פרויקט גמר לשנה"ל תשע"ט

**automated detection of offensive language as a binary classification with Naive Bayes**

**הוכן לשם השלמת הדרישות**

**לקבלת תואר ראשון בהנדסה B.Sc**

**מאת**

**רן אוזן ליאור שוסטרמן**

**מנחה**

**דר' מרינה ליטבק**

**הוגש למחלקה להנדסת תוכנה**

**המכללה האקדמית להנדסה סמי שמעון**

**באר שבע**

**תשע"ט 2019**

**אישור המנחה \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**אישור ראש המחלקה \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Acknowledgements:**

We would like to thank all the people who guided and helped us along the way together we made this project happen

Firstly, and mainly we want to thank our mentor Marina Litvak that contributed to our project from her vast experience and knowledge and led us to solve the difficult problem our project deals with

We would like to express our thanks to the dean of student for giving us a scholarship on building a socially driven final project

Contents

[**1.** **English Summary** 5](#_Toc10748328)

[**2.** **Hebrew summery** 7](#_Toc10748329)

[**3.** **Intro** 8](#_Toc10748330)

[**3.1.** **Project motivation** 8](#_Toc10748331)

[**3.2.** **Project purpose** 8](#_Toc10748332)

[**3.3.** **Our main steps in this project** 8](#_Toc10748333)

[**3.4.** **How the algorithm works** 8](#_Toc10748334)

[**4.** **SRS** 9](#_Toc10748335)

[**4.1.** **Introduction** 9](#_Toc10748336)

[**4.1.1.** **Purpose** 9](#_Toc10748337)

[**4.1.2.** **Intended Audience and Reading Suggestions** 9](#_Toc10748338)

[**4.1.3.** **Product Scope** 9](#_Toc10748339)

[**4.1.4.** **Main user/client** 9](#_Toc10748340)

[**4.1.5.** **Objectives** 9](#_Toc10748341)

[**4.1.6.** **Problems resolved by the system** 9](#_Toc10748342)

[**4.2.** **Overall Description** 9](#_Toc10748343)

[**4.2.1.** **Product Functions** 9](#_Toc10748344)

[**4.2.2.** **Development environment** 10](#_Toc10748345)

[**4.2.3.** **Work Plan** 10](#_Toc10748346)

[**4.3.** **Interface Requirements** 10](#_Toc10748347)

[**4.3.1.** **User Interfaces** 10](#_Toc10748348)

[**4.4.** **System Requirements** 10](#_Toc10748349)

[**4.4.1.** **Functional requirements** 10](#_Toc10748350)

[**4.4.2.** **Non-functional Requirements** 12](#_Toc10748351)

[**5.** **Teamwork management** 12](#_Toc10748352)

[**6.** **Development environments and languages** 13](#_Toc10748353)

[**6.1.** **Development environment** 13](#_Toc10748354)

[**6.2.** **Development languages** 13](#_Toc10748355)

[**6.3.** **External libraries** 13](#_Toc10748356)

[**7.** **Budget estimate** 13](#_Toc10748357)

[**7.1.** **Funding source** 13](#_Toc10748358)

[**8.** **Assumptions, Constraints and Risks** 13](#_Toc10748359)

[**8.1.** **Risks** 13](#_Toc10748360)

[**9.** **Naïve Bayes** 15](#_Toc10748361)

[**10.** **software test description – STD** 17](#_Toc10748362)

[**11.** **Literature Review** 26](#_Toc10748363)

[**11.1.** **Introduction** 26](#_Toc10748364)

[**11.2.** **Social network** 26](#_Toc10748365)

[**11.3.** **Online offensive language** 26](#_Toc10748366)

[**11.4.** **Impact of online offensive language** 27](#_Toc10748367)

[**12.** **Finding offensive language in social network** 27](#_Toc10748368)

[**13.** **Machine learning** 27](#_Toc10748369)

[**13.1.** **What is machine learning** 27](#_Toc10748370)

[**13.2.** **Supervised learning** 28](#_Toc10748371)

[**14.** **Gantt chart** 29](#_Toc10748372)

[**15.** **UML** 30](#_Toc10748373)

[**15.1.** **Class Diagram** 30](#_Toc10748374)

[**15.2.** **Use Case Diagram** 31](#_Toc10748375)

[**15.3.** **Activity Diagram** 32](#_Toc10748376)

[**16.** **Algorithm analysis examples** 35](#_Toc10748377)

[**17.** **Website on PC and mobile** 37](#_Toc10748378)

[**18.** **Poster** 39](#_Toc10748379)

[**References** 41](#_Toc10748380)

# **English Summary**

**The problem**

Due to the tremendous growth and popularity of current web technologies and social networking, the online presence of the person has now become permanent. Twitter, one of the largest and most popular online social network currently (2019) has 333 million active users and it keeps on growing by millions every quarter

this high number of active users all from different cultures and educational background creates among other things a lot of toxic content (toxic content is a text that is abusive or offends an individual or group of individuals).

the main places this content can be found are:

* social networks
* texting and messaging applications
* online chats
* online games
* forums

each of the above produces a lot of offensive content daily it is impossible to check them all manually, a system is needed to help analyze and identify the toxic content and the toxic users who make it. and by doing so we can keep our online environment nicer and safer.

**Our solution**

we have developed and trained a learning algorithm that is based on **multinomial naïve Bayes** method. using a data set composed of 100k tweets we have taught the algorithm to classify offensive and non-offensive text. to validate our algorithm, we performed 10-fold cross-validation and found we had a **97% correct rate**.

as an interface to the algorithm using **Flask Framework for python,** we have built a free to use the website for users to analyze their texts in 2 ways:

1. **textual analysis** – using a synthetic parser called spacy we break the text to proper sentences then analyze them
2. **line by line analysis**- recommended for chats this option analyses the text by each line

these options allow easy tracking of offensive content inside different types of texts.

our vision is to give society an accurate, free and easy to use offensive language classifier to allow any website or application that wants to eliminate toxicity a way to do so.

**Keywords –** text classification, NLP, Machine learning, Multinomial naïve Bayes, Naïve Bayes classifier, Flask framework, 10-fold cross-validation, Spacy

|  |  |  |
| --- | --- | --- |
| סיווג טקסט פוגעני  בשיטת  **Multinomial naïve**  **Bayes** |  |  |
| **BS\_SCE-19** |
| **מנחים: ד"ר מרינה ליטבק** | ליאור שוסטרמן | רן אוזן |
| [liorsh5@ac.sce.ac.il](mailto:liorsh5@ac.sce.ac.il) | [ranuz@ac.sce.ac.il](mailto:ranuz@ac.sce.ac.il) |

# **Hebrew summery**

**תקציר**

הבעיה:

בשל גדילה עצומה בפופולאריות של טכנולוגיות הרשת העכשוויות הכוללות רשתות חברתיות , משחקי רשת , פורומים וכו' נוכחות האדם ברשת הפכה לדבר שהוא כמעט חובה

ניקח לדוגמא את טוויטר שנכון להיום (2019) יש לה כ-330 מיליון משתמשים פעילים וכמות המשתמשים ממשיכה לגדול במיליונים בכל שנה , וזו רק רשת חברתית אחת מתוך רבות אחרות בעלות מספר דומה של משתמשים.

המספר העצום הזה של המשתמשים נכנס כולו למקום אחד ומפגיש בין אנשים עם חינוך, תרבות, ורקע שונים. מפגש זה יכול לעתים ליצור גם חיכוכים ואף אלימות מילולית ושימוש בשפה פוגענית

המקומות העיקריים ברשת בהם ניתן למצוא שפה פוגענית ברשת הם:

* רשתות חברתיות
* אפליקציות לצ'אטים ושליחת הודעות
* משחקי רשת מרובי משתתפים
* פורומים
* וכו'

כל אחד מהמקומות הנ"ל "מייצר" ברשת בין היתר הרבה תוכן המכיל שפה פוגענית ואף אלימות מילולית אך אין זה אפשרי לעשות בדיקה ידנית של כל התוכן לכן יש צורך במערכת שתנתח את הטקסט באופן אוטומטי ותעזור לזהות את התוכן הפוגעני ואת האנשים שיוצרים אותו ובכך לשמור על הרשת סביבה נעימה בטוחה יותר

הפתרון:

אנו פיתחנו ולימדנו אלגוריתם לומד מבוסס על multinomial naïve bayes ומשתמש בדטה סט המורכב מכ-100 אלף טוייטים לסווג טקסט בין פוגעני ללא פוגעני, על מנת לבצע ולידציה לנכונות האלגוריתם ביצענו בדיקת **10-fold cross validation** ותוצאות הבדיקה היו ש**האלגוריתם צדק בכ-97%** מהמקרים בניתוח ציוצים פוגעניים

בתור ממשק לאלגוריתם בנינו אתר בעזרת Flask Framework המאפשר למשתמש לנתח את הטקסטים במספר דרכים

1. **ניתוח טקסטואלי** – על ידי שימוש במנתח תחבירי סינטטי אנו מפרקים את המשפטים ואז מנתחים כל אחד בפני עצמו
2. נ**יתוח לפי שורות** – מומלץ לצ'אטים , האופציה הזאת מפרקת את הטקסט לשורות ומנתחת כל שורה בנפרד

האפשרויות הנ"ל מאפשרות פירוק מעקב קל יותר אחר מציאת התוכן הפוגעני בטקסטים ארוכים יותר

החזון של הפרויקט הוא לתת לכל המעוניין (אפליקציות, אתרים, משחקים) מסווג טקסט פוגעני מדויק חינמי וקל לשימוש אשר יעזור לשמור על סביבת הרשת בטוחה יותר

**מילות מפתח –** סיווג טקסט Naïve Bayes classifier , Machine learning ,

, NLP, Flask framework 10-fold cross validation

# **Intro**

## **Project motivation**

The project idea was first given to us by our mentor Marina Litvak we saw this project a chance to learn many new things that were never got to during our degree (such as machine learning and NLP) and by dealing with those subjects and creating this project we have proved we are ready to go outside of the academy and into the tech world

## **Project purpose**

The purpose of our project was giving society a free accurate and easy to use offensive language classifier which will allow anyone to analyze any kind of text and the end result will hopefully be a safer online environment

## **Our main steps in this project**

* + Researching a machine learning method to build a classifier
  + Researching natural language processing
  + Collecting offensive and nonoffensive data set
  + Building the algorithm using the methods we researched
  + Processing the data set to match the algorithm needs
  + Testing the algorithm accuracy using 10-fold cross-validation
  + Building a site that acts as an interface to the algorithm

## **How the algorithm works**

Words are broken down into a vector representation

Our classifier received a data set containing approximately 80k examples of offensive and non-offensive text to help it distinguish offensive vectors from non-offensive ones

Using these vectors, the algorithm creates an offensive zone and a non-offensive zone

When given a new text the algorithm breaks it into a vector and based on its “zone” it decides the text class

# **SRS**

## **Introduction**

### **Purpose**

The purpose of this document is to give a detailed description of the requirements for the “offensive language detection” software. It will illustrate the purpose and complete declaration for the development of the system. It will also explain system constraints, interface, and interactions with other external applications.

### **Intended Audience and Reading Suggestions**

this document is intended for anyone interested in reading about our system (developers, project managers, teachers, etc.) for Sexual harassment detection.

### **Product Scope**

The” offensive language detection” application is a software-based system that its purpose is to help mitigate offensive language. We plan on teaching a learning algorithm to detect the offensive language in a given text and give a Boolean decision if a text is offensive or not

### **Main user/client**

The main user of the system will be the anyone looking to analyze text for offensive language. The client for our system is intended to be social network owners, messaging applications, or any website with a reply option. these will require our service in order to keep their online community safer from offensive language

### **Objectives**

The objectives of the <O.L.D> are as follows:

* Building an offensive language database
* Achieving 80% or higher correct decision from the learning algorithm
* building a free to use website which implements the learning algorithm
* giving all online users a safer environment in the future

The future goals of the O.L.D system are to work with more environment and to allow automatic detection of offensive language in any web site or messaging application.

### **Problems resolved by the system**

The main problem that the system resolves the automation of finding an offensive language. Our system automates the decision-making process and by that opens a better way to analyze text containing offensive language in social networks.

## **Overall Description**

### **Product Functions**

the 3 major functions of our system

* Automate finding offensive text
* Building a continuously learning algorithm that improves over time
* building an ever-growing data set of offensive text

### **Development environment**

we will use python as our working environment also we will use libraries such as Pandas, NumPy for the learning algorithm part and for our website backend we will use Flask framework.

### **Work Plan**

|  |  |
| --- | --- |
| Goals | Month |
| Completing documentation (SRS, charter, risk management, literature review) | December |
| Initial system development   * building a training dataset | January |
| Building and training the learning algorithm  Completing a prototype of the system  Building a website to allow usage of the algorithm | February- March |
| Completing the development and testing of the system | April |
| Completing all documentation and presentation material for the final project review | May |
| Present the final project | June |

## **Interface Requirements**

### **User Interfaces**

the user interface in our website will be built in a simple and minimal manner just to show the algorithm in action to our users and/or potential clients

## **System Requirements**

### **Functional requirements**

**REQ-1**

|  |  |  |  |
| --- | --- | --- | --- |
| O.L.D | System name | 001 | Id |
| TBD | Urgency | Offensive Data Set | Requirement name |
| TBD | End date | 13.11.2018 | Request date |
|  | | | |
| The system will have a Data set of offensive tweets ,replies, posts etc.’ containing offensive words to teach the learning algorithm. | | Detailed description of implementation method | |

**REQ-2**

|  |  |  |  |
| --- | --- | --- | --- |
| O.L.D | System name | 002 | Id |
| TBD | Urgency | Regular Data Set | Requirement name |
| TBD | End date | 13.11.2018 | Request date |
|  | | | |
| The system will have a Data set of regular (non-offensive no profane words no hate speech etc.’) tweets, replies, posts etc.’ to teach the learning algorithm. | | Detailed description of implementation method | |

**REQ-3**

|  |  |  |  |
| --- | --- | --- | --- |
| O.L.D | System name | 003 | Id |
| TBD | Urgency | Retrieve data using API | Requirement name |
| TBD | End date | 13.11.2018 | Request date |
|  | | | |
| The system will draw tweets, posts, replies from twitter to analyze them for offensive content and to continuously test and improve our learning algorithm | | Detailed description of implementation method | |

**REQ-4**

|  |  |  |  |
| --- | --- | --- | --- |
| O.L.D | System name | 004 | Id |
| TBD | Urgency | Final output | Requirement name |
| TBD | End date | 13.11.2018 | Request date |
|  | | | |
| The final output of the system will be a Boolean decision if a post/reply is offensive or not, it will base its decision on all the information gathered in previous phases | | Detailed description of implementation method | |

**REQ-5**

|  |  |  |  |
| --- | --- | --- | --- |
| O.L.D | System name | 005 | Id |
| TBD | Urgency | Supervised learning algorithm | Requirement name |
| TBD | End date | 13.11.2018 | Request date |
|  | | | |
| The system will have a learning algorithm that will be based on a supervised learning approach to detect offensive text in the form of tweets, replies, posts | | Detailed description of implementation method | |

**REQ-6**

|  |  |  |  |
| --- | --- | --- | --- |
| O.L.D | System name | 006 | Id |
| TBD | Urgency | Learning algorithm | Requirement name |
| TBD | End date | 13.11.2018 | Request date |
|  | | | |
| The learning algorithm will identify texts containing offensive language and will have a correct decision rate of at least 80% | | Detailed description of implementation method | |

**REQ-7**

|  |  |  |  |
| --- | --- | --- | --- |
| O.L.D | System name | 006 | Id |
| TBD | Urgency | website | Requirement name |
| TBD | End date | 13.11.2018 | Request date |
|  | | | |
| the user's interface to the algorithm will be a simple website that will allow users to enter text to be analyzed by the learning algorithm | | Detailed description of implementation method | |

### **Non-functional Requirements**

* **Performance Requirements**

the learning process will not exceed 5 seconds

* **Safety Requirements**

at the moment we don’t have any safety requirements

* **Security Requirements**

to protect the privacy of offender and/or victim we will keep all data found private and we will not share or post any of the private information with anyone.

# **Teamwork management**

* For tasks management, we used an Application called Meistertask that lets you organize all the team’s task’s and see exactly at what point of the task a person is and at what step of the development a task is.
* For code management, we used GitHub which allows easy code share and is vastly used in the industry
* Weekly meetings were performed to view the progress of tasks and requirements

# **Development environments and languages**

## **Development environment**

* we used VS code IDE as our code development environment which allowed us to easily see errors in the code and debug them

## **Development languages**

* the algorithm was developed in Python
* The frontend of the website was developed using HTML CSS and JavaScript and the backend was developed in Python

## **External libraries**

* Pandas –open source library providing high-performance, easy-to-use data structures and data analysis tools for the [Python](https://www.python.org/) programming language.
* Flask Framework a micro [web framework](https://en.wikipedia.org/wiki/Web_framework) written in [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) used for web development
* Spacy – a free open-source library for Natural Language Processing in Python

# **Budget estimate**

## **Funding source**

We have received a grant for a social project.

Other than that, our project is a final year college project, therefore, we don’t require any additional funds.

# **Assumptions, Constraints and Risks**

## **Risks**

**Risk Identification and analysis**

1. Requirements are on a High level and some are ambiguous

Because we are not yet sure on how we will implement and develop our system to the lowest level, our requirements are on a higher level which means each represents a big component in our system.

1. Needed skills are based on self-learning

Many of the technologies and tools we will need to use to build for our system are new for us and thus need to be researched and learned before used or implemented.

1. Trouble keeping with Schedule

Despite having prepared a work plan we still might fall behind on schedule and development of the system especially because of risk (2).

**Risk evaluation**

Risk 1 is likely and low severity

Risk 2 is likely and medium severity

Risk 3 is likely and medium-high severity

**Risk treatment and response**

Risk 1 - As we learn and implement our current high-level requirements, we will break them down into smaller lower level requirements and avoid ambiguous in the requirements thus mitigating the risk.

Risk 2 - the project team will use their free time to self-learn the technologies and tools needed to build the system using sites like StackOverflow/YouTube/Udemy/freecodecamp thus mitigating the risk.

Risk 3 - To prevent falling behind schedule the team will use Meistertask tool to keep track of current work and assignments also the team will always try to be a month ahead of schedule and schedule a meeting with the project instructor once every 2 weeks at least to review project progress.

**Risk monitor and review**

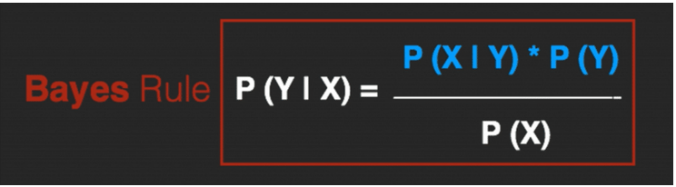
To monitor our risks, we will perform a team meeting 3 times a week to review progress made on learning/implementing new things and a meeting with project instructor at least once every 2 weeks to review progress made and make new goals, we believe these meetings will help us avoid/mitigate all the risks

# **Naïve Bayes**

**Naïve Bayes classifier:**

The Naive Bayes Classifier technique is based on the so-called Bayesian theorem and is particularly suited when the dimensionality of the inputs is high. Despite its simplicity, Naive Bayes can often outperform more sophisticated classification methods.

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes’ Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.



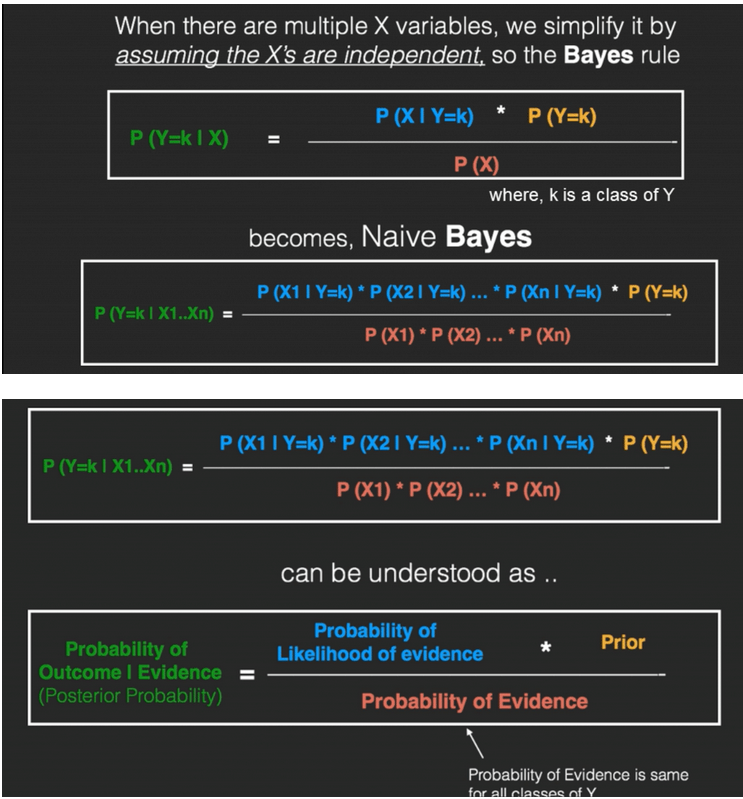
**The Bayes Rule:**

**The Naïve Bayes:**

The Bayes Rule provides the formula for the probability of Y given X. But, in real-world problems, you typically have multiple X variables.

When the features are independent, we can extend the Bayes Rule to what is called Naive Bayes.

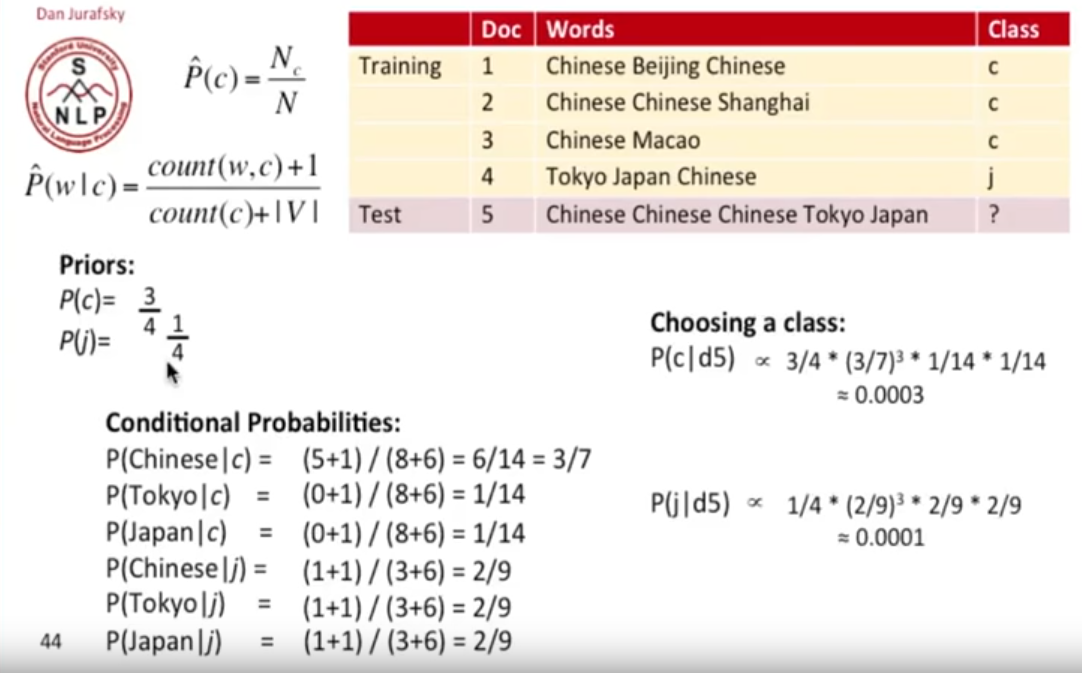
It is called ‘Naive’ because of the naive assumption that the X’s are independent of each other. Regardless of its name, it’s a powerful formula.



**Multinomial Naïve Bayes:**

Multinomial Naive Bayes is a specialized version of Naive Bayes that is designed more for text documents. Whereas simple naive Bayes would model a document as the presence and absence of particular words, multinomial naive Bayes explicitly models the word counts and adjusts the underlying calculations to deal with it.

Example:



# **software test description – STD**

1. 10-fold cross-validation
2. load test
3. beta/acceptance testing
4. sanity testing
5. integration testing

**10-fold cross-validation**

(k-fold, where k=10 is most popular)

is commonly used in applied machine learning to compare and select a model for a given predictive modeling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

**The procedure:**

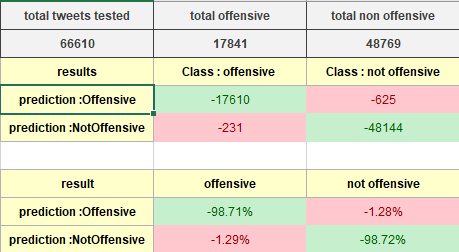
1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. For each unique group:

* Take the group as a holdout or test data set
* Take the remaining groups as a training data set
* Fit a model on the training set and evaluate it on the test set
* Retain the evaluation score and discard the model

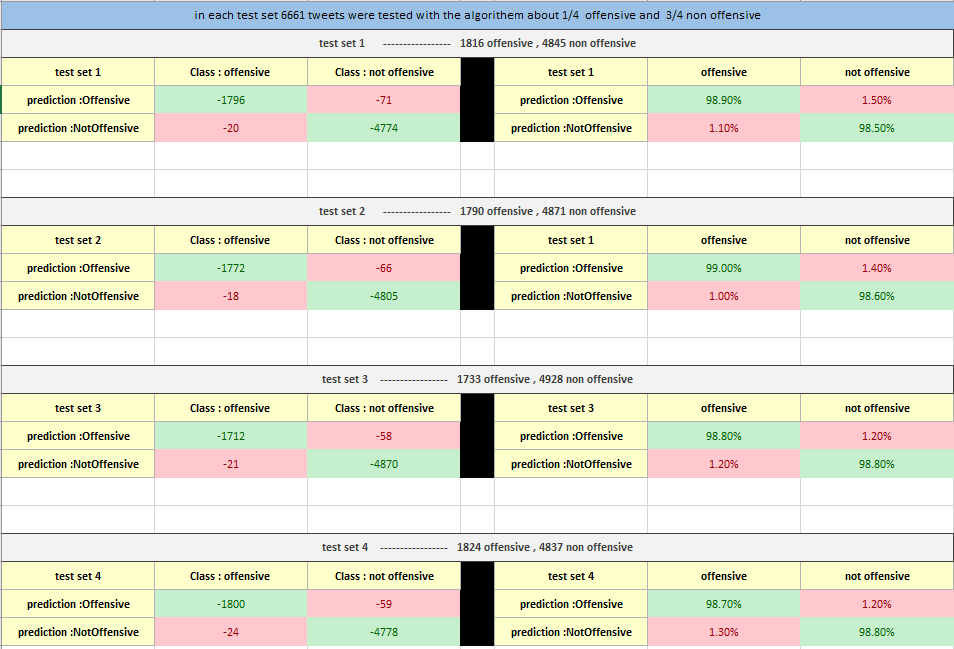
1. Summarize the skill of the model using the sample of model evaluation scores

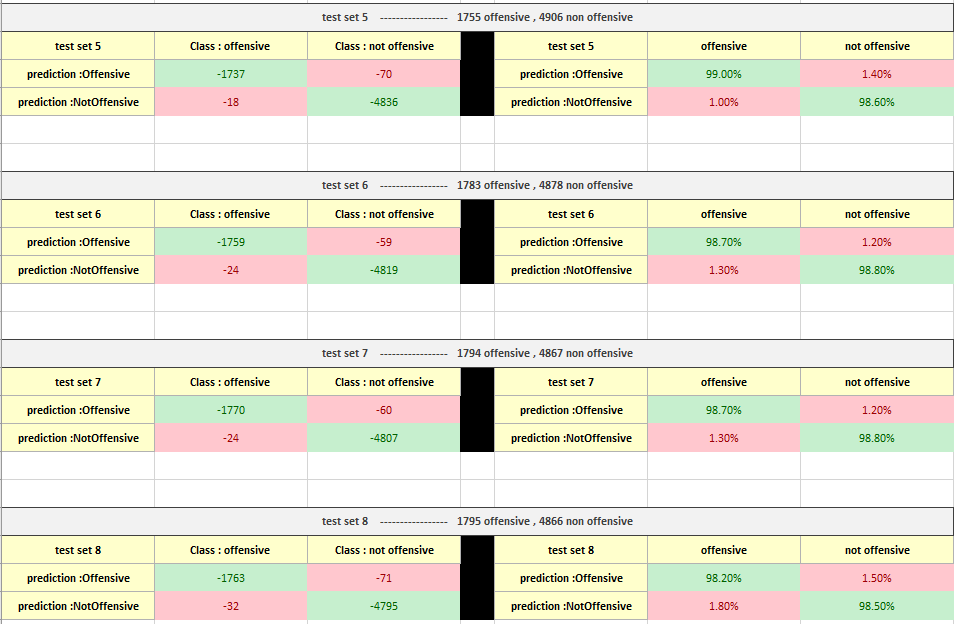
**Our 10-fold cross validation result**

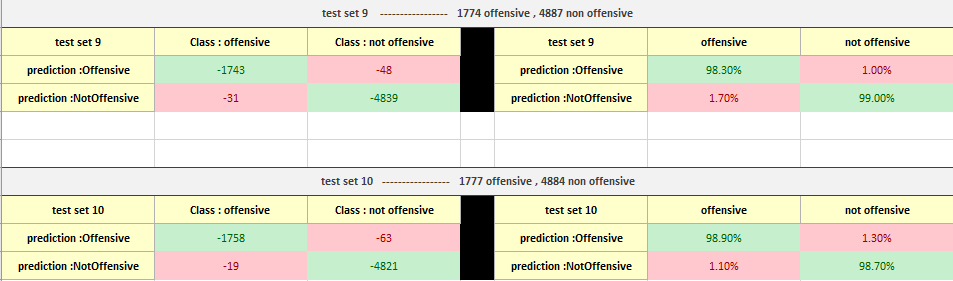
**Final result**



**Detailed result**







**Load testing**

Load testing is a kind of [Performance Testing](https://www.guru99.com/performance-testing.html) which determines a system's performance under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously.

**This testing usually identifies -**

* The maximum operating capacity of an application
* Determine whether the current infrastructure is sufficient to run the application
* Sustainability of application with respect to peak user load
* Number of concurrent users that an application can support, and scalability to allow more users to access it.

**Our load testing:**

The only part of our system that is vulnerable to load is the site we build as an interface to the algorithm

To check how the system handles the load, we have tested it ourselves by using 6 devices to send multiple requests to the local host in addition as part of our social project scholarship we performed a live demo of the entire system and allowed multiple users to connect to the site and query their texts. The system handled both cases well and did not crash meaning we were unable to find the exact number of concurrent users, but we know for sure it will hold for future presentation of the system.

**Sanity testing**

Sanity testing is a kind of Software Testing performed after receiving a software build, with minor changes in the code, or functionality, to ascertain that the bugs have been fixed and no further issues are introduced due to these changes. The goal is to determine that the proposed functionality works roughly as expected. If sanity test fails, the build is rejected to save the time and costs involved in more rigorous testing.

**Our sanity testing:**

After each requirement/feature, we added to the system we performed a sanity check to verify we did not create new bugs/issues with the current version

**integration testing**

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

**Our integration testing:**

In our project, we have integrated between the algorithm and the interface to the algorithm meaning the site had to work with python code we wrote

To test that we….?

**functionality testing**

Functional Testing is defined as a type of testing which verifies that each function of the software application operates in conformance with the requirement specification. This testing mainly involves black box testing and it is not concerned about the source code of the application.

**Our functionality testing:**

We have tested the site interface and inputs and outputs as follow:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | | | | **Test #** |
| Functionality – home button | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected Result** | **Step description** | **Steps:** |
|  | passed | web site launches quickly | Go to offensiveclassifier.com |  |
|  | passed | Home page loads quickly | Push home button |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2 | | | | **Test #** |
| Functionality – text box input | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected Result** | **Step description** | **Steps:** |
|  | passed | web site launches quickly | Go to offensiveclassifier.com |  |
|  | passed | The site shows red line over the line containing the offensive text  And the entire text that was entered in the text box is shown on screen | Insert a text with offensive words (for example “fuck you all”.) and press send button |  |
|  | passed | Home page loads quickly | Press the home button |  |
|  | passed | The text is classified as “nonoffensive” and is broken into proper English sentences  And the entire text that was entered in the text box is shown on screen | Insert a none offensive long text (at least 2 sentences) and press send button |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | | | | **Test #** |
| Functionality – textual analysis | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected Result** | **Step description** | **Steps:** |
|  | passed | web site launches quickly | Go to offensiveclassifier.com |  |
|  | passed | Text is broken into lines each a sentence | In the text box insert a long text with at least 5 sentences, select textual analysis, and press send |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | | | | **Test #** |
| Functionality – line by line analysis | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected Result** | **Step description** | **Steps:** |
|  | passed | web site launches quickly | Go to offensiveclassifier.com |  |
|  | passed | Text is broken into lines each a line of the chat | In the text box insert a text that resembles a chat conversion with each row being |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5 | | | | **Test #** |
| Functionality – upload a file | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected Result** | **Step description** | **Steps:** |
|  | passed | web site launches quickly | Go to offensiveclassifier.com |  |
|  | passed | Upload a file page loads quickly | Push upload a file button |  |
|  | Passed | file uploaded is cleared | Upload a file with a type different then TXT and press send |  |
|  | passed | Text in the file is analyzed in a textual analysis manner | Upload a TXT file and press send |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | | | | **Test #** |
| Functionality – about us page | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected result** | **Step description** | **Steps:** |
|  | passed | passed | web site launches quickly |  |
|  | passed | About us page loads quickly | Push about us button |  |
|  | passed | How it works page loads quickly | Push more about how it works click here |  |
|  | Passed | Contact us page loads quickly | Push contact us click here |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | | | | **Test #** |
| Functionality – how it works page | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected result** | **Step description** | **Steps:** |
|  | passed | passed | web site launches quickly |  |
|  | passed | how it works page loads quickly | Push how it works button |  |
|  | passed | The file opens and displays correctly | Download 10fldcrsval.xlsx and open it |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 8 | | | | **Test #** |
| Functionality – contact page | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected result** | **Step description** | **Steps:** |
|  | passed | passed | web site launches quickly |  |
|  | passed | contact us page loads quickly | Push contact button |  |
|  | passed | Lior linked in page opens | Press Lior’s Linked in link |  |
|  | passed | Ran linked in page opens | Press Ran’s Linked in link |  |
|  | passed | SCE Facebook page opens | Press bottom page face button |  |
|  | passed | SCE Instagram page opens | Press bottom page Instagram button |  |
|  | passed | SCE YouTube page opens | Press bottom page YouTube button |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 9 | | | | **Test #** |
| Functionality – contact us page | | | | **Test purpose** |
| A device (mobile/pc) connected to same WIFI as localhost | | | | **Preconditions** |
| **Actual result** | **Passed / failed** | **Expected result** | **Step description** | **Steps:** |
|  | passed | web site launches quickly | Open browser and go to “local\_host\_ip” |  |
|  | passed | Pages display correctly, and GUI is intact | Push |  |

# **Literature Review**

## **Introduction**

Due to the tremendous growth and popularity of current web technologies and social networking, the online presence of the person has become permanent now. Through social networking connections, people usually express feelings, opinions, and emotions

Those opinions and emotions can be negative and sometimes even abusive thus becoming cyberbullying or harassing Therefore, measures need to be put in place to monitor and detect potentially harmful online activities.

Twitter, one of the largest and most popular online social network (OSN) has of today (2018) 335 million active users and keeps on growing by millions every quarter

because of this high number of active users, the number of potentially harmful online activities rises as well, it is impossible to do a manual check for all the posts on the social network because of the large number of users, our mission in this project is to automate finding offensive language in a given text.

## **Social network**

Where social networks were once a niche activity for a teenager. Now social networking absorbs tens of millions of internet users. People of all ages races and ethnic background interact on social networks. Ranging from social communities and discussion groups to recommendation engines, tagging systems, mobile social networks, games, and virtual worlds

Many people believe that the growing use of social networks will have an impact on society worldwide networks will introduce new ideas and cultures while other think that spending too much time in front of a computer keyboard will have a negative impact. They question wheatear young people will develop adequate face-to-face communication skills. Others point out that as people feel less inhibited behind a computer screen, they may be more likely to participate in cyberbullying. Some young people especially girls, may post increasingly sexual pictures of themselves and add a graphic sexual comment to their profiles. Some worry that this online sexualization may lead to promiscuous behavior offline and attract dangerous attention from predators.

## **Online offensive language**

Online offensive language is a pervasive and pernicious problem. Techniques like natural language processing and machine learning are promising approaches for identifying the offensive abusive language. offensive language can occur in many ways in our project we will focus on identifying any offensive language using naïve Bayes focused learning algorithm

## **Impact of online offensive language**

Offensive language can hurt people in many ways. The experience and impact of online offensive language are unique to the individual and can be felt both in the short-term but also can have long-term impacts on mental health and wellbeing. Long term impacts can be amplified because of re-victimization if the content is re-shared online, or because the initial trauma of the incident resurfaces much later. It is important to recognize that there is no single way that a person may experience online offensive language and that it might also affect others who witness it.

# **Finding offensive language in social network**

social networking sites are being widely used to express, discuss, exchange views and opinions on various topics. it has been often observed that user conversations sometimes quickly derail and become inappropriate such as hurling abuses, passing rude and discourteous comments on individuals or certain groups/communities. Detecting inappropriate language is challenging due to various natural language phenomena such as spelling mistakes and variations, polysemy, contextual ambiguity, and semantic variations.

A given text is defined as offensive if its intent is any of the following –

(a) rude or discourteous or exhibiting lack of respect toward certain individuals or group of individuals.

(b) to cause or capable of causing harm (to oneself or others)

(c) has extreme verbal violence.

# **Machine learning**

## **What is machine learning**

Machine learning is a category of [algorithm](https://whatis.techtarget.com/definition/algorithm) that allows software applications to become more accurate in predicting outcomes without being explicitly programmed. The basic premise of machine learning is to build algorithms that can receive input data and use [statistical analysis](https://whatis.techtarget.com/definition/statistical-analysis) to predict an output while updating outputs as new data becomes available.

The processes involved in machine learning require searching through data to look for patterns and adjusting program actions accordingly. Many people are familiar with machine learning from shopping on the internet and being served ads related to their purchase. This happens because [recommendation engines](https://whatis.techtarget.com/definition/recommendation-engine) use machine learning to personalize online ad delivery in almost real time.

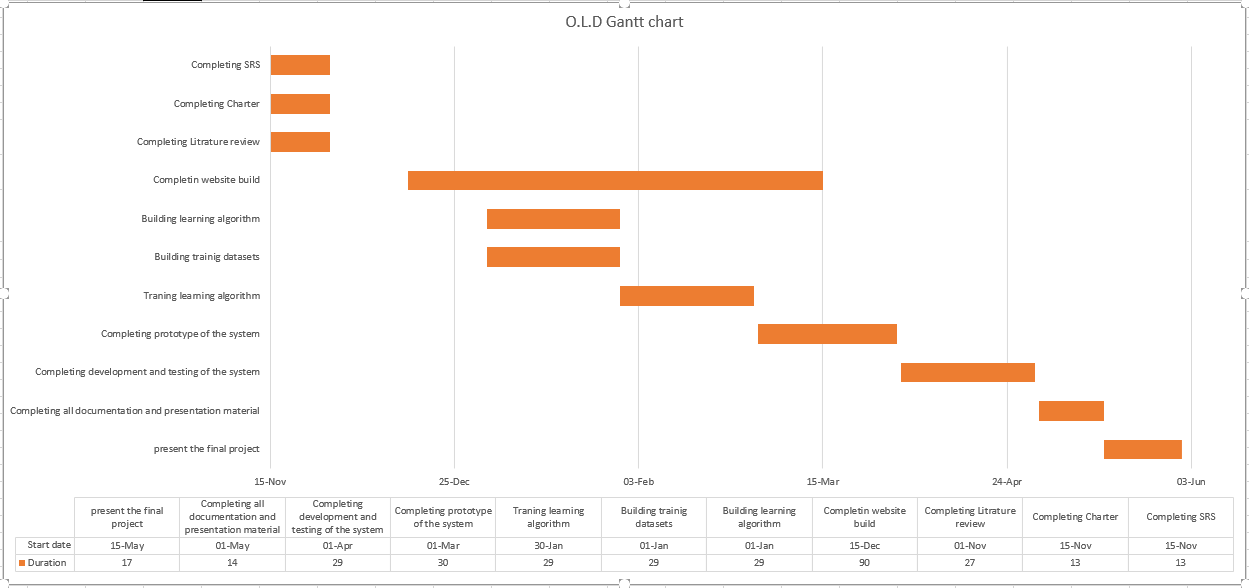
## **Supervised learning**

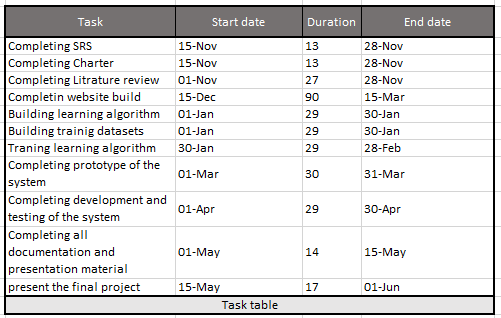
Supervised techniques adapt the model to reproduce outputs known from a training set (e.g. [recognize car types on photos](https://www.netguru.co/blog/machine-learning-and-augmented-reality-combined-in-one-sleek-mobile-app-how-we-built-car-lens)). In the beginning, the system receives input data as well as output data. Its task is to create appropriate rules that map the input to the output. The training process should continue until the level of performance is high enough. After training, the system should be able to assign output objects which it has not seen during the training phase. In most cases, this process is fast and accurate.

There are two types of Supervised Learning techniques: Regression and Classification. Classification separates the data, Regression fits the data.

To put it simply, we train an algorithm and at the end pick the model that best predicts some well-defined output based on the input data.

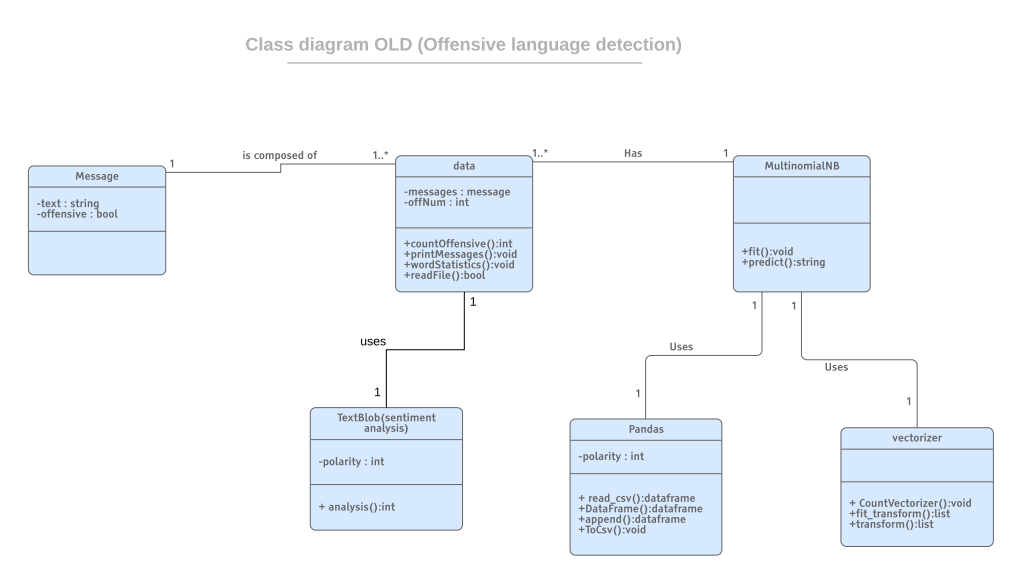
# **Gantt chart**



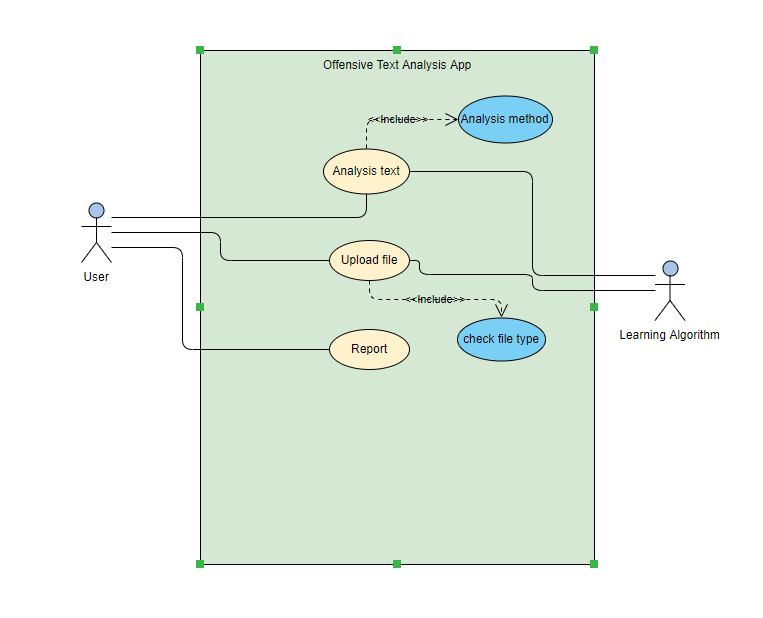


# **UML**

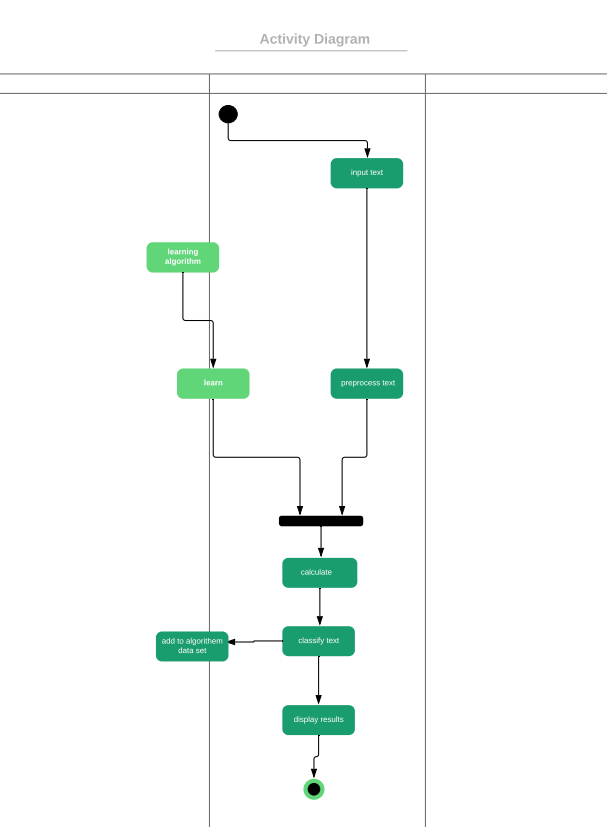
## **Class Diagram**

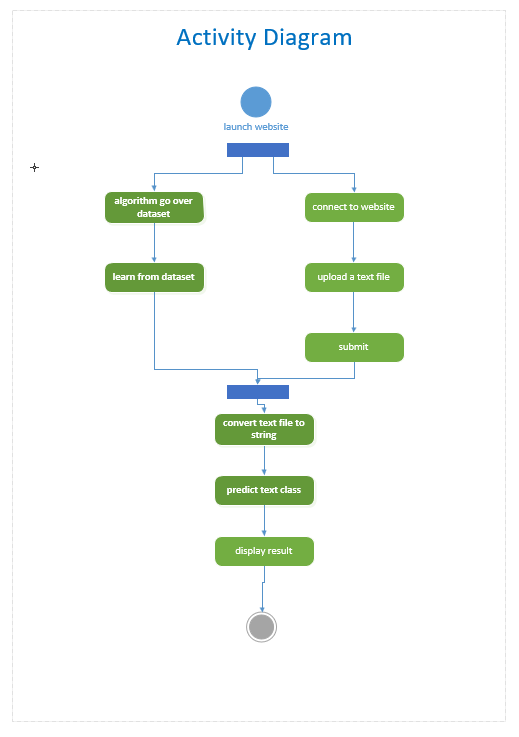


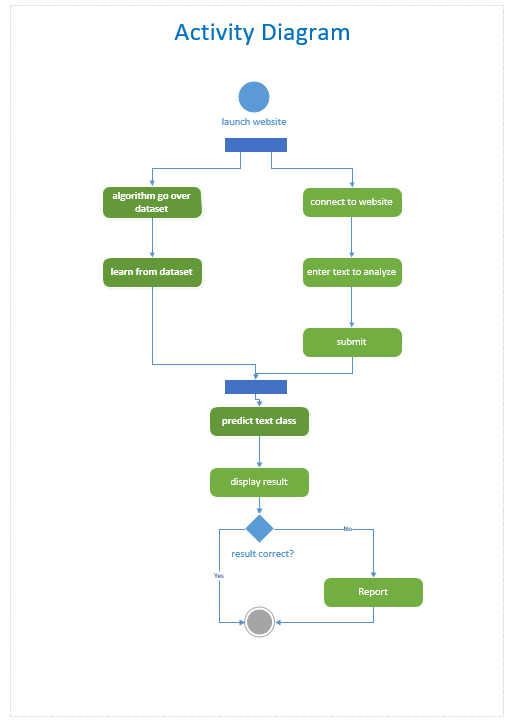
## **Use Case Diagram**



## **Activity Diagram**



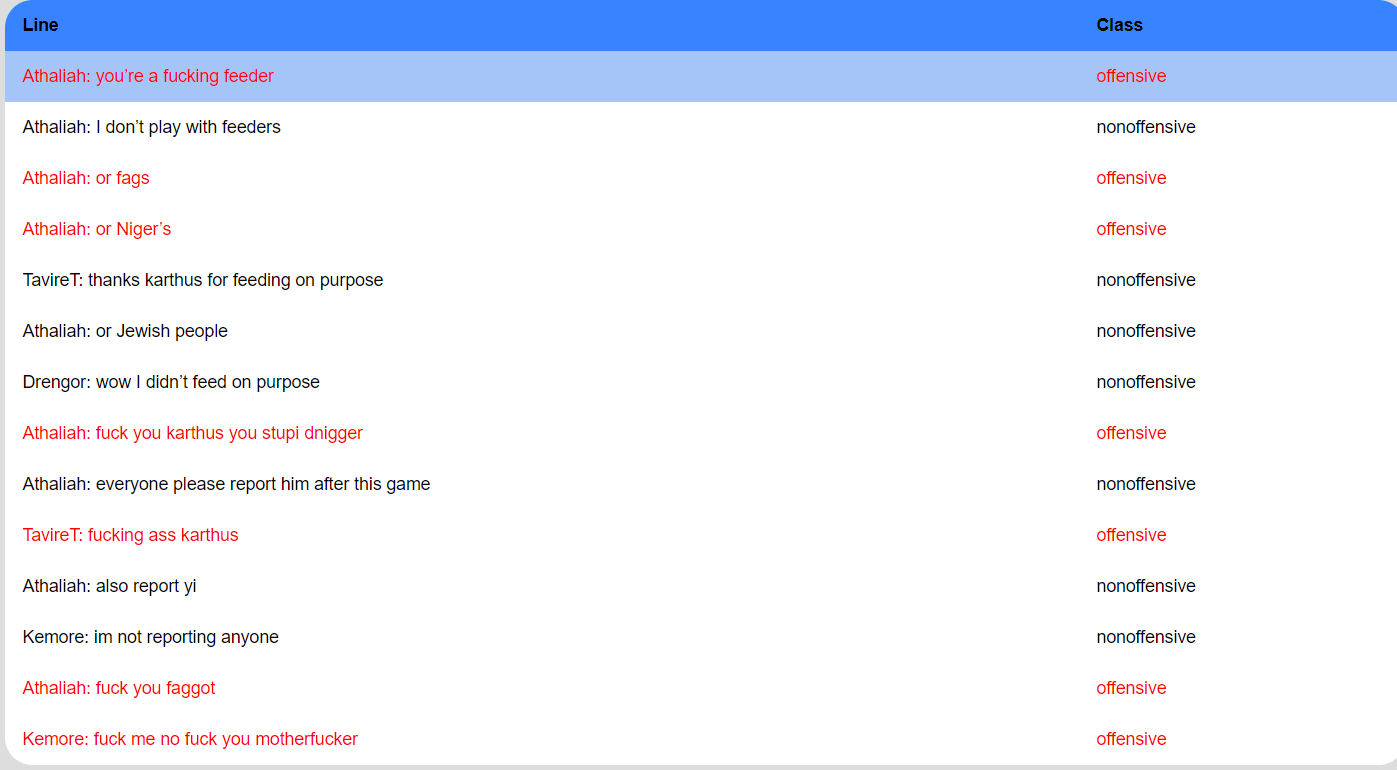




# **Algorithm analysis examples**

**Algorithem chat analysis example**





**Algorithem textual analysis example**

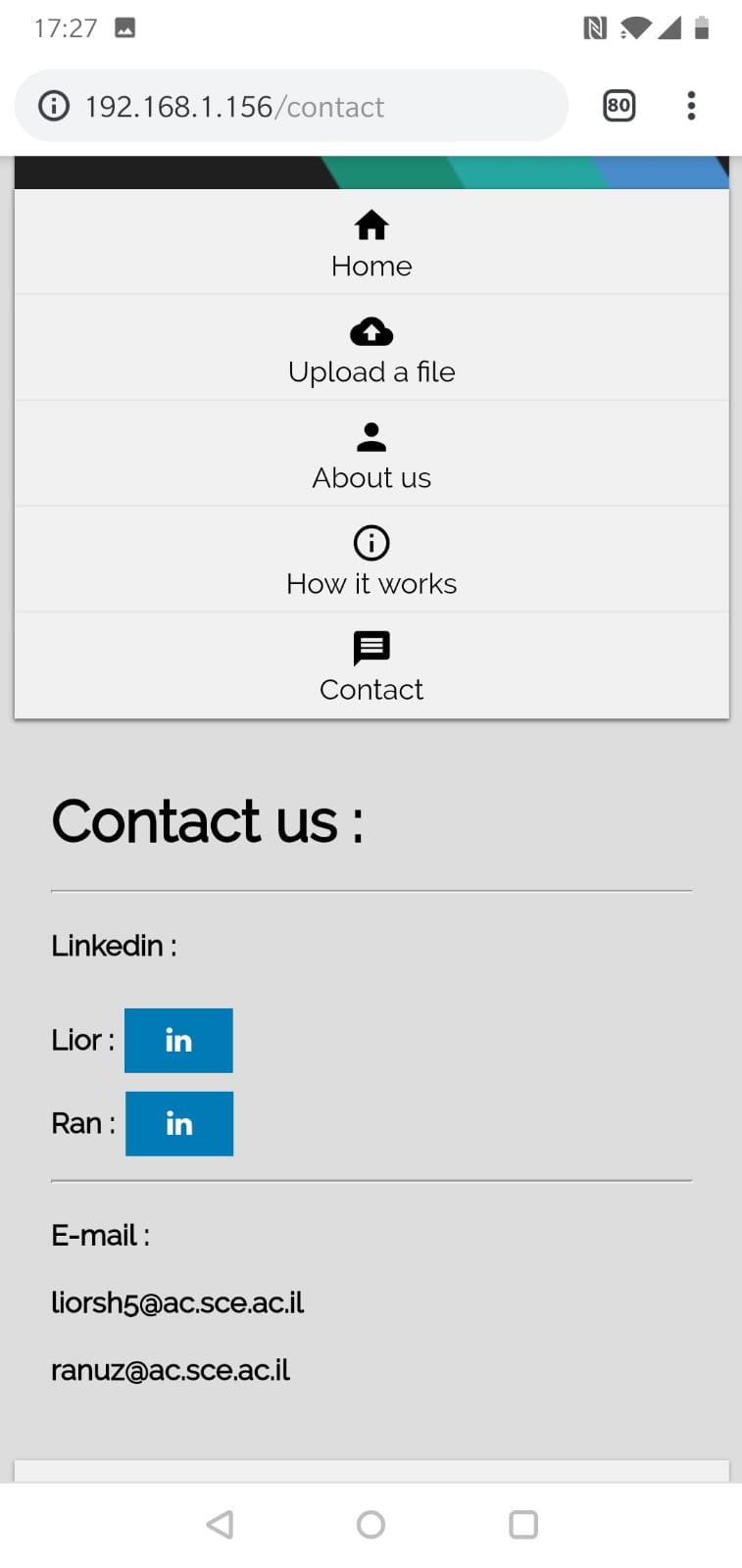
A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

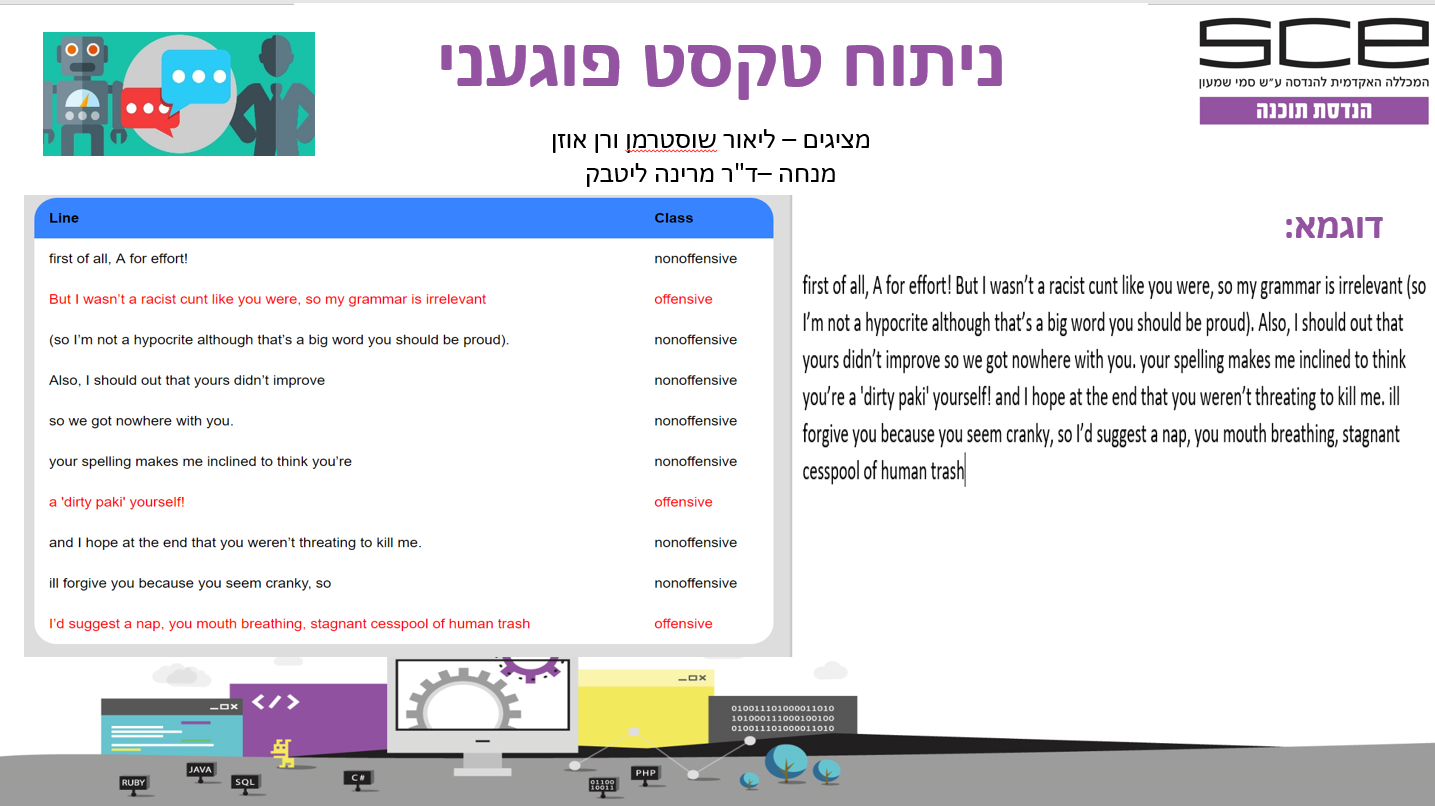
# **Website on PC and mobile**





# **Poster**







# **References**

1. Agichtein, Eugene, et al. "Finding high-quality content in social media." Proceedings of the 2008 international conference on web search and data mining. ACM, 2008.

2. N. Vanetik, M. Litvak, E. Levi and A. Vashchenko “Twitter event detection, analysis and summarization” (2017)

3. Kansara, Krishna B., and Narendra M. Shekokar. "A framework for cyberbullying detection in social network." International Journal of Current Engineering and Technology 5.1 (2015): 494-498

4. Smith, Peter K., et al. "An investigation into cyberbullying, its forms, awareness and impact, and the relationship between age and gender in cyberbullying." Research Brief No. RBX03-06. London: DfES (2006)

5. Edupuganti, Venkatesh. Harassment Detection on Twitter using Conversations. Diss. Wright State University, 2017.‏

6. Hammer, Hugo Lewi, et al. "Improving Classification of Tweets Using Linguistic Information from a Large External Corpus." Industrial Networks and Intelligent Systems (2016).

7. Mooney, Carla. Online social networking. Greenhaven Publishing LLC, 2009.‏

8. Gudivada, Venkat N., et al. "Information retrieval on the world wide web." IEEE Internet Computing 1.5 (1997): 58-68.‏

9. Blackwell, Lindsay, et al. "Classification and Its Consequences for Online Harassment: Design Insights from HeartMob." PACMHCI 1. CSCW (2017): 24-1

10. Wang, Wenbo, et al. "Cursing in english on twitter." Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing. ACM, 2014.

11. Yenala, Harish, et al. "Deep learning for detecting inappropriate content in text." International Journal of Data Science and Analytics 6.4 (2018): 273-286.

12. Kibriya, Ashraf M., et al. "Multinomial naive bayes for text categorization revisited." Australasian Joint Conference on Artificial Intelligence. Springer, Berlin, Heidelberg, 2004.‏