Knowledge Discovery For Interpretability and Insight into Network Approaches to Automatic Fake News Detection

Ali Mohammad Mehr
University of British Columbia
Vancouver, BC, Canada
alimm@cs.ubc.ca

Stephen Kasica
University of British Columbia
Vancouver, BC, Canada
kasica@cs.ubc.ca

ABSTRACT

Network-based approaches to fake news detection have resulted in low verification error rates within examples that suffer from threats to external validity. By fake new detection, we mean a supervised learning approaching to categorizing conversational threads as being binary rumor or non-rumor. Given the disproportionate number of fake news article and the resources available to determine their veracity, automated approaches to classification are one compelling approach to solving this problem. Our project concerns the evaluation and engineering of many features for network approaches to automatic fake new detection on social networks. We provide detailed explanation and evaluation for mining features from data available through the Twitter API.

CCS Concepts

•Fake news detection → **Fake news analysis;** **•Fake News** → *Feature Analysis;* **•Social Networks** → Rumour Detection; Graph-based Features;

Keywords

ACM proceedings; LaTeX; text tagging

1. INTRODUCTION

In 2018, social media has become an increasingly important, if not primary, outlet for the propagation of daily news in society. In the United States, about two-thirds of American adults say they at least occasionally get news on social media [8]. The propagation of information through the news machinery of the 20th Century was curated and controlled by professional journalists and editors, for better or worse. In the 21st century, social networks, such as Twitter and Facebook, have given anyone a platform to tell their stories, for better or worse. It seems as though it has been for the worse given the enormous volume of factitious stories posing as legitimate news. In the 2016 U.S. Presidential election 155 fake stories tied to the election were shared on Facebook a total of 37.6 million times, resulting in 760 million instances of users clicking through and reading a fake news story [1]. Not only is misinformation so prevalent on social networks, but it often outcompetes real news when measured by user engagement. This phenomenon did not originate with the 2016 U.S. Presidential Election, was brought into the national dialogue at the time when a BuzzFeed News analysis discovered that the top fake election stories generated more total engagement on Facebook than the combined, top election stories from 19 major news outlets [9]. Interesting work has already been done on machine learning-based network approaches to the containment and mitigation of misinformation through a social network, be it an undirected or directed graph such as Facebook and Twitter, respectively. This work makes the following contributions:

- A throughout analysis of the features extractable from data available through Twitter's Public API and their importance in various machine learning classification models.
- Tabularized datasets of Tweets surrounding breaking news event, and
- Open-source code concerning data cleaning and feature extraction of Twitter data presented in Jupyter Notebooks for public use on GitHub.

2. RELATED WORK

Classifying news fake by any approach other than analyzing the content of deceptive messaging sounds impossible, yet an increasing body of research claims to have accomplished just that with surprisingly accurate results. This method of classification is called a Network Approaches in the literature and incorporates machine learning techniques for transforming message metadata, author metadata, and structured networked data into meaningful features in trained classifiers [6]. These machine learning technique are generally supervised and include both probabilistic and linear classifiers, random forests, and meta learner classifiers. For example, CREDULIX is a fake news classifier based on Bayesian classifier working on the intuitive assumption that the posterior probability of a user sharing misinformation across a social network depends upon the prior probability of the user sharing fact-checked information. The authors claim that this algorithm has a validation error of 0.01, with no Type I errors [2]. In addition to probabilistic classification, some work has been published on classification algorithms. In a case study on activity on Twitter

© 2018 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

following a 8.8 magnitude earthquake that off the central coast hit Chile in February 2010, a 2013 paper by Castillo and colleagues evaluated Network Approach features used in random forest, logistic regression, and meta learning. Each classifier reported a validation error of less than 0.4, with logistic regression being the lowest at 0.32 [4] [5]. When selecting features for their classifiers, authors identified the following features as being the most useful for their classifiers: number of users the tweet author follows; fraction of tweets containing a URL, the most frequent URL in the thread, URLs to the top 10,000 most visited website, first-person pronouns, third-person pronouns, user mentions, a question mark, an exclamation mark, containing smiling emoticon, positive sentiment, negative sentiment; average tweet length; maximum depth of the propagation tree; and the average number of tweets posted by authors of the tweets in the topic in the past. In 2017, Buntain et al. performed feature analysis on three datasets, not including the dataset used by Castillo et al., of accuracy assessments for events on social media [3] and compared predictive performance on models training on data classified by users on Mechanical Turk and journalists. This paper also includes a section on feature selection and analysis which largely inspired this project. This paper identified the following relevant features: proportions and frequency of tweets sharing media, account age, author friends, frequency of smiley emoticons, proportions of tweets sharing hashtags, proportions of tweets containing ïňĄrst- and third-person pronouns, proportions of tweets expressing disagreement, and the slope of the average number of authorsâĂŹ friends over time. The fact that both of these studies found similar features that most contributed to the accurate classification in their machine-learning classifiers suggests that there may be a general set of features for network approaches to automatic fake news.

2.1 The PHEME Rumor Non-Rumor Dataset

The PHEME Rumor Non-Rumor dataset contains a collection of tweets posted during 9 breaking news event. First, on August 9, 2014, 18-year-old, African American Michael Brown by a Darren Wilson, a 28-year-old white Ferguson, Missouri, police officer. On October 22, 2014, a shooting at Parliament Hill in Ottawa, Canada, resulted in the death of two, including the perpetrator. On January 7, 2015, two armed gunmen killed 12 people injuring 11 others at the French satirical weekly newspaper Charlie Hebdo in Paris. On March 24, 2015, a flight operated by Germanwings from Barcelona to Dusseldorf crashed 100 kilometers north-west of Nice, France, killing all 150 people aboard. Finally, on December 15, 2014, a lone gunman held 18 people hostage in a cafe in Sydney, Australia. The two day standoff ended in three deaths, including the perpetrator, and four non-fatal injuries. For each of these events, the conversation is recorded around tweet threads, a quasi-tree with the source tweet at the root and reply tweets as the children. Each thread is classified as either rumor or non-rumor by a team of journalists who partnered with researchers [10] [11]. Given the high volume of information around these event, tweet threads in the PHEME dataset possessed the highest number of retweets. Previous dataset on rumor classification, were gathered with the rumor known a priori, where tweets were harvested after a news event was identified to a rumor. Tweets in the PHEME dataset were gathered as soon as a news event occured, and tweets were labeled as rumor or non-rumor later. We choose to use the PHEME dataset because the threat to external validity is less with a classifier trained on this dataset than with ones gathered a priori. If a event has already been identified as rumor, then there is no need for a machine learning classifier.

3. BACKGROUND NOTION (OR PRELIMINARIES)

3.1 Classification

Machine learning is nowadays being used in many classification problems. Problems which have lots of features, force us to think about machine learning rather than finding exact formulas for different problems. Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y). Because of the no free lunch algorithm, in order to predict any concept, we need to find the best model that models the actual characteristics of that concept. According to [7], who tested 179 classifiers on 121 real-world data sets, the classifiers that are most likely to to be the best classifiers are Random Forest. The second best model, according to them, was SVM with Gaussian kernel. Based on these previous research, we also think that starting from these classifiers is the best option to start predicting rumour news. Another reason why SVM is more popular among all the models is that the weights generated by the SVM in the process of learning helps us do feature selection: The features whose weights are larger in size have more effect on the classification and therefore are the most relevant features in the explored classification problem. Using L1-regularization instead of L2-regularization in SVM will result in sparse weights which will help up even more in feature selection because most of the selection process is done by the model itself.

3.2 Latent Factor models

The main idea of Latent Factors is to reduce the dimensionality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent. The dimensionality reduction achieved by Latent Factor models are useful in many ways. One of the most important usage of Latent factor models is to visualize the data. Another usage is to find the hidden features that result from combining different features. If a latent factor model can distinguish between some groups of data, using those latent factor models will make way to a very simple classification of the test samples. With the hope of achieving good separation between rumour and non-rumour news items, we will try to extract the latent factors from the dataset using different Latent Factor models.

4. TECHNICAL SECTIONS (RESULTS)

4.1 Evaluating Temporal Distribution with Swarmplots and Density Plots

Our project started with the Germanwings Crash dataset because it was the smallest dataset. Thus, it had the fastest processing time in our data pipeline as we experimented with extracting different features from the Twitter metadata. When performing univariate selection, we observed that several features related to the time of the tweet for this particular dataset highly correlated with whether or not it was classified as rumor or non-rumor. To further examine the distribution of rumor and non-rumor tweets overtime, a swarm plot was adapted to fit out needs, as seen in Figure 1. The results is a sort of discretized distribution of tweets overtime, encoding whether or not the tweet was classified as rumor by color. The legend to the top right is interactive, and clicking one of the labels toggles the opacity of tweets with this label from 0.1 to 1.0. The circles in the plot are also interactive. Hovering the cursor over an individual circle renders a pop-up windows containing the individual tweet, time of the tweet, and author's Twitter handle. By incorporating the tweet text into a detail view of the data, users interacting with this visualization also get a sense of why a particular tweet was labeled as rumor or non-rumor. Upon interpreting this chart, it appears that there were more non-rumor tweets immediately following news of the crash, followed a smaller band of non-rumor tweets sandwiched on both sides by rumor tweets. Although for the Germanwings Crash event, the time at which the tweet occured appears to be useful feature in predicting whether or not it's fake news, in general this feature did not apply well to other events. When comparing the distribution of rumor vs. non-rumor tweets, normalized by the number of days following the event, we see mostly noise and little signal, as seen in Figure 2. In conclusion, if there was a clear pattern to the temporal propagation of rumor vs. non-rumor tweets, then we should see a different patterns between rumor and non-rumor tweets but a similar pattern with in rumor category and between events. Although the ottawa shooting and the sydney hostage crisis both have similar patterns between rumor category, making it unlikely that the dataset could be separated into rumor category based on this feature. Therefore, we reject the hypothesis that news events share a common temporal propagation pattern.

4.2 Violin Plots for Feature Analysis

Similar to box-and-whisker plots, a violin plot shows the distribution of quantitative data. Unlike box-and-whisker plots, violin plots show the kernel density estimation of the underlying distribution. For feature analysis, we plot the normalized distribution of continuous-valued features, separated by rumor and non-rumor. This view allows us to compare the distribution of rumor and non-rumor tweets across many distributions. Interpretation of these plots of feature selection entails looking for asymmetries. Ideally, a good feature for a classifier have two distributions so different that a t-test between the two would result in a p-value $<$ 0.05. The violin plot below in Figure 3 shows the distribution of features identified as being highly ranked by feature selection algorithms.

4.3 Investigating Conversational Thread Structure

Admittedly, network approaches to automatic fake news detection on Twitter are heavy on tweet and user metadata but light on actual network structures. Twitter's API does not provide easy accesses to the follower-followee network structure, remember Twitter's a directed social network where any user, say Bob, can follow any other user, Alice, and Alice does not have to follow Bob. The closest feature to actual network structures available on Twitter is follower count. This feature is synonymous with node in-degree for a graph G = (V,E) such that V is the set of all users and E is a set of edges representing a follow relationship. However, the post-processed Twitter data that is included in the PHEME dataset does include some network structures, namely the thread conversation. When comparing the number of source tweets, user who original post to Twitter, and the number of non-source tweets, replies to a source tweet, we see that the number of replies greatly out numbers the source tweets. As Figure 4 shows, there are many more tweets in reply to a tweet than source tweets. This may mean that there exists some interesting features made out of the structure of these Twitter conversations. Tweets in the PHEME dataset are organized into threads, proxies for conversations threads. When one tweet replies to another, it constitutes a thread. Since the vast majority of tweets in this dataset are replies to other tweets, we examined the distribution of thread length, the number of individual tweets in a thread. In order to get a better idea of the conversational structure, we graphed user-to-user interactions in Figure 6. Because the dataset is so large, we randomly choose 50 threads to display (although the seed number is set for reproducibility and demonstration purposes). We start to see some interesting patterns in the user-to-user response network in a thread. One of the most interesting interpretations is that mix of orange rumor and dodger blue non-rumor in some of the components. Tweet threads are pre-classified into rumor and non-rumor; however, when looking at user-user networks we see rumors who have engaged in rumor behavior interacting with users who have engaged in non-rumor behavior. Thus, transcending Twitter threads. Some of the structures can be quite complicated. Figure 7 is the largest structure available to the dataset.

4.4 Latent Factor Models

In this section, we train different latent factor models for events in the PHEME dataset, trying to see if latent factor models can separate rumour and non-rumour threads. We will also train these latent factor models on all the events to see if they can differentiate between different events. We need thread-level features to train these models on, so we use the thread-level features extracted by a utility code that we have and saved in csv files. The result of training and transforming the thread-level samples in germanwings-crash event is shown in the figure 8. It can be seen that none of these latent factor models can distinguish between rumour and non-rumour events. Seems like based on the features that we have extracted, the use of a latent factor model to model the differences between rumour and non-rumour threads is not possible. Next, we train the latent factor models on all the samples of the datasets. Now, we can have two sets of plots for the latent factors. The first set of the plots show samples with different colors for rumour and non-rumour samples (figure 9) and the second set of the plots show samples of the same event with the same color (figure 10). Looks like TSNE is able to distinguish between different events in the dataset by looking at our extracted features. This can have some positive and negative interpretations. The negative interpretation is that if different events are distinguishable, generalizing the models trained

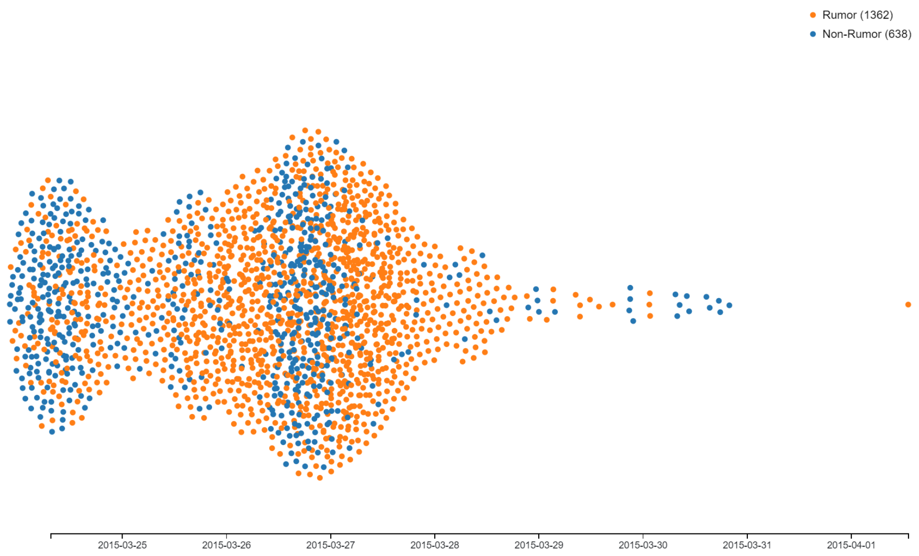


Figure 1: This swarm plot of tweets surrounding the Germanwings Crash encodes when the tweet was created and whether it was labeled rumor or non-rumor. It's also interactive and hovering the cursor over an individual circle in the plot reveals the exact tweet's text, the time it was created, and the author's Twitter handle.

for some events to predict on new events is hard. This result is also seen in the linear classifiers that we use in the classification models section: The linear models trained on some events can not detect rumour in other events.

4.5 Prediction Accuracy and Classification Models

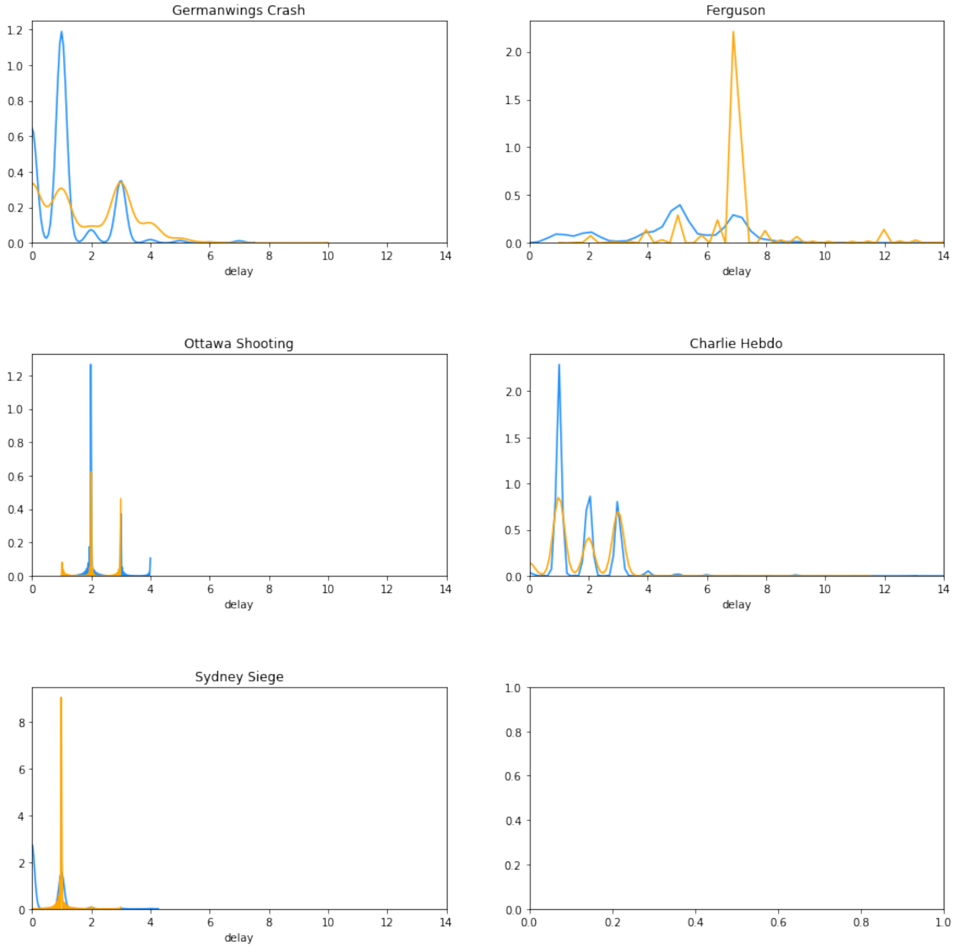
In this section we try to train different classification models on the samples of different events. We have two ultimate goals for running these classification models. Firstly, we want to analyse the generalization accuracy of different well-known models on rumour detection - we want to see how well these models perform when trained on some events, but tested on other new events. This is important because we want to detect new rumours that our models have not seen any samples of before. Secondly, we want to do feature selection based on the weights of the linear SVM models - with L1 or L2 regularization. In this section we will only focus on the first goal mentioned above. We will discuss the second goal in the next section - Univariate Feature Selection. The training and testing accuracy of different models that we used in this section can be found in Table . According to the results, training most of the models on 75% of the samples of an event is enough to make a good prediction on the rest of the samples of that event. This means that rumour items in single events have common distinguishable features from non-rumour items in those events. On the other hand, based on the results in this section, we can see that training models - esp. linear models - on some events and testing them on other events results in a very low test accuracy (The generalization accuracy is very low) . The fact that TSNE could distinguish different events in the Latent Factor Models section and the fact that the generalization accuracy is very low, shows us that samples from different events fall in different areas of the feature space using the features that we have in this paper.

4.6 Univariate Feature Selection

As mentioned in the previous section for our goals, in this section, we will try to do feature selection based on the weights of the Linear SVM model. We choose the most important features as the features that have the 15 most negative weights in the Linear SVM and the 15 most positive weights in the Linear SVM. These features are shown in the Figure 11.

5. FUTURE WORK

There is a lot to be done in fake news detection until there is a solid framework where fake news is detected as soon as possible before it is spread among many people. The most important part of any classification model is the data set behind it on which it is trained. We believe that creating and maintaining a large data set which has lots of samples of fake and real news items will be the most important step in fake news detection. The mentioned data set needs to have lots of different features measured. Based on our findings, it seems like different events have different characteristics and fall in different areas of feature space if we only use the features that are extracted in this paper. This calls for extraction of more features, some of which are really difficult unless we use Neural Networks. We believe that using neural networks as models to predict rumour might have more generalization accuracy. There exist some Latent factor models which are based on neural networks - e.g. encoder-decoder neural networks. These latent factor models might also be able to distinguish rumour and non-rumour items. Indeed, we believe that using neural networks as classifiers or as latent factor models might result in interesting results. As it can be seen, some models seem to have higher test accuracy than others in the rumour detection problem. We believe that the characteristics of this problem is better modeled by some of the models, so a new area of research might focus on why some models perform better than others in fake news


Germanwings Crash

Ferguson

Ottawa Shooting

Charlie Hebdo

Sydney Siege

Figure 2: The distribution of tweets following the date of occurrence for each breaking news event in the PHEME dataset. The variable delay represents the number of days since the occurrence of the news event. The kernel density estimation for rumor tweets is in orange and non-rumor tweets are in Dodger Blue. The sixth plot is intentionally left empty.

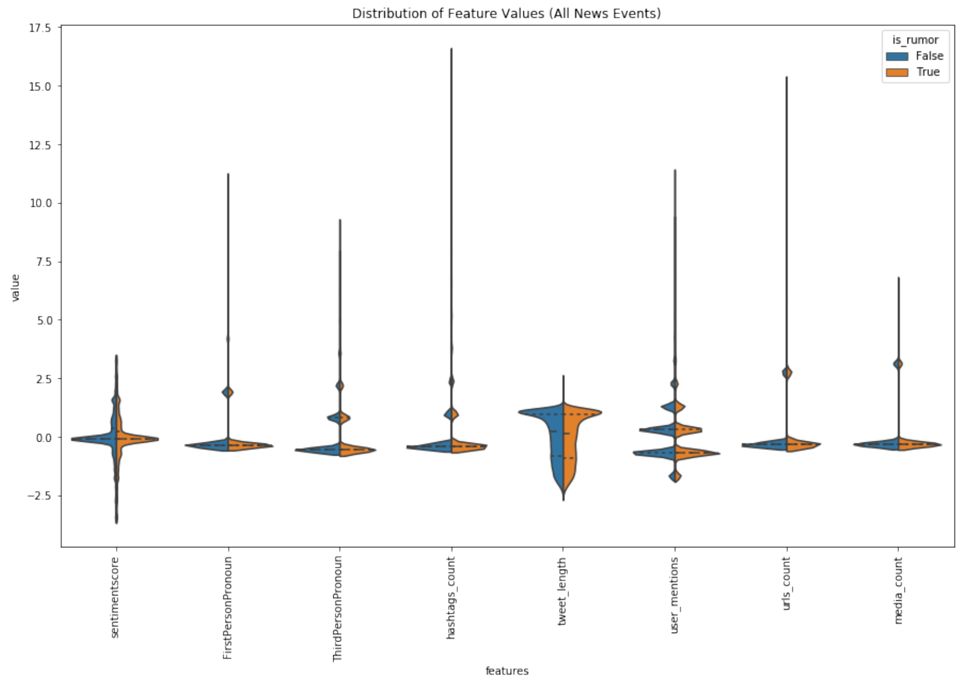
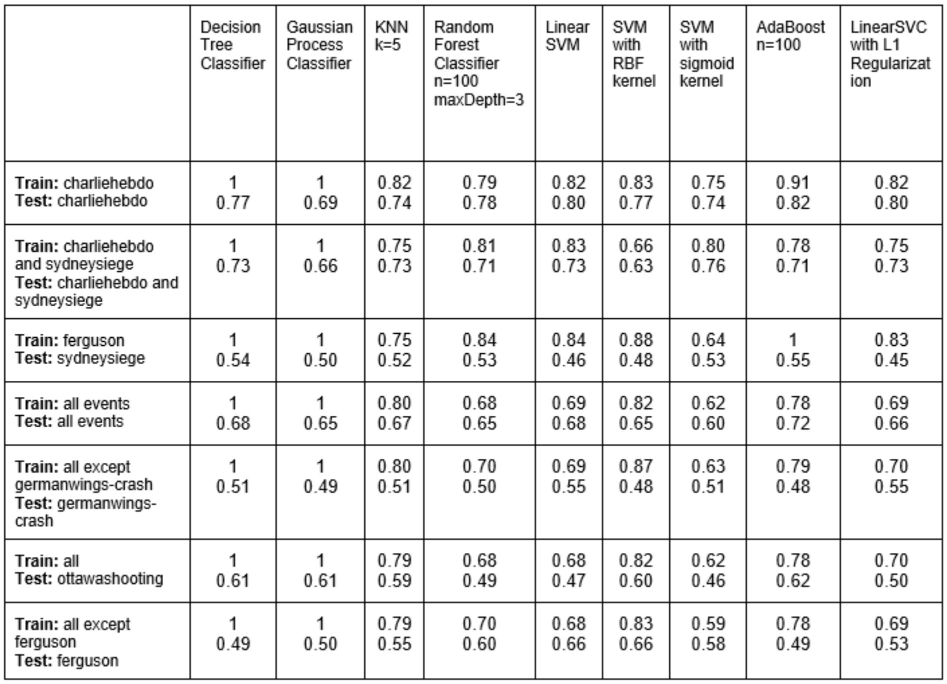


Figure 3: A violin plot of sentiment score, first-person pronoun, third-person pronoun, hashtag count, tweet length, user mentions, urls_count, and media count when combining events. Each of these "violins" looks essentially symmetrical, suggesting that these features would be useful in classification.

Table 1: A table containing the training and testing accuracy of 9 models trained and tested on different parts of the PHEME dataset. In each cell, the top value is the training accuracy and the bottom value is the test accuracy(the higher the better). The event samples were split 75% training data and 25% test data in the test cases where the training and test sets contain the same events.



	Decision Tree Classifier	Gaussian Process Classifier	KNN k=5	Random Forest Classifier n=100 maxDepth=3	Linear SVM	SVM with RBF kernel	SVM with sigmoid kernel	AdaBoost n=100	LinearSVC with L1 Regularization
Train: charliehebdo **Test:** charliehebdo	1 0.77	1 0.69	0.82 0.74	0.79 0.78	0.82 0.80	0.83 0.77	0.75 0.74	0.91 0.82	0.82 0.80
Train: charliehebdo and sydneysiege **Test:** charliehebdo and sydneysiege	1 0.73	1 0.66	0.75 0.73	0.81 0.71	0.83 0.73	0.66 0.63	0.80 0.76	0.78 0.71	0.75 0.73
Train: ferguson **Test:** sydneysiege	1 0.54	1 0.50	0.75 0.52	0.84 0.53	0.84 0.46	0.88 0.48	0.64 0.53	1 0.55	0.83 0.45
Train: all events **Test:** all events	1 0.68	1 0.65	0.80 0.67	0.68 0.65	0.69 0.68	0.82 0.65	0.62 0.60	0.78 0.72	0.69 0.66
Train: all except germanwings-crash **Test:** germanwings-crash	1 0.51	1 0.49	0.80 0.51	0.70 0.50	0.69 0.55	0.87 0.48	0.63 0.51	0.79 0.48	0.70 0.55
Train: all **Test:** ottawashooting	1 0.61	1 0.61	0.79 0.59	0.68 0.49	0.68 0.47	0.82 0.60	0.62 0.46	0.78 0.62	0.70 0.50
Train: all except ferguson **Test:** ferguson	1 0.49	1 0.50	0.79 0.55	0.70 0.60	0.68 0.66	0.83 0.66	0.59 0.58	0.78 0.49	0.69 0.53

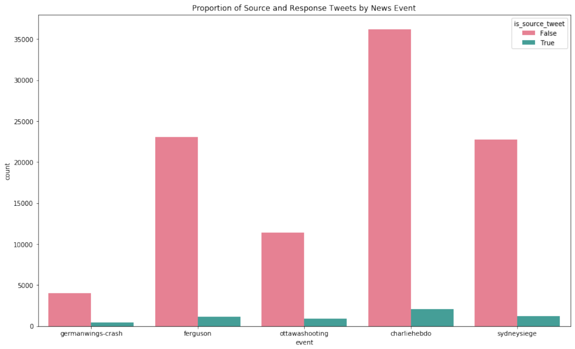


Figure 4: the distribution of source and reply (non-source) tweets by news event.

detection and based on these studies, try to find the optimal model that best expresses the characteristics of fake news detection problem. The dataset that is used in this paper has many information about the creator of the main tweet and the people who replied to the tweet, but it also lacks lots of information about the graph structure of the paths the fake news took to reach to different people. It also does not have some information about how different users reacted to these tweets. A simple feature such as the number of the users who followed up on the replies of a tweet might help detect fake news better. Therefore, trying to include some graph-related features might help the fake news detection problem.. Our underlying assumption in this feature analysis is that there exists a linear relation between each of these features and whether or not a tweet is labeled as rumor or non-rumor. Another interesting direction to explore would be non-linear feature analysis using non-linear regression for classification.

6. REFERENCES

[1] G. M. Allcot, H. Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, 31(2):211–236, Spring 2017.

[2] O. e. a. Balmau. Limiting the spread of fake news on social media platforms by evaluating users' trustworthiness. *TUGboat*, 2018.

[3] G. J. Buntain C. Automatically identifying fake news in popular twitter threads. *IEEE International Conference on Smart Cloud (SmartCloud), New York, NY,*, pages 208–215, 2017.

[4] P. B. Castillo C, Mendoza M. Predicting information credibility in time-sensitive social media. internet research. *ACM Trans. Program. Lang. Syst.*, 23:560–588, 2013.

[5] e. a. Conroy, N.J. Automatic deception detection: Methods for finding fake news. 52, 2015.

[6] N. J. Conroy, V. L. Rubin, and Y. Chen. Automatic deception detection: Methods for finding fake news. 2015.

[7] S. B. D. A. Manuel FernÃąndez-Delgado, Eva Cernadas. Do we need hundreds of classifiers to solve real world classification problems? 2014.

[8] S. E. Matsa, K. News use across social media platforms 2018. 2018.

[9] C. Silverman. This analysis shows how viral fake election news stories outperformed real news on facebook. 2017.

[10] M. L. Zubiaga, Arkaitz and R. Procter. Learning reporting dynamics during breaking news for rumour detection in social media. 2016.

[11] P. R. W. S. H. G. T. P. Zubiaga A, Liakata M. Analysing how people orient to and spread rumours in social media by looking at conversational threads. 2016.

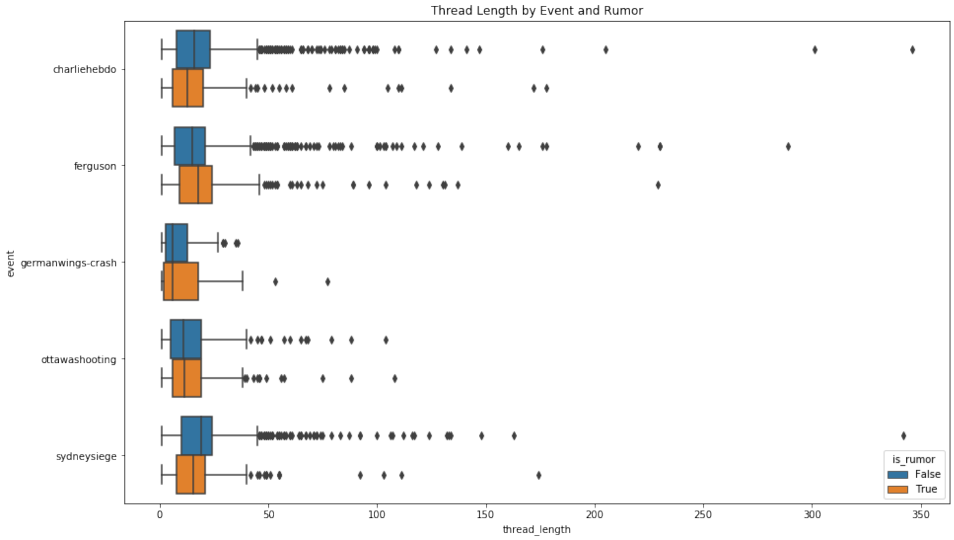


Figure 5: A box plot of thread length for different events

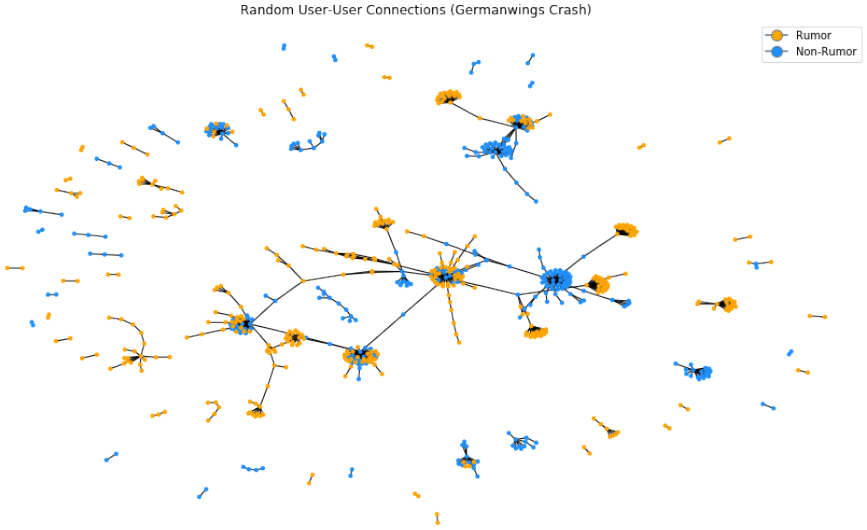


Figure 6: A graph showing user-user connections in geranwings-crash event

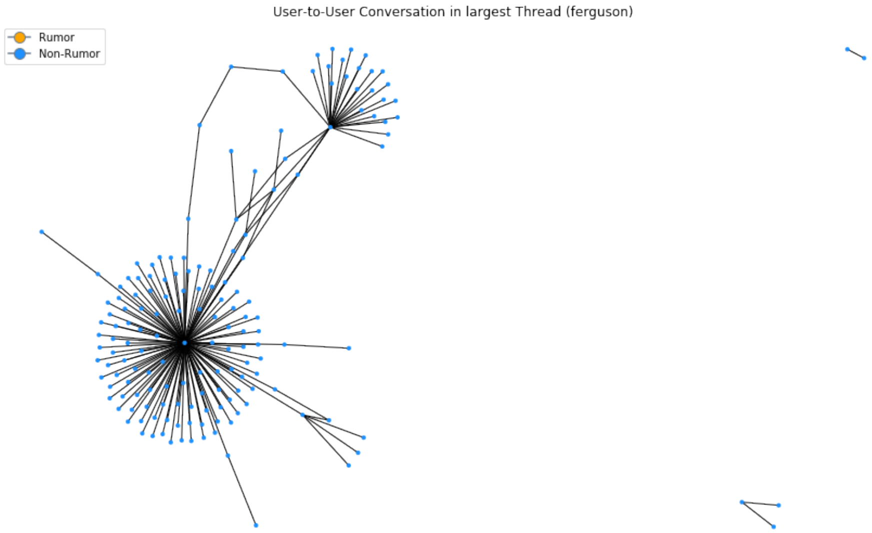


Figure 7: The largest user-to-user conversational thread in the dataset belong to the Ferguson event and includes entirely non-rumor users. Due to a technical bug, arrows along the edges are not rendering, but this is a directed graph.

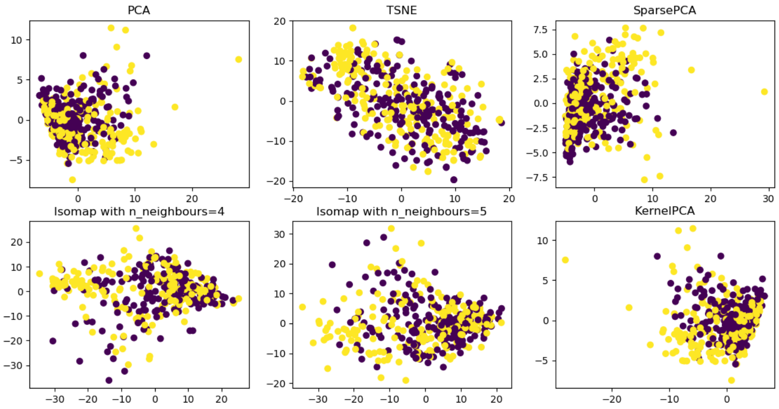


Figure 8: Scatter plots containing the latent factors generated by different latent factor models. The yellow dots are non-rumor threads while the dark purple points are the rumour threads.

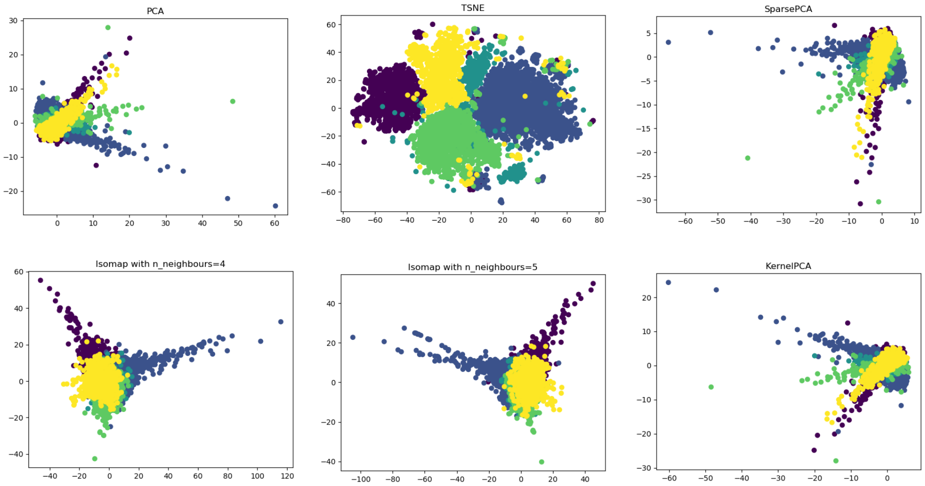


Figure 9: Latent factor models on all the samples of the PHEME dataset. The yellow dots are non-rumor threads while the dark purple points are the rumour threads. As it can be seen, none of the tried latent factor models can distinguish between rumour and non-rumour samples.

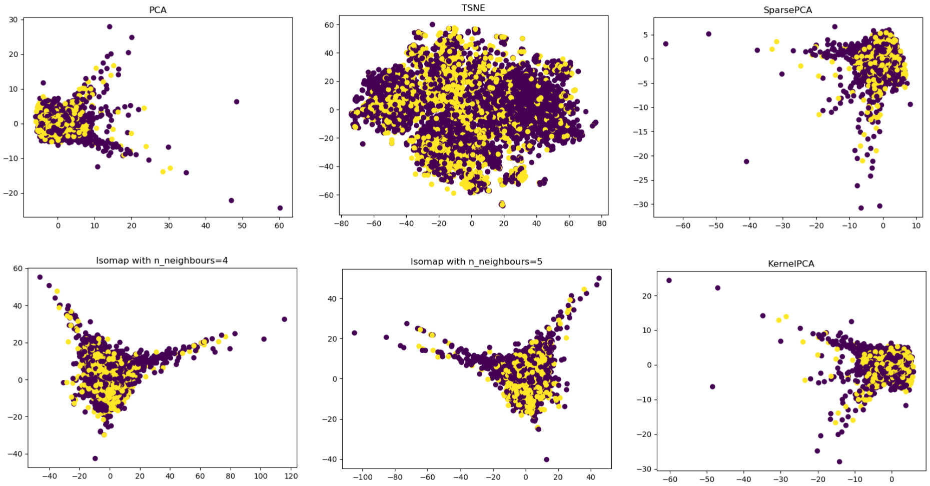


Figure 10: Latent factor models on all the samples of the PHEME dataset. Different colors show different events. As it can be seen, TSNE has been able to identify the events very well.

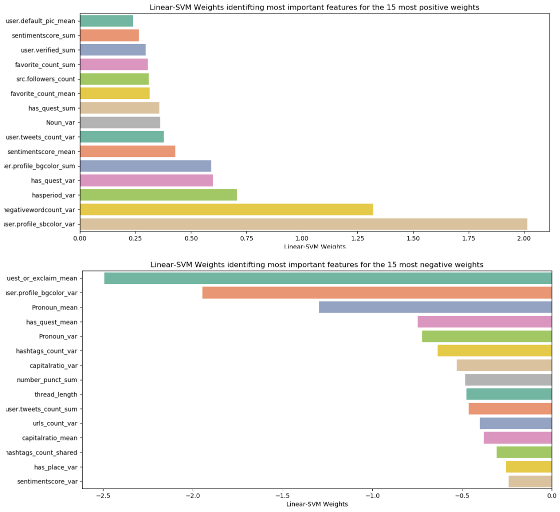


Figure 11: Features that have the 15 most positive or the 15 most negative weights in a Linear SVM trained on the charliehebdo event. We believe that these features are the most relevant features in rumour detection in Twitter.