# A Proposed Model for Preventing the spread of misinformation on Online Social Media using Machine Learning

**Shobha Tyagi [1], Adarsh Pai [2], Jeson Pegado [3], Ajinkya Kamath [4]**

*[1,2,3,4]Department of Computer Engineering, St. Francis Institute of Technology, Mumbai, India*
*[1]shobhatyagi@sfitengg.org, [2]paiadarsh.a@gmail.com, [3]jesonpegado@gmail.com, [4]ajinkyakamath6@gmail.com*

***Abstract: With more than 71% of internet users using Online Social Media (OSM), it has become an important platform for people to share ideas, information and various forms of expressions. However, there is no guarantee about the credibility of the information i.e. how legitimate is the information due to the use of crowd sourcing and absence of any central moderation. This makes it easier for malicious users and some anti-social elements to circulate rumors and create panic among the public, particularly during any real time incident or a disaster by generating fake content. Among the OSMs, the most popular micro-blogging website, Twitter, becomes an easy target for malicious users to spread misinformation having a wide variety of crowd from general public to celebrities, politicians and even large organizations. The system aims to detect such misleading information on Twitter and provide possible measures that can be adopted by the social media company to prevent the spread of misinformation and by the users who contribute to the spread without verifying the veracity of the content.***

***Keywords: Fake news; Machine Learning; Text Analysis; Twitter***

## I. INTRODUCTION

The paper proposes a machine learning solution to the problem of misinformation on OSMs, specifically on Twitter. Among the OSMs, Twitter is a microblog where users post messages called 'tweets', of maximum length 280 characters (which was revised in 2017 from the earlier limit of 140 characters). The two most important terms involved with Twitter are 'follower' and 'followee'. If a user wishes to get messages from another user, they have to become their 'follower' and the one being followed becomes the 'followee' of that user. This typical 'follower-followee' relationship that exists in Twitter, facilitates quick information flow over the social network. Moreover, there is another feature in Twitter called 're-tweet' that facilitates even more faster spread of information. A re-tweet is like a repost or forward of a tweet by another user. The terms misinformation, fake news or rumor are all used to indicate some unverified assertion which originates from one or more than one sources and gradually spreads over the social network. Since most trending topics on Twitter are news related, it becomes a popular source of news among the social media users, and other internet users as well. Hence it can be understood as to why Twitter becomes the favorite target for malicious users to spread misinformation as compared to other OSMs and that how important it is to stop this spread. We propose to design a system which will be able to detect misinformation on Twitter, using Data mining and Machine Learning algorithms. Once the veracity of any information is understood, suitable measures can be adopted by the social media company (such as which accounts to be suspended or to revise any verified account that is directly or indirectly involved in spamdexing) and also the users who contribute to the spread by sharing random news without verifying the veracity of the information.

The remainder of this paper is as follows. Section II discusses about the literature that is reviewed for this work. Section III identifies the challenges faced in the design and implementation of the proposed system. Section IV specifies the problem definition while Section V describes the proposed system methodology along with the tools and algorithms used in preventing the spread of misinformation on Twitter. Sections VI and VII accentuate on the performance evaluation parameter and the experimental setup required for the system. Section VIII finally concludes the study and specifies the scope for future work.

## II. LITERATURE SURVEY

Raveena Dayani et al. [1] analyzed Twitter data by considering parameters like the date when the tweet was posted, the user id, content of the tweet, tweet label, tweet id and some other related features. The Twitter data was collected using the Twitter Search API and stored in MySQL database. The authors used their own pre-processing algorithm before applying the classification algorithms. The algorithms used for classification include K-Nearest Neighbor (k-NN) and Naïve Bayes. In case of k-NN, the Euclidian distance was calculated over the user-based features. However, the prediction accuracy for endorses was 73.8% and that for denies was as low as 40.9%. The authors justify the low accuracy of k-NN by the fact that the User based features had actually no correlation with the rumor detection. To apply Naïve Bayes' algorithm the events that were considered were the word frequencies present in the tweets. Two important categories of factors that were considered were the User based factors (like the time when user created the account, the number of followers and followees, the number of tweets posted by the user and total number of favorites obtained for each user) and the Content

**978-1-5386-9346-9/19/$31.00 ©2019 IEEE**

based factors. The Naïve Bayes method had more prediction accuracy than the k-NN.

Sardar Hamidian et al. [2] performed the analysis with three data sets-Obama and Palin, and a mixed data set (MIX) that consists of all the data from selected five rumors. They considered number of parameters to achieve a better prediction accuracy like lexical features, unigrams, bigrams, parts of speech, sentiments, emoticons, replies, re-tweets, user id, hash tags and the time it was tweeted. Different levels of preprocessing such as lemmatization, removing punctuation, lowercasing and removal of stop words were applied to the contents of tweets. The authors concluded that the accuracy is not improved much from applying preprocessing, but instead may result in loss of some valuable information. Classification was done with the help of J48 Decision Tree and WEKA platform for training and testing.

Qiao Zhang et al. [3] proposed the rumor detection as a binary classification problem, using an automatic rumor detection classification method which is based on the combination of new proposed implicit features and shallow message features. Shallow features are those that are usually extracted from basic user or content attributes, whereas implicit ones are generated by mining deep information from user or content. The proposed system was a three-step process that involved data cleaning, feature extraction and training the model. The content based implicit features considered were regarding the popularity orientation, sentiment polarity, opinion of comments and internal-external consistency. Internal-external consistency provides interrelation between message content and the content of the corresponding external page. The message is less likely to be a rumor if they are more relevant. User based implicit features like social influence, opinion re-tweet influence and match degree of messages was considered. Support vector Machine (SVM) was used for classification. One of the important conclusions made by the authors was that the user credibility is an important factor that directly or indirectly impacts the credibility of the information.

Aditi Gupta et al. [4] used the Twitter Streaming API. The two events that were considered for analysis were the Boston Marathon Blasts (2013) and the Hurricane Sandy (2012). To analyze the temporal distribution of tweets, the number of tweets posted in each hour after the event occurred were considered. One of the important conclusions made was that fake content propagates faster than the real content and occurs much during the very beginning of the event. They not only analyzed the posted tweets but also the tweets from suspended users as well. User based features like the number of followers and following were also considered. Naïve Bayes and Decision Tree were used for classification of which Decision Tree provided higher prediction accuracy of 96.65% as compared to Naïve Bayes which provided a prediction accuracy of 91.52%.

Vahed Qazvinian et al. [5] performed analysis in two steps - extracting tweets about the controversial aspects of the story using Twitter Search API and identification of users who believe the misinformation versus who refuse or question the rumor (Belief classification). Over 10,400 tweets were annotated. To calculate the annotation accuracy, 500 annotations were annotated twice and the comparisons of the annotations was done using the Kappa coefficient. The goal of first task was to classify as rumor and non-rumor while that of second task was to use the tweets that are marked as rumorous, to identify users that trust the rumor versus users those who deny or question it. The first task comprised of building various Bayes classifiers for high-level features and then learning a linear function of the classifiers while second task involved classification. Content-based (like lexical patterns, parts of speech, unigrams and bigrams), Network-based and Twitter specific features were considered. Also, hashtags, which are an important terminology in Twitter, were analyzed to understand whether those used in rumor related tweets were similar or different from the other tweets. Bayes classifier was again used for Belief classification.

Carlos Castillo et al. [6] collected data using Twitter monitor which keeps track of sharp increase in frequency of set of keywords found in every burst of messages. The tweets were then classified into two labels- news and chats. Different features like Message-based (length, symbols, hash tags, re-tweets), User-based (registration details, age, number of followers, number of tweets), Topic-based (sentiments of tweets, URLs) and Propagation-based (building a propagation tree from the re-tweets of the message) were considered. Classification algorithms used were SVM, Decision Tree and Bayes network of which, Decision Tree provided best results.

## III. CHALLENGES IDENTIFIED

The following challenges were identified:

### *A. Deciding the best classification algorithm*

Our research and study on various machine learning algorithms brings us to the conclusion that Support Vector Machine (SVM) may not help much in case large number of features are used for classification while Decision Tree and Random Forest might sometimes suffer from the over fitting problem (although in previous papers it has provided good accuracy). The probabilistic Naïve Bayes classification assumes statistical independence whereas the K-Nearest Neighbor (k-NN) is observed to provide low efficiency in prior research papers [1]. While our research suggests using the probabilistic Naïve Bayes classification algorithm, which is one of the most preferred for text analysis and the Decision Tree classification which has provided better efficiency in prior research works, we might consider trying with other algorithms as well, if time permits, and then select the one that provides highest prediction accuracy.

### *B. Verifying the veracity of the output*

Although the system classifies the tweets based on their degree of credibility, there might be a need for a mechanism for real time sensitive topics (such as disasters, bomb-blasts, terrorist attacks and other real-world emergencies), to validate whether the output is really correct and to an appreciable extent. To deal with this, web scrapping and web crawling from known, legitimate and uncompromised sources (such as verified and official news channel websites) might be a good source to verify the veracity that has been predicted by the model. Our research in that field is still going on.

## IV. PROBLEM DEFINITION

The spread of misinformation be it intentionally or accidentally, in social media, especially in real-world sensitive incidents, can have harmful effects on not only the individuals but the society as a whole. With a wide variety of crowd from general public to celebrities, politicians and even large organizations, the most popular micro-blogging website, Twitter becomes an easy and favorite target for malicious users to spread misinformation. We identified two main issues regarding the spread of misinformation on online social media. The first issue is that the fake content propagates faster than the real content and much during the very beginning of the event. Hence what comes out first is believed to be true by most of the people and that's precisely where the problem becomes worse as the users start sharing the news over the network without verifying the veracity. The second and an inevitable issue is the absence of any central moderation on the flow of information over the social network due to the use of crowd sourcing. Anyone who is registered on the social media platform has the complete liberty to post any content from his/her account. This makes it difficult not only to control the spread of misinformation over the social network but also to trace the exact source of information.

## V. PROPOSED SYSTEM METHODODLOGY

### *A. Block Diagram*

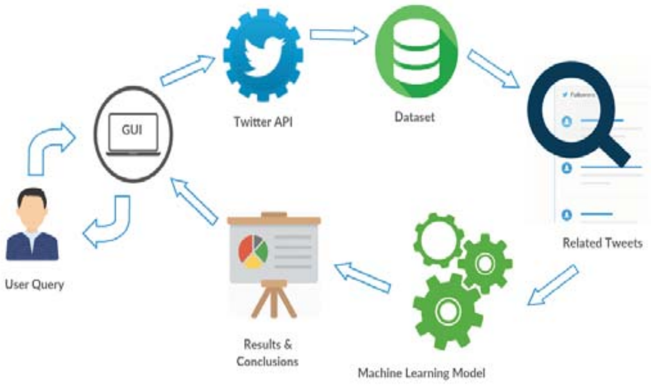


**Fig. 1. Block Diagram of proposed solution for preventing the spread of misinformation on OSM using Machine learning.**

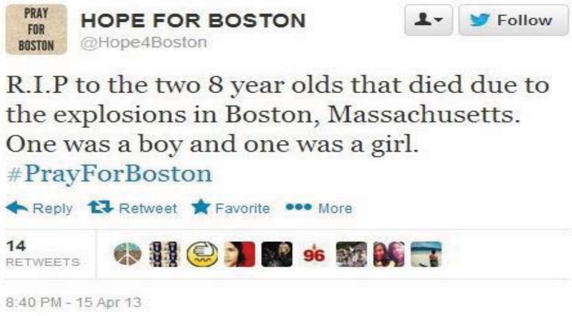
### *B. Proposed Solution*

The system will be designed in a way to provide the user with a classification of the information based on veracity. For that, the user interface of the system will query the user for the hash tags or keywords of the event whose veracity is to be predicted. Based on the user input, the system will use Tweepy to extract real time tweets related to that event. The information received from the API will constitute the dataset for the machine learning model. After collecting the tweets, the information will be fed to the model which will use Python's NLTK module for text analysis followed by the Naïve Bayes and Decision Tree learning algorithms for classification. The algorithm that will provide classification with a better prediction accuracy will be adopted.

Among the available Twitter libraries for Python such as Twython, Tweepy, TwitterSearch, TwitterAPi, we selected Tweepy since it is the official Python API for Twitter that uses REST 1.1 API and supports both OAuth and Streaming API. It is also easy to use and is well maintained. One of the important advantages is that the API can provide as much as 3200 latest tweets per user which is helpful in preparing a good amount of training set for the machine learning model.

The Python Natural Language Tool Kit (NLTK) module is one of the most popular Python tools for Natural Language Processing and handling human language data. Natural Language Processing is a branch of Artificial Intelligence that deals with the interactions between computers and human languages and programming them for analysis of a large amount of natural language data. NLTK supports classification, tokenization, parsing, stemming, tagging, and semantic reasoning functionalities as well. NLTK provides various functionalities such as splitting sentences from paragraphs, splitting words, identifying the part of speech of those words, highlighting important subjects and helping the machine understand what the text is all about. It is widely used for text analysis in the fields on Opinion mining and Sentiment analysis.

Consider a basic example of tokenization of the following fake tweet which was tweeted during the Boston Marathon Blasts.



HOPE FOR BOSTON
@Hope4Boston

R.I.P to the two 8 year olds that died due to the explosions in Boston, Massachusetts. One was a boy and one was a girl. #PrayForBoston

14 RETWEETS

8:40 PM - 15 Apr 13

**Fig. 2. A sample fake tweet as an example for NLTK tokenization.** [7]

The NLTK *sent_tokenize()* function, will perform tokenization of the given tweet to provide a list of tokens as follows:

*[ 'R.I.P to the two 8 year olds that died due to the explosions in Boston, Massachusetts.' , 'One was a boy and one was a girl' , '#PrayForBoston']*

Also, one of the important aspects of Natural Language Processing that is provided in the NLTK module is the concept of Stemming. In simple words, stemming is the process of reducing derived words to their stems (base form) so that they can be analyzed as a single item. This feature is of extreme use while analyzing tweets. For example, a tweet that may contain the words like 'affected', 'affecting', 'affects' are all stemmed to the word 'affect' so that it can be understood that the tweet might be related to a real time event like effect of a verdict, or some disaster or a spread of some disease or any related issue. While searching with stems, it is possible to retrieve many irrelevant terms which have the same roots but are not related to the topic of search. Hence performing some text processing using NLTK, before using the classification algorithms may prevent those algorithms from working on words that indicate the same thing or less relevant to the topic.

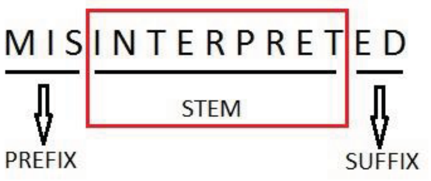


**Fig. 3. Stemming in Natural Language Processing with NLTK.** [8]

The Naïve Bayes classifier is one of the simplest yet surprisingly powerful classification algorithm and one of the most popular method for text categorization. It is a group of simple probabilistic classifiers which use the popular Bayes' theorem on probability. Based on some prior knowledge or conditions that might be related to an event, Bayes' theorem describes the probability of that event. Bayes' theorem can be stated mathematically as:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

**Fig. 4. Mathematical form of Bayes' Theorem.** [9]

where,

- P (A | B) is a conditional probability, that denotes the likelihood of event A occurring, given that B is true.
  This is known as 'Posterior Probability'.
- P (B | A) is a conditional probability, that denotes the likelihood of event B occurring, given that A is true.
  This is known as 'Likelihood'.
- P (A) and P (B) are probabilities of observing A and B respectively, independent of each other.

  These are known as 'Prior Probability' and 'Marginal Likelihood' respectively.

For a given input B, its posterior probability is calculated for different likelihoods and prior probabilities such that the constructed classifiers assign class labels to problem instances. The input is classified in the class for which the posterior probability is the highest. We selected the probabilistic Naïve Bayes for our classification because it is fast, works good for a dataset with a large number of features, is not sensitive to irrelevant data and is usually the preferred algorithm for text classification. Also, the Naïve Bayes classifier can be used to classify feature vectors of any kind into any arbitrary number of categories, which makes it suitable for a multiclass classification. Although the Naïve Bayes classification assumes statistical independence, it has been observed that it still performs well even if the statistical independence assumption is not 100% correct.

The second machine learning algorithm that will be used for classification is the Decision Tree. Decision tree learning can be used for both regression as well as classification by using the concept of Decision Trees. A Decision Tree is a tree in which each branch node indicates a choice between the available alternatives while each leaf node denotes represents a decision. The two important terms in this context are Entropy and Information Gain. Entropy measures the uncertainty of a random variable. Higher the entropy, more the information content. When a node is used in a decision tree to partition the training instances into smaller subsets, the entropy changes. A parameter called Information gain is a measure of this change in entropy. A decision tree classifies inputs by segmenting the input space into regions. Thus, a Decision Tree analyses a dataset in order to construct a set of rules or questions which are used to predict a class. One of the desirable properties of a Decision Tree is that its construction is not much time consuming

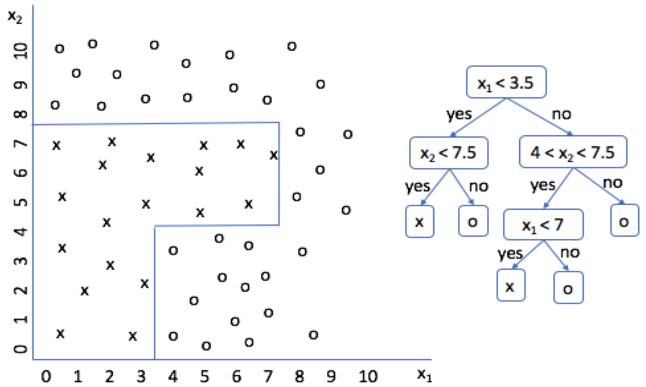


**Fig. 5. Example of Decision tree classification.** [10]

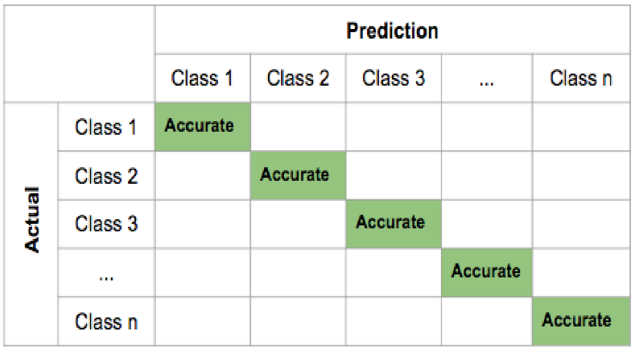
One of the important advantages of Decision Tree classifier is that normalization of data is seldom required and any non-linear relationships between the parameters do not affect the tree performance. The output of a Decision Tree can be easily interpreted and the prediction is fast in most cases.

Thus, the output of our system will be a classification into any of the four classes based on the degree of credibility – fake, seems fake, seems credible or credible.

## VI. PERFORMANCE EVALUATION PARAMETER

To evaluate the performance of the model, a Confusion matrix is used. A confusion matrix is a brief representation of the prediction results on a given classification problem. It derives its name from the fact that it denotes the ways in which a classification model gets confused when it makes predictions. It gives an insight not only into the errors being made by a classifier but more importantly the types of errors that are being made. For example, from a confusion matrix one can not only get the count of errors but can also easily identify that class 2 is identified as class 1. This gives a better insight as to where exactly the improvement as to be made. The fact that it can be used even for multiclass classification problems makes it suitable for our classification model.

**TABLE I: Confusion Matrix for a multiclass classification.** [11]



| | | Prediction | | | | |
|---|---|---|---|---|---|---|
| | | Class 1 | Class 2 | Class 3 | ... | Class n |
| Actual | Class 1 | Accurate | | | | |
| | Class 2 | | Accurate | | | |
| | Class 3 | | | Accurate | | |
| | ... | | | | Accurate | |
| | Class n | | | | | Accurate |

The metrics obtained from the Confusion matrix are:

- Accuracy rate = Correct predictions / Total
- Error rate = Incorrect predictions / Total

## VII. EXPERIMENTAL SET UP

Software requirements for the proposed system:

- Install the Anaconda distribution which provides latest Python 3.x version along with a complete collection of all scientific computation libraries and modules required for machine learning.
- Additional Python packages not included under Anaconda like PyQt (for Python GUI), Tweepy (Twitter API for extracting tweets) and NLTK (for text analysis), can be installed manually or using pip command.
- Active internet connection for real time extraction of tweets.

Dataset: The system does not use a static dataset but will instead create it dynamically by extracting tweets using Tweepy for a given set of keywords provided by the user.

Train test split: The ideal 75 – 25 train-test split (75% for training, 25% for testing) which is the default in scikit-learn will be used.

## VIII. CONCLUSION AND FUTURE SCOPE

The impact of social media on individuals and the society as a whole cannot be neglected. Although social media, and in particular Twitter, has been successful in connecting millions of individuals across the world, many a times the social network is used to spread fake news and circulate malicious content which may even turn out dangerous, particularly in real world emergencies. The proposed system uses powerful machine learning algorithms to curb this issue in real time and thus contributes to the development of a better society. The same concept can be further extended to other popular OSMs like Facebook and instant messaging applications like WhatsApp. Further, a system can be developed that verifies the veracity by taking the inputs from more than one OSMs and then decide the credibility of the information. The prediction of such as system would be highly accurate and would prevent the spread of misinformation more rapidly.

## REFERENCES

[1] Raveena Dayani, Nikita Chhabra, Taruna Kadian and Rishabh Kaushal.

[2] (2015). “Rumor Detection in Twitter: An Analysis in Retrospect”. IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS).

[3] Sardar Hamidian and Mona Diab. (2015). “Rumor Detection and Classification for Twitter Data”. SOTICS 2015: The Fifth International Conference on Social Media Technologies, Communication, and Informatics.

[4] Qiao Zhang, Shuiyuan Zhang, Jian Dong, Jinhua Xiong, Xueqi Cheng. (2015). “Automatic Detection of Rumor on Social Network”. NLPCC 2015 Proceedings of the 4th CCF Conference on Natural Language Processing and Chinese Computing - Volume 9362.

[5] Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, Anupam Joshi. (2014) “Analyzing and Measuring the Spread of Fake Content on Twitter during High Impact Events”. Security and Privacy Symposium 2014, CSE - IIT-Kanpur.

[6] Vahed Qazvinian, Emily Rosengren, Dragomir R. Radev, Qiaozhu Mei.

[7] Nidhi Chandra, Sunil Kumar Khatri, Subhranil Som, (2017) “Anti-Social Comment Classification based on kNN Algorithm”, 6th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), IEEE Conference, indexed with SCOPUS

Sep. 20-22, 2017, Amity University Uttar Pradesh, Noida, India.

[8] (2011). "Rumor has it: Identifying Misinformation in Micro blogs".

[9] Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing.

[10] Carlos Castillo, Marcelo Mendoza, Barbara Poblete. (2011). "Information Credibility on Twitter". WWW 2011 – Session: Information Credibility

[11] What Really did Happen During Boston Marathon Bombing? Blog post at unsusocmed.wordpress.com. Retrieved from https://i2.wp.com/i2.cdn.turner.com/cnn/dam/assets/130416112938-twitter-boston-misinformation-story-top.jpg

[12] Introduction to Natural Language Processing with NLTK.

[13] Open data Science Blog, Stemming. Retrieved from https://old.opendatascience.com/wp-content/uploads/2017/04/stem.jpg

[14] Bayes' theorem. Wikipedia, The free Encyclopedia. Retrieved from https://wikimedia.org/api/rest_v1/media/math/render/svg/b1078eae6dea894bd826f0b598ff41130ee09c19

[15] Decision Trees. Jeremy Jordan | Data Science. Retrieved from https://www.jeremyjordan.me/content/images/2017/03/Screen-Shot-2017-03-11-at-10.15.37-PM.png

[16] Confusion matrix for multiclass classification (n number of classification). WSO2 Machine Learner documentation. Retrieved from https://docs.wso2.com/display/ML110/Model+Evaluation+Measures

[17] Shobha Tyagi, Subhranil Som, Qamar Parvez Rana (2017) "Trust based Dynamic Multicast Group Routing ensuring Reliability for Ubiquitous Environment in MANETs", International Journal of Ambient Computing and Intelligence (IJACI), Volume 8, Issue 1, ESCI, Scopus Indexed, ISSN: 1941-6237, DOI: 10.4018/IJACI, Pages 70 – 97, January – March 2017. (http://www.igi-global.com/journals/abstract-announcement/158348)