## Indian Institute of Information Technology Surat

****

# Lab Report on

# Natural Language Processing (CS 601) Practical

**Submitted by**

### [RAHUL KUMAR SINGH] (UI21CS44)

**Course Faculty**

### Mrs. Nidhi Desai

## Department of Computer Science and Engineering

## Indian Institute of Information Technology Surat

## Gujarat-394190, India

**Jan-2024**

## Lab No: 5

**Aim:**

Perform n-gram operations on the dataset.For eg unigram, bi-gram and tri-gram. Analyze the outcomes of different n-gram and perform comparative Analysis.

**Description:**

* Objective: Perform n-gram operations (uni-gram, bi-gram, tri-gram) on text datasets and compare their outcomes.
* N-gram Extraction: Convert text into n-gram formats using TF-IDF vectorization.
* Datasets: Utilize NLTK datasets like Movie Reviews, Reuters Corpus, Twitter Samples, Product Reviews, Names Dataset, and Web Text Corpus.
* Comparative Analysis: Analyze and compare the accuracy, precision, recall, and F1-score of each n-gram model.
* Classification Task: Apply Logistic Regression for text classification.
* Outcome: Determine the optimal n-gram configuration based on dataset and classification task.

## Source Code:

import nltk

from nltk.corpus import reuters, twitter\_samples, product\_reviews\_1, names, webtext

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

import pandas as pd

nltk.download('movie\_reviews')

nltk.download('twitter\_samples')

nltk.download('names')

def prepare\_movie\_reviews():

documents = [(list(movie\_reviews.words(fileid)), category)

for category in movie\_reviews.categories()

for fileid in movie\_reviews.fileids(category)]

reviews = [' '.join(doc) for doc, \_ in documents]

labels = [1 if label == 'pos' else 0 for \_, label in documents]

return reviews, labels

def prepare\_twitter\_samples():

positive\_tweets = twitter\_samples.strings('positive\_tweets.json')

negative\_tweets = twitter\_samples.strings('negative\_tweets.json')

reviews = positive\_tweets + negative\_tweets

labels = [1] \* len(positive\_tweets) + [0] \* len(negative\_tweets)

return reviews, labels

datasets = [

('Movie Reviews', prepare\_movie\_reviews),

('Twitter Samples', prepare\_twitter\_samples),

]

def evaluate\_model(ngram\_range, ngram\_type, X\_train, X\_test, y\_train, y\_test):

vectorizer = TfidfVectorizer(ngram\_range=ngram\_range)

X\_train\_vec = vectorizer.fit\_transform(X\_train)

X\_test\_vec = vectorizer.transform(X\_test)

clf = LogisticRegression(max\_iter=1000)

clf.fit(X\_train\_vec, y\_train)

y\_pred = clf.predict(X\_test\_vec)

accuracy = accuracy\_score(y\_test, y\_pred)

precision = precision\_score(y\_test, y\_pred, average='weighted')

recall = recall\_score(y\_test, y\_pred, average='weighted')

f1 = f1\_score(y\_test, y\_pred, average='weighted')

return {

'ngram\_type': ngram\_type,

'accuracy': accuracy,

'precision': precision,

'recall': recall,

'f1\_score': f1

}

for dataset\_name, dataset\_fn in datasets:

print(f"Evaluating on {dataset\_name} dataset...")

reviews, labels = dataset\_fn()

dataset\_df = pd.DataFrame({'text': reviews, 'label': labels})

print(dataset\_df.head())

X\_train, X\_test, y\_train, y\_test = train\_test\_split(reviews, labels, test\_size=0.3, random\_state=42)

results = []

results.append(evaluate\_model((1, 1), 'Uni-grams', X\_train, X\_test, y\_train, y\_test))

results.append(evaluate\_model((1, 2), 'Uni + Bi-grams', X\_train, X\_test, y\_train, y\_test))

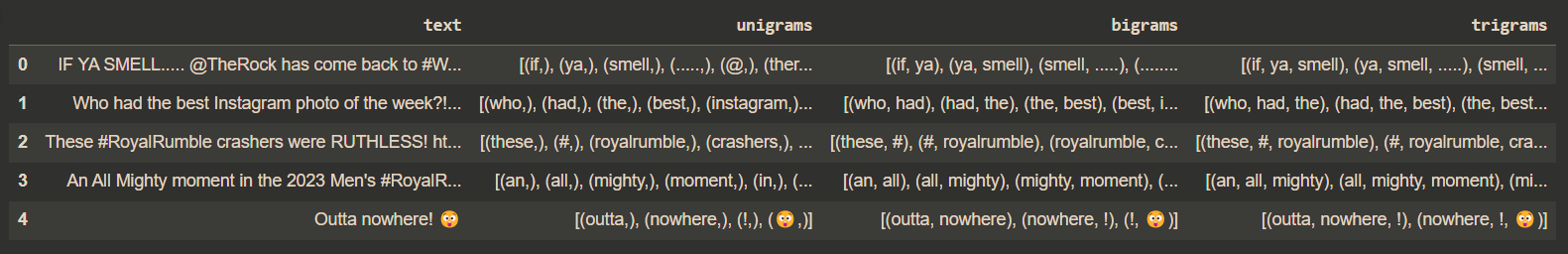
results.append(evaluate\_model((1, 3), 'Uni + Bi + Tri-grams', X\_train, X\_test, y\_train, y\_test))

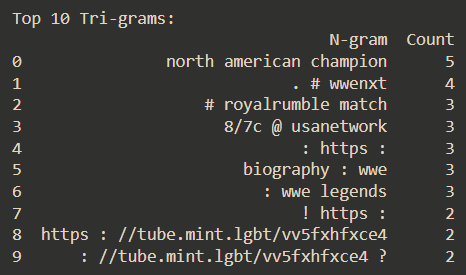
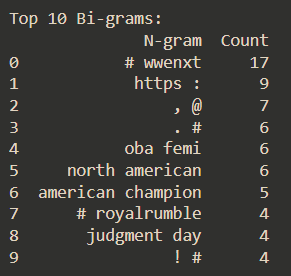
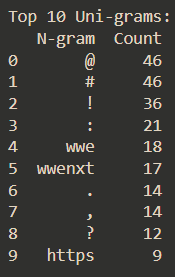
print("Evaluation Metrics: ")

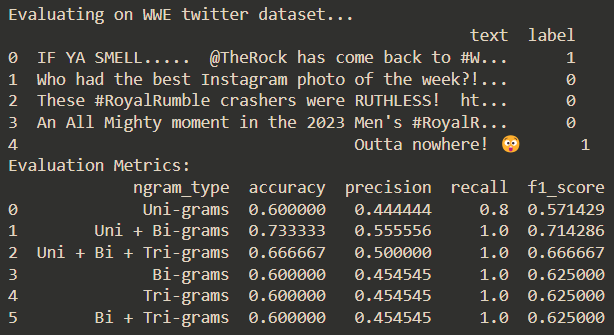
results\_df = pd.DataFrame(results)

print(results\_df)

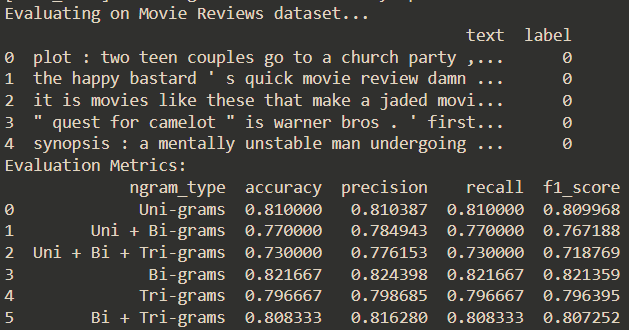
## Output:

**n-gram:  
**

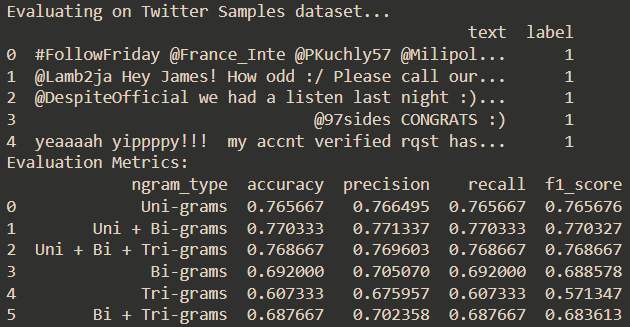
****

**Comparative Analysis on WWE Twitter Dataset:  
**

**Comparative Analysis on Movie Reviews Dataset:**

****

**Comparative Analysis on Twitter Samples Dataset:**

****

## Conclusion:

* WWE Twitter Dataset: (Uni + Bi) n-grams yield optimal performance.
* Movie Reviews Dataset: Bi-grams provide the best results.
* Twitter Samples Dataset: (Uni + Bi) n-grams achieve top accuracy.
* Performance varies across datasets making it essential to optimize n-grams.