**1.1 Hate speech is a concern for cyber security, social peace, and mental health**

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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn1). 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 on the basis of who they are, in other words, based on their religion, ethnicity, nationality, race, colour, descent, gender or other identity factor.

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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-castano-pulgarin_internet_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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn2), 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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn3). 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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn4). Some researchers have even put forward the hypothesis of a causal link between social media use and offline violence ([Cinelli et al. 2021](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-cinelli_dynamics_2021); [Calvert 1997](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-calvert_hate_1997); [Chan, Ghose, and Seamans 2016](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-chan_internet_2016)). Carley ([2020](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-carley_social_2020)) summarizes that hate speech constitutes a major threat not only to democracy and civil rights including freedom but also for individual mental and psychosocial health. For example, Wypych and Bilewicz ([2022](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-wypych_psychological_2022)) conducted an online survey among N=726 Ukrainian immigrants living in Poland. The authors aimed to investigate the association between exposure to hate speech, stress, and mental health. They conclude that (prolonged) exposure to hate speech causes mental health problems in 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 against hate speech by fostering research endeavors for detecting hate speech.

**1.2 Present state of hate speech detection research**

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 positives (false accusing of hate speech) are like Skylla and Charybdis, the opposing monsters of the ethical consequences of faults and shortcomings in such decision[5](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn5). 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 method, where an ontology or dictionary is used to identify text containing potentially hateful keywords ([MacAvaney et al. 2019](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-macavaney_hate_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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-macavaney_hate_2019)). More advanced techniques include machine learning models ranging from word count methods (e.g., TFIDF) to complex BERT models ([Jahan and Oussalah 2021](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-jahan_systematic_2021)). Successful detection models use more than one approach, including hybrid models that combine different techniques for more accurate results ([Alkomah and Ma 2022](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-alkomah_literature_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 and Oussalah 2021](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-jahan_systematic_2021); [Pang 2022](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-pang_deep_2022); [Yin and Zubiaga 2021](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-yin_towards_2021); [Velankar, Patil, and Joshi 2022](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-velankar_review_2022)). 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, Pang, and Hengel 2022](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-malik_deep_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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-cresci_decade_2020)).

**1.3 High-level ideas of machine learning**

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 and Bakshi ([2018](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-bakshi_considerations_2018)). ML constitutes the intersection between statistics and computer science, and its rapid progress has largely been driven by the ongoing reduction in computational costs ([Jordan and Mitchell 2015](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-jordan_machine_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 for at least a century, are among the most prototypical examples of how patterns in data can be grasped. Once patterns 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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-stigler_seven_2016)). In fact, the usefulness of a model hinges on its ability to be reductive. To be clear, there is no causal knowledge necessary in order for some model to predict some event, which probably fuelled 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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-pearl2009causality)). ML algorithms of the present day are highly flexible allowing them to “fit an elephant”, as von Neumann remarked in a similar matter[6](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn6). 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 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 humans 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, in order to strike a balance between unnecessary complexity and exaggerated parsimony ([James et al. 2021](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-james_introduction_2021)).

**1.4 Research gap and research objectives**

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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn7). 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 summarised with plots such as [Figure 1](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fig-sentiplot2). Next, we describe the design ideas of this project.

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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "ref-plesser_reproducibility_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 License. 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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-ushey_renv_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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-germeval)), so it is easy to compare your own results with those of the tool presented here.[8](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn8)

**2.2 State-of-the-art shallow learner via “Tidymodels”**

We used the Tidymodels framework ([Kuhn and Wickham 2020](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-kuhn_tidymodels_2020)) as our ML API. Tidymodels in turn builds on the idea of “Tidyverse” ([Wickham et al. 2019](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-wickham_welcome_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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn9). The Tidyverse authors state that the “primary goal of the Tidyverse is to facilitate a conversation between a human and a computer about data” [Wickham et al. ([2019](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-wickham_welcome_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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-knuth_literate_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 and Kuhn ([2022](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-silge_tidy_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 preprocessing 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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn10). 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 and Thomas 2000](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-hunt_pragmatic_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”, which is built around functional programming ideas ([Landau 2021](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#ref-landau_targets_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 ML pipeline of the hate speech barometer**

[Figure 2](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "fig-pipe3) shows the pipeline of the hate speech barometer[11](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "fn11); an interactive version of the diagram is available online[12](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html" \l "fn12). 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 ([Table 1](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#tbl-targets)).

Ein Bild, das Screenshot, Diagramm enthält.

Automatisch generierte Beschreibung

Figure 2: pipeline of the hate speech barometer

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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn13).

**3.1 Data preprocessing**

* path: Defines the (relative) paths to the data.[14](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn14)
* 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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn15)
* recipe2, recipe2\_prepped, d\_train\_baked, d\_test\_baked: Defines the preprocessing “recipe” of the data (prior to modelling) and applies it to the data sets.[16](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn16)
* recipe\_plain, recipe\_plain\_prepped: As the data preprocessing 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, in order to save computation time. During the modelling workflow, a minimal preprocessing (“recipe”) took place.

**3.2 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](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn17)
* wf1, wf2, wf3: In Tidymodels, a workflow (wf) consists of preprocessing and the ML algorithm plus optional postprocessing. wf1 is the workflow consisting of the Lasso as ML algorithm; wf2: boosting, wf3: random forest. The preprocessing was identical in all three workflows.[18](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn18)
* wf1\_fit etc.: The cross-validated, tuned workflow, i.e., the fitted model, where the model parameters have been estimated[19](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn19)
* 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) 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).

**3.2.1 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; in order to save computational time, a random sample of tweets can be drawn (tweets\_df).
* tweets\_baked: The tweets are subjected to the same preprocessing 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).

**3.2.2 Pipeline results**

* preds\_summarized: Proportion of hate speech per Twitter account, per year, s. Fig. [20](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn20)
* preds\_summarized\_plot: Plot for preds\_summarized[21](https://sebastiansauer.github.io/hate-speech-barometer/hate-speech-barometer-ms-v01.html#fn21)

**3.3 Constants**

Any ML pipeline 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 Limitations**

As this project is in an early development phase, there are several threads suitable for further buildup. For example, the documentation of the project is still largely lacking, which renders 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 a substantial burden on computational resources. As of yet the present tool is not yet fully optimized to save resources.

There is a substantial number of case studies and tutorials 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.