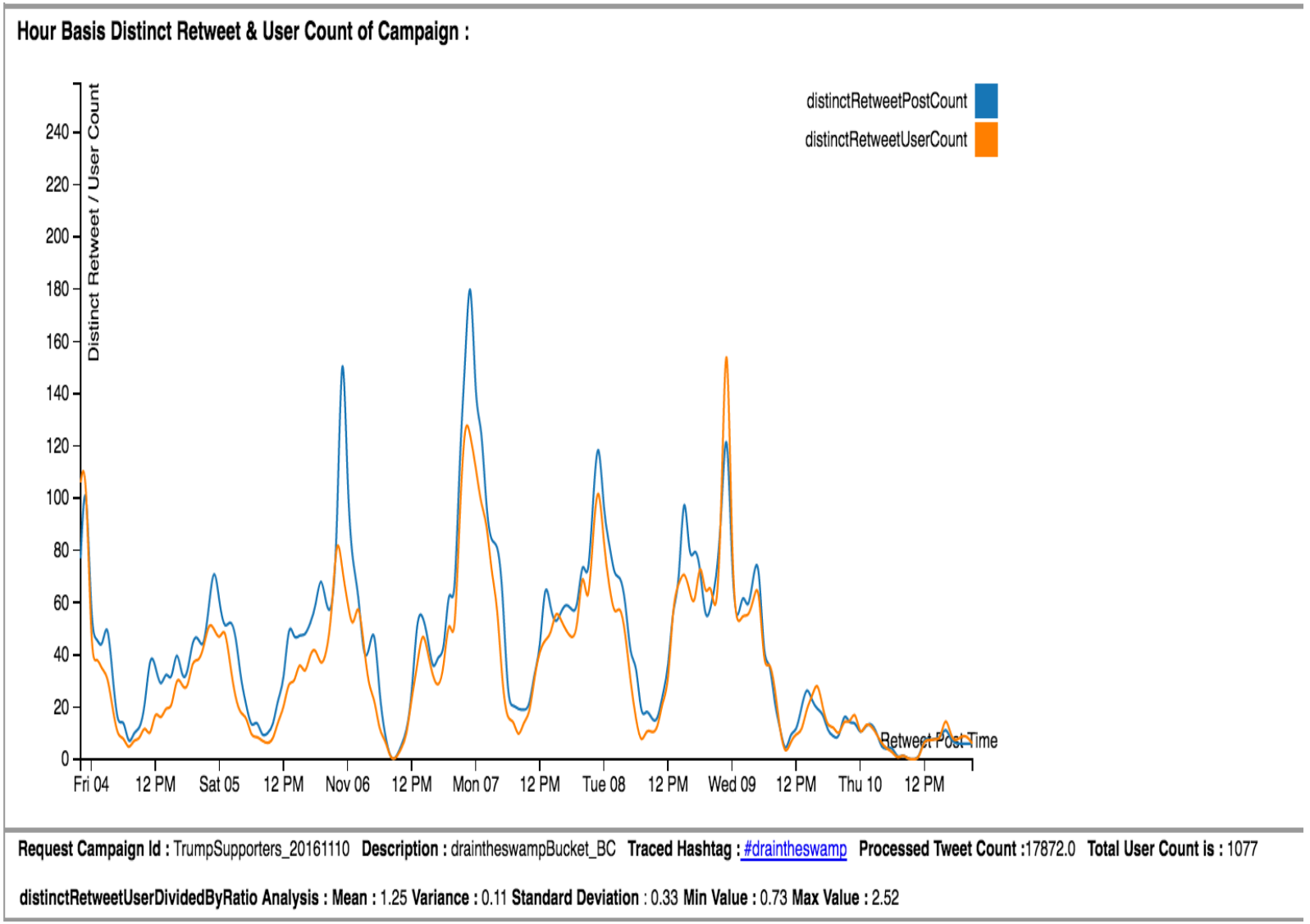


Figure 6.24. Count of hourly #draintheswamp retweets and distinct users, who post tweets.

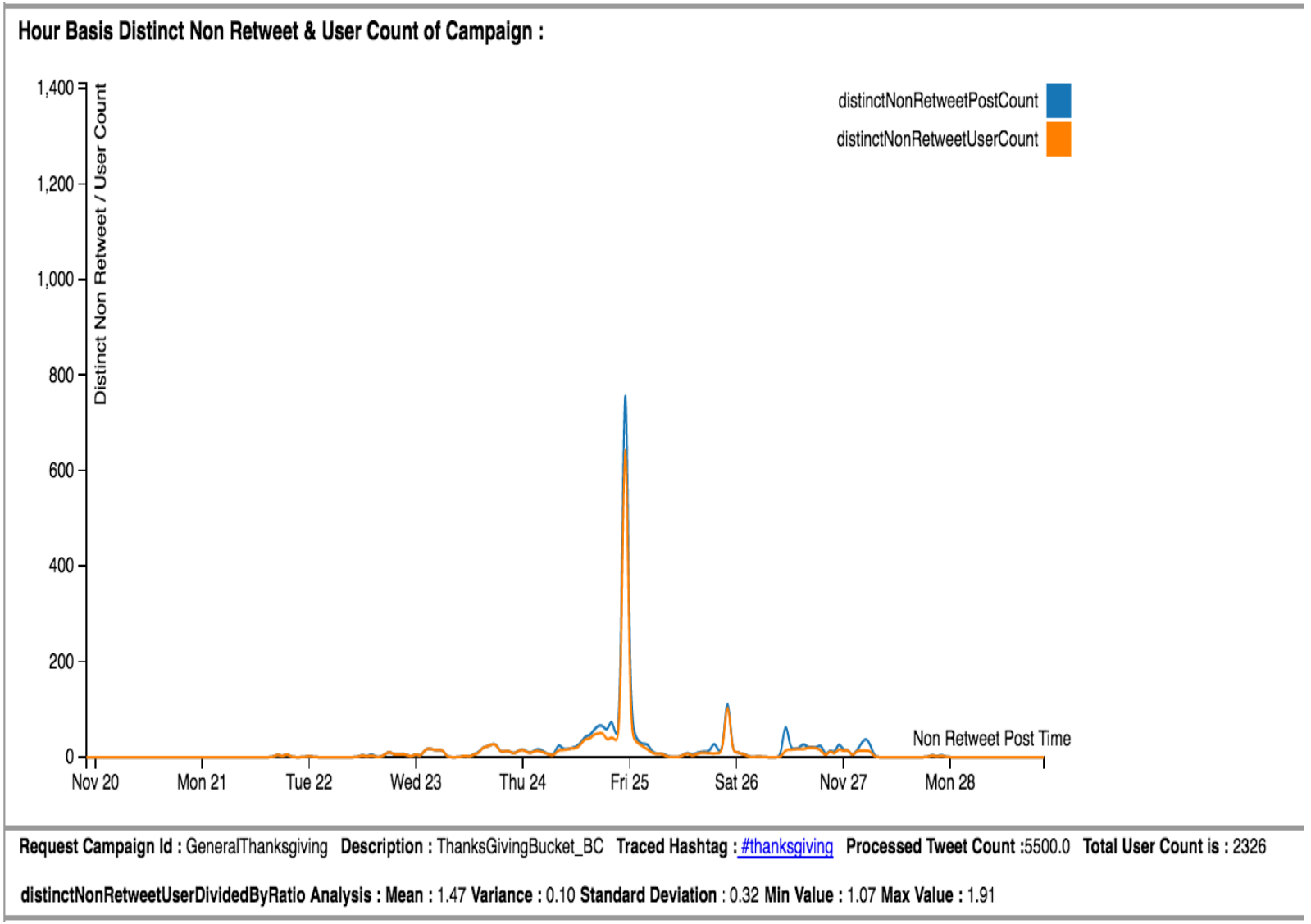


Figure 6.25. Count of hourly #thanksgiving non-retweets and distinct users, who post tweets.

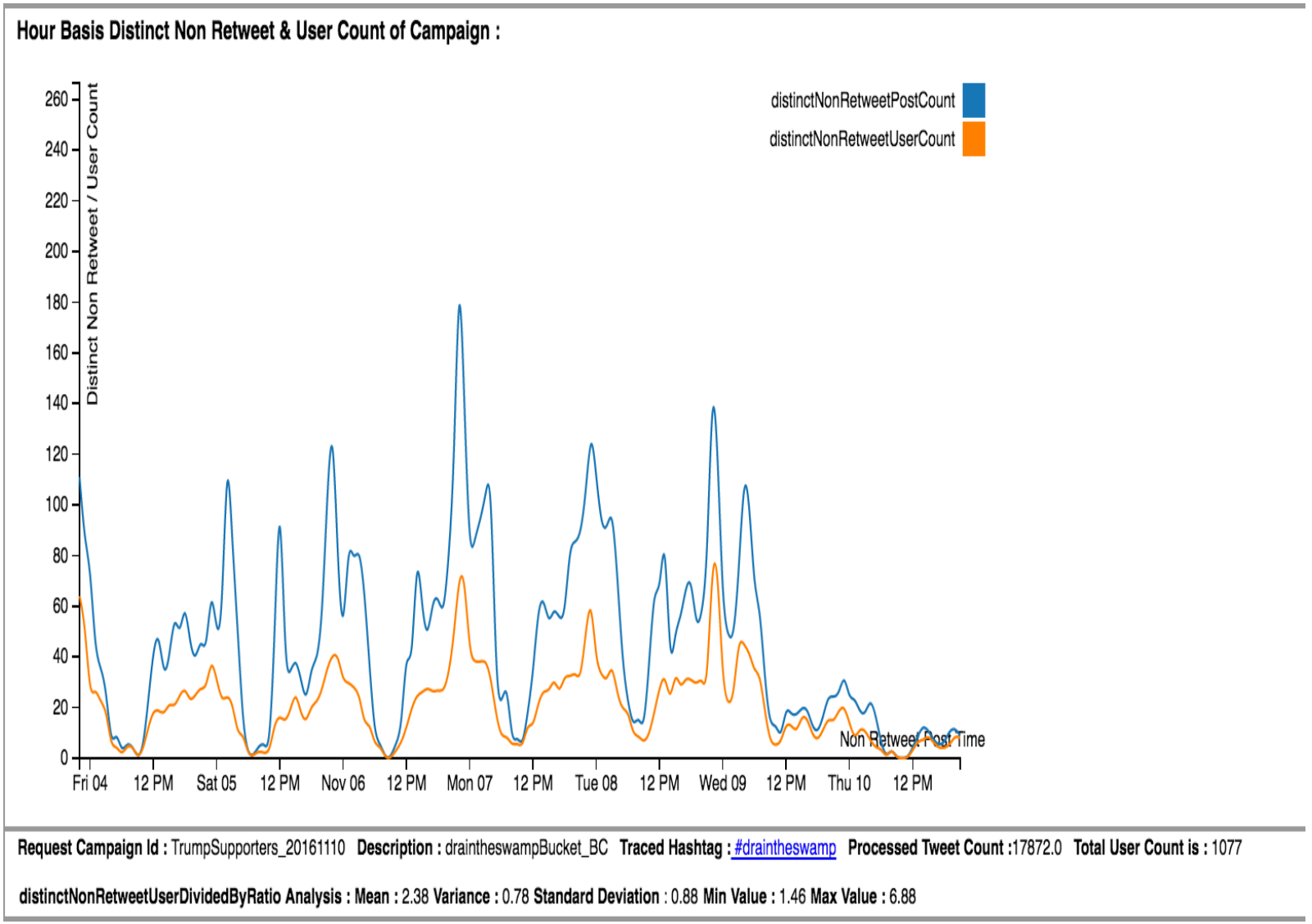


Figure 6.26. Count of hourly #draintheswamp non-retweets and distinct users, who post tweets.

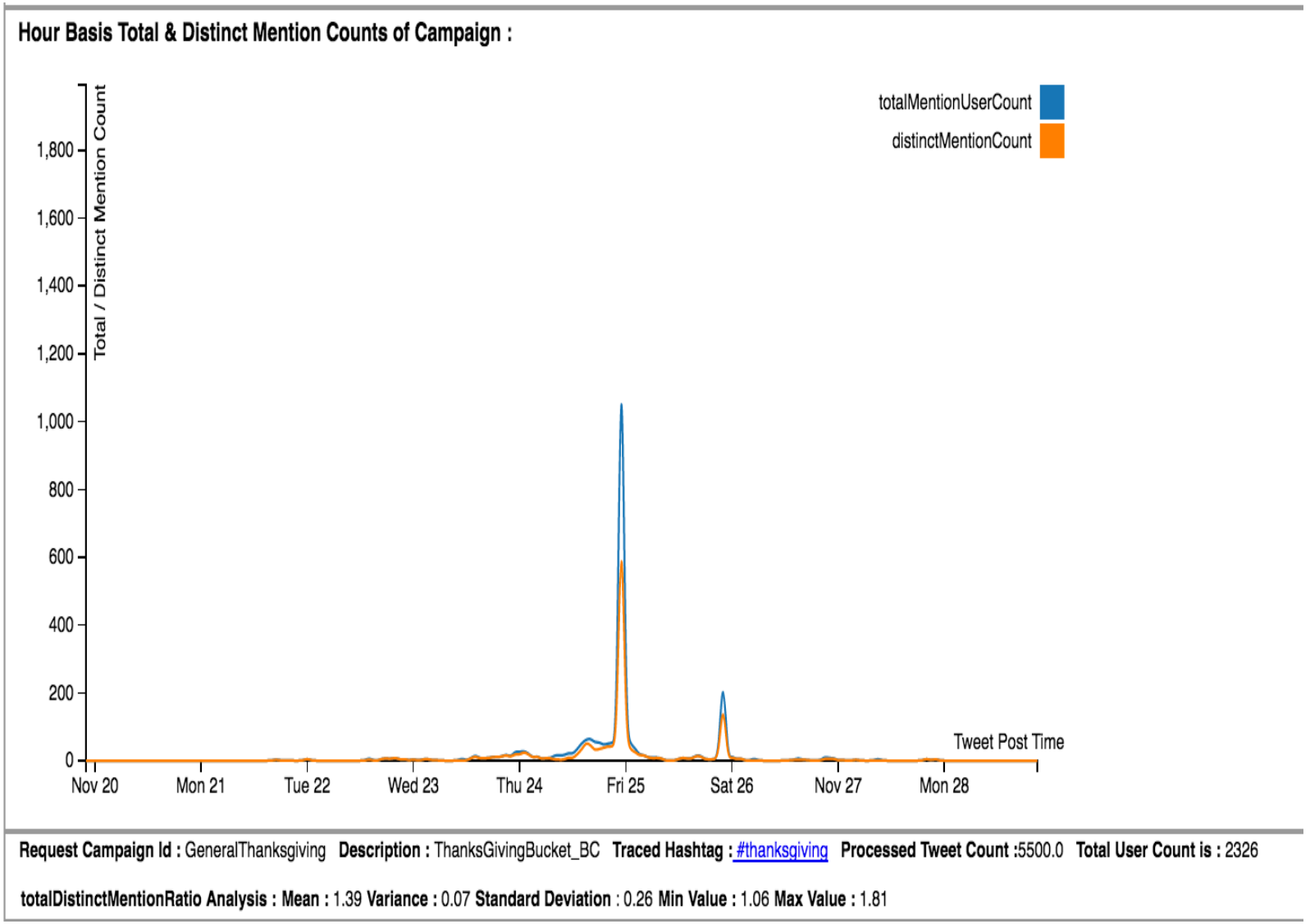


Figure 6.27. Count of hourly total and distinct mentions for #thanksgiving.

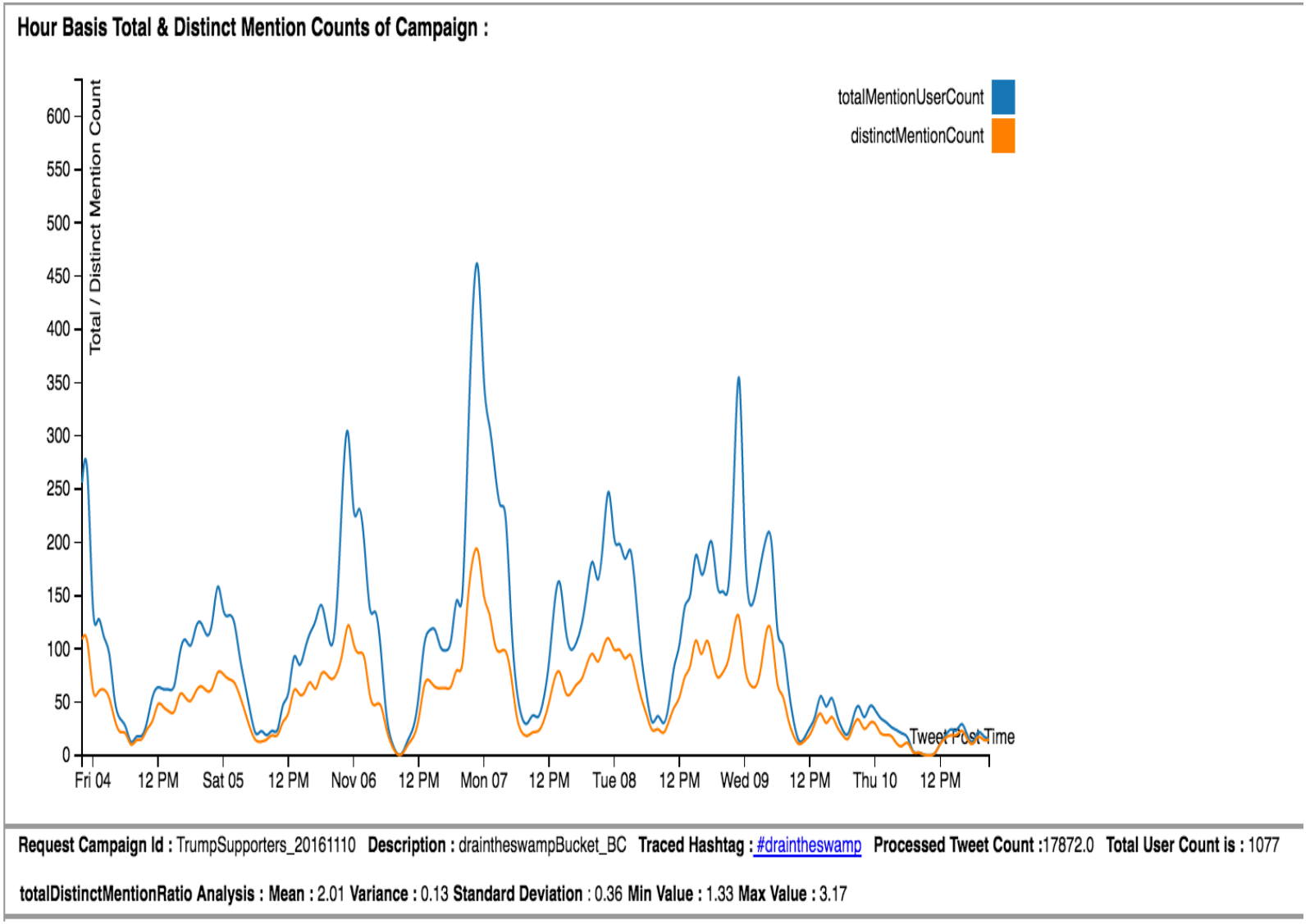


Figure 6.28. Count of hourly total and distinct mentions in #draintheswamp tweets.

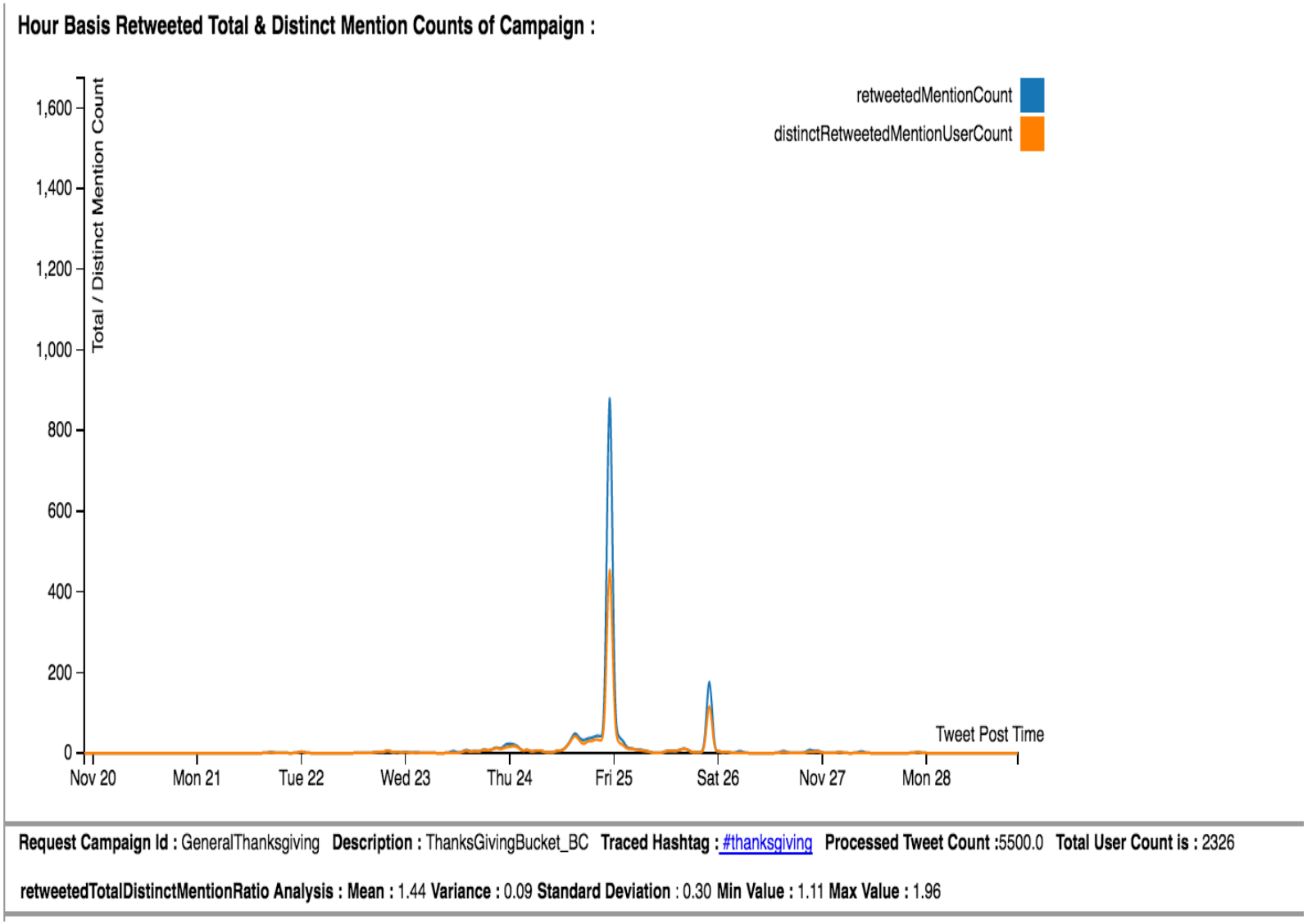


Figure 6.29. Count of hourly total and distinct mentions in #thanksgiving set of retweets.

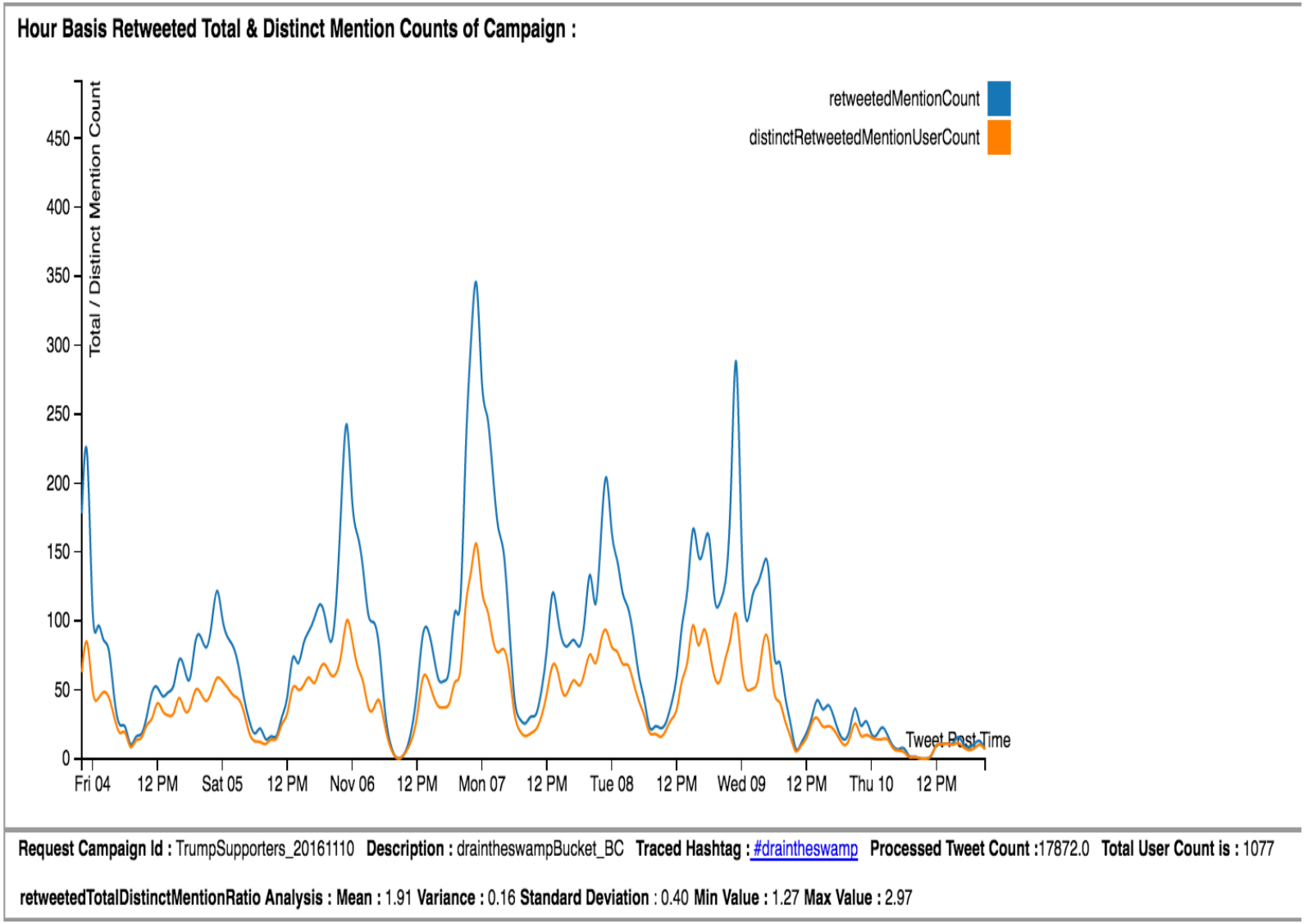


Figure 6.30. Count of hourly total and distinct mentions in #draintheswamp set of retweets.

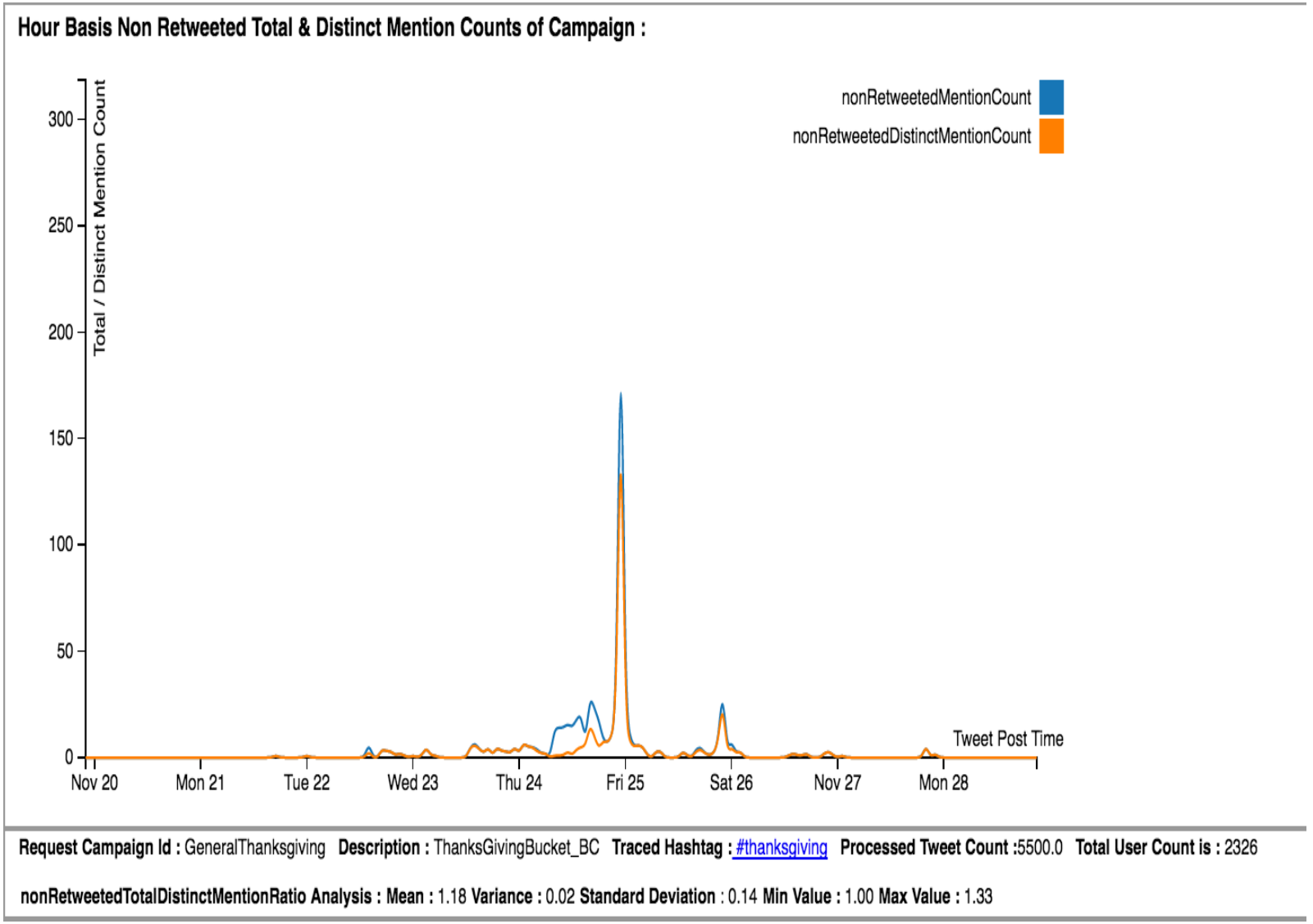


Figure 6.31. Count of hourly total and distinct mentions in #thanksgiving non-retweets.

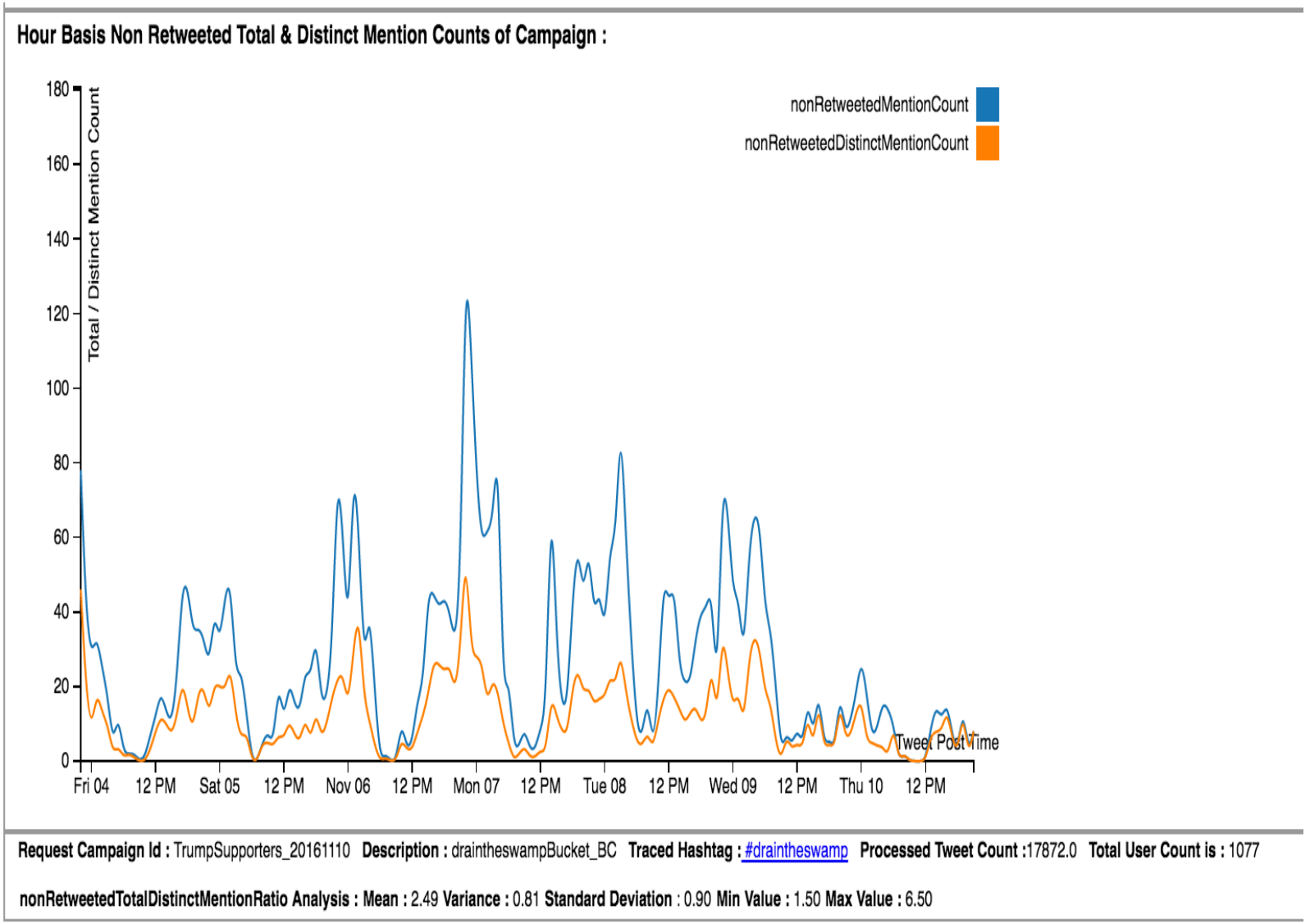


Figure 6.32. Count of hourly total and distinct mentions in #draintheswamp non-retweets.

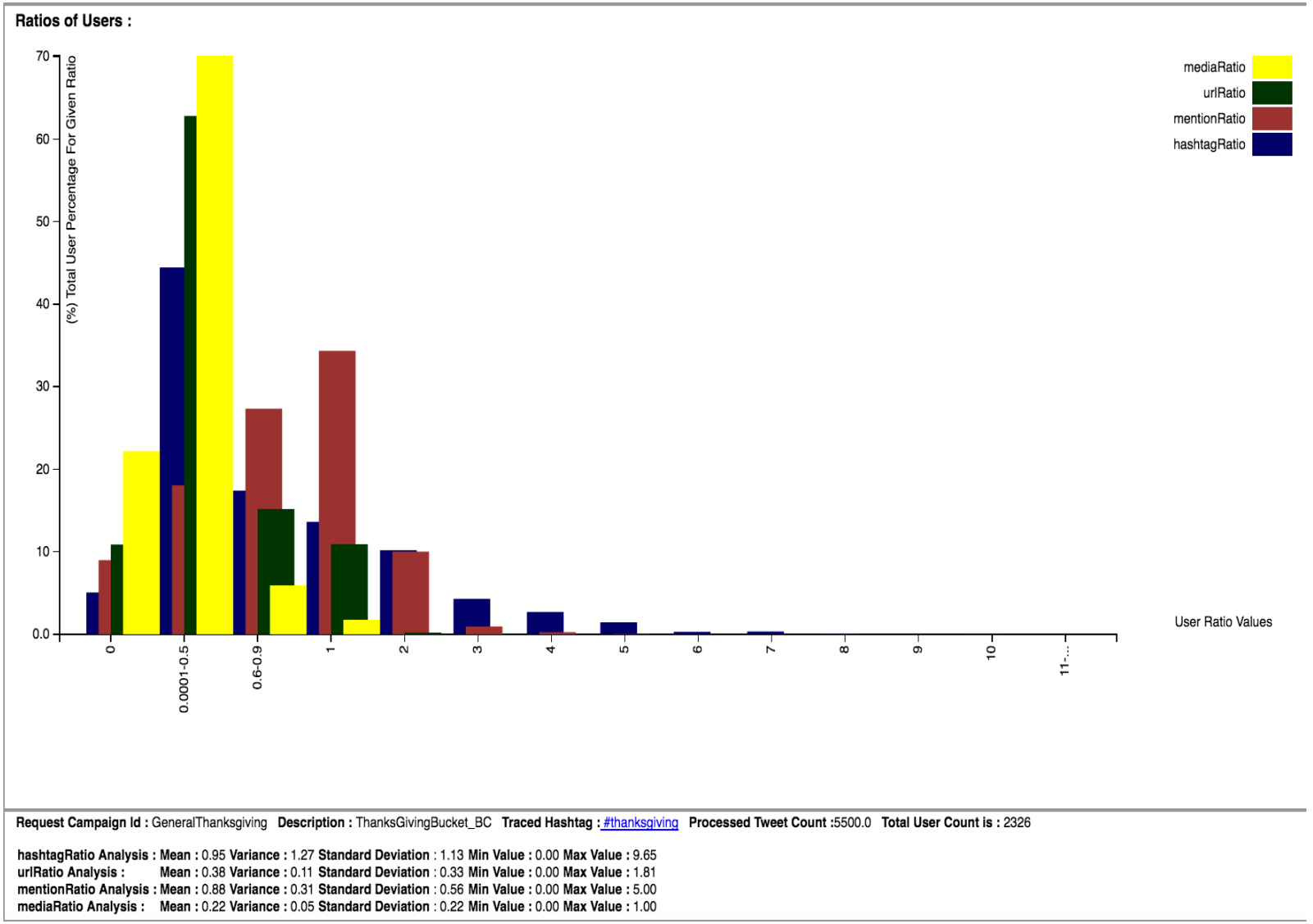


Figure 6.33. Entity ratios of #thanksgiving users.

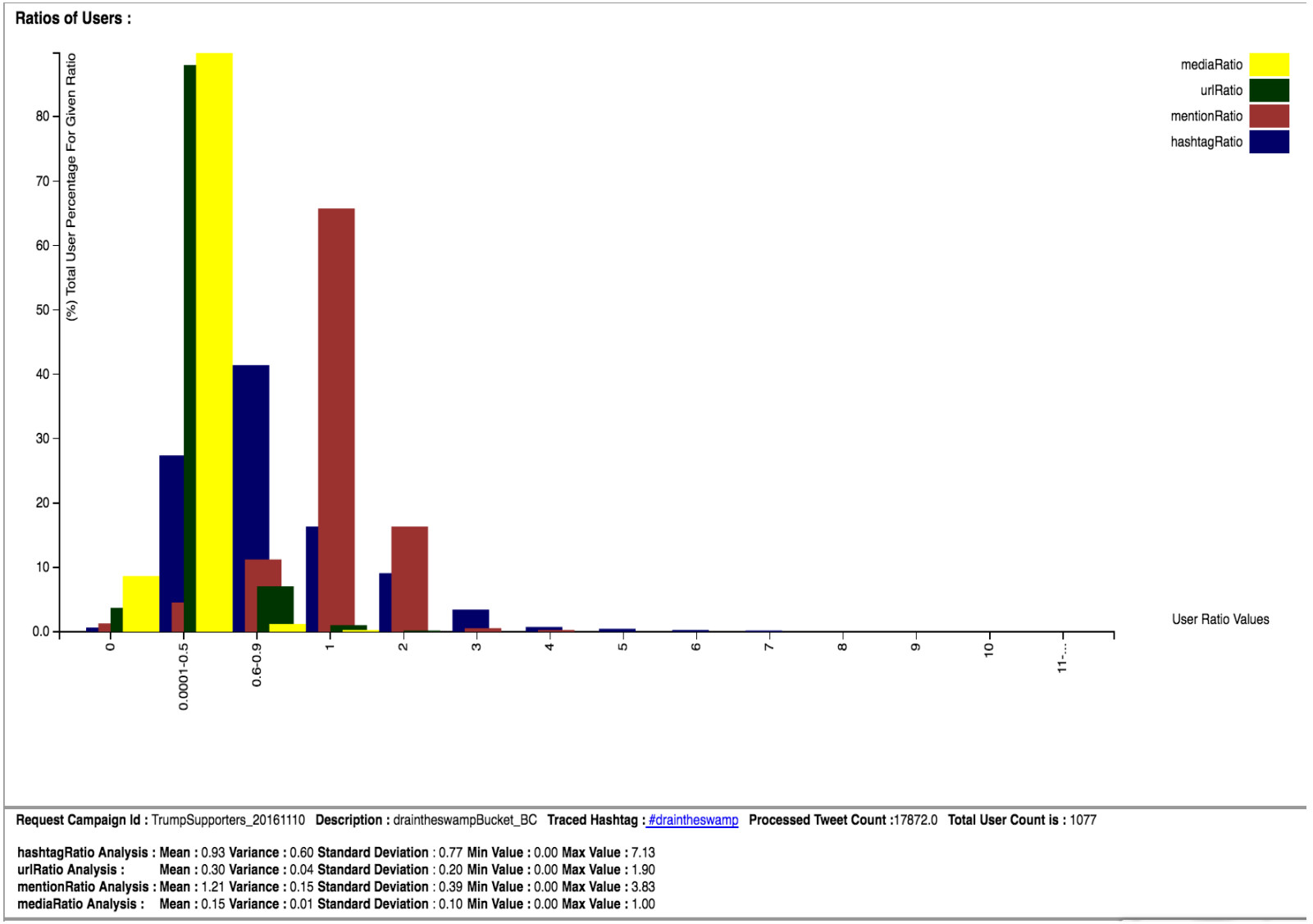


Figure 6.34. Entity ratios of #draintheswamp users.

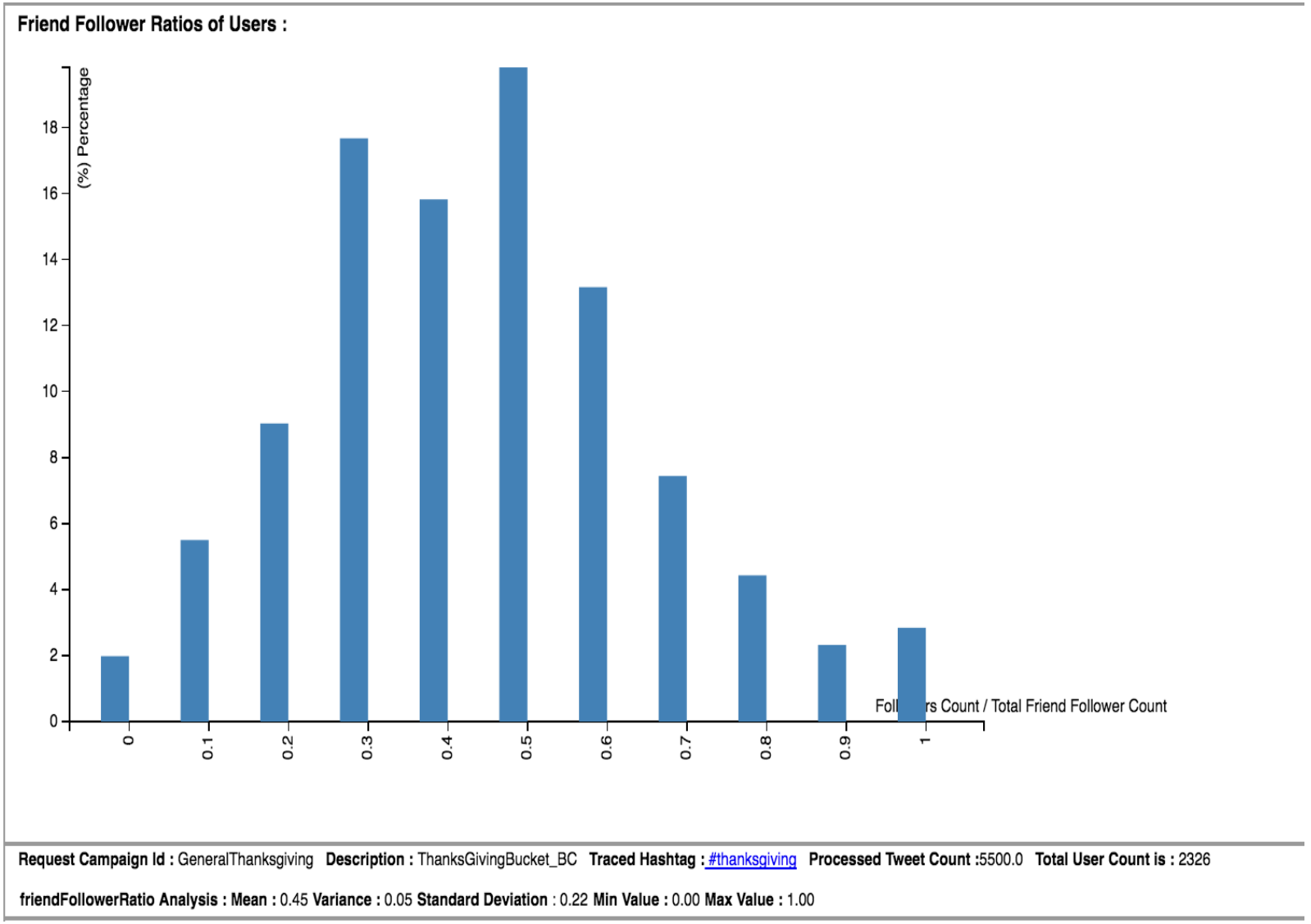


Figure 6.35. Friend follower ratios of #thanksgiving users.

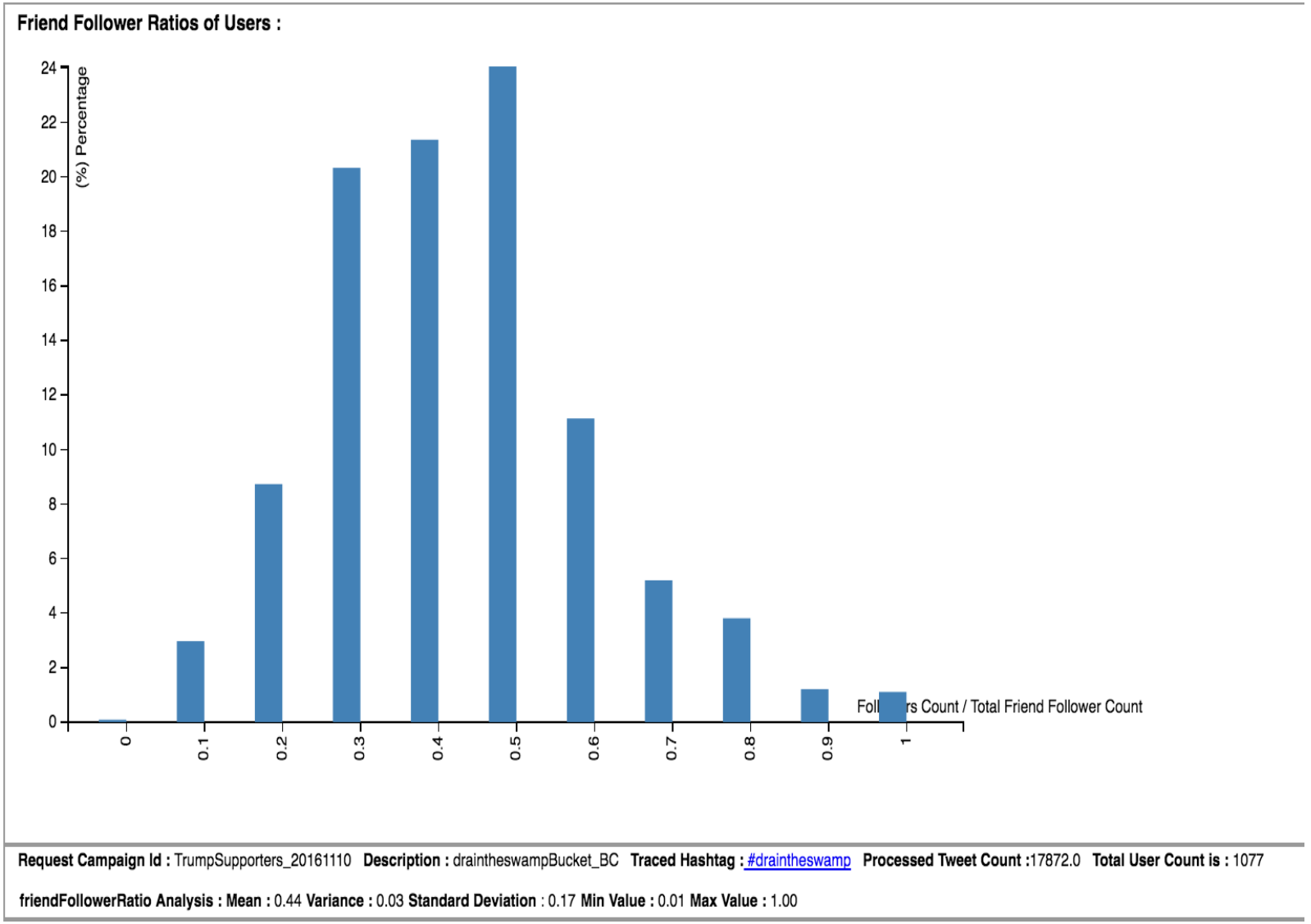


Figure 6.36. Friend follower ratios of #draintheswamp users.

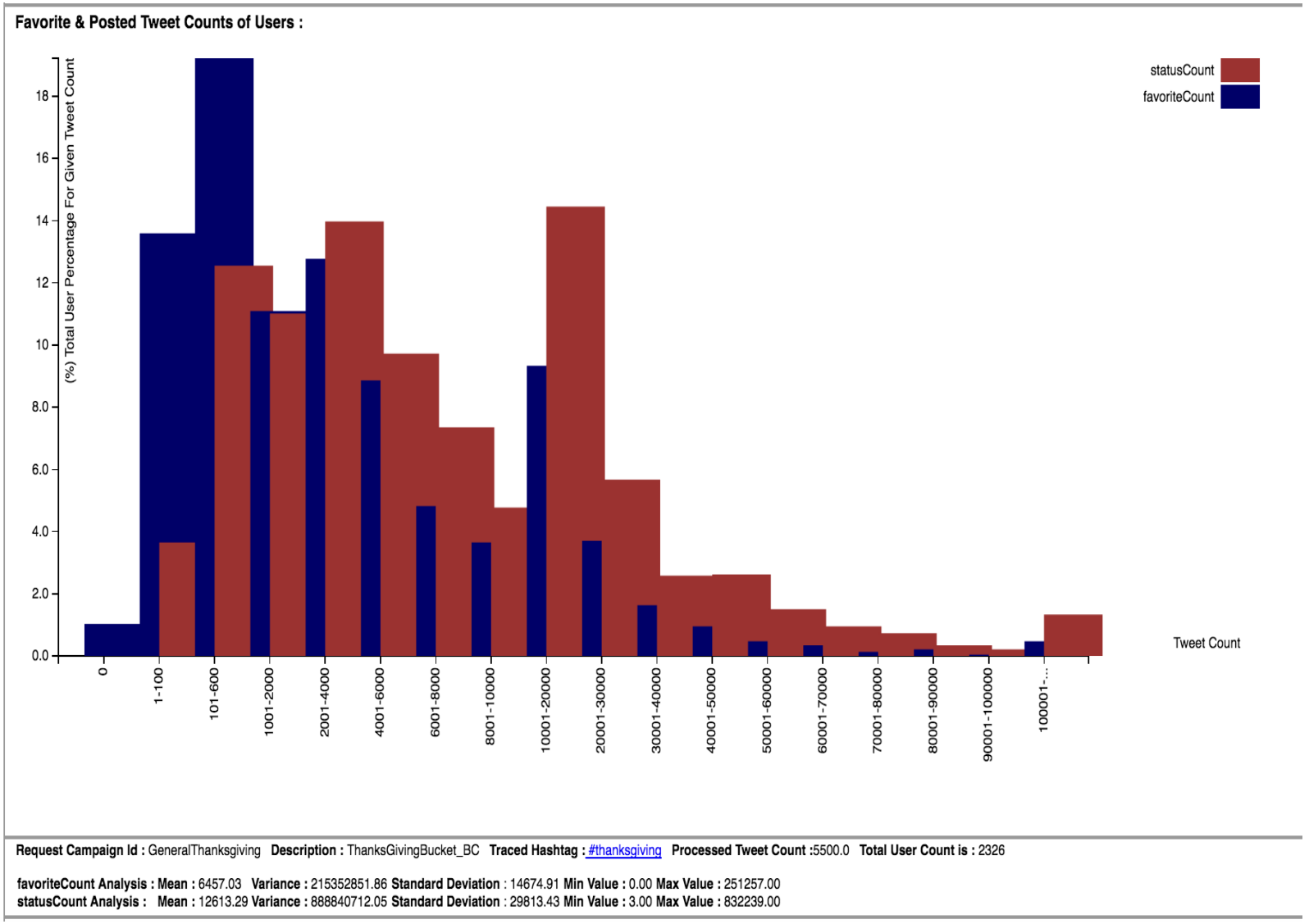


Figure 6.37. Tweet and favorited tweet counts of #thanksgiving users.

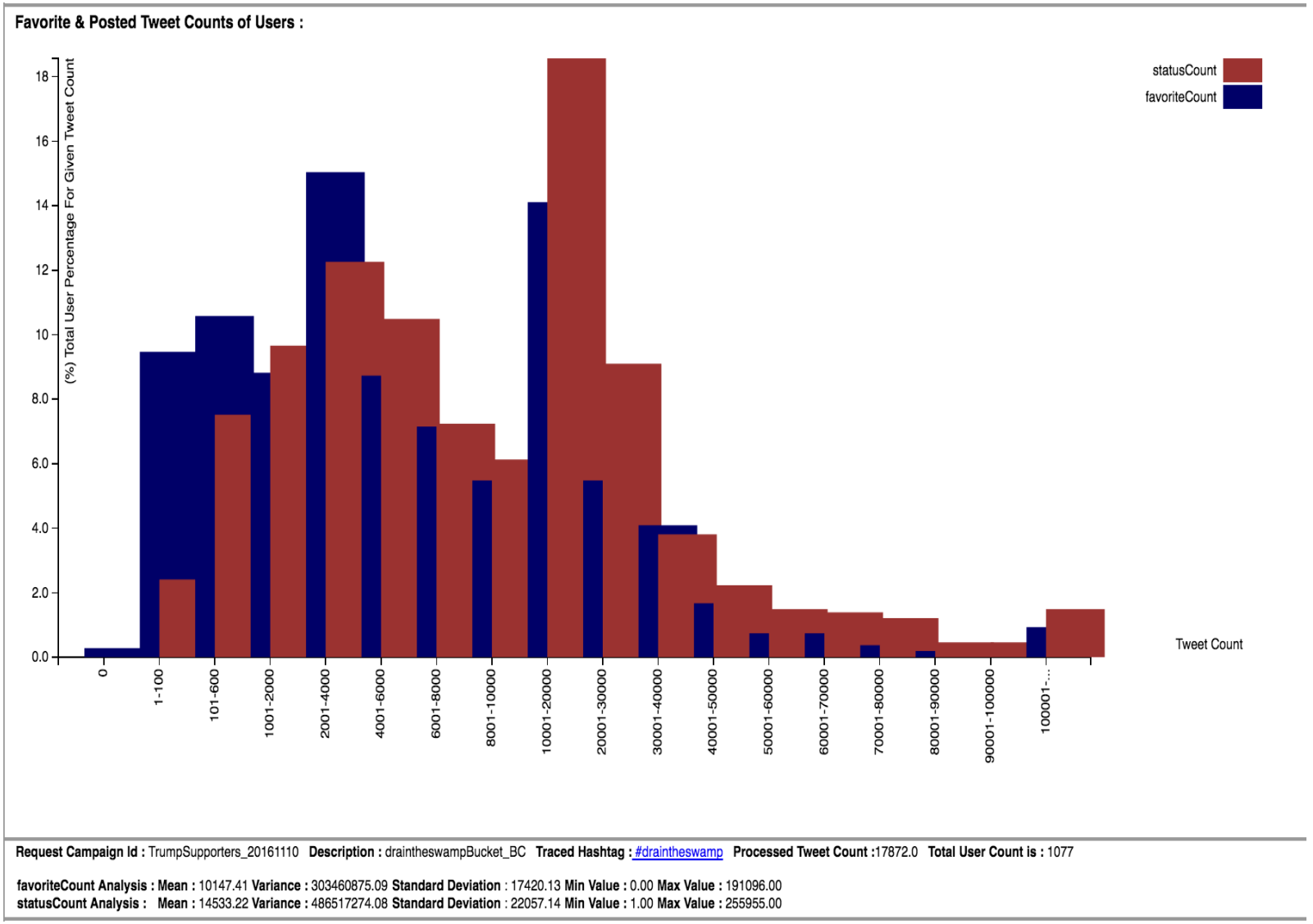


Figure 6.38. Tweet and favorited tweet counts of #draintheswamp users.

7. EXPERIMENTS AND RESULTS

In this thesis, the data set is prepared by analyzing more than 200 million tweets, which are collected with Twitter Search API and User Timeline API. As it is stated in earlier chapters, in order to create training data set, campaigns with traced hashtags are created and then these tweets and users of them are analyzed. Later on, as a result of this analysis, features are obtained in order to be used in training data set.

In this work, the campaigns were created based on hashtags about the following topics :

- the 2016 U.S Presidential Election,
- special days like Thanksgiving, Labor Day, and Women's Day,
- events that emerges in in U.S.A. like #NoDapl and #boycottUnitedAirlines, and
- advertisements & announcements #LadyGagaNewVideo.

The training data set has more than 290 features, which are described in Sections 5.6.1.1 and 5.6.2. The minimum, maximum, standard deviation, and variance of all these features are also used as a features.

7.1. Labeling Campaigns

Records in the training data set are labeled manually both for organic and organized class. For labeling organized and organic records, following steps are executed.

7.1.1. Labeling Organized Records

For the collected hashtags in the category of "U.S. Presidential Election 2016", the analyzed records are tagged as "organized", if there exists a study or news that states existence of significant amount of bot account. Example hashtags and references are as below.

- #tcot, #pjnet [1]
- #nevertrump, #jillnothill, #trumpwon [4]
- #AmericaFirst, #deplorable, #ImWithHer, #LoveTrumpsHate [45]

Besides from these, the analyzed record for a traced hashtag is tagged as "organized", if an important amount of these hashtags users also tweeted with other same hashtags in different time intervals (see Section 7.1.3). Furthermore, an analyzed record is tagged as "organized" also by manually examining tweets and users of the traced hashtag.

7.1.2. Labeling Organic Records

Analysis of hashtags, which belongs to emerged events causing instant reactions in the society, are tagged as "organic". For example, hashtags related with United Airlines, such as #unitedairlines, #boycottUnitedAirlines, had become popular after a passenger is forcibly removed from the plane.

Furthermore, analysis of hashtags related with special days like Thanksgiving, Labor Day, and Women's Day are also tagged as "organic", since these hashtags are used by different portions of the society.

7.1.3. Same User Analysis

In order to understand the user diversity in a hashtag, "Same User Analysis" is applied. This analysis is done as following. In Table 7.1, an example of process result is given.

(i) Other hashtags, which are in the same line with traced hashtag and used in different time intervals, are analyzed in other retrieved campaigns.

(ii) Users in the tweets of traced hashtag are compared with users in other hashtags' data set.

(iii) Population of same users, who are posted about both traced hashtag and compared hashtag, are found.

Table 7.1. Example same user analysis result.

Traced Hashtag	**Date**	**User Count**	
#votetrump	04-10 Nov 2016	28.114	
Compared Campaigns			
Traced Hashtag	**Date**	**User Count**	**Same User Count**
#podestaemails	17-24 Oct 2016	32.794	7.625
#podestaemails14	17-24 Oct 2016	16.466	5.121
#hillaryemails	27-31 Oct 2016	12.448	3.591
#maga	20 -28 Nov 2016	29.130	7.752
#obamacarefail	27 Oct - 2 Nov 2016	6.123	2.281
#benghazi	24 Oct - 1 Nov 2016	7.854	3.126

As it is seen in Table 7.1, considerably amount of users, who post tweets for hashtag #votetrump, had been also posted in similar hashtags in different times. This suggests existence of an organized activity, since percentage of same users are not that much high in organic hashtags. To give an example, when same users are analyzed in hashtags #Thanksgiving, #Oscars, #internationalwomensday , #NationalSiblingsDay, #WorkersDay, #NationalPetDay #MothersDay, at most 9% of users are same with 6% mean. However, in Table 7.1, it is illustrated that in hashtags, which includes organized activities in it, 40% of users are also taken part in other hashtags in different times.

In this thesis, during manual labeling of data, this analysis is also used for some hashtags in order to understand whether there is an “organized” activity in the hashtag or not. Note that, outputs of this analysis only used for manual labeling process, they are not used as features in the training data set in order to prevent memorizing in the classification model.

7.1.4. Training Data Set Preparation

As it is also stated in Chapter 5, each record in the training data set represents extracted features, which are retrieved after an analysis of traced hashtag in a campaign.

In the beginning of this study, in order to create a non-biased and accurate classifier, it is aimed to create a training data set with at least 10.000 records, which means analysis of 10.000 hashtags. However, because of time, processor, and storage limits, this can not be achieved.

Recall that during extraction of features for each hashtag analysis, the implemented system collects each user’s tweets, which are posted in a time interval of seven days. In the implemented system, this step causes time problems as a result of limited access to User Timeline API. Although, a Twitter connection pool is implemented, the system still waits during retrieval of tweets.
Therefore, analyzing of more hashtags means more waiting.

Furthermore, even if the implemented system works in multi-threaded, tweets retrieved from User Timeline API is saved to one MongoDB collection. Therefore, bottlenecks are experienced after some point in MongoDB and this bottleneck in the DB causes other processes to slow down.

Also, retrieval of all users tweet, causes high storage usages even if a tweet is not saved twice in the implemented system. For example, analysis of 107 hashtags, allocated 426 gigabytes.

Because of these reasons, the training data set is generated by applying a bootstrapped resampling approach. Recall from Section 3.4.1, resampling methods are used in order to train non biased classifiers and increase training data samples in case of actual training data set is small. In this study, this approach is applied as following,

(i) For analysis of a hashtag in the campaign, tweets of hashtag are separated into buckets with sizes of 1500 in the campaign.
(ii) Each bucket is analyzed individually, and saved as one record into training data set
(iii) By this way, even if there are multiple records for same hashtag in the training data set, there exists a diversity in each record since each bucket composed of different users

In conclusion, a training data set with 800 records is created from tweets, which consists of 107 hashtags. As a result of diversity of users in each buckets for the same hashtag, bias in the classifier is aimed to be minimized.

7.2. Results

The results of generated classifier models with Random Forest, Logistic Regression & SVM algorithms are given in Table 7.3, 7.4, 7.5, 7.6. Results in all of these tables are obtained by applying ten-fold cross validation in order to check bias and overfitting condition in generated classifier model. As it is seen in these tables, the success rate of classification methods are tested on four different data sets, which are explained in Table 7.2.

As it is explained in Table 7.2, Data Set 3 and Data Set 4 are created by eliminating features related with traced hashtag. The aim of removing these features is to illustrate the proposed model detects an organized activity in a given tweet set whether it is grouped by a hashtag or not, although the data in campaigns are queried with hashtags.

Table 7.2. Classifier results using all features.

Data Set	**Explanation**
Data Set 1	Data set contains all features
Data Set 2	Data set of principal components of all features after applying PCA
Data Set 3	Data set contains all features, except the ones extracted based on traced Hashtag (such as daily post count of traced Hashtag)
Data Set 4	Data Set contains principal components of features in Data Set 3

Table 7.3. Classifier results for Data Set 1.

	Accuracy	**F-Measure**	**ROC Area Under curve**
Random Forest	0.9972	0.9979	0.9964
Logistic Regression	0.6996	0.8226	0.5
SVM	0.7107	0.8252	0.5372

Table 7.4. Classifier results for Data Set 2.

	Accuracy	**F-Measure**	**ROC Area Under curve**
Random Forest	0.9748	0.9821	0.9606
Logistic Regression	0.9934	0.995	0.991
SVM	0.9951	0.9967	0.9967

Table 7.5. Classifier results for Data Set 3.

	Accuracy	**F-Measure**	**ROC Area Under curve**
Random Forest	0.9946	0.9959	0.9959
Logistic Regression	0.9946	0.9959	0.9930
SVM	0.7107	0.8252	0.5372

Table 7.6. Classifier results for Data Set 4.

	Accuracy	**F-Measure**	**ROC Area Under curve**
Random Forest	0.9756	0.9827	0.9647
Logistic Regression	0.9933	0.9950	0.9902
SVM	0.9951	0.9966	0.9954

7.3. Effective Features

In order to understand the most effective features for detection of organized activity in Random Forest classifiers, features in Data Set 1 and Data Set 3 are analyzed using "ClassifierSubsetEval" attribute evaluator and "BestFirst" search method in Weka.

7.3.1. Data Set 1 Features

In Data Set 1, the most effective features are as below.

- percentages of users with media ratio between 0.6 & 0.9
- variance of all users' media ratio
- percentages of users with friend follower ratio value 1
- percentages of users who have posted tweets between 21 - 50
- percentages of users who have posted tweets between 51 - 100
- percentages of users who have favorite tweets between 1 - 100
- minimum value of tweet counts among all users
- percentage of users who have "averageDailyPostCountRatio" between 51 -100
- minimum value of "averageDailyPostCount" among all users
- maximum value of "averageDailyPostCount" among all users
- minimum value of "averageDailyPostCountInHashtagDays" among all users
- variance of "hashtagPostCount" among all users
- percentages of users who have "averageDailyPostCount" between 6-10
- variance of hourly tweet counts with "VERY_POSITIVE" sentiment among all hour intervals of tweet set

7.3.2. Data Set 3 Features

In Data Set 3, the most effective features are as below.

- percentages of users with media ratio between 0.6 & 0.9
- average of all users' hashtag ratio
- percentages of users with friend follower ratio value 1
- percentages of users who have 0 favorite tweets
- percentages of users who have favorite tweets between 1 - 100
- minimum value of "favoriteCount" among all users
- standard deviation of hourly retweet ratios among all hour intervals of tweet set
- percentages of users who have posted tweets between 21 - 50
- percentages of users who have posted tweets between 51 - 100

7.4. Unsupervised Approach

Unsupervised learning methods are also applied on the training data set in order to test, would it be possible to detect organic and organized activities.

The intent of applying unsupervised algorithms was to understand whether or not extracted features are good enough to separate two different types of behaviors without need of labeling. In order to understand this, "EM", "HierarchicalClusterer", "FilteredClusterer", "MakeDensityBasedClusterer", "XMeans" ,"Canopy" ,"Cobweb", "FarthestFirst", and "SimpleKMeans" clustering algorithms in Weka are applied to training set. However, in none of the results, organic and organized records are grouped clearly in different clusters (see Figure 7.1).

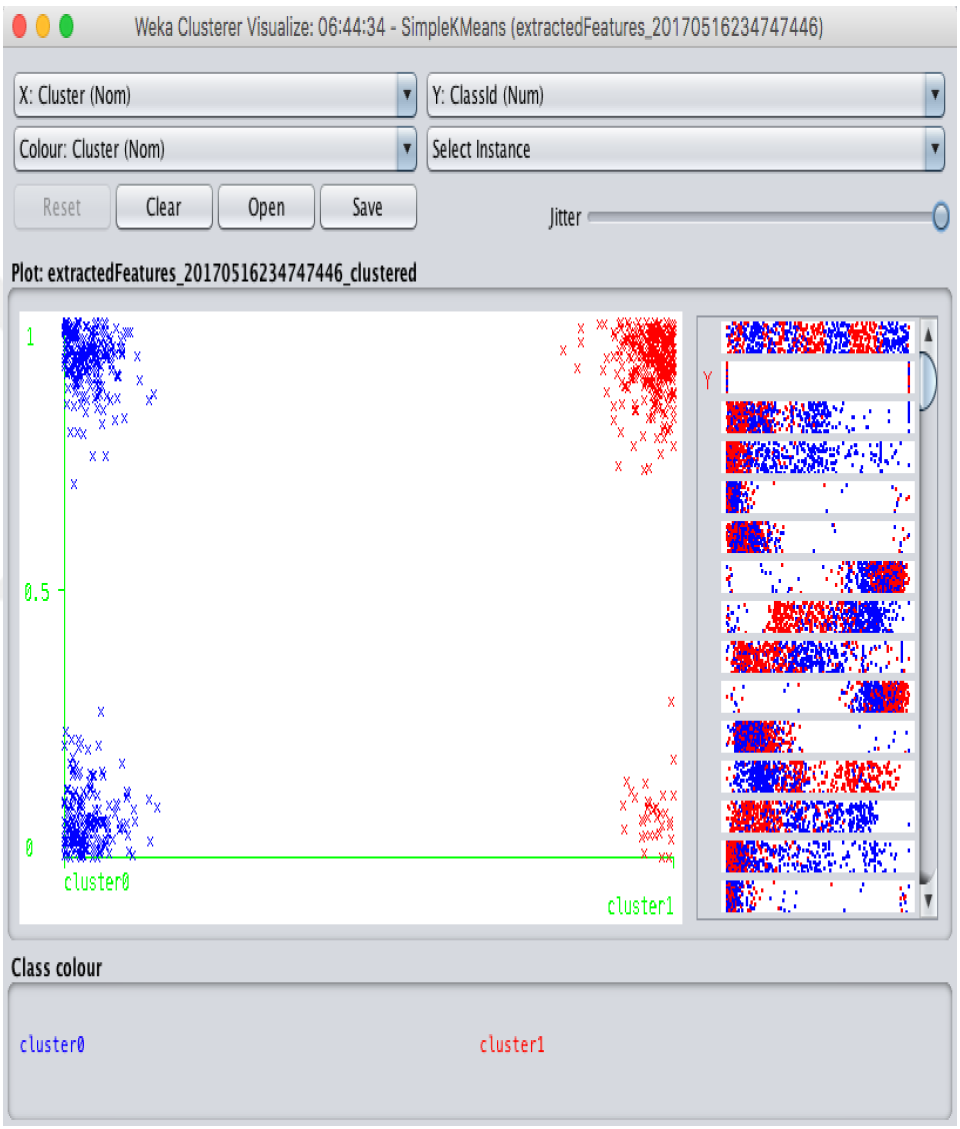


Figure 7.1. Unsupervised clustering of the Training Data.

8. DISCUSSION AND FUTURE WORK

The proposed model in this thesis, can detect organized behaviors in a given set. However, with the increase of bot usages and activity of hired users , in Twitter the behavior patterns for organized behaviors will more likely change within time. Therefore, in order to solve this problem, there is still a lot to do in domain.

The results of classifier models illustrated in Chapter 7 are promising. Furthermore, those results shows that the extracted features are useful for detecting organized activities in a given set. However, despite usage of bootstrapping techniques in order to increase training data set size, the obtained data is not big enough. Therefore, it is more likely that the generated classifier models might give lower results than in Chapter 7, when it is tested with a larger training data set.

During the implementation of the system, at first closeness centrality and similarity of hourly tweets are also calculated. For calculation of closeness centrality, friends and followers of campaign users are retrieved and then friends and followers of retrieved friend, followers are also extracted by Twitter API. In other words, users who have a distance of two hops to campaign users are retrieved. For persistence of these users, Neo4j graph database is used. However, since some users have friend and followers more than 50K, after some point the system started to save most of the users in Twitter ironically. Therefore, calculations of each user's closeness centrality by considering its distance to campaign users become problematic due to data volume, algorithm complexity and Neo4j capability.

Furthermore, the similarity calculation of all campaign users' tweets which are posted in same hour interval was also very costly. Since the similarity calculation of each tweets requires a complexity of $\mathcal{O}(n^2)$, the analysis of campaigns with more than 5.000 tweets resulted to end in two months even if no other analysis is run on the system.

As a result of these, features of closeness centrality and similarity of hourly tweets are removed from the data set although they can be beneficial for detection of organized activities.

Moreover, as it is also mentioned in Chapter 7, a system with MongoDB and concurrent thread are not solely enough for systems which handles big data volumes. In order to detect organized behaviors in real time, big data environment like Cloudera and Hortonworks can be used. Furthermore, since existence of stream data in a real time detection, queue applications like Kafka should be also considered.

For future work, in the short term, it is aimed to add semantic features to the training data set. These semantic features will be obtained by extracting semantic topics from a user's all tweets and calculating semantic distance of topics in those tweets. By this way, a score, which reflects the topic diversity in user's tweets will be obtained. The proposed approach only understands whether there exists a organized activity or not in a given tweet set. However, there can be multiple organized groups, who use the same hashtag in order to achieve their goal. Therefore, it is also planned to develop an approach which understands different organized communities in an organized activity. On the other hand, in the long term, it is aimed to use information flow analysis for detection of organized behaviors.

9. CONCLUSION

To sum up, this thesis focuses on automatically identifying tweet sets that are "organized" or "organic". It is discovered that organized activities can be detected by using supervised learning models generated by Random Forest, Logistic Regression & SVM algorithms. Moreover, cross validated results of these models illustrate that Random Forest gives the highest metrics in all data sets.

Results show that the proposed model also detects organized behaviors in such cases with high accuracy and f-measure. Furthermore, in order to show the generated model is also independent from hashtag related features, classifier models are also tested on data sets, which do not contain features related with the traced hashtag.

During this study, the most important part of this problem was to have a system, which extracts features for training data set. The implementation of this system covered most of the time on this thesis. While implementing this system, lots of performance issues are experienced as a result of processing big data volumes. These experiences showed that in order to implement and design a real time system, which will detect organized behaviors in Twitter, a know-how in the Big Data domain is certainly needed. Systems with high memories need to be used in order to prevent excessive DB input outputs.

Also, it is observed that features like closeness centrality and similarities of tweets are very costly, and it is possible to detect organized behaviors without using them.

Besides from these, in order to understand whether or not the characteristics of extracted features are available for unsupervised approaches, also clustering algorithms are applied on training data set. However, results illustrated that organized and organic records can not be separated clearly as two clusters.

REFERENCES

1. Berger, J., *Nazis vs. ISIS on Twitter: A Comparative Study of White Nationalist and ISIS Online Social Media Networks*, 2016, accessed at June 2017.

2. Fou, A., *Cross Channel Media Costs Comparison CPM Basis*, 2014, `goo.gl/jenRok`, accessed at June 2017.

3. Berger, J. and J. Morga, *The ISIS Twitter Census, Defining and describing the population of ISIS supporters on Twitter*, 2015, `https://goo.gl/LAcwiA`, accessed at June 2017.

4. Bessi, A. and E. Ferrara, *Social Bots distort the 2016 U.S Presedential Election Online Discussion*, 2016, `https://goo.gl/ggJTTb`, accessed at June 2017.

5. Kollanyi, B. and P. N. Howard, *Bots and Automation over Twitter during the Second U.S. Presidential Debate*, 2016, `https://goo.gl/9204dP`, accessed at June 2017.

6. Hunt, E., *What is fake news? How to spot it and what you can do to stop it*, 2016, `https://goo.gl/ZetBt1`, accessed at June 2017.

7. Armstrong, S., *Pizza, politics and pure fiction: the rise of fake news*, 2017, `goo.gl/JpP30s`, accessed at June 2017.

8. Long, S., Z. Wenping, S. L. Stephen and C. K. Ron, *A Critical Analysis Of The State-Of-The-Art On Automated Detection Of Deceptive Behavior In Social Media*, 2012, `https://goo.gl/gEW2Hn`, accessed at June 2017.

9. Cao, C. and J. Caverlee, *Detecting Spam URLs in Social Media via Behavioral Analysis*, pp. 703–714, Springer International Publishing, Cham, 2015.

10. Stringhini, G., C. Kruegel and G. Vigna, "Detecting Spammers on Social Networks", *Proceedings of the 26th Annual Computer Security Applications Conference*, ACSAC '10, pp. 1–9, ACM, New York, NY, USA, 2010.

11. Stringhini, G., C. Kruegel and G. Vigna, "Detecting Spammers on Social Networks", *Proceedings of the 26th Annual Computer Security Applications Conference*, ACSAC '10, pp. 1–9, ACM, New York, NY, USA, 2010.

12. Ratkiewicz, J., M. D. Conover, M. Meiss, A. Flammini and F. Menczer, "Detecting and tracking political abuse in social media", *In Proceedings of the 5th AAAI International Conference on Weblogs and Social Media (ICWSM'11*, 2011.

13. Cao, C., J. Caverlee, K. Lee, H. Ge and J. Chung, "Organic or Organized?: Exploring URL Sharing Behavior", *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pp. 513–522, ACM, New York, NY, USA, 2015.

14. *Classification of group behaviors in social media via social behavior grammars*, Vol. 9097, 2014.

15. Ratkiewicz, J., M. D. Conover, M. Meiss, A. Flammini and F. Menczer, "Detecting and tracking political abuse in social media", *In Proceedings of the 5th AAAI International Conference on Weblogs and Social Media (ICWSM'11*, 2011.

16. Bitly, *Bitly Click API Description*, 2017, `https://dev.bitly.com/index.html`, accessed at June 2017.

17. Howard, P. N. and B. Kollanyi, "Bots, #StrongerIn, and #Brexit: Computational Propaganda during the UK-EU Referendum", *CoRR*, Vol. abs/1606.06356, 2016.

18. Twitter, *About Twitter*, 2017, `https://goo.gl/8nsnwM`, accessed at June 2017.

19. Twitter, *Retweet Description*, 2017, `https://goo.gl/3CwESx`, accessed at June

2017.

20. Twitter, *Hashtag Description*, 2017, `https://goo.gl/Y3uAH6`, accessed at June 2017.

21. Twitter, *Spam Description*, 2017, `https://goo.gl/1hA0yW`, accessed at June 2017.

22. Otte, E. and R. Rousseau, "Social network analysis: a powerful strategy, also for the information sciences", *Journal of Information Science*, Vol. 28, No. 6, pp. 441–453, 2002.

23. Du, D., *Social Network Analysis: Centrality Measures*, 2017, `goo.gl/tAS5qs`, accessed at June 2017.

24. Gasevic, D., *Network Measures*, 2014, `https://goo.gl/iYBJJN`, accessed at June 2017.

25. Pesce, A., *Natural Language Description*, 2013, `goo.gl/DKKOQP`, accessed at June 2017.

26. Manning, C. D., M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit", *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60, 2014.

27. *Text Similarity Description*, 2015, `goo.gl/bTrabA`, accessed at June 2017.

28. Alpaydin, E., *Introduction to Machine Learning*, The MIT Press, 2nd edn., 2010.

29. Wikipedia, *Resampling Description*, `https://goo.gl/HtPzYw`, accessed at June 2017.

30. Adèr, H. J., G. J. Mellenbergh and D. J. Hand, *Advising on research methods : a consultant's companion*, Johannes van Kessel Pub., 2008.

31. OpenML, *10 Fold Cross Validation Description*, 2014, https://goo.gl/Z6UMi1, accessed at June 2017.

32. Cameron, A., J. Gelbach and D. Miller, "Bootstrap-Based Improvements for Inference with Clustered Errors", *The Review of Economics and Statistics*, Vol. 90, No. 3, pp. 414–427, 2008.

33. Kunsch, H. R., "The Jackknife and the Bootstrap for General Stationary Observations", *Ann. Statist.*, Vol. 17, No. 3, pp. 1217–1241, 09 1989.

34. Wikipedia, *Bootstrapping Description*, https://goo.gl/XYu0Rb, accessed at June 2017.

35. Zumel, N. and J. Mount, *Practical Data Science with R*, Manning Publications Co., Greenwich, CT, USA, 1st edn., 2014.

36. Burges, C. J., "A Tutorial on Support Vector Machines for Pattern Recognition", *Data Mining and Knowledge Discovery*, Vol. 2, No. 2, pp. 121–167, 1998.

37. Ray, S., *Support Vector Machines Description*, 2015, https://goo.gl/F1XJQg, accessed at June 2017.

38. Weka, *Weka Description*, https://goo.gl/1oxS44, accessed at June 2017.

39. Spark, *Spark Machine Learning Library Description*, https://goo.gl/XDd13b, accessed at June 2017.

40. MongoDB, *No Sql Description*, https://goo.gl/wo4iFL, accessed at June 2017.

41. Obar, J. A., *Canadian Advocacy 2.0: An Analysis of Social Media Adoption and Perceived Affordances by Advocacy Groups Looking to Advance Activism in Canada*, 2013, https://goo.gl/GOaWxQ, accessed at June 2017.

42. Ghobadi, S. and S. Clegg, ""These Days Will Never Be Forgotten ...": A Critical

Mass Approach to Online Activism", *Inf. Organ.*, Vol. 25, No. 1, Jan. 2015.

43. Edwards, F. and P. N. J. Howard, *Digital Activism and Non Violent Conflict*, 2013, `https://goo.gl/3oKi5V`, accessed at June 2017.

44. Chu, Z., S. Gianvecchio, H. Wang and S. Jajodia, "Detecting Automation of Twitter Accounts: Are You a Human, Bot, or Cyborg?", *IEEE Trans. Dependable Secur. Comput.*, Vol. 9, No. 6, pp. 811–824, Nov. 2012.

45. Silva, S., *Trumpś Twitter debate lead was 'swelled by bots'*, October 2016, `https://goo.gl/LLPHkm`.