Classification of the Original Author-Party of a Political Document Using Logistical Regression, Stochastic Gradient Descent and Multinomial Naïve Bayes Classifiers

Anton Mo Eriksson, *TDDE16, Text Mining,*
September 12, 2019, anter491@student.liu.se,
Linköping University – Linköping – Sweden

Abstract—This paper outlines the procedure to mine web API:s, data cleaning, and general text mining pre-processing, to achieve a $\mathcal{N}$-gram and TF-IDF model of the political documents collected. All this to be able to classify the original author-party of a political document using Logistical Regression, Stochastic Gradient Descent and Multinomial Naïve Bayes classifiers. The best classifier performed with an accuracy of 88.4%, reasons for that are found in the discussion.

1 Introduction

Modern software engineering has given humanity the ability to work with big data and massive computational power offered by parallel computing in the cloud [1]. The massive expansion of the computational power introduces new domains and opportunities start to show themselves, machine learning and text mining are old ideas, which in recent years has been made plausible by the newly acquired computational powers [2].

Moreover, how should we use these newly acquired computational resources? Some would argue investment banking or smart city planning [3] [4]; however, an unexplored area is the political domain which will be the focus of this project. Then one may ask, what can we benefit from applying these computational resources to the area of politics. The answer to that question is quite simple, to use the full power of text mining to uncover what politicians and political parties *actually* stand for. All this to improve the democracy we live and thrive in, to transform our democracy from *democracy-1.0* to *democracy-2.0*.

In this project, the goal was to determine which Swedish political party was the original authors of a certain document. The concept of which party authored the document can reveal, interesting information regarding the possible political allies. This might be confusing to the politically engaged, which would claim only two solid political blocks exist. However, shown by the recent shake-up of the Swedish government, there is a need for new thinking and innovation in this area. Furthermore, it also presents the opportunity to tie certain politicians within a party to certain policies, which can be of great interest to both the public as well as the party.

Finally, the task to classify the original author-party of documents collected from the open API[1] to investigate the unions and coupling within the Swedish parliament, boils down in the research question:

- *Can predictions be made of the original political faction based on the motion text? With a Multinomial Naïve Bayes classifier trained on a unigram model as baseline and compare its classification with; Multinomial Naïve Bayes trained on a TF-IDF model, Logistical Regression and Stochastic Gradient Decent*

1. https://data.riksdagen.se/

1.1 Limitations

The project will only focus on the party level, thus not the individual politician perspective, which is too broad of a scope for this project. But encourage the reader to pursue that area of research.

2 Theory

This section aims to give the reader some theoretical background to the technologies and methodologies used.

2.1 $\mathcal{N}$–gram model

The $\mathcal{N}$–gram model represents the sequence of $\mathcal{N}$ continues words from a text or a speech. The model aims to capture the conditional probability given by the $\mathcal{N}-1$ previous words, thus making the meaning of verbs more important to the model [5]. The $\mathcal{N}$-gram model can be implemented with any number, where $\mathcal{N}=1$ is known as unigram or Bag of Words, followed by bigrams for $\mathcal{N}=2$, where the probability model is described as the following:

$$P(\boldsymbol{w}_1^{\mathcal{N}}) = \prod_{k=1}^{\mathcal{N}} P(w_k|\boldsymbol{w}_{k-\mathcal{N}+1}^{k-1}). \tag{1}$$

where: $\boldsymbol{w}_1^{\mathcal{N}}$: the vector of words $(w_1,\ldots,w_{\mathcal{N}})$,
$\mathcal{N}$: the order of the model,
w_k : the word at index k.

2.1.1 Bag-of-Words

A special case of the $\mathcal{N}$–gram model is the unigram, also known by the term, Bag of Words. The idea behind Bag of Words can be derived from the name, a dictionary structure of the occurrences of words, where the key is the word in

question, and the value is the frequency. This technique is good at capturing a representation of a string input, where the string could be a comprehensive text, even if the use of a Bag of Words model can result in high dimensionality [6].

The procedure for Bag of Words can be described in the following steps; convert all alphabetical characters to lowercase and remove none alphabetic characters. Tokenize the text and then iterate through the text with the tokens as key and increment the frequency value.

2.2 Term Frequency-Inverse Document Frequency

The term frequency-inverse document frequency or (TF-IDF) is one of the most popular types of model in the text mining field [7]. It models the importance of a word in a given document collection. TF-IDF is defined by two sub-formulas, term frequency and inverse document frequency , both presented below:

$$tf(t, D) = \frac{1}{2} + \frac{f_{t,d}}{2 \cdot \max\{f_{t',d} : t' \in d\})} \tag{2}$$

where: $tf_{t,d}$: the term frequency,
$f_{t,d}$: the raw frequency,
t : the count of the word,
t' : the most frequent word,
d : the document.

Followed by the second one,

$$idf(t, D) = log(\frac{N}{1 + |d \in D : t \in d|}) \tag{3}$$

where: idf : the inverse document frequency
t : the count of the word,
N : the amount of documents in D,
d : the document,
D : the documents.

Those two formulas together defines the function for TF-IDF presented below:

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D) \tag{4}$$

The use of TD-IDF is an attempt to make words occurring frequently in few documents, not be over represented by the model [7].

2.3 Naïve Bayes

One of the most classic and old families of algorithms in the text mining field, but despite its age, it still packs a punch [8]. Thus, comparing equally or better than more advanced machine learning algorithms (depending on the area) [6]. Naïve Bayes in a probability classifier, using the famous Bayes theorem, with the prerequisite of independence between the features in the dataset. The term naive is derived from the assumption of the independence between the features.

All the algorithms in the Naïve Bayes family label the input feature vector according to a set of pre-defined labels. The procedure will be illustrated in the formula below:

$$\overline{\mathcal{L}} = arg \max_{c \in \mathcal{C}} P(c) \cdot \prod_{i=1}^{n} P(x_i|c) \tag{5}$$

where: $\overline{\mathcal{L}}$: the predicted labels
$\mathcal{C}$: the set of labels,
c : the concrete label,
n : the size of feature-vector,
x_i : i:th component in the feature vector.

There are several different variations of the Naïve Bayes classification [6], and thus in this project, the following has been chosen.

2.3.1 *Multinomial Naïve Bayes*

A variation on the Naïve Bayes is the Multinomial Naïve Bayes classifier, which differs on the assumption that the probability distribution is a multinomial [6], defined by the formula below:

$$P(w_1, ..., w_n|c) \sim \mathcal{M}(\theta_{si}, n_i). \tag{6}$$

Hence, assuming that the distribution of the probability is multinomial. The formula for the Multinomial Naïve Bayes classifier is given by,

$$P(w_1, ..., w_n|c) = \prod_{i}^{n} P(w_i|c_i). \tag{7}$$

One of the reasons for using the Multinomial Naïve Bayes is due to research presented by Kibriys *et al.* which pointed out the great result from a Multinomial Naïve Bayes classification in the area of text classification [6].

2.4 Stochastic Gradient Descent (SGD)

SGD has in the academic community been selected as one of the most prominent machine learning algorithms to apply to text classification problems, where the data are abundant as in a vocabulary [9]. The mathematical formula behind SGD is illustrated in formula (8), and if the reader is interested in the background around it, Bottou provides a more in-depth explanation in *"Large-scale machine learning with stochastic gradient descent"* [9].

$$w_{t+1} = w_t - \gamma_t \Delta_w Q(z_i, w_t) \tag{8}$$

where: w : weight vector,
γ : a pre-defined gain,
$Q(z, w)$: minimized loss function,
z : randomly picked example.

2.5 Logistical Regression (LR)

LR comes in many forms; binary, ordinal and multinomial, in our case LR will refer to the multinomial logistic regression. Where the multinomial generalizes the binary LR to a multi-class problem, thus the output vector has more than two states (eighth in this case). The analytical formula for LR is illustrated in equation (9) and for the interested

reader a more verbose explanation is made available by Krishnapuram *et al.* in *"Sparse multinomial logistic regression: Fast algorithms and generalization bounds"* [10].

$$P(y^{(i)} = 1|\boldsymbol{x}, \boldsymbol{w}) = \frac{\boldsymbol{x}e^{\boldsymbol{w}^{(i)^T}}}{\sum_{j=1}^{m} \boldsymbol{x}e^{\boldsymbol{w}^{(j)^T}}}. \quad (9)$$

where: $\boldsymbol{w}^{(i)}$: weight vector corresponding to class i,
$y^{(i)} = 1$: true prediction corresponding to class i,
$y^{(i)} = 0$: false prediction corresponding to class i,
$\boldsymbol{x}$: the input vector,
^{T} : the superscript^T is the transpose.

2.6 Metrics

The following metrics were used to evaluate and validate the output from the used classifiers.

2.6.1 Accuracy–Score

The accuracy is the number of correct predictions made by the classifier divided by the total number of predictions [11], illustrated in the formula below:

$$\mathcal{A} = \frac{tp + tn}{tp + tn + fp + fn} \quad (10)$$

where: fn : false negative,
fp : false positive,
tn : true negative,
tp : true positive.

Despite seeming like a precise metric, accuracy can be misleading if it is the only metric used, due to if tn is large the result will be twisted.

2.6.2 Recall–Score

This metric describes the ability of the classification to label the correct with missing [11].

$$\mathcal{R} = \frac{tp}{tp + fn} \quad (11)$$

2.6.3 Precision–Score

This metric is an indication on how well the classification has performed regarding correct classification divided with the total classification of that class [11].

$$\mathcal{P} = \frac{tp}{tp + fp} \quad (12)$$

2.6.4 F1–Score

The F1 score is a combination of the precision and recall in one of Pythagoras means (harmonic mean) between the two [11].

$$\mathcal{F}_1 = \frac{2 \cdot \mathcal{P}\mathcal{R}}{\mathcal{P} + \mathcal{R}} \quad (13)$$

2.6.5 Confusion Matrix

This is a metric design to give a good overview of the result, where high numbers along the diagonal represent desirable results. An example of a confusion matrix is illustrated in table 1.

TABLE 1: Example of confusion matrix.

		Expected	
		True	False
Predicted	True	tp	fp
	False	fp	tf

N

3 Data Set

The data set in question is a result of querying the API[2] provided by the Swedish parliament. The API provides several different formats for the API request, e.g., JSON, HTML with more. In this project, HTML has been the favored data transfer format, due to the helpfull Python[3] library `BeautifulSoup`, which parses HTML.

In the files retrieved from the API contained political motions, from any party in the Swedish parliament, thus giving eight unique parties, which was used as a label corresponding to the respective document.

Furthermore, the pre-processing and data representation. Several Python libraries have been used to develop the classification system, the list of the most important follows below:

- `Pandas`[4] – An open source data analysis library, which provides excellent data structures to use in text mining.
- `scikit`[5] – Open source data mining and analyzing library, where the project classifiers are imported from.
- `SpaCy`[6] – An natural language processing library.
- `BeautifulSoup` – Great library for HTML parsing.
- `Pickle`[7] – A library for serialization of data structures.

4 Methodology

The project can be divided into three distinct sub-sections; *data collection*, *pre-processing* and *training and analyses*.

4.1 Data Collection

The data collection process began with the task of acquiring the necessary amount of political document to perform text mining. The area of *motions* was selected, this because motions should contain concrete politics reflecting the parties opinions. All motions can be listed, and from which the HTTP requests yields a HTML version of the original `PDF` or `docx`.

To avoid making the HTTP request on every execution the collected data was saved with the help of the Python library `Pickle` which makes it easy to save and load the Python data structure e.g. `Dictionary`.

2. http://data.riksdagen.se/
3. https://www.python.org/
4. https://pandas.pydata.org/
5. https://scikit-learn.org
6. https://spacy.io/
7. https://docs.python.org/3/library/pickle.html

4.2 Pre-processing

Pre-processing was one of the most difficult and important areas of the project, to clean up the inconsistently structured data to be able to get reasonable classification results. The data cleaning was done in different steps, first the extraction of the plain-text from the HTML version of the motions and the corresponding labels of the text, the author party. To extract the author party prove to be one of the troublesome thing in the project due to the inconsistency of the structures in the provided files, e.g. the party was mostly labeled as ”(V)” or ”(MP)” but in some cases ”(båda M)” and sometimes at multiple places, which made it tough to create robust and trustworthy regular expression to extract those.

When the text and the party labels had been extracted, the task to remove numerical values, stop words and converting it to lowercase where conducted. In regards to stemming of the data, recent studies by Schofield *et al.* have indicated that stemming can produce a worse result compared to none-stemming, so it was left out [12].

4.3 Training & Analysis

Lastly, in the training and analysis the 1700 documents collected, where divided in to test and training data (20% to 80%). Then from the training data the different classifiers was trained with the unigram (Bag of Words) or TF-IDF data structure as input. The prediction from the classifiers then was displayed with 5 different metrics; accuracy, recall, precision, F1, and confusion matrix.

5 Result

The data collection process yielded 1700 number of political motions, authored by one of the eight Swedish parliament parties. The distribution among them will be presented in figure 1.

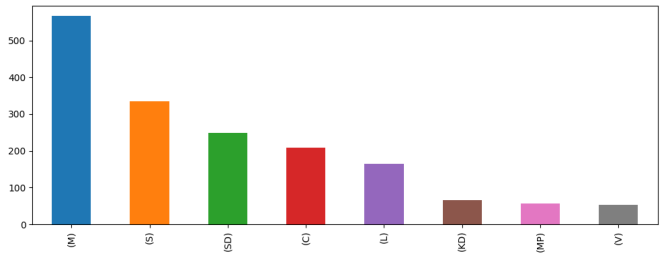

Fig. 1: Distribution of the collected motion among the parties.

5.1 Metrics

All of the measured metrics on the different classifiers will be presented below. Beginning with the class independent metric accuracy for the different classifiers. Then moving forward to the class dependent metrics; precision, recall, and F1. A weighted average has been introduced to simplify the discussion about the class dependent metrics. Thus the weighted average yields a summary perspective of the results.

5.1.1 Accuracy

In table 2, the different accuracies are presented, and we can see that both the LR and SGD perform well according to this metric, as well as the poor performance of the Multinomial Naïve Bayes classifier trained on the TF-IDF data representation.

TABLE 2: The accuracy metric for the different classifiers.

Machine Learning Algorithm	Accuracy in %
Multinomial Naïve Bayes (BoW)	73.5
Multinomial Naïve Bayes (TF-IDF)	58.0
Stochastic Gradient Descent	82.7
Logistic Regression	88.4

5.1.2 Precision

The precision for the different parties and classifiers are found in tables 3, 4, 5 and 6. When observing the weighted averages for the different classifiers, we see that the best performing classifier is the LR with a precision of 89% and the worst performing, the Multinomial Naïve Bayes (TF-IDF) with 69%.

5.1.3 Recall

Continuing on to the recall metric for the parties in the classifiers, see tables 3, 4, 5 and 6 . Where we also observe the weighted averages for the classifiers and see that the LR still performs the best with 88% and the Multinomial Naïve Bayes (TF-IDF) is still the worst with 58%.

5.1.4 F1

F1, as presented earlier a harmonic average between precision and recall, where we observe in the tables 3, 4, 5 and 6 that the best performances is found by the LR 88% and the worst, still Multinomial Naïve Bayes (TF-IDF) with 53%.

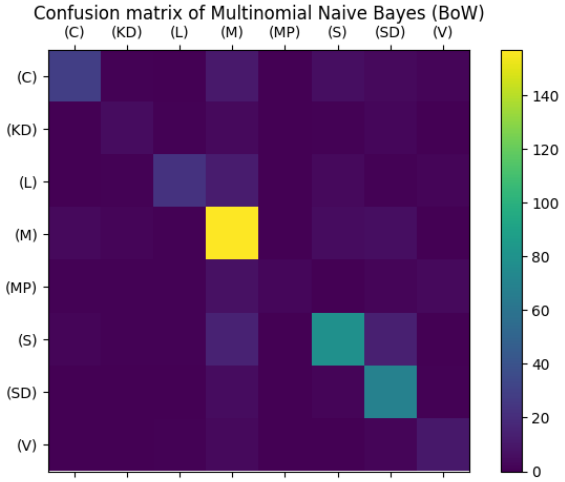

Fig. 2: Confusion matrix for the Multinomial Naïve Bayes (BoW).

5.1.5 Confusion Matrix

The generated confusion matrices are illustrated in the figures 2, 3, 4 and 5. Where we can see that the performance for the Multinomial Naïve Bayes trained on the Bag of Words representation outshines the Multinomial Naïve

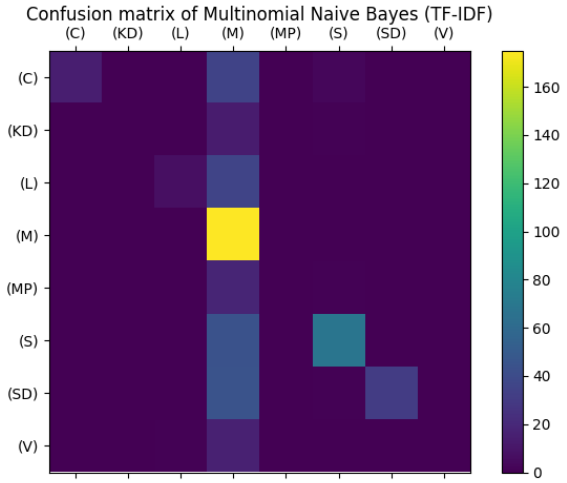

Fig. 3: Confusion matrix for the Multinomial Naïve Bayes (TF-IDF).

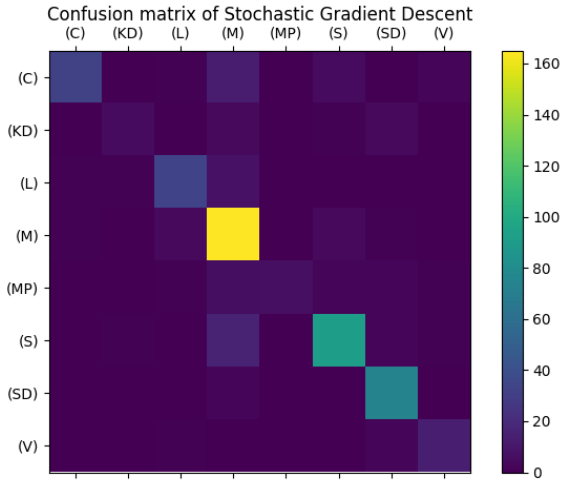

Fig. 4: Confusion matrix for the Stochastic Gradient Decent.

Bayes trained on the TF-IDF data representation. Furthermore, we also see that both the SGD and LR works excellent on the data.

6 Discussion

With the result in mind, we can see that the Multinomial Naïve Bayes (BoW) classifier performed reasonable in regards to the five applied metrics and can act as a good baseline for comparison. As it can be observed in tables

TABLE 3: Metrics for Multinomial Naïve Bayes (BoW).

Party	Precision	Recall	F1-Score	Support
C	0.81	0.55	0.65	53
KD	0.45	0.36	0.40	14
L	0.82	0.53	0.65	43
M	0.73	0.90	0.81	175
MP	1.00	0.16	0.27	19
S	0.81	0.71	0.76	112
SD	0.81	0.71	0.76	112
V	0.53	0.59	0.56	17
Weighted Average	0.75	0.74	0.72	510

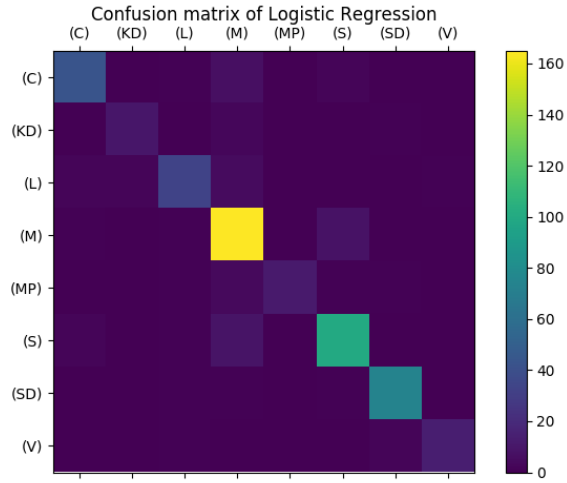

Fig. 5: Confusion matrix for the Logistic Regression.

TABLE 4: Metrics for Multinomial Naïve Bayes (TF-IDF).

Party	Precision	Recall	F1-Score	Support
C	1.00	0.28	0.44	53
KD	0.00	0.00	0.00	14
L	0.88	0.16	0.27	43
M	0.46	1.00	0.63	175
MP	0.00	0.00	0.00	19
S	0.92	0.61	0.73	112
SD	1.00	0.40	0.57	77
V	0.00	0.00	0.00	17
Weighted Average	0.69	0.58	0.53	510

3-6 there are two algorithms LR and SGD who performs superior to the baseline and one who performs worse, the Multinomial Naïve Bayes (TF-IDF). The fact that both LR and SGD outperforms the baseline is not unexpected, because the literature regards both of them as useful algorithms for classification of natural language texts [13]. Then if we look at why the Multinomial Naïve Bayes (TF-IDF) failed to capture the true nature of the data, there

TABLE 5: Metrics for Stochastic Gradient Descent.

Party	Precision	Recall	F1-Score	Support
C	0.94	0.60	0.74	53
KD	0.71	0.36	0.48	14
L	0.82	0.77	0.80	43
M	0.77	0.94	0.85	175
MP	1.00	0.37	0.54	19
S	0.88	0.82	0.85	112
SD	0.87	0.96	0.91	77
V	0.78	0.82	0.80	17
Weighted Average	0.84	0.83	0.82	510

TABLE 6: Metrics for Logistic Regression.

Party	Precision	Recall	F1-Score	Support
C	0.90	0.81	0.85	53
KD	0.83	0.71	0.77	14
L	0.85	0.77	0.80	43
M	0.85	0.94	0.89	175
MP	1.00	0.63	0.77	19
S	0.89	0.89	0.89	112
SD	0.95	0.96	0.95	77
V	0.93	0.82	0.87	17
Weighted Average	0.89	0.88	0.88	510

could, of course, be many reasons for this but one identified problem was that the party "(M)" is overrepresented in the number of documents they authored in the given time interval. However, it still came as a surprise to the author that the TF-IDF was outperformed by the Multinomial Naïve Bayes classifier trained on the Bag of Words data representation.

Another possible error source for the classification is the problematic task to acquire enough data to satisfy the general case fitting. The problem with acquiring enough data has its roots in the poorly structured API of the Swedish parliament, where the lack of consistency between the documents puts a pin in the wheel of effective document collection. However, that comes as no surprise, because there is nothing but the Swedish principle of open government that forces the availability of the documents.

Continuing to the research question for the project, which have yielded positive result for the SGD and LR and negative result for the Multinomial Naïve Bayes (TF-IDF) classification compared to the baseline.

Furthermore, possible improvements areas have been identified as, more training and testing data, at this point the document count is 1700 which is roughly half of the motions from the period 2017-2018. Moreover, a different variation of $\mathcal{N}$-gram should be tested and evaluated to use the $\mathcal{N}$ with the best performance. Lastly, a trial with a different classification model would bring new insight to the project.

7 Conclusion

The conclusions that can be derived from this project is the importance of coherence in the data, which will be used in the project. In this case, the data collected from the Swedish parliament API contained inconsistency, which required much time and effort to clean up.

Moreover, the findings that both LR and SGD successfully can classify the original author faction of a political motion with an accuracy of 88.4% and 82.7% compared to the baseline Multinomial Naïve Bayes (BoW), which accuracy was 73.5%. The less successful classification was made with the Multinomial Naïve Bayes (TF-IDF) trained classifier, who had an accuracy of only 58%.

Lastly, the use of classification in the area of politic, more special predicting the author of a document could be a great tool for the future if this approach were developed beyond the scope of this course e.g. see the work of Yan *et al.* [14]. Finalizing with the reflection that with more time and effort the classification of the documents original author is a possibility when appropriate classification algorithms are chosen.

References

[1] T. Lin and S. Wang, "Cloudlet-screen computing: a multi-core-based, cloud-computing-oriented, traditional-computing-compatible parallel computing paradigm for the masses," in *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pp. 1805–1808, IEEE, 2009.

[2] T. G. Dietterich, P. Domingos, L. Getoor, S. Muggleton, and P. Tadepalli, "Structured machine learning: the next ten years," *Machine Learning*, vol. 73, no. 1, p. 3, 2008.

[3] R. Choudhry and K. Garg, "A hybrid machine learning system for stock market forecasting," *World Academy of Science, Engineering and Technology*, vol. 39, no. 3, pp. 315–318, 2008.

[4] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities," in *Proceedings of the ASE BigData & SocialInformatics 2015*, p. 28, ACM, 2015.

[5] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-gram models of natural language," *Computational linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[6] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive bayes for text categorization revisited," in *Australasian Joint Conference on Artificial Intelligence*, pp. 488–499, Springer, 2004.

[7] S. Vijayarani, M. J. Ilamathi, and M. Nithya, "Preprocessing techniques for text mining-an overview," *International Journal of Computer Science & Communication Networks*, vol. 5, no. 1, pp. 7–16, 2015.

[8] H. Zhang, "The optimality of naive bayes," *AA*, vol. 1, no. 2, p. 3, 2004.

[9] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, pp. 177–186, Springer, 2010.

[10] B. Krishnapuram, L. Carin, M. A. Figueiredo, and A. J. Hartemink, "Sparse multinomial logistic regression: Fast algorithms and generalization bounds," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 6, pp. 957–968, 2005.

[11] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[12] A. Schofield, M. Magnusson, and D. Mimno, "Understanding text pre-processing for latent dirichlet allocation," in *Proceedings of the 15th conference of the European chapter of the Association for Computational Linguistics*, vol. 2, pp. 432–436, 2017.

[13] L. Zhou, J. K. Burgoon, D. P. Twitchell, T. Qin, and J. F. Nunamaker Jr, "A comparison of classification methods for predicting deception in computer-mediated communication," *Journal of Management Information Systems*, vol. 20, no. 4, pp. 139–166, 2004.

[14] H. Yan, A. Lavoie, and S. Das, "The perils of classifying political orientation from text," *Linked Democracy: Artificial Intelligence for Democratic Innovation*, vol. 858, p. 8, 2017.