Contribution to the research problem of Hate Speech Detection by comparing classical Machine Learning models and neural networks

Lukas Dech (W18b, Matr.-Nr 9376)
Mara Niggemann (W18b, Matr.-Nr 9110)

Abstract

Automatic Hate Speech detection becomes more important as more users produce content on Social Media platforms daily and legislation is confronting this problem with regulations. In this study, we aim at making a contribution to automatic Hate Speech detection by developing Machine Learning (ML) models and neural networks that automatically detect Hate Speech. We utilized feature engineering for four different algorithms and trained them in Weka. Furthermore, we established two identical neural networks, which were trained with different quantities of training data. In the experiment we alter the architecture, test different optimizers, and modify the epochs and batch size. We evaluate the performances of all models and draw conclusions from comparing both approaches with each other.

1 Introduction (Mara Niggemann)

Social Media content on Facebook, Twitter and miscellaneous is growing rapidly. For instance, the number of active Facebook users is increasing constantly, reaching a new pinnacle in Q4 2019 with roughly 2.4 billion users worldwide (Facebook, 2020b). Furthermore, the daily usage time of social networks such as Instagram, Snapchat and Facebook is predicted to grow by 4.3-13.8 % in 2020 (eMarketer, 2020). Another occurrence of this growing user behavior is the rise of offensive language in user-generated content. The online Social Media platforms are confronted with the question how offensive and inappropriate content should be detected and managed. Since 2016 the European Union commission initiated legislation to decrease Hate Speech and required Facebook, Youtube, Twitter and Microsoft to remove illegal Hate Speech in less than 24 h. (Hern, 2016) Such task is not trivial because of the gross amount of user-generated content daily, which is even increasing as the statistics emphasize. Automated methods that make use of Natural Language Processing (NLP) and Machine Learning (ML) have been actively researched in recent years to provide a solution for this problem. However, the field of automated Hate Speech detection remains a strenuous task (Vidgen et al.). The goal of this work is to make a contribution to Hate Speech detection. In this paper, we address the terminology, the status and challenges of research in Hate Speech detection in a first step. Secondly, we analyze the performance and comparability of the two state-of-the-art approaches by experimenting with ML with feature engineering and neural networks. In our evaluation we assess the comparability between both approaches and identify strengths and weaknesses for their applicability. A more detailed description of the methodology is provided in chapter 4.

2 Terminology (Mara Niggemann)

While the Hate Speech detection problem has received an accumulating amount of attention in recent years, scholars do not use a unique term for defining the task. MacAvaney et al. (2019) show that there does not exist a universally accepted definition of Hate Speech. We have examined existing definitions of Hate Speech published by various sources and have concluded that an agreeable intersection can be formed.
According to the Council of Europe, “Hate Speech covers all forms of expressions that spread, incite, promote or justify racial hatred, xenophobia, anti-Semitism or other forms of hatred based on intolerance.” (Freedom of Expression, 2020) Facebook’s understanding of Hate Speech is: “We

define Hate Speech as a direct attack on people based on what we call protected characteristics — race, ethnicity, national origin, religious affiliation, sexual orientation, caste, sex, gender, gender identity, and serious disease or disability." (Facebook, 2020a) Twitter speaks in their rules and policies of "Hateful conduct", defining it as the following: "You may not promote violence against or directly attack or threaten other people on the basis of race, ethnicity, national origin, caste, sexual orientation, gender, gender identity, religious affiliation, age, disability, or serious disease." (Twitter, 2020) Many authors aggregate all kinds of subtasks under general terms such as "abusive language", "harmful speech", "Hate Speech", "toxic language" or "profanity". (Fortuna & Nunes, 2018; Nobata, Tetreault, Thomas, Mehdad, & Chang, 2016; Schmidt & Wiegand, 2017)

In another study authors propose a two-fold typology for subtasks, that considers first whether the abuse is directed at a specific target and secondly the degree to which it is explicit. (Waseem, Davidson, Warmsley, & Weber, 2017) De facto the exemplary selected definitions address different aspects of Hate Speech. All of them explicitly state that Hate Speech has specific targets. Accordingly, this seems to be the agreed intersection. Some mention the aspect of inciting violence and hate, while another not solidly confirmed aspect is the intention of attacking or diminishing. For this reason we agree with and use for this work the rather openly formulated definition from Fortuna and Nunes (2018): "Hate Speech is language that attacks or diminishes, that incites violence or hate against groups, based on specific characteristics such as physical appearance, religion, descent, national or ethnic origin, sexual orientation, gender identity or other, and it can occur with different linguistic styles, even in subtle forms or when humour is used."

3 Related Work (Mara Niggemann)

Research on Hate Speech detection has been conducted even before the turn of the century (Spertus, 1997). Implicationally, many scholars have advanced with the subject. Especially since 2014 the number of publications per year in the "Computer Science and Engineering" domain has increased (Fortuna & Nunes, 2018). There are two survey studies that have investigated approaches used in the field of Hate Speech detection by other researchers (Fortuna & Nunes, 2018; Schmidt & Wiegand, 2017). Considering that these works offer an in-depth analysis of the state of the art, we present more recent work in the following.

Most research is conducted with the English language. Recently, research is extended to multilingual classifiers. Patricia Chiril et al. (2019) developed a multitarget and multilingual Hate Speech detection system using both features-based models and a neural model focusing on Hate Speech towards immigrants, women and sexism. The models were developed for French and English. They used unigrams, bigrams and trigrams as a baseline in all experiments. Feature-based models were equipped with surface features (such as the length in words, presence of punctuation marks, URLs, and user mentions), Sentiment features and Emojis features (by building an Emojis lexicon). In their experiments they tested a) Naïve Bayes, Logistic Regression, SVM, Decision Tree and Random Forest models with different combinations of features and b) a bidirectional LSTM for the neural model. For the Hate Speech task, Random Forest using as features the length of the tweet and the number of words in a profanity lexicon performed best.

Sigurbergsson and Derczynski (2019) investigated four automatic classifiers working both on Danish and English. The authors identified a Fast-BiLSTM model as their best performing one for English, while Danish worked better with Logistic Regression.

Vu et al. (2020) present Hate Speech detection on Vietnamese text at the VLSP workshop 2019. The best performing approach among the teams is a Bi-LSTM.

Besides multilingual approaches researchers are experimenting with different classifiers and are comparing different approaches with each other.

Zhang and Luo (2019) apply Deep Neural Networks to tackle the problem of the lack of unique, discriminative features which are more difficult to discover. The authors adapted the models from other ML tasks and proved their benefit on Hate Speech detection.

MacAvaney et al. (2019) propose a multi-view SVM model, thus combining multiple SVMs as it was previously confirmed as an effective approach in image processing tasks. In the Hate Speech application, the meta-classifier outperformed the single view classifiers.

Mou et al. (2020) designed a new Hate Speech detection framework by implementing CNNs for sub-word information extraction, LSTMs for word-level information extraction, as well as statistics for better feature extraction. Moreover, the authors incorporated phonetic-level information. The performance of their model was above modern approaches.
Finally, while many scholars progressed in the field of Hate Speech detection, Arango et al. (2019) show that often there is an overestimation of the performance of state-of-the-arts models by replicating experiments. The authors expose methodological issues and identify the impacts of biases in dataset annotations.

4 Methodology (Mara Niggemann)

The main goal of this paper is to contribute to the problem solving of automatic Hate Speech detection. We want to accomplish this goal by developing and comparing models based on classical ML algorithms and a neural network.
The classical ML models were trained with Weka, on open-source ML software. We engineered features and applied four different algorithms. In chapter 4.3 we describe the features that have been implemented. In chapter 4.4 we delineate the different algorithms that have been utilized. After the training we evaluated the performance of each feature. We did not alter, add, or delete features for our models. In chapter 6 we have presented findings from the analysis of the features and the final model.
Simultaneously, we trained two neural networks. The only aspect altered is the amount of data used to train the model. The architecture and the training procedure are described in chapter 4.4. Our results follow in chapter 5, an evaluation is provided in chapter 6 as well.

4.1 Data (Lukas Dech)

The basic dataset for both approaches included 900 labeled comments and comes from Davidson et al. (2017). Furthermore, we used a bad word list and a good word list[1] for the feature engineering for the classical ML models. We added more data to the basic dataset to train the second neural network. The combined dataset consists of 250,000 labeled comments. The datasets come from the platform Kaggle[2]. The reason behind adding more data for the neural network is, because in some cases the results with neural networks directly correlate with the amount of data used in the training process (Alom et al., 2019). Both neural networks have been trained with the glove pre-trained word embedding with 100 dimensional vectors. The bad word list was also integrated into the embedding layer.

4.2 Pre-Processing (Lukas Dech)

Pre-Processing has been kept at a minimum to reduce the influence of manipulation. Especially in the case of the neural network, too much pre-processing and manipulation might have a negative impact on the results of a neural network (Crone, Lessmann, & Stahlbock, 2006). Therefore, we only deleted empty comments and eliminated comments in other languages than English.

4.3 Feature Engineering (Mara Niggemann)

The first feature implemented was the number of capitalized words in a comment. With a variable the number of words with a capitalized letter at the beginning was counted. We extended this feature to also calculate the ratio of capitalized words. Therefore, the number of capitalized words was divided by the total number of words in the given comment.
As a third feature we utilized the length of a given comment, because we observed the tendency of non-Hate Speech comments to be longer than Hate Speech comments. In our next feature we investigated the occurrence of bad words in the text. We used the bad-word dictionary and compared each word in the comment text with the dictionary. As feature we counted the number of bad words in a comment. Complementary we assumed that the opposite of bad words might be helpful as well. Accordingly, we calculated the number of nice words in a comment with the nice-word dictionary. Next, we examined the presence

[1] https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf

[2] https://www.kaggle.com/pandeyakshive97/hate-speech-dataset
https://www.kaggle.com/vkrahul/twitter-hate-speech?select=train_E6oV3lV.csv
https://www.kaggle.com/mrinaal007/hate-speech-detection?select=toxic_train.csv
https://www.kaggle.com/mrinaal007/hate-speech-detection?select=toxic_test.csv

of punctuation marks in the comment text. Our attention especially fell on the marks "@", "!", "?" and "$". The frequency of these punctuation marks in the comment text was used as feature. Furthermore, we also had the number of pronouns in a comment counted. We perceived this could be a beneficial feature because Hate Speech is often targeted at a person. Additionally, we figured that Hate Speech comments contained more URLs. We therefore looked for the presence of URLs and counted the frequency in each comment. Besides, we have observed that in Hate Speech comments a slang spelling is increasingly used. For example, in the English ing-form ("seeing") the "g" is often omitted ("seein"). We implemented the levenshtein-distance as feature to catch such intentional misspelling.

4.4 Model training (ML: Mara Niggemann, NN: Lukas Dech)

In this chapter we first describe the training process for the ML models. Afterwards, we delineate the same for the neural networks.
For the classical ML models, we experimented with 4 different algorithms in Weka as optimizer. The first one is the stochastic gradient descent (SGD). SGD picks a random sample from the dataset, which is the initial value. The gradient is the slope of the cost function. The parameters are iteratively adjusted until the gradient becomes minimal. (Witten, Pal, Frank, & Hall, 2017, p. 38)
Secondly, we trained a SVM. SVMs are capable of the Kernel trick and therefore can optimally divide a dataset into two classes through a hyperplane. SVMs are commonly applied in text classification tasks such as detecting spam or sentiment analysis. Further, they are utilized in image recognition. (Witten et al., 2017, p. 37)
The third algorithm we tried is Naïve Bayes, which is a probabilistic algorithm. It is based on the Bayes theorem and assumes, that the presence of a feature in a class is not related to the presence of other features. Naïve Bayes are known to learn fast and are used for multiclass prediction. Many studies apply Naïve Bayes in text classification tasks such as Sentiment Analysis. (Witten et al., 2017, p. 35)
The last algorithm tested is the Logistic Regression (LR). We chose LR because the classification task is a binary problem (Hate-Speech or No Hate-Speech). Like Linear Regression, LR tries to fit the Sigmoid function to the observations. (Witten et al., 2017, p. 37)

The neural networks have been trained on a tesla K80 GPU. The first layer was designed as an embedding layer with the pre-trained word embedding described in chapter 4.1. 40 % of the connections were dropped out directly after the embedding layer to prevent overfitting. We used 4 LSTM layers each with 128 units as hidden layers. The first three layers return a sequence, which enabled us to connect the LSTM layers with each other. After the 4th layer we dropped out 40 % of the connections again. Since the data must be categorized into two categories only, we used a dense layer with two neurons and a sigmoid activation function as output layer. We used the RMSprop optimizer to compile the model. The best results have been achieved when the model was trained with 20 epochs and a batch size of 128. The sequence length has been limited to 100. For evaluation and optimization during the training process, 10% of the training data was separated to form an evaluation set. Since we only distinguish between two distinct classes, we experimented with categorical crossentropy or binary crossentropy as loss function.

5 Results

The results produced in Weka for the ML approach can be seen in the appendix A.
The neural network trained with the basic dataset of 900 comments scored 93.5 % (accuracy, precision, and recall) on the evaluation set, and 88 % on the test set.
The network trained with the extended dataset of 250,000 comments had a precision, recall and accuracy of 96 % on the evaluation set and 93 % on the test set.[3]

6 Evaluation

6.1 Classical ML models (Mara Niggemann)

With Weka we analyzed the utility of the features for the models. In Weka, a list of the features can be created, ranking each feature according to the

[3] Link to Git-Repository of our project: https://github.com/lukas328/Hate-Speech-detection-with-DL-and-ML

gain of informational content. The feature with the greatest benefit is the frequency of bad words in a comment. The number of capitalized words and the ratio of capitalized words are on the second and third place. The frequency of pronouns was ranked 4th, the frequency of hyperlinks 5th and the length of the comment 6th. The features levenshtein distance, frequency of nice words in the text, and the occurrence of punctuation marks did not have a significant impact at all.

Overall, three of the four models showed a similar performance. SGD, SVM and LR all had an accuracy of 86 % on the test set. We did not observe significant differences in the performance of these algorithms. The confusion matrices show that the models are more effective in predicting Non-Hate Speech correctly (True Negatives). The models had more difficulties with the False Positives. About 27.5 % ($\frac{11}{40}$) were predicted as Non-Hate Speech when the comments actually were labeled as Hate Speech. We judge this is a less desirable outcome, as the models should rather not leave out Hate Speech comments if they really are Hate Speech.

Naïve Bayes performed worse with an accuracy of 77 %. The algorithm achieved slightly better results with the False Positives than the others but has a worse performance on the True Negatives. Essentially, our results in ML are below the results of other studies by renowned researchers. Surely the cause for this occurrence is the quality and informational content of the features we programmed, as they mostly were simple surface features. Better success might have been possible if we had programmed better linguistic and word generalizing features. Another possible influence for better performance could have been more thorough pre-processing.

6.2 Neural Network (Lukas Dech)

We experimented with different architectures for the neural network. We also tested the impact of additional dense layers, consisting of different numbers of neurons with a sigmoid activation function. However, the results were worse and severely raised the runtime. Secondly, we also tried different optimizers. The Adam and RMSprop optimizers produced the best results, with RMSprop performing best in most cases. Subsequently, we tested higher dropouts and changed the number of epochs and the batch size. However, this led to faster learning of the model and we figured it could overfit easily. With 10 epochs the model always scored 98-99 % on the training set. This score was significantly higher than the result on the evaluation set and the test set, which is a signal for overfitting. Furthermore, we attempted to reach better performance through higher dimensioned word embeddings. In case of a small sequence of numbers in a comment the additional embeddings could not imply a better performance. On top of that the sequence length has been altered as well, which did not improve performance. More LSTM-units did not induce better results either. Overall, all models scored similarly in accuracy, recall and precision on the evaluation set, ranging between 93-96 %. We did not realize any difference in performance with either loss function.

We observed that some comments were always misclassified. For instance, the comment *"are you a pig or are you a bird"* was labeled as non-Hate Speech, but the model always predicted the contrary. The sentence has many stop words and therefore low informational content. In order to comprehend it as non-Hate Speech one must be aware of the context. Another misclassification example was "*Yankees Damn you DirectTV blackout the Yankee game. With a freaking WALK OFF! You're killing me satellite douche.*" This comment was classified as Hate Speech because it contains many features that could signal a Hate Speech uttering. In a nutshell, it is difficult to predict Hate Speech correctly if the context cannot be considered. Our model had difficulties with the correct prediction of non-Hate Speech comments if many bad words were included in the text. Also, if a person was compared to something bad the model made a false prediction. False predicted Hate Speech was mostly misclassified when words with a bad connotation were used. In that case we often questioned the label itself as well.

6.3 Comparison of both approaches (Lukas Dech)

Our study demonstrates that both approaches potentially create good results in Hate Speech detection. In our case, the neural networks had a better performance. However, related work from other researchers shows that ML can produce equally good results. We conclude that the quality of our features is the main cause for the weaker performance of our ML models.

A key factor for the performance of neural networks is the amount of data used for the training. Our research confirms the fact, that neural networks require a higher amount of data to perform better, as the model trained on 250,000 comments reached higher scores in accuracy, precision and recall than the model trained on the basic dataset only. Another challenge we observed is the comprehension of how the neural network must be manipulated in case it classifies specific comments wrong. Due to the depth of the network, we could not identify reasons for the misclassification. Accordingly, it is hard to determine which parameters must be changed. On the contrary, ML is easier to manipulate because the utility of the features is determinable.

7 Conclusion

Despite of receiving an accumulating amount of attention in research, automatic Hate Speech detection remains a strenuous task. The importance of solving the problem is increasing due to the growing amount of user generated content on Social Networks daily. Moreover, legislation pressurizes platforms to manage Hate Speech rigorously and quickly. Our goal with this work was to contribute to the problem solving of automatic Hate Speech detection by developing multiple models using both state-of-the-art approaches (ML and neural network) and comparing them with each other. We developed four classical ML models with Weka, trained on a dataset consisting of 900 labeled comments. We engineered 9 features and employed the four algorithms SGD, SVM, LR and Naïve Bayes. Our best feature was the frequency of bad words in a comment, followed by the number and ratio of capitalized words in a comment. Three of the four algorithms (SGD, SVM, LR) showed similar results in performance with an accuracy of 86 % on the test set. These models were more effective in the True Negatives, and the confusion matrices showed they performed worse on the False Positives. We concluded that this was a less desirable outcome. In our experiment, Naïve Bayes overall accomplished worse results, but was slightly better with the False Positives. In summary for ML, our results were below the ones from renowned researchers, and we accounted the quality of our features for that.

We also built two neural networks with the same architecture but used different amounts of training data. The first network was trained on the basic data set while for the second model we extended the training data set to 250,000 comments. The architecture of the neural networks consisted of an embedding layer with pre-trained word embeddings, four hidden LSTM layers each with 128 units and a 40 % dropout after the first and after the last layer again to prevent overfitting. A dense layer with two neurons and sigmoid activation function was used as output layer. We experimented with the architecture of the networks, tested different optimizers, changed epochs and batch size, sequence length and the number of LSTM units to optimize the model. Eventually, we finished with an accuracy, precision and recall of 93.5 % for the neural network trained on 900 comments, and 96 % for the neural network trained on 250,000 comments. Our evaluation showed that specific comments recurrently were misclassified and identified the causes for that.

In our comparison of both approaches we analyzed that the amount of data available for training is one of the main influencing factors on the performance of neural networks. However, the experiment showed that with neural networks it is more difficult to comprehend and manipulate parameters effectively if certain misclassifications occur frequently.

Lastly, we would like to emphasize the need for further research in the field of Hate Speech detection, as the task is not yet fulfilled.

References

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., et al. (2019). A State-of-the-Art Survey on Deep Learning Theory and Architectures. *Electronics, 8*(3), 292.

Arango, A., Pérez, J., & Poblete, B. (2019). Hate Speech Detection is Not as Easy as You May Think. In B. Piwowarski, M. Chevalier, E. Gaussier, Y. Maarek, J.-Y. Nie, & F. Scholer (Eds.), *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 45–54). [S.l.]: Association for Computing Machinery.

Crone, S. F., Lessmann, S., & Stahlbock, R. (2006). The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research, 173*(3), 781–800.

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). *Automated Hate Speech Detection and the Problem of Offensive Language,* from https://arxiv.org/pdf/1703.04009.

eMarketer (2020). *Wachstumsprognose der durchschnittlichen täglichen Nutzungsdauer für Snapchat, Instagram und Facebook in den USA im April 2020.* Statista GmbH. Retrieved November 07, 2020, from https://de.statista.com/statistik/daten/studie/1119644/umfrage/wachstum-der-nutzungsdauer-fuer-social-media-in-den-usa/.

Facebook (2020a). *Community Standards.* Retrieved November 18, 2020, from https://m.facebook.com/communitystandards/hate_speech/.

Facebook (2020b). *Umsatz und Nettoergebnis von Facebook weltweit in den Jahren 2007 bis 2019 (in Millionen US-Dollar) [Graph].* Retrieved November 07, 2020, from https://de.statista.com/statistik/daten/studie/217061/umfrage/umsatz-gewinn-von-facebook-weltweit/.

Fortuna, P., & Nunes, S. (2018). A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys, 51*(4), 1–30.

Freedom of Expression (2020). *Hate Speech.* Retrieved November 18, 2020, from https://www.coe.int/en/web/freedom-expression/hate-speech.

Hern, A. (2016, May 31). Facebook, YouTube, Twitter and Microsoft sign EU hate speech code. *The Guardian.* Retrieved November 18, 2020, from https://www.theguardian.com/technology/2016/may/31/facebook-youtube-twitter-microsoft-eu-hate-speech-code.

MacAvaney, S., Yao, H.-R., Yang, E., Russell, K., Goharian, N., & Frieder, O. (2019). Hate speech detection: Challenges and solutions. *PloS one, 14*(8), e0221152.

Mou, G., Ye, P., & Lee, K. (10192020). SWE2: SubWord Enriched and Significant Word Emphasized Framework for Hate Speech Detection. In M. d'Aquin, S. Dietze, C. Hauff, E. Curry, & P. Cudre Mauroux (Eds.), *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (pp. 1145–1154). New York, NY, USA: ACM.

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., & Chang, Y. (2016). Abusive Language Detection in Online User Content. In J. Bourdeau (Ed.), *Proceedings of the 25th International Conference on World Wide Web, Montreal, Canada, May 11 - 15, 2016* (pp. 145–153). Geneva: International World Wide Web Conferences Steering Committee.

Patricia Chiril, Farah Benamara, Véronique Moriceau, Marlène Coulomb-Gully, & Abhishek Kumar (2019). Multilingual and Multitarget Hate Speech Detection in Tweets. *Actes de la Conférence sur le Traitement Automatique des Langues Naturelles (TALN) PFIA 2019. Volume II : Articles courts*, 351–360.

Schmidt, A., & Wiegand, M. (2017). A Survey on Hate Speech Detection using Natural Language Processing. In L.-W. Ku & C.-T. Li (Eds.), *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media* (pp. 1–10). Stroudsburg, PA, USA: Association for Computational Linguistics.

Sigurbergsson, G. I., & Derczynski, L. (2019). *Offensive Language and Hate Speech Detection for Danish,* from https://arxiv.org/pdf/1908.04531.

Spertus, E. (1997). Smokey: Automatic Recognition of Hostile Messages. In *Proceedings of IAAI-97, the 9th Conference on Innovative Application of Artificial Intelligence* (97th ed., pp. 1058–1065).

Twitter (2020). *Hateful conduct policy.* Retrieved November 18, 2020, from https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy.

Vidgen, B., Harris, A., Nguyen, D., Tromble, R., Hale, S., & Margetts, H. Challenges and frontiers in abusive content detection. In S. T. Roberts, J. Tetreault, V. Prabhakaran, & Z. Waseem (Eds.), *Proceedings of the Third Workshop on Abusive Language Online* (pp. 80–93). Stroudsburg, PA, USA: Association for Computational Linguistics.

Vu, X.-S., Vu, T., Tran, M.-V., Le-Cong, T., & Nguyen, H. T. M. (2020). *HSD Shared Task in VLSP Campaign 2019:Hate Speech Detection for Social Good,* from https://arxiv.org/pdf/2007.06493.

Waseem, Z., Davidson, T., Warmsley, D., & Weber, I. (2017). Understanding Abuse: A Typology of Abusive Language Detection Subtasks. In Z. Waseem, W. H. K. Chung, D. Hovy, & J. Tetreault (Eds.), *Proceedings of the First Workshop on Abusive Language Online* (pp. 78–84). Stroudsburg, PA, USA: Association for Computational Linguistics.

Witten, I. H., Pal, C. J., Frank, E., & Hall, M. A. (2017). *The WEKA Workbench: Online Appendix for: "Data Mining: Practical machine learning tools and techniques"*. Retrieved December 10, 2020, from https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf.

Zhang, Z., & Luo, L. (2019). Hate speech detection: A solved problem? the challenging case of long tail on twitter. *Semantic Web, 10*(5), 925–945.

Appendix

Appendix A: Results

SGD

```
=== Summary ===

Correctly Classified Instances         755               83.7029 %
Incorrectly Classified Instances       147               16.2971 %
Kappa statistic                          0.674
Mean absolute error                      0.163
Root mean squared error                  0.4037
Relative absolute error                 32.5943 %
Root relative squared error             80.7394 %
Total Number of Instances              902

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,892     0,218     0,804       0,892   0,846      0,678    0,837     0,772     0
                 0,782     0,108     0,878       0,782   0,827      0,678    0,837     0,795     1
Weighted Avg.    0,837     0,163     0,841       0,837   0,837      0,678    0,837     0,783

=== Confusion Matrix ===

   a   b   <-- classified as
 403  49 |   a = 0
  98 352 |   b = 1
```

Trainingset

```
Correctly Classified Instances          86               86      %
Incorrectly Classified Instances        14               14      %
Kappa statistic                          0.7208
Mean absolute error                      0.14
Root mean squared error                  0.3742
Total Number of Instances              100

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,939     0,216     0,807       0,939   0,868      0,730    0,862     0,788     0
                 0,784     0,061     0,930       0,784   0,851      0,730    0,862     0,840     1
Weighted Avg.    0,860     0,137     0,870       0,860   0,859      0,730    0,862     0,814

=== Confusion Matrix ===

  a  b   <-- classified as
 46  3 |  a = 0
 11 40 |  b = 1
```

Testset

SVM

```
Correctly Classified Instances         755               83.7029 %
Incorrectly Classified Instances       147               16.2971 %
Kappa statistic                          0.674
Mean absolute error                      0.163
Root mean squared error                  0.4037
Relative absolute error                 32.5944 %
Root relative squared error             80.7396 %
Total Number of Instances              902

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,892    0,218    0,804      0,892    0,846      0,678    0,837     0,772     0
                 0,782    0,108    0,878      0,782    0,827      0,678    0,837     0,795     1
Weighted Avg.    0,837    0,163    0,841      0,837    0,837      0,678    0,837     0,783

=== Confusion Matrix ===

   a   b   <-- classified as
 403  49 |   a = 0
  98 352 |   b = 1
```

Trainingset

```
Correctly Classified Instances          86               86      %
Incorrectly Classified Instances        14               14      %
Kappa statistic                          0.7208
Mean absolute error                      0.14
Root mean squared error                  0.3742
Total Number of Instances              100

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,939    0,216    0,807      0,939    0,868      0,730    0,862     0,788     0
                 0,784    0,061    0,930      0,784    0,851      0,730    0,862     0,840     1
Weighted Avg.    0,860    0,137    0,870      0,860    0,859      0,730    0,862     0,814

=== Confusion Matrix ===

 Nicht HS  HS
  a   b   <-- classified as
 46   3 |   a = 0
 11  40 |   b = 1
```

Testset

Logistic Regression

```
Correctly Classified Instances         758               84.0355 %
Incorrectly Classified Instances       144               15.9645 %
Kappa statistic                          0.6806
Mean absolute error                      0.2542
Root mean squared error                  0.3543
Relative absolute error                 50.8465 %
Root relative squared error             70.8585 %
Total Number of Instances              902

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,898    0,218    0,806      0,898    0,849      0,685    0,893     0,869     0
                 0,782    0,102    0,884      0,782    0,830      0,685    0,893     0,897     1
Weighted Avg.    0,840    0,160    0,845      0,840    0,840      0,685    0,893     0,883

=== Confusion Matrix ===

   a   b   <-- classified as
 406  46 |   a = 0
  98 352 |   b = 1
```

Trainingset

```
Correctly Classified Instances          86               86      %
Incorrectly Classified Instances        14               14      %
Kappa statistic                          0.7208
Mean absolute error                      0.2444
Root mean squared error                  0.3478
Total Number of Instances              100

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,939    0,216    0,807      0,939    0,868      0,730    0,879     0,853     0
                 0,784    0,061    0,930      0,784    0,851      0,730    0,879     0,907     1
Weighted Avg.    0,860    0,137    0,870      0,860    0,859      0,730    0,879     0,880

=== Confusion Matrix ===

 a  b   <-- classified as
46  3 |   a = 0
11 40 |   b = 1
```

Testset

Naive Bayes

```
Correctly Classified Instances          77               77      %
Incorrectly Classified Instances        23               23      %
Kappa statistic                          0.5393
Mean absolute error                      0.2738
Root mean squared error                  0.3978
Relative absolute error                 54.7478 %
Root relative squared error             79.5477 %
Total Number of Instances              100

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,735    0,196    0,783      0,735   0,758      0,540    0,863     0,821     0
                 0,804    0,265    0,759      0,804   0,781      0,540    0,863     0,903     1
Weighted Avg.    0,770    0,231    0,771      0,770   0,770      0,540    0,863     0,863

=== Confusion Matrix ===

  a  b   <-- classified as
 36 13 |  a = 0
 10 41 |  b = 1
```

Trainingset

```
Correctly Classified Instances          77               77      %
Incorrectly Classified Instances        23               23      %
Kappa statistic                          0.5393
Mean absolute error                      0.2738
Root mean squared error                  0.3978
Total Number of Instances              100

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,735    0,196    0,783      0,735   0,758      0,540    0,863     0,821     0
                 0,804    0,265    0,759      0,804   0,781      0,540    0,863     0,903     1
Weighted Avg.    0,770    0,231    0,771      0,770   0,770      0,540    0,863     0,863

=== Confusion Matrix ===

   Nicht HS  HS
   a  b   <-- classified as
 36 13 |  a = 0
 10 41 |  b = 1
```

Testset