

Governors State University

## OPUS Open Portal to University Scholarship

---

All Capstone Projects

Student Capstone Projects

---

Fall 2023

### Classification of Online Toxic Comments Using Machine Learning Algorithms

Shiva Kumar Bolu

Follow this and additional works at: <https://opus.govst.edu/capstones>

---

For more information about the academic degree, extended learning, and certificate programs of Governors State University, go to [http://www.govst.edu/Academics/Degree\\_Programs\\_and\\_Certifications/](http://www.govst.edu/Academics/Degree_Programs_and_Certifications/)

Visit the [Governors State Computer Science Department](#)

This Capstone Project is brought to you for free and open access by the Student Capstone Projects at OPUS Open Portal to University Scholarship. It has been accepted for inclusion in All Capstone Projects by an authorized administrator of OPUS Open Portal to University Scholarship. For more information, please contact [opus@govst.edu](mailto:opus@govst.edu).

# **Classification of Online Toxic Comments Using Machine Learning Algorithms**

By

**Shiva Kumar Bolu**

B.Tech., Vignan's institute of technology and Aero Engineering, 2015

GRADUATE CAPSTONE SEMINAR PROJECT

Submitted in partial fulfillment of the requirements

For the Degree of Master of Science,

With a Major in Computer Science



Governors State University  
University Park, IL 60484

2023

## **ABSTRACT**

This application classification of online toxic comments using machine learning algorithms project was implemented for the Graduate Capstone Seminar Project for the Master of Science Degree with a Major in Computer Science. In this project, different Machine learning algorithms are used to find toxic comments. In discussions, toxic comments are disrespectful and abusive which makes other people leave the discussion. So, many social networking sites difficult to promote discussions effectively. The main aim of the project is to examine the data of online harassment and classify it into different labels to find toxicity correctly. In this project, we are going to use six machine learning algorithms, apply them to our data and find which algorithm is best by analyzing evaluation metrics for toxic comments classification. We will aim to examine the toxicity with high accuracy to limit its adverse effects and help organizations take the necessary steps.

# Table of Content

<b>1</b>	<b><i>Project Description</i></b> .....	<b><i>1</i></b>
1.1	Competitive Information .....	1
1.2	Relationship to Other Applications/Projects .....	1
1.3	Assumptions and Dependencies .....	1
1.4	Future Enhancements.....	1
1.5	Definitions and Acronyms .....	1
<b>2</b>	<b><i>Project Technical Description</i></b> .....	<b><i>1</i></b>
2.1	Application Architecture .....	1
2.2	Interactions with other Applications .....	2
2.3	Capabilities .....	2
2.4	Risk Assessment and Management.....	2
<b>3</b>	<b><i>Project Requirements</i></b> .....	<b><i>2</i></b>
3.1	Identification of Requirements .....	2
3.2	Operations, Administration, Maintenance and Provisioning (OAM&P) .....	2
3.3	Release and Transition Plan.....	3
<b>4</b>	<b><i>Project Design Description</i></b> .....	<b><i>3</i></b>
<b>5</b>	<b><i>Internal/external Interface Impacts and Specification</i></b> .....	<b><i>3</i></b>
<b>6</b>	<b><i>Design Units Impacts</i></b> .....	<b><i>8</i></b>
6.1	Functional Area A/Design Unit A .....	8
6.1.1	<b><i>Functional Overview</i></b> .....	8
6.1.2	<b><i>Impacts</i></b> .....	8
<b>7</b>	<b><i>Open Issues</i></b> .....	<b><i>8</i></b>
<b>8</b>	<b><i>Acknowledgements</i></b> .....	<b><i>8</i></b>
<b>9</b>	<b><i>References</i></b> .....	<b><i>9</i></b>
<b>10</b>	<b><i>Appendices</i></b> .....	<b><i>9</i></b>

## ***1 Project Description***

At initial days of internet, people used to communicate through emails only. In those days it is difficult to classify email as spam or not. As days passed, with the growth of social media platforms it is extremely important to classify text as positive or negative to prevent harm to society. In recent times many people have been arrested by posting abusive comments on others. So, it is necessary to build a system that to detect toxic comments before it is publishing because these negative comments may affect people adversely. So in this project six machine learning algorithms are used to classify text and to find the toxicity. In this project we use logistic regression, random forest, SVM classifier, naive bayes, decision tree, and KNN classification to solve the problem of text classification. So we apply machine learning algorithm on data sets and calculate and compare the results.

### ***1.1 Competitive Information***

This project Classification of online toxic comments using machine learning algorithms is used to classify toxic comments. The system classifies given data into 6 categories i.e. threat, insult, toxic, severe toxic, obscene, and identity hate. Different machine learning algorithms are applied to data sets acquired from Kaggle.com. We apply different machine learning algorithms on data sets to solve the text classification and find the best algorithm by analyzing evaluation metrics.

### ***1.2 Relationship to Other Applications/Projects***

There are some efforts to increase online safety through crowd-sourcing techniques but failed to find toxicity. Wulczyn introduced crowd-sourcing and machine-learning techniques to find possible attacks. Convolutional Neural Networks (CNN) are used to find toxic comments on online data but fail to implement text classification. So, in our project, we have used different machine learning algorithms for text classification on online comments. The Kaggle data set is used to test the system.

### ***1.3 Assumptions and Dependencies***

We have used the Kaggle data set. We classified online comments into 6 six categories only. We tested the system with a given data set where 90% of data is non-toxic so we may get higher accuracy than the real-time. KNN algorithm gave 54% accuracy but in real time we get 83-84% accuracy. So algorithm accuracy may vary based on text comments in the data set. So, instead of selecting accuracy as a parameter we have selected hamming loss as the metric value.

### ***1.4 Future Enhancements***

The project can be updated shortly if any new requirement arises because it is flexible in terms of expansion. In the future, we can classify text into more categories. We can test with different data sets not only Kaggle to get better accuracy of algorithms.

### ***1.5 Definitions and Acronyms***

**Data Set:** Raw data downloaded from Kaggle.com

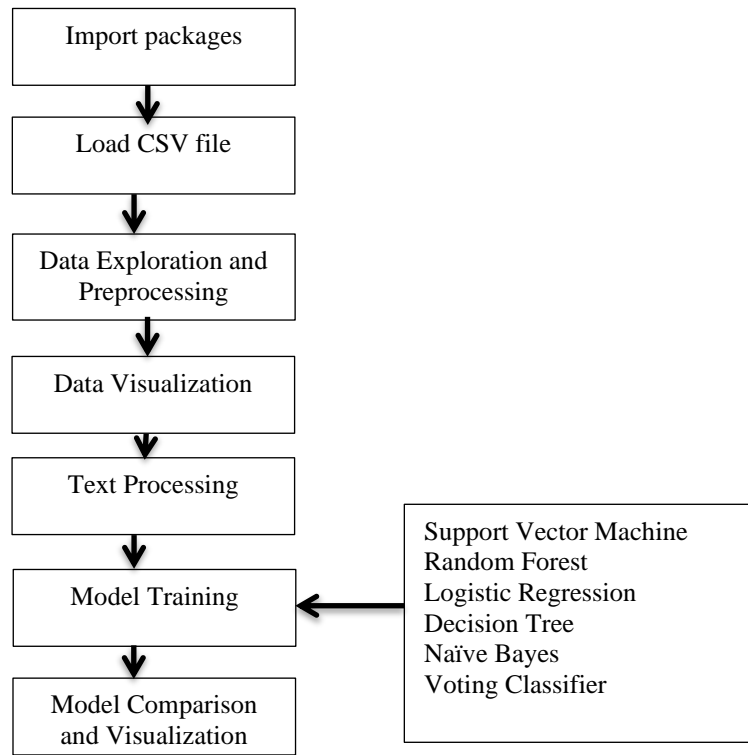
**Machine learning:** Predicting current situation based on history

## ***2 Project Technical Description***

Jupyter Notebook with Python is used for the coding of this project. The notebook is easy to install and easy to operate. The Kaggle data set is used to test the model. Six machine learning algorithms are implemented and calculated accuracy and hamming loss.

### ***2.1 Application Architecture***

Following diagram figure 1 shows the flow diagram of the Classification of the online toxic comments project.



**Figure1. Flow diagram of Proposed System**

## **2.2 Interactions with other Applications**

This application is designed to communicate with other applications. By using this application we can not only classify text comments but also we can classify any social networking website comments like Twitter and Facebook.

## **2.3 Capabilities**

This application is developed using different open-source software. Python is used as a programming language. Python is platform-independent and machine-independent. No database is required in this project. Instead of a database, we use a Kaggle data set. Reading and writing data from data sets is performed using numpy and pandaspy.

## **2.4 Risk Assessment and Management**

In this project, a separate risk management team is maintained to handle risks. They will identify risk and mitigate them early to reduce the impact on the system and maintains a risk management log [1] for future purpose. Risks are handled with the highest priority first order.

## **3 Project Requirements**

### **3.1 Identification of Requirements**

A huge amount of data is released daily through social media sites. This huge amount of data is affecting the quality of human life significantly, but unfortunately due to the presence of toxicity that is there on the internet, it is negatively affecting the lives of humans [2]. Due to this negativity, there is a lack of healthy discussion on social media sites since toxic comments restrict people from expressing themselves and having dissenting opinions [3]. So, it is the need of the hour to detect and restrict antisocial behavior in online discussion forums [4].

### **3.2 Operations, Administration, Maintenance and Provisioning (OAM&P)**

In this project we have implemented six machine-learning algorithms used to solve the problem of text classification.

- 1) Logistic regression
- 2) Random Forest
- 3) SVM classifier
- 4) Naive bayes

- 5) Decision tree
- 6) KNN classification

For each algorithm, we have calculated accuracy and evolution metrics like hamming loss and log loss. Following are the step-by-step operations performed in this project.

- 1) Import the required packages.
- 2) Load csv file.
- 3) Read data from a .csv data file.
- 4) Perform data exploration and preprocessing.
- 5) Data visualization.
- 6) Perform text processing.
- 7) Import word cloud and extract toxic words.
- 8) Applying six machine learning algorithms and calculates and displays accuracy and confusion matrix.
- 9) Model comparison and visualization.

### 3.3 Release and Transition Plan

This project is independent of the operating system; we can deploy it in any operating system. To run the application all the software is installed and tested successfully. Whenever there is a change in the data set structure we will perform code-up gradations at periodical intervals.

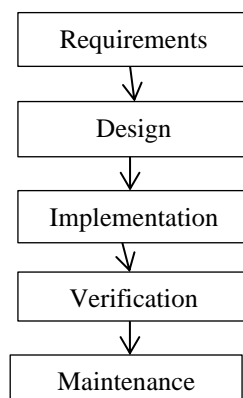
## 4 Project Design Description

This project is implemented using Jupyter Notebook. Jupyter Notebook is an open-source platform for building machine learning and data science applications. Adding new modules to Jupyter is easy compared to other software. Python is used as a programming language. Python is a fundamental language that helps to build stand-alone applications, web applications and machine learning applications. The following are the main modules performed in this project.

- 1) Data set collection
- 2) Data Exploration
- 3) Data preprocessing
- 4) Data Visualization
- 5) Feature Extraction
- 6) Model implementation
- 7) Compare different model accuracy

## 5 Internal/external Interface Impacts and Specification

To Design interfaces, we have used different technologies that participate in making user-friendly interfaces, The Waterfall model [5] is used to develop our application, because it is suitable for small, time-bound applications. The waterfall model is a sequential model. In our project, there is no change in requirements i.e. extracting data from the data set. So, the waterfall model is selected to develop this application. The following diagram shows the waterfall model used in this application.



**Figure2. SDLC model for Classification of Online toxic comments**

## Output Screen shots

### Data set Visualization

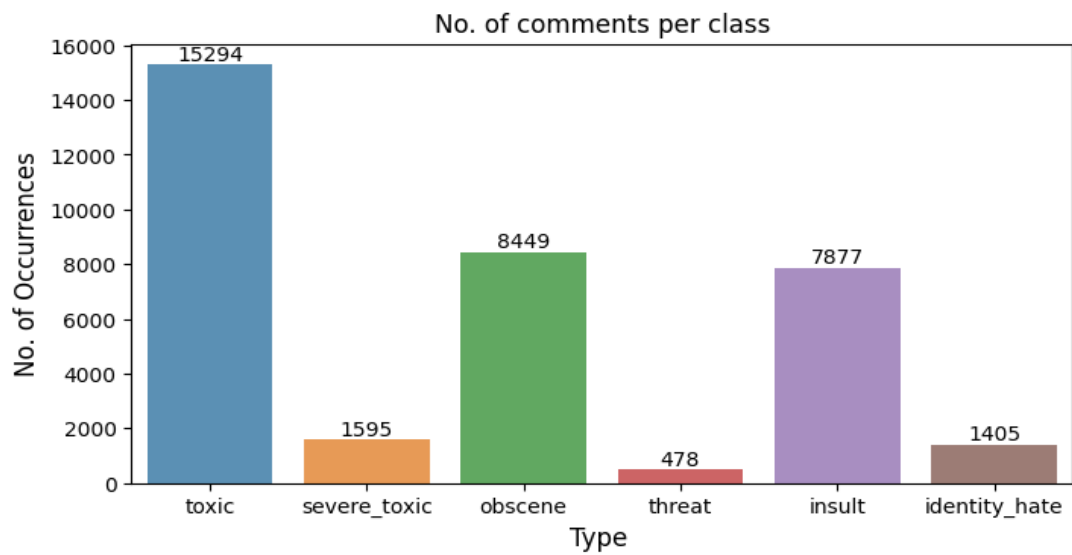


Figure3. Data set visualization

### Percentage of Comments in various categories

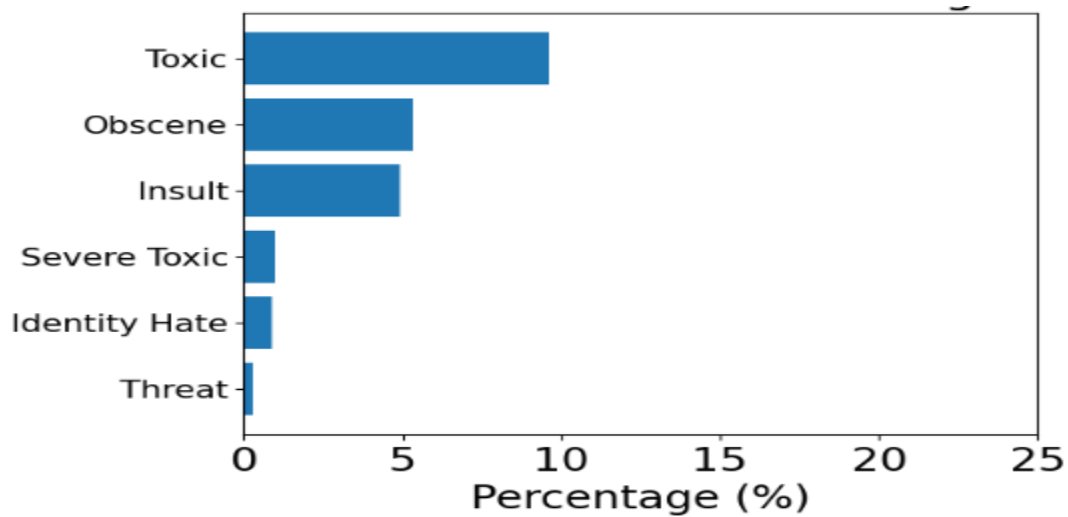


Figure4. Percentage of comments in various categories



## Support Vector Machine

```
In [31]: from sklearn.svm import SVC
SVM = SVC()
SVM.fit(X_train_fit, y_train)
predictions = SVM.predict(X_test_fit)
val1 = (accuracy_score(y_test, predictions)*100)
print("*Accuracy score for SVM: ", val1, "\n")
print("*Confusion Matrix for SVM: ")
print(confusion_matrix(y_test, predictions))
print("*Classification Report for SVM: ")
print(classification_report(y_test, predictions))
```

\*Accuracy score for SVM: 86.46666666666667

\*Confusion Matrix for SVM:

```
[[1344  119]
 [ 287 1250]]
```

\*Classification Report for SVM:

	precision	recall	f1-score	support
0	0.82	0.92	0.87	1463
1	0.91	0.81	0.86	1537
accuracy			0.86	3000
macro avg	0.87	0.87	0.86	3000
weighted avg	0.87	0.86	0.86	3000

Figure5. SVM Model and Confusion matrix

### Decision Tree classifier implementation

## Decision Tree Classifier

```
In [33]: from sklearn.tree import DecisionTreeClassifier
DT = DecisionTreeClassifier()
DT.fit(X_train_fit, y_train)
predictions = DT.predict(X_test_fit)
val3 = (accuracy_score(y_test, predictions)*100)
print("*Accuracy score for DT: ", val3, "\n")
print("*Confusion Matrix for DT: ")
print(confusion_matrix(y_test, predictions))
print("*Classification Report for DT: ")
print(classification_report(y_test, predictions))
```

\*Accuracy score for DT: 81.63333333333334

\*Confusion Matrix for DT:

```
[[1237  226]
 [ 325 1212]]
```

\*Classification Report for DT:

	precision	recall	f1-score	support
0	0.79	0.85	0.82	1463
1	0.84	0.79	0.81	1537
accuracy			0.82	3000
macro avg	0.82	0.82	0.82	3000
weighted avg	0.82	0.82	0.82	3000

Figure6. Decision Tree classifier and Confusion matrix

### KNeighborsClassifier

```
In [34]: from sklearn.neighbors import KNeighborsClassifier
KNN = KNeighborsClassifier()
KNN.fit(X_train_fit, y_train)
predictions = KNN.predict(X_test_fit)
val4 = (accuracy_score(y_test, predictions)*100)
print("*Accuracy score for KNN: ", val4, "\n")
print("*Confusion Matrix for KNN: ")
print(confusion_matrix(y_test, predictions))
print("*Classification Report for KNN: ")
print(classification_report(y_test, predictions))
```

\*Accuracy score for KNN: 53.43333333333333

\*Confusion Matrix for KNN:

```
[[1457  6]
 [1391 146]]
```

\*Classification Report for KNN:

	precision	recall	f1-score	support
0	0.51	1.00	0.68	1463
1	0.96	0.09	0.17	1537
accuracy			0.53	3000
macro avg	0.74	0.55	0.42	3000
weighted avg	0.74	0.53	0.42	3000

Figure7. KNeighbors Model and Confusion matrix

### LogisticRegression

```
In [35]: from sklearn.linear_model import LogisticRegression
LR = LogisticRegression()
LR.fit(X_train_fit, y_train)
predictions = LR.predict(X_test_fit)
val5 = (accuracy_score(y_test, predictions)*100)
print("*Accuracy score for LR: ", val5, "\n")
print("*Confusion Matrix for LR: ")
print(confusion_matrix(y_test, predictions))
print("*Classification Report for LR: ")
print(classification_report(y_test, predictions))
```

\*Accuracy score for LR: 86.43333333333332

\*Confusion Matrix for LR:

```
[[1330 133]
 [ 274 1263]]
```

\*Classification Report for LR:

	precision	recall	f1-score	support
0	0.83	0.91	0.87	1463
1	0.90	0.82	0.86	1537
accuracy			0.86	3000
macro avg	0.87	0.87	0.86	3000
weighted avg	0.87	0.86	0.86	3000

Figure8. Logistic Regression Model and Confusion Matrix

## Random Forest Classifier

```
In [32]: from sklearn.ensemble import RandomForestClassifier
RF = RandomForestClassifier(n_estimators=100, random_state=42)
RF.fit(X_train_fit, y_train)
predictions = RF.predict(X_test_fit)
val2 = (accuracy_score(y_test, predictions)*100)
print("**Accuracy score for RF: ", val2, "\n")
print("**Confusion Matrix for RF: ")
print(confusion_matrix(y_test, predictions))
print("**Classification Report for RF: ")
print(classification_report(y_test, predictions))
```

\*Accuracy score for RF: 84.56666666666666

\*Confusion Matrix for RF:

```
[[1339 124]
 [ 339 1198]]
```

\*Classification Report for RF:

	precision	recall	f1-score	support
0	0.80	0.92	0.85	1463
1	0.91	0.78	0.84	1537
accuracy			0.85	3000
macro avg	0.85	0.85	0.85	3000
weighted avg	0.85	0.85	0.85	3000

Figure9. Random Forest Model and Confusion matrix

## Voting Classifier Implementation

### Voting Classifier

```
In [37]: from sklearn.ensemble import VotingClassifier
from sklearn.tree import DecisionTreeClassifier
estimator = []
estimator.append(('LR',
                  LogisticRegression(solver='lbfgs',
                                     multi_class='multinomial',
                                     max_iter=200)))
estimator.append(('SVC', SVC(gamma='auto', probability=True)))
estimator.append(('DTC', DecisionTreeClassifier()))
vot_hard = VotingClassifier(estimators=estimator, voting='hard')
vot_hard.fit(X_train_fit, y_train)
predictions = vot_hard.predict(X_test_fit)
val7 = (accuracy_score(y_test, predictions)*100)
print("**Accuracy score for Voting Classifier: ", val7, "\n")
print("**Confusion Matrix for Voting Classifier: ")
print(confusion_matrix(y_test, predictions))
print("**Classification Report for Voting Classifier: ")
print(classification_report(y_test, predictions))
```

\*Accuracy score for Voting Classifier: 82.96666666666667

\*Confusion Matrix for Voting Classifier:

```
[[1390  73]
 [ 438 1099]]
```

\*Classification Report for Voting Classifier:

	precision	recall	f1-score	support
0	0.76	0.95	0.84	1463
1	0.94	0.72	0.81	1537
accuracy			0.83	3000
macro avg	0.85	0.83	0.83	3000
weighted avg	0.85	0.83	0.83	3000

Figure10. Voting Classifier Model and Confusion matrix

## Model Comparison

### Model Comparson

```
In [38]: score = [val1,val2,val3,val4,val5,val7]
#make variabel for save the result and to show it
classifier = ('Suport Vector Machine','Random Forest','Decision Tree','KNNeighbors','Logistic Regression','Voting Classifier')
y_pos = np.arange(len(classifier))
print(y_pos)
print(score)

[0 1 2 3 4 5]
[86.46666666666667, 84.56666666666666, 81.63333333333334, 53.43333333333333, 86.43333333333332, 82.96666666666667]
```

Figure11. Model Comparison

## 6 Design Units Impacts

Following are the design impacts considered during the implementation of this project.

- 1) Any user can download the Kaggle data set.
- 2) Classification of text is done on data sets only.
- 3) Machine learning algorithm accuracy may change with different data sets.
- 4) Text is classified into only 6 categories.

### 6.1 Functional Area A/Design Unit A

#### 6.1.1 Functional Overview

In the software development life cycle, requirement specification is a main criterion; we have used many software and hardware are used to build this application. Following are the software and hardware technologies are used in this application

#### Software and Hardware Requirements

- Programming Language : Python
- Operating System : Window's 10/11
- IDE : Jupyter Notebook
- RAM : 4GB or above
- Hard Disc : 80GB or above

#### 6.1.2 Impacts

There is a huge impact in providing an accurate data set and making their classification successful, but in terms of functional impacts, there are no wide area effects on the application; later, the data set must be increased to take care of the huge online comments.

## 7 Open Issues

We have collected data sets from the Kaggle website. The Data is classified into six categories. The data set is irregular and inaccurate and some data is missing in data sets. After applying all the machine learning algorithms, the KNeighbors algorithm gives 53% accuracy but in real-time data sets it gives more than 83% accuracy. This is because some of the toxic words are not reported in the data set. In this project, the Kaggle data set is used directly. The quality of the data set must improve to produce better classifications. After applying different machine learning algorithms some time SVM classifier performs better and sometimes logistic regression performs better. But hamming loss is less for Logistic Regression so the system suggested to use of logistic regression for further classification.

## 8 Acknowledgements

We want to express our sincere gratitude to my department for giving me the good opportunity to work as a team and to work on the selected platforms for the completion of the project. We are grateful to our professor, Dr. Yunchuan Liu for helping us choose and carry out the project. We completed our project successfully under our professor's guidance and the valuable weekly feedback is given to us. This project plays a vital role in our academic career.

## 9 References

- [1]. Risk Analysis & Risk Management in Software Engineering, <https://www.guru99.com/risk-analysis-project-management.html>
- [2]. M. Duggan, “Online harassment 2017,” Pew Res., pp. 1–85, 2017, doi: 202.419.4372.
- [3]. M. A. Walker, P. Anand, J. E. F. Tree, R. Abbott, and J. King, “A corpus for research on deliberation and debate,” *Proc. 8th Int. Conf. Lang. Resour. Eval. Lr.* 2012, pp. 812–817, 2012.
- [4]. J. Cheng, C. Danescu-Niculescu-Mizil, and J. Leskovec, “Antisocial behavior in online discussion communities,” *Proc. 9th Int. Conf. Web Soc. Media, ICWSM 2015*, pp. 61–70, 2015.
- [5]. Mara Calvella, “Waterfall Methodology: How to Use It for Your Next Big Project”, <https://www.g2.com/articles/waterfall-methodology>
- [6]. C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, “Abusive language detection in online user content,” *25th Int. World Wide Web Conf. WWW 2016*, pp. 145–153, 2016, doi:10.1145/2872427.2883062.

## 10 Appendices

- 1. <https://www.baeldung.com/spring-mvc-tutorial>
- 2. <https://www.guru99.com/jsp-mvc.html>
- 3. <https://www.guru99.com/software-development-life-cycle-tutorial.html>
- 4. <https://www.geeksforgeeks.org/html/>
- 5. <https://www.mysqltutorial.org/>