UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DESIGN AND DEVELOPMENT OF A SYSTEM FOR DETECTING CYBERBULLYING IN TWITTER BASED ON MACHINE LEARNING TECHNIQUES

JAIME JOSÉ PALOS PEREIRA

2018

TRABAJO FIN DE GRADO

Título:	Diseño y Desarrollo de un Sistema de Detección de Ciberacoso en Twitter basado en Técnicas de Aprendizaje Automático.
Título (inglés):	Design and development of a System for Detecting Cyberbullying in Twitter based on Machine Learning Techniques
Autor:	Jaime José Palos Pereira
Tutor:	Carlos A. Iglesias Fernández
Departamento:	Ingeniería de Sistemas Telemáticos

MIEMBROS DEL TRIBUNAL CALIFICADOR

Presidente:

Vocal:

Secretario:

Suplente:

FECHA DE LECTURA:

CALIFICACIÓN:

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE
INGENIEROS DE TELECOMUNICACIÓN

Departamento de Ingeniería de Sistemas Telemáticos
Grupo de Sistemas Inteligentes

TRABAJO FIN DE GRADO

DESIGN AND DEVELOPMENT OF A SYSTEM FOR DETECTING CYBERBULLYING IN TWITTER BASED ON MACHINE LEARNING TECHNIQUES

Jaime José Palos Pereira

Enero de 2018

Resumen

Internet ha cambiado por completo el orden de la sociedad, la manera de trabajar, de aprender y, de forma radical, la manera de comunicarnos. Y a pesar de las innumerables ventajas que esto aporta, este trabajo va a centrarse en una de las principales desventajas que acarrea como consecuencia de la impunidad y el anonimato de esta nueva forma de comunicación.

Según el estudio Global Youth Online Behavior Survey desarrollado por Microsoft, ya en 2012, uno de cada tres jóvenes españoles sufrían ciberacoso [37] y, en 2016, España se convirtió en uno de los países donde más ciberacoso sufrían los menores, en especial los adolescentes de 13 años, según un informe de la Organización Mundial de la Salud (OMS), que alerta del riesgo de depresión y suicidio como consecuencia del llamado "Cyberbulling", siendo este definido como el uso de medios de comunicación digitales para acosar a una persona o grupo de personas, mediante ataques personales, divulgación de información confidencial o falsa entre otros medios [20].

El ciberacoso es uno de los problemas a destacar en las nuevas generaciones, siendo tal su importancia, que durante el mes de junio de 2017, el Instituto Nacional de Ciberseguridad (INCIBE) puso en marcha una línea directa de ayuda telefónica, gratuita y confidencial, destinada a resolver problemas sobre el uso de Internet. El objetivo de este proyecto es ayudar a acabar con este problema social de una manera rápida, automática y eficiente, sin la necesidad de esperar que un niño cometa el acto de valentía de denunciar a sus acosadores.

En este proyecto se analizará este tipo de comportamiento y lenguaje empleado, centrándose en lenguaje ofensivo, hostil o agresivo que se pueda caracterizar como ciberacoso y la fuente de información escogida es Twitter. El estudio se centrará además en el análisis del ciberacoso de carácter sexual, ya que según la organización Internet Safety 101 [1], 1 de cada 7 niños recibe una invitación sexual a través de internet. Este análisis se basa en el tipo de lenguaje, estructuras gramaticales y palabras o insultos utilizados por ciberacosadores en las redes sociales. La principal tarea es el desarrollo de un sistema clasificador que permita predecir si un mensaje de Twitter es ofensivo u hostil, posee connotaciones sexuales o encaja dentro del perfil de un posible depredador sexual y puede

ser clasificado como ciberacoso. Para ello, se ha escogido el lenguaje de programación Python usando herramientas de aprendizaje automático como Scikit-learn con técnicas de Lenguaje Automático Supervisado y herramientas de Procesado de Lenguaje Natural (en inglés Natural Language Process (NLP)). Para la prueba y evaluación de este proyecto se utilizarán distintos modelos y algoritmos, analizando posteriormente sus resultados para escoger aquel con mayor precisión. Como fuente de datos se ha escogido un conjunto de Tweets organizados por usuarios en inglés, así como datos sacados de Chatcoder [16] y el conjunto de datos dados en la competición PAN12 [32]. Finalmente la implementación del sistema como un servicio se llevará a cabo mediante la creación de un plugin en la plataforma Senpy que nos permite dicha implementación.

Palabras clave: Aprendizaje Automático, Python, Twitter, Senpy, Predador Sexual, Ciberacoso.

Abstract

Internet has completely changed our society, the way we work, the way we learn and, in a radical way, the way we communicate. Today we live in a world in which face to face conversations have been replaced by 140 character statements. And despite the great benefits all these changes implies, this project is going to focus in one of the main disadvantages isolating us due to the impunity and anonymity that comes with this new way of communication.

According to the Global Youth Online Behavior Survey developed by Microsoft in 2012, already one of every three young Spaniards were suffering cyberbullying [37] and, in 2016, Spain became one of the top countries where children were suffering cyberbullying, especially 13 year olds according to a report from the World Health Organization (WHO), that alerts of the great risk of depression and suicide as a consequence of "Cyberbullying", being defined as the use of electronic communication to bully a person or group of people, typically by sending messages of an intimidating or threatening nature or disclosure of confidential or fake information [20].

For all of his, cyberbullying is nowadays one of the main problems among new generations, to the point of, in June, 2017, the Instituto Nacional de Ciberseguridad (INCIBE) initiated a direct help-line destined to solve problems regarding the use of internet. This free and confidential service is destined to children and teenagers worried by any aspect regarding internet. The goal of this project is to end with this social problem in a quick, automatic and efficient way, without needing to wait for a child to commit the act of valor of reporting their stalker or bully.

This kind of behavior will be studied in this project focusing in offensive, aggressive or hostile language that could be characterized as Cyberbullying and the source of information chosen is Twitter. This study will mainly focus on the detailed analysis of sexual predatory behaviors, since as Internet Safety 101 organization says, one out of every seven children receives a sexual invitation throughout Internet [1]. This analysis is based on words or insults and grammatical structures used by cyber-bullies in social networks. The main idea is to develop a classification system that is able to predict whether a Tweet is offensive

or hostile, contains sexual connotations or fits within a sexual predator's profile, and can be classified as cyberbullying. To accomplish this, the programming language chosen is Python, using automatic-learning tools such as Scikit-learn with supervised machine learning techniques Natural Language Process (NLP) tools. To test and evaluate this project different models will be used, analyzing later their results to choose the more accurate one. A data set of tweets in English has been chosen as the source of data as well as data from de Chatcoder [16] and the dataset given in PAN12 competition [32]. Finally the system's implementation as a service will be done by creating a plugin in the platform Senpy, which allows us this implementation.

Keywords: Machine Learning, Python, Twitter, Senpy, Sexual Predator, Cyberbullying

Agradecimientos

"Ninguno de nosotros es tan inteligente como todos nosotros." – Ken Blanchard

En este apartado quería dar las gracias por todo el apoyo y ayuda recibidos a lo largo de toda mi vida universitaria, a todas esas personas que han confiado en mi y que han estado ahí conmigo durante este recorrido y han hecho amena y posible esta etapa.

Gracias a mis padres por hacer posible que yo pudiera realizar estos estudios, a mis hermanos por ayudarme en los momentos más difíciles, incluso sin ellos saberlo, y a Marta por aguantar estos años con todo lo que ha conllevado, por todo el apoyo y ayuda y sobretodo, por siempre creer en mí.

A todas esas personas que han estado ahí en los momentos buenos y en los más difíciles, que han pasado conmigo tantos días (y noches) en la biblioteca estudiando, y también han celebrado conmigo cada fin de exámenes y cada aprobado. Afortunadamente son demasiados para nombrarlos a todos, ellos saben quienes son.

A todos los miembros de Eurielec y EESTEC por toda la ayuda, explicaciones, apuntes, etc. recibidos y por todos esos momentos de diversión y desconexión fuera de la universidad.

A todos los compañeros de trabajo de las prácticas realizadas, por ayudarme a crecer profesionalmente.

Agradecer también a mis compañeros del GSI que han sido siempre amables y siempre dispuestos a echar una mano.

Finalmente agradecer a mi tutor Carlos A. Iglesias toda su ayuda y apoyo durante, no sólamente el desarrollo de este proyecto, si no desde aquella ADSW en segundo. Gracias a él he aprendido muchas cosas nuevas e interesantes y me ha prestado una ayuda fundamental para el desarrollo y finalización de este proyecto.

Contents

List of Figures

List of Tables

List of Acronyms

AI Artificial Intelligence

API Application Programming Interface

CSV Coma Separated Value

EU European Union

FOAF Friend of a Friend

FP False Positive

FN False Negative

IBK Instance Based Algorithm

INCIBE Instituto Nacional de Ciberseguridad

LDA Latent Dirichlet Allocation

LIBSVM Library for Support Vector Machines

LM Language Model

ML Machine Learning

NLP Natural Language Process

NLTK Natural Language Toolkit

NN Neural Networks

OMS Organización Mundial de la Salud

POS Part of Speech

REST Representational State Transfer

SCI Suspicious Conversations Identification

SMO Sequential Minimal Optimization

SVC Support Vector Classification

SVM Support Vector Machines

TP True Positive

TN True Negative

VFP Victim From Predator

WHO World Health Organization

XML Extensible Markup Language

CHAPTER 1

Introduction

1.1 Context

As Statista adequately points out "One of the defining phenomena of the present times reshaping the world as we know it, is the worldwide accessibility to the internet. The lovechild of the World Wide Web is social media, which comes in many forms, including blogs, forums, business networks, photo-sharing platforms, social gaming, microblogs, chat apps, and last but not least, social networks. The power of social networking is such that, the number of worldwide users is expected to reach some 2.95 billion by 2020, around a third of Earth's entire population" [45]. One of the social networks with more users is Twitter, all of these users uploading 140 characters tweets generates a huge amount of data that can be analyzed to many different purposes.

In this project we will use all this data to try to find tweets containing any sexual content or predatory activity. As we have seen previously, the most cyber-bullied are teenagers, and according to a study done by Statista as of the third quarter of 2014, 30% of Twitter users are under 25 years old, which made it the fourth most used social media among this public just behind Youtube, Instagram and Tumblr [44] which makes it one of the perfect social networks to find this kind of content.

This is an interesting task because this is usually done manually or using not very accurate techniques and it can save a lot of people a lot of time and resources and it only depends on the data you receive, so eventually it could be implemented on many different social networks, blog sites, on-line chats, etc.

To accomplish this task, this project is firstly going to analyze a set of conversations gathered at PAN12 and Chatcoder and then we will analyze a set of data previously tagged as containing cyberbullying, or sexual content, their keywords and sentences and learn in order to predict cyberbullying in new untagged data.

1.2 Project goals

The amount of data available in Twitter is huge and, for this, it sometimes is very difficult to process all of it in order to obtain valuable data and many cyber-predators pass unnoticed. This project's goal is to show the benefits of Machine Learning in processing big amounts of data and NLP to identify and separate valuable from useless information, detecting cyber-predators.

More precisely the goal is to see if it is possible to create an accurate enough system to detect cyberbullying, focused on sexual predators, using machine learning and NLP tools.

Another goal would be to see if it would be possible and/or useful to implement it on Twitter.

And to finalize another project's goal would be to implement a Senpy plugin so we can operate it as a service which allows us to use it interchangeably.

1.3 Task Methodology

This document can be branched into six main tasks that need to be accomplished for a correct development of the project:

- Correct election of a useful dataset.
- Correct selection, organization and preprocessing of the data provided in the datasets so it can be correctly handled.
- Implementation of a cyberbullying detecting system, targeted on sexual harassers, following the next steps:

1. Creation of a balanced training and a testing dataset from the original one in order to train and test the classifier.
2. Selection, definition and extraction of the different features to be analyzed by the classifier.
3. Testing the classifier using various algorithms to use the one with higher accuracy.
4. Testing the classifier using various datasets to ensure it works fine.
5. Electing the algorithm with best results, or that best fits our solution.

- Trying out the classifier with different datasets to improve its performance and efficiency.
- Development of a Senpy plugin where our system will be deployed so we can use it as a real application making real-time predictions.

1.4 Structure of this document

In this section we provide a brief overview of the chapters included in this document. The structure would be the following:

Chapter 1 explains the context and motivations of why this project is developed. Furthermore, it describes the main goals of this project.
Chapter 2 describes the main technologies on which this project relies and uses. Specifically Machine Learning and some Python libraries such us Scikit-Learn, Pandas or Natural Language Toolkit (NLTK) and it will also include a glimpse into Senpy.
Chapter 3 includes a case study, or description of our dataset and analyzes previous related work developed on the subject.
Chapter 4 illustrates the architecture of this project, including the design and implementation phase. It provides a general description of the classifier construction and evaluation according to the metrics later defined.
Chapter 5 presents the Senpy plugin developed for this project.
Chapter 6 discusses the conclusions drawn from this project, achieved goals, problems faced and suggestions for a future work.

CHAPTER 2

Enabling Technologies

2.1 Introduction

In this chapter, we are going to give an insight into the techniques and technologies used in this project.

This project is based on the Artificial Intelligence (AI) specialty of Machine Learning (ML) [39]. Without being hard coded, it gives the machine the power to learn. It begins with sample input. With this, ML creates model algorithms. These algorithms can learn from data, and then make predictions on similar data. This way, ML can make decisions or predictions, rather than follow program instructions strictly. In other words, ML tries to put the human process of making decisions into code.

There are three things which must be fulfilled to use Machine Learning for making a decision for a problem. First, there must be a pattern in the input data for the algorithm to understand and form an opinion about. Second, there must be a large amount of input data. Third, because a human cannot create a mathematical formula which describes and classifies a problem, Machine Learning is used for two things: to "understand" the data, and to "learn" in a structured way to get to a mathematical approximation that describes the way a problem behaves. Without these three conditions, it would not be correct to

attempt to use Machine Learning.

Figure 2.1: Machine Learning Stages [8]

As shown in Figure 2.1 there are three components in Machine Learning [24]. The first part is the input data. The last component is the output data. The middle component, the Machine Learning algorithm, is the key part. This part can be one of three types: Reinforcement Learning, Unsupervised Learning, and Supervised Learning. [8]

2.1.1 Reinforcement Learning

Reinforcement learning permits machines and software agents to automatically decide the best behavior within a specific context, to maximize its performance. Simple reward feedback is required for the agent to learn from its behavior; this is known as the reinforcement signal. "Reinforcement Learning is concerned with how an agent ought to take actions in an environment so as to maximize some notion of long-term reward [...] Reinforcement learning differs from standard supervised learning in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected" [13].

Reinforcement Learning algorithms attempt to find a policy that maps states of the world to the actions the agent ought to take in those states. Reinforcement Learning is used where the data is not labeled or grouped by a property, but there is some type of feedback available for the predictive steps or actions. This is the AI we usually see in movies such as the famous HAL900 in 2001: A Space Odyssey (Figure 2.2a). To my generation the most well-known example of this type of Machine Learning could be Syndrome's Omnidroid 01 through Omnidroid 10 in Disney's movie The Incredibles (Figure 2.2b).

(a) HAL900 [12] (b) Omnidroid v10 [21]

Figure 2.2: AI in movies

2.1.2 Unsupervised Learning

Unsupervised learning is useful in cases where the data is not labeled or grouped by a property, and the algorithm must deduce or imply a relationship as explained in Figure 2.3. Unsupervised Learning can be a goal: for example, when it must discover a hidden pattern in data. Or it can be used to discover the representations needed to detect features or classifications from input data: then Unsupervised Learning is a means to an end [39].

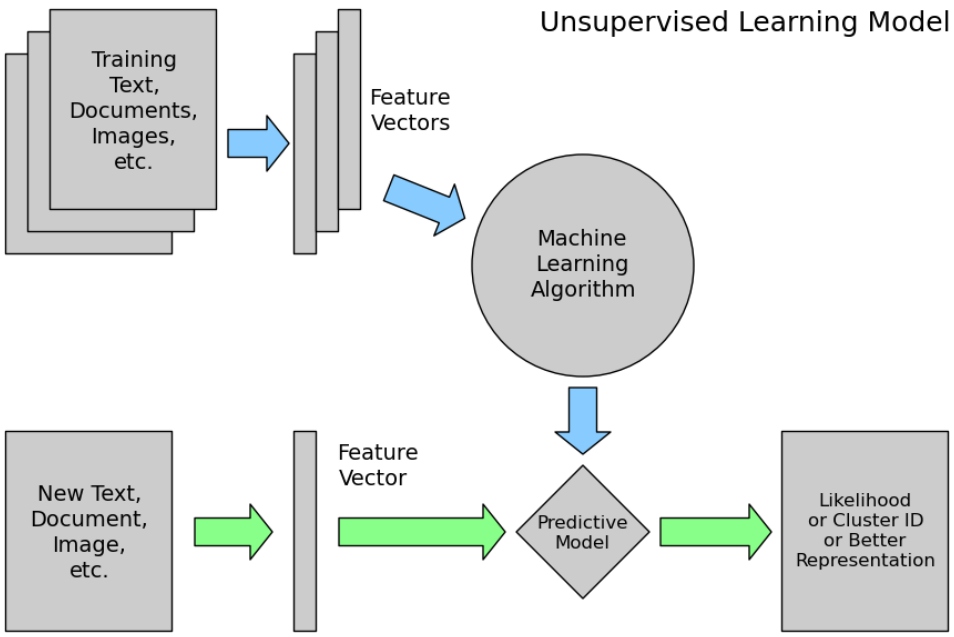


Figure 2.3: Unsupervised Machine Learning [19]

2.1.3 Supervised Learning

Supervised learning is used for cases where a label or property is available for the set of data that is going to be used for "training" the algorithm (Figure 2.4) [39, 19].

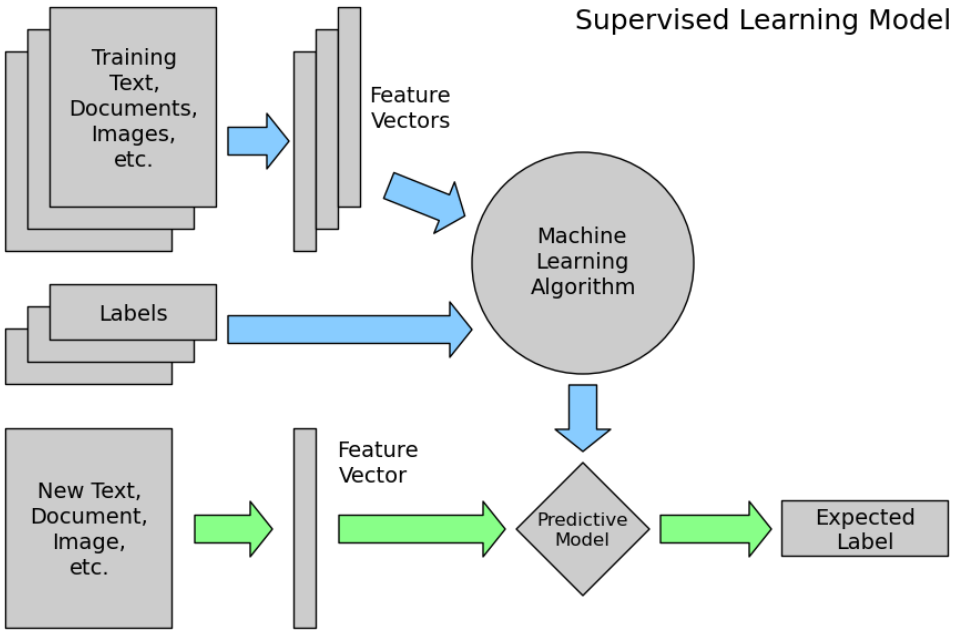


Figure 2.4: Supervised Machine Learning [19]

For Supervised Learning, algorithms can be of various types. Here we see some of the most popular ones and used in our project:

- ***Decision Tree:*** Decision Tree algorithms permit approaching the problem in a systematic and structured manner and arrive at a logical conclusion because the decisions have Yes/No responses. We see an example for "The Iris Dataset" [40] in Scikit-Learn [34] in Figure 2.5.

- ***Ordinary Least Squares Regression:*** Ordinary Least Squares Regression employs the least squares linear regression method to perform linear regression. "Linear" because the data is fitted to the linear regression model, and "least squares" because the data is being minimized over this kind of error measurement [26], as shown in Figure 2.6.

- ***Logistic Regression:*** Logistic Regression measures the relationship between a variable dependent on a category and one or more independent variables. This one is usually used for credit scoring, and predicting the probability of earthquakes on par-

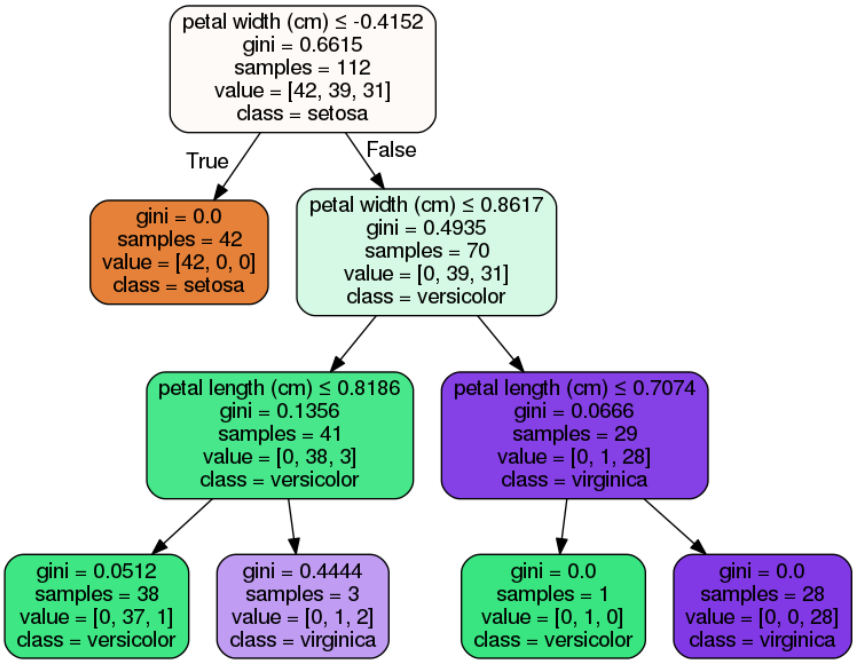


Figure 2.5: Decission Tree for Pythons Iris dataset

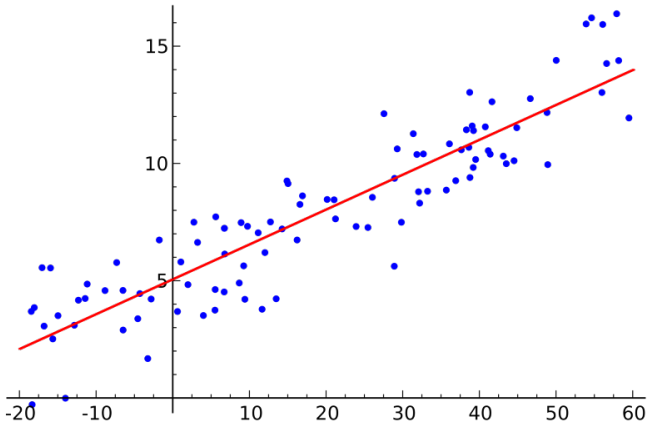


Figure 2.6: Least Square Regression [26]

ticular days [26] (Figure 2.7).

- ***Support Vector Machines (SVM):*** SVM is a binary classification algorithm. It works with a set of points of 2 types, and it separates them into 2 groups. It was used to solve display advertising and gender recognition based on images [26] (Figure 2.8).

- ***Naïve Bayes Classification:*** It is usually used to classify e-mail as spam or non-spam, news articles into sports/technology/politics/etc., and to check texts for expressing emotions as positive or negative emotions. The formula for this is based on

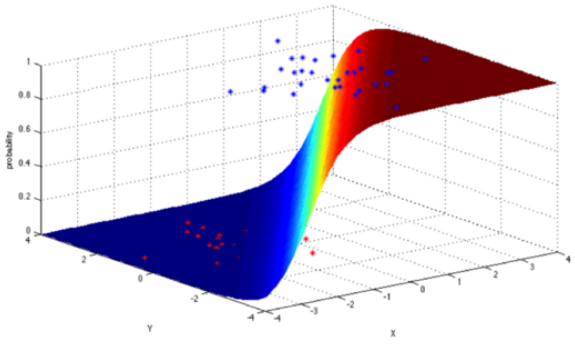


Figure 2.7: Logistic Regression [26]

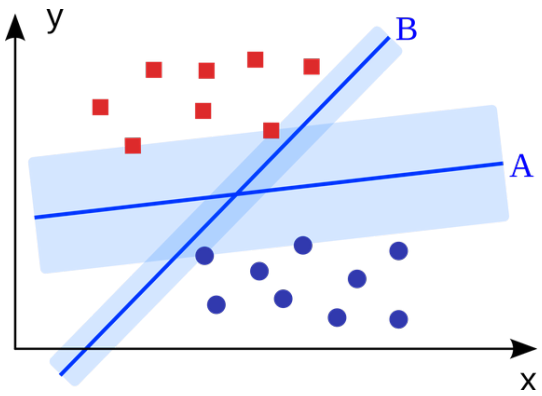


Figure 2.8: Support Vector Machines [26]

Bayes' Theorem. It makes groups of simple classified probabilities based on applying the theorem with very non-sophisticated ("naïve") independent suppositions between the labels or properties [26]. The equation's values are shown in Equation 2.1.

P(A—B) = posterior probability

P(B—A) = chance

P(A) = class prior probability

P(B) = predictor prior probability.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.1}$$

2.2 Scikit-Learn

The SciPy Stack is a small core of open source software packages (listed below), named The SciPy Ecosystem, for computing in Python [42].

Extensions, or modules, for SciPy are named SciKits. The module that provides learning algorithms is named Scikit-learn [34].

Scikit-learn provides a variety of unsupervised and supervised learning algorithms through a consistent interface in Python. It is an open-source library for data analysis and data mining [6].

SciPy Stack's small core of packages are the following:

- ***Python:*** a general-purpose programming language, very good for interactive work and quick prototyping, and yet powerful enough to write code for large applications.
- ***NumPy:*** the fundamental package for numerical computation. It defines the numerical array and matrix types, and does basic operations on them.
- ***The SciPy Library:*** a group of numerical algorithms and toolboxes which are specific to a domain. It includes, among many others, signal processing, optimization, and statistics.
- ***Matplotlib:*** a popular and mature plotting package which generates 2D plotting with the quality for publications, and simple quality 3D plotting.

Based on these packages, the SciPy Ecosystem has some general, and some specialized, tools for managing data and computation, for productive experimentation, and for high-performance computing. Although there are many more, here are some key packages used in this project from the SciPy Ecosystem [42].

- Data and computation packages:
  - ***Pandas:*** provides high-performance data structures which are easy to use.
  - ***SciKit-Learn:*** a collection of algorithms and tools for machine learning.
- Productivity and high-performance computing packages:
  - ***The Jupyter Notebok:*** provides iPython functionality and other items on a web browser, for easy documentation of computations in a easily reproducible form.

Scikit-learn has a uniform interface for all the estimators, some methods are only available if the estimator is supervised or unsupervised, while others are available for both types of estimators [19]:

- Available in all estimators:
  - ***model.fit():*** fit training data. For supervised learning applications, this accepts two arguments: the data X and the labels y (e.g. model.fit(X, y)). For unsupervised learning applications, this accepts only a single argument, the data X (e.g. model.fit(X)).
- Available in supervised estimators:
  - ***model.transform():*** given a trained model, predict the label of a new set of data. This method accepts one argument, the new data X_new (e.g. model.predict(X_new)), and returns the learned label for each object in the array.
  - ***model.predict_proba():*** For classification problems, some estimators also provide this method, which returns the probability that a new observation has each categorical label. In this case, the label with the highest probability is returned by model.predict().
- Available in unsupervised estimators:
  - ***model.transform():*** given an unsupervised model, transform new data into the new basis. This also accepts one argument X_new, and returns the new representation of the data based on the unsupervised model.
  - ***model.fit_transform():*** some estimators implement this method, which performs a fit and a transform on the same input data.

2.3 Pandas

Pandas [33] is a Python library that provides easy-to-use data structures and data analysis tools. Pandas is built on top of NumPy so it is usually necessary to import both, but since it is in Python we don't have the need to move to a more specific analytic language such as R.

The main advantage of Pandas is that provides extensive facilities for grouping, merging and querying pandas data structures, and also includes facilities for time series analysis, as well as i/o and visualization facilities.

Pandas' two main data structures are:

- ***Series:*** is a one dimensional labelled object, capable of holding any data type (integers, strings, floating point numbers, Python objects, etc.).. It is similar to an array,

a list, a dictionary or a column in a table. Every value in a Series object has an index.

- ***Data Frames:*** is a two dimensional labelled object with columns of potentially different types. It is similar to a database table, or a spreadsheet. It can be seen as a dictionary of Series that share the same index.

In this project, Pandas has been used for various purposes such as:

- Read Coma Separated Value (CSV) files after being converted from XML using Python.
- Analyzing the acquired Data Frame.
- Preprocessing the data, handling missing data, outliers, etc.
- Viewing data and results.

2.4 NLP and TextBlob

Natural Language Process (NLP) is the specialty of the crossing of AI, with computer science, and with linguistics. Its objective is to make the principal structure of language accessible for analysis and manipulation with computer programs.

TextBlob [29] is a library for Python for processing textual data. NLP tasks are taken up by TextBlob, which gives a consistent API. TextBlob does NLP tasks like part-of-speech tagging, extraction of noun phrases, analysis of sentiments, etc. TextBlob is based on NLTK and pattern libraries. NLTK provides easy-to-use interfaces to corpora and lexical resources, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, semantic reasoning, and wrappers for industrial-strength NLP libraries. The "pattern" module contains a fast part-of-speech tagger (identifies nouns, adjectives, verbs, etc. in a sentence), sentiment analysis, tools for verb conjugation, classification (Naive Bayes, Decision Tree), and a WordNet integration.

2.5 Twitter API

Twitter's developer platform offers several tools and Application Programming Interfaces (APIs). The Twitter API provides programmatic access to read and write Twitter data. It is based on a Representational State Transfer (REST) architecture allowing the system to

access and manipulate textual representations of Twitter web resources using a predefined set of stateless operations [46]. The Twitter API has several methods, which return a timeline of a user's tweet data.

In this project we have registered as a developer in Twitter in order to get the necessary credentials and the API has been used to collect tweets for our second dataset as we will see in Section 4.7, as well as for our Senpy's plugin in Chapter 5. Since we only found tweets' ids and authors' ids from Twitter containing cyberbullying and/or sexual predatory content. We used Twitter's API to assemble all those tweets to be later analyzed.

2.6 Senpy

Senpy is a framework for sentiment and emotion analysis services developed at GSI ETSIT UPM [18]. Senpy lets you analyze sentiment and emotions from a text input through a web interface, Senpy also accepts request using its simple API. Services built with Senpy are interchangeable and easy to use because they share a common API [47].

The framework consists of two main modules: Senpy core, which is the building block of the service, and Senpy plugins, which consist of the analysis algorithm. The Figure 2.9 depicts a simplified version of the processes involved in an analysis with the Senpy framework [47].

Despite our project not being related to sentiment analysis we used two Senpy's plugins as we will explain in subsection 4.3.4 to get our author's age and gender.

Lastly we used Senpy for the development of a plugin where we can deploy our system as a service with a web user interface. This way we can test and share our system easily.

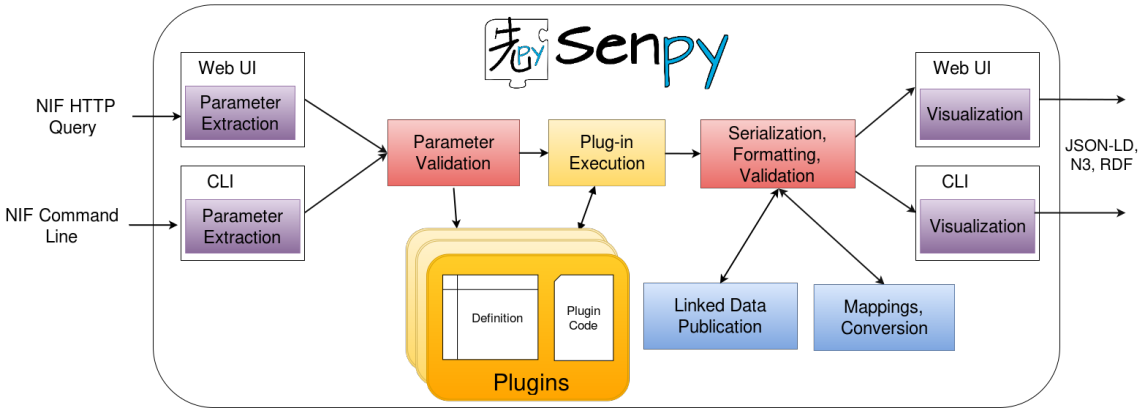


Figure 2.9: Senpy's Architecture [47]

CHAPTER 3

Case Study and Related Work

3.1 Introduction

In this chapter, we will start off by giving an introduction about the chosen dataset. Secondly, we will analyze related works that have been published (Mostly in PAN 2012 competition [32])

3.2 Case Study

For our case study we decided to use the data provided at PAN 2012 competition. As the organization describes it on the web page "PAN fosters digital text forensics research by organizing shared task evaluations. Shared tasks are computer science events that invite researchers and practitioners to work on a specific problem of interest, the task [1]".

On 2012 PAN selected as a task to identify sexual predators using Machine Learning techniques [32]. The PAN 2012 Sexual Predator Identification competition was comprised of two subtasks: the first is identification of user ids (anonymized) that are "owned" by

[1] http://pan.webis.de/index.html

Internet sexual predators; the second was identifying sexually explicit or predatory lines inside that users posts.

To define the term "sexual predator" for the competition they referred to the definition given in the New Oxford American Dictionary which is defined as "a person or group that ruthlessly exploits others" and to the interpretation and notation given in Wikipedia that the term "is used pejoratively to describe a person seen as obtaining or trying to obtain sexual contact with another person in a metaphorically 'predatory' manner".

We will describe in more depth the data provided by PAN12 in section 4.2. It basically consisted of an Extensible Markup Language (XML) file with conversations between chatters and it provided information about the conversations' and authors' ID as well as the time and text of the messages sent by each author.

Later on, after having our results from the PAN12 dataset, we wanted to try our model with new data coming from Twitter. We collected a set of tweets used by Abhijeet Kasture in the thesis submitted to Auckland University of Technology for the fulfillment of the requirements for the degree of Master of Computer and Information Science (MCIS) [23], in order to have a more accurate prediction in our Senpy's Plugin. This dataset consist of 385 Tweets, labeled by Abhijeet as cyberbullying tweets. Afterwards we will talk about its results in Section 4.7.

3.3 Related Work

There has not been many previous useful work on detecting cyberbullying, and what it exists is the work done at PAN 2012.

We are going to talk about the different approaches on solving these problems as well as some of the 2012 PAN solutions. ***CAW 2.0:*** Previous to PAN 2012, a misbehavior detection task was offered by the organizers of CAW 2.0, but only one submission was received. Yin, et. al determined that "the baseline text mining system (using a bag-of-words approach) was significantly improved by including sentiment and contextual features. Even with the combined model, a support vector machine learner could only produce a recall level of 61.9%" [50].

Chatcoder: Kelly Reynolds, April Kontostathis and Lynne Edwards [38] created Chatcoder, a tool used to detect cyber abuse in an online chat rooms. In a study they did in 2011 they used a dataset of questions asked in Formspring.me, previously labeled as having or not cyberbullying content by three Amazon workers and divided into two sets, one for

testing and one for training. All posts contained the following information:

1. Does this post contain cyberbullying (Yes or No)?
2. On a scale of 1 (mild) to 10 (severe) how bad is the cyberbullying in this post (0 for no cyberbullying)?
3. What words or phrases in the post(s) are indicative of the cyberbullying?
4. Any additional information the worker would like to share about this post.

At least two of the three workers had to agree in order for a post to receive a final class label of "Yes" or "No" in their training and testing sets. They downloaded a set of "bad words" and gave them a severity level and trained their models to get the total number of bad words on a post and their total severity, calculated as the sum of all the bad word's severities in a post.

They used the following algorithms to train the model:

- J48: The J48 option uses the C4.5 algorithm to create a decision tree model [36].
- JRIP: JRIP is a rule based algorithm that creates a broad rule set then repeatedly reduces the rule set until it has created the smallest rule set that retains the same success rate [10].
- IBK: The Instance Based Algorithm (IBK) implemented in Weka is a k-nearest neighbor approach [2].
- SMO: The Sequential Minimal Optimization (SMO) algorithm is a function-based support vector machine algorithm [35].

But since their dataset had a lot of false positives, their results were also filled with false positives (Which are better than false negatives when detecting predators as they alleged) and the accuracy was 78.5%.

Chatcoder at Pan12: April Kontostathis, Will West, Andy Garron, Kelly Reynolds and Lynne Edwards participated in both subtracts of PAN 2012 [25]. In their solution they studied the communicative patterns of cyber predators and their victims. They used machine learning approaches, combined with an in depth study of communicative patterns, to identify posts that fall into one of three categories that are often used by cyber predators when they communicate with their victims following some previous work done in 2011 [30].

The three mentioned categories are: Personal Information Exchange, Grooming, and Approach. They used a dictionary and a set of 15 attributes which were collected for each post. The attributes were:

- Total number of words in a line (Strings of characters separated by spaces)
- Number of first person pronouns in a line (e.g. I, me)
- Number of second person pronouns in a line (e.g. you, your)
- Number of third person pronouns in a line (e.g. he, them)
- Number of personal information nouns (e.g. age, pic)
- Number of relationship nouns (e.g. boyfriend, date)
- Number of activities nouns (e.g. movie, favorite)
- Number of family nouns (e.g. mom, sibling)
- Number of communicative desensitization verbs (e.g. kiss, suck)
- Number of communicative desensitization nouns (e.g. bra, orgasm)
- Number of communicative desensitization adjectives (e.g. horny, naked)
- Number of communicative desensitization words (e.g. sex, penis)
- Number of re-framing verbs (e.g. teach, practice)
- Number of approach verbs (e.g. meet, see)
- Number of approach nouns (e.g. hotel, car)

They then trained their model with files with conversations separated by authors (previously labeled as predators or not) searching for this attributes, counting the number of lines for each author that were labeled as containing personal information, grooming and approach.

They created three classification systems using the Weka data mining tool kit [49] to identify the predatory authors, and they also used the C4.5 decision tree learner (implemented as J48 in Weka) [36], and the RIPPER rule-learning algorithm (implemented as JRip in Weka) [10], for their learning experiments. By dividing the problem into this attributes and also by determining that only one-on-one conversations could be predatory in nature, they achieved better results than the previous year (87%) but it was still pretty weak because they had a lot of false positives.

They came to two category-related conclusions, "first that Comments such as 'too old for you', 'you are very young', 'go to school', 'get someone to drive you', etc. are indicative of age, and that should be classified in a different category besides Personal Information Exchange as it was in their work, and the second thing is something they refer to as 'awareness of guilt'. with comments like 'are you a cop', 'we can get into a lot of trouble for this', 'I wish you were older', etc. instead of their Approach category".

Colin Morris' Master of Science: In 2013 Colin Morris in his Master of Science paper "Identifying Online Sexual Predators by SVM Classification with Lexical and Behavioral Features" expanded his work done with Graeme Hirst for PAN 2012 [31] in which they ranked fourth with a 0.8652 in F0.5 score. During PAN 2012 they used a standard bag-of-words model. "They experimented with a number of standard text preprocessing routines including lowercasing, stripping punctuation, and stemming. None of these routines improved performance, thus their final results used simple space-separated tokens as features. They also tried to add 'smarts' to their lexical features with some transformation rules and introduced special tokens such as emoticons from Wikipedia [2] , male and female names [3], \NUM for any sequence of digits used mostly for age recognition, and Phone numbers, but these transformations seemed to add little discriminative power to their model".

Their machine learning algorithm of choice was SVM, using the Library for Support Vector Machines (LIBSVM) [9]. In testing their models, they used cross-validation with n = 5.

They developed a second filter to identify predators from victims (different from the differentiation of predators from no-predators) because they came to realize that most of all false positives they were getting were actually victims. Because predators and victims are discussing the same topics and are virtually identical in terms of number and length of conversations, they were motivated to establish their "symmetry-breaking" behavioral features such as message ratio, number of repeated messages, and number of initiations. So what they did was identifying predatory from no-predatory conversations and predators from victims. Because the conversation could have been between two consenting adults.

They tried various combinations of these features or steps getting the following results:

Note that they used vectors where every token t that appears more often than a threshold [4] yields two features: "the number of times the focal author utters t, and the number of times any of the focal author's interlocutors utters t". In that way they were able to

[2] http://en.wikipedia.org/wiki/List_of_emoticons
[3] http://www.galbithink.org/names/
[4] Threshold empirically set to 10

Variation	Recall	Precision	F1-Score
-	0.73	0.88	0.80
Partner Flip	0.73	0.92	0.81
Predator-Victim classification	0.65	0.89	0.76
Predator-Victim classification and Partner Flip	0.65	0.91	0.76
Transformation Rules	0.71	0.90	0.80
Transformation Rules and Partner Flip	0.70	0.93	0.80
Only Lexical Features	0.74	0.93	0.82
Only Lexical Features with Partner Flip	**0.74**	**0.95**	**0.83**
Only Focal Lexical Features	0.69	0.87	0.77
Only Behavioral Features	0.70	0.47	0.56
Baseline	1.0	0.001	0.003

Table 3.1: Results Obtained By Colin Morris

process all data said by predators and said to predators. It may be interesting to highlight that the most common answer given to predators was: wtf. Their classifier turned out to be very accurate at telling predator from victim but not so much at distinguishing predator from non-predator.

PAN12 Winners: The winners of the first task (Identifying predators) were Esaú Villatoro-Tello, Antonio Juárez-González, Hugo J. Escalante, Manuel Montes-y-Gómez, and Luis Villaseñor-Pineda obtaining a F0.5 score punctuation of 0.9346 [48]. Their proposed method for detection of misbehaving users in chats is based on two main hypotheses: "(i) terms used in the process of child exploitation are categorically and psychologically different than terms used in general chatting; and (ii) predators usually apply the same course of conduct pattern when they are approaching a child".

They didn't apply any preprocessing filters to their data (remove punctuation marks or stemming processes) because of the atypical grammar of chats (informal and full of orthographic errors). Although they didn't use a pre-processing filter they did use a pre-filtering process. This pre-filtering stage consisted in removing all the conversations that

accomplish at least one of the following conditions:

- Conversations that had only one participant
- Conversations that had less than 6 interventions per-user
- Conversations that had long sequences of unrecognized characters (apparently images).

For distinguishing victim from predator they divided every conversation in sets of interventions from different users, and then they tested them with conversations from victims and predators. For the victim from predator classifier they obtained features vectors of 16709 elements. They used two classifiers from the CLOP toolbox [3] in the text classification experiments; these are Neural Networks (NN) and SVM classifiers. The NN classifier was set as a two layer neural network with a single hidden layer of 10 units. For the SVM they tried linear and polynomial kernels.

During the development phase they used two-fold cross validation to test their performance. "For the baseline experiment we employed a BOW representation using either a boolean or a TF-IDF weighting scheme. By following the same procedure established for the Suspicious Conversations Identification (SCI) and the Victim From Predator (VFP) classifiers", under this configuration they obtained features vectors of 117015 elements with which they achieved a 0,98 accuracy (0.95 F-Score) in the SCI classifier and a 0.94 in the VFP classifier.

The final classifier consisting in both VFP and SCI obtained an F0.5 Score of 0.8936 being the winner of the PAN 2012 competition.

3.4 Conclusions

To summarize and taking into account the results published by PAN12 organization [22] we could see the results and methods for both problems:

- ***Problem 1. identify predators:*** The main problem in this competition was that since they wanted to have realistic solutions, they implemented both training and testing data sets full of false positives (they had less than 1% true positives). Most of the teams overcame this problem by using a two stage classifier where the first one was in charge of distinguishing between conversations with or without predators.

The organization said: "Apart from one case where participants used machine learning approaches that work at character level [...], we can divide the used features into two main categories: 'lexical' features and 'behavioral' features. Lexical features are those that can be derived from the raw text of the conversation [...]. It is to be noted that, in general, lexical features have been used without any stemming or stop-word removal, to preserve each author own style, including misspelling and grammatical errors. Behavioral are all those features that captures the 'actions' of a user within a conversation.[...] One of the most common approach was the creation of a single set of features for each author, to be able to profile him and exploit his predator potential."

In the classification step it could be observed different proposed methods, but SVM were the most used. "In general, they were used in most cases for the first (predator-vs-all), then also for the second step of the classification (predator-vs-victim)".Other classifiers such as a Neural Network classifier, based on Maximum-Entropy, decision trees, k-NN and random forest as well as Naïve Bayes were also used by the participants in this competition.

- ***Problem 2. identify predators' lines:*** It is important to note that for this second problem, no training data was available for the participants. "The straightforward solution was to return as relevant all the conversations lines of all the identified predators from the first problem. One of the most used method was a filtering of all the predator conversations through a dictionary of 'perverted' terms or with a particular score (e.g. TF-IDF weighting).

  Similar to this approach, another first computed the Language Models (LMs) of the part of the conversation considered predatory and then computed the differences between the actual conversation and the LMs. To conclude, the last approach was simply to return those lines already labeled as predatory in the proposed algorithm by the default method for problem 1 (working at line level)".

CHAPTER 4

Model Building and Evaluation

4.1 Overview

In this chapter we will present and define the global architecture of the project, which can be easily identified by examining our project's pipeline, presented in Figure 4.1, as well as the different procedures adopted to achieve our solution. Later we will focus on analyzing the obtained results explaining the different algorithm used and presenting the analysis of our classification model. Last but not least, we will analyze the results obtained using the same model but a different dataset.

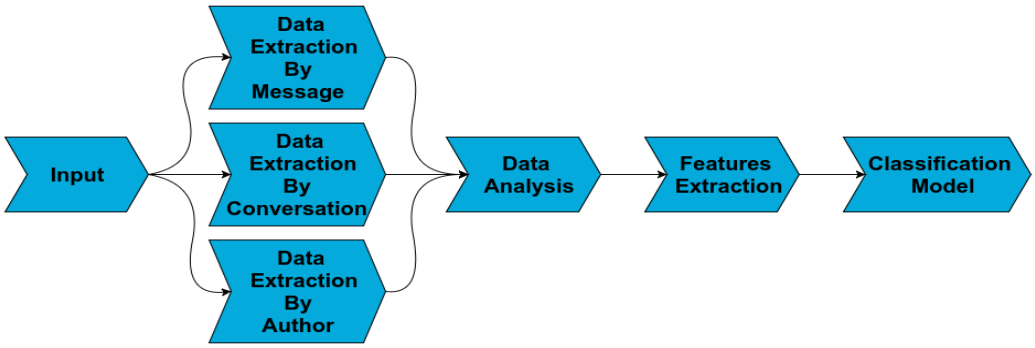


Figure 4.1: Stages in pipeline

4.2 Input

We will start off by describing in more depth the dataset used in this project. In order to train our model with a realistic and complete dataset, we extracted the dataset provided by PAN12 [32] at their competition. The dataset consists of 66927 conversations from various chats in an XML file. The XML file is organized as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<conversations>
<conversation id="id_of_the_conversation">
<message line="1">
<author>author_1_id</author>
<time>02:56</time>
<text>Bla bla bla bla</text>
</message>
<message line="2">
<author>author_2_id</author>
<time>02:56</time>
<text>bla bla</text>
</message>
[...]
<message line="n">
<author> author_1_id </author>
<time>07:12</time>
<text>bla bla bla</text>
</message>
</conversation>
</conversations>
```

Every conversation has its own and unique id. For each conversation, every message is tagged with a unique line number inside the conversation. Each message is produced by an author with a unique Id and we can also find the text of the message and the time when it was said. Since the normal thing is not to have a chat full of cyber abusers, we have around 66000 conversations with only 2016 conversations containing a predator. The rest of the conversations where regular conversations between two adults, and other conversations that could mislead our classifier such as non-sexual conversations between adults and children as well as sexual conversations between two consenting adults.

We have organized the dataset based in three different procedures, which we will describe in the following subsections, using python functions to place them into rows of three data frames in Pandas [33], previously converting it into a CSV file.

4.3 Preprocessing

Before training our model we need to extract correctly the information. We need to select which way of data extraction and organization is the most optimal and to preprocess, clean and convert the information, in data our model can understand and work with. We tried three different data extraction methods, grouping them in different tables with the headers shown in Table 4.1.

By Message	**By Conversation**	**By Author**
1. Conversation Id	1. Conversation Id	1. Author Id
2. Number of participants in conversation	2. Number of participants in conversation	2. Number of conversations the author has participated
3. Message Number inside the conversation	3. Number of messages in conversation	3. Number of messages in all conversations
4. Author of the message	4. Messages	4. Author is predator
5. Time of the message	5. Predator in conversation	
6. Text of the message	6. Predator started conversation	
7. Is the Author of the message a predator?		
8. Is there a predator in the conversation?		

Table 4.1: Extracted Data Table Headers

4.3.1 Data Extraction by Message and Analysis

In this first organization we grouped the dataset into a Pandas table, having one message of our dataset in each row. We obtained data from 903607 messages.

After examining this Data Frame we concluded that there was no significant relation between the attributes 1, 3, 4 and 5 of Table 4.1 with the question: “Is the author of that message a predator?”.

Since our predators are not only marked as sexual predators because of a single message but because a whole conversation, there was no clear relation between the text of the message and the mentioned question. The only parameter that did answer our question was the second one: "Number of participants in conversation" which we would see even clearer with our following grouping.

For all this, we saw that this grouping was actually not very useful to our model,meaning that it did not give us any useful information, and decided to arrange the next two groups.

4.3.2 Data Extraction by Conversation and Analysis

We decided to gather all the information and group it by conversation. We got a table with 66927 conversations and the attributes listed in Table 4.1 in its header.

By examining the data set organized by conversation we noted that it is very unbalanced, having very few conversations containing predators in relation to the ones that don't. We have over almost 65000 conversations without predators in contrast to 2016 containing them as it is presented in the Figure 4.2.

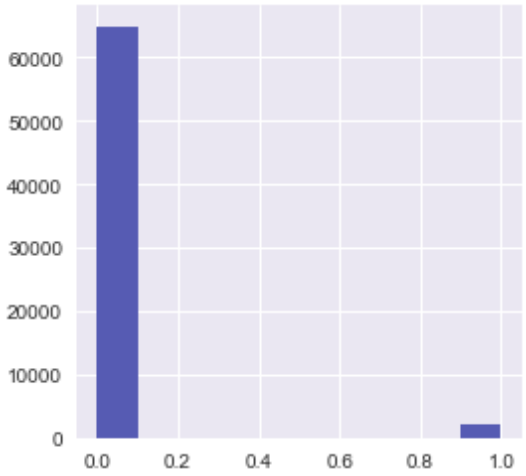


Figure 4.2: Predator in conversation

In the "Messages" column we unified all the messages in each conversation. By having a "Predator Started Conversation" column we wanted to check if predators usually start conversations or not. We studied only the conversations containing predators and we saw that we had 4 times more conversations where the predator was the first to speak as we see in the graphic shown in the Figure 4.3.

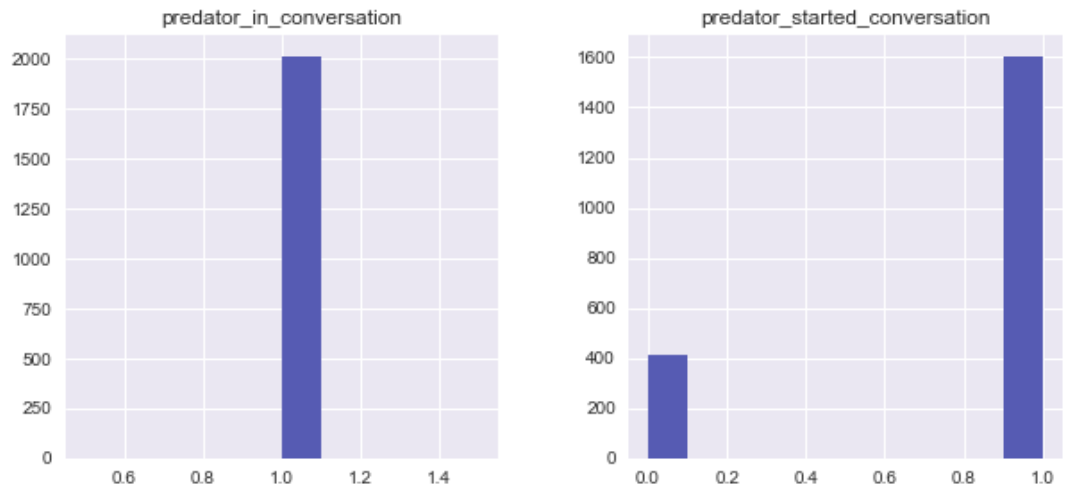


Figure 4.3: Predator Started Conversation

We see that it is a very high relationship but it is not highly deterministic in distinguishing predator from victim since we still have 400 conversations not started by predators, and in our case we are more concern about getting false negatives than false positives (We rather tag someone as a sexual predator and clear them after having analyzed their case than tag some predator as a non-predator). In other words, we prefer having false alarms than misses.

As shown in the Figure 4.4, we spotted another high feature in the number of participants in a conversation, having only four conversations with predators and three participants and only one conversation with four participants and containing predators. This indicates that almost all predators only talk in conversations with only two participants (one plus the predator) or only one participant in case no one answered the predators messages.

4.3.3 Data Extraction by Author and Analysis

In this section we scanned the dataset separating messages per author. We extracted the data in a table with the header explained in Table 4.1.

We obtained information from 97689 authors of which only 142 were classified as predators.

One new feature relation we discovered by analyzing the dataset by authors was related to the total number of messages. We discovered that predators send a noteworthy higher number of messages than non predators.

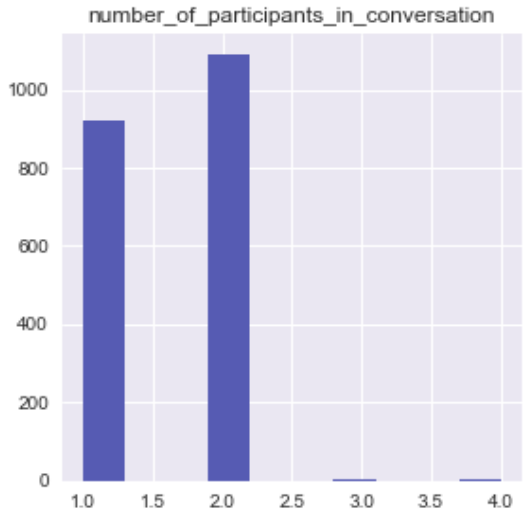
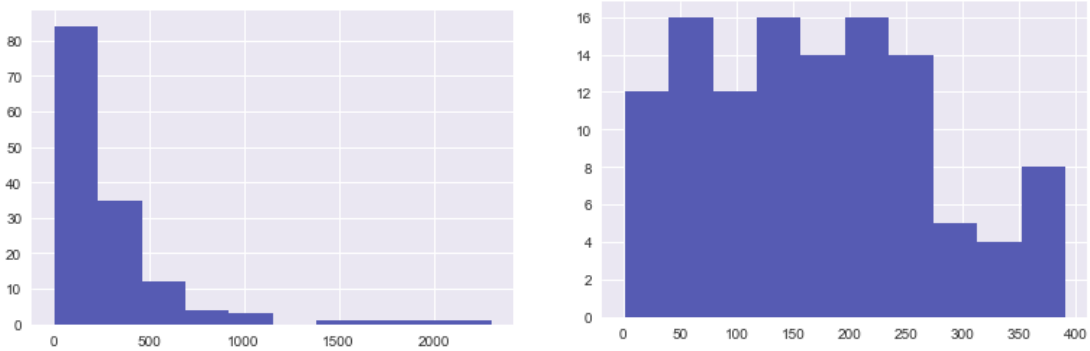


Figure 4.4: Number of participants in a predatory conversation

Since the number of non-predators in our dataset is humongous compared to that of predators, we obviously obtain a substantially higher number of messages but in Figures 4.5 and 4.6 we can really appreciate this enormous difference in relation to the total number of messages.

In Figures 4.5a and 4.6a we can see the distribution of the number of messages sent by predators and non-predators. In Figures 4.5b and 4.6b we zoomed in the plot from 1 message sent to 400 messages sent. We can easily see that predators send much more messages, while most non-predators send less than 50.

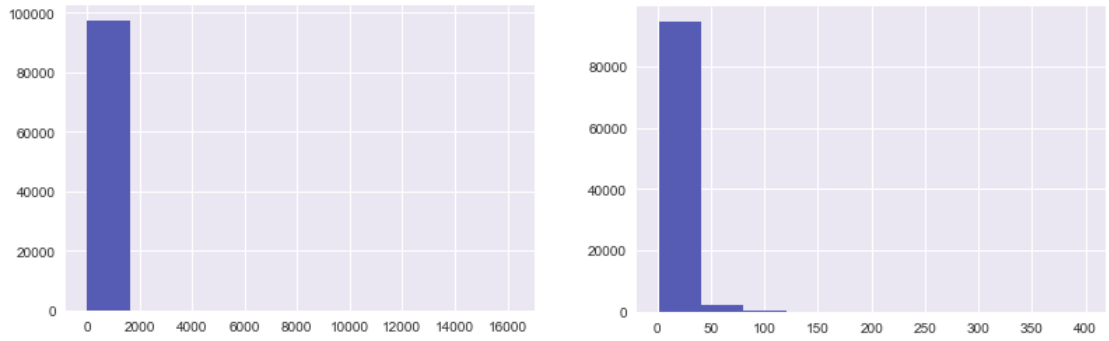
With respect to the number of conversations each author has participated, we found no clear distinction between predators and non-predators.



(a) Number of messages

(b) Number of messages bellow 400

Figure 4.5: Number of messages of predators



(a) Number of messages

(b) Number of messages bellow 400

Figure 4.6: Number of messages of non-predators

4.3.4 Data Cleaning and Feature Extraction

Once our data has been analyzed, it is necessary to extract the key distinguishing features of our data to train our model, and to clean our data from parts we don't need.

We had some numeric features in our Data Frame that could be of use to help us train the model, but we have our user's messages as well. For our model, raw text gives no information at all, so in order to obtain some useful information, we need to apply some processing such as NLP.

First we divided the text into words. After that, we decided to use Stemming and Lemmatization to reduce the number of words that mean the same. Then, we removed the stop words included in NLTK, but since we think sexual predators may be inclined to use some aggressive language (such as exclamation marks !!!) we decided not to remove the punctuation marks, also, due to the specific nature of chat-like language we decided not to remove rare words or typos.

We tried applying Part of Speech (POS) Tagging [14], that is the process of assigning a grammatical category to a word but, since it enlarged a lot the size of our data, we decided to apply a variation where we got only assigned the classification in table 4.2 to each word.

We also decided to use n-grams, with n = 3. This are unigrams, bigrams and trigrams. An n-gram is a continous sequence of n consecutive elements in a text sequence, in our case, the text sent by our authors. This way we decided not only to analyze word by word our data but to analyze every sequence of two or three consecutive words.

Last but not least, we decided to import two plugins developed at GSI GitLab [18], "Age" and "Gender", with the hope it would help us train our model better. With the help of

Tag	Meaning	English Examples
ADJ	adjetive	new, good, high, special, big, local
ADP	adposition	on, of, at, with, by, into, under
ADV	adverb	really, already, still, early, now
CONJ	conjunction	and, or, but, if, while, although
NOUN	noun	year, home, costs, time, Africa
NUM	numeral	twenty-four, fourth, 1991, 14:24
PRON	pronoun	he, their, her, its, my, I, us
VERB	verb	is, say, told, given, playing, would

Table 4.2: POS classification

these we were able to determine the age of a text's author and whether it was written by a male or female author.

After having cleaned our data we decided the features for our pipeline.

1. "**Lexical Stats**": We gathered the number of words an author had said.
2. "**Words**": All the different words said, having removed the ones with same meaning and/or the stop words.
3. "**nGrams**": All our unigrams, bigrams and trigrams.
4. "**POS Stats**": The POS classification of the words in our messages.
5. "**Latent Dirichlet Allocation (LDA)**": LDA is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar.
6. "**Gender and Age**": Gender and Age Senpy Plugins [47] to determine the Gender and Age of our messages' author.

Lastly, aside of studying our model scores (percentage rate of accuracy in predictions) we will also study the confusion matrix (Table 4.3) which shows all the True Positives (TPs), True Negatives (TNs), False Positives (FPs) and False Negatives (FNs), because, as

previously stated, one of our project goals is to obtain more "False Positives" than "False Negatives".

		Predicted Values	
		Positive	Negative
True Values	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Table 4.3: Confusion Matrix

Having our data well tagged and cleaned and our features extracted, we used NLP and NLTK as explained in subsection 2.4.

4.4 Model Building

4.4.1 Classification Algorithm Selection

As for the model we chose, once we had all our data in an optimal organization, we first tried Linear Support Vector Classification (SVC) (Of SVM) as recommended in the Scikit-Learn page [41] following the diagram in Figure 4.7 found in their "Choosing the right estimator" page. And since the data we are analyzing is mainly text we also chose to use Naive Bayes (MultinomialNB).

We then tried out algorithms that looked like they would work well on our model but, obviously, it can not be proved until they are tested.

We tried all of them using K-Fold Cross Validation with k = 10. Cross-Validation is a technique for assessing how the results of a statistical analysis will generalize to an independent data set. One round of cross-validation involves partitioning a sample of data into complementary subsets, in our case 10 subsets, performing the analysis on one subset (called the training set), and validating the analysis on the other subset (called the validation set or testing set) and then do this iteration the same number of times as the number of subsets, so every subset is tested on at least once [15].

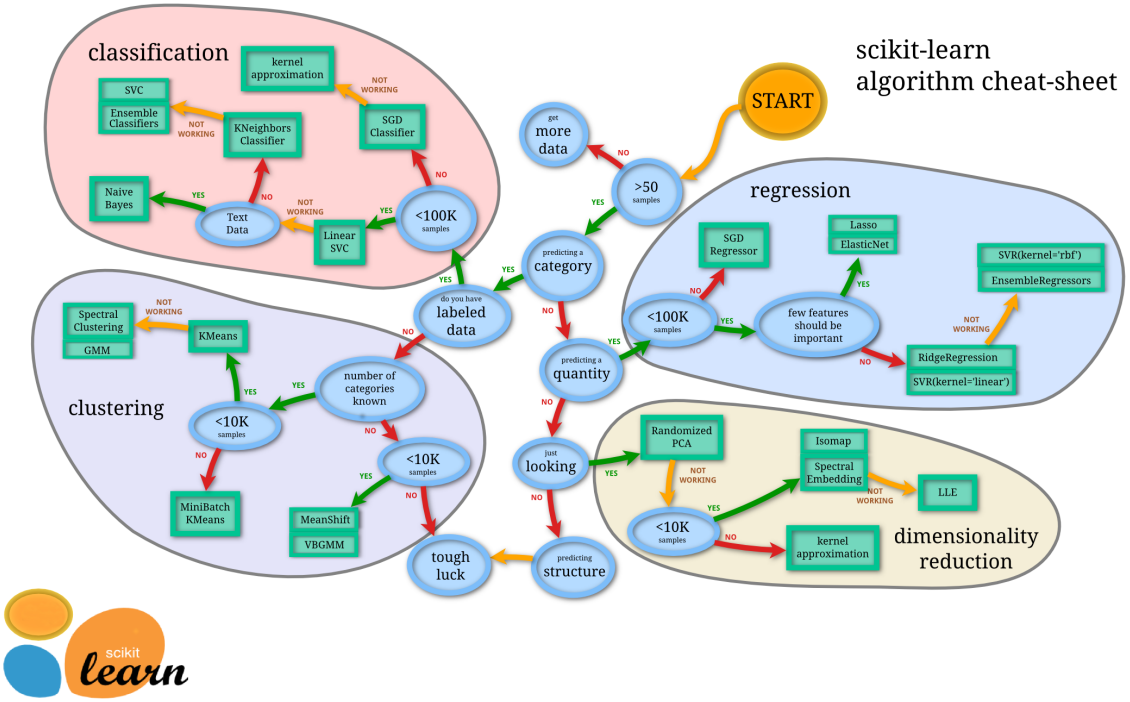


Figure 4.7: Choosing the right estimator [41]

4.4.2 Model's Classifier Parameter Tunning

We decided to use various algorithms as described in subsection 2.1.3. These algorithmic classifiers allow us to set certain parameters.

Manually changing the parameters to find their optimal values is not practical. Instead, we considered finding the optimal value of the parameters as an optimization problem. Sklearn comes with several optimization techniques for this purpose, one of which is "grid search" [27]. Grid Search does an exhaustive search over specified parameter values for an estimator. It also gives you the possibility of having a grid of parameters to optimize, being able to enhance more than one parameter.

We have left the default parameters in the Logistic Regression and Decision Tree Classifiers but manually changed some of the parameters of the following classifiers:

- ***SVM***
  - *C:* This parameter indicates how much you want to avoid misclassification, traded off against simplicity of the decision surface. A low C makes the decision surface smooth while a high level tries to find the smallest margin between samples, enabling you to classify more accurately.

Since our dataset is small we would want a smaller value of the C parameter. The value we selected was $C = 0.05$.

- *Kernel:* It sets the kernel type to be used in the algorithm. It must be one of "linear", "poly", "rbf", "sigmoid", "precomputed" or a callable. The one we selected was $Kernel = linear$.
- *Probability:* This parameter describes whether to enable probability estimates or not. This must be enabled prior to calling fit, and will slow down that method. So we also set this parameter to $True$.

- ***Multinomial Naive Bayes:***
  - *Alpha:* It is an additive smoothing parameter. It accepts a "float" type parameter, having 0 for no smoothing. We wanted some smoothing but not too much so we had a small float number. We set this parameter to $alpha = 0.1$.

- ***Random Forest Classifier:***
  - *n_estimators:* Random forest classifier creates several decision tree classifiers. This parameter sets the number of trees in the forest. By increasing the number of trees the accuracy keeps getting better, at a computational cost, until it converges to a certain point when it just keeps constant. The chosen value for our project is $n\_estimators = 100$.

4.5 Results

After having trained our model with these algorithms we tried out various others, some of which we explained in subsection 2.1.3, we got various results, some of the better ones are shown in table 4.4, where accuracy is calculated from the cross_val_scores function [28].

We also calculated the $F_1 - Score$ [11], which is a way of measuring a test's accuracy. It takes into account both the precision ($\frac{TruePositives}{TruePositives+FalsePositives}$) and recall ($\frac{TruePositives}{FalseNegatives}$) of the test and calculates the F_β-Score with Ecuation 4.1, which translates into Ecuation 4.2 using TPs, TNs, FPs and FNs.

$$F_\beta = (1 + \beta^2) \cdot \frac{Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{4.1}$$

$$F_\beta = \frac{(1 + \beta^2) \cdot \mathrm{TP}s}{(1 + \beta^2) \cdot \mathrm{TP}s + \beta^2 \cdot \mathrm{FN}s + \mathrm{FP}s} \tag{4.2}$$

For our model evaluation we decided to use an $F_1 - Score$ which is calculated by having $\beta = 1$ in Ecuation 4.2, which results in ecuation 4.3.

$$F_\beta = \frac{2 \cdot \text{TP}s}{2 \cdot \text{TP}s + \text{FN}s + \text{FP}s} \tag{4.3}$$

Algorithm	**Accuracy (mean (std))**	$\mathbf{F_1 - Score}$
SVM	0.96 (+/- 0.06)	0.959
Multinomial Naive Bayes	0.96 (+/- 0.05)	0.956
Decision Tree Classifier	0.94 (+/- 0.03)	0.944
Logistic Regression	**0.98 (+/- 0.05)**	**0.975**
Random Forest Classifier	0.97 (+/- 0.05)	**0.975**
Bernouilli Naive Bayes	0.83 (+/- 0.11)	0.790

Table 4.4: Algorithms and Results of PAN12 DataSet

As mentioned before, an important part of our model would be to have the least FNs as possible or at least have less FNs than FPs (We do not want predators being tagged as innocents). So we took into deeper examination the confusion matrix for all the different algorithms, represented in Table 4.5.

Algorithm	True Positive	False Positive	False Negative	True Negative
SVC	417	9	14	128
Multinomial Naive Bayes	406	20	**5**	137
Decision Tree Classifier	407	19	13	129
Logistic Regression	423	3	11	131
Random Forest Classifier	423	3	11	131
Bernouilli Naive Bayes	426	0	98	44

Table 4.5: Algorithms' Confusion Matrices of PAN12 DataSet

4.6 Evaluation

In this section we will evaluate the results shown in Tables 4.4 and 4.5 in order to determine the best algorithm used in our model. We will select the best one in terms of accuracy and $F_1 - Score$ and taking into account their confusion matrices.

- ***Accuracy and* $\mathbf{F_1}$ – Score:** Taking a look at Table 4.4 we can see that, in terms of accuracy and $F_1 - Score$, the ones that got the best results are "Logistic Regression" and "Random Forest Clasifier". But they got 0.98 and 0.97 in accuracy and 0.975 in the $F_1 - Score$ which is really close to the other algorithms. By doing various tests with different data they continued to be the best, sometimes with Multinomial Naive Bayes as well, but the rest were always close behind. This result could change if we changed datasets (another algorithm could take first place) but the rest would still work just fine.

  With this results we can safely say that, concerning accuracy and $F_1 - Score$; All classifiers, except maybe for "Bernouilli Naive Bayes", work good with our dataset. There is not enough difference between them to say that only one is the best.

- ***Confusion Matrix:*** When we take a look at Table 4.5 things change. Taking into account that one of our main goals is to not clear a predator by saying he/she is innocent, meaning that we want to minimize as much as possible our FNs, we clearly have a winner on this topic. Multinomial Naive Bayes was clearly the one with the least FNs. Even though in total it had more false predictions (FNs + FPs), we would rather have 25 wrong predictions, as the case of Multinomial Naive Bayes, with only 5 being FNs, than having only 13 false predictions with 11 FNs.

So to our criteria the best algorithm used in this case was Multinomial Naive Bayes, followed by Random Forest Classifier and Logistic Regression.

4.7 Alternative Model Evaluation: Twitter Data

4.7.1 Overview

In this section we will see the results obtained by applying the same preprocessing we saw in subsection 4.3.4 to our model but this time with the new dataset defined in subsection 3.2 [23].

This dataset is composed of 385 cyberbullying-tagged tweets along with 385 regular conversations. That means, 770 new text data.

For this part we used the same algorithms and same parameter tunning as in subsections 4.4.1 and 4.4.2 to be able to really compare our model against different datasets.

4.7.2 Results and Evaluation

In this case we obtained the accuracies and $F_1 - Scores$ shown in Table 4.6. As we can see the results are a bit lower than in the previous chapter but they are still pretty high. In this case our best Algorithm in Accuracy as well as $F_1 - Score$ is "Random Forest Classifier" followed close by "Logistic Regression".

Using this dataset we see that now, "Naive Bayes Classifiers" are the worst of our list, which makes sense since the data now being analyzed are tweets that, even though they are still text, it is full of typos and colloquial words.

Algorithm	**Accuracy (mean (std))**	$\mathbf{F_1 - Score}$
SVM	0.91 (+/- 0.05)	0.918
Multinomial Naive Bayes	0.89 (+/- 0.06)	0.893
Decision Tree Classifier	0.92 (+/- 0.06)	0.910
Logistic Regression	0.95 (+/- 0.03)	0.955
Random Forest Classifier	**0.96 (+/- 0.04)**	**0.964**
Bernouilli Naive Bayes	0.60 (+/- 0.14)	0.524

Table 4.6: Algorithms and Results of Tweet-Based DataSet

As for our goal to not miss-classify guilty users, if we take a look at Table 4.7 we can spot a clear winner. "Random Forest Classifier" is not only the one with least False Predictions, but also the one with the least FNs.

Algorithm	True Positive	False Positive	False Negative	True Negative
SVC	345	40	23	362
Multinomial Naive Bayes	316	69	13	372
Decision Tree Classifier	355	30	39	346
Logistic Regression	362	23	12	373
Random Forest Classifier	366	19	**9**	376
Bernouilli Naive Bayes	77	308	0	385

Table 4.7: Algorithms' Confusion Matrices of Tweet-Based DataSet

So, as a conclusion, for this new dataset, the optimal classifier, judging by it's Acurracy, $F_1 - Score$ and number of FNs, is the "Random Forest Classifier".

CHAPTER 5

Predatory Prediction on Twitter with Senpy

5.1 Introduction

Twitter is an online news and social networking service where users post and interact with messages, called "tweets". As of the third quarter of 2017 Twitter had 330 million monthly active users [43]. Many people all around the world use Twitter as a way of chatting with friends, share different aspects of their lives, events and opinions. Famous people, companies and even news channels use Twitter, making it one of the most important platforms nowadays.

Legal measures are being taken to enforce content control by tech firms, which have been warned by the European Union to remove extremist content faster [17].

Figure 5.1: Twitter Timeline [7]

Due to the humongous size of the platform, monitoring all its content manually is absolutely impossible, so automation is necessary. This chapter describes the process we have followed for creating a Senpy's plugin [47] to be able to deploy our product as a user-friendly service on Twitter in order to detect predatory behavior on a specific user.

5.2 Structure

Our plugin's pipeline structure can be seen in Figure 5.2.

For a correct implementation of the plugin we wrapped our code into a couple of python files ("*.py*") with functions to collect our data, preprocess it through our pipeline, train our model and predict based on new entries.

A plugin description file ("*.senpy*") is also needed. Here we specify the libraries needed by our system so they are loaded before the first run, the configuration parameters, our Twitter developer credentials, and where our dataset is located.

We also need our Twitter developer credentials on another file, although we will start to use the Environment Variable BITTER_CONFIG of Bitter [4] so that our credentials are not in a public docker image that everyone could see.

Last but not least, we need our dataset file to train our model the first time we use the plugin.

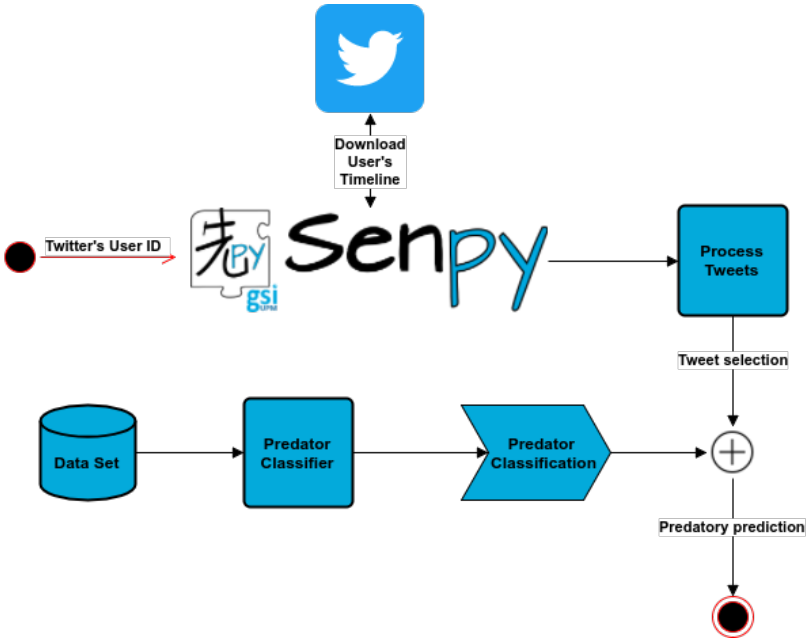


Figure 5.2: Senpy's pipeline

5.3 Obtaining Data

In order to obtain our data we made use of Twitter's API. With this tool combined with bitter [4] and Senpy [47], we are able to get a user's Twitter's timeline knowing that user's Twitter screen-name and/or Id.

The number of tweets that we can analyze is limited by Twitter's and Senpy's APIs so we obtain the last 200 tweets of each timeline. We then remove the retweets from the 200 tweets of the timeline, to have only content written by our author, and we get only the texts out of them (We eliminate photos, gifs, videos. But we don't remove links nor hashtags).

After having selected all our tweets, we pass them to our trained pipelined model to preprocess them, extract the features, and predict whether, based on that author's conversations, we found a sexual predator or not.

5.4 Displaying data

In order to display our results we used Senpy's dashboard. As shown in Figure 5.3, we can type in the Twitter's user name of the user we want to check. And the plugin we want to run, in our case, senpy_plugin is our sexual predator plugin.

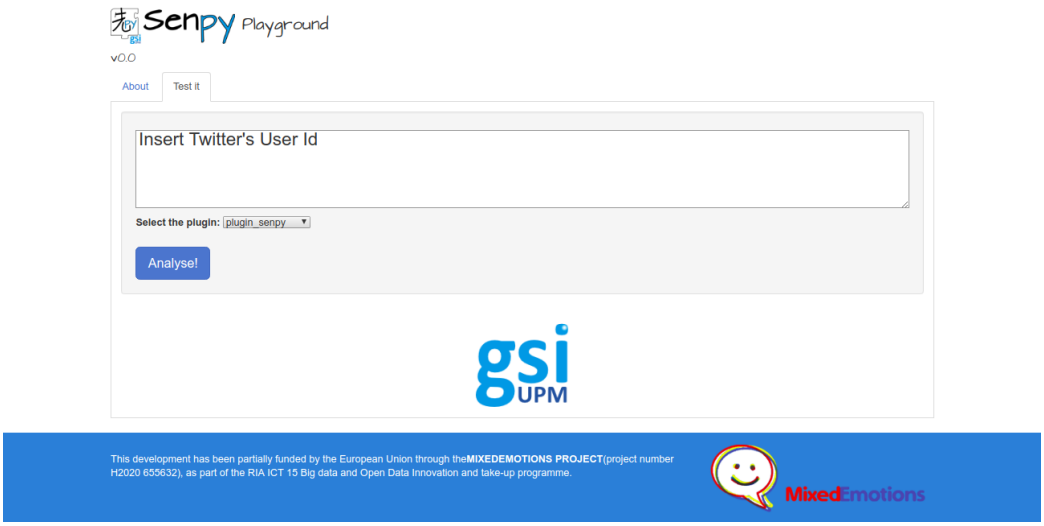


Figure 5.3: Senpy's Plugin

After analyzing the timeline, we receive the response in our example's Figure 5.4. There we can see interesting data such as the account name we analyzed, which plugin generated such response, and the most important one, if our author is a sexual predator or not.

As we see, our classification system gives the response as a Friend of a Friend (FOAF) ontology which is a machine-readable ontology describing persons, their activities and their relations to other people and objects [5]. FOAF is a computer language defining a dictionary of people-related terms that can be used in structured data.

For this first test we introduced Elon Musk's Twitter account and, as expected, our system classified him as a non-predator Twitter user as we see in Figure 5.4.

Figure 5.4: Elon Musk's Results

For this second test we introduced a predator's Twitter account and, as expected, our system classified him as a predator Twitter user. For sensitive issues and to protect the identity of this user we blurred the account name in Figure 5.5

Figure 5.5: Predator's Results

CHAPTER 6

Conclusions and future work

6.1 Introduction

In this chapter we will describe the conclusions extracted from this thesis, problems faced during it's realization, achievements reached and overpassed and suggestions about related future work.

6.2 Conclusions

This thesis' main goal was to create a system for detecting cyberbullying based on Machine Learning (ML). In addition to the main objective of making accurate predictions, we take into consideration that we value more to be mistaken by identifying an innocent as a predator (False Positive (FP)) than the other way around, and clear a predator (False Negative (FN)).

We studied related work to see how other people tried to accomplish this task. We extracted and preprocessed our data and tested our model with two different datasets, evaluating then, the obtained results in both cases.

These objectives were fully achieved and even overpassed our expectations in both cases. Using our PAN12 competition's data and testing with various algorithms we accomplished to get a very high "Accuracy" (0.98) and "$F_1 - Score$" (0.97) while having, in some algorithms, a very low number of FN (5 of 568).

On our second test, using our tweet-based dataset with 385 "bad" tweets, we achieved also very high results of "Accuracy" (0.96) and "$F_1 - Score$" (0.964), even though they were a bit lower than with the previous set. We got a very low number of FN as well (9 of 770).

In this project we have also created a plugin for sexual predator detection of Twitter users based on Senpy. This plugin counts with a user-friendly interface to easily obtain real-timed data of a user.

In the following sections we will describe in more depth the achieved goals, the problems faced during the project's development and some suggestions for future related work.

6.3 Achieved goals

During the development of this project we achieved various features and goals. The three most important ones are described bellow.

Build a pipeline for cyberbullying and sexual harassment detection. All our data needed to be well preprocessed in order to correctly build and train our model. We created a pipeline for this task to which we fed our training and testing data to collect all the key features.

Implement the system as a service. In order to show our system's functionalities we have deployed a Senpy's predator detection plugin which can be easily used by anyone interested in examining some shady Twitter accounts.

Vanquish all our targets. For both of our goals (Obtaining high Accuracy, $F_1 - Score$ and low FNs), and using two different datasets, we succeeded to attain better results than expected and better than most of the previous work on the subject, with a 98% accuracy, 97.5% $F_1 - Score$ and a low number of FN.

6.4 Problems faced

During the development of this project we had to face some problems. Next will note and give a brief description to the most significant ones.

Find Datasets. One of the main resources needed for the development of this project are good datasets to train our model. It has been harder than expected to find these datasets since, because of this project's topic, the content we needed is usually removed and hidden, not available to the public.

Senpy's API Limitations. So as to obtain our classification we made requests to our plugin implemented in Senpy. This plugin counts with a maximum text length limit as well as a request rate limit. This made us set a maximum of the 200 last tweets of a Twitter account instead of all the user's timeline.

Senpy's Ontology. Senpy is configured to Sentiment and Emotion analysis, not for predator classification, so we had to adapt our code to work fine with Senpy.

Twitter's API Limitations. Twitter, as Senpy, also limits the request rate on it's API. One can only do so many requests in a period of time.

Time to Process Our first dataset had many entries and features, consequently, it took a lot time to process all the information and delayed our projects time schedule.

Twitter's Moderators Thankfully Twitter's Moderators do a thoroughly and good work on detecting cyberbullying, racist and sexist content on it's platform and rapidly block such content and/or users. This however, made our work to try our plugin with some real predator, very hard.

6.5 Future work

In this section we will explain the possible new features or improvements that could be done to the project. Furthermore, we will study the possible future applications of our current system.

Use bag of words or dictionaries. When analyzing text, a common approach is to add a bag of words or dictionary with words classified as "bad words" or "predatory words". This way, our model could add another feature by counting the number of times the words in the dictionary appear in a user's texts.

In our case, since a regular conversation differs so much from a predatory one, we did not implement it since it did not improve very much our model's accuracy and increased the processing time. But with a good dictionary our accuracy and $F_1-Score$ could rise as to make the processing time worth it.

Catalog cyberbullying types. Another extra feature could be to distinguish cyberbullying types amongst a certain list such as: sexist, racist, violent, pedophile or bulling. This could be useful to search or filter for a specific type amid a big dataset containing various file types.

New Languages. In this project we utilized two datasets with English as the only language. In the case of our Senpy's plugin this means that only English-speaker users could be analyzed by our system.

By adding multiple models trained with various datasets in different languages we could broaden our target spectrum to many other countries and cultures.

Bibliography

[1] Internet Safety 101. Ciberbullyin, sexting y predadores sexuales. cómo proteger a tus hijos. https://familias.com/19/ciberbullying-sexting-y-predadores-sexuales-como-proteger-a-tus-hijos, 2017. Accessed: 2018-01-03.

[2] D. W. Aha and D. Kibler. *"Instance-based Learning Algorithms," Machine Learning, vol. 6, pp. 37–66, 1991.* Kluwer Academic Publishers, 1991.

[3] Amir Reza Saffari Azar Alamdari and Isabelle Guyon. Quick start guide for clop. Technical report, Technical Report, May 2006. http://ymer. org/research/files/clop/QuickStartV1. 0. pdf, 2006.

[4] Balkian. Bitter. https://github.com/balkian/bitter. Accessed: 2018-01-03. Commit 57eb73b53b88669d5f0bb4af4770f74e8dbe90dd.

[5] Dan Brickley and Libby Miller. Foaf vocabulary specification 0.99, namespace document 14 january 2014-paddington edition. *Recovered from http://xmlns.com/foaf/spec*, 2014. Accessed: 2018-01-03.

[6] Jason Brownlee. A gentle introduction to scikit-learn: A python machine learning library. https://machinelearningmastery.com/a-gentle-introduction-to-scikit-learn-a-python-machine-learning-library/. Accessed: 2018-01-03.

[7] businessinsider. Twitter timeline caption. http://static3.businessinsider.com/image/56eaf6d3dd089516398b4798-1190-625/how-to-get-your-old-twitter-timeline-back.jpg. Accessed: 2018-01-03.

[8] Tanushri Chakravorty. How machine learning works. https://thenewstack.io/how-machine-learning-works-an-overview/. Accessed: 2018-01-03.

[9] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[10] William W. Cohen. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.

[11] Wikipedia contributors. F1 score — wikipedia, the free encyclopedia, 2017. [Online; accessed 3-January-2018].

[12] Wikipedia contributors. Hal 9000 — wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=HAL_9000&oldid=817692436, 2017. [Online; accessed 3-January-2018].

[13] Wikipedia contributors. Machine learning — wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Machine_learning#Reinforcement_learning, 2017. [Online; accessed 3-January-2018].

[14] Wikipedia contributors. Part-of-speech tagging — wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Part-of-speech_tagging&oldid=816616286, 2017. [Online; accessed 3-January-2018].

[15] Wikipedia contributors. Cross-validation (statistics) — wikipedia, the free encyclopedia, 2018. [Online; accessed 3-January-2018].

[16] April Edwards (formerly Kontostathis), Lynne Edwards, and Dr. Brian D. Davison. Chatcoder. http://www.chatcoder.com, 2012. Accessed: 2018-01-03.

[17] Samuel Gibbs. Eu warns tech firms: remove extremist content faster or be regulated. european commission tells facebook, google, youtube, twitter and others that legislation is being considered if self-regulation continues to fail. https://www.theguardian.com/technology/2017/dec/07/eu-warns-tech-firms-facebook-google-youtube-twitter-remove-extremist-content-regulated-european-commission. Accessed: 2018-01-03.

[18] GSI. Gitlab. https://lab.cluster.gsi.dit.upm.es/. Accessed: 2018-01-03.

[19] GSI-UPM. Machine learning. https://github.com/gsi-upm/sitc/blob/master/ml1/2_5_0_Machine_Learning.ipynb, 2017. Accessed: 2018-01-03. Commit: 23073b3431529798f7ff9380126b212c17f5ec16.

[20] Heraldo. España es uno de los países donde más ciberacoso sufren los menores. http://www.heraldo.es/noticias/suplementos/salud/2016/03/15/espana-uno-los-paises-donde-mas-ciberacoso-sufren-los-menores-817032-1381024.html, 2016. Accessed: 2018-01-03.

[21] http://disney.wikia.com. Incredibles-disneyscreencaps.com-11200.jpg. http://disney.wikia.com/wiki/File:Incredibles-disneyscreencaps_com-11200.jpg, 2017. Accessed: 2018-01-03.

[22] Giacomo Inches and Fabio Crestani. Overview of the international sexual predator identification competition at pan-2012. In *CLEF (Online working notes/labs/workshop)*, volume 30, 2012.

[23] Abhijeet Sudhir Kasture. A predictive model to detect online cyberbullying. Master's thesis, Auckland University of Technology. School of Computer and Mathematical Sciences, 2015.

[24] PATRICK J. KIGER. Artificial intelligence is getting so smart, we need it to show its work. https://science.howstuffworks.com/artificial-intelligence-machine-learning-methodology.htm. Accessed: 2018-01-03.

[25] A Kontostathis, W West, A Garron, K Reynolds, and L Edwards. Identify predators using chatcoder 2.0-notebook for pan at clef 2012. *Forner et al.[5]*, 2012.

[26] James Le. The 10 algorithms machine learning engineers need to know. https://www.kdnuggets.com/2016/08/10-algorithms-machine-learning-engineers.html. Accessed: 2018-01-03.

[27] Scikit Learn. Gridsearchcv. http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html. Accessed: 2018-01-03.

[28] Scikit Learn. sklearn.model_selection.cross_val_score. http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html. Accessed: 2018-01-03.

[29] Steven Loria. Textblob: Simplified text processing. https://textblob.readthedocs.io/en/dev/. Accessed: 2018-01-03.

[30] I. McGhee, J. Bayzick, A. Kontostathis, L. Edwards, A. McBride, and E. Jakubowski. *Learning to Identify Internet Sexual Predation.* International Journal on Electronic Commerce. Volume 15, Number 3. Spring 2011., 2011.

[31] Colin Morris and Graeme Hirst. Identifying sexual predators by svm classification with lexical and behavioral features. In *CLEF (Online Working Notes/Labs/Workshop)*, volume 12, page 29, 2012.

[32] PAN2012. Pan2012 competition: Sexual predator identification. http://pan.webis.de/clef12/pan12-web/author-identification.html, 2012. Accessed: 2018-01-03.

[33] Pandas. Python data analysis library. https://pandas.pydata.org/. Accessed: 2018-01-03.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[35] John C. Platt. Advances in kernel methods. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods*, chapter Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pages 185–208. MIT Press, Cambridge, MA, USA, 1999.

[36] R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, 1993.

[37] La Razón. Uno de cada tres jóvenes españoles sufre ciberacoso. http://www.larazon.es/historico/9907-uno-de-cada-tres-jovenes-espanoles-sufre-ciberacoso-OLLA_RAZON_469638, 2012. Accessed: 2018-01-03.

[38] Kelly Reynolds, April Kontostathis, and Lynne Edwards. Using machine learning to detect cyberbullying. In *Machine learning and applications and workshops (ICMLA), 2011 10th International Conference on*, volume 2, pages 241–244. IEEE, 2011.

[39] Willi Richert. *Building Machine Learning Systems with Python*. Packt Publishing Ltd, 2013.

[40] Scikit-learn. The iris dataset. http://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html. Accessed: 2018-01-03.

[41] Scikit-Learn. Choosing the right estimator. flowchart. http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html, 2017. Accessed: 2018-01-03.

[42] SciPy. Scientific computing tools for python. the scipy ecosystem. https://www.scipy.org/about.html. Accessed: 2018-01-03.

[43] Statista. Number of monthly active twitter users worldwide from 1st quarter 2010 to 3rd quarter 2017 (in millions). https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/. Accessed: 2018-01-03.

[44] Statista. Age distribution of active social media users worldwide as of 3rd quarter 2014, by platform; active social network and active app users, excluding china. https://www.statista.com/statistics/274829/age-distribution-of-active-social-media-users-worldwide-by-platform/, 2017. Accessed: 2018-01-03.

[45] Statista. Social media statistics and facts. https://www.statista.com/topics/1164/social-networks/, 2017. Accessed: 2018-01-03.

[46] Twitter. Twitter api docs. https://developer.twitter.com/en/docs. Accessed: 2018-01-03.

[47] GSI UPM. Senpy. http://senpy.readthedocs.io/en/latest/senpy.html. Accessed: 2018-01-03.

[48] Esaú Villatoro-Tello, Antonio Juárez-González, Hugo Jair Escalante, Manuel Montes-y Gómez, and Luis Villasenor Pineda. A two-step approach for effective detection of misbehaving users in chats. In *CLEF (Online Working Notes/Labs/Workshop)*, 2012.

[49] E. Witten and I. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers, 2005.

[50] Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D Davison, April Kontostathis, and Lynne Edwards. Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7, 2009.