CE807 – Assignment 2 - Final Practical Text Analytics and Report

Student ID: 2201277

Offensive Language Detection And Nature of Data

```
train_df = pd.read_csv(train_file)
print(train_df.head(5))
```

```
      id                                                tweet label
0  42884  @USER I'm done with you as well. An INTENTIONA...   NOT
1  92152  I now have over 6k followers.  Only 94k to go ...   NOT
2  65475  @USER Tom was bought! He is more interested in...   NOT
3  22144  @USER @USER Even her brother thinks she is a m...   OFF
4  81048  @USER @USER @USER @USER @USER I can understand...   OFF
```

```
print(train_df.info())
print(train_df["label"].unique())
print(train_df["label"].value_counts())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12313 entries, 0 to 12312
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   id      12313 non-null  int64 
 1   tweet   12313 non-null  object
 2   label   12313 non-null  object
dtypes: int64(1), object(2)
memory usage: 288.7+ KB
None
['NOT' 'OFF']
NOT    8221
OFF    4092
Name: label, dtype: int64
```

Method 1

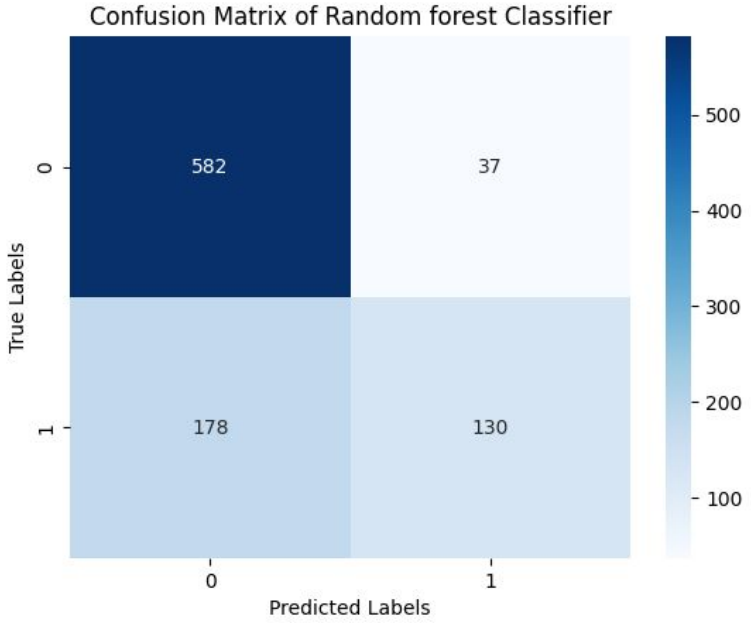
Random Forest Classifier:

- Widely used for classification tasks like offensive language detection
- Builds a collection of decision trees, with the majority vote of trees used for final prediction
- Highly accurate algorithm that is less prone to overfitting than other methods like Decision Trees
- Can handle a large number of features, which is important for high-dimensional natural language processing tasks
- Can handle missing data without requiring imputation, which is useful for working with real-world datasets

Method 2

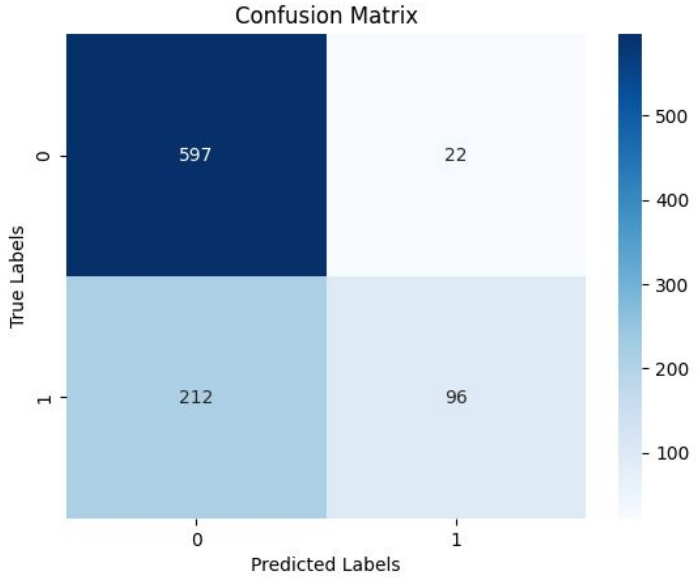
Gradient Boosting Classifier:

- A powerful and flexible machine learning model that can handle a wide range of classification tasks
- Good at handling noisy data and has shown to perform well on various text classification tasks
- Builds an ensemble of weak learners, each focusing on correcting the errors made by the previous learner
- Less prone to overfitting compared to other algorithms, which is important when working with limited training data
- Can handle imbalanced data and can be fine-tuned to achieve high accuracy by adjusting hyperparameters



Best hyperparameters: {'max_depth': None, 'min_samples_leaf': 2, 'min_samples_split': 4, 'n estimators': 100}
Accuracy: 0.7659115426105717
Recall: 0.6893240039443593
Precision: 0.7547966285948242
F1 Score: 0.7032409681745156

Best hyperparameters: {'learning_rate': 0.5, 'max depth': 5, 'n estimators': 200}
Accuracy: 0.761596548004315
Recall: 0.6934335858003063
Precision: 0.7409099817089104
F1 Score: 0.7056986437546777



Classification metric:

F1 score

Model	F1 Score
Model 1	0.7072
Model 2	0.6988
SoA model 2 with 100% data	0.7225

Table 2: Model Performance

Future Scope