# Dataset

The hate speech and offensive language dataset was used for the classification of text data. The proposed dataset was downloaded from Kaggle that was based on the tweets. Each tweet in the dataset was labeled with three different types of classification including the hate language, offensive language and neutral. In the process of compiling the dataset, each tweet is categorized by different CrowdFlower user that assign the label. Each label feature contains the number of users that predict the respective label. The downloaded dataset initially contains the 25296 tweet samples followed by the 3 labels. The initial statistics of the dataset is recorded in the below table.

**Table 1:** Number of Samples and Attributes in dataset

|  |  |  |
| --- | --- | --- |
| **Hate Speech and Offensive Language** | **Number of Samples in Dataset:** | 25296 |
| **Number of Attributes** **in Dataset:** | 7 |

## Dataset – Summary of Features

The hate speech and offensive language dataset is a tabular dataset and stored in .csv file. It contains the 7 distinct features mainly contain the information od tweets and their related category. The tweet attribute of the dataset contains the textual data while rest of the attributes contained the numerical data only. The datatype of each variable in dataset and its description is available in the below table.

**Table 2:** Dataset features and their description.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Variable Name | Description | Type |
| 1 | ID | Serial Number / identification Number | Numeric |
| 2 | Tweet | Content of tweet related to different topics | Text |
| 3 | Count | Count of users who labeled the tweet | Numeric |
| 4 | Hate-speech | Count of workers who label the tweet as hate-speech | Numeric |
| 5 | Offensive language | Count of workers who label the tweet as offensive language | Numeric |
| 6 | neutral | Count of workers who label the tweet as neutral | Numeric |
| 7 | Class | Final decision related to tweet based on majority votes. | Numeric |

## Target Attribute Description

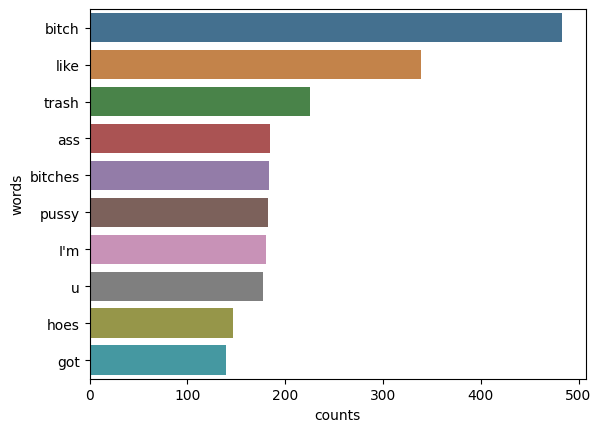
The hate speech and offensive language dataset contain the numerous column that can be used as the target variable. The count, hate-speech, offensive language, neutral and class attributes can be used as the target variable. Here, we need to specify the target variable for the proposed problem. For the better classification of tweets, three distinct attributes including the hate-speech, offensive language, and neutral were selected as the target variable and tweet text was used as the input variable. As the multiple features were selected as the target variable, the proposed problem was deal as a multi label classification problem. The distinct values of the target variables were considered as the classes. The specifications of input and out attributes are also presented in below table.

**Table 3:** General Statistics about dataset.

|  |  |
| --- | --- |
| Features | ID, Tweet, Count, Hate-speech, Offensive Language, Neutral, Class |
| Target | Hate-speech, Offensive Language, Neutral |
| Input | Tweet Text |
| Classes | Yes, no |

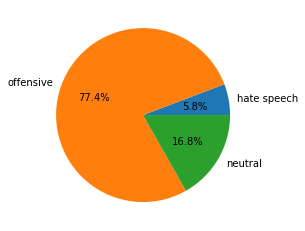
## Most Frequently Used Word Analysis

The most frequently used word analysis is originally based on the calculating the frequency of all words in the corpus and extracting the most frequently used words. The main of this analysis is to explore the dataset and words strength in the tweets. Moreover, the analysis will also help in targeting the most important words as features during classification. In this analysis, we also calculate the frequency of each word in tweet and extract the high frequently occurring top ten words. The top 10 most frequently used words with their frequency in the dataset are plotted in below bar cat.



## Hate Speech and Offensive Language Analysis

For the well understanding of the tweets dataset, we analyze the label distribution of the dataset. Class distribution reflect the count of each class in the corpus. We also count the number of tweets against each class using the class attribute if downloaded dataset. The below pie chat represents the class distribution of hate speech and offensive language dataset.



**Figure 1:** Initial Class Distribution of the Dataset.

# Methodology

Tweet Sentiment Analysis is the process of finding the hidden emotions of handler at the time of writing the tweet. It can be recognized by using the different techniques of natural language processing (NLP). The term employed in the writing, expression analysis methods, lexical phrase dictionaries, and the polarity of the words can all be applied to determine the writer's emotions. The concept of "polarity" states to whether the subject's had an offensive or neutral attitude when they produced it. The most commonly active filed in natural language processing is the recognition of text sentiment that commonly known as opinion mining or sentiment analysis. The recognitions of text sentiment have the numerous applications in economy, marketing, sociology, psychology, politics and digital marketing.

In the projected study, we identify the sentiment of tweet by analyzing its text either it contains hate speech, offensive language or neutral tweet. For the identification of tweets, we used 3 labels and classify them by 2 different multi label classifiers. Lastly, the results of both models were compared for the fair analysis.

## Data Preprocessing

The primary phase in formulating a dataset for machine learning algorithms is to clean and renovate the data. In this regards, we implement a numerous data preprocessing approaches, for cleaning and tokenization.

For the cleaning of the data, we abolish all stop words (like coma, punctuation marks etc.) from tweets texts. Additionally, we also remove commas, special characters, integers between the words, and URLs from the tweet's content. Lastly, we also remove the emojis character from the tweet text and convert the rest of the text into lower case letters. The ending step in the data preprocessing was words stemming that transformed the words into their conventional form. For converting the words into their original case, a well know stemming algorithm label as Porter Stemmer was used from NLTK library. Additionally, it contributes in the abstraction of input features. The sample of tweet before and after text processing is presented in below table.

**Table 4:** Sample of Raw Tweet and Processed Tweet

|  |  |
| --- | --- |
| Raw Tweet Text | Processed Tweet Text |
| !!! RT @mayasolovely: As a woman you shouldn't complain about cleaning up your house. &amp; as a man you should always take the trash out... | RT @mayasolovely : As woman shouldn't complain cleaning house man always take trash |

## Feature Extraction

After the tweets have been transformed into stem words, the necessary elements for categorization must be extracted, and the unnecessary words that have no particular bearing on sentiment must be eliminated. The attributes from textual data must be extracted in order to classify any text data. For the extraction of features from tweets content, TF-IDF (Term Frequency Inverse Document Frequency) and Word2Vec are the most frequently used NLP techniques. TFIDF is a quantitative method for estimating word significance. Each word's frequency in the dataset was computed, and the frequency was then represented numerically. The frequency of each unique word is represented by a vector of numbers created using TFIDF. The length of the created vector is equal to the distinct words in the corpus. For the features extraction from hate speech and offensive language dataset, we used the built-in Count Vectorizer function of scikit-learn library. It converted the distinct words into bag of words and for each tweet text, it placed 1 for existing words and 0 for missing words. Resultantly, all tweets were converted into asperse matrix of numerical features using the Count Vectorizer.

## Train Test Split

After preprocessing of tweets text and feature extraction techniques, the dataset must be partitioned into numerous sections for the training, testing, and the model validation. The dataset was divided into three parts using the train-test-split module created by the scikit-learn package. The built-in train test split method picks a random selection from each class using some percentage for one group and the leftover data for the second subset. It did not cancel out the observations in the partitioned subset. Using the reprocessing data, we split the tweets text into two distinct subsets. The training and testing portions of the extracted features dataset were divided in a ratio of 70% to 30%. The train and testing dataset contain the 213863 and 53466 tweets after the dataset has been divided. The statistics of divided training and testing set is also available in below table.

**Table 3:** Tweets in train, test and validation set.

|  |  |  |
| --- | --- | --- |
|  | Training Set | Testing Set |
| Count of Samples | 2520 | 980 |
| Count of Features | 11079 | 11079 |

## Machine Learning Models Development and Training

For the classification of tweets sentiment, we used the support vector machine (SVM) machine learning model with chainer and powerset. Both models did not develop from scratched but imported from the built-in libraries. By using the scikit-learn library of python, we firstly initialized the SVM model and then pass it to chainer and powerset classifier for the recognition of tweets sentiment. We only tunned the kernel hyper parameter of SVM classifier while rest of the hyper parameters were used with their default values. The linear Kernel of SVM was used with chainer classifier while sigmoid kernel was used with powerset classifier. After the initialization of the both models, the training tweets text was passed to the models for training process.

## Model Evaluation

Researchers evaluated the classification algorithm using a range of evaluation criteria, including accuracy, confidence, sensitivity, and f1-score. Recall (sensitivity) is the percentage of correctly predict positive instances over the all-positive instances. Contrarily, precision (confidence) is the percentage of true positive cases that are among the expected positive cases.   The f1-score is a harmonic assessment of recall and precision. The 980 tweets text were used to evaluate the models using described evaluation measures. All the evaluation measures were calculated using their corresponding formula. The formula for calculating the evaluation scores is described in equation 1-4.

Eq. 1

Eq. 2

Eq. 3

Eq. 4

# Results and Discussion

## Environmental Setup

We will discuss the learning environment for the suggested systems, along with the programming language and packages that were employed. The scripting language utilized for all of the projects was Python. The Python 3.7 version was used for the tests. Python 3.7 was used to build a Conda environment for the models' development. The necessary libraries were fully installed in the constructed environment.  Several libraries were set up in our current environment. The table that follows lists all essential public libraries along with their version numbers.

**Table 4:** List of libraries used in environment

|  |  |
| --- | --- |
| **Library Name** | **Version** |
| Matplotlib | 3.5.1 |
| scikit-learn | 1.0.2 |
| NLTK | 3.7.0 |
| Pandas | 1.2.9 |
| Numpy | 1.21.0 |
| Seaborn | 0.11.2 |

## Experiments Details

For the tweet text sentiment analysis, we perform different experiments including the data split, models training and evaluation. The detail description of each experiment is also available in below table.

**Table 5:** List of implemented Experiments in proposed work

|  |  |  |
| --- | --- | --- |
| **Experiment no.** | **Experiment Name** | **Description** |
| Experiment # 1 | Train Test Split | Split the dataset into two different groups labeled as training set and testing set. The percentage ration for training and testing sets was set to 80% and 20% respectively. The process of split the dataset has been done by using the built-in train test split function of scikit-learn library. |
| Experiment # 2 | SVM with Chainer | Firstly, initialize the SVM model from scikit learn library and pass this object to the chainer classifier. The tweet text of 2530 training samples was used for the training of the model. The selected evaluation measures were calculated using the text of test samples. |
| Experiment # 3 | SVM with Powerset | For the SVM with Powerset, we also used the same technique. Firstly, load the SVM model from library and pass to the powerset classifier. The training samples of tweets dataset was used for the training of the model. Lastly, the trained model with test samples used to calculate the evaluation measures. |
| Experiment # 4 | Comparative Study | By following the training of both models, the comparative study was performed to compare the performance of the trained models. The comparative study was based on the comparison of all selected evaluation measures. |

## Experiments Results

The experiment result section will elaborate the performance of the model using the text of test samples with different graphs and tables.

### Train Test Split

The train and test subsets were extracted by splitting the tweets text with the percentage of 80% and 20%. The train and test subsets were extracted using the train-test-split function of sk-learn library. The class distribution in the extracted train and test set is also calculate and plotted in bar chat. The bar chat of class distribution in train and test corpus is presented in below figures.

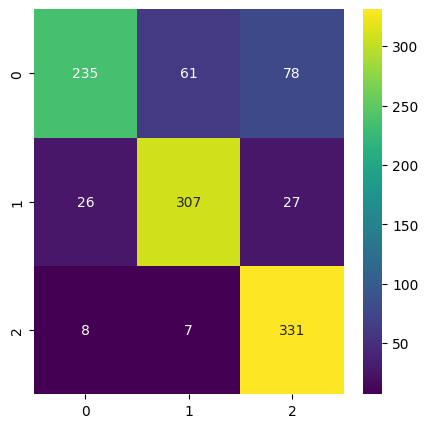
|  |  |
| --- | --- |
|  |  |

### SVM with Chainer

SVM with the chainer classifier was used for the recognition of tweets sentiment and detect the hate speech and offensive language. The SVM with chainer classifier was developed and trained in learning environment using the 2520 tweets samples of training set. By following the complete learning of the model, the 980 tweets of test set were used for the evaluation of the model. The complete classification report of the chainer model was calculated using the text of test samples. The classification report of the SVM chainer model is available in below table and the confusion matrix of the model is also presented in below figure.

**Table 6:** Multi Label Sentiment Classification – SVM Chainer Report

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classification Report for SVM Chainer Classifier** | | | | |
|  | precision | recall | f1-score | support |
| Hate-speech | 0.87 | 0.63 | 0.73 | 374 |
| Offensive | 0.82 | 0.85 | 0.84 | 360 |
| Neutral | 0.76 | 0.96 | 0.85 | 346 |
|  |  |  |  |  |
| accuracy |  |  | 0.81 | 1080 |
| macro avg | 0.82 | 0.81 | 0.80 | 1080 |
| weighted avg | 0.82 | 0.81 | 0.80 | 1080 |
| Accuracy Score (Training): 0.8316 Accuracy Score (Testing): 0.8083 | | | | |

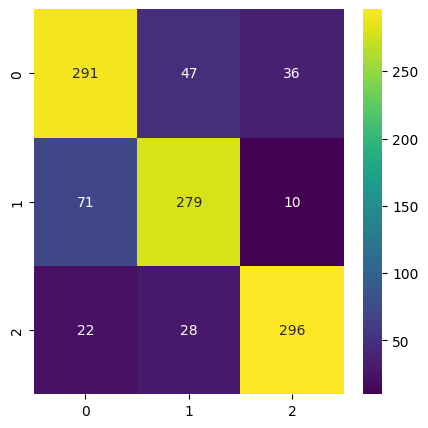


### SVM with Powerset

Secondly, the SVM model with powerset classifier was used for the detection of hate speech and offensive language. The text of 2520 training tweets was used for the learning of the proposed model. Bu following the complete learning process of the models, the text of testing tweets was used for evaluating the performance of the trained model. The trained powerset model performance was evaluated on the basis of selected evaluation measures including the accuracy, confidence, sensitivity and f1-score. For the complete understanding of the model performance, the classification report of the trained model was generated using the test instances. The comprehensive classification report of SVM powerset classifier and confusion matrix of the classifier is presented below.

**Table 7:** Multi Label Sentiment Classification – SVM Powerset Report

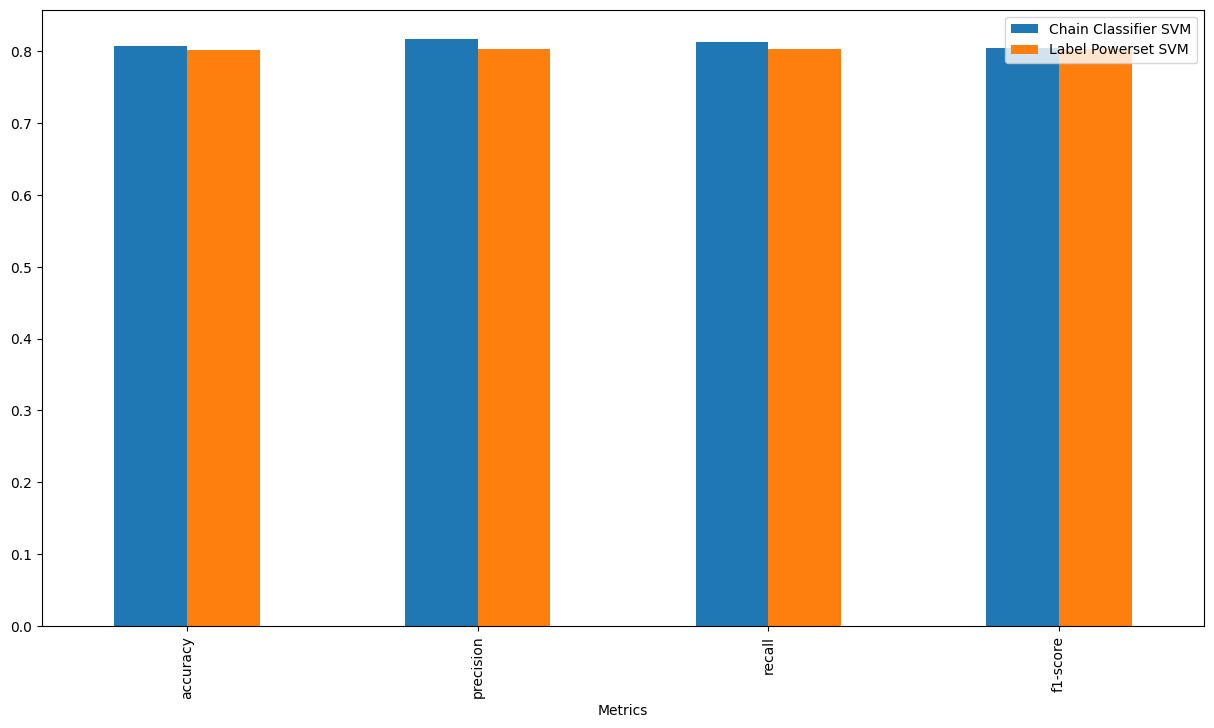
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classification Report for SVM Powerset Model** | | | | |
|  | precision | recall | f1-score | support |
| Hate-speech | 0.76 | 0.78 | 0.77 | 374 |
| Offensive | 0.79 | 0.78 | 0.78 | 360 |
| Neutral | 0.87 | 0.86 | 0.86 | 346 |
|  |  |  |  |  |
| accuracy |  |  | 0.80 | 1080 |
| macro avg | 0.80 | 0.80 | 0.80 | 1080 |
| weighted avg | 0.80 | 0.80 | 0.80 | 1080 |
| Accuracy Score (Training): 0.8316 Accuracy Score (Testing): 0.8018 | | | | |



### Models Comparative Analysis

The comparison study of the trained model will help to pick the model with best performance for detecting the hate speech and offensive language. We compare all the chosen evaluation measures of trained model and pick the model with highest evaluation scores. Both models showed the approximately equal accuracy score. But the precision and recall that defined the actual positive score and model predicted positive score are slightly high of SVM Chainer relative to SVM Powerset. The score comparison of both model is also available in below table. The evaluation score of both models collectively plotted in below bar chat for fair comparison of model performance.

|  |  |  |
| --- | --- | --- |
| **Models Evaluation Scores Comparison** | | |
| **Metrics** | Chain Classifier SVM | Powerset SVM |
| Accuracy | 0.8083 | 0.8018 |
| Precision | 0.8171 | 0.8038 |
| Recall | 0.8125 | 0.8028 |
| F1-score | 0.8042 | 0.8032 |



# Future Direction

In the proposed study we got the approximately 80% accuracy score with SVM chainer and SVM powerset. For the recognition of hate speech and offensive language in tweets text, the performance of the machine learning model may be increased in future studies. For increasing the performance of the recognition, the combination of datasets from different sources can be used in future work. Different feature extraction methods and combination of different features set can used for the increased performance of the prediction model. Lastly, the machine learning and deep learning models from different domain like ensemble learning can be used increase the efficiency of hate speech and offensive language recognition. Collectively, the different feature extraction techniques, different set of features and transfer learning technique with different models be used to increase the performance of the prediction model.

# Conclusion

In the proposed study, we used machine learning models for the detection of hate speech and offensive language in tweets text. For the detection of tweets sentiment, the proposed study trained the SVM model with chainer and powerset classifier with the tweets dataset from Kaggle. Firstly, the dataset was split into training and testing set just after the preprocessing of the tweets text. Later, both models were trained using the text of training tweets and evaluated using the text if testing set. We got the approximately balance accuracy score (80%) for both models. As the processed dataset was the class balanced dataset and the accuracy is best evaluation measure for class balance corpuses, we select the accuracy score as the base evaluation measure. But in the evaluation process, we got the approximately same accuracy score for both models. The SVM with chainer classifier showed the high precision and recall score compare to the powerset classifier. Although our base measure is accuracy and it is best for class balance corpuses, but for selecting the best model in case of equal accuracy score, we refer to other selected evaluation score. By considering the all-evaluation scores, we analyze that performance of the SVM with chainer classifier is more robust as compare to the SVM with powerset classifier. The significance of evaluation score also reveal that trained model is robust enough to make detection in real world environment.