Machine Learning Project

Why so Harsh?

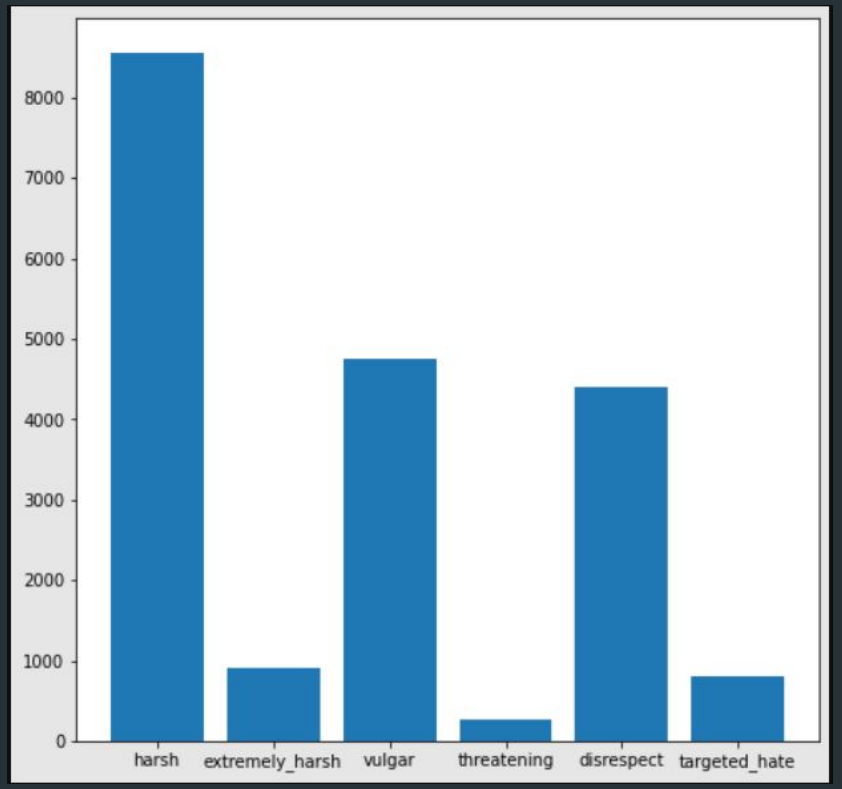
Team Two

By
Vivek IMT2020110
Harshadeep IMT2020085

Data Analysis

Counting the number of comments in each class.

We can see that the data is unbalanced.



Wordcloud

- Useful to get an overview of which term has more frequency (or) importance.

- Importance of the word is directly related to its size in the cloud.

threatening

Preprocessing

- Fixing contractions
- Converting to lower-case.
- Replacing emoticons and punctuations.
- Removing all numbers and words with numbers.
- Removing stop words.
- Correcting spellings of incorrect words.
- Lemmatization
- Stemming
- Vectorization

```
emoticons = {
    ':-)': 'happy',
    ':)': 'happy',
    ':-(': 'frown',
    ':(': 'frown',
    'xD': 'laugh',
    ':/': 'sad',
    ':|': 'indecision',
    ':o': 'surprise',
    '<3': 'heart'
}
```

Some commonly used emoticons and their meanings from wikipedia

- **Converting to lower-case.**
- **Replacing emoticons and punctuations.**
- **Removing all numbers and with numbers.**
- **Removing stop words**

words

These 4 Operations can be done using basic functions and with the help of regex(Regular Expression package).

Contractions

Resolving contractions and words used in slang.

Ex:
you’re -> you are
ima -> I am going to
yall -> you all
gotta -> got to

Corrections of spellings

Using symspell, we can correct spellings of words in a sentence.

Ex:
Dugs -> Dogs

Lemmatization

Lemmatization is the process of reducing different forms of word into a single form.

Eg:

cats -> cat

better -> good

geese -> goose

Stemming

Stemming is the process of reducing derived words to their root form.

Eg:

dogs -> dog

Note: Lemmatization and Stemming are used separately.

TFIDF - Vectorization (Term Frequency-Inverse Document Frequency)

- Word and char vectorization are done separately.
- Both vectorizers are fitted using whole text (by concatenating test comments and train comments.)
- Then both word and char features are merged using the hstack function.

TFIDF - Vectorization: Parameters

- min_df = 2

  ignore terms that appear in less than 2 documents

- Set sublinear_tf = True

  Apply sublinear tf scaling, i.e. replace tf with 1 + log(tf)

- Max_features = None

  So as to consider all features for better results.

Pickle for storing data

- After data preprocessing and vectorizing the data, pickle files are generated for test and train data.

- This saves time as we can load our processed data via pickle files instead of doing computations.

- Sizes of pickle files for test and train data is ~500MB and ~1.2GB respectively.

TRAINING & PREDICTION MODELS

- Logistic regression
- Classifier chain
- SGDClassifier
- Ridge

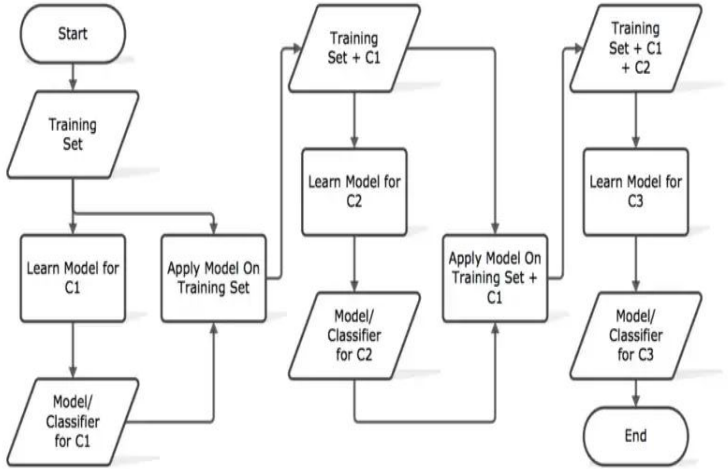
Note: RandomizedSearchCV and GridSearchCV are used for hyperparameter tuning.

Logistic Regression

- class sklearn.linear_model.LinearRegression
- As our data is unbalanced, using class_weight = ‘balanced’ gave best results.
- $C = 1$, Inverse of regularization strength, smaller values specify stronger regularization
- Best solver = ‘lbfgs’

Classifier Chains

- class sklearn.multioutput.ClassifierChain
- Order = ‘random’ (random ordering will be used)
- random_state = 0
- Approximately same accuracy as LogisticRegression.



SGDClassifier (Stochastic gradient descent classifier)

- class sklearn.linear_model.SGDClassifier
- class_weight = ‘balanced’
- loss = ‘log’
- max_iter = 100000
- Fast, but slightly less accuracy

Ridge Model

- Minimizes the objective function: $||y - Xw||^2 + \alpha||w||^2$
- class sklearn.linear_model.Ridge
- copy_X = True (If True, X will be copied; else, it may be overwritten.)
- solver = ‘sag’
- random_state = 33 (Shuffles data)
- alpha = 45
- Best accuracy

Miscellaneous

- scikit-learn-intelex package is used to accelerate scikit-learn algorithms by 10-100 times.

- Time taken for notebook execution is approximately 15-20 minutes.

References

- https://www.geeksforgeeks.org/python-lemmatization-with-nltk/
- https://en.wikipedia.org/wiki/Stemming
- https://scikit-learn.org/stable
- https://www.geeksforgeeks.org/understanding-tf-idf-term-frequency-inverse-document-frequency/
- https://symspellpy.readthedocs.io/en/latest/api/symspellpy.html
- https://en.wikipedia.org/wiki/List_of_emoticons


Thank
you