2018 IEEE 16th Int. Conf. on Dependable, Autonomic & Secure Comp., 16th Int. Conf. on Pervasive Intelligence & Comp., 4th Int. Conf. on Big Data Intelligence & Comp., and 3rd Cyber Sci. & Tech. Cong.

Fake News Detection Enhancement with Data Imputation

Chandra Mouli Madhav Kotteti, Xishuang Dong
CREDIT Center
Prairie View A&M University
Texas A&M University System
Prairie View, TX 77446, USA
ckotteti@student.pvamu.edu, dongxishuang@gmail.com

Na Li
Department of Computer Science
Prairie View A&M University
Texas A&M University System
Prairie View, TX 77446, USA
nali@pvamu.edu

Lijun Qian
CREDIT Center
Prairie View A&M University
Texas A&M University System
Prairie View, TX 77446, USA
liqian@pvamu.edu

***Abstract*—Raw datasets collected for fake news detection usually contain some noise such as missing values. In order to improve the performance of machine learning based fake news detection, a novel data preprocessing method is proposed in this paper to process the missing values. Specifically, we have successfully handled the missing values problem by using data imputation for both categorical and numerical features. For categorical features, we imputed missing values with the most frequent value in the columns. For numerical features, the mean value of the column is used to impute numerical missing values. In addition, TF-IDF vectorization is applied in feature extraction to filter out irrelevant features. Experimental results show that Multi-Layer Perceptron (MLP) classifier with the proposed data preprocessing method outperforms baselines and improves the prediction accuracy by more than 15%.**

***Index Terms*—fake news, machine learning, data imputation, feature extraction**

I. Introduction

Social media has become the fast and easy way to proliferate news across the world and they make news readily available for the news consumers. However, fake news on social media has been proliferated for personal or social benefits. Fake news is typically a piece of false information in nature, where its primary purpose is to deceive or mislead readers. It has many similarities with spam messages since they share common features such as grammatical mistakes, false information, using similar limited set of words, and they contain emotionally colored information that affects the reader's opinion [1].

Detecting fake news is a layered process that involves analysis of the news contents to determine the truthfulness of the news. The news could contain information in various formats such as text, video, image, etc. Combinations of different types of data make the detection process difficult. In addition, raw data collected is always expected to be unstructured and contains missing values in the data. As fake news produces the big, incomplete, unstructured, and noisy data [2], raw data pre-processing is extremely important to clean and structure the data before feeding it into detection models.

Fake news detection is a nontrivial task as fake news is mainly intended to mislead readers. To enhance the existing fake news detection algorithms, only news content is not adequate. This opens the gates for the necessity of auxiliary information, such as user social engagements on social media for the better detection of fake news [2]. The availability of good quality datasets for fake news detection is also a big challenge. Wang [3] presents that inadequate availability of manually labeled datasets on fake news problem causes limited opportunity to use currently available advanced computational power and enhanced machine learning models. Another challenge is manually labeling a dataset is a time-consuming task [4]. Fortunately, LIAR dataset[1] is one of the available labeled datasets for fake news detection. It provides not only the actual news content but also auxiliary information, such as subject type, context of the news and speaker's details. However, this dataset contains combination of both categorical (text) and continuous (numbers) values and the muddled presence of missing values. These missing values reduce the detection performance significantly if left untreated.

In this paper, we pre-processed the data by employing imputing strategies for the missing values in the dataset, where sklearn-pandas[2] categorical imputing and sklearn's Imputer[3] with mean imputing strategies are employed for categorical data and continuous data, respectively. Categorical and numerical features are handled together using sklearn-pandas[2] DataFrameMapper method. After the data pre-processing and feature extraction phases are completed, we supplied the cleaned dataset into classifiers such as Support Vector Machines, Decision Tree, Multi-layer Perceptron and Gradient Boosting for our analysis and comparison. Our experimental results show that the state-of-the-art prediction accuracies are improved by 16%.

The contributions of this paper include:

- The raw dataset has many missing values spread across multiple columns. We successfully process the missing categorical and continuous values by categorical imputer and mean imputer.
- We combine traditional machine learning models that are capable of handling multi-class classification tasks with

[1]https://www.cs.ucsb.edu/~william/data/liar_dataset.zip
[2]https://github.com/scikit-learn-contrib/sklearn-pandas
[3]http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Imputer.html

our preprocessing methods, and show that multi-layer perceptron model significantly outperforms the state-of-the-art [3].

The outline of the paper is as follows: Section II introduces the LIAR dataset. Our proposed method is discussed in Section III. Experimental results and analysis are shown in Section IV. Related work is discussed in Section V followed by Section VI that concludes our work.

II. LIAR Dataset

LIAR dataset[1] is a benchmark dataset for fake news detection collected from PolitiFact[4]. It includes both categorical and numerical features combined for a total of 14 columns. Columns containing categorical (text) data include statement identifier, statement, subjects discussed by the speaker, and meta-data for each speaker, such as speaker's job title, state, party, and the location of the speech. The numerical features contain the speaker's total credit history count, including the current statement which are named as, barely true, false, half true, mostly true, and pants on fire counts [3]. The target labels consist of six classes including *pants-fire, false, barely-true, half-true, mostly-true, and true*. This dataset is human labeled and each statement is evaluated by a PolitiFact editor for its truthfulness. Overall dataset contains 12,836 records in which training set has 10,269 records, validation and testing sets have 1,284 and 1,283 records, respectively. The training, validation, and test sets are supplied in separate files.

III. Proposed Methodology

A. Data preprocessing

1) Mitigate Missing Values: LIAR dataset[1] consists of a combination of categorical and numerical features. This dataset has many randomly located missing values for both type of features. It is not possible to check that the observed data contains missing values of Missing Completely at Random (MCAR) or Missing Not at Random (MNAR) [5]. Therefore, missing data imputation would be a good solution to handle these missing values. Typical imputation methods such as "mean" or "mode" rely on explicit model assumptions. In general, mean is preferred for quantitative data and mode is for qualitative data [5].

In this study, we use scikit-learn's Imputer with "mean" strategy for handling missing values in the numerical columns which replaces the missing values with the mean along the axis (0 - along columns, 1 - along rows) [6]. CategoricalImputer is a new method available in sklearn-pandas module for handling categorical missing values. It is applied to data columns that are of type "string" and it substitutes null values with the most frequent value in the column. Researchers who use scikit-learn module cannot impute missing categorical values since scikit-learn module imputing methods are limited to numerical data. Therefore, the CategoricalImputer method is helpful in imputing missing categorical values whereas imputing methods in the scikit-learn module could be applied to numerical data.

[4]http://www.politifact.com/

2) Feature Extraction: Wu et. al [7] stated that extracting useful features from the actual news content is a challenging task because fake news spreaders could make the content of the fake news look like real news. In our work, we used term frequency and inverse document frequency (TF-IDF) to identify the useful features from news contents. TF-IDF technique is used to produce a composite weight for each term in the document which is called tf-idf weight [8]. Calculating *tf-idf* weight has great importance in information retrieval and text mining tasks as it determines the significance of a term or word in a document as well as in a corpus.

$$tf - idf_{t,d} = tf_{t,d} \times idf_t \tag{1}$$

In equation 1, t means a term and d refers to a document. The term frequency $tf_{t,d}$ means the measure of the frequency for a particular term t in a document, in other words, how many times term t appeared divided by total number of terms in the document and inverse document frequency idf_t is the logarithm of total number of documents in the corpus divided by the number of documents where term t appears. idf_t measure helps in knowing the importance of term t.

B. Model

We treat fake news detection as a multi-class classification problem. Traditional machine learning classifiers such as Support Vector Machines (SVM), Decision Trees, Multi-layer Perceptron and Gradient BoostingGradient Boosting are selected. For SVM models, we use classical SVC, Linear SVC with "crammer_singer", "one-vs-rest" multi-class strategies, and Nu-SVC as classifiers.

1) Support Vector Machines: Support vector machine has great importance in solving classification problems consisting of nonlinearly separable classes. In our work, we used Support Vector Classification (SVC), Nu-Support Vector Classification (NuSVC) and Linear Support Vector Classification (LinearSVC) to handle multi-class classification tasks. The one-vs-one scheme is implemented by SVC and NuSVC for multi-class classification, where the classifiers are constructed based on the number of classes presented in the dataset. NuSVC is similar to SVC, but NuSVC controls the number of support vectors and training errors using a parameter ν. LinearSVC is also similar to SVC but the kernel used for classification is "linear". It can implement "one-vs-rest" and "crammer_singer" multi-class strategies in which the former strategy is generally preferred as the latter strategy is more expensive to compute and better performance is rarely achieved.

2) Decision Trees: Decision Trees is a supervised classification and regression model that relies on the decision rules derived from the data features. It could be applied to binary classification problems as well as multi-class problems. It is capable of handling both categorical and numerical data and requires little data preparation. On the other hand, sometimes this model could create over-complex trees (i.e. over-fitting). Data alteration may change the complexity of the decision tree.

3) Multi-layer Perceptron: Multi-layer Perceptron is a supervised learning algorithm that learns a function $f(\cdot): R^m \rightarrow R^o$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output. It consists of one or more non-linear layers, called hidden layers between input and output layers. Input features are a set of neurons $\{x_i | x_1, x_2, \ldots, x_m\}$, and in the hidden layer each neuron transforms previous layers values by using a weighted linear summation $w_1x_1 + w_2x_2 + \ldots + w_mx_m$ and non-linear activation function $g(\cdot): R \rightarrow R$. Values from the last hidden layer are transformed into output values by the output layer. It is useful for on-line learning and to learn non-linear models.

4) Gradient Boosting: Gradient Tree Boosting is one of the ensemble-based methods. Gradient Boosting builds a forward stage-wise additive model. It could be used for both classification and regression problems. In this model heterogeneous features are naturally handled, but scalability is an issue because of the sequential nature of boosting.

IV. Experiment

We employ LIAR dataset[1] to verify our models. Four evaluation metrics, namely accuracy, precision, F1-score and recall are used to evaluate the performance of our models. One of the major challenges of performing classification on this dataset is to handle missing values. To mitigate this problem, we applied three data preprocessing methods on the dataset, and examine how effectively each method could impact the performances of the classifiers. For all three methods we performed feature extraction on the dataset as discussed in Section III-A2 and we utilized all features except *statement id* for our analysis. Additionally, we examine the computational complexity of the models by monitoring the training and prediction time for different classifiers, and they are presented in hours, minutes, seconds and milliseconds (HH:MM:SS:ms) format. The three methods we used are as follows:

A. Delete records containing missing values

In this method, we simply deleted records consisting of missing values. This method removed more than 4,000 records from the dataset. MLP classifier outperformed other classifiers in predicting validation and test sets. The performances are shown in TABLES II and III.

B. Replace missing values with empty text

We used *empty* text to replace the missing values in the dataset. With this method we successfully prevented the data loss problem because no records are deleted. Again, MLP classifier stood top in the list in terms of performance. This time the prediction accuracies for validation and test sets are improved compared to that using delete method. TABLES V and VI show the respective performance results for validation and test sets with replace method.

TABLE I
Training time of different classifiers with delete method.

Classifier	Training Time (HH:MM:SS:ms)
SVC	0:21:33.292383
LinearSVC_CS	0:03:11.075396
LinearSVC_OVR	0:00:09.173900
NuSVC	0:13:13.883099
DecisionTree	0:00:06.170634
MLPClassifier	1:40:09.743315
GradientBoosting	0:21:42.929440

TABLE II
Performance results on validation set with delete method.

Classifier	Prediction Time (HH:MM:SS:ms)	Accuracy %	F1-Score	Precision	Recall
SVC	0:01:56.706200	0.283	0.182	0.217	0.234
LinearSVC_CS	0:00:00.634472	0.174	0.173	0.179	0.181
LinearSVC_OVR	0:00:00.021058	0.265	0.228	0.258	0.237
NuSVC	0:01:29.970123	0.258	0.240	0.255	0.244
DecisionTree	0:00:00.047126	0.326	0.324	0.328	0.322
MLPClassifier	0:00:00.105282	0.416	0.370	0.515	0.370
GradientBoosting	0:00:00.071388	0.400	0.377	0.441	0.368

TABLE III
Performance results on test set with delete method.

Classifier	Prediction Time (HH:MM:SS:ms)	Accuracy %	F1-Score	Precision	Recall
SVC	0:01:58.420455	0.286	0.174	0.179	0.230
LinearSVC_CS	0:00:00.040109	0.173	0.166	0.172	0.174
LinearSVC_OVR	0:00:00.025069	0.225	0.210	0.224	0.217
NuSVC	0:01:26.568999	0.247	0.229	0.239	0.238
DecisionTree	0:00:00.044117	0.339	0.343	0.338	0.351
MLPClassifier	0:00:00.079210	0.394	0.359	0.515	0.356
GradientBoosting	0:00:00.086864	0.390	0.391	0.438	0.381

C. Impute missing values using data imputation techniques

In this method, we evaluated our data preprocessing method as discussed in Section III-A1 using different machine learning classifiers on validation and test datasets after these models are trained successfully. TABLES VIII and IX show the performance results on the validation set and test set, respectively. It is observed that the classifiers with data imputation outperform those with delete method in Section IV-A. Moreover, replace and data imputation methods achieved almost similar performance results. With delete method we obtain examples by eliminating records with any missing values, which reduces the actual dataset size and causes information loss. This method is suggested only for large datasets with small percentage

TABLE IV
TRAINING TIME OF DIFFERENT CLASSIFIERS WITH REPLACE METHOD.

Classifier	Training Time
	(HH:MM:SS:ms)
SVC	1:38:25.423104
LinearSVC_CS	0:06:32.412224
LinearSVC_OVR	0:00:15.322742
NuSVC	1:05:17.864797
DecisionTree	0:00:24.834177
MLPClassifier	0:42:30.996866
GradientBoosting	1:12:06.434310

TABLE V
PERFORMANCE RESULTS ON VALIDATION SET WITH REPLACE METHOD.

Classifier	Prediction Time	Accuracy	F1-Score	Precision	Recall
	(HH:MM:SS:ms)	%			
SVC	0:06:49.043784	0.243	0.198	0.291	0.230
LinearSVC_CS	0:00:00.083121	0.191	0.183	0.183	0.183
LinearSVC_OVR	0:00:00.066884	0.280	0.273	0.294	0.277
NuSVC	0:06:25.410218	0.354	0.347	0.363	0.342
DecisionTree	0:00:00.110789	0.393	0.391	0.394	0.389
MLPClassifier	0:00:00.474289	0.458	0.454	0.553	0.443
GradientBoosting	0:00:00.157420	0.446	0.441	0.486	0.432

TABLE VI
PERFORMANCE RESULTS ON TEST SET WITH REPLACE METHOD.

Classifier	Prediction Time	Accuracy	F1-Score	Precision	Recall
	(HH:MM:SS:ms)	%			
SVC	0:07:35.188419	0.248	0.188	0.254	0.224
LinearSVC_CS	0:00:00.084752	0.184	0.174	0.176	0.175
LinearSVC_OVR	0:00:00.061115	0.256	0.242	0.258	0.249
NuSVC	0:06:59.119033	0.353	0.341	0.351	0.337
DecisionTree	0:00:00.116335	0.370	0.381	0.379	0.384
MLPClassifier	0:00:00.502584	0.434	0.434	0.533	0.434
GradientBoosting	0:00:00.205547	0.426	0.432	0.465	0.426

of missing values occurrence, and analysis of the complete examples should not make dataset seriously biased [9]. On the other hand, with replace and data imputation methods we eliminate the problem of data loss. For the replace method, we consider missing values as blank values and treat them in the same way as other values. Data imputation methods are simple and effective solutions when the missing values caused by missing at random (MAR) mechanism which is the case here [9].

Compared to the state-of-the-art [3], our proposed method for data preprocessing together with MLP Classifier has significantly improved the accuracies on validation set and test set by 21% and 16%, respectively. Training iterations are limited to 200 with a fixed random state value, and we employed *stochastic gradient descent* to optimize MLP classifier. Gradient Boosting, Decision Tree and NuSVC classifiers also achieved satisfactory performances where Decision Tree Classifier consumed less time for training. It is also observed that classifiers including SVC, LinearSVC with "crammer_singer" and "one-vs-rest" strategies performed poorly and achieved less accuracy scores since the dimensionality of the feature is high. Additionally, we measured the total time consumed for the prediction on both validation and test sets, as well as some other metrics, such as F1$-$Score, Precision, Recall.

TABLE VII
TRAINING TIME OF DIFFERENT CLASSIFIERS WITH DATA IMPUTATION METHOD.

Classifier	Training Time
	(HH:MM:SS:ms)
SVC	1:37:29.514189
LinearSVC_CS	0:07:52.727027
LinearSVC_OVR	0:00:20.665244
NuSVC	1:32:44.329309
DecisionTree	0:00:12.401620
MLPClassifier	0:48:19.175377
GradientBoosting	1:02:10.105386

TABLE VIII
PERFORMANCE RESULTS ON VALIDATION SET WITH DATA IMPUTATION METHOD.

Classifier	Prediction Time	Accuracy	F1-Score	Precision	Recall
	(HH:MM:SS:ms)	%			
SVC	0:08:10.334100	0.245	0.200	0.293	0.232
LinearSVC_CS	0:00:00.086328	0.195	0.190	0.189	0.191
LinearSVC_OVR	0:00:00.150023	0.267	0.264	0.274	0.272
NuSVC	0:07:54.447672	0.367	0.359	0.394	0.349
DecisionTree	0:00:00.075579	0.394	0.395	0.400	0.393
MLPClassifier	0:00:00.491813	0.457	0.455	0.504	0.444
GradientBoosting	0:00:00.107796	0.442	0.437	0.484	0.428

We run MLP Classifier with our proposed methods for ten rounds to observe its performance without using random state value. The number of iterations are limited to 300 for the MLP classifier. TABLE X lists the results for the training set. It shows that the MLP classifier combined with proposed data preprocessing method is stable by maintaining training loss consistency.

Figure 1 gives the training loss curves versus the number of iterations. TABLES XI and XII show the details of the MLP Classifier performance for ten rounds on validation and test sets. It is observed that the training loss curves for all the ten rounds are consistent with average final loss value of 1.279.

TABLE IX
Performance results on test set with data imputation method.

Classifier	Prediction Time (HH:MM:SS:ms)	Accuracy %	F1-Score	Precision	Recall
SVC	0:08:15.804666	0.248	0.188	0.254	0.224
LinearSVC_CS	0:00:00.088085	0.178	0.170	0.171	0.171
LinearSVC_OVR	0:00:00.209324	0.239	0.231	0.238	0.244
NuSVC	0:07:37.068152	0.360	0.342	0.366	0.338
DecisionTree	0:00:00.078278	0.381	0.391	0.386	0.397
MLPClassifier	0:00:00.462904	0.436	0.440	0.492	0.435
GradientBoosting	0:00:00.104769	0.426	0.432	0.463	0.426

TABLE X
Performance of MLP Classifier.

Round	Training Time (HH:MM:SS:ms)	Training Loss	No. of Iterations
1	0:39:48.745564	1.303	127
2	0:49:54.880182	1.280	154
3	0:47:42.052116	1.286	148
4	0:55:40.648856	1.270	171
5	1:04:42.601990	1.265	177
6	1:16:18.708549	1.254	197
7	0:37:16.882210	1.322	116
8	0:48:07.396703	1.265	175
9	0:48:46.390700	1.266	177
10	0:41:38.946055	1.282	152

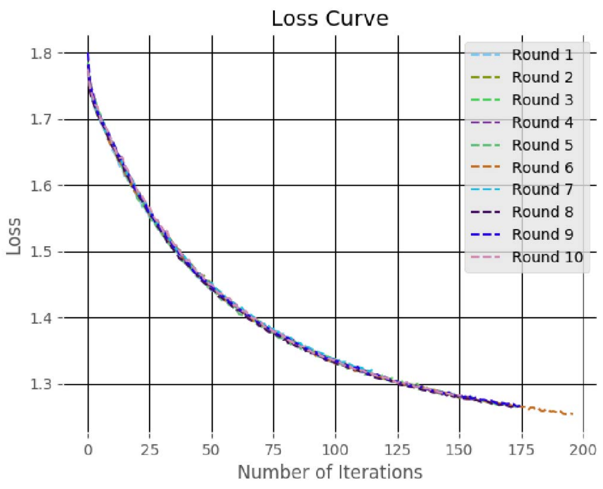

Fig. 1. Training loss curves.

V. Related work

Fake news detection attracts amount of attentions because of its application values. And they employ ideas for detecting rumor [10] from texts for implementing fake news detection based on similarities between fake news and rumor. Machine learning, especially deep learning, plays a key technique for fake news detection. Ma et.al [11] automatically detect deep data representations for the enhancement of the rumor detection. Their experiments focus on using variations in the contextual information of relevant posts over time for rumor detection instead of using manually extracted features. Not only relying on the contents of the news, the network structure of the news could also be helpful for identification of fake news [12]. Early detection is also a proactive method to deal with fake news detection problem. In [13], they present early detection of rumors in social media based on identifying signature text phrases in social media posts, for example, “Is this true?, Really?”. In [14], they propose an automatic mechanism for fake news classification using four important processes i.e. extracting features for prediction accuracy, dataset alignment, per-set feature selection and evaluating model transfer.

Fake news detection could be addressed based on propagation patterns of fake news as well. Ma et. al [15] use the propagation structure technique for rumor detection problem.

TABLE XI
MLP Classifier performance results on validation set.

Prediction Time (HH:MM:SS:ms)	Accuracy %	F1-Score	Precision	Recall
0:00:00.480897	0.465	0.454	0.574	0.446
0:00:00.357233	0.470	0.462	0.577	0.454
0:00:00.385952	0.453	0.446	0.571	0.438
0:00:00.481820	0.451	0.445	0.570	0.437
0:00:00.469991	0.469	0.458	0.577	0.452
0:00:00.395029	0.467	0.458	0.557	0.449
0:00:00.334350	0.450	0.449	0.486	0.439
0:00:00.434091	0.461	0.453	0.576	0.447
0:00:00.345711	0.459	0.453	0.555	0.443
0:00:00.355381	0.462	0.457	0.560	0.448

TABLE XII
MLP Classifier performance results on test set.

Prediction Time (HH:MM:SS:ms)	Accuracy %	F1-Score	Precision	Recall
0:00:00.453560	0.443	0.439	0.532	0.439
0:00:00.437473	0.430	0.429	0.533	0.428
0:00:00.355367	0.449	0.444	0.551	0.445
0:00:00.470456	0.443	0.441	0.548	0.441
0:00:00.622739	0.449	0.443	0.550	0.443
0:00:00.294496	0.445	0.441	0.538	0.443
0:00:00.443745	0.448	0.455	0.483	0.447
0:00:00.251151	0.436	0.433	0.540	0.433
0:00:00.331610	0.444	0.440	0.539	0.441
0:00:00.283757	0.443	0.438	0.539	0.441

They used propagation trees to identify clues on how an original message is spread over time. Kwon et. al [16] employ temporal, structural and linguistic characteristics of rumor propagation, and propose a new periodic time series model to identify temporal features. They also identified key structural and linguistic features in the rumor propagation and achieved better performance results over existing state of the arts on rumor classification.

Extracting and selecting useful features could enhance the performance of machine learning based fake news detection. In [17], they explored the importance of content-based features, network-based features and microblog-specific memes for the identification of rumors. Content-based features are extracted from text data whereas network-based features focus on user's behavior. Moreover, features such as hangtags and URLs extracted from microblog-specific (Twitter specific) memes could be helpful in enhancement of rumor detection models.

VI. Conclusion

In this paper, data imputation preprocessing method is proposed for enhancing machine learning based fake news detection. Our proposed method focuses on how to process the missing values in the raw data using data imputation techniques. Experimental results show that machine learning models combined with the proposed data preprocessing method outperform baselines. It would be interesting to test our proposed method on other data sets for fake news detection and it will be one of our future works.

Acknowledgment

This research work is supported in part by the U.S. National Science Foundation (NSF) under award 1712496 and the U.S. Office of the Under Secretary of Defense for Research and Engineering (OUSD(R&E)) under agreement number FA8750-15-2-0119. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. NSF or the Office of the Under Secretary of Defense for Research and Engineering (OUSD(R&E)) or the U.S. Government.

References

[1] M. Granik and V. Mesyura, "Fake news detection using naive bayes classifier," in *2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)*, May 2017, pp. 900–903.

[2] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *CoRR*, vol. abs/1708.01967, 2017. [Online]. Available: http://arxiv.org/abs/1708.01967

[3] W. Y. Wang, ""liar, liar pants on fire": A new benchmark dataset for fake news detection," *CoRR*, vol. abs/1705.00648, 2017. [Online]. Available: http://arxiv.org/abs/1705.00648

[4] A. Vlachos and S. Riedel, "Fact checking: Task definition and dataset construction," in *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, 2014, pp. 18–22.

[5] J. Poulos and R. Valle, "Missing Data Imputation for Supervised Learning," *ArXiv e-prints*, Oct. 2016.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[7] L. Wu and H. Liu, "Tracing fake-news footprints: Characterizing social media messages by how they propagate," 2018.

[8] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[9] J. Kaiser, "Dealing with missing values in data," vol. 5, pp. 42–51, 01 2014.

[10] A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata, and R. Procter, "Detection and resolution of rumours in social media: A survey," *CoRR*, vol. abs/1704.00656, 2017. [Online]. Available: http://arxiv.org/abs/1704.00656

[11] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha, "Detecting rumors from microblogs with recurrent neural networks." in *IJCAI*, 2016, pp. 3818–3824.

[12] L. Wu, F. Morstatter, X. Hu, and H. Liu, "Mining misinformation in social media," *Big Data in Complex and Social Networks*, pp. 123–152, 2016.

[13] Z. Zhao, P. Resnick, and Q. Mei, "Enquiring minds: Early detection of rumors in social media from enquiry posts," in *Proceedings of the 24th International Conference on World Wide Web*, ser. WWW '15. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2015, pp. 1395–1405. [Online]. Available: https://doi.org/10.1145/2736277.2741637

[14] C. Buntain and J. Golbeck, "Automatically identifying fake news in popular twitter threads," in *2017 IEEE International Conference on Smart Cloud (SmartCloud)*, Nov 2017, pp. 208–215.

[15] J. Ma, W. Gao, and K.-F. Wong, "Detect rumors in microblog posts using propagation structure via kernel learning," in *ACL*, 2017.

[16] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang, "Prominent features of rumor propagation in online social media," in *2013 IEEE 13th International Conference on Data Mining*, Dec 2013, pp. 1103–1108.

[17] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei, "Rumor has it: Identifying misinformation in microblogs," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 1589–1599. [Online]. Available: http://dl.acm.org/citation.cfm?id=2145432.2145602