Abusive Language Detection Against Immigrants and Women

[Introduction 3](#_Toc37069966)

[Project Aims 3](#_Toc37069967)

[Intended Audience 3](#_Toc37069968)

[Project Scope 3](#_Toc37069969)

[Approach 3](#_Toc37069970)

[Assumptions 4](#_Toc37069971)

[Summary of Outcomes 4](#_Toc37069972)

[Background 5](#_Toc37069973)

[Project Context 5](#_Toc37069974)

[Identified Problem 5](#_Toc37069975)

[Likely Stakeholders 5](#_Toc37069976)

[Theory Associated with Problem Area 6](#_Toc37069977)

[Constraints on Approach 6](#_Toc37069978)

[Existing Solutions 6](#_Toc37069979)

[Methods and Tools 6](#_Toc37069980)

[Specification and Design 8](#_Toc37069981)

Introduction

# Project Aims

The aim of this project is to design and build the most effective classifier of offensive tweets towards the target groups of immigrants and women as possible. The project is based on a similar project run online, and as such I will be comparing my work and results to the results of other participants of the competition. As a result of there being other people working with the same data set that I have used for this project, one goal of the project was to beat the highest score of the competition, through a thorough understanding of natural language processing, and experimentation with different forms of classifiers, as well as neural networks.

# Intended Audience

The intended audience of this project is anyone interested in natural language processing, but also anyone interested in the practical application of offensive speech detection online. This is especially beneficial in the current world, given the rise of hate speech online on popular platforms such as Twitter. While this project was developed and tested on a specific set of tweets, with some modification it could be used to classify tweets in real time. Therefore, I would argue that the techniques and models explored in this project would be beneficiary to anyone looking at implementing an automated hate speech detector into their website.

# Project Scope

The scope of this project adapted over time, due to the fact that at the start of the project the libraries and concepts I was going to be working with were largely unknown to me. Therefore, it was hard to estimate the amount of time it would take me to complete pieces of work.

# Approach

The way that I approached carrying out the project was through regular meetings with my supervisor, coupled with a Kanban approach to working. This meant that I was focussed on small chunks of deliverable work, that I could then discuss with my supervisor and plan out what to work on next. This also meant that the scope was able to remain flexible throughout the project, and I was able to pivot quickly in the event that circumstances changed, which they did at some parts of the project.

The way in which I ran the experiments that lead to my outcomes changed throughout my project as the data available to me changed, and as such can be split into two methods. Initially I did not have access to the test set that others who attempted this problem had, and as such I combined the two data sets I did have (train set and dev set), and then split this one set into multiple splits using a method known as KFold splits. I was able to get access to the third data set during my project, and once I had this I approached the project in the following way, which is considered best practice:

* Implement a method of converting the corpus into features.
* Select a classifier model.
* Provide the model with the train set and features.
* Predict whether the tweets in the dev set were offensive or not.
* Compare these predictions to actual results.
* Record results.
* Tweak the classifier, parameters, and features.
* Repeat

Once I had done this to a point where I was happy with each of the different

methods I had implemented, I then generated predictions for the test set, and compared these predictions to the actual results. These scores were then used for my final results.

# Assumptions

[INSERT ASSUMPTIONS HERE]

# Summary of Outcomes

In terms of the important outcomes from the project, there were a number of interesting outcomes, as follow:

* Word embeddings based on Word2Vec were the most effective way of determining features for the models to use.
* SVC was the most effective model at classifying tweets from the corpus accurately.

Background

# Project Context

The wider context of the problem of detection of abusive language towards women and immigrants is clear, as the rise of social media platforms and the anonymity that they provide has precipitated a large amount of online abuse, with one survey suggesting 18% of teenagers experienced abusive language while communicating online[[1]](#footnote-1). Given the frequency of abusive language use online, it is clear there is need for specific detection of abusive language.

The main aim of this project is focussed around research and analysis into how best to approach the design and implementation of a program for detection of abusive language towards women and immigrants. As a result of this, my project has a lot of varying methods of essentially solving the same problem, with varying levels of success. I have been able to identify which methods and models work best together to produce the best results, thus informing how best to solve the problem.

A secondary aim is to produce a piece of software that given a tweet via the command line, will be able to accurately predict if the tweet is offensive or not towards the target groups. This was successfully developed using the research outcomes I found from the main aim of the project.

# Identified Problem

As identified in my initial plan, the pervasive nature of abuse towards women and immigrants’ online shows that this is a problem that needs solving. Although the focus of this project was experimentation and analysis, the outcomes and findings are useful in determining the best approach to developing an accurate automated method for detection of abuse towards women and immigrants. The findings of this project could be quickly implemented to create a robust automated abusive language detection system.

# Likely Stakeholders

In terms of likely stakeholders in the problem area, those that would benefit from developing and implementing an automated system for the detection of abuse language would be social media sites and law enforcement agencies. There is a vested interest among social media site owners to stop and remove abusive language on their platforms, in order to make their platform more welcoming and nicer to use, as end users will stop using the site if they are receiving abuse. As well as this, some of the abusive language that could be posted online could constitute hate speech, which is a crime in the United Kingdom, and therefore it would be beneficial to law enforcement agencies if they could automatically detect the abusive language, as they would then be able to quickly punish those posting the hate speech.

# Theory Associated with Problem Area

[Clarify this] Should I discuss the way in which my development approach was not initially best practice? And the theory behind why best practice (train on train, test on dev until finished, then test on test for final results) is best practice? Potentially include things about KFold and stratified KFold? I’m not really sure what to put in this section.

# Constraints on Approach

As stated in my introduction, I was originally constrained from approaching the project in what is considered the best practice method for developing a models to classify offensive tweets due to not having access to the test data set that is used to generate final results, and ultimately compare these results to other teams who have worked on the same problem. This constraint was solved halfway through the project, and I was able to work in the best practice approach for the remainder of the project.

Due to this constraint, I did have to adopt an alternative approach for the initial period of my work on this project, involving using KFold splits to alleviate the issue created by not having a test set. This proved to be a viable approach to the project, as the results and outcomes generated during this period of the project were essentially similar to when I approached the project with best practice in mind. The trends methods I found to get the best results were the same and could be adopted when I was given access to the test set.

# Existing Solutions

[Clarify this] What should go in this section?

# Methods and Tools

My project was developed using Python 3.7.6, and in particular I made use of 3 specific libraries to complete the project: Sci-Kit Learn, Gensim, and Numpy. In terms of what makes up my program, each of my models is made up of a form of classifier (For example, a decision tree of Logistic Regression), and a form of feature extraction that helps to train the model with features that are deemed as important markers for if a tweet is offensive or not.

Sci-Kit Learn is a library that I used to give me access to a wide number of different classifier models that I could train my corpus on, and use to generate predictions for if a tweet is offensive or not. As well as this, at once stage I also used this library to extract features from the corpus as a further method of training the classifier model.

Gensim is a library that allowed me to implement a form of neural networks into the project, using a model of feature extraction called Word2Vec. Word2Vec works by creating word embeddings for each word in the corpus, and then looking at what words are “near” that word, to determine links. I implemented two forms of feature extraction in my project, with one Word2Vec model that was trained on my data set, and another Word2Vec model that was trained on a much larger data set of tweets. Both of these models were used in my work, and both produced varying results. I also used Word2Vec to analyse what words were “nearest” to certain words, to see what words are commonly used along with what could typically be considered as abusive language.

Numpy is a library that extends the operations that Python can perform on data structures, particularly arrays and matrices. This is useful for my project given that Sci-Kit Learn makes extensive use of sparse matrices to represent the predictions output by the classifier models, and by using Numpy it is much easier to record these results.

Specification and Design

# Introduction

Given the research based nature of this project, the design of the system has been adapted and evolved a number of times to suit the improvements I have been able to implement throughout the process of the project. The structure of the program used throughout the project has gone from monolithic, to a much more modular design which allowed me to make use of varying methods of feature extraction and classifier models.

# Data Flow

The data flow of the program is as follow:

[diagram of off not text files -> program -> predictions -> results.csv]

1. <https://www.researchgate.net/publication/320914119_Analysis_of_foul_language_usage_in_social_media_text_conversation> [↑](#footnote-ref-1)