BMT23
5th International Conference
Business Meets Technology.
Valencia, 13th-15th July 2023

THE SOCIAL MEDIA HATE SPEECH BAROMETER: MAKING OF

Sauer, Sebastian [a1]; Piazza, Alexander [a2]; and Schacht, Sigurd [a3]

[a]*University of Applied Sciences Ansbach. Germany. ([a1] sebastian.sauer@hs-ansbach.de, [a2] alexander.piazza@hs-ansbach.de, [a3] sigurd.schacht@hs-ansbach.de)*

ABSTRACT: Hate speech, particularly on social media channels, is a pressing cybersecurity concern and can even threaten the very foundations of societal stability. While there is a growing body of literature on how to detect and mitigate hate speech, applied researchers lack a state-of-the-art yet easily accessible infrastructure to build their own hate speech detection pipelines. We aim to provide an example of such an infrastructure that can serve as a template for other researchers. The infrastructure we present is based on the latest machine learning technologies available in the R environment: The Tidymodels framework and its extension Tidytext, plus the Targets project management approach, are the building blocks of our proposed infrastructure. In short, our data pipeline starts with downloading and pre-processing tweets, using various methods to convert text into numerical information. We then apply state-of-the-art supervised machine learning pipelines, drawing on a range of learning algorithms and incorporating new tuning capabilities. The focus of this paper is to explain the setup and rationale of the infrastructure. Our infrastructure is freely available on Github at https://github.com/sebastiansauer/hate-speech-barometer.

KEY WORDS: *Hate speech; machine learning; cybersecurity; natural language processing; artificial intelligence.*

1. INTRODUCTION

According to the United Nations, “hate speech” can be defined as offensive discourse targeting a group or an individual based on personal characteristics such as race, religion, or gender.[1] The UN amends that hate speech may threaten social peace. Although there is a lack of a widely accepted definition, the UN proposes the following definition of hate speech:

> any kind of communication in speech, writing or behaviour, that attacks or uses pejorative or discriminatory language with reference to a person or a group based on who they are, in other words, based on their religion, ethnicity, nationality, race, colour, descent, gender or other identity factor.

[1] https://www.un.org/en/hate speech/understanding-hate speech/what-is-hate speech, accessed 2024-05-24

How to cite: Sauer, S., Piazza, A., and Schacht, S. 2023. The social media hate speech barometer: Making of. In Proc.: *5th International Conference Business Meets Technology*. Valencia, 13th-15th July 2023. 143-154. https://doi.org/10.4995/BMT2023.2023.16724

Although hate speech is nothing new, it has been given a boost by the internet, which has made it possible for threats, conspiracies, and lies to travel quickly throughout the globe (Castaño-Pulgarín et al., 2021). Hate speech is having a visible impact on society: there are many commonalities between the January 2023 assaults on Brazil's government buildings,[2] and the attack on the US Capitol on January 6, 2021, including that each event happened after certain groups continuously used threatening language and false allegations against others. According to a BBC news article, online hate speech in the UK and US has risen by approx. 20% since the start of the Covid pandemic.[3] Given the surge of hate speech, defense mechanisms are on the rise too, albeit without being able to turn the tide, at least so far. For example, Iceland's government is the 34th to join ratification concerning the criminalization of acts of a racist and/or xenophobic nature committed through computer systems.[4] Some researchers have even put forward the hypothesis of a causal link between social media use and offline violence (Calvert, 1997; Chan et al., 2016; Cinelli et al., 2021). Carley (2020) summarizes that hate speech constitutes a major threat not only for democracy and civil rights including freedom, but also for individual mental and psychosocial health. For example, Wypych & Bilewicz (2022) conducted an online survey among N=726 Ukrainian immigrants living in Poland. The authors aimed at investigating the association between exposure to hate speech, stress, and mental health. They conclude that (prolonged) exposure to hate speech causes mental health problems of the target population. In sum, albeit a monetary or similar quantification is difficult, it can be concluded that hate speech is a substantial menace to society. It is the aim of the research presented in this paper to fight back hate speech by fostering research endeavors for detecting hate speech.

1.2 Related work

It is important to address hate speech to prevent violence against protected characteristics and to promote a safe and respectful online environment. However, setting limits on speech at a global scale in various languages and cultures is complex and identifying hate speech can be difficult in an online global community. One aspect that contributes to the difficulties in hate speech detection is that false negatives (missing hate speech) and false positive (false accusing of hate speech) are like Scylla and Charybdis, the opposing monsters of the ethical consequences of faults and shortcomings in such decision.[5] Augmenting the already high difficulties in detecting hate speech is that annotators are not necessarily reliable, and a universal definition of hate speech does not exist (as to yet).

Different methodologies for detecting hate speech have been developed and are in widely circulated use, comprising deep learning, shallow learning, and text-mining (non-machine learning) approaches. One basic text mining approach is a keyword-based

[2] https://en.wikipedia.org/wiki/2023_Brazilian_Congress_attack

[3] https://www.bbc.com/news/newsbeat-59292509, accessed 2024-05-24

[4] https://www.coe.int/en/web/cyberviolence/-/iceland-joins-the-first-additional-protocol-to-the-convention-on-cybercrime-on-countering-xenophobic-and-racist-acts-committed-through-computer-systems, accessed 2024-05-24

[5] cf. https://about.fb.com/news/2017/06/hard-questions-hate-speech/, accessed 2024-05-24

method, where an ontology or dictionary is used to identify text containing potentially hateful keywords (MacAvaney et al., 2019). However, simply using a hateful slur is not enough to constitute hate speech according to a study of different definitions of hate speech (MacAvaney et al., 2019). More advanced techniques include machine learning models ranging from word count methods (e.g., TFIDF) to complex BERT models (Jahan & Oussalah, 2021). Successful detection models use more than one approach, including hybrid models that combine different techniques for more accurate results (Alkomah & Ma, 2022). Advancements in natural language processing (NLP) and machine learning have greatly improved the detection of hate speech. With the help of machine learning algorithms, particularly deep neural networks, NLP can be used to identify linguistic patterns and features that are indicative of hate speech (Jahan & Oussalah, 2021; Pang, 2022; Velankar et al., 2022; Yin & Zubiaga, 2021). Various approaches have been used to detect specific features or linguistic patterns that denote hate speech in text, including rule-based classification models and, more recently, a proliferation of deep learning methods like Long Short-Term Memory networks and Transformer-based architectures (Malik et al., 2022).

Whereas hate speech detection is an active field of investigation, the border between hate speech and other forms of questionable social behavior is blurry. For example, bot detection is an emerging research (and engineering) branch that has sparked a substantial number of research activities. For a research overview, see Cresci (2020).

1.3 Methodology

Machine learning (ML) is often considered as a subset of artificial intelligence (AI). AI is a broad field that aims to emulate human abilities, while machine learning focuses on training a machine to learn and adapt through experience Bakshi & Bakshi (2018). ML constitutes the intersection between statistics and computer science, and its rapid progress have largely been driven by the ongoing reduction in computational costs (Jordan & Mitchell, 2015). In its score, ML is a new interpretation of the old quest of finding patterns in data. Correlations, which have been a subject of statistical studies at least since a century, are among the most prototypical examples of how patterns in data can be grasped. Once pattern have been found in the data, predictions can be inferred. The action of reducing a data set with many variables to a (potentially very) simple rule, is what has been dubbed a “model” (Stigler, 2016). In fact, the usefulness of a model hinges on its ability to be reductive. To be clear, there is not causal knowledge necessary for some model to predict some event, which probably fueled its widespread use given the fact that causal knowledge is very hard to gain and surpasses a purely statistically oriented research agenda (cf. Pearl, 2009). ML algorithms of the present day are highly flexible allowing to “fit an elephant”, as von Neumann remarked to a similar matter.[6] On the pro side, highly flexible algorithms can pick up even minute and complex patterns in data, which may in some circumstances be useful, as many phenomena, particularly in

[6] https://math.stackexchange.com/questions/2970219/was-von-neumann-right-that-with-four-parameters-you-can-fit-an-elephant, accessed 2024-05-24

the social sciences, tend to behave in complex ways. However, there are drawbacks of highly flexible algorithms as well: Such algorithms tend to perceive signals where in fact there only is noise, a phenomenon well known as Pareidolia in perception research, and as overfitting in ML. To be fair, one may argue that human suffer from Pareidolia at least as much as machines do. However, countermeasures against overfitting are in place. Two common procedures are, described in high-level terms, (1) testing the model's predictions on new data, data unknown to the model, and (2) "prune" or "penalize" the model for complexity, to strike a balance between unnecessary complexity and exaggerated parsimony (James et al., 2021).

1.4 Purpose and value added of the paper

Given the social impact of hate speech and the vibrant advances in ML, applied researchers desperately need tools and templates to investigate social science research questions. Even without being experts in machine learning, social scientists need access to state-of-the-art tools. This research aims to support this by providing a template for hate speech detection. Our target audience is social scientists with intermediate technical knowledge in statistics and ML. Fortunately, typical ML pipelines, at least in their basic form, are quite mechanical and simple, and can therefore be automated quite easily. However, given the prosperity and rapid progress of ML, it would be inappropriate to provide polished point-and-click interfaces. Rather, script-based approaches to ML pipelines are advantageous because they can be quickly adapted to new developments. Indeed, most new developments in statistics and ML, at least in the last few years, have been in the R and Python programming languages. For this reason, we provide a template that makes use of the R language and its rich ecosystem of statistics and ML tools. Our aim is to make it easier for applied researchers in the social sciences to conduct their own hate speech analyses without having to worry too much about the intricate technicalities of ML.

2. RESEARCH DESIGN

This paper describes a tool that facilitates a classical ML pipeline focused on hate speech detection. The source code is freely available online.[7] To this end, we provide a typical ML pipeline, including well-known steps such as tuning different ML algorithms and applying resampling schemes. In addition, we have made use of GNU-Make-like project management tools to improve reproducibility and usability, see details below. The results of the analyses facilitated by this project could be summarized with plots such as Figure 1. Next, we describe the design ideas of this project.

[7] https://github.com/sebastiansauer/hate-speech-barometer

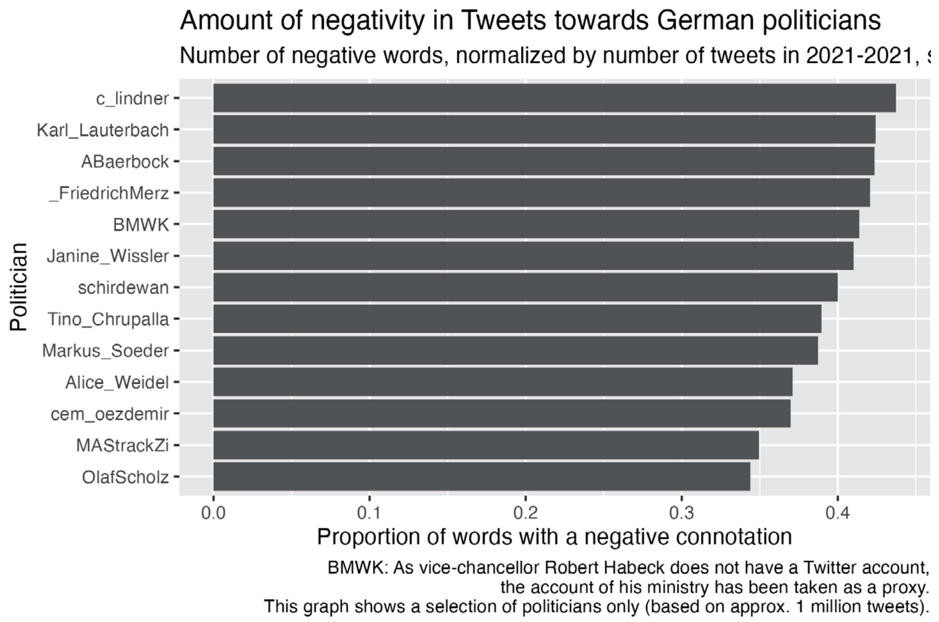

Figure 1. Hate speech proxies based on Tweets to German politicians.

2.1 Reproducibility

Reproducibility has been described as a hallmark of science (Plesser, 2018), and the present research builds strongly on this idea. Our analysis is based entirely on scripts that are freely available and licensed under the GNU General Public Licence. Git has been used as a versioning tool so that all changes to the code base can be made explicit. A package management tool (renv; Ushey (2023)) is used to ensure that users have the correct version of the R packages. The training and test samples are openly available (cf. Wiegand, 2019), so it is easy to compare your own results with those of the tool presented here.[8]

2.2 State-of-the-art shallow learner via "Tidymodels"

We used the Tidymodels framework (Kuhn & Wickham, 2020) as our ML API. Tidymodels in turn builds on the idea of "Tidyverse" (Wickham et al., 2019), an idiosyncratic approach that tries to strike a balance between being powerful enough to produce high performance models and being easy to use. Some of the packages in the Tidyverse ecosystem are among the most downloaded and relied upon R packages.[9] The Tidyverse authors state

8 https://heidata.uni-heidelberg.de/dataset.xhtml?persistentId=doi:10.11588/data/0B5VML

9 https://www.r-pkg.org/, accessed 2023-05-25

that the “primary goal of the Tidyverse is to facilitate a conversation between a human and a computer about data” [Wickham et al. (2019); p1]. One advantage of any (good) approach that is widely accepted is that it provides a standard for how things should be done. Perhaps one of the main reasons for the success of the Tidyverse is that it addresses key problems faced by data practitioners and strikes a sensible balance between conflicting goals. In short, the authors describe their design principles as (a) human-centeredness, meaning that the software is designed to be read and written by humans, and only for computers to execute, similar to literate programming (Knuth, 1984), (b) consistency, so that all functions work in a similar way, (c) additivity, so that complex problems can be solved by breaking them down into small pieces, and (d) inclusivity, so that the community can participate in development. A more detailed introduction to Tidymodels is given by Silge & Kuhn (2022).

Tidymodels support a wide range of features that encompass recent requirements for ML software. The most important is the unified API for all ML algorithms and the complete coverage of all (typical) ML steps. For example, Tidymodels allows intelligent tuning of grid search methods such as simulated annealing. It provides outer and inner loops in cross-validation and includes pre-processing in cross-validation (e.g., tuning the number of components in a PCA). It provides many steps that makes data pre-processing simple such as dummyfying nominal variables, effect-coding them or over-/undersample their levels in the case of a class imbalance. Due to the rich ML ecosystem in R, from which many ML algorithms are made available in Tidymodels, users can choose from a wide array of state-of-the-art algorithms.

2.3 Project management via “Targets”

It has been said that perhaps the hardest problem in computer science is naming objects.[10] Then perhaps the next most difficult is dealing with complexity, at least from a helicopter perspective. To illustrate how complexity can creep in, consider the following example. Given a set of 10 possible actions, where you must choose the right 3 to solve a problem, you are left with 120 possibilities (as combinatorial mathematics requires). However, given a situation where you must choose 3 from a set of 20 again, you are faced with an enormous 1120 possible combinations (3 out of 30: 4060). In short, there’s an explosion of complexity. Even a moderate increase in the number of possible actions can dramatically increase the number of possible combinations to choose from. The bad news is that there’s no way out, thanks to the purity of mathematics. The good news is that the only thing to do is to reduce complexity to a level that is just low enough to be manageable. That’s where project management comes in. There are many aspects to project management in software development; A well-known idea is “don’t repeat yourself” (DRY), which could be translated as using macros (functions) to avoid repetition in code (Hunt & Thomas, 2000). A key feature of R is its functional programming orientation, which allows code to be cleanly composable. The project management tool used in this project is called “Targets”,

[10] https://stackoverflow.com/questions/33497879/why-is-the-hardest-part-of-programming-is-naming-things

which is built around functional programming ideas (Landau, 2021). It is a GNU-Make like pipeline toolkit for R. Like Make, Targets ensures that the objects in a pipeline are updated when and only when necessary. That is, if an “upstream” object changes, and if that object is an input to a downstream object, then (and only then) will the downstream object be updated. Given the high cost of computation, it can be vital to know when an update is not needed. On the other hand, it is equally important not to miss an update when it is out of date. In short, as a project management toolkit, Targets (a) updates objects in a pipeline, and (b) keeps the pipeline tidy. An example is given below.

3. RESULTS

3.1 ML pipeline of the hate speech barometer

Figure 2 shows the pipeline of the hate speech barometer;[11] an interactive version of the diagram is available online.[12] In this graph, each node describes a target, and each edge shows dependencies between the targets with the arrows heading downstream. The appendix provides an overview in tabular form of the targets of the ML pipeline.

In the following, we describe the steps of the pipeline in some detail so that practitioners know what each step accomplishes. Instead of a “step” the term “target” could be used when seen from a functional programming view, focusing on the value (or output, result) of a function. For each step (or target) of the pipeline, we provide its name as used in the code along with a short description of what is achieved by the step. Where the step is complex enough to merit its own function, we provide the URL to the function.[13]

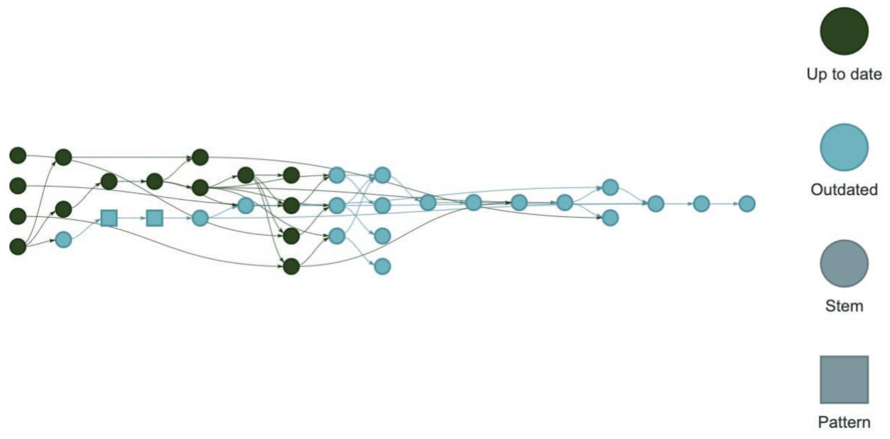

Figure 2. pipeline of the hate speech barometer.

[11] a “Pattern” refers to an object which is looped over; “Stems” are single-element Targets

[12] https://sebastiansauer.github.io/hate-speech-barometer/tar-visnetwork-pipeline3.html

[13] Please note that the URLs to the functions are still subject to change as the project is an early development phase.

3.2 Data pre-processing

Path: Defines the (relative) paths to the data.[14]

D_train and d_test: Imports train- and test-sample (based on the paths, that's why the object path is the input for this target).[15]

Recipe2, recipe2_prepped, d_train_baked, d_test_baked: Defines the pre-processing "recipe" of the data (prior to modelling) and applies it to the data sets.[16]

Recipe_plain, recipe_plain_prepped: As the data pre-processing was time consuming and had no tuning parameters it was taken out from the model workflow (and the cross validation) and conducted before the modelling, to save computation time. During the modelling workflow, a minimal pre-processing ("recipe") took place.

3.3 Modelling

Model_lasso, model_boost, model_rf: These three learning algorithms, i.e., the Lasso (L1 penalized regression), gradient boosting, random forests were computed in this analysis.[17]

Wf1, wf2, wf3: In Tidymodels, a workflow (wf) consists of pre-processing and the ML algorithm plus optional postprocessing. wf1 is the workflow consisting of the Lasso as ML algorithm; wf2: boosting, wf3: random forest. The pre-processing was identical in all three workflows.[18]

Wf1_fit etc.: The cross-validated, tuned workflow, i.e., the fitted model, where the model parameters have been estimated.[19]

Wf1_autoplot etc.: Diagrams depicting model performance (mean and sd) according to the selected performance measures (ROC-AUC in this case).

Wf_fits_l, wf_fits_roc, wf_fits_best: All models stored in a list-object (wf_fits_l) in order to render access to the model performance simple; wf_fits_roc contains all the performance measures (ROC-AUC), and wf_fits_best identifies the model exhibiting the best fit.

Wf3_finalized, final_fit, preds_test: The best performing workflow is chosen and initiated with the best performing tuning parameters (wf3_finalized), then, the whole train sample is fit with the best model, giving final_fit; on this base the test sample is predicted (preds_test).

[14] https://github.com/sebastiansauer/hate-speech-barometer/blob/main/R/set-path.R
[15] https://github.com/sebastiansauer/hate-speech-barometer/blob/main/R/read-test-data.R
[16] https://github.com/sebastiansauer/hate-speech-barometer/blob/main/R/def-recipes.R
[17] https://github.com/sebastiansauer/hate-speech-barometer/blob/main/R/def-models.R
[18] https://github.com/sebastiansauer/hate-speech-barometer/blob/main/R/tune-wf.R
[19] https://github.com/sebastiansauer/hate-speech-barometer/blob/main/R/tune-wf.R

3.3.1 Tweet classification

Tweets_path: A folder containing tweet data; in case of changes in the folder the target will be updated.

Tweets, tweets_df: The tweets are imported into R (tweets) using parallel processing due to large size; to save computational time, a random sample of tweets can be drawn (tweets_df).

Tweets_baked: The tweets are subjected to the same pre-processing as the train sample.

Preds, tweets_baked_preds: The tweets get classified (predicted) and the predictions are added as an extra column to the processed tweets (tweets_data_preds).

3.3.2 Pipeline outcomes

Preds_summarized: Proportion of hate speech per Twitter account, per year, s. Fig.[20]

Preds_summarized_plot: Plot for preds_summarized.[21]

3.4 Constants

Any ML pipelines depends on some constants, for organizational reasons (e.g., paths) or for model (hyper)parameters, to name two usual suspects. In this project, users can easily change their configuration in a (yaml) text file called config.yml. Advanced users can make use of the different Git branches of this project; whereas the main branch provides the standard pipelines, the “dev” branch offers more pipelines and experimental features.

4. CONCLUSIONS

4.1 Limitations

As this project is an early development phase, there are several threads suitable for further buildup. For example, the documentation of the project is still large lacking, which renders the access more difficult for less advanced users. In addition, deep learning methods are not yet implemented (although planned). Of course, users of any technical system strive for two opposing goals: feature richness and simplicity. The optimal balance between the two goals partly depends on the user’s background and goals. That said, this project draws from an array of tools which implies that the user is accustomed to these tools (R, Git, Github, Targets, Tidymodels) and the underlying theory. Limited knowledge will place a barrier to easy access to the system. A further issue is that working with text data can place substantial burden on the computational resources. As to yet the present tool is not yet fully optimized to saving resources.

[20] https://github.com/sebastiansauer/hate-speech-barometer/blob/main/R/helper-funs.R

[21] https://github.com/sebastiansauer/hate-speech-barometer/blob/main/R/plots.R

4.2 Practical implications

There is a substantial number of case studies and tutorial on ML pipelines freely available on the web. However, there's still, to the best of our knowledge, no similar or template for a complex data analysis incorporating Tidymodels and Targets or similar tools within the R programming language.

In sum, it is our hope that the present research contributes to the detection of hate speech by providing a scaffold to the applied researchers so that he or she can focus on the phenomenon of hate speech rather on the technical intricacies of ML.

5. APPENDIX

The following table provides an overview of the targets of the analytic pipeline of the hate speech barometer.

Target name	Definition in R code
model_rf	def_model_rf()
path	set_path()
model_lasso	def_model_logistic()
model_boost	def_model_boost()
tweets_path_files	path$tweets %>% list.files(full.names = TRUE, pattern = "rds$")
d_test	read_train_test_data(path$data_test)
d_train	read_train_test_data(path$data_train)
tweets_path	tweets_path_files
recipe2	def_recipe2(d_train)
tweets	tweets_path %>% read_and_select() %>% drop_na()
recipe2_prepped	prep(recipe2)
tweets_df	tweets %>% sample_n(size = config$n_rows) %>% drop_na() %>% group_by(id)
d_train_baked	bake(recipe2_prepped, new_data = NULL)
d_test_baked	bake(recipe2_prepped, new_data = d_test)
tweets_baked	bake(recipe2_prepped, new_data = tweets_df)
recipe_plain	def_recipe_plain(d_train_baked)
wf1	fit_wf(model_lasso, recipe_plain)
wf2	fit_wf(model_boost, recipe_plain)
wf3	fit_wf(model_rf, recipe_plain)
recipe_plain_prepped	prep(recipe_plain)
wf1_fit	tune_my_anova(wf1, data = d_train_baked)
wf2_fit	tune_my_anova(wf2, data = d_train_baked, grid = 1)
wf3_fit	tune_my_anova(wf3, data = d_train_baked, grid = 1)
wf1_autoplot	autoplot(wf1_fit)
wf2_autoplot	autoplot(wf2_fit)
wf3_autoplot	autoplot(wf3_fit)
wf_fits_l	list(wf1 = wf1_fit, wf2 = wf2_fit, wf3 = wf3_fit)
wf_fits_roc	wf_fits_l %>% map(~collect_metrics(.x) %>% filter(.metric == "roc_auc")) %>% list_rbind(names_to = "id") %>% arrange(-mean)
wf_fits_best	wf_fits_roc %>% slice_head(n = 1)
wf3_finalized	wf3 %>% finalize_workflow(wf_fits_best)
final_fit	fit(wf3_finalized, d_train_baked)
preds_test	predict(final_fit, d_test_baked)
preds	predict(object = final_fit, new_data = tweets_baked)
tweets_baked_preds	enrich_preds(tweets_df, preds, tweets_baked)
preds_summarized	summarise_preds(tweets_baked_preds)
preds_summarized_plot	plot_preds_summarized(preds_summarized)

REFERENCES

Alkomah, F., & Ma, X. (2022). A Literature Review of Textual Hate Speech Detection Methods and Datasets. *Information*, *13*(6, 6), 273. https://doi.org/10.3390/info13060273

Angra, S., & Ahuja, S. (2017). Machine learning and its applications: A review. *2017 International Conference on Big Data Analytics and Computational Intelligence (ICBDAC)*, 57–60. https://doi.org/10.1109/ICBDACI.2017.8070809

Bakshi, K., & Bakshi, K. (2018). Considerations for artificial intelligence and machine learning: Approaches and use cases. *2018 IEEE Aerospace Conference*, 1–9. https://doi.org/10.1109/AERO.2018.8396488

Calvert, C. (1997). Hate speech and its harms: A communication theory perspective. *Journal of Communication*, *47*(1), 4–19. https://doi.org/10.1111/j.1460-2466.1997.tb02690.x

Carley, K.M. (2020). Social cybersecurity: An emerging science. *Computational and Mathematical Organization Theory*, *26*(4), 365–381. https://doi.org/10.1007/s10588-020-09322-9

Castaño-Pulgarín, S.A., Suárez-Betancur, N., Vega, L.M.T., & López, H.M.H. (2021). Internet, social media and online hate speech. Systematic review. *Aggression and Violent Behavior*, *58*, 101608. https://doi.org/10.1016/j.avb.2021.101608

Chan, J., Ghose, A., & Seamans, R. (2016). The Internet and Racial Hate Crime: Offline Spillovers from Online Access. *Management Information Systems Quarterly*, *40*(2), 381–403. https://aisel.aisnet.org/misq/vol40/iss2/8

Cinelli, M., Pelicon, A., Mozetič, I., Quattrociocchi, W., Novak, P.K., & Zollo, F. (2021). Dynamics of online hate and misinformation. *Scientific Reports*, *11*(1, 1), 22083. https://doi.org/10.1038/s41598-021-01487-w

Cresci, S. (2020). A decade of social bot detection. *Communications of the ACM*, *63*(10), 72–83. https://doi.org/10.1145/3409116

Hunt, A., & Thomas, D. (2000). *The pragmatic programmer from journeyman to master*. Addison-Wesley.

Jahan, M.S., & Oussalah, M. (2021, May 22). A systematic review of Hate Speech automatic detection using Natural Language Processing, *arXiv:2106.00742*. https://doi.org/10.48550/arXiv.2106.00742

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning: With applications in R* (Second edition). Springer. https://link.springer.com/book/10.1007/978-1-0716-1418-1

Jordan, M.I., & Mitchell, T.M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260. https://doi.org/10.1126/science.aaa8415

Knuth, D.E. (1984). Literate Programming. *The Computer Journal*, *27*(2), 97–111. https://doi.org/10.1093/comjnl/27.2.97

Kuhn, M., & Wickham, H. (2020). *Tidymodels: A collection of packages for modeling and machine learning using tidyverse principles.* [Manual]. https://www.tidymodels.org

Landau, W.M. (2021). The targets R package: A dynamic Make-like function-oriented pipeline toolkit for reproducibility and high-performance computing. *Journal of Open Source Software*, *6*(57), 2959. https://doi.org/10.21105/joss.02959

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. https://doi.org/10.1371/journal.pone.0221152

Malik, J.S., Pang, G., & Hengel, A. van den. (2022, February 18). *Deep Learning for Hate Speech Detection: A Comparative Study*. https://doi.org/10.48550/arXiv.2202.09517

Pang, G. (2022, March 7). Deep Learning for Hate Speech Detection: A Large-scale Empirical Evaluation. *Medium*. https://towardsdatascience.com/deep-learning-for-hate-speech-detection-a-large-scale-empirical-evaluation-92831ded6bb6

Pearl, J. (2009). *Causality*. Cambridge university press.

Plesser, H.E. (2018). Reproducibility vs. Replicability: A Brief History of a Confused Terminology. *Frontiers in Neuroinformatics*, *11*, 76. https://doi.org/10.3389/fninf.2017.00076

Silge, J., & Kuhn, M. (2022). *Tidy Modeling with R*. https://www.tmwr.org/

Stigler, S.M. (2016). *The seven pillars of statistical wisdom*. Harvard University Press.

Ushey, K. (2023). *Renv: Project environments* [Manual]. https://rstudio.github.io/renv/

Velankar, A., Patil, H., & Joshi, R. (2022, September 12). *A Review of Challenges in Machine Learning based Automated Hate Speech Detection*. https://doi.org/10.48550/arXiv.2209.05294

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T., Miller, E., Bache, S., Müller, K., Ooms, J., Robinson, D., Seidel, D., Spinu, V., … Yutani, H. (2019). Welcome to the Tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Wiegand, M. (2019). *GermEval-2018 corpus (DE)*. heiDATA. https://doi.org/10.11588/data/0B5VML

Wypych, M., & Bilewicz, M. (2022). Psychological toll of hate speech: The role of acculturation stress in the effects of exposure to ethnic slurs on mental health among Ukrainian immigrants in Poland. *Cultural Diversity and Ethnic Minority Psychology*. https://doi.org/10.1037/cdp0000522

Yin, W., & Zubiaga, A. (2021). Towards generalisable hate speech detection: A review on obstacles and solutions. *PeerJ Computer Science*, *7*, e598. https://doi.org/10.7717/peerj-cs.598