

## LP1 Assignment 14

Title:- Sentiment Analysis

Date of Completion:- 27/11/20

Problem Statement:- Twitter Data Analysis: Use Twitter data for sentiment analysis. The dataset is 3mb in size & has 31962 tweets. Identify the tweets which are hate tweets and which are not. ~~So~~

Learning Objectives:-

- 1) Understand text processing
- 2) Understand sentiment analysis

Learning Outcomes:- Students will be able to

- 1) To do sentiment analysis on twitter dataset using classification algorithm

Software/Hardware Requirements:- Python Libraries, OS (Linux)

Theory:-

1) Sentiment Analysis:-

It is the process of determining whether a piece of writing is positive, negative or neutral.

2) Applications:-

- a) Understand customer's attitude towards products & services.
- b) Marketers can get public opinion of their company.



- and product
- c) Gather critical feedback about problems in newly released products.

### 3) Steps

- Gather relevant tweets from twitter
- Preprocessing (stopword removal)
- Feature Extraction
- Feature Selection

#### a) Gather relevant tweets

Gather sample data from relevant sources & datasets to prepare training and testing set.

#### b) Preprocessing:-

- Without preprocessing there is high chance of text having noisy or inconsistent data.
- Punctuations, special characters, numbers are not required to do sentiment analysis.
- Removing punctuations, special characters, numbers
- Remove short words
- Tokenize data :- split strings into tokens (smaller words)
- Stemming :- getting the root word. (removing suffixes).

#### c) Feature extraction:-

- Selection of useful words is called feature extraction.
- Three different features namely unigram, bigram, n-gram
- Part of speech tags.
- Negation :- It may change the prior polarity of sentiment



#### d) Feature Selection:-

4 types

- i) Natural language processing
- ii) Statistical
- iii) Clustering based
- iv) Hybrid
- v) Classification

Dataset :- Twitter dataset

Result:-

Test case

Confusion matrix  $\begin{bmatrix} 7373 & 57 \\ 230 & 331 \end{bmatrix}$

accuracy score :- 0.96

recall score :- 0.59

Precision score :- 0.85

F1 score :- 0.69

Conclusion:- Thus I have understood sentiment analysis and how to perform it on String dataset. I have also completed the assignment successfully.

## CODE

```
import pandas as pd
import re
train=pd.read_csv("train.csv")
train.head()
train.drop("id",inplace=True,axis=1)

import nltk
nltk.download()

from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
def clean_sentences(text):
    text = text.lower()
    text = re.sub(r"^[a-z0-9^,!\.\\"]", " ", text)
    text = " ".join(text.split())
    text = " ".join(stemmer.stem(word) for word in text.split())
    return text
x = train['tweet']
y = train['label']
x = x.map(lambda a: clean_sentences(a))
x.head()

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,stratify=y,random_state=42)

x_train.head()

from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(stop_words='english')
x_train = vectorizer.fit_transform(x_train)
x_test = vectorizer.transform(x_test)
print(x_test)
from sklearn.svm import LinearSVC
model = LinearSVC(C=1.02, tol=0.3)
model.fit(x_train,y_train)

from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,
f1_score,recall_score

print(confusion_matrix(y_test,model.predict(x_test)))
print(accuracy_score(y_test,model.predict(x_test)))
print(recall_score(y_test,model.predict(x_test)))
print(precision_score(y_test,model.predict(x_test)))
print(f1_score(y_test,model.predict(x_test)))
```

# OUTPUT

Confusion Matrix:  $\begin{bmatrix} 7369 & 61 \\ 228 & 333 \end{bmatrix}$

Accuracy: 0.9638343136028031

Recall: 0.5935828877005348

Precision: 0.8451776649746193

F1 Score: 0.6973821989528797

The screenshot displays a Jupyter Notebook environment with a code editor on the left and a variable explorer and IPython console on the right.

**Code Editor:**

```
373 text = " ".join(text.split())
374 text = " ".join(stemmer.stem(word) for word in text.split())
375 return text
376 x = train['tweet']
377 y = train['label']
378 x = x.map(lambda a: clean_sentences(a))
379 x.head()
380
381 from sklearn.model_selection import train_test_split
382 x_train, x_test, y_train, y_test = train_test_split(x,y,stratify=y,ra
383
384 x_train.head()
385
386 from sklearn.feature_extraction.text import TfidfVectorizer
387 vectorizer = TfidfVectorizer(stop_words='english')
388 x_train = vectorizer.fit_transform(x_train)
389 x_test = vectorizer.transform(x_test)
390
391 from sklearn.svm import LinearSVC
392 model = LinearSVC(C=1.05, tol=0.5)
393 model.fit(x_train,y_train)
394
395 LinearSVC(C=1.05, tol=0.5)
396
397 from sklearn.metrics import confusion_matrix, accuracy_score, precisi
398
399 print(confusion_matrix(y_test,model.predict(x_test)))
400
401 print(accuracy_score(y_test,model.predict(x_test)))
402
403 print(recall_score(y_test,model.predict(x_test)))
404
405 print(precision_score(y_test,model.predict(x_test)))
406
```

**Variable Explorer:**

Name	Type	Size	Value
train	DataFrame	(31962, 2)	Column names: label, tweet
x	Series	(31962,)	Series object of pandas.core.series module
y	Series	(31962,)	Series object of pandas.core.series module
y_test	Series	(7991,)	Series object of pandas.core.series module

**IPython console:**

```
In [18]: print(confusion_matrix(y_test,model.predict(x_test)))
...:
...: print(accuracy_score(y_test,model.predict(x_test)))
...:
...: print(recall_score(y_test,model.predict(x_test)))
...:
...: print(precision_score(y_test,model.predict(x_test)))
...:
...: print(f1_score(y_test,model.predict(x_test)))
[[7366  64]
 [ 225 336]]
0.9638343136028031
0.5989304812834224
0.84
0.6992715920915712

In [19]:
```