DETECCIÓN DE DISCURSO DE ODIO *ONLINE* UTILIZANDO MACHINE LEARNING

ONLINE HATE SPEECH DETECTION USING MACHINE LEARNING

TRABAJO FIN DE GRADO
CURSO 2021-2022

AUTORA
ELA KATHERINE SHEPHERD ARÉVALO

DIRECTORES
GONZALO MÉNDEZ POZO
PABLO GERVÁS GÓMEZ-NAVARRO

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

DETECCIÓN DE DISCURSO DE ODIO *ONLINE* UTILIZANDO MACHINE LEARNING

ONLINE HATE SPEECH DETECTION USING MACHINE LEARNING

TRABAJO DE FIN DE GRADO EN INGENIERÍA INFORMÁTICA
DEPARTAMENTO DE INGENIERÍA DEL SOFTWARE E INTELIGENCIA ARTIFICIAL

AUTORA
ELA KATHERINE SHEPHERD ARÉVALO

DIRECTORES
GONZALO MÉNDEZ POZO
PABLO GERVÁS GÓMEZ-NAVARRO

CONVOCATORIA: SEPTIEMBRE - 2022

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

16 DE SEPTIEMBRE DE 2022

INSCRIPTION

To myself, for pushing through all these years without even knowing where I was going.

ACKNOWLEDGEMENTS

Firstly, I'd like to thank Gonzalo Méndez and Pablo Gervás for guiding me through this project, and for being so patient and kind through all of it. I also want to thank my parents and my friends for supporting me throughout my degree, especially those in ASCII: Silvia, Belén, V, Zero, Fer, Resic, Becca, and so many others, who were my second home in the faculty. You guys helped me academically and emotionally, and I appreciate it so, so much.

But most of all of them, I want to help my partner and best friend, Pablo, for being my rock in every single possible way. This degree was so much easier because of you.

And finally, I want to thank anybody who reads this document, as it took a lot of time and effort to write, and it's my final goodbye to this degree.

RESUMEN

El discurso de odio dirigido a personas marginadas es un problema muy común en línea, especialmente en redes sociales como Twitter o Reddit. La detección automática del discurso de odio en dichos espacios puede ayudar a reparar Internet y a transformarlo en un entorno más seguro para todos. La detección del discurso de odio encaja en la clasificación de texto, donde se organiza en categorías. Este proyecto[1] propone el uso de algoritmos de *Machine Learning* para localizar discurso de odio en textos *online* en cuatro idiomas: inglés, español, italiano y portugués. Los datos para entrenar los modelos se obtuvieron de *datasets* disponibles públicamente en línea. Se han utilizado tres algoritmos diferentes con distintos parámetros para comparar su rendimiento. Los experimentos muestran que los mejores resultados alcanzan una precisión del 82,51 % y un valor F1 de alrededor del 83 % en italiano. Los resultados para cada idioma varían dependiendo de distintos factores.

Palabras clave

Inteligencia Artificial, *Machine Learning*, Discurso de odio, PLN, Clasificación de textos, Redes sociales, Twitter

[1] **Enlace al repositorio público del proyecto:** https://github.com/NILGroup/TFG-2122-HateSpeechDetection

ABSTRACT

Hate speech directed towards marginalized people is a very common problem online, especially in social media such as Twitter or Reddit. Automatically detecting hate speech in such spaces can help mend the Internet and transform it into a safer environment for everybody. Hate speech detection fits into text classification, a series of tasks where text is organized into categories. This project[2] proposes using Machine Learning algorithms to detect hate speech in online text in four languages: English, Spanish, Italian and Portuguese. The data to train the models was obtained from online, publicly available datasets. Three different algorithms with varying parameters have been used in order to compare their performance. The experiments show that the best results reach an 82.51% accuracy and around an 83% F1-score, for Italian text. Each language has different results depending on distinct factors.

Keywords

Artificial Intelligence, Machine Learning, Hate Speech, NLP, Text classification, social media, Twitter.

[2] **Link to public project repository:** https://github.com/NILGroup/TFG-2122-HateSpeechDetection

CONTENT INDEX

FIGURE INDEX

TABLE INDEX

EQUATION INDEX

Chapter 1 - Introduction

In this first chapter I will explain the motivation behind this project and my main objectives. I will also describe how this document is divided, briefly explaining each section.

My personal motivation for this project is as follows: as a bisexual person on the Internet who has presented as a woman all their life, I'm no stranger to hate speech from peers or people online, either towards me or towards my loved ones, so I believe that it is important to overview such content, not for the purpose of censoring, but to avoid people experiencing trauma from a young age and the possibility of other people falling into a bigoted mentality.

1.1 Motivation

Hate speech is described by the United Nations as "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".[3] It is a pandemic that has been rampant for a very long time, and in recent decades the rising of social media has made it easier to bully, mock and degrade oppressed social groups. Having the freedom to express every thought and feeling is a double-edged sword; when everybody is able to communicate everything with anybody without limits, hateful communities can form, new insults arise (Pascoe & Diefendorf, 2019) and targeted attacks can be planned (Bliuc, Faulkner, Jakubowicz, & McGarty, 2018). It has even been studied that online hate speech can predict violence: research carried out by (Blake, O'Dean, Lian, & Denson, 2021) shows a correlation between misogynistic tweets and domestic violence.

The task of this project is to discover efficient Machine Learning approaches to detect hate speech online. This means both aggressive hate speech, which incites

[3] Link: https://www.un.org/en/hate-speech/understanding-hate-speech/what-is-hate-speech

violence towards a target, and non-aggressive hate speech, which promotes harmful preconceived notions about marginalized people without encouraging violence. During this project different Machine Learning (ML) approaches will be used to classify hate speech in four languages: English, Spanish, Italian and Portuguese. These languages have been chosen for a number of reasons. Firstly, there is more available data to use for this kind of task in English than in any other language. At the same time, it is interesting to analyse current data in other languages to do a comparison of data quality and quantity, and see how this can affect performance of the algorithms. Spanish was the first language that came to mind when brainstorming, since this project's author and University are Spanish. Also, hate speech detection projects done on languages other than English are scarce. The other two remaining languages, Italian and Portuguese, were chosen because of their similarity to Spanish, both linguistically and in terms of amount of dedicated hate speech detection projects.

After the data has been classified, a comparison of the results will be done to reveal what algorithms detect hate speech better, and how other factors such as language play a role in accuracy.

1.2 Objectives

The main objectives that will be targeted in this project are:

- Choosing and developing ML algorithms able to predict, in four languages, whether a short piece of text contains hate speech or not.
- Reach a conclusion as to which ML algorithms and what variants are better at classifying hate speech, by comparing results measured by accuracy.
- Explain the causes for high or low accuracy across the different used models by examining patterns in the results and thus gaining a deeper understanding of the characteristics of all elements of each trained ML algorithm

1.3 Document structure

Chapter 2 contains information on all the concepts, tools and data in the project. It is intended as a background to understanding the project development.

Chapter 3 documents the pre-processing of the data carried out to create a corpus for the project.

Chapter 4 describes the setup and implementation of the accuracy tests done using the data and algorithms detailed above.

Chapter 5 shows the results of the experiments and analyses them in terms of different factors of the test: language, algorithm, etc.

Chapter 6 is dedicated to the conclusions of the project, along with proposed future work.

Appendix A contains a glossary of terms used in the document that might need a more detailed definition.

Appendix B expands on discarded corpus data, explaining the reasons for rejection.

Appendix C is a short explanation on bias in NLP, a concept that should be kept in mind in projects such as this one.

Finally, Appendix D is dedicated to explaining n-grams and k-fold cross validation, concepts not explored during the project experiments, but are briefly mentioned when comparing other studies.

Chapter 2 - State of the art

In this chapter I will look at the most important aspects of the current technological and academic status of the project topic. Firstly, there will be a short explanation of each public online dataset used for the project corpus. The next sections are introductions to several concepts used and explored during the development of the project, such as Natural Language Processing, certain Machine Learning algorithms (Naïve Bayes, Support Vector Machines and Logistic Regression), and Deep Learning. Finally, the technologies used in the project and manual annotation process will be presented.

2.1 Datasets on Hate Speech

In order to use Machine Learning models, it is essential to have classified data to feed them, so research on existing datasets on hate speech was done. This could be both general and specific hate speech, (racism, homophobia, sexism, etc).

Each language has its own subcorpus within the project corpus. For each one, the geolocation of the comment or text posted (hereinafter referred to as 'posts') was not taken into consideration. So, for example, European Spanish and South American Spanish, Brazilian Portuguese and European Portuguese, and British English and American English were analysed together.

Table 1 shows all datasets relevant to hate speech that have been considered to be well suited for the project, along with the languages that they contain, the number of total posts and the percentage of rows considered to be hate speech. For each dataset a short explanation will be provided in the next sections.

Dataset id	**Languages**	**Nº of total rows**	**Percentage of text with hate speech** ***(before pre-processing)***
EXIST*	English, Spanish	English: 5644 Spanish: 5701	English: 49.5% Spanish: 50.24%
AMI 2020*	Italian	5409	47.09%

HatEval**	English, Spanish	English: 13000 Spanish: 6600	English: 42.08% Spanish: 41.5%
Automated Hate Speech Detection	English	24783	5.77%
IHSC	Italian	6928	18.63%
Hierarchically-Labeled Portuguese Hate Speech	Portuguese	5668	19.85%
Hateful Symbols or Hateful People?	English	16909	31.63%
Are You a Racist or Am I Seeing Things?	English	6909	18.18%
HaSpeeDe 2**	Italian	6837[4]	40.46%
ToLD-BR	Portuguese	21000	1.79%
OffComBR	Portuguese	1033	19.55%
Hate speech dataset from a white supremacist forum	English	10944[5]	10.93%
The Gab Hate Corpus*	English	27546	8.52%

Table 1. List of datasets used in this project

Usually, in order to evaluate Machine Learning algorithms, a division of the gathered data is made to assign each batch a certain role. Training sets (also known as 'train sets') are used to feed the model so it can learn the patterns in the data. Test sets provide unbiased evaluations of the models that have been fitted to the training set. On occasions, there are also validation sets, which are used before the test sets

[4] This dataset contained a train, test and validation set, but since only the training set had the column names, this number refers to the number of rows from that set, since it was the only one used.

[5] This dataset came with a train and test set along with a folder containing all the data in one place, so such data was used in the project as a single set.

cooperating with the training set in order to fine-tune the models. However, the model does not learn from this data directly.

In Table 1, datasets with a single asterisk (*) in their name contained a train and a test set, while those with two (**) had train, test and validation datasets.

If the data comes undivided, an optimal split must be decided in order to prevent both overfitting and underfitting. Overfitting occurs when the model learns the training data so thoroughly, including noise and random fluctuations, that it can't accurately predict the test data. Underfitting is its opposite, and so the model has a poor performance in both training and test data. If the model is too simple or the training data size is too small, there could be underfitting, but if there is too much data or model complexity, overfitting might be an issue.

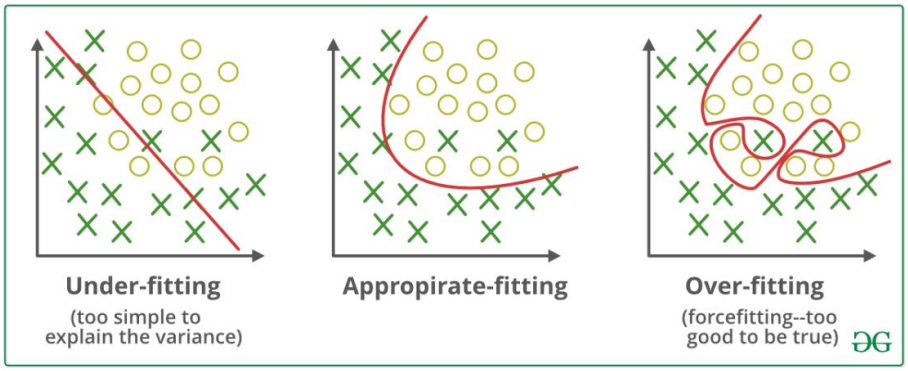

Figure 1. Visual example of underfitting and overfitting[6]

2.1.1 EXIST

IberLEF[7] is a shared evaluation campaign for NLP (Natural Language Processing) systems in Iberian languages, such as Spanish and Portuguese. Their goal is to encourage research in this field so more state-of-the-art tasks are done in these languages. Every year there is a call for different task proposals, and those interested can apply, making it an international collaboration with interesting outcomes for those taking part.

[6] Image credit: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

[7] https://sites.google.com/view/iberlef2021

In 2021, one of the shared tasks was EXIST: sEXism Identification in Social neTworks (Rodríguez-Sánchez, et al., 2021). Its objective was to be able to detect sexism in social media posts, from small acts of micro-aggression to violent misogyny. The data to be classified was a list of tweets, from the social network Twitter; and gab posts, from the far-right social network Gab. All text was obtained by collecting many sexist terms used on the Internet, and subsequently extracting tweets and gab posts that used those expressions.

Participants were asked to classify the data in accordance with two tasks:

- **Sexism identification**: A binary classification of whether a text was sexist or not. The "degree" of sexism is not important.
- **Sexism categorization**: If a text is sexist, what kind of sexism was present.

test_case	id	source	language	text	task1	task2
EXIST2021	10280	gab	es	puta madre	non-sexist	non-sexist
EXIST2021	10534	twitter	es	No puedo más con las zorras	sexist	misogyny-non-sexual-violence
EXIST2021	007019	twitter	en	At what point did I slut-shame anyone? I said that wasn't how I got into uni.	non-sexist	non-sexist

Table 2. Sample of original EXIST dataset

2.1.2 AMI

Evalita[8] is an annual NLP evaluation campaign in Italian. It has been organizing shared tasks since 2007 and is endorsed by the Italian Association for Artificial Intelligence and the Italian Association for Speech Sciences. (Fersini, Nozza, & Rosso, 2020) presented a shared task on Automatic Misogyny Identification, shortened as AMI.

[8] https://www.evalita.it/

However, this wasn't the first time it has appeared in a shared task. Both IberEval[9] and Evalita had an AMI shared task in 2018. These tasks have data in English, Spanish and Italian. Unfortunately, the only data obtained for this project was the train and test set from the 2020 shared task, which is in Italian.

The goal of this task was, not only to identify misogyny, but to recognize whether or not a piece of text was aggressive. Therefore, the data, apart from the text, had two columns:

- **Misogynous**: A binary classification of whether a text was sexist or not. The "degree" of sexism is not important.
- **Aggressiveness**: A binary classification of whether a text was aggressive or not.

2.1.3 HatEval

SemEval[10] is a collection of research workshops on NLP whose aim, much like IberLEF's, is to advance the state of the art in this field and to create datasets for many shared tasks on natural language semantics. SemEval's tasks are an annual event, and in 2019, 13 new tasks on semantic evaluation were announced, the fifth being HatEval: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. (Basile, et al., 2019)

This task's goal was to detect hate speech directed at two vulnerable social groups: immigrants and women. The data to classify in this task was in English and Spanish. Here, participants were asked to identify several aspects of the tweets in two different tasks:

- **Hate Speech Detection against Immigrants and Women**: Whether or not a tweet contained hate speech towards women or immigrants.

[9] https://sites.google.com/view/ibereval-2018

[10] https://semeval.github.io/

- **Aggressive behavior and Target Classification**: Describing whether the hate speech in the text is aggressive, and also identifying if the hate speech is directed at a specific person or to a group of individuals.

2.1.4 Automated Hate Speech Detection

(Davidson, Warmsley, Macy, & Weber, 2017) created a dataset that categorized thousands of tweets into three categories: hate speech, offensive text and neither. It is important to make this distinction, because many times a slur can be used in a way that is not hate speech. Black people use racial slurs used against them as slang between each other, and sometimes this can be offensive, but not hate speech.

Davidson's team had workers from CrowdFlower manually annotate the tweets. In the end, the dataset available to the public was formed by these columns:

- **Count:** Number of CrowdFlower users who annotated the tweet
- **Hate_speech:** Number of CrowdFlower users who considered the tweet to be hate speech
- **Offensive_language:** Number of CrowdFlower users who judged the tweet as offensive
- **Neither:** Number of CrowdFlower users who described the tweet as neither offensive nor non-offfensive
- **Class:** Class label for majority of CrowdFlower users (0: hate speech; 1: offensive text; 2: neither)
- **Tweet:** Full text of the tweet

2.1.5 IHSC

(Sanguinetti, Poletto, Bosco, Patti, & Stranisci, 2018) contributed an Italian dataset about hate speech in tweets about, mainly, immigrants. The dataset has 6 different columns:

- **Tweet_id:** The Twitter ID of the tweet
- **Hs:** Indication of whether a tweet contained hate speech or not (yes/no)

- **Aggressiveness:** Value describing if a tweet has the intention of being aggressive or harmful (no/weak/strong)
- **Offensiveness:** Indication if a tweet was hurtful (no/weak/strong)
- **Irony:** Indication if a tweet contained sarcasm, satire or irony to imply a certain message (yes/no)
- **Stereotype:** Whether a tweet implicitly or explicitly expresses beliefs society has about different groups of people (yes/no)

2.1.6 Hierarchically-Labeled Portuguese Hate Speech

(Fortuna, Silva, Soler-Company, Wanner, & Nunes, 2019) built a Portuguese dataset for hate speech detection research. They annotated tweets they obtained in two ways: binary and hierarchical. In the binary annotation annotators had to classify as hate speech (1) or not hate speech (0). The hierarchically annotated dataset had a large number of columns: each one of them being a group of people a tweet could potentially be targeted at. These classes work in a tree-like structure, where one class can have one or more child classes ("Asians" is a child class of "Racism").

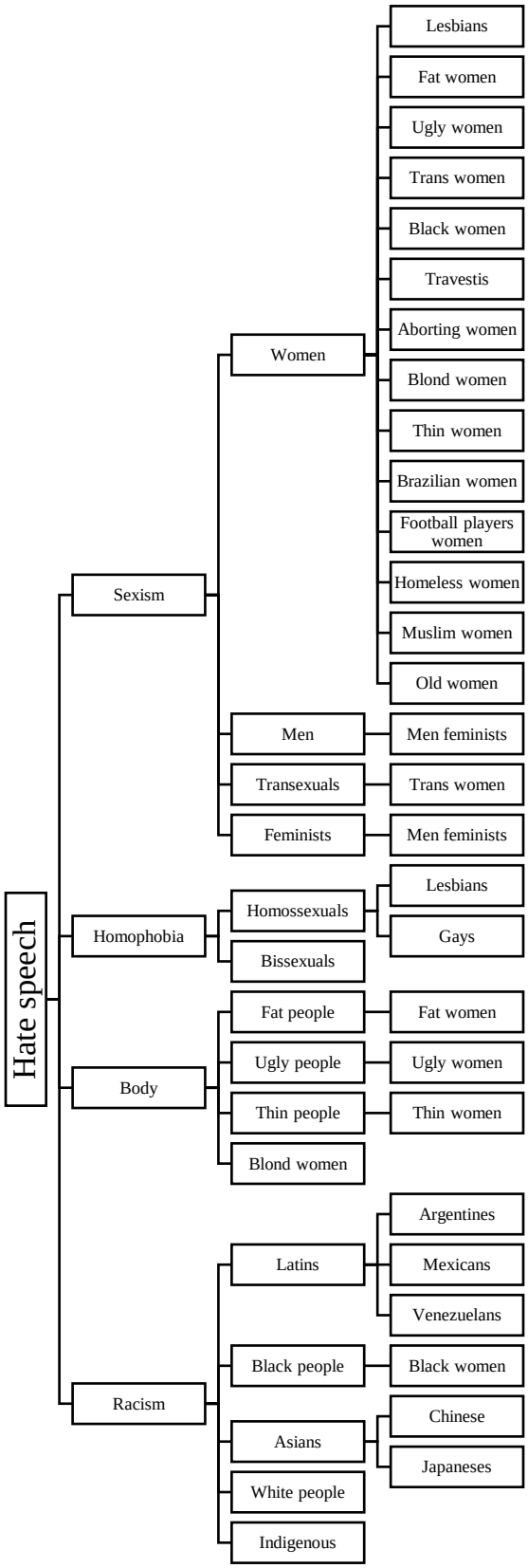

Figure 2. Data structure for Hierarchically-Labeled Portuguese Hate Speech set, part I

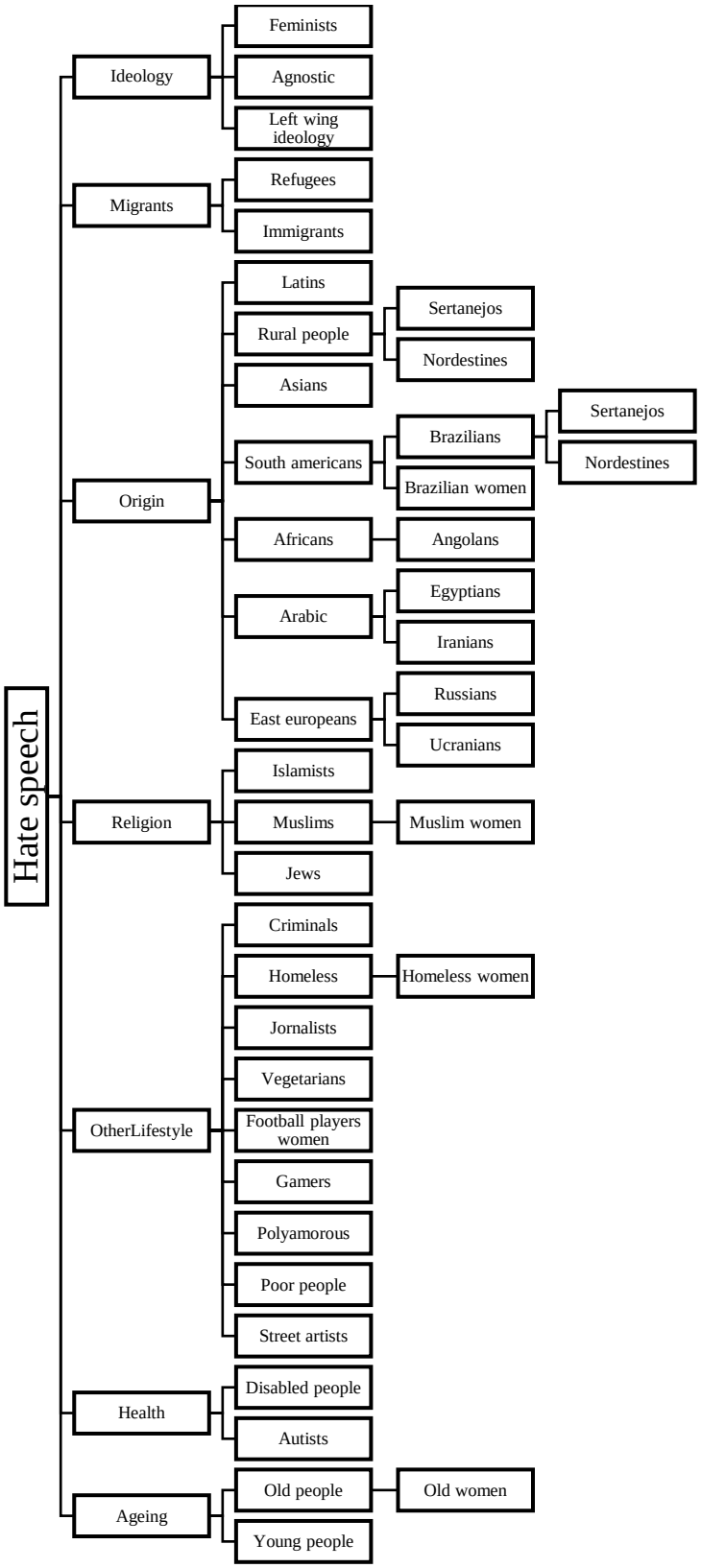

Figure 3. Data structure for Hierarchically-Labeled Portuguese Hate Speech set, part II

2.1.7 Hateful Symbols or Hateful People?

(Waseem & Hovy, 2016) created a classified dataset of tweets to detect sexist and racist hate speech online. Their criteria for classifying the tweets is founded in critical race theory (Delgado & Stefancic, 2017), a social movement from the United States which studies the intersection between race, law and power in society. The criteria was also used in this project for manual re-classification when this was necessary.

The dataset contained two columns: the first being a tweet ID, used to retrieve the corresponding tweet using Tweepy, a Python library used for accessing Twitter's API. The second column contained a string that indicated the hate speech which the tweet contained ("sexism", "racism", "none").

2.1.8 Are You a Racist or Am I Seeing Things?

(Waseem, 2016) wanted to look more into the annotation of hate speech, and how different ways of annotating data can influence how good the annotations are. The tweets were classified by amateur annotators from CrowdFlower and by expert annotators – anti-racist and feminist activists with much knowledge of the subject. In his conclusions he explains that having intimate knowledge about a subject helps a great deal when it comes to detecting hate speech related to that topic.

Waseem used some of the tweets from (Waseem & Hovy, 2016), plus many new ones. For each row, the set contained many columns: the first being the Twitter ID of the tweet. Secondly, there is a "Expert" column, indicating the classification made by the expert annotator assigned to the text. Finally, we find a varying number of "amateur" columns, which indicate the classification of the tweet by one of the amateur annotators judging the tweet. The tweets could be classified as four different categories: "racism", "sexism", "neither" and "both", indicating that a tweet contained racism and sexism.

2.1.9 HaSpeeDe 2

The first shared task on hate speech in Italian was Evalita's HaSpeeDe (Hate SPEEch DEtection) shared task in 2018. Thanks to the amount of people that

participated and the positive results, Evalita hosted a new edition of this shared task in 2020 called HaSpeeDe 2 (Sanguinetti, et al., 2020). The goal of this task was to take the state of the art on this topic to the next level by introducing novelties, such as stereotype analysis.

This second edition of the task focused on three tasks:

- **Hate speech detection:** Checking the presence of hate speech in the text
- **Stereotype detection:** Classifying whether text contains a stereotype or not. This does not detect if it's hateful, a stereotype could be talked about in a non-hateful way
- **Identification of Nominal Utterances:** Recognition of nominal utterances in hateful text

2.1.10 ToLD-Br

(Leite, Silva, Bontcheva, & Scarton, 2020) proposed a large-scale dataset for detecting toxicity in tweets in Brazilian Portuguese. The dataset was created with the help of 129 annotators who fine-tuned monolingual and multilingual BERT models.

In this set, "toxicity" isn't equivalent to hate speech. For every tweet, there are 6 columns each indicating a different aspect of toxicity: 'Homophobia', 'Obscene', 'Insult', 'Racism', 'Misogyny', and 'Xenophobia'. The categories can overlap.

Every column contains values between 0 and 3, signifying the number of annotators that considered that the tweet belonged to such category (each tweet was labelled by 3 annotators).

2.1.11 OffComBR

Since hate speech detection in Portuguese (and specifically, Brazilian Portuguese) has little research, (Pelle & Moreira, 2017) made some contributions that were helpful, by, among other results, creating a Portuguese dataset with hateful and

non-hateful comments from the Brazilian news site *g1.globo.com*[11], specifically from the politics and sports sections of the site, since news from these sections contained the majority of hateful comments.

For each comment, the data contains an id, a class or classification, and the comment's full text. There are only two categories for labelling comments, hateful or non-hateful, written as "yes" if the comment was hateful or "no" if otherwise.

Two datasets where developed. The first contained all extracted comments, and the assigned class was the one picked by at least two out of the three judges to annotate the comment. The second dataset only contained the comments where all annotators agreed on the class. This way, although the total amount of data was smaller, accuracy was higher.

2.1.12 Hate speech dataset from a white supremacist forum dataset

(Gibert, Perez, García-Pablos, & Cuadros, 2018) created a hate speech dataset based on posts on the white supremacist, neo-Nazi Internet forum, Stormfront. In this dataset, posts were classified into four different categories:

- **Hate**: Text that contains hate speech
- **No hate**: Text that doesn't contain hate speech
- **Relation**: Text that, by itself, doesn't convey hate, but combining it with other sentences in this category, does
- **Skip**: Sentences in other languages or so neutral that it doesn't enter in any of the other categories

2.1.13 The Gab Hate Corpus dataset

The data obtained from (Kennedy, et al., 2022) is a set formed by posts from Gab, and the typology that they use to categorize the posts in different column is as follows:

[11] https://g1.globo.com/

HD: Meaning "assault on human dignity"

- **CV**: Meaning "call for violence"
- **VO**: Meaning "Vulgarity/Offensive language directed at an individual"

2.2 Discarded datasets

Apart from the data detailed above, other datasets were examined for potential use in for these experiments. However, since the classification of such data was deemed incorrect, these sets were discarded.

These datasets are: Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior dataset (Founta, et al., 2018) and the 2019, 2020 and 2021 HASOC datasets (Mandl T. a., 2019) (Mandl T. a., 2020).

The rejected datasets contain a large enough portion of Internet posts that have been judged as hate speech. However, they do not actually contain it. These posts fall into a number of categories that indicate that a piece of text doesn't contain hate speech:

- **Condemning crimes:** Offensive text directed at someone who has committed a crime that only refers to the crime (i.e., no negative mention of race, gender, or any characteristic that makes the criminal a minority).
- **AAVE as hate speech:** Text that contains AAVE, specifically, racial or sexist slurs that in this form of English are reclaimed by minorities, and used in a casual or humorous way without being hateful towards anybody.
- **Political beliefs and/or job, and not social group:** Offensive text against a person or people based on their job or political beliefs.
- **Referencing hate speech without engaging in it:** Text that comments on another source's hateful speech, either by using reported speech or quote marks, without agreeing with it.
- **Normal text without hateful or offensive connotations:** Posts with no hateful intentions or references whatsoever.

- **Offensive but not hateful:** Offensive text, generally insults, towards someone or something, without any political or social implications.

Further explanation on this can be found in Appendix B.

2.3 Natural language processing

Natural Language Processing (NLP) is a subfield of computing and linguistics which intends to help computers understand human language (Wikipedia, 2022), also known as natural language. Some common applications for NLP are chatbots, virtual assistants like Siri or Alexa, and text classification, which is what this project focuses on.

For Machine Learning tasks, NLP techniques are useful when pre-processing data to make them easier to understand. Notable examples are:

- **Tokenization:** Separates text into chunks called tokens. A simple example of tokenizing would be to transform the sentence "Hello world" into two tokens: "Hello" and "world".
- **Stopword removal:** Removes stopwords from the text. Stopwords are words used so frequently that they carry very little importance and information. Examples in English are "a", "the" and "you".
- **Lemmatization & stemming:** Processes where words are cut to a base form. Lemmatization focuses on morphological analysis, so, for example, the word "studies" would be reduced to "study". Meanwhile, stemming removes common particles such as suffixes and prefixes, so the word "studies" would transform into "studi".
- **POS tagging:** Short for Parts of Speech tagging, here words are labelled as different types of words, such as nouns, verbs, adjectives… depending on their definition and context in the text.

Additionally, when working with Internet posts such as tweets, as in this project, other additional pre-processing should be applied in order to deal with emojis, hashtags, urls, and other specific elements. Some examples are url elimination, transformation of emojis into text, or removal of the hash (#) character in hashtags.

TF-IDF (Term frequency – Inverse document frequency) is a measure used on occasions in NLP Machine Learning tasks to produce a weight or importance for each term in each document (Manning, Raghavan, & Schütze, 2008). Its name can be broken down into two concepts:

- **TF:** Measurement of how frequent a word appears in text.
- **IDF:** Estimation of the importance of a word in a document. The rarer the word in the corpus, the more importance it has.

$$tf\text{–}idf(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

Equation 1. TF-IDF formula

tf-idf(t, d, D) assigns a higher value to the term t when it is used many times in a small number of documents within the corpus. If the term occurs fewer times in one document or in a large number of documents, the importance lowers.

If TF-IDF is not used, when tokenizing, all string elements (e.g. words or characters) will have the same weight, meaning every element has the same importance in the text.

2.4 Machine Learning

Machine Learning is a subfield of Artificial Intelligence which is broadly defined as the capability of a machine to imitate intelligent human behaviour (Brown, 2021). Specifically, this project uses supervised learning algorithms, a subcategory of Machine Learning that uses labelled datasets to train models in order to accurately classify unknown input data. There are two types of supervised learning approaches: classification and regression. Classification focuses on dividing the data into the number of classes that exist, while regression tries to understand how dependant and independent variables correlate. This project uses both classification and regression.

Machine Learning is used in many fields, including robotics, computer security and natural language processing (NLP). This last subfield focusses on how computers interact with human language, also called natural language. This project's goal fits into NLP, and has a similar development to that of a specific type of NLP technique: sentiment analysis, a task which finds the opinions of authors about specific entities

(Feldman, 2013). In this case, data is classified as either hate speech or non-hate speech.

Three of the most commonly used algorithms for sentiment analysis in Machine Learning are Naïve Bayes, Logistic Regression and Support Vector Machines. These algorithms either use classification or regression, and were used in the duration of this project.

2.4.1 Naïve Bayes

The Naive Bayes Classifier is an algorithm used to assign the most likely class to a given example by its feature vector (Rish & others, 2001). It assumes such features in the input data are independent from one another — a *naive* assumption, since in many cases this isn't true. Despite this, the Naïve Bayes classifier is very successful in practice. This classifier is based on Bayes' theorem, which calculates the probability of an event occurring based on previous knowledge:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Equation 2. Bayes' theorem

The adaptation for the Machine Learning algorithms states thus, (p(yi | x1, x2, x3…)) meaning the probability of a class given certain feature values.

$$\mathbf{p(yi \mid x1, x2, \ldots, xn)} = \frac{\mathbf{p(x1, x2, \ldots, xn \mid yi).p(yi)}}{\mathbf{p(x1, x2, \ldots, xn)}}$$

Equation 3. Bayes' theorem for Naïve Bayes Classifier

There are three main types of Naïve Bayes algorithm, each useful in different situations:

- **Multinomial Naïve Bayes:** Here, feature vectors use a multinomial distribution (p1, p2, …, pn) to represent the probability of certain events. Each feature vector counts the number of times each event has been observed (x1, x2, …, xn). This type of Naïve Bayes is widely used typically in text classification tasks.

- **Bernoulli Naïve Bayes:** For this variation, the features are independent binary values, 0s or 1s. This represents the presence or absence of each feature. Like Multinomial Naïve Bayes, this type of algorithm is popular for document classification.
- **Gaussian Naïve Bayes:** If there is continuous data, this Naïve Bayes assumes that those values, which are associated with each feature, follow a normal distribution, also known as a Gaussian distribution. In cases where this is not the case for the input data, it is best not to use this kind of algorithm.

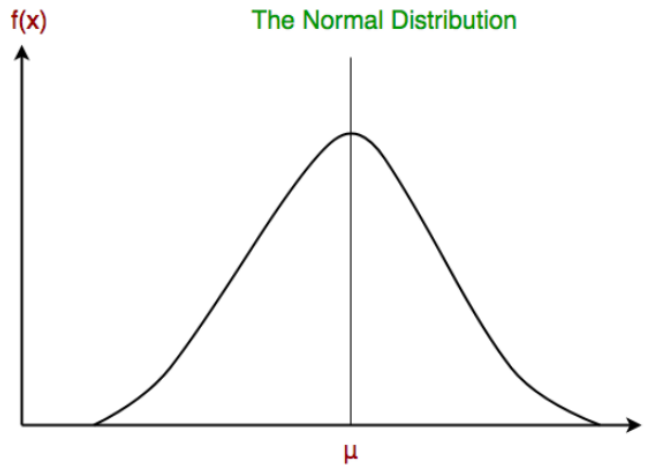

Figure 4. Visual representation of a Gaussian distribution[12]

Naïve Bayes is a popular algorithm in many fields. For example, it is a popular approach for NLP tasks such as spam detection in emails since the 1990s (Wikipedia, 2022). It has also proved to be successful for predicting medical diagnostics (Kononenko, 2001), and used for creating better approaches in bank credit scoring (Okesola, Okokpujie, Adewale, John, & Omoruyi, 2017) and loan risk assessment (Krichene, 2017).

[12] Image credit: https://www.geeksforgeeks.org/naive-bayes-classifiers/

2.4.2 Logistic Regression

Logistic regression is a regression algorithm that acts as a supervised binary classifier, meaning it can distinguish 2 categories (in this case, "hate speech" and "not hate speech")[13]. This algorithm can predict the probability of a result by analysing the relationship between one or more existing, independent variables. In this algorithm, the data is fitted to an "S" shaped line which follows a sigmoid function and goes from one category to the other.

$$sigmoid(x) = \frac{1}{1 + e^{-x}}$$

Equation 4. Sigmoid function used in Logistic Regression

Looking at the following image, the rightmost green dot is an outlier, since an instance with that measurement would have a probability higher than 50% of belonging to Category 1, and therefore would be classified as belonging to such category.

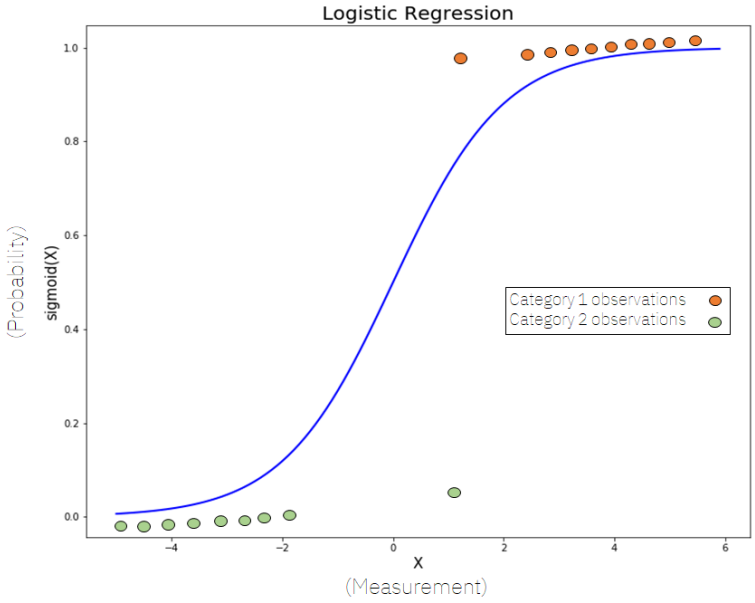

Figure 5. Visual representation of Logistic Regression algorithm[14]

[13] References: GeeksforGeeks , Towards Data Science

[14] Image credit: https://towardsdatascience.com/logistic-regression-explained-9ee73cede081

Logistic Regression is a continuously used algorithm a number of fields, especially in business and medicine. Business-wise, it can be used, for example, to predict when financial distress in a company can occur in the long run (Chen, 2011). In the medical field, Logistic Regression is used in medical research (Schober & Vetter, 2021) and prediction programs such as the Trauma Quality Improvement Program (Wikipedia, 2021), which uses this algorithm to predict mortality in injured patients.

2.4.3 Support Vector Machines

Support vector machine (Gandhi, 2018) is a classification algorithm based on the idea of creating an optimal N-dimensional hyperplane, N being the number of features, that separates them in the best way. An optimal hyperplane is one where the distances between all classes and the hyperplane (margin) are the same. A support vector, in particular, is defined by the margin, and is a specific point of the data which is close to the hyperplane, and therefore, influential as to how the hyperplane is shaped, located and oriented.

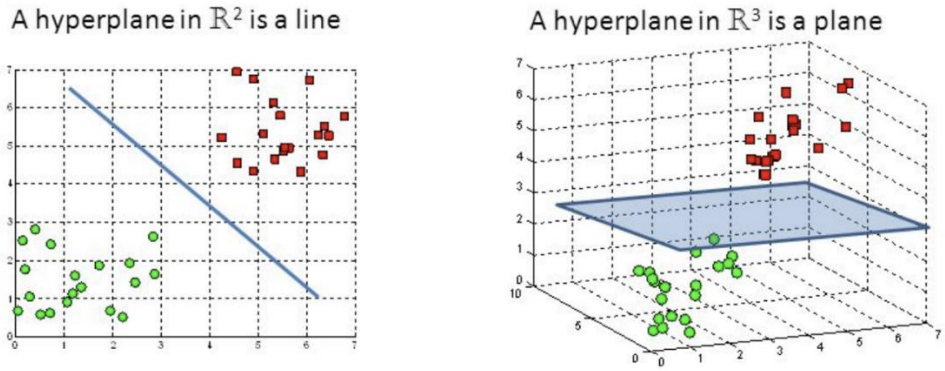

Figure 6. Visual representations of hyperplanes in SVM[15]

SVM is defined by its support vectors only, meaning that data samples outside of the margin are ignored. The larger the margin, the more observations are taken into consideration. Usually, a large margin is considered a good margin, but if too small, this could lead to underfitting.

[15] Image credit: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

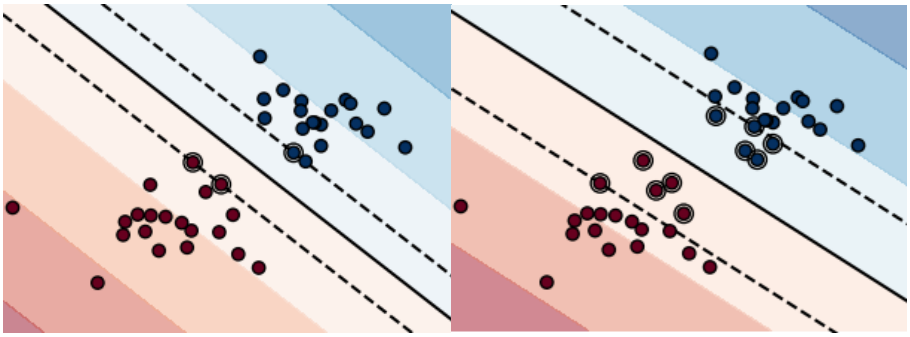

Figure 7. SVM with small margin (left) and large margin (right)[16]

Usually, data is placed in ways that the hyperplane is automatically linear. However, this is not always the case, since data can scatter around in many forms. This is where the kernel trick comes in. Mathematical functions are used to transform the input in ways that make the classification problem a linear one. There are many types of kernel functions. Two of the most notable ones are:

- **Linear:** Useful for when the number of features is large. The function is:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j.$$

Equation 5. Linear kernel function

- **RBF (radial basis function):** Stems from a linear kernel function, but is able to handle cases where the relation between labels and attributes is nonlinear by changing the data samples to form a higher-dimensional space and using a linear hyperplane. This kernel functions goes as so:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2), \ \gamma > 0.$$

Equation 6. RBF kernel function

Image and text classification stand out as one of SVM's main applications. This algorithm has been used, for instance, to detect plant diseases in images (Khan, 2020) (Tian, Zhao, Lu, & Guo, 2011). It has also proved to work successfully in classifying webpages based on their text and context features like HTML tags and hyperlinks (Sun, Lim, & Ng, 2002).

[16] Image credit: https://scikit-learn.org/stable/auto_examples/svm/plot_svm_margin.html

2.5 Deep Learning

Machine Learning algorithms are not the only way to solve NLP tasks. Deep learning tries to solve problems by imitating how a human brain would work, through a combination of inputs, weights and bias. In order to do so, neural networks are used (IBM Cloud Education, 2020).

Deep learning surprises the world constantly in new ways: One example is the text-to-image model Dall-E[17], which uses GPT-3, a deep learning language model released in 2020, to generate an image with a text prompt. It is very tempting to try out deep learning models for NLP projects, however, there is a reason this idea was discarded. Deep learning requires massive amounts of data to function properly, normally millions of labelled data instances; and this project's corpus does not satisfy this requirement. Also, deep learning is much slower and computationally expensive, and the time spent on this project would not have been enough for an equal project based on Deep Learning.

Nevertheless, this approach is being looked into. For example, in the final conclusions from (Plaza-Del-Arco, Molina-González, Ureña-López, & Martín-Valdivia, 2020), an article which uses data from the previously mentioned HatEval task (see section 2.1.3), a deep learning approach was done. Also, future interest in deep learning models such as ELMO (Peters, et al., 2018) or BERT (Muller, 2022) is shown.

2.6 Hate speech detection

One of the main tasks in NLP is text classification, which consists of classifying text into different groups. Text classification includes many popular applications, such as detecting spam in emails, identifying what language a text is in, labelling documents according to its topic, or sentiment analysis of pieces of text such as online movie reviews. This project's task is hate speech detection in social media posts, and is differentiated from other kinds of text classification, firstly, because there are only two categories the text can be labelled as: "hate speech" and "non-hate speech". These

[17] https://openai.com/blog/dall-e/

labels are quite different from other types of text classification, since they denominate whether data is hateful towards a marginalized target. In addition, the length of the data samples is much shorter than for other tasks, such as spam detection in emails.

Automated hate speech detection, especially online, has been gaining popularity in recent years. A 2019 survey on Automatic misogyny detection (Shushkevich & Cardiff, 2019) explains two main approaches for it:

- Using classical Machine Learning algorithms such as Support Vector Machines, Naïve Bayes and Logistic Regression
- Using neural networks. Some examples cited in the article are (Zhang & Luo, 2018), (Park & Fung, 2017) and (Badjatiya, Gupta, Gupta, & Varma, 2017)'s use of Convolutional Neural Networks (CNNs) and (Goenaga, et al., 2018) and (Badjatiya, Gupta, Gupta, & Varma, 2017)'s approaches with Recurrent Neural Networks (RNNs), specifically with Long Short-Term Memory networks (LSTMs).

This project's approach is similar to the first one. Many studies before have explored this topic using such algorithms. Some notable examples are described below.

Automatic Hate Speech Detection using Machine Learning: A Comparative Study

(Abro, et al., 2020) found that existing studies on classification of hate speech using Machine Learning lacked comparative analysis when it came to different approaches, and so, this study uses, among others, eight different Machine Learning algorithms, including Naïve Bayes, Support Vector Machines and Logistic Regression. Publicly available datasets were collected for the corpus. The data was compiled and classified by CrowdFlower into three classes, "hate speech", "offensive but not hate speech" and "not offensive". 16% of the text corresponded to the "hate speech" class. The pre-processing of the data consisted of several techniques: converting text to lowercase, stemming, tokenization and removal of all bad symbols, stopwords, punctuation marks, URLs, usernames, hashtags and white spaces. The chosen train/test split was 80/20. In addition to the eight different ML algorithms, three types of features were explored: bigrams with TF-IDF, Word2vec and Doc2vec.

The final results showed that bigrams with TF-IDF obtained the best accuracy. Regarding which algorithm is best, SVM proved to have the best performance out of them all, obtaining a 79% accuracy. Meanwhile, Logistic Regression's results were a 75% accuracy, and Naïve Bayes a 73% accuracy.

Detecting Hate Speech in Social Media

(Malmasi & Zampieri, 2017) presents a study where the main aim was establishing a lexical baseline to differentiate profanity from hate speech. As well as that, they examine methods to detect hate speech in social media, as another objective is to distinguish hate speech from simple bad language. The dataset used in their experiments is the same dataset mentioned in section 2.1.4, which they refer to as the Hate Speech Dataset (Davidson, Warmsley, Macy, & Weber, 2017). The tweets were classified into three categories: "hate", "offensive" and "ok", meaning no offensive content. For the pre-processing, the text was transformed into lowercase and all URLs and emojis were removed. Instead of trying out different classifiers, like the current project does, they applied SVM classifiers with the LIBLINEAR package implementation and explored two different types of features: surface n-grams and word skip-grams. Instead of using a train/test split, they evaluated their method with a 10-fold cross validation. Finally, the best obtained result was a 78% accuracy for a character 4-gram model.[18]

Detecting Hate Speech on the World Wide Web

(Warner & Hirschberg, 2012) show their approach to detecting hate speech online, describe pilot classification experiments done in the study, and present their own definition of hate speech. They partnered up with Yahoo! and the American Jewish Congress for resources, and received data from online news groups and webpages which contained hate speech. The data differs from the current project's corpus because the length of Warner's data was much bigger, and because the present study's classification experiments were made at paragraph level. As pre-processing,

[18] The concepts of n-grams (including bigrams) and k-fold cross validation are explained in Appendix D

features were generated from the corpus using the template-based strategy presented in (Yarowsky, 1994), and POS tagging was applied to each sentence. Because hate speech takes different forms depending on who the target is, they considered that creating a language model for each stereotype was important, and so they began in this study by making one for antisemitism. There was interest in identifying correlation between antisemitism and other forms of hate speech, so the amount of possible classes text could be classified as was high. Some of these classes include "anti-semitic", "anti-woman" or "anti-black". Generally, the study focuses more on the words inside the data than the current project. Classification was done both manually by humans and automatically, using a SVM classifier with 10-fold cross validation with varying feature sets, in order to do a comparison between their results. While testing different SVM variations, they noticed that adjusting the parameter C had no effect. Finally, the best obtained results for human annotation were a 96% accuracy and a 0.63 F1-score, while the best performance when using a SVM classifier resulted in a 94% accuracy and a 0.63 F1-score.

As well as existing studies, all data obtained for this project was collected for hate speech detection, either general hate speech or specific hate speech, such as racism or sexism.

2.7 Manual annotation on data

Finding a robust definition of hate speech is a complicated task; there is no formal definition for it in International Human Rights Law, and every country has different laws when it comes to hate speech.

However, as mentioned before, in this project the criteria used to determine the presence of hate speech in tweets in (Waseem & Hovy, 2016) was followed:

"A tweet is offensive if it:

- Uses a sexist or racial slur.
- Attacks a minority.
- Seeks to silence a minority.
- Criticizes a minority (without a well founded argument).

- Promotes, but does not directly use, hate speech or violent crime.
- Criticizes a minority and uses a straw man argument.
- Blatantly misrepresents truth or seeks to distort views on a minority with unfounded claims.
- Shows support of problematic hash tags. E.g. "#BanIslam", "#whoriental", "#whitegenocide"
- Negatively stereotypes a minority.
- Defends xenophobia or sexism.
- Contains a screen name that is offensive, as per the previous criteria, the tweet is ambiguous (at best), and the tweet is on a topic that satisfies any of the above criteria"

The term "minority" used here needs expanding on. Often hate speech is intertwined with social prejudice; and therefore, a comment making fun of a white person's race isn't the same as one vilifying dark skin, since white people haven't been consistently oppressed. The case is the same with any other social division (religion, gender identity, etc.)

Taking all of this into consideration, I disregarded hateful comments targeted towards privileged groups classified as hate speech. These comments are definitely offensive, hurtful and unnecessary, but are not supported by the oppressive backbone of today's society.

This criteria was also used when manual annotation is required during this project, as well as deciding when certain data wasn't useful.

2.8 Used tools

During the project an array of tools and technologies were used for the processes done explained in the next chapters. The most notable are:

- **Python:** All the code in this project was implemented using this programming language. It was chosen because it is the ideal coding language for NLP and Machine Learning tasks.

- **Google colab:** A Python environment similar to Jupyter notebooks which is stored in Google drive.
- **Pandas:** Python library useful for data manipulation. Pandas was mainly used for the pre-processing of the data explained in chapter 3.
- **Sklearn:** Short for Scikit-learn (Pedregosa, 2011), it is a Python library which includes many implementations for different types of Machine Learning algorithms. It was used for the experiments described in chapter 4.

2.9 Conclusions

To recapitulate, for this project many datasets in four different languages (Spanish, Italian, Portuguese, English) have been compiled to form a corpus. Having the data, traditional Machine Learning approaches were chosen instead of Deep Leaning ones, because of the computational cost and amount of additional data it would require. The selected algorithms will be Naïve Bayes, Support Vector Machines and Logistic Regression, and they will be used in a task which classifies the data as "hate speech" or "not hate speech". In some cases, a manual re-classification will be needed, and this will be done following a certain established criterion.

This task will be developed in Python inside the Google Colab environment, using the Pandas and Sklearn libraries. Specifically, chapter 3 centres around the pre-processing of the picked data, and this was done using Pandas. Afterwards, in chapter 4, Sklearn was used extensively for the accuracy tests on the chosen models that will take place.

Chapter 3 - Pre-processing of the data

In this chapter we will go through each dataset used in this project; describing the processing undertaken with the objective of unifying and cleaning the sets for the experiments, in such a way that all datasets are reduced to two columns: the text of the post and whether it contains hate speech (1) or not (0).

It is important to note that for each dataset with train and test sets, both sets (and, if present, validation set) were combined after processing in order to have a mixed corpus of all corpora, which would later be split for the classification.

After the described pre-processing, the final data's percentage of hate speech for each language was as presented:

Language	**Nº of total rows**	**Percentage of text with hate speech (*after pre-processing)***	**Nº of rows containing hate speech**
English	94180	17.43%	16415
Spanish	12301	45.55%	5603
Portuguese	27701	6.15%	1703
Italian	17451	35.17%	6137

Table 3. Data statistics after pre-processing

3.1 Datasets with basic pre-processing

EXIST dataset

Since this project uses more than one dataset to form the corpus, only the EXIST sets were classified by type of sexism, so the information on the task on sexism categorization was not used and the focus was on the first, binary-classifying task on sexism identification.

AMI dataset

Only the information from the column on misogyny was kept, so the aggressiveness column was ignored.

HatEval dataset

To unify the datasets of train, test and validation, I only used the values in the column describing the values for the task on hate speech detection against immigrants and women.

Automated Hate Speech Detection dataset

The "class" column, which determines whether a tweet is hateful or not, was divided by different labels: 0: hate speech; 1: offensive text; 2: neither.

This is the only column that was necessary to keep for the experiments, and since it wasn't a binary value, I changed the values of the columns as so:

- $0 \rightarrow 1$
- $1 \rightarrow 0$
- $2 \rightarrow 0$

IHSC dataset

Since the text of the tweets aren't available in this dataset, it was necessary to use Twitter's API with Tweepy to extract it. All rows where a tweet couldn't be extracted (the account of the user was suspended, the tweet was deleted, etc.) were eliminated.

The only column that was made use of was the column that indicated if a tweet contained hate speech or not (called "hs").

Hateful Symbols or Hateful People? dataset

Tweepy was used just like in the IHSC dataset to return the tweet text. All rows where the text couldn't be extracted were deleted, and finally, a manual checking was carrried out on the classification of all tweets that overlapped with the dataset presented in the next point.

Are You a Racist or Am I Seeing Things? Dataset

All overlapping tweets between this dataset and the one presented in the previous point were deleted, and the classification of the relevant dataset was double checked.

In order to create a single binary classification column, I considered the tag "none" to be a 0, and in the rest of the cases I would tag the tweet as 1. On the other

hand, to unify all different columns that annotated a tweet, I considered the tweet to contain hate speech if there were more 1s than 0s, and vice versa (e.g.: If the expert annotator plus two amateur annotators classified the tweet as 1, and only one amateur annotator classified it as 0, a new classification column used for this project would contain a 1). Finally, since there was a discrepancy in tweets where the majority wasn't the same as the Expert classification, I manually classified those myself.

HaSpeeDe 2 dataset

Only the classification of the task on hate speech detection would be used from this dataset.

ToLD-Br dataset

Here, for some rows the different categories of the dataset can overlap (a tweet can be obscene and also racist), but in the cases where they don't, those tweets that are exclusively obscene and/or insulting have also been classified as not hate speech. Such tweets are definitely toxic, but this was done because the target isn't judged according to social category.

Recalling this set's structure, every column contains values between 0 and 3, signifying the number of annotators that considered that the tweet belonged to such category. Since this project uses binary values, if a row had the value 1 in a column, that value would be replaced by a 0. Only those tweets with where the majority of annotators have judged them to be hateful would be categorized as hate speech.

OffComBR

Here, as recalled, two datasets were created. The dataset used for this project was the one which only contained the comments where all annotators agreed on the class. This way, there was less data, but more accuracy. All that was left to do was to change the classes names from "yes" and "no" to 1 and 0.

The Gab Hate Corpus dataset

I discarded the columns "CV" (call for violence) and "VO" (violence/offensive), since text that endorses violence and text that is vulgar aren't necessarily hate speech.

For example, one could call for violence against a politician for purely political reasons, and not because of their race, gender, religion, body, etc.

The "HD" column is all the information I needed from this dataset, since this column checks if the text contains superiority over a certain social group by using slurs, stereotypes or references.

3.2 Hierarchically-Labeled Portuguese Hate Speech dataset

Even though the binary annotation of this set could be used as is, it was decided that a more complex analysis of the different classes present in the hierarchal annotation was a better option, because not everything categorized as hate speech in the binary annotation was finally considered so.

Here are the classes that didn't count as hate speech for this project, and so, would be deleted from the dataset (I also deleted the global existing parent class "hate speech", since it wasn't necessary):

- Body
- Ideology
- Agnostic
- Criminals
- Journalists
- Left wing ideology
- Men Feminists
- Old people
- Polyamorous
- Russians
- Street artist
- Ukrainians
- Vegetarians
- White people
- Young people
- Men
- East Europeans

- Thin people
- Ageing

In the cases where I was on the fence (polyamorous, old people, east Europeans), these columns were deleted because text targeted towards them would be an extremely small percentage of the data.

But knowing how many of these classes have parent classes, that would mean that if we just deleted these columns, their remaining parent classes would still have a positive value indicating hate speech stemming from their deleted child. So, before deleting these columns, their parents' (plus their own) column values were set to 0.

All these columns were then unified as a new, single "hate speech" column, that would have a 1 if at least one of the remaining columns had the value of 1, and a 0 in the other case.

3.3 Hate speech dataset from a white supremacist forum dataset

To only have two values, hate speech (1) and non-hate speech (0), all rows classified as "skip" were deleted; and as for the "relation" posts, I wanted to manually classify them myself, because I had the suspicion that some of them could potentially contain hate speech in their own right, in a more subtle way.

And this was the case. In some of the posts categorized as "relation", the users affirm false, hateful stereotypes or refer to a group of people they despise in a pejorative way, even though they are not insults or slurs.

For example, the post "The same way Jews run the government ." is a sentence that defends an antisemitic stereotype of Jews being powerful overlords.

In another case, in the phrase "Maaaaany pinders and Asians here .....", the user uses "pinder" as an ethnic slur against East Asians. But use of slurs alone is not enough to categorize text as hateful. As (Bianchi, 2014) explains, many marginalized communities have reclaimed slurs directed at them and use them in a friendly context between them. The most known example is the use of nigg*r within the black community. However, it is important to know what slurs have been reclaimed, because

if it isn't the case, it would certainly mean that the person is simply using it in a negative way (which is the case in the example, since this is taken from a neo-Nazi forum).

A final example we'll show is the post "(Includes : one pair of baggy pants , one pistol , a set of golden grills , a looting guide titled ' But I dendu nuffin ' , and one race card)". There's no explicit reference to a group to stereotype, nor are there any slurs present; but we do find the term "dindu nuffin", which is an anti-black expression that originated in 2014. This is quite recent, and I point this example out to explain that consciousness about rapidly changing hateful speech on the Internet is essential in these tasks.

Taking all of this into consideration, I manually classified the "relation" posts into hate speech or not hate speech.

3.4 General ML pre-processing

After preparing each dataset individually, some NLP pre-processing techniques were studied and applied to clean up the text.

URLS, mentions and Retweet indicators

URLs and mentions to other users (words beginning with @) were removed in order to not incorrectly teach the model that mentioning a certain user could determine if the text was hateful or not.

Also, if a tweet began with the letters "RT" (meaning that the tweet was a Retweet), they would be deleted.

Stopwords

For each language in this project, a JSON file[19] containing stopwords in that language was called for such parameter when vectorizing.

Sklearn also has built-in stopword lists for different languages, but since their own page[20] recommends alternatives to using this because of issues in English, this option wasn't used for any language in order to make the code more generic.

[19] Obtained from https://github.com/6/stopwords-json

Hashtags and emojis

For emojis and hashtags, two different datasets for each language were created, depending on how they were handled. The first type of dataset would have all emojis and hashtags removed. The second type would maintain them all, leaving the hashtags as they are and replacing each emoji for their "demojized" version using the emoji python library. So, for example, the thumbs up emoji (👍) would be replaced by ":thumbs_up:". The colons on the sides were not eliminated since they serve as good indicators for emojis.

For context as to why this decision was made, I'll briefly explain the importance of hashtags and emojis in Internet culture below.

Hashtags

Unlike with URLs and mentions, hashtags may show the sentiment of a tweet. For example, in the context of feminist and anti-feminist online content, if a tweet contains any of these hashtags, it's very possible the tweet is defending women's rights:

- #yositecreo: ("I do believe you", a Spanish expression used to support female victims of sexual assault that weren't believed by peers or the justice system)
- #metoo: (An online movement where women publicized their experiences of sexual abuse)
- #niunamenos ("Not one less", a Spanish expression that demands that no more women be killed by men)

On the other hand, there are also some hashtags that are usually attributed to sexism online:

- #notallmen (An expression originated among Men's Rights Activists responding to feminists movements, commonly used to dismiss feminist talking points)
- #redpill (A term deriving from The Wachowski sisters' *The Matrix* and coined by incels describing the process in which men "realize" that they do not hold

[20] Link to Sklearn's page on CountVectorizer

systemic power, rather, that women are the true social, economic and sexual oppressors)

- #feminazi (A pejorative term for feminists popularized by conservatives)

Emojis

Emojis, much like hashtags, can also influence the sentiment of a tweet, but unlike hashtags, they can't indicate bigoted tones. But some of them can specify certain moods:

- 😭 😢: Sad emojis can express negative feelings, but could also be used sarcastically
- 🤣 😂: Laughter emojis can indicate positivity and are also used sarcastically
- 👏: Applause shows support or agreement

Futhermore, some emojis that originally weren't made for a certain emotion or implied meaning have had their implied meaning changed over time:

- 💀: The skull emoji has been recently used to express laughter, as to indicate something is so funny, the person "died"
- 🍆: Some fruit emojis have been claimed by the Internet as a way to indicate a sexual tone

Because of all of this, emojis and hashtags have a big role in explaining the true meaning of online posts, as they may be indicators of whether somebody is seriously being hateful, or using such language in a sarcastic way, by actually critiquing hate speech. So, comparing results of removing them from the data may show how important they really are.

Chapter 4 - Carrying out the experiments

In order to find out the best way to use Machine Learning to detect hate speech, accuracy experiments on different model variants were done. This chapter contains the explanation of the experiments, plus the variants chosen for them.

4.1 Setup

1344 tests were run in total, each one having a distinct language, model, vectorizer, etc. This section is dedicated to displaying all elements of these permutations. As explained at the beginning of the project, data in four different languages was fed for the training and testing: **English, Spanish, Italian and Portuguese.** The details on the used data have been described in previous chapters.

The final percentage of hate speech presented in the data is, at most, almost 50% (in Spanish dataset), all the way down to 6% (in Portuguese dataset). Therefore, the corpus is unbalanced, and can produce too many cases of false negatives. So, during the experiments there will be tests with both **Balanced data and Unbalanced data.** For the balanced permutations, non-hateful rows are deleted so that the amount of hate speech is 50/50. Mostly, the unbalanced data results will be discarded when discussing what model and variants are optimal, because performing only accuracy tests without F1-score tests is not a problem if the classes are balanced.

From the three mentioned types of Naïve Bayes[21] in chapter 2, this project uses two of them: **Multinomial Naïve Bayes and Bernoulli Naïve Bayes.** Gaussian Naïve Bayes was discarded, because such a model requires continuous values. Also, this model does not accept sparse matrixes, and since this project's data only contains discrete variables represented in a sparse matrix (which will be explained later), it is futile to use Gaussian Naïve Bayes.

[21] Every model mentioned was tested with Sklearn's ready-made models and functions to use Machine Learning algorithms.

Out of all the tests, the SVM's execution time was the longest, especially when using English data, which was considerably larger than the rest of the languages. When using support vector machines, Sklearn's Support Vector Classifier was used. This implementation is based on an open-source SVM library called LIBSVM. For the variants, two types of kernels and three different values for the regularization parameter (also called C) were used. The higher the value of C, the less regularization and the smaller margin there exists for the hyperplane, and therefore, the more it avoids misclassifying the training data. The chosen kernels for the project are the ones previously explained in chapter 2: **Linear kernel and RBF kernel.** Knowing that a linear kernel is less suitable for datasets with less features, less accuracy is expected as compared to tests done with a RBF kernel. Finally, in regards to SVMs, three different values of C have been chosen: **Large value (10), Standard value (1), Low value (0.1)**.

Much like SVMs, for Logistic regression two factors were chosen for variants: the solver and the regularization parameter, C (which follows the same logic as SVM). The solver is a parameter present in Sklearn's implementation of LR, which is an algorithm that is used in the optimization problem. The two solvers chosen are the **Lbfgs solver** and the **Liblinear solver,** both deemed suitable for smaller datasets**.** Currently, the default solver for Sklearn's logistic regression implementation is lbfgs. However, this wasn't always the case: before version 0.22, the default solver was liblinear; hence the choice to use these two solvers. Meanwhile, the chosen values for the regularization parameter C are the same as in the SVM tests: **Large value (10), Standard value (1) and Low value (0.1)**

When splitting the data for training and testing, usually 80/20 is a good ratio. This is explained by the Pareto principle[22]. But generally, if there is more training data than test data in NLP, results will be satisfactory. So, in this project tests have been done with three kinds of train/test split: **60% train, 40% test; 70% train, 30% test; and 80% train, 20% test**

[22] This principle states that "for many outcomes, roughly 80% of consequences come from 20% of causes"

In order to properly prepare the data for the model, a vectorizer is needed. This project ran tests with two of Sklearn's vectorizers: **CountVectorizer and TfidfVectorizer.** Sklearn's Countvectorizer transforms text into a matrix of token counts and produces a sparse matrix. TfidfVectorizer does the same thing, but afterwards transforms the matrix into a TF-IDF representation.

Finally, as explained before, two different datasets for each language were created, in regards to how emojis and hashtags are treated: one with **data where all these elements have been eliminated** and another **where they have been maintained**. This can give insight into the importance of these components.

4.2 Execution

This section's purpose is to showcase the experiments in more detail by going through the process in the code. For each test, the program was run 30 times to obtain a list of different accuracies. The average and standard deviation of those results were portrayed in an Excel file[23]. For each variation, the process for obtaining the accuracy was as follows:

Firstly, the corresponding data according to language and emoji and hashtag treatment was loaded using Pandas. Before starting, a final cleansing was done, where all rows that weren't either classified as "0" or "1" were eliminated. Secondly, the data was balanced if it was required, by eliminating a random selection of rows of the majority category (which was always "not hate speech"/0) so that the percentage of hateful text was about 50%. Afterwards, the data was split into train and test data using Sklearn's train/test split. Later, the corresponding stopwords were loaded, depending on the language. Having the stopwords, the according vectorizer was used in order to tokenize the text. Next, the corresponding Sklearn model function was called in order to create and fit it with the input data. When this was done, the model was used to predict the classification of the test data. Those predictions were compared to the

[23] https://github.com/NILGroup/TFG-2122-HateSpeechDetection/blob/main/data/accuracy_data/Accuracy_data_complete.csv

actual labels of the data to obtain the accuracy. Having this, all that is left to do is to print the accuracy scores into the corresponding text file.

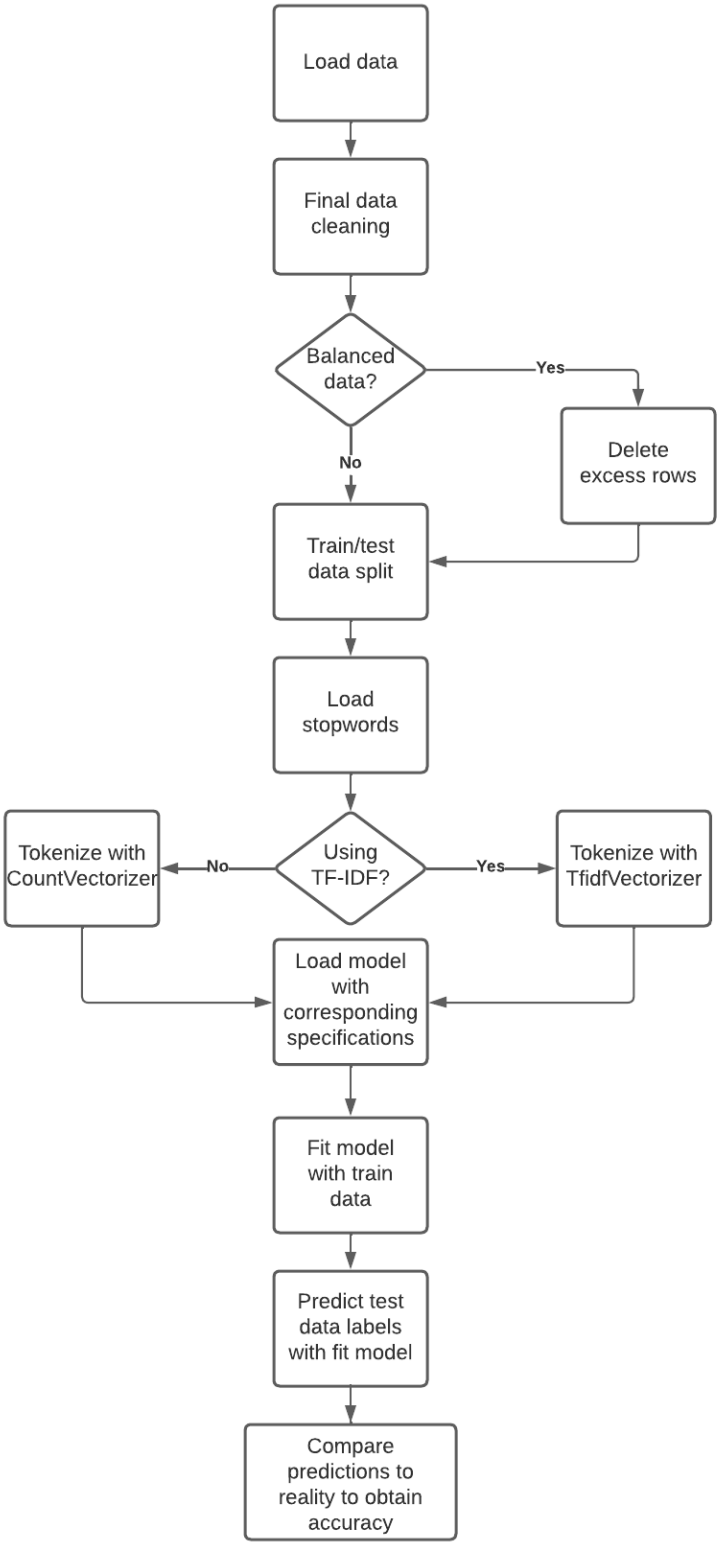

Figure 8. Accuracy experiment code flowchart

Chapter 5 - Results

In this chapter, the results of the accuracy tests will be compared in terms of language, model, TF-IDF use, and other aspects. The accuracy scores shown will be a good approximation of the averages, while confusion matrixes used to visualize the results and other experiment outputs, such as F1-score, will be from one instance of executing the code.

5.1 Language and data balancing

This section is dedicated to contrasting the different experiment performances as regards to language and data balancing. These comparisons were done together because large portions of the corpus were discarded solely because of their language or data balancing, and future sections do not consider such data.

When looking at the top accuracy scores, Portuguese, unbalanced data results were in the top 168 rows, going from 95.36% to 93.41%. However, this is because out of all the Portuguese data, only around 6% of it contains hate speech. This causes the algorithm to assign the majority of rows the label of "non-hate speech". It correctly classifies non hateful speech but rarely labels any text as hateful, let alone classify it well. Meanwhile, if Portuguese data is balanced, there is a lower accuracy, but there is a much higher number of true positives. Still, balancing Portuguese data means having less than 4000 posts as input data.

This phenomenon where there's a better accuracy score with unbalanced data at the cost of worse precision and recall for the "hate speech" class occurs for every language, but especially in Portuguese and English, since they are the datasets with the lowest percentage of hate speech. Figures 9 to 12 and Table 4 show how balancing data affects performance in all four languages by comparing results from two identical experiments except for data balancing.

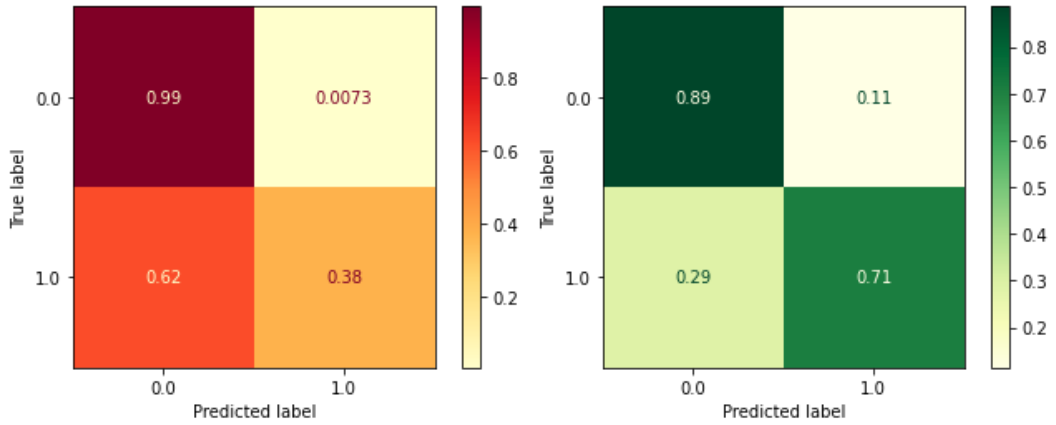

Figure 9. Comparison between two Portuguese experiments with unbalanced (left) and balanced data (right)

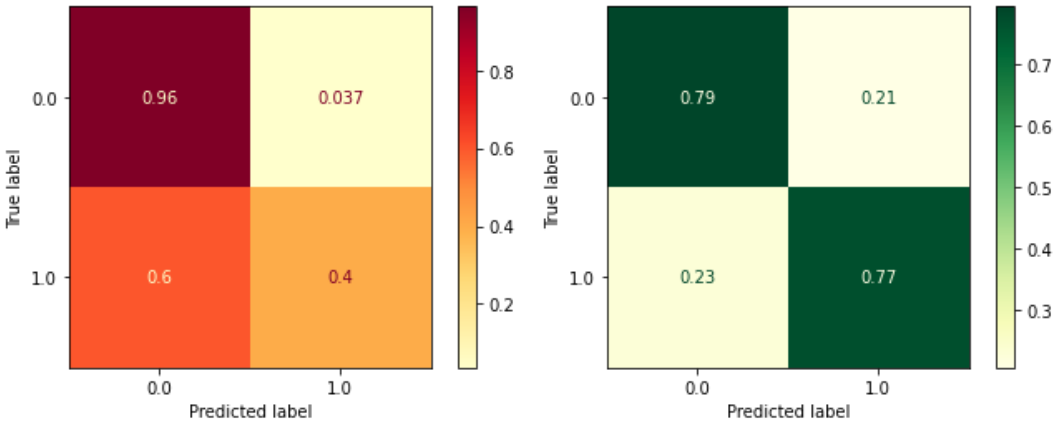

Figure 10. Comparison between two English experiments with unbalanced (left) and balanced data (right)

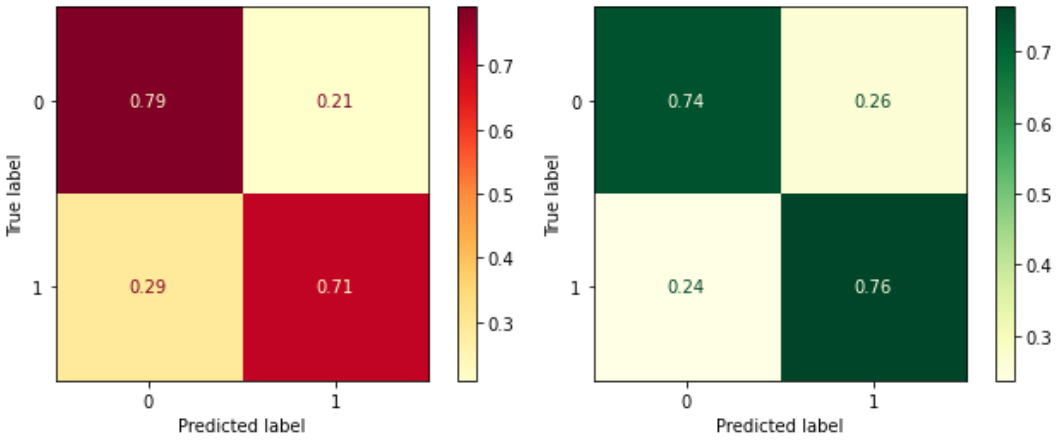

Figure 11. Comparison between two Spanish experiments with unbalanced (left) and balanced data (right)

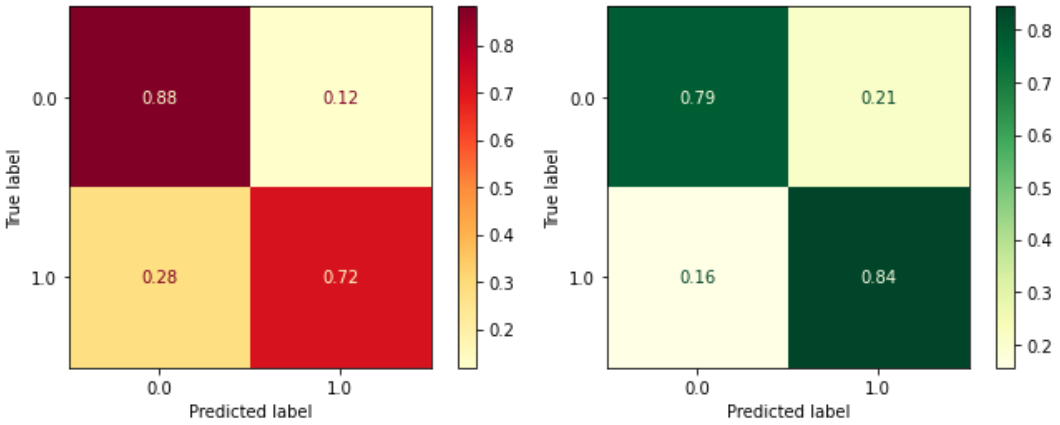

Figure 12. Comparison between two Italian experiments with unbalanced (left) and balanced data (right)

Language	Balanced data	Other specifications	Accuracy
Portuguese	No	All emojis and hashtags, 80/20 train test split, SVM RBF kernel, large C No TF-IDF[24]	**95.36%**
Portuguese	Yes		81.51%
Spanish	No		74.74%
Spanish	Yes		74.13%
Italian	No		82.77%
Italian	Yes		81.53%
English	No		86.33%
English	Yes		77.95%

Table 4. Accuracy scores, comparison of language and data balance

One last interesting aspect to consider in regard to language and data balance is the standard deviation of accuracy results. The highest figures (Fig. 15) are all associated to balanced datasets (Fig. 16), while all of the lowest results (Fig. 13) for standard deviation are all related to unbalanced datasets (Fig. 14).

As regards language, the lowest standard deviation of accuracy score is found for, firstly, English data, and secondly, for Portuguese data. On the other hand, the

[24] These specifications were chosen because the highest accuracy was obtained using them with unbalanced Portuguese data

highest numbers are found among, most importantly, Portuguese data, followed by Spanish data, and in a very low percentage, Italian and English data. The majority of the most extreme cases occur using Portuguese data, where two different executions of the same experiment could return from a 53% accuracy to a 79%. This could be because each dataset is formed by several other datasets. When balancing data, a big part of some of the datasets could be cut from the training data, but not from the test set, therefore having a low accuracy, because the algorithm would not have learnt how to classify all possible data that could appear when testing. In other cases, maybe only data from mostly one dataset would be used for both training and testing, resulting in a higher accuracy. Since cut data and the split between train and test is chosen randomly, accuracy scores have a large range. In order to make this mistake again, future work on this project would include usage of k-fold cross validation instead of a standard train/test split.

But, for unbalanced data, this does not happen, so accuracy stays consistent. Also, the Portuguese and English datasets are the ones with most rows when unbalanced, so this also helps to have a low standard deviation, since the algorithm has more input to learn from.

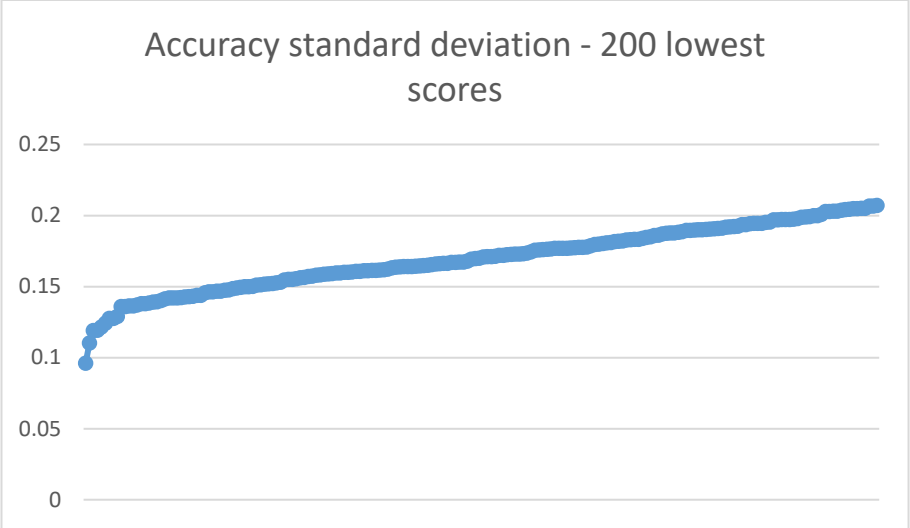

Figure 13. General accuracy scores' standard deviation, lowest values[25]

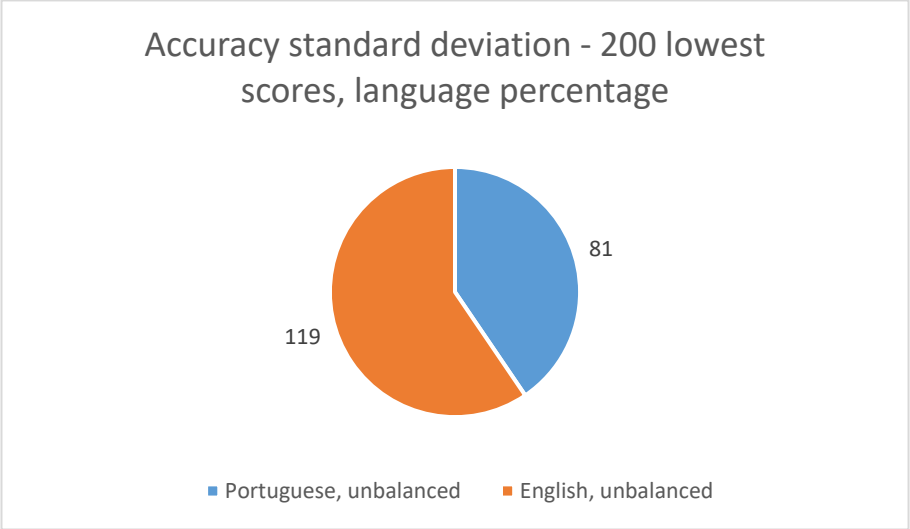

Figure 14. General accuracy scores' standard deviation, lowest values language percentage

[25] In all line graphs, the horizontal axis represents different experiments that fit into the specific features defined by the graph title

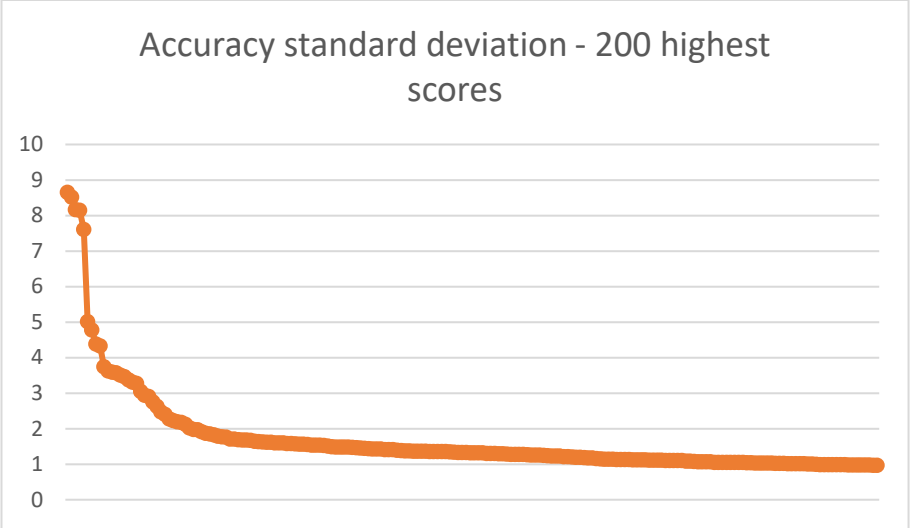

Figure 15. General accuracy scores' standard deviation, highest values

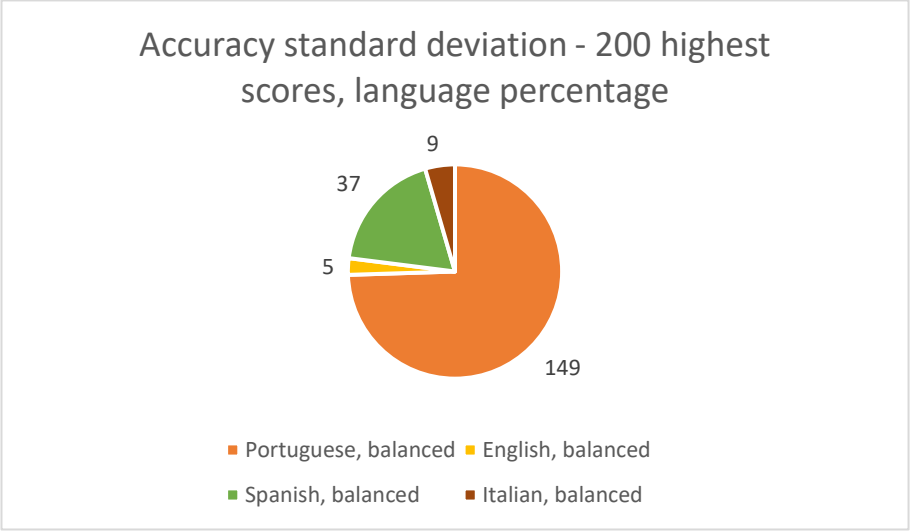

Figure 16. General accuracy scores' standard deviation, highest values language percentage

Taking all this into consideration, from this point on, all accuracy scores from experiments with Portuguese data or unbalanced data will be obsolete, due to the findings regarding the obtained accuracy scores discovered to be untrustworthy. By comparison, balanced Spanish, English and Italian results will be prioritized, since they showed better results.

5.2 Model

This section is dedicated to comparing results for each kind of ML model, as well as a general comparison between all three types. The specific contrasting is necessary because each algorithm has more than one variation.

5.2.1 Naïve Bayes

For Naïve Bayes, there are only two variants: Multinomial and Bernoulli Naïve Bayes. For every language, except English, Multinomial NB gives slightly better results. However, the opposite is true for English. It's easy to try and correlate this with the volume of the data, but if the same accuracy tests are run on a reduced version of the English corpus, with only 10630 rows instead of 94180, the results are virtually the same as the outcome when using 100% of English data. Table 5 showcases all of these findings.

Language	**Naïve Bayes**	**Other specifications**	**Accuracy**	**Std deviation**
English	Bernoulli	All emojis and hashtags, 70/30 train test split, TF-IDF	75.03%	0.53
English	Multinomial		72.64%	0.61
English - reduced	Bernoulli		73.36%	0.78
English - reduced	Multinomial		70.19%	1.07
Italian	Bernoulli		79.99%	0.50
Italian	Multinomial		**80.27%**	0.57
Spanish	Bernoulli		70.95%	0.73
Spanish	Multinomial		72.31%	0.63

Table 5. Sample of Naive Bayes experiments results

The reason why Bernoulli works well for English data is because of the size of the vocabulary. (McCallum, Nigam, & others, 1998) show that when testing on different datasets, when the size of the vocabulary rose, Multinomial NB had a better accuracy score than Bernoulli, which was better for a smaller vocabulary size. However, in the whole English dataset there are over 54000 words, while Italian and Spanish data had fewer than 30000 words. But these are absolute values: If compared to the number of

rows, it appears that the English corpus has a much smaller vocabulary than its Spanish and Italian counterparts.

Figures 17 and 18 correlate to the accuracy differences shown in the above table. The pattern repeats itself throughout all the data, not just the instances shown in the figures. When choosing what kind of NB to use, it seems that Multinomial is preferred for a higher relative vocabulary size.

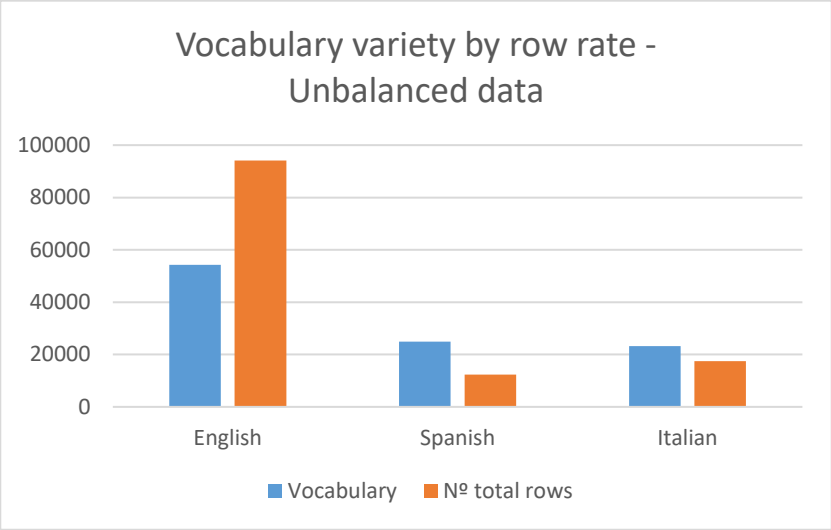

Figure 17. Vocabulary variety by row rate for unbalanced English, Spanish and Italian data

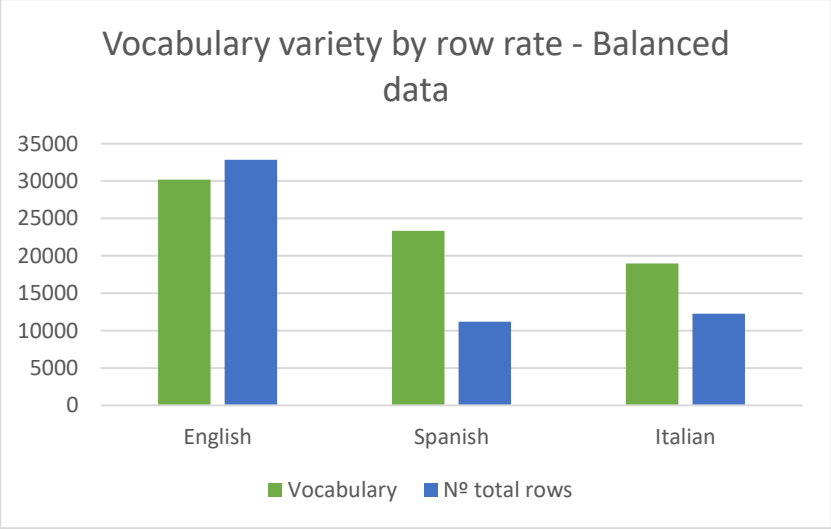

Figure 18. Vocabulary variety by row rate for balanced English, Spanish and Italian data

Finally, regarding general accuracy values, there doesn't seem to be a correlation with extreme values to the kind of NB chosen: both highest (around 80%)

and lowest accuracy scores (around 71%) have Multinomial and Bernoulli experiments associated to them. However, it appears that all highest accuracy scores belong to experiments that used Italian data.

5.2.2 Support Vector Machines

For Support Vector Machines, two parameters were chosen to create the different variations: kernel function: linear or RBF, and C value: 0.1, 1 or 10.

The most notable pattern when contrasting the kernel functions is that there is a higher accuracy for the RBF kernel when using the standard value of C (1) and a large value of C (10). When using a small C value, the opposite happens, and accuracy for the RBF kernel is lower than for a linear kernel function. Specifically for each kernel, the order of highest to lowest accuracies for the linear kernel is small / standard > large. In comparison, such ranking for the RBF kernel is usually standard > large > small. This is visually displayed in Table 6.

The issue of choosing a good value for C is that a value that is too low can result in underfitting, while too large of a value can create overfitting (and a longer execution time), both making accuracy lower. In this case, for the RBF experiments, a value of 0.1 was too small, and a value of 10 was, in most occasions, too big; while for the linear kernel experiments, a larger value of 10 resulted in overfitting, and whether the lower value of C beat the standard value at accuracy values or not fluctuated plenty of times. This parameter, along with others that have not been explored during this project, do not have a determined perfect value, many iterations must be run with different values to find the ideal fit.

Language	**Kernel function**	**C value**	**Other specifications**	**Accuracy**
English	Linear	Small		77.80%
English	Linear	Standard		75.64%
English	Linear	Large		71.81%
English	RBF	Small		71.40%
English	RBF	Standard		79.10%

English	RBF	Large	All emojis and hashtags, 70/30 train test split, TF-IDF	77.82%
Italian	Linear	Small		79.72%
Italian	Linear	Standard		79.60%
Italian	Linear	Large		77.60%
Italian	RBF	Small		71.66%
Italian	RBF	Standard		**80.81%**
Italian	RBF	Large		80.74%
Spanish	Linear	Small		74.36%
Spanish	Linear	Standard		71.60%
Spanish	Linear	Large		69.81%
Spanish	RBF	Small		65.73%
Spanish	RBF	Standard		74.48%
Spanish	RBF	Large		73.91%

Table 6. Sample of Support Vector Machines experiments results

Even though the lowest accuracy scores for linear kernel are usually higher than the lowest scores for RBF kernel, the highest scores of RBF experiments outweigh their linear kernel counterparts (Table 6). These results come as no surprise: they have been observed in studies from different fields (Yekkehkhany, Safari, Homayouni, & Hasanlou, 2014). Also, a linear kernel is better suited for data with a high number of features, and RBF is basically an adaptation of a linear kernel, so linear kernel accuracy doesn't tend to have better results than with an RBF kernel.

A majority of the highest accuracy results (around 81%) came from RBF kernel experiments, especially with a higher C (Fig. 19). Meanwhile, slightly more than half of the lowest scores (from 57% to 71%) belonged to RBF kernel with a small C (Fig. 20).

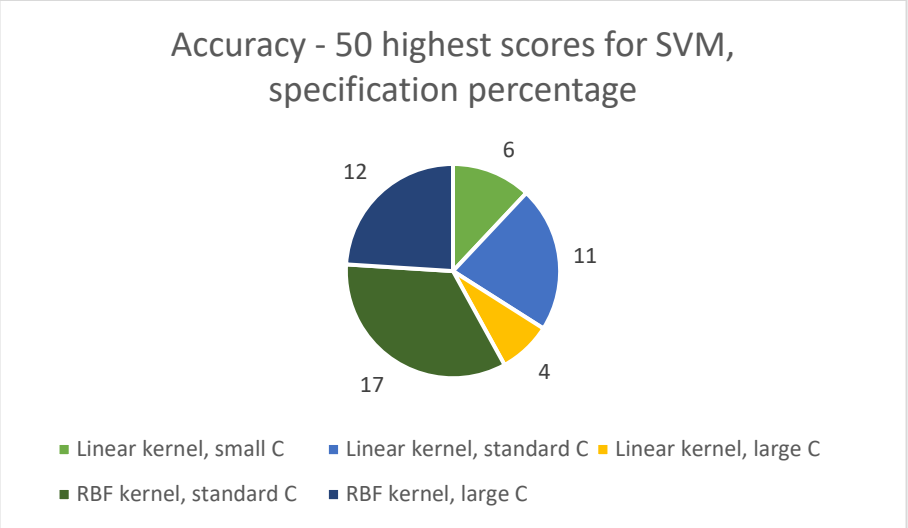

Figure 19. SVM accuracy scores, highest values, specification percentage

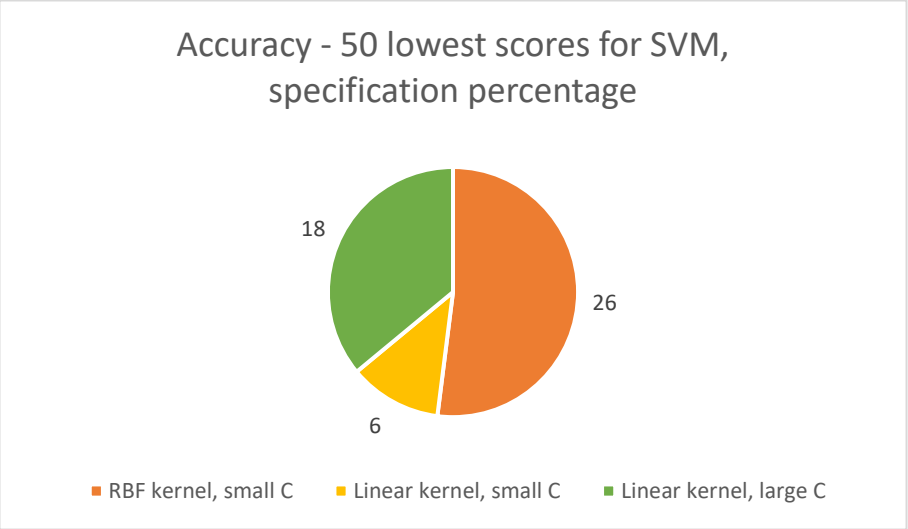

Figure 20. SVM accuracy scores, lowest values, specification percentage

Finally, regarding languages, all high accuracy values consistently use Italian data, while the lowest score come from using English data and Spanish data.

5.2.3 Logistic Regression

Just as with Support Vector Machines, two factors were chosen to create the different variations: the solver: lbfgs or liblinear, and the three values also used in SVM for C value: 0.1, 1 or 10.

When comparing outcomes for different values of C, accuracies from the two solvers were clustered together. This is because when comparing both types of solvers,

there is no notable difference between the accuracy values. Both solvers are adequate when using smaller datasets, and similar results show that there are no factors here which make one solver or the other a perfect match. It would be interesting as future work to incorporate other solvers fit for large datasets, like the sag and saga, into the mix and see the difference.

Concerning C, there is a variety of patterns presents as to what value creates higher accuracy. The patterns are correlated to language and use of TF-IDF. However, the findings show that using a value of 1 is usually the best choice, since it's always in either the first or second place when ranking the values in each distinct pattern (Fig. 21, Table 7).

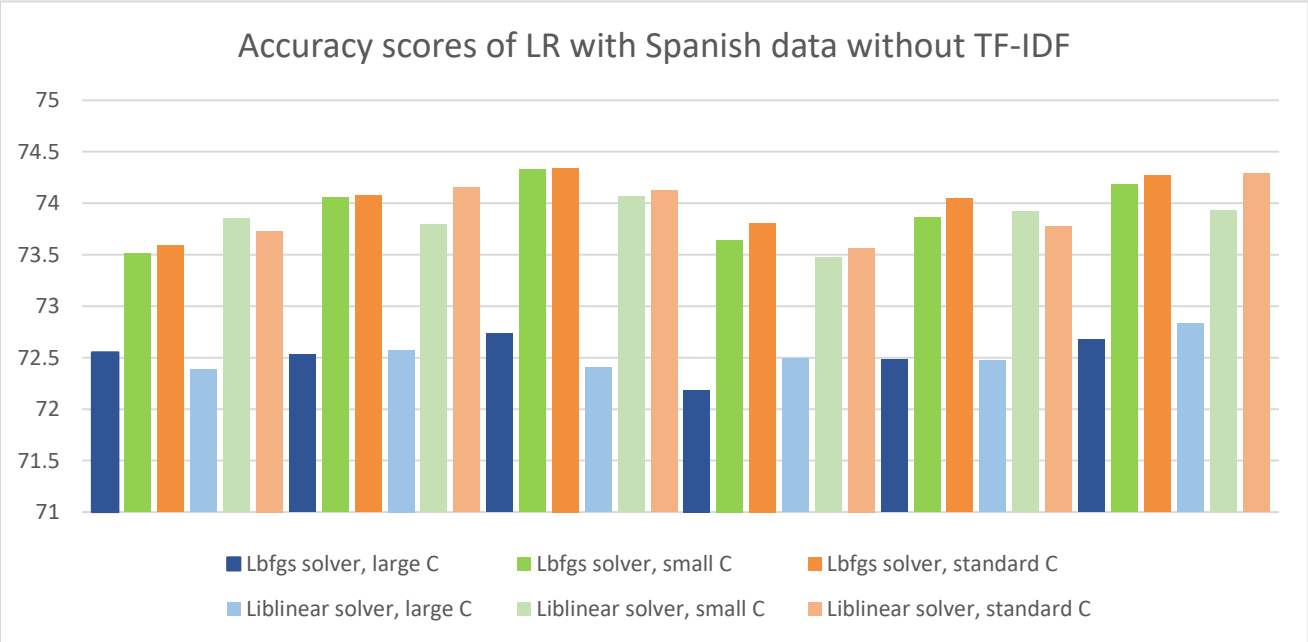

Figure 21. Sample of accuracy results in LR, solver comparison

Language	**TF-IDF**	**C value**	**Average accuracy**	**Accuracy ranking**
English	No	0.1	76.45%	Small C > Standard C > Large C
English	No	1	76.13%	
English	No	10	73.46%	

English	Yes	0.1	74.84%	Standard C > Small C > Large C
English	Yes	1	76.42%	
English	Yes	10	74.39%	
Italian	No	0.1	77.94%	Standard C > Large C > Small C
Italian	No	1	**80.29%**	
Italian	No	10	79.50%	
Italian	Yes	0.1	76.96%	Large C > Standard C > Small C
Italian	Yes	1	79.59%	
Italian	Yes	10	80.13%	
Spanish	No	0.1	73.88%	Standard C > Small C > Large C
Spanish	No	1	73.98%	
Spanish	No	10	72.53%	
Spanish	Yes	0.1	72.32%	Standard C > Large C > Small C
Spanish	Yes	1	74.45%	
Spanish	Yes	10	73.25%	

Table 7. Sample of accuracy results in LR, parameter C comparison

Just like NB and SVM experiments, the highest accuracy values (around 80%) are found when using Italian data, and the lowest results (around 71%) are obtained when using English and Spanish data.

5.2.4 General comparison

When comparing the three analysed ML algorithms (NB, SVM, LR) between them, the results show that Support Vector Machines take the lead (Table 8). The highest obtained accuracy (excluding Portuguese and unbalanced data) is 82.51%, from an experiment which uses an SVM classifier with an RBF kernel and a value of 10 for C. From the 50 top results, 50% of them use Support Vector Machine classifiers, 38% use Logistic Regression and 12% use Naïve Bayes (Fig. 22). For the first two leading models, the main value for C seen in high accuracy values is 1, and 10 is in second place. On the other

end of the spectrum, the lowest accuracies, Support Vector Machines are also the main algorithm, mostly those that use an RBF kernel with a value of 0.1 for C or a linear kernel with C equal to 10 (Fig. 23).

Language	**Algorithm**	**Algorithm specifications**	**Other specifications**	**Accuracy**
Italian	SVM	RBF kernel, large C	All emojis & hashtags, 80/20 train/test split, TF-IDF	**82.51%**
Italian	SVM	RBF kernel, standard C	All emojis & hashtags, 80/20 train/test split, TF-IDF	82.28%
Italian	SVM	RBF kernel, standard C	No emojis or hashtags, 80/20 train/test split, TF-IDF	81.96%
Italian	SVM	RBF kernel, large C	No emojis or hashtags, 80/20 train/test split, TF-IDF	81.83%
Italian	SVM	RBF kernel, large C	All emojis & hashtags, 70/30 train/test split, TF-IDF	81.79%
Italian	SVM	RBF kernel, large C	All emojis & hashtags, 80/20 train/test split, no TF-IDF	81.53%
Italian	SVM	RBF kernel, standard C	All emojis & hashtags, 70/30 train/test split, TF-IDF	81.51%
Italian	SVM	Linear kernel, standard C	All emojis & hashtags, 80/20 train/test split, TF-IDF	81.44%
Italian	LR	Lbfgs solver, standard C	All emojis & hashtags, 80/20 train/test split, no TF-IDF	81.39%
Italian	LR	Liblinear solver, large C	All emojis & hashtags, 80/20 train/test split, TF-IDF	81.37%

Table 8. Highest 10 accuracy results (excluding unbalanced and Portuguese data)

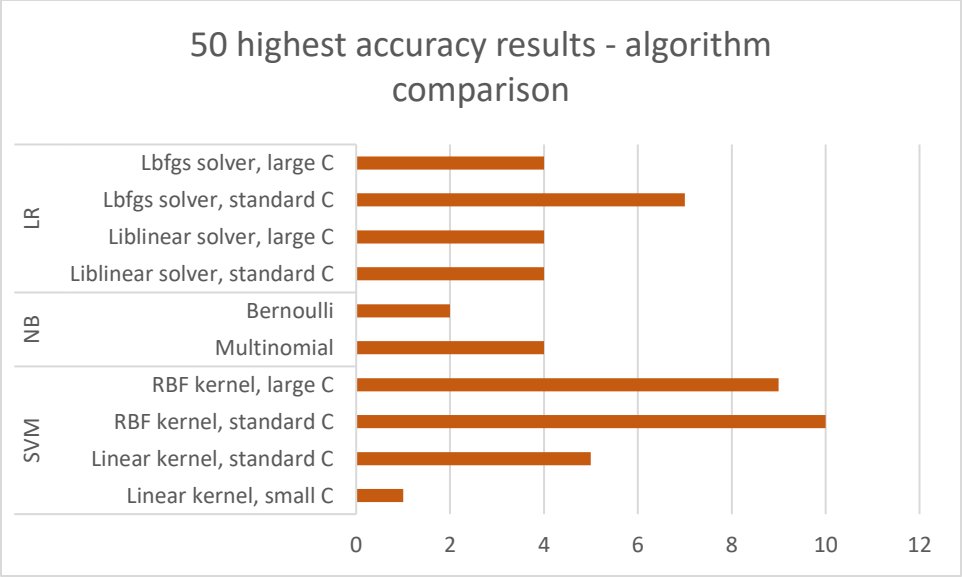

Figure 22. Highest general accuracy values, algorithm comparison

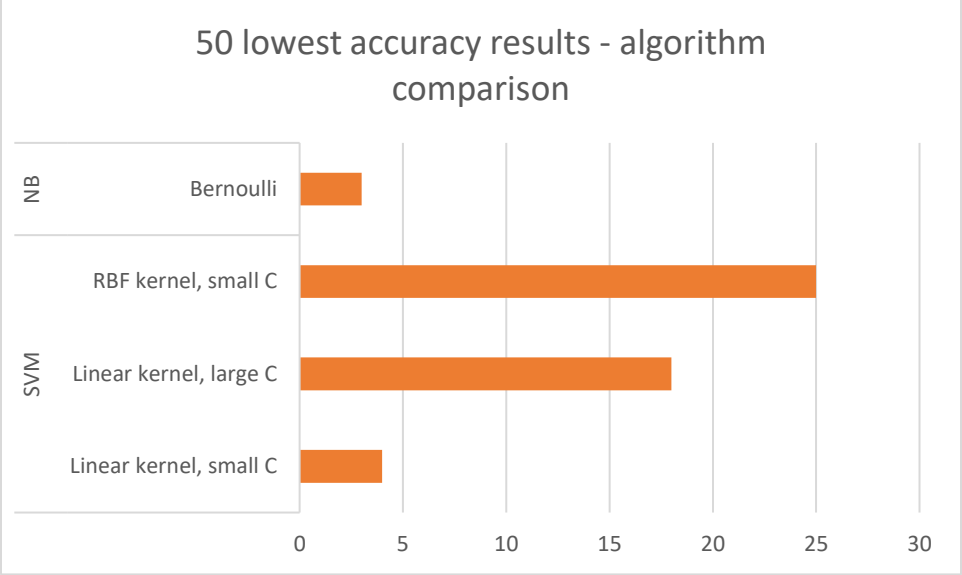

Figure 23. Lowest general accuracy values, algorithm comparison

One thing to consider is that the 94 highest results use Italian data, while the bottom 23 results use either Spanish or English data, so a separate contrasting process for each language was performed. Nevertheless, in all languages there is a similar tendency, where the highest accuracies are obtained using SVM classifiers with a RBF kernel function and a standard value for C (Fig. 24), while the lowest usually use SVM classifiers with a RBF kernel, but with a small value for C (Fig. 25).

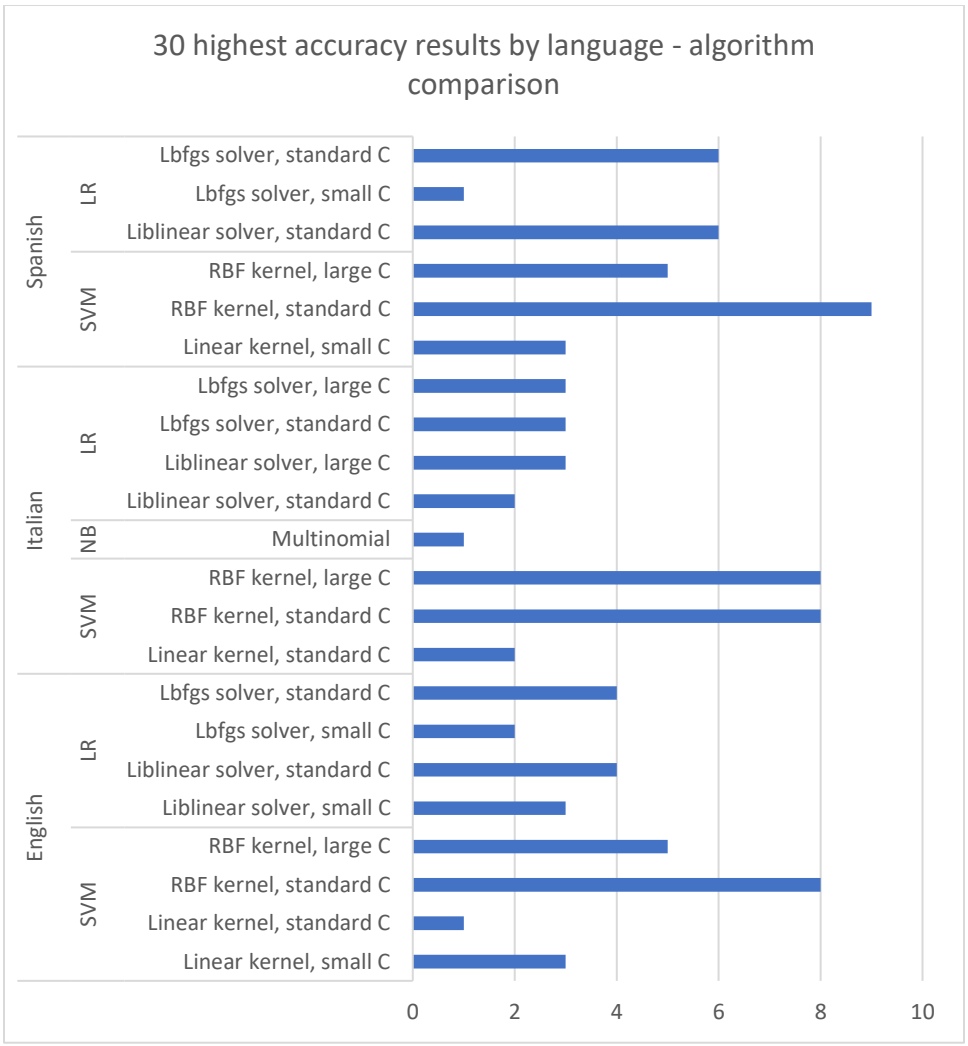

Figure 24. Highest accuracy values by language, algorithm comparison

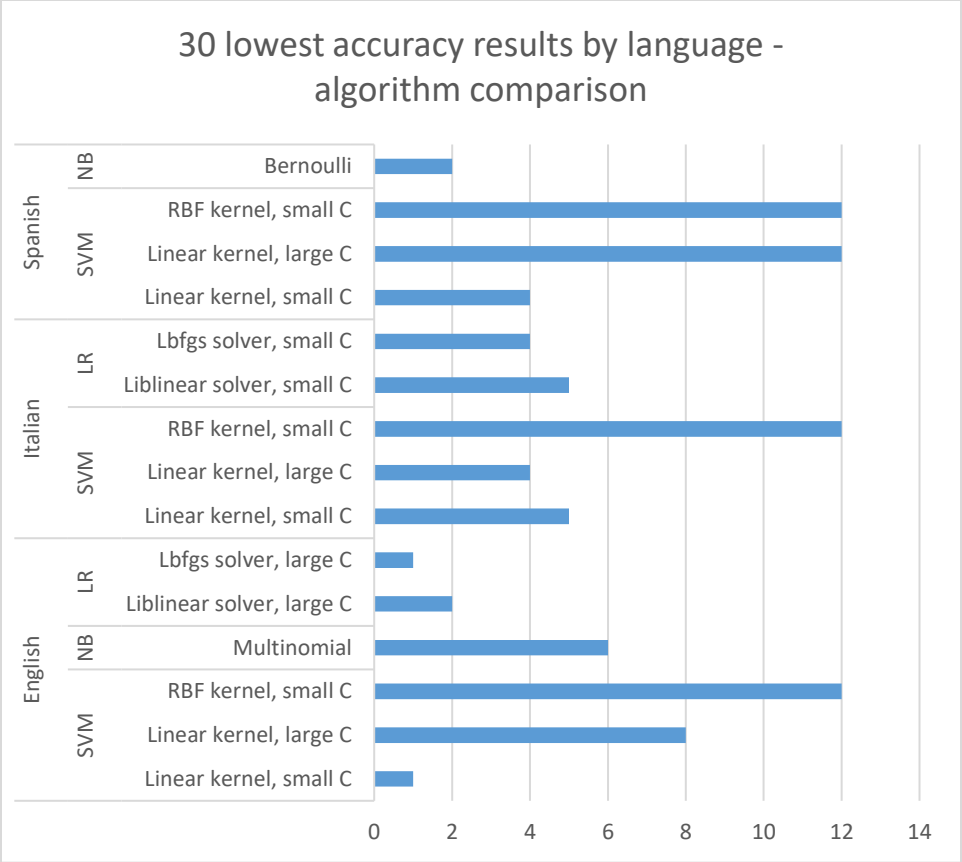

Figure 25. Lowest accuracy values by language, algorithm comparison

5.3 Train/test split

For these experiments, the train/test splitting method was used, and three different splits were used: 80/20, 70/30, and 60/40. Throughout all languages, algorithm types and other factors, the best results are obtained when using an 80/20 split (Figs. 26-28). There are few outliers, where 70/30 split creates the highest accuracy. There are no cases where a 60/40 split gives the best result.

Much like the C parameter in SVM and LR, the ideal train/test split depends on each project, due to how different the data can be regarding its features, size, and other factors. In this case, it seems that 80/20 is a good starting point. Since there are some outliers, exploring the range between 70/30 and 80/20 could lead to a better split.

However, the graphs below show that the difference between results of the splits is small. Other factors mentioned in this chapter contribute to bigger increases and decreases of accuracy, and therefore have more weight in this analysis.

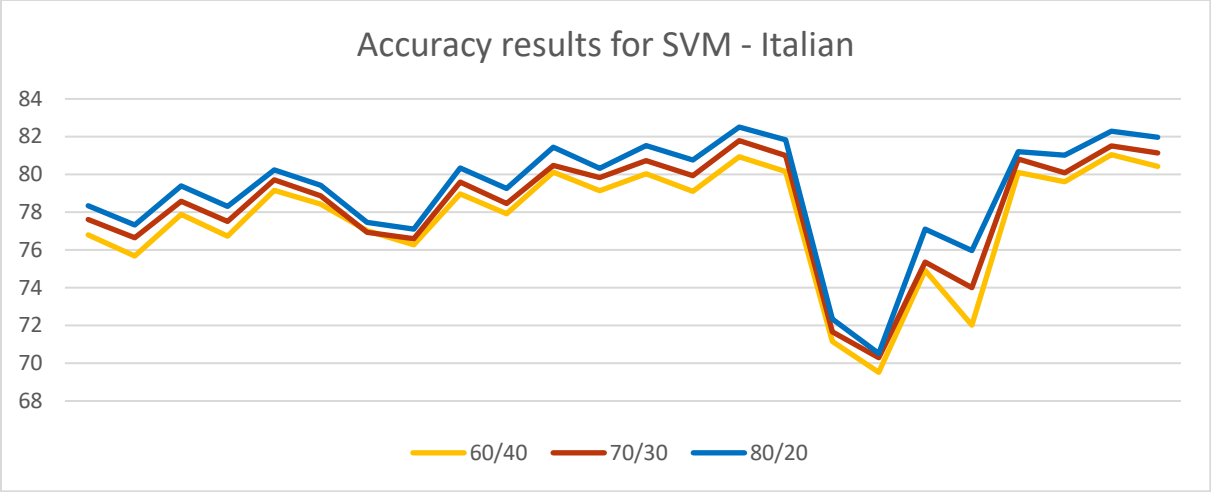

Figure 26. Accuracy results for SVM using Italian data, train/test split comparison

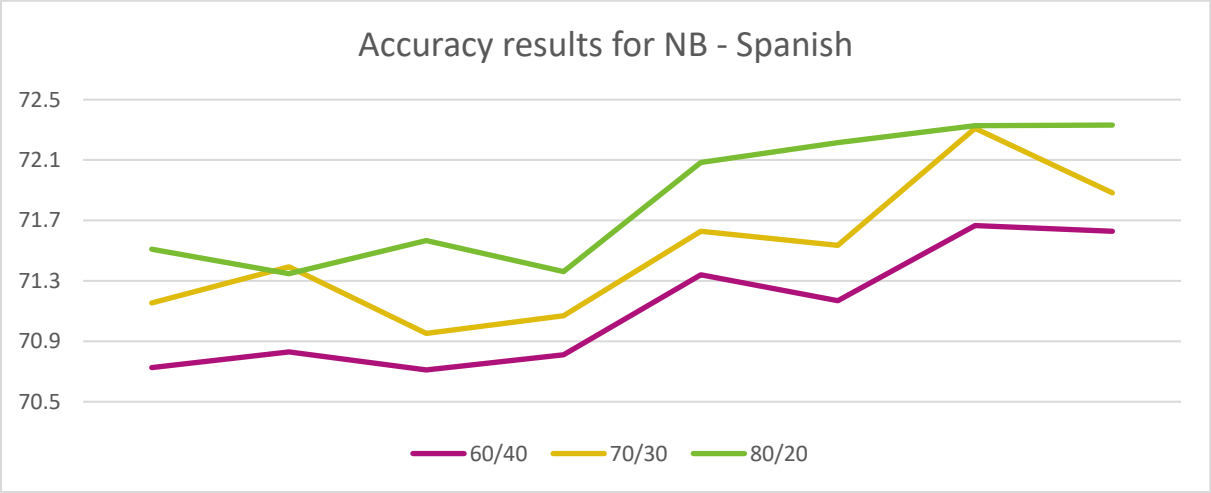

Figure 27. Accuracy results for NB using Spanish data, train/test split comparison

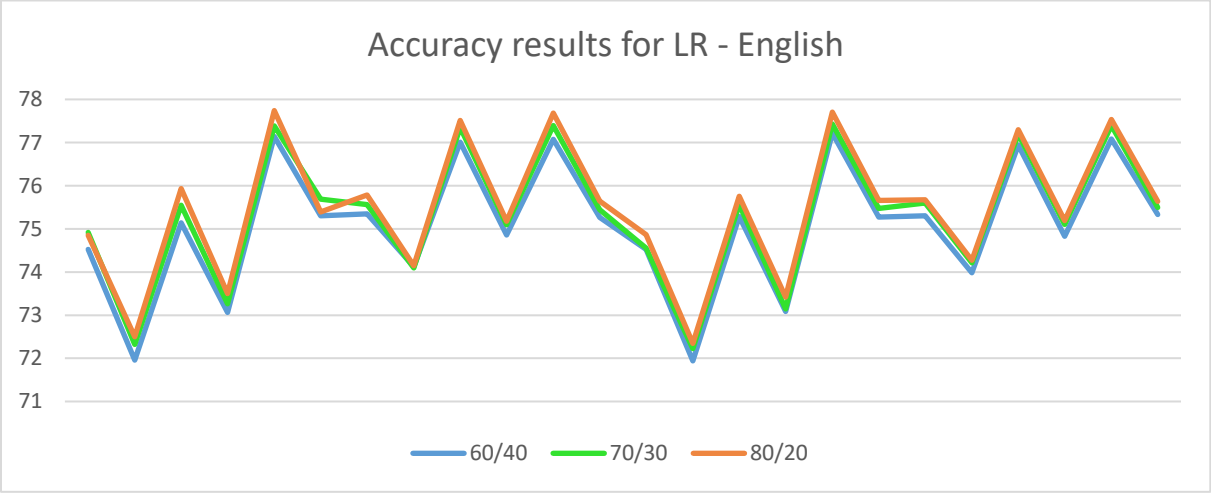

Figure 28. Accuracy results for LR using English data, train/test split comparison

5.4 Use of TF-IDF

For tokenization, two different Sklearn vectorizers were tested. Tfidfvectorizer, which applies TF-IDF, and Countvectorizer, which doesn't, so every word weighs the same.

Throughout all results, those from Logistic Regression and Support Vector Machines stood out because of the wide array of factors that determined whether using TF-IDF was better or not. For Logistic Regression, when using a value of 0.1 for C, there were consistently better results when not using TF-IDF. When raising that value to 10, the opposite was true. However, if the standard value for C was applied, 1, the results varied with language. With Italian data, not using TF-IDF produced higher accuracies, but with English and Spanish data that same action caused poorer performance. Figures 30 to 32 show these findings.

For Support Vector Machines, the relation between C value and usage of TF-IDF was somewhat similar to the patterns in Logistic Regression, but language and kernel function often affected the results. For a low value of C, not using TF-IDF almost always provided better results, except when using Italian data with an RBF kernel. For both the standard and a larger value of the parameter, using TF-IDF was the better option, except when using English data with an RBF kernel. These patterns can be seen in Figures 33 through 38.

This odd pattern can raise some questions as to why this occurs. One reason that could be first hypothesized would be data size or vocabulary volume; however, Spanish and Italian data are similar in those aspects, and experiments with Spanish data follow English data patterns more than Italian data patterns. So, another more convincing reason is that the data in each language is very different in terms of how people who speak the language interact and use vocabulary, for example, slang or online terms. The rarity of some words might not always mean association with hate speech, it might just be, for example, specific colloquialisms or slurs that can be used non-hatefully in specific contexts. Because of details like this, results in this project, and not only usage of TF-IDF, may not extend to other aspects of text classification.

Excluding exceptions due to language, the general conclusion for LR and SVM is that it's best to use TF-IDF unless there is a small value for C, like 0.1. Since a smaller C means that the algorithm is more prone to misclassification, if a data sample with rare words is misclassified, it has a more negative impact if TF-IDF is used.

For Naïve Bayes, depending on the language there seems to be a different pattern. For English data, not using TF-IDF produces better results the majority of the time, while the opposite is true for Spanish and Italian data (Fig. 29).

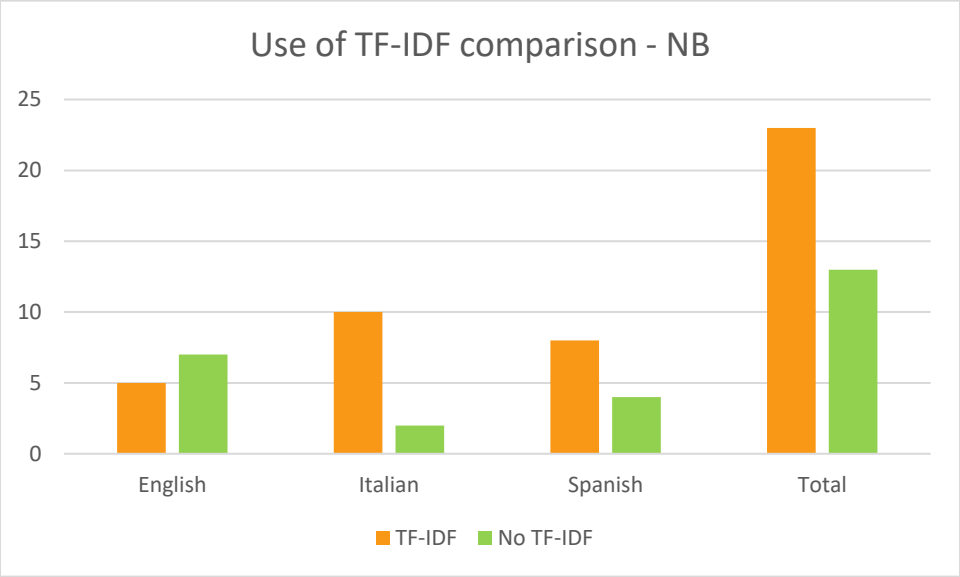

Figure 29. Comparative of accuracy between use/ non-use of TF-IDF in NB by language

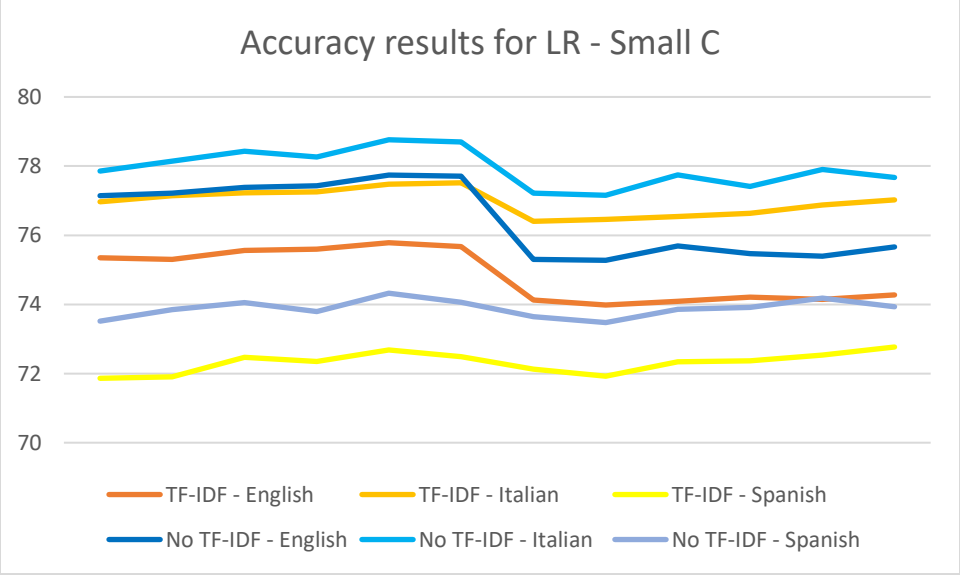

Figure 30. Accuracy results for LR with small C, use of TF-IDF comparison

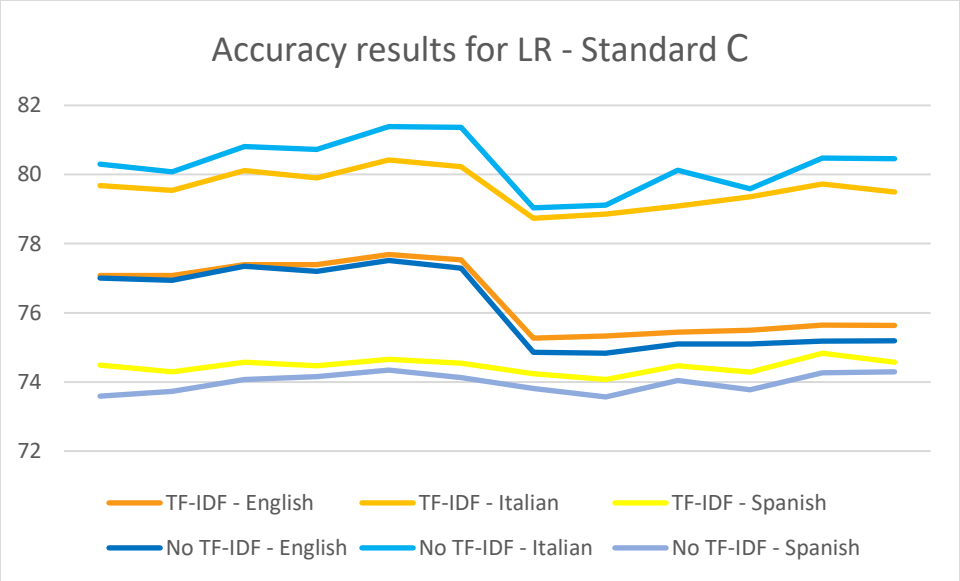

Figure 31. Accuracy results for LR with standard C, use of TF-IDF comparison

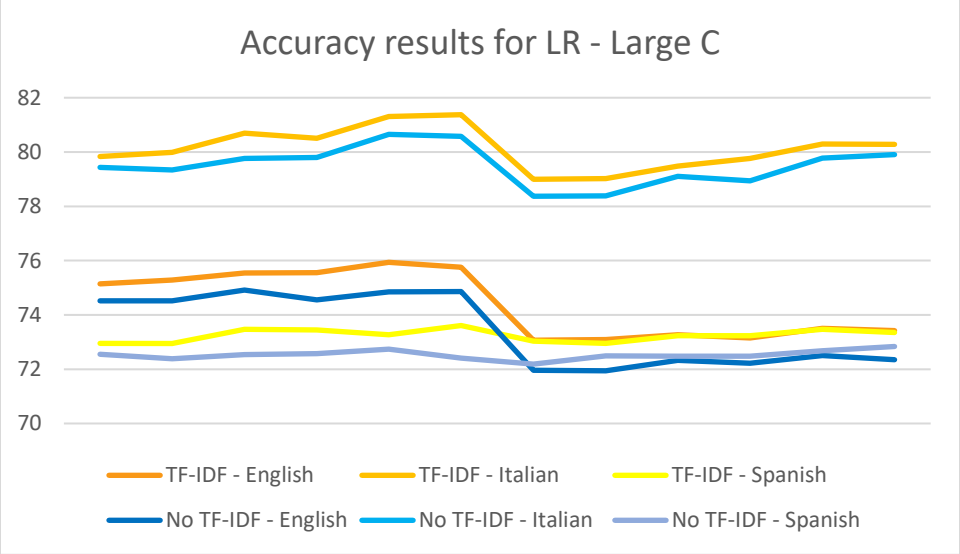

Figure 32. Accuracy results for LR with large C, use of TF-IDF comparison

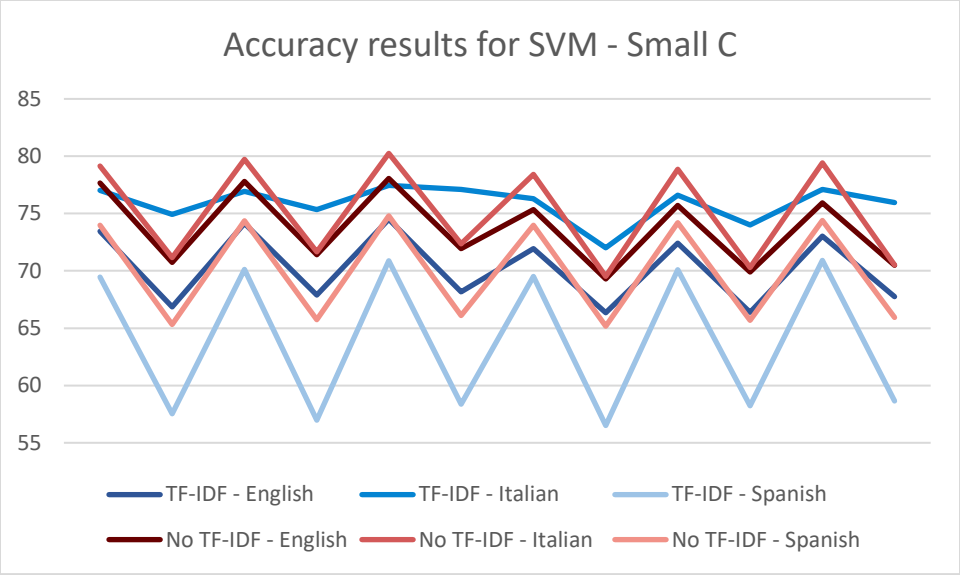

Figure 33. Accuracy results for SVM with small C, use of TF-IDF comparison

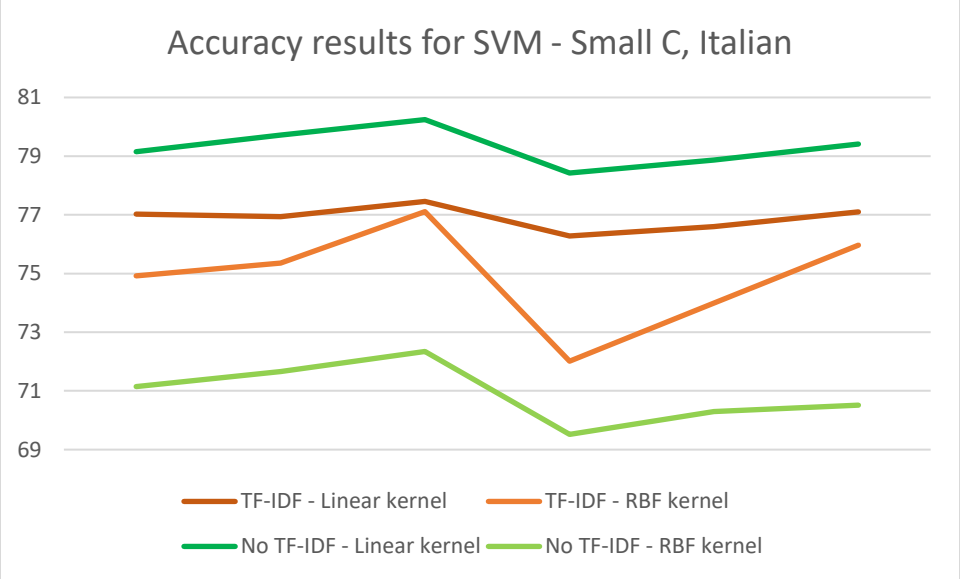

Figure 34. Accuracy results for SVM with small C, Italian data, use of TF-IDF comparison

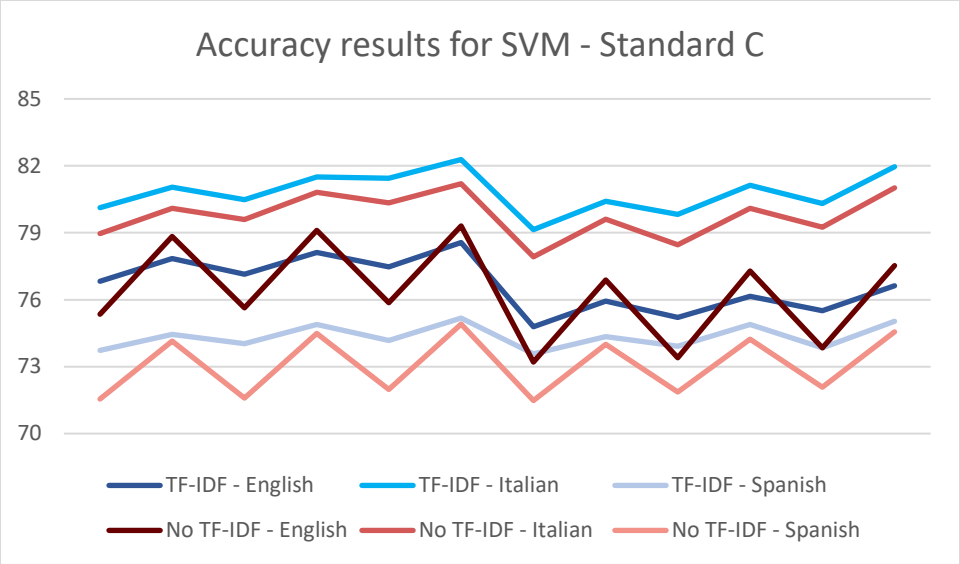

Figure 35. Accuracy results for SVM with standard C, use of TF-IDF comparison

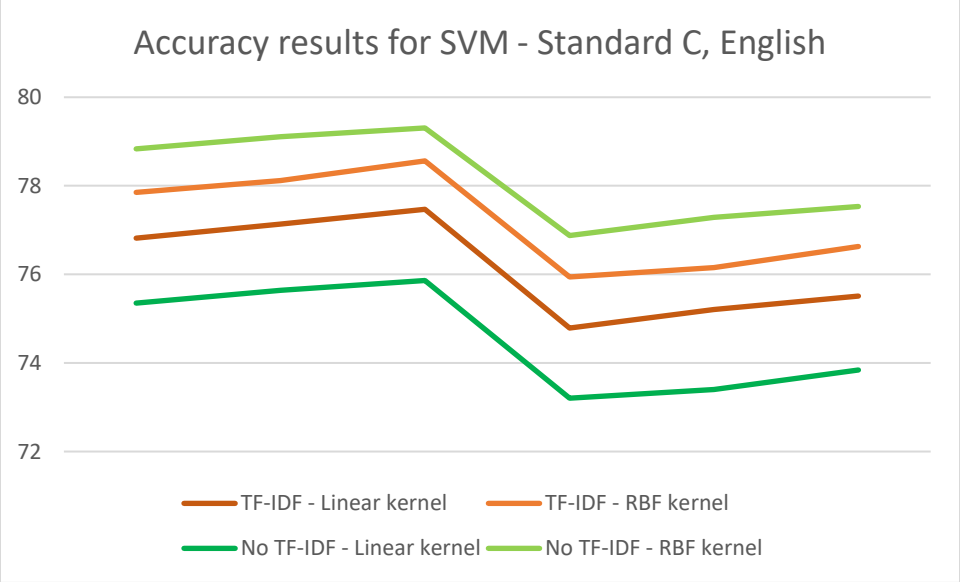

Figure 36. Accuracy results for SVM with standard C, English data, use of TF-IDF comparison

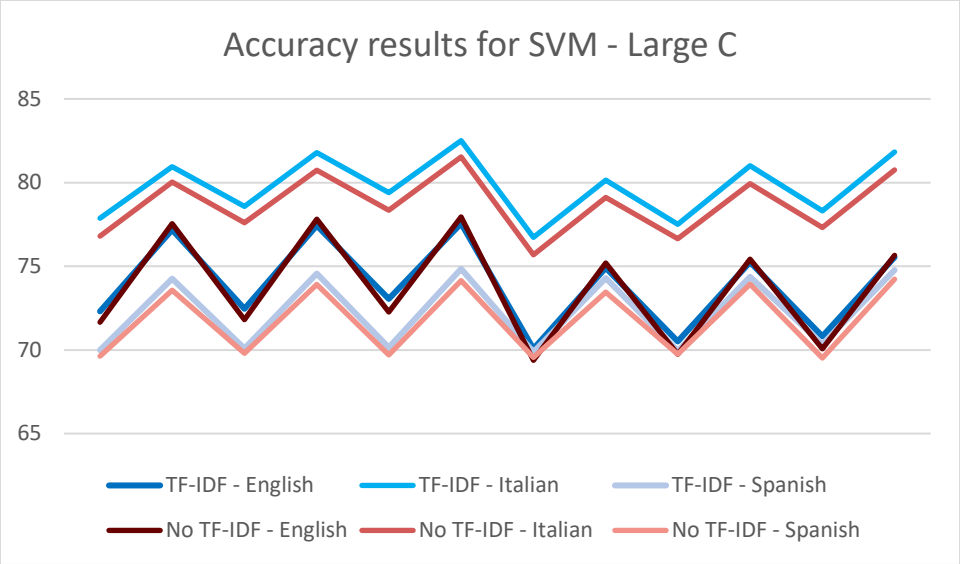

Figure 37. Accuracy results for SVM with large C, use of TF-IDF comparison

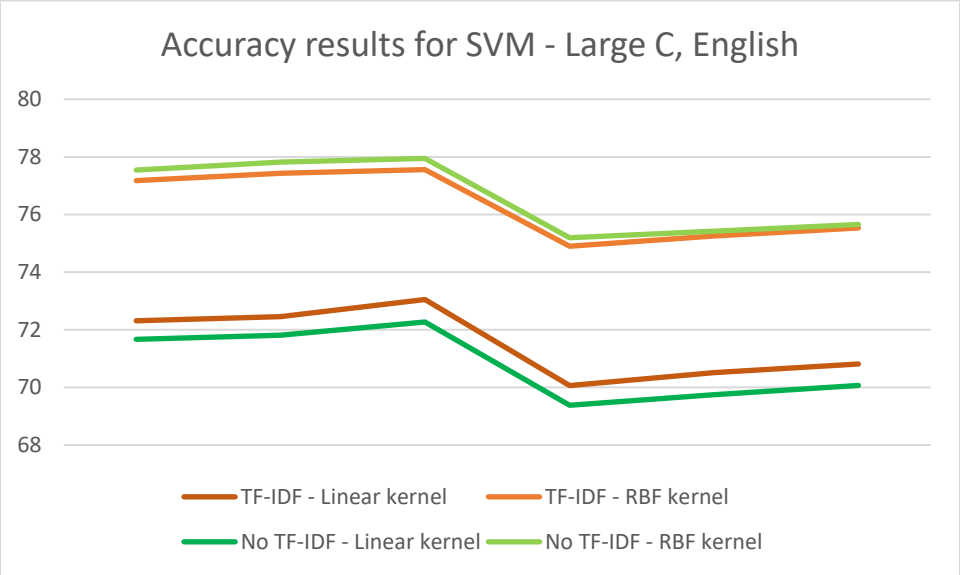

Figure 38. Accuracy results for SVM with large C, English data, use of TF-IDF comparison

5.5 Emoji & hashtag pre-processing

Since the data consists of online posts like tweets, many of them contain emojis and hashtags. These elements are different from normal words, both visually and contextually, so what to do with them can change for every project. In this case, two separate ways to deal with them have been tested: in the first, these elements were maintained: hashtags stayed the same and emojis were transformed into a written format. The second approach is to eliminate them all.

For all algorithm variations in both English and Italian data, using emojis and hashtags proved to produce a higher accuracy 100% of the time (Figs 39-40). However, for Spanish data, the accuracy differences were minimal, and results for data with emojis and hashtags were better only 63% of the time (Fig. 41). This is due to the low rate of elements per row (elements referring to both emojis and hashtags) in Spanish data in comparison to English and Italian data, where a higher amount of valuable information that could teach the model is eliminated (Table 9).

Nevertheless, keeping all elements is a more efficient approach, not only because of these accuracy results with testing data, but because excluding hashtags and emojis from the data prevents the model from learning how people truly write online. Even though these elements don't play the same role as a word in a sentence, they still transmit meaning and emotion, especially when dealing with hate speech, where a single emoji or hashtag can radically switch the tone of the text. For example, in the following hypothetical tweets, the standard text is the same, but the hashtag is not.

- "This woman should just shut up already #GoBackToTheKitchen"
- "This woman should just shut up already #RacismIsMurder"

The first example indicates blatant sexism, while the second could be somebody speaking about a woman who had defended racist beliefs. If emojis and hashtags are cut, the original emotion and meaning can be removed.

Language	**Unique emojis**	**Unique hashtags**	**Emoji instances**	**Hashtag instances**	**Rows**	**Elements/row**
English	510	12893	14131	35400	94180	0.53
Italian	181	3149	1541	8819	17451	0.59
Spanish	234	1818	1691	2656	12301	0.35

Table 9. Emoji & hashtag volume by language

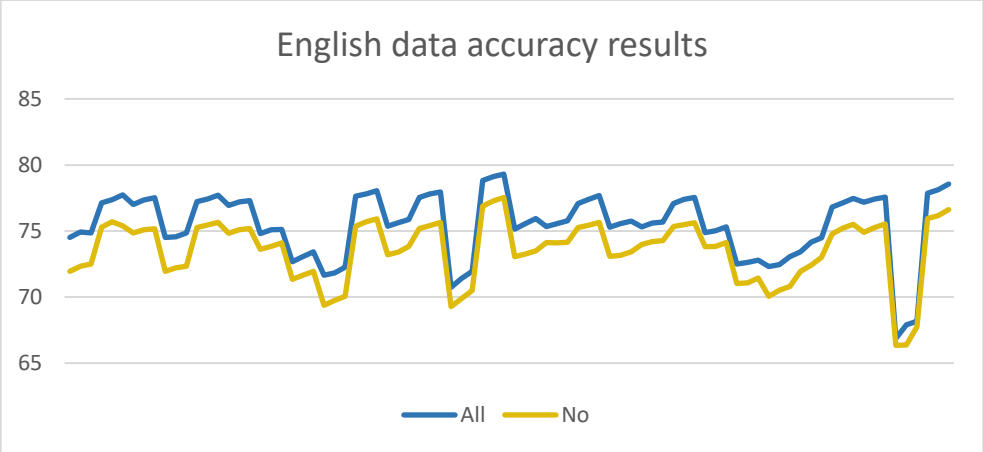

Figure 39. Accuracy results for English data, emoji & hashtag management comparison

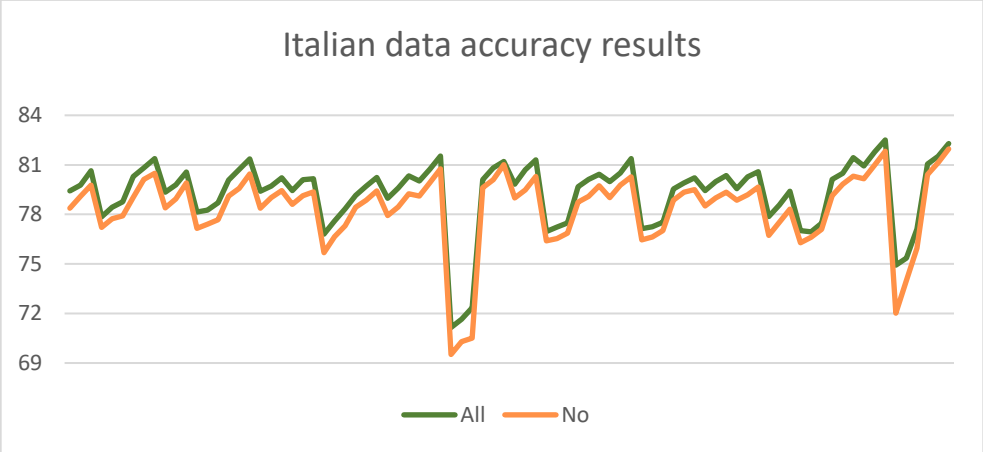

Figure 40. Accuracy results for Italian data, emoji & hashtag management comparison

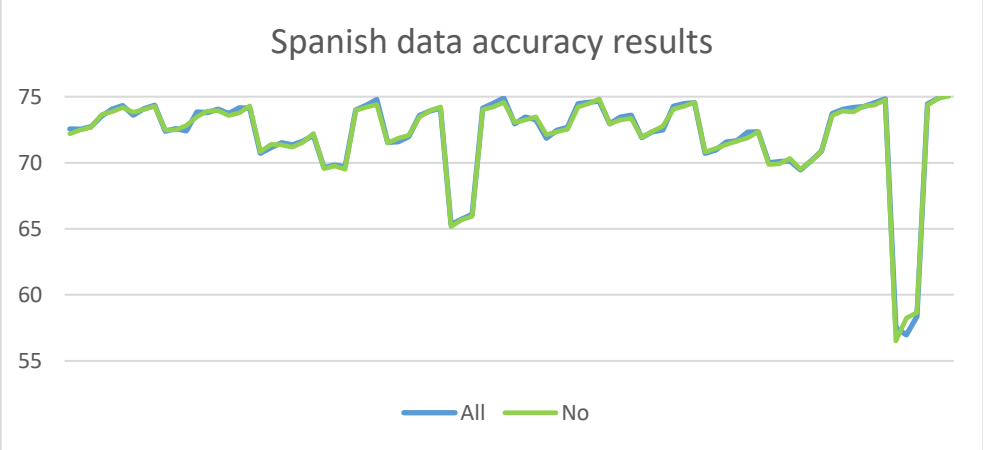

Figure 41. Accuracy results for Spanish data, emoji & hashtag management comparison

5.6 Final findings

Throughout all the experiments, the best accuracy, 95.36%, was obtained using Portuguese, unbalanced data with all emojis and hashtags, an 80/20 train/test split, a SVM classifier with an RBF kernel function and a value of 10 for C, and no TF-IDF applied. However, previous sections have showed why Portuguese and unbalanced data are untrustworthy. If this data is disregarded, the best accuracy is an **82.51%**, using Italian, balanced data with all emojis and hashtags, an 80/20 train/test split, a SVM classifier with an RBF kernel function and a value of 10 for C, with TF-IDF. An iteration of this experiment shows a F1-score of 83%. For each of the three considered languages, Spanish, Italian and English, the best results occur when using all emojis and hashtags in the data, having an 80/20 split and using an SVM classifier with an RBF kernel and either a value of 1 or 10 for C.

The lowest accuracy (not counting Portuguese and unbalanced data) was a 56.51% using Spanish, balanced data with no emojis or hashtags, a 60/40 split, a SVM classifier with an RBF kernel function and a value of 0.1 for C, with TF-IDF. For all languages, the worst performance was obtained when using data without emojis or hashtags, a 60/40 split and an SVM classifier with an RBF kernel and a small C value.

There are many factors than can affect why results are different depending on language. Firstly, the datasets for each language use text from different sources: Twitter, Gab, news sites, forums, etc. Each online space contains different ways to convey messages: in news sites, people react to a piece of news and make a comment on it. On Twitter, tweets do not need to be a reaction to other content. In forums, people interact with each other and can simulate normal conversations. Additionally, some datasets focus on detecting specific kinds of hate speech (sexism, racism, etc.) Having too much data on a particular type of hate speech can worsen performance when detecting hate speech as a whole. On top of that, as mentioned in section 5.4, people use vocabulary in different ways depending on language. This can affect how the ML algorithms learn.

In section 2.6, three different studies on hate speech detection were presented as past examples of usage of the technology in this project. Although they have a

similar objective and approach to this project, some elements are different, and so, the best accuracy values in each of the four projects are not identical.

When comparing best results with (Abro, et al., 2020), this project has a better performance, the aforementioned 82.51% accuracy, versus a 79% accuracy for the external study, obtained with a SVM classifier using bigrams. However, Abro's study data is in English, as is the data in all three example studies. The highest accuracy in this project for English, balanced data is 79.31%. Although these results are similar between the two studies, many elements are different. The most notable contrast is that Abro's study does not balance data: only 16% of it is classified as hate speech, but F1-score is not affected, with a score of 77% for the experiment with the highest accuracy. The other differences are the pre-processing process, the use of bigrams, and how hashtags are treated. This project does not convert text to lowercase or use stemming. Although the first mentioned technique might not always produce better results, since text in all capitals online can indicate emotional information, like excitement or anger, using stemming or lemmatization could have been beneficial to this project's pre-processing. Also, the use of n-grams has not been explored, so bigrams were not used. Instead, when using either CountVectorizer or TfidfVectorizer, the parameter "ngram_range" was not called, so it set to the default value, which is only unigrams. Even though half of this project's experiments delete all hashtags in the data, the highest accuracies occur when this is not done. Usage of hashtags in data proved to be an advantage here, however, it could have been a disadvantage in the external study. Regardless, their results are good even with the handicap of having unbalanced data thanks to the pre-processing and parameter adjusting this project lacks.

The best performance in the next analysed study, (Malmasi & Zampieri, 2017) was a 78% accuracy when using a SVM classifier with character 4-grams. Since they reported their results in terms of accuracy only, the F1-score is unknown. However, the implementation of the classifier is not the same: this project uses Sklearn's, while Malmasi uses the LIBLINEAR package. Just like in the previous comparison, there are differences in pre-processing: this study removes all emojis, URLs and converts text into lowercase. The elimination of emojis is comparable to hashtag elimination in Abro's study. In addition, 10-fold cross validation is used instead of a train/test split, and

different types of n-grams were analysed. There is also no mention of whether the data is balanced or not. Despite all this, the highest accuracy in English in this project exceeds that of Malmasi's study by 1.31%.

In the final examined study (Warner & Hirschberg, 2012), for non-humanly classified experiments, the highest obtained accuracy and F1-score was 94% accuracy and a 0.63 F1-score, which is a big drop from the accuracy. These results came from using a SVM classifier. Even though their best accuracy is higher than that of this project's, the low F1-score indicated that more pre-processing or balancing of the data is needed, since this project's experiments with balanced data always show an F1-score similar to the accuracy. This study's development is the most different to this project than any of the other presented examples, mainly because of how the data is structured. Instead of short posts, such as tweets, the corpus was formed of full paragraphs. Other differences include the choice of using a 10-fold cross validation instead of a train/test split and the usage of POS tagging in the data. A final interesting detail is that Warner's study notes that adjusting the C parameter had no effect, which is not the case here.

Something all the studies had in common with each other but not with this development is that the number of classes data could fit in is higher than two. Instead of just a "hate speech" and "non-hate speech" classification, the studies either used a tri-class system, by separating non-offensive text, offensive text and hateful text, or a multi-class system, where hate speech text could be categorized into many labels which indicated the specific target. Also, in the first two cases the size of the data is smaller than the obtained corpus for this project (for Warner's study, this is unknown). This might lead to thinking that data size affects performance, and to an extent this is true, as proven with how different train/test splits and emoji and hashtag treatment are important, but the highest accuracy scores have been obtained using Italian data, which contains fewer rows than English data, so data volume should be investigated more thoroughly to reach a certain conclusion.

These findings may not extend to other types of text classification, since hate speech is characterized by using a series of particular words or phrases to degrade marginalized social groups. How these terms are used is specific to hate speech. For

example, in categorization of news articles into topics, the appearance of certain terms alone can indicate the topic of an article (if the text contains "nominees" and "movie" it is a safe bet to say the article is about film awards). However, for detecting hate speech, this is not the case. In sentiment analysis of online reviews, sarcasm may occur as a result of anger in a negative review, just like with hate speech. Even so, hate speech differs because of the use of dog whistles and stereotypical phrases that at first glance, may not seem harmful.

Chapter 6 - Conclusions and future work

6.1 Conclusions

At the beginning of this project, a series of goals were set. In these final conclusions, those objectives will be revised in order to evaluate to what extent they have been achieved.

The first objective was to choose and develop ML algorithms able to predict hate speech in four languages. As showed previously, good performance has been achieved in three languages: Italian, Spanish, and English. Portuguese experiments were discarded completely due to unreliable results: although accuracy was very high, this was because most instances were labelled as not hate speech, misclassifying a large number of data samples. Unbalanced data experiments were also ignored for the same reasons. The best outputs came from using Italian data, reaching accuracies of around 82%, English best accuracy tops 79% and Spanish accuracy results peak at around 75%.

The second goal was to reach conclusions as to what approach is best when detecting hate speech. During the development of the project, many variations of three kinds of ML algorithms were tested for hate speech detection in four languages. In total, 1344 different experimental results were obtained. Results showed that for the chosen data, data pre-processing and models, balancing the data, i.e., cutting data from the corpus so that the amount of data corresponding to each category is 50/50, is a preferable choice for better performance. For each language, the experiments produced different outcomes.

For all considered languages, these additional conclusions were reached:

- Using a 80/20 train/test split proved to be the optimal choice, compared to other splits with a lower train data percentage.
- Support Vector Machines showed better performances compared to Naïve Bayes and Logistic Regression.
    - Using an RBF kernel function is preferable to a linear kernel function.

    - Having a higher value for the regularization parameter C, for example 1 or 10, is far better than a very low value, like 0.1.
- Using all emojis and hashtags in the data as opposed to removing them retains more valuable information for the models to learn, and so, leads to better results.

As regards to use of TF-IDF, whether the use of it helps obtain higher accuracies or not fluctuates depending on language, algorithm, and other factors. However, in most cases TF-IDF is the preferred technology to implement for better results.

Finally, this project did not intend to merely observe when the better results occur, but to also understand such outcomes. For each different factor in the experiments, an interpretation of the results was carried out as described in chapter 5, thus successfully meeting this project's final objective.

6.2 Future work

Because of lack of time, F1-score, precision and recall tests were not able to be run and analysed. For future work, they should be considered in order to carry out a more thorough analysis. This way, more unbalanced datasets could be used for input: a high accuracy score would not be misleading, since the rest of the results could reveal a poor performance. Also, it would be interesting to experiment with other elements that were mentioned when comparing to other studies, such as n-grams, more pre-processing techniques, use of k-fold cross validation, etc. In addition, exploring Deep Learning approaches such as neural networks and comparing the results with Machine Learning algorithms would be a good addition to the project.

It is also important to focus on the data that is used for training. During this project there have been occasions where manual classification was necessary. Doubt when labelling data cannot exist if accurate prediction is the goal. But correct labelling is not limited to just nuances in the text, mentioned in previous sections. Context is very important, especially online, when satire, inside jokes and criticism take new forms at a rapid pace. For example, the following tweet was classified as hateful in the used data:

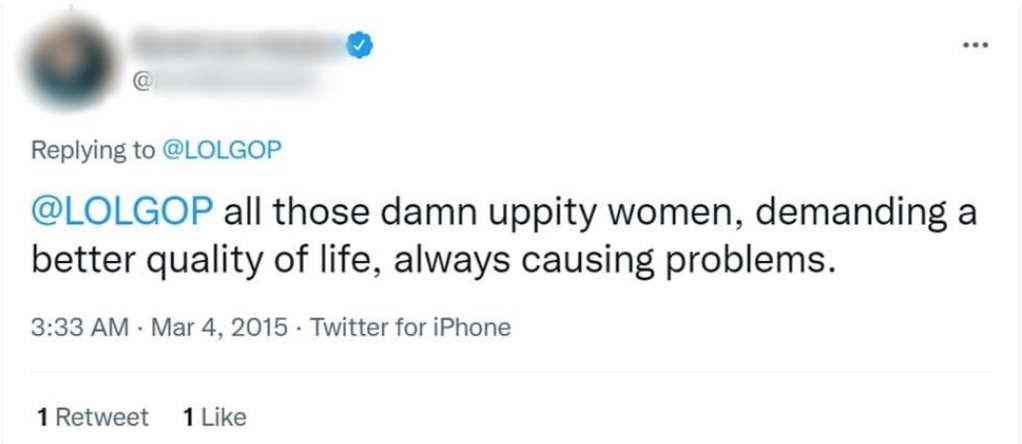

Figure 42. Tweet categorized as hate speech in the data

However, this person tweeted this satirically. When looking at the rest of the account, we see support towards women and other minorities, so, potentially, if a hate speech detecting tool classified this as hate speech, it would be wrong, since there was no malice or ignorance in this person's intentions, only sarcasm.

Therefore, I propose further research when obtaining data by examining the context of possible hateful speech online: this means taking into consideration other posts from the accounts, and deep research into Internet culture and knowledge on hateful dog whistles that may appear both online and in real life. This would require a lot of work, and larger teams, but I believe that such effort would be beneficial for finding solutions in the future for the problem of letting hate speech run wild on the ever so influential place that is the Internet.

BIBLIOGRAPHY

Abro, S., Shaikh, S., Khand, Z. H., Zafar, A., Khan, S., & Mujtaba, G. (2020). Automatic hate speech detection using machine learning: A comparative study. *11(8)*. Science and Information (SAI) Organization Limited.

Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017). Deep Learning for Hate Speech Detection in Tweets. Retrieved from http://arxiv.org/abs/1706.00188

Basile, V., Bosco, C., Fersini, E., Nozza, D., Patti, V., Rangel Pardo, F. M., . . . Sanguinetti, M. (2019, June). SemEval-2019 Task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. *Proceedings of the 13th International Workshop on Semantic Evaluation*, 54-63. Minneapolis, Minnesota, USA: Association for Computational Linguistics. doi:10.18653/v1/S19-2007

Bianchi, C. (2014). Slurs and appropriation: An echoic account. *66*, 35-44. Journal of Pragmatics. doi:https://doi.org/10.1016/j.pragma.2014.02.009

Blake, K. R., O'Dean, S. M., Lian, J., & Denson, T. F. (2021, March). Misogynistic Tweets Correlate With Violence Against Women. *Psychological Science, 32*, 315-325. doi:10.1177/0956797620968529

Bliuc, A.-M., Faulkner, N., Jakubowicz, A., & McGarty, C. (2018). Online networks of racial hate: A systematic review of 10 years of research on cyber-racism. *87*, 75-86. Retrieved from https://doi.org/10.1016/j.chb.2018.05.026.

Blodgett, S. L., Barocas, S., III, H. D., & Wallach, H. (2020). Language (Technology) is Power: A Critical Survey of "Bias" in NLP. Retrieved from https://arxiv.org/abs/2005.14050

Brown, S. (2021, April). Machine learning, explained.

Butt, S., Ashraf, N., Sidorov, G., & Gelbukh, A. (2021). Sexism Identification using BERT and Data Augmentation – EXIST2021.

Caselli, T., Basile, V., Mitrović, J., & Granitzer, M. (2020). HateBERT: Retraining BERT for Abusive Language Detection in English. Retrieved from https://arxiv.org/abs/2010.12472

Chen, M.-Y. (2011). Predicting corporate financial distress based on integration of decision tree classification and logistic regression. *38(9)*, 11261-11272. Expert Systems with Applications. doi:https://doi.org/10.1016/j.eswa.2011.02.173

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017, May). Automated Hate Speech Detection and the Problem of Offensive Language. *11*, 512-515. Retrieved from https://ojs.aaai.org/index.php/ICWSM/article/view/14955

Delgado, R., & Stefancic, J. (2017). *Critical race theory: An introduction.* NyU press.

Feldman, R. (2013, April). Techniques and Applications for Sentiment Analysis. *56(4)*, 82-89. New York, NY, USA: Association for Computing Machinery. doi:10.1145/2436256.2436274

Fersini, E., Nozza, D., & Rosso, P. (2020). AMI @ EVALITA2020: Automatic Misogyny Identification. doi:10.4000/books.aaccademia.6764

Fersini, E., Rosso, P., & Anzovino, M. (2018). Overview of the Task on Automatic Misogyny Identification at IberEval 2018.

Fortuna, P., Silva, J. R., Soler-Company, J., Wanner, L., & Nunes, S. (2019). A Hierarchically-Labeled Portuguese Hate Speech Dataset. *Proceedings of the 3rd Workshop on Abusive Language Online (ALW3)*.

Founta, A.-M., Djouvas, C., Chatzakou, D., Leontiadis, I., Blackburn, J., Stringhini, G., . . . Kourtellis, N. (2018). Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior. Retrieved from http://arxiv.org/abs/1802.00393

Gandhi, R. (2018, June 7). Support Vector Machine — Introduction to Machine Learning Algorithms. Towards Data Science.

García-Díaz, J. A., Cánovas-García, M., Colomo-Palacios, R., & Valencia-García, R. (2021). *Detecting misogyny in Spanish tweets. An approach based on linguistics features and word embeddings* (Vol. 114). doi:10.1016/j.future.2020.08.032.

Gibert, O. d., Perez, N., García-Pablos, A., & Cuadros, M. (2018, October). Hate Speech Dataset from a White Supremacy Forum. *Proceedings of the 2nd Workshop on Abusive Language Online ({ALW}2)*, 11-20. Brussels, Belgium. doi:10.18653/v1/W18-5102

Goenaga, I., Atutxa, A., Gojenola, K., Casillas, A., Ilarraza, A. D., Ezeiza, N., . . . Viñaspre, O. P. (2018). Automatic Misogyny Identification Using Neural Networks.

Hsu, C.-W. a.-C.-J. (2003). A practical guide to support vector classification. Taipei, Taiwan.

IBM Cloud Education. (2020, May 1). Deep Learning. IBM.

Kennedy, B., Atari, M., Davani, A. M., Yeh, L., Omrani, A., Kim, Y., . . . al., e. (2022, April). The Gab Hate Corpus. doi:10.17605/OSF.IO/EDUA3

Khan, M. A. (2020). Detection and classification of plant diseases using image processing and multiclass support vector machine. *Int. J. Comput. Trends Technol, 68(4)*, 5-11.

Kononenko, I. (2001). Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine, 23(1)*, 89-109. doi:https://doi.org/10.1016/S0933-3657(01)00077-X

Krichene, A. (2017). Using a naive Bayesian classifier methodology for loan risk assessment: Evidence from a Tunisian commercial bank. *22(42)*, 3-24. Emerald Publishing Limited.

Leite, J. A., Silva, D. F., Bontcheva, K., & Scarton, C. (2020). Toxic Language Detection in Social Media for Brazilian Portuguese: New Dataset and Multilingual Analysis. Retrieved from https://arxiv.org/abs/2010.04543

Malmasi, S., & Zampieri, M. (2017). Detecting hate speech in social media.

Mandl, T. a. (2019). Overview of the HASOC Track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages. 14-17. India: Association for Computing Machinery. doi:10.1145/3368567.3368584

Mandl, T. a. (2020). Overview of the HASOC Track at FIRE 2020: Hate Speech and Offensive Language Identification in Tamil, Malayalam, Hindi, English and German. USA. doi:10.1145/3441501.3441517

Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. *Cambridge University Press*.

McCallum, A., Nigam, K., & others. (1998). A comparison of event models for naive bayes text classification. *AAAI-98 workshop on learning for text categorization, 752(1)*, 41--48.

Mitchell, T. M. (1997). Machine learning. *1(9)*. McGraw-hill New York.

Muller, B. (2022, March 2). BERT 101 🤗 State Of The Art NLP Model Explained. Hugging Face.

Okesola, O. J., Okokpujie, K. O., Adewale, A. A., John, S. N., & Omoruyi, O. (2017). An improved bank credit scoring model: A naïve Bayesian approach. *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*, 228--233.

Park, J. H., & Fung, P. (2017). One-step and Two-step Classification for Abusive Language Detection on Twitter. Retrieved from http://arxiv.org/abs/1706.01206

Pascoe, C. J., & Diefendorf, S. (2019). No Homo: Gendered Dimensions of Homophobic Epithets Online. doi:10.1007/s11199-018-0926-4

Paula, A. F., Silva, R. F., & Schlicht, I. B. (2021). Sexism Prediction in Spanish and English Tweets Using Monolingual and Multilingual BERT and Ensemble Models.

Pedregosa, F. a. (2011). Scikit-learn: Machine Learning in Python. *12*, 2825--2830. Journal of Machine Learning Research.

Pelle, R. d., & Moreira, V. (2017). Offensive Comments in the Brazilian Web: a dataset and baseline results. *Anais do VI Brazilian Workshop on Social Network Analysis and Mining*. São Paulo. doi:10.5753/brasnam.2017.3260

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. CoRR. Retrieved from http://arxiv.org/abs/1802.05365

Plaza-Del-Arco, F.-M., Molina-González, M. D., Ureña-López, L. A., & Martín-Valdivia, M. T. (2020). *Detecting Misogyny and Xenophobia in Spanish Tweets Using Language Technologies* (Vol. 20). New York, USA: ACM Trans. Internet Technol. doi:10.1145/3369869

Rish, I., & others. (2001). An empirical study of the naive Bayes classifier. *IJCAI 2001 workshop on empirical methods in artificial intelligence, 3(22)*, 41-46.

Rodríguez-Sánchez, F., Carrillo-de-Albornoz, J., Plaza Morales, L., Gonzalo Arroyo, J., Rosso, P., Comet, M., & Donoso, T. (2021). Overview of EXIST 2021: sEXism Identification in Social neTworks. doi:10.26342/2021-67-17

Sanguinetti, M., Comandini, G., Nuovo, E., Frenda, S., Stranisci, M., Bosco, C., . . . Russo, I. (2020, December). HaSpeeDe 2 @ EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task.

Sanguinetti, M., Poletto, F., Bosco, C., Patti, V., & Stranisci, M. (2018). An Italian Twitter Corpus of Hate Speech against Immigrants. *Proceedings of the 11th Conference on Language Resources and Evaluation*, 2798-2895. Miyazaki, Japan.

Schober, P., & Vetter, T. R. (2021). Logistic regression in medical research. *Anesthesia and analgesia, 132(2)*, 365. Wolters Kluwer Health.

Shushkevich, E., & Cardiff, J. (2019). *Automatic Misogyny Detection in Social Media: A Survey* (Vol. 23). Mexico City: scielomx. doi:10.13053/cys-23-4-3299

Sun, A., Lim, E.-P., & Ng, W.-K. (2002). Web Classification Using Support Vector Machine. New York, NY, USA: Association for Computing Machinery. doi:10.1145/584931.584952

Tian, Y., Zhao, C., Lu, S., & Guo, X. (2011). Multiple Classifier Combination For Recognition Of Wheat Leaf Diseases. *Intelligent Automation & Soft Computing, 17(5)*, 519-529. Taylor & Francis. doi:10.1080/10798587.2011.10643166

Warner, W., & Hirschberg, J. (2012). Detecting hate speech on the world wide web. *Proceedings of the second workshop on language in social media*, 19--26.

Waseem, Z. (2016). Are You a Racist or Am I Seeing Things? Annotator Influence on Hate Speech Detection on Twitter. *Proceedings of the First Workshop on NLP and Computational Social Science*, 138-142. Austin, Texas: Association for Computational Linguistics. Retrieved from http://aclweb.org/anthology/W16-5618

Waseem, Z., & Hovy, D. (2016). Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. *Proceedings of the NAACL Student Research Workshop*, 88-93. San Diego, California: Association for Computational Linguistics. Retrieved from http://www.aclweb.org/anthology/N16-2013

Wikipedia. (2021). Trauma Quality Improvement Program.

Wikipedia. (2022). Naive Bayes spam filtering.

Wikipedia. (2022). Natural language processing.

Yarowsky, D. (1994, June). DECISION LISTS FOR LEXICAL AMBIGUITY RESOLUTION: Application to Accent Restoration in Spanish and French. *32nd Annual Meeting of the Association for Computational Linguistics*, 88-95. Las Cruces, New Mexico, USA. doi:10.3115/981732.981745

Yekkehkhany, B., Safari, A., Homayouni, S., & Hasanlou, M. (2014). A comparison study of different kernel functions for SVM-based classification of multi-temporal polarimetry SAR data. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 40(2)*, 281. Copernicus GmbH.

Zhang, Z., & Luo, L. (2018). Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter. Retrieved from http://arxiv.org/abs/1803.03662

APPENDICES

Appendix A. Glossary

This appendix consists of short descriptions of some terms and expressions used during this document.

- **AAVE**: Acronym for African American Vernacular English, a native form of English spoken mainly by working class black people in the United States; with exclusive grammar, vocabulary and accent. An example of this is drag queen Rupaul's expression "she done already done had herses" (meaning "she already has hers").
- **CNN**: Meaning Convolutional Neural Network, these Deep Learning algorithms are designed to be able to process pixelated data, and so, are used in video recognition, image classification and medical image analysis, among other applications.
- **Emoji**: Pictograms used in digital spaces such as texting or social media in order to convey a specific, meaning, emotion or joke. An example of use of emojis would be "Wish me luck on the game! 🏈🥰"
- **Hashtag**: Word or phrase used online with the hash sign (#) in front of it. On Twitter, clicking on a hashtag takes the user to all tweets that contain the hashtag.
- **Incel:** Short for "involuntary celibate", incel refers to, usually, a young man that sees himself as unable to be intimate with women because of his physical appearance. Incels are usually hostile towards themselves and women, blaming them for not being attracted to them. This community developed online on forums such as Reddit and 4chan.
- **LSTM:** Long Short-Term Memory are a more advanced adaptation of traditional RNNs (see below). These neural networks regulate information flow better throughout the unit with the gates that form it. Its applications include, among others, speech recognition and robotic control.

- **Men's Rights Activists:** Members of the Men's Rights Movement, an anti-feminist group that discuss topics in defence of men and mainly supported by the "alt-right", a far-right white nationalist movement.
- **RNN:** Short for Recurrent Neural Network, RNNs are a type of neural network which are known for having a memory, and taking into consideration previous inputs and not just the current one. RNNs are usually utilized for tasks such as speech and handwriting recognition.

Appendix B. Discarded data

In this appendix a more thorough explanation on the discarded datasets will be given, with examples of the different categories several of the tweets fit in.

B.1 Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior dataset

In the research done in (Founta, et al., 2018), a large-scale hate speech dataset with 100 thousand tweets was annotated using the crowdsourcing platform CrowdFlower.

Since the dataset was so large, and only the tweet IDs were provided, it was necessary to extract the tweets with Tweepy. However, since execution time was long, 44.45% of the tweets were extracted and analysed.

In this dataset there is not a particular category tweets fit in, more so, the set isn't fit for this project due to an array of reason in equal significance:

- **Offensive but not hateful**
    - There's always that one idiot in the class! STFU 😒😒
    - People are dumb af in these Jurassic Park movies! Just screaming knowing damn well not to because it attracts attention!!
    - I fucking hate yall sm, jesus christ https://t.co/yULySun4mX
    - Not only is this terrible and all over the place...I hate his fucking makeup so much it's so ugly https://t.co/3BysrqAcpy
- **AAVE as hate speech**
    - RT @nyctophil3: Pineapples do not belong on pizza. Y'all niggas are nasty.
    - Niggas keep talking about women wearing weave but be sick when a bitch up a fro on they ass. 😭
    - RT @BBErika_: Hate a nigga that try and run me, "you can't wear this, you can't go out, you can't chill with them" likeeeeee is u my man or…
    - my nigga simba went from bad mon to battybwoy. what world is this??

- **Political beliefs and/or job, and not social group**
    - IM SO FUCKING PISSED!!! I HATE YOU TRUMP, I WANTED TO START WW3
    - RT @Salon: ISIS calls President Trump “an idiot who does not know what Syria or Iraq or Islam is” https://t.co/3kaUawRqN7
    - May the evil witch Thatcher rot in hell now and forever #trampthedirtdown
- **Referencing hate speech without engaging in it**
    - Line from this #film "I don't take orders from a fucking woman!" - https://t.co/Rf4P2NCJEd https://t.co/Du7TlROFtU
    - JAP Battle (EXPLICIT) - "Crazy Ex-Girlfriend" https://t.co/65vwl3oOL4 Awe Snap, Scarsdale!!! https://t.co/65vwl3oOL4
- **Normal tweets without hateful or offensive connotations**
    - I hate when people in chats are talking about shoes and they mention Jordans, then I slowly realize they aren't talking about me 🙄😂
    - What a morning!! Amazing views from Glide HQ today! We hope everyone enjoys their Monday! #MotivationMonday https://t.co/sp9mwukRwD
    - Also, last time I checked, God would rather have the LGBT community than Bryan. https://t.co/n9uFzPxGMS
    - How was my night? Well, I nearly cried watching a scene in which Paul Rudd helped a disabled teen pee off a bridge. So pretty good I'd say.
    - I'm never taking the metro to go see a dude who has a car. If he doesn't offer to come pick you up, he doesn't care about you, sis.

B.2 HASOC

FIRE[26] is an organization whose goal is to encourage research in multilingual information access in South Asian languages. In 2019 (Mandl T. a., 2019), 2020 (Mandl T. a., 2020) and 2021 they presented an evaluation task called HASOC (HAte Speech and Offensive Content detection). Their objective was to evaluate technology for finding hate speech and offensive language in online text. HASOC's available data comes in a variety of languages, English being one of them.

Each year's data is very different, so the data was analysed separately.

2019

This year's data is highlighted by the number of hateful tweets about former president of the United States, Donald Trump, who was still in office; and soon to be prime minister of the United Kingdom, Boris Johnson. These tweets are not considered hate speech, since the tweets are about the politicians' political actions and ideals.

- **Offensive but not hateful**
    - Sometimes they try but end up looking even more stupid. It's all about finances. You cannot say no to more than 60% of the revenue.
    - @nowthisnews I hope no one else hires this #Douchebag
    - fuck off imagine being 15 years old and thinking you know better than a literal doctor shut up pic.twitter.com/PWBOWX3oN8
- **Political beliefs and/or job, and not social group**
    - If you support trump you support racism, bigotry and homophobia. They are his policies. #FamiliesBelongTogether #BlackLivesMatter #NoMuslimBanEver #TransRightsAreHumanRights #gaymarriage #loveislove

[26] http://fire.irsi.res.in/fire/2022/home

- @realDonaldTrump Your fear is showing again. You tweet like the mad man you are when either things are not going your way or big info is going to come out soon. #DerangedDonald #TrumpIsATraitor
- @BorisJohnson is not a harmless buffoon. he is a cold, calculating, vile man who uses his persona to get away with his racism, homophobia and elitism. Oh and he's soon to be our new prime minster, help us all #BorisJohnsonShouldNotBePM #borisjohnson https://t.co/ZfozJi2t8d
- Men's Unisex Anti Trump AF T-Shirt in Extended Sizes - Small up to 5x https://t.co/cCONVvr7xF #fucktrump #antitrump #votebluenomatterwho https://t.co/2AUW8BhR16
- I don't know how much more I can take! 45 is a compulsive liar! #Trump30Hours #TrumpIsATraitor

- **Referencing hate speech without engaging in it**
  - Wash. Post 1'He grabbed my hand so I could feel his erection' 2'He wants to take my clothes off and have sex ... I realize, he's ejaculating." -- #MemoirProject,Ch9(Future) #SEX #WhatWomenWant? (WWW?) #WomenAttractMenAttracted #MenSeduceWomenSeduced #MenFromMarsWomenFromVenus pic.twitter.com/coh15jRj3H
  - In #Alabama.....#Rapists.... get #Parental rights. So a teen Raped by their #Rapist, can't get an #Abortion but they MUST let their #Rapist see the child. https://t.co/C2aECd6JaL #Auburn #Huntsville #Birmingham #Montgomery #Tuscaloosa #Florencesc #Dothan #gadsen #OrangeBeach

- **Normal tweets without hateful or offensive connotations**
  - i told my doctor i cant sleep and he said have sex ???? come over and fuck me yourself you coward

- **Condemning crimes**
    - Please do not follow O.J.Simpson. We don't need to give ANOTHER egomaniacal narcissist a platform...Thank you. #OJSimpson #evil #MURDERER

2020

This year's data had a mix of different invalid tweets, mainly ones about politics or tweets from people using AAVE.

- **Offensive but not hateful**
    - What a stupid idea.
    - On god—men do not understand my humor. You all are lame as fuck
    - RT @blcksiren_: You can block me, unfriend me, unfollow me But you can't uneat my ass https://t.co/x8IhSs2nxU
- **AAVE as hate speech**
    - Y'all being mad corny about the VIP info in MIA. MY NIGGAS NEED TO KNOW
    - This shit got me dead💀💀💀 but y'all wild as hell yall basically calling this nigga a child molester
    - RT @bniceloco: Niggas do the funniest shit when they're high 😭😭 https://t.co/t85BSS87JY
- **Political beliefs and/or job, and not social group**
    - @BernieSanders Bernie will never be the President of the United States of America. https://t.co/O0FPo0bTc9
    - When I tell ya I hate cops
    - Cops are fucking disgusting, you can't change my mind. https://t.co/2igqOzCB13
    - RT @sohmer: @realDonaldTrump The Importer pays the tariffs, you fucking moron. You've levied a sales tax on yours own citizens.
- **Referencing hate speech without engaging in it**
    - RT @JordanUhl: Turning Point USA's UNLV president: "We're Gonna Run the Country! White Power! F**k N*****s!" https://t.co/JaNT20nrHa

- **Normal tweets without hateful or offensive connotations**
    - RT @canzaynnot: ARE YALL SEEING THIS???? ARE YALL REALLY ????? HOSEOK PERFORMANCE IS UNTOUCHABLE HOLY SHIT https://t.co/hSfvWI0tOk
    - And if you find a Filipina, and she's fine, you better keep her. Cuz there ain't nothing like a Filipina girl. 😜
    - WHAT THE FUCK #GameOfThrones

2021

This year's data contained a large number of tweets talking about the poor management in India during the late 2010's pandemic of COVID-19. Much like 2019's data, these tweets are not hate speech, as they are complaining about the Indian government and politicians, and not about a group of marginalized persons.

- **Offensive but not hateful**
    - Narcissists piss me the fuck off!!
- **Political beliefs and/or job, and not social group**
    - Why Modi as Prime Minister is big Disaster for India ? * Daily 400000 case. * Daily 4000 deaths. But he is still working on his image! #ResignModi #MakeGadkariPM https://t.co/JBEtYvbMvm
    - People dying without any medical treatment and oxygen is happening only in India. Shame #ModiKaVaccineJumla
    - @OpIndia_com Even in pandemics, shameless vultures..... nothing except #Islamophobia
    - @im_seerat What a pity there are some criminals who do not pay for their crimes, right? #Islamophobia is a major concern in present times and it all started with bollywood as we eat and laugh watching them dedmae Mulsims as community and fram them terrorist?
    - I'm literally crying. The government is not only NOT doing anything to help, now they are actively helping is MAKING THINGS WORST. IT SHOULD BE ILLEGAL TO BE THIS STUPID?! #Mangaluru #IndiaNeedsOxygen #IndiaCovidCrisis #IndiaFightsCOVID19 https://t.co/w6FlBhmbVR

Appendix C. Bias in NLP

In this appendix I'll explain what I believe is a very important point on data annotation. In the figure below we can see the sex, sexual orientation and ethnicity of the annotators of the ToLD-Br dataset seen previously. As we can see, the main ethnicity is white and the main sexual orientation is heterosexual. This means that there is a bias in this dataset. This doesn't mean that the data is not trustworthy, it simply means that it's necessary to take this into consideration when working with manually labelled data.

	Categories	# annotators
Sex	Male	18
	Female	24
Sexual orientation	Heterosexual	22
	Bisexual	12
	Homosexual	5
	Pansexual	3
Ethnicity	White	25
	Brown	9
	Black	5
	Asian	2
	Non-Declared	1

Figure 43. ToLD-Br annotator demographic (Leite, Silva, Bontcheva, & Scarton, 2020)

Even though not all of the used datasets have specific information about the annotators, It's safe to say the data would still have biases.

(Blodgett, Barocas, III, & Wallach, 2020) presented a paper on NLP bias where they reference many articles that present some kind of bias and analyse them to explain the possible flaws in techniques used, and finally propose recommendations to researchers to not make these mistakes (the used datasets used in this project are not cited in this piece).

Appendix D. N-grams and k-fold cross validation

During the presentation and comparison of previous studies with similar objectives and methods to this project, two concepts used in Machine Learning and NLP were mentioned: n-grams and k-fold cross validation. This appendix is dedicated to explaining them.

N-grams

An n-gram is a sequence of n elements that are adjoined in a text sample. N-grams of size 1 are called unigrams, those of size 2 are bigrams, and so on. When using one of Sklearn's vectorizers, adjusting the parameter "ngram_range" means deciding what becomes a token, and the parameter "analyzer" chooses whether features are made of word n-grams or character n-grams. Since the default is set to only word unigrams, each word is tokenized separately. However, if, for word n-grams, the range is set to (1,2), that means every unigram and bigram is a different token. To predict the probability of finding certain n-grams in any sequence of words, n-gram language models are used.

For example, in the sentence "Romanes eunt domus", there are three unigrams: "romanes", "eunt" and "domus". In the same sentence there are two bigrams: "romanes eunt" and "eunt domus".

K-fold cross validation

Just like the train/test split method, k-fold cross validation is a process used to evaluate Machine Learning models. Here, the data is split into k equally-sized samples. One of the samples is saved for testing, while the rest of the data is used for training the model. This is repeated k-1 more times, using a different sample for testing in each iteration. The k obtained results are then averaged to obtain the final output. K-fold cross validation ensures that every part of the data is used for both training and testing, and can lead to more balanced results.