Twitter Sentiment Analysis

May 6, 2023

1 Memebers:

Meteb Almadi	200304
Faisal Alotaibi	200141
Rayan Almudawah	200203

2 Code

2.1 Import Libraries

```
[ ]: # Importing required packages
import pyspark
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from pyspark.sql import functions as func
from pyspark.sql.types import StringType,FloatType
from pyspark.sql import SparkSession
import nltk
from nltk.corpus import stopwords
from  nltk.stem import SnowballStemmer
import re
from wordcloud import WordCloud
from pyspark.ml.feature import Tokenizer, CountVectorizer, IDF
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.evaluation import BinaryClassificationEvaluator
from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
```

Create Spark Session to extract features

```
[ ]: # Creating a spark session
spark = SparkSession.builder.appName('SentimentAnalysis').getOrCreate()
```

2.1.1 Load Data into a dataframe

```
[ ]: # Read CSV file into a DataFrame
     df = spark.read.csv('data.csv', inferSchema=True)


     df.show()
```

```
+---+----------+--------------------+--------+---------------+------------------
--+
|_c0|       _c1|                 _c2|     _c3|            _c4|
_c5|
+---+----------+--------------------+--------+---------------+------------------
--+
|  0|1467810369|Mon Apr 06 22:19:…|NO_QUERY|_TheSpecialOne_|@switchfoot
http:…|
|  0|1467810672|Mon Apr 06 22:19:…|NO_QUERY|  scotthamilton|is upset that he
…|
|  0|1467810917|Mon Apr 06 22:19:…|NO_QUERY|       mattycus|@Kenichan I
dived…|
|  0|1467811184|Mon Apr 06 22:19:…|NO_QUERY|        ElleCTF|my whole body
fee…|
|  0|1467811193|Mon Apr 06 22:19:…|NO_QUERY|         Karoli|@nationwideclass
…|
|  0|1467811372|Mon Apr 06 22:20:…|NO_QUERY|       joy_wolf|@Kwesidei not
the…|
|  0|1467811592|Mon Apr 06 22:20:…|NO_QUERY|        mybirch|         Need a
hug |
|  0|1467811594|Mon Apr 06 22:20:…|NO_QUERY|           coZZ|@LOLTrish hey
lo…|
|  0|1467811795|Mon Apr 06 22:20:…|NO_QUERY|2Hood4Hollywood|@Tatiana_K nope
t…|
|  0|1467812025|Mon Apr 06 22:20:…|NO_QUERY|        mimismo|@twittera que me
…|
|  0|1467812416|Mon Apr 06 22:20:…|NO_QUERY| erinx3leannexo|spring break in
p…|
|  0|1467812579|Mon Apr 06 22:20:…|NO_QUERY|   pardonlauren|I just re-
pierced…|
|  0|1467812723|Mon Apr 06 22:20:…|NO_QUERY|           TLeC|@caregiving I
cou…|
|  0|1467812771|Mon Apr 06 22:20:…|NO_QUERY|robrobbierobert|@octolinz16 It
it…|
|  0|1467812784|Mon Apr 06 22:20:…|NO_QUERY|    bayofwolves|@smarrison i
woul…|
|  0|1467812799|Mon Apr 06 22:20:…|NO_QUERY|     HairByJess|@iamjazzyfizzle
I…|
|  0|1467812964|Mon Apr 06 22:20:…|NO_QUERY| lovesongwriter|Hollis' death
sce…|
|  0|1467813137|Mon Apr 06 22:20:…|NO_QUERY|       armotley|about to file
taxes |
|  0|1467813579|Mon Apr 06 22:20:…|NO_QUERY|     starkissed|@LettyA ahh ive
a…|
|  0|1467813782|Mon Apr 06 22:20:…|NO_QUERY|      gi_gi_bee|@FakerPattyPattz
…|
+---+----------+--------------------+--------+---------------+------------------
--+
only showing top 20 rows
```

Rename Columns in the dataframe for later use

```
[ ]: df = df.withColumnRenamed('_c0','target').withColumnRenamed('_c1','id').
       ↪withColumnRenamed('_c2','date')\
        .withColumnRenamed('_c3','flag').withColumnRenamed('_c4','user').
       ↪withColumnRenamed('_c5','text')
     df.show()
```

```
+------+----------+--------------------+--------+---------------+---------------
-----+
|target|        id|                date|    flag|           user|
text|
+------+----------+--------------------+--------+---------------+---------------
-----+
|     0|1467810369|Mon Apr 06 22:19:…|NO_QUERY|_TheSpecialOne_|@switchfoot
http:…|
|     0|1467810672|Mon Apr 06 22:19:…|NO_QUERY|  scotthamilton|is upset that
he …|
|     0|1467810917|Mon Apr 06 22:19:…|NO_QUERY|       mattycus|@Kenichan I
dived…|
|     0|1467811184|Mon Apr 06 22:19:…|NO_QUERY|        ElleCTF|my whole body
fee…|
|     0|1467811193|Mon Apr 06 22:19:…|NO_QUERY|
Karoli|@nationwideclass …|
|     0|1467811372|Mon Apr 06 22:20:…|NO_QUERY|       joy_wolf|@Kwesidei not
the…|
|     0|1467811592|Mon Apr 06 22:20:…|NO_QUERY|        mybirch|         Need a
hug |
|     0|1467811594|Mon Apr 06 22:20:…|NO_QUERY|           coZZ|@LOLTrish hey
lo…|
|     0|1467811795|Mon Apr 06 22:20:…|NO_QUERY|2Hood4Hollywood|@Tatiana_K nope
t…|
|     0|1467812025|Mon Apr 06 22:20:…|NO_QUERY|        mimismo|@twittera que
me …|
|     0|1467812416|Mon Apr 06 22:20:…|NO_QUERY| erinx3leannexo|spring break in
p…|
|     0|1467812579|Mon Apr 06 22:20:…|NO_QUERY|   pardonlauren|I just re-
```

```
pierced…|
|     0|1467812723|Mon Apr 06 22:20:…|NO_QUERY|           TLeC|@caregiving I
cou…|
|     0|1467812771|Mon Apr 06 22:20:…|NO_QUERY|robrobbierobert|@octolinz16 It
it…|
|     0|1467812784|Mon Apr 06 22:20:…|NO_QUERY|    bayofwolves|@smarrison i
woul…|
|     0|1467812799|Mon Apr 06 22:20:…|NO_QUERY|     HairByJess|@iamjazzyfizzle
I…|
|     0|1467812964|Mon Apr 06 22:20:…|NO_QUERY| lovesongwriter|Hollis' death
sce…|
|     0|1467813137|Mon Apr 06 22:20:…|NO_QUERY|       armotley|about to file
taxes |
|     0|1467813579|Mon Apr 06 22:20:…|NO_QUERY|     starkissed|@LettyA ahh ive
a…|
|     0|1467813782|Mon Apr 06 22:20:…|NO_QUERY|
gi_gi_bee|@FakerPattyPattz …|
+------+----------+--------------------+--------+---------------+---------------
-----+
only showing top 20 rows
```

2.2 Inspect and Preprocessing the Dataset

Display the first 5 rows

```
[ ]: df.head(5)
```

```
[ ]: [Row(target=0, id=1467810369, date='Mon Apr 06 22:19:45 PDT 2009',
     flag='NO_QUERY', user='_TheSpecialOne_', text="@switchfoot
     http://twitpic.com/2y1zl - Awww, that's a bummer.  You shoulda got David Carr of
     Third Day to do it. ;D"),
      Row(target=0, id=1467810672, date='Mon Apr 06 22:19:49 PDT 2009',
     flag='NO_QUERY', user='scotthamilton', text="is upset that he can't update his
     Facebook by texting it… and might cry as a result  School today also. Blah!"),
      Row(target=0, id=1467810917, date='Mon Apr 06 22:19:53 PDT 2009',
     flag='NO_QUERY', user='mattycus', text='@Kenichan I dived many times for the
     ball. Managed to save 50%  The rest go out of bounds'),
      Row(target=0, id=1467811184, date='Mon Apr 06 22:19:57 PDT 2009',
     flag='NO_QUERY', user='ElleCTF', text='my whole body feels itchy and like its on
     fire '),
      Row(target=0, id=1467811193, date='Mon Apr 06 22:19:57 PDT 2009',
     flag='NO_QUERY', user='Karoli', text="@nationwideclass no, it's not behaving at
     all. i'm mad. why am i here? because I can't see you all over there. ")]
```

Display number of rows and columns

```
[ ]: print(f"There are {df.count()} rows and  {len(df.columns)} columns in the␣
      ↪dataset.")
```

```
There are 1600000 rows and  6 columns in the dataset.
```

Display missing data if found

```
[ ]: df.select([func.count(func.when(func.isnan(c),c)).alias(c) for c in df.
      ↪columns]).toPandas().head()
```

```
[ ]:    target  id  date  flag  user  text
     0       0   0     0     0     0     0
```

Check for duplicates

```
[ ]: df = df.dropDuplicates()
     print(f"Number of rows in the dataframe after dropping the duplicates: {df.
      ↪count()}")
```

```
Number of rows in the dataframe after dropping the duplicates: 1600000
```

Display data type of all columns

```
[ ]: df.dtypes
```

```
[ ]: [('target', 'int'),
      ('id', 'bigint'),
      ('date', 'string'),
      ('flag', 'string'),
      ('user', 'string'),
      ('text', 'string')]
```

Drop all columns with no impact on the predictions

```
[ ]: drop_cols= ("id","date","flag","user")
     df = df.drop(*drop_cols)
```

Checking the schema

```
[ ]: df.printSchema()
```

```
root
 |-- target: integer (nullable = true)
 |-- text: string (nullable = true)
```

Display 5 random rows

```
[ ]: df.show(5, truncate = False)
```

```
+------+--------------------------------------------------------------------------
-----------------------------------------------------+
|target|text
```

```
|
+------+------------------------------------------------------------------------
-----------------------------------------------------+
|0     |At work
|
|0     |@astewart87 oh my gosh that made me emotional haha idk why!!! i dont
want to get old                                       |
|0     |i need new glasses…mines is hangnon 1 arm
|
|0     |Getting changed in the hopes that, that means we can go to the store
now!  Poor cat is out of food…oops.                  |
|0     |really now, time for sleep. dreaming of my city, more tattoos, and other
great things. waking up to early morning sociology |
+------+------------------------------------------------------------------------
-----------------------------------------------------+
only showing top 5 rows
```

Display the distinct target values in the target column where 0 is Negative and 4 for Positive

```
[ ]: df.select("target").distinct().show()
```

```
+------+
|target|
+------+
|     4|
|     0|
+------+
```

Changing the number that stands for positive from 4 -> 1

```
[ ]: df.createOrReplaceTempView('temp')
     df = spark.sql('SELECT CASE target WHEN 4 THEN 1.0  ELSE 0 END AS label, text␣
      ↪FROM temp')
     df.show(5, truncate = False)
```

```
+-----+------------------------------------------------------------------------
---------------------------------------------------+
|label|text
|
+-----+------------------------------------------------------------------------
---------------------------------------------------+
|0.0  |At work
|
|0.0  |@astewart87 oh my gosh that made me emotional haha idk why!!! i dont want
to get old                                      |
|0.0  |i need new glasses…mines is hangnon 1 arm
|
|0.0  |Getting changed in the hopes that, that means we can go to the store now!
```

```
Poor cat is out of food…oops.                    |
|0.0  |really now, time for sleep. dreaming of my city, more tattoos, and other
great things. waking up to early morning sociology |
+-----+--------------------------------------------------------------------------
---------------------------------------------------+
only showing top 5 rows
```

Print the last 5 rows

```
[ ]: df.tail(5)
```

```
[ ]: [Row(label=Decimal('1.0'), text='Installing office to my little netbook, going
 to do some work at a friends house in a short while. '),
  Row(label=Decimal('1.0'), text='man.. Ab got work.. ahhh!!! but later imma shop
 till i drop!! '),
  Row(label=Decimal('1.0'), text="@paupaula I'm joining The SPectrum maybe.
 Idk… But yeah I'm dedicating myself to my studies too  How? Idk."),
  Row(label=Decimal('1.0'), text='Friend over  My god! 300th Update!!'),
  Row(label=Decimal('1.0'), text='TIME MAG ONLINE is very interesting. Watching
 videos this morning. Enjoying free content… while it lasts. ')]
```

Count of positive and negatve tweets according to the dataset

```
[ ]: df.groupBy("label").count().show()
```

```
+-----+------+
|label| count|
+-----+------+
|  0.0|800000|
|  1.0|800000|
+-----+------+
```

2.3 Text Preprocessting

Remove stopwords, punctuations, links, and stem the data

```
[ ]: nltk.download('stopwords')
     stop_words = stopwords.words("english")
     stemmer = SnowballStemmer("english")
     text_cleaning_re = "@\S+|https?:\S+|http?:\S|[^A-Za-z0-9]+"
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
[ ]: def preprocess(text, stem=False):
         # Remove link,user and special characters
         text = re.sub(text_cleaning_re, ' ', str(text).lower()).strip()
         tokens = []
```

```
    for token in text.split():
        if token not in stop_words:
            if stem:
                tokens.append(stemmer.stem(token))
            else:
                tokens.append(token)
    return " ".join(tokens)
```

[ ]:
```
%%time
clean_text = func.udf(lambda x: preprocess(x), StringType())
df = df.withColumn('text_cleaned',clean_text(func.col("text")))
```

```
CPU times: user 6.88 ms, sys: 98 µs, total: 6.98 ms
Wall time: 77.9 ms
```

[ ]:
```
df.show()
```

```
+-----+--------------------+--------------------+
|label|                text|        text_cleaned|
+-----+--------------------+--------------------+
|  0.0|            At work |                work|
|  0.0|@astewart87 oh my...|oh gosh made emot...|
|  0.0|i need new glasse...|need new glasses ...|
|  0.0|Getting changed i...|getting changed h...|
|  0.0|really now, time ...|really time sleep...|
|  0.0|pfff i want to go...|pfff want go back...|
|  0.0|Currently watchin...|currently watchin...|
|  0.0|What a bad day! N...|bad day need comf...|
|  0.0|Tried to install ...|tried install twi...|
|  0.0|Having casual, un...|casual unprotecte...|
|  0.0|Good morning worl...|good morning worl...|
|  0.0|@pmarnandus re: d...|daily gossip well...|
|  0.0|Someone somewhere...|someone somewhere...|
|  0.0|@weblivz What a b...|boot would demand...|
|  0.0|@jobeaz damn, sor...|damn sorry missed...|
|  0.0|@klariza that's a...|awesome love stuf...|
|  0.0|my tv husbands ri...|tv husbands rick ...|
|  0.0|@thecoveted Oooh!...|oooh love earring...|
|  0.0|managed to fractu...|managed fracture ...|
|  0.0|@eolai Tea is lov...|tea lovely accide...|
+-----+--------------------+--------------------+
only showing top 20 rows
```

Since we have another column of cleaned text we will drop the original text column

[ ]:
```
df = df.drop("text")
```

2.4 Displaying Word Cloud

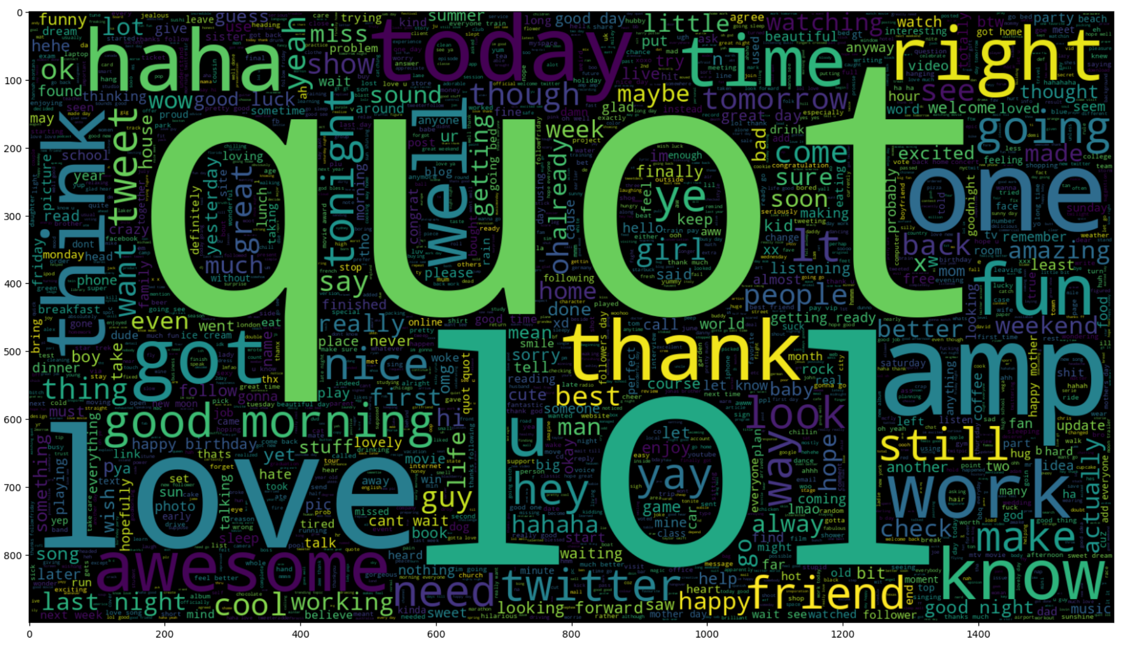
```
[ ]: pandas_df = df.toPandas()
     pandas_df.head()
```

```
[ ]:   label                                      text_cleaned
     0   0.0                                              work
     1   0.0  oh gosh made emotional haha idk dont want get old
     2   0.0                need new glasses mines hangnon 1 arm
     3   0.0  getting changed hopes means go store poor cat …
     4   0.0  really time sleep dreaming city tattoos great …
```

2.4.1 Positive Sentiments Word Cloud

```
[ ]: plt.figure(figsize = (20,16))
     wc = WordCloud(max_words = 2000 , width = 1600 , height = 900).generate(" ".
      ↪join(pandas_df[pandas_df["label"]==1.0].text_cleaned))
     plt.imshow(wc , interpolation = 'bilinear')
```

```
[ ]: <matplotlib.image.AxesImage at 0x7f8bb0d559f0>
```

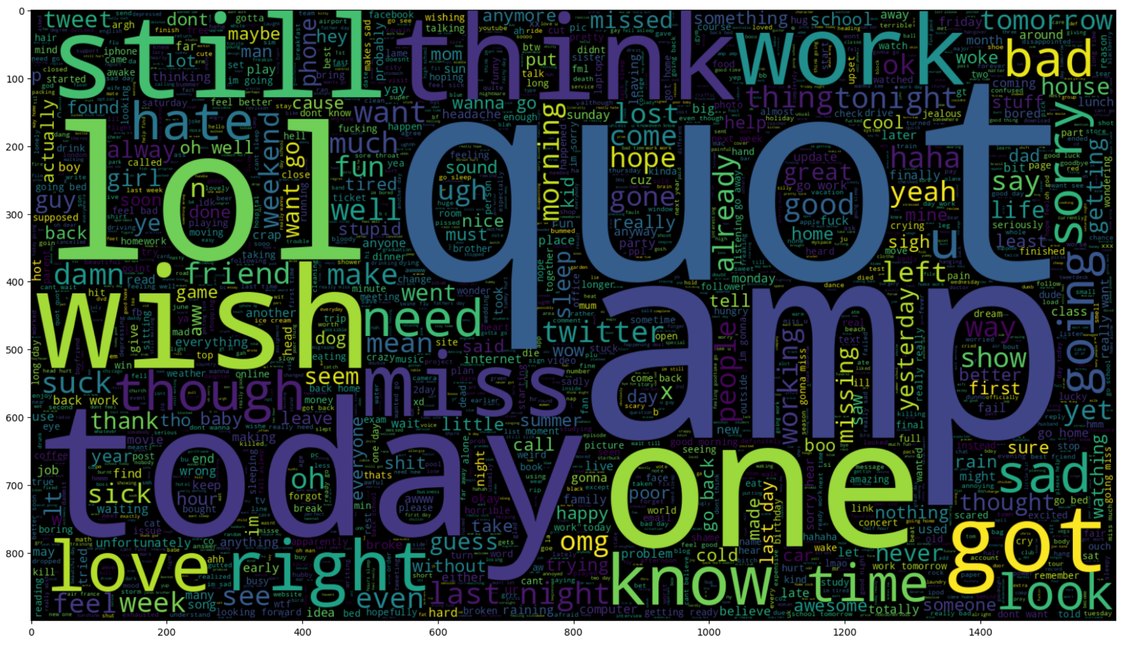


2.4.2 Negative Sentiments Word Cloud

```
[ ]: plt.figure(figsize = (20,16))
     wc = WordCloud(max_words = 2000 , width = 1600 , height = 900).generate(" ".
      ↪join(pandas_df[pandas_df["label"]==0.0].text_cleaned))
```

```
plt.imshow(wc , interpolation = 'bilinear')
```

[ ]: `<matplotlib.image.AxesImage at 0x7f8bb0cee8c0>`



2.5 Preparing Data for Model Building

Tokenizing the Text

[ ]:
```
tokenizer = Tokenizer(inputCol="text_cleaned", outputCol="words_tokens")
words_tokens = tokenizer.transform(df)
words_tokens.show()
```

```
+-----+--------------------+--------------------+
|label|        text_cleaned|        words_tokens|
+-----+--------------------+--------------------+
|  0.0|                work|              [work]|
|  0.0|oh gosh made emot…|[oh, gosh, made, …|
|  0.0|need new glasses …|[need, new, glass…|
|  0.0|getting changed h…|[getting, changed…|
|  0.0|really time sleep…|[really, time, sl…|
|  0.0|pfff want go back…|[pfff, want, go, …|
|  0.0|currently watchin…|[currently, watch…|
|  0.0|bad day need comf…|[bad, day, need, …|
|  0.0|tried install twi…|[tried, install, …|
|  0.0|casual unprotecte…|[casual, unprotec…|
|  0.0|good morning worl…|[good, morning, w…|
|  0.0|daily gossip well…|[daily, gossip, w…|
|  0.0|someone somewhere…|[someone, somewhe…|
|  0.0|boot would demand…|[boot, would, dem…|
|  0.0|damn sorry missed…|[damn, sorry, mis…|
|  0.0|awesome love stuf…|[awesome, love, s…|
|  0.0|tv husbands rick …|[tv, husbands, ri…|
|  0.0|oooh love earring…|[oooh, love, earr…|
|  0.0|managed fracture …|[managed, fractur…|
|  0.0|tea lovely accide…|[tea, lovely, acc…|
+-----+--------------------+--------------------+
only showing top 20 rows
```

Applying CountVectorizer

```
[ ]: count = CountVectorizer (inputCol="words_tokens", outputCol="rawFeatures")
model = count.fit(words_tokens)
featurizedData = model.transform(words_tokens)
featurizedData.show()
```

```
+-----+--------------------+--------------------+--------------------+
|label|        text_cleaned|        words_tokens|         rawFeatures|
+-----+--------------------+--------------------+--------------------+
|  0.0|                work|              [work]|  (262144,[7],[1.0])|
|  0.0|oh gosh made emot…|[oh, gosh, made, …|(262144,[2,26,30,…|
|  0.0|need new glasses …|[need, new, glass…|(262144,[25,33,10…|
|  0.0|getting changed h…|[getting, changed…|(262144,[4,56,235…|
|  0.0|really time sleep…|[really, time, sl…|(262144,[12,18,35…|
|  0.0|pfff want go back…|[pfff, want, go, …|(262144,[4,6,13,2…|
|  0.0|currently watchin…|[currently, watch…|(262144,[5,32,61,…|
|  0.0|bad day need comf…|[bad, day, need, …|(262144,[1,33,48,…|
|  0.0|tried install twi…|[tried, install, …|(262144,[7,39,121…|
|  0.0|casual unprotecte…|[casual, unprotec…|(262144,[20,81,20…|
|  0.0|good morning worl…|[good, morning, w…|(262144,[0,10,35,…|
|  0.0|daily gossip well…|[daily, gossip, w…|(262144,[24,39,40…|
|  0.0|someone somewhere…|[someone, somewhe…|(262144,[3,147,98…|
|  0.0|boot would demand…|[boot, would, dem…|(262144,[15,49,29…|
|  0.0|damn sorry missed…|[damn, sorry, mis…|(262144,[21,51,14…|
|  0.0|awesome love stuf…|[awesome, love, s…|(262144,[8,31,78,…|
|  0.0|tv husbands rick …|[tv, husbands, ri…|(262144,[66,201,3…|
|  0.0|oooh love earring…|[oooh, love, earr…|(262144,[8,31,124…|
|  0.0|managed fracture …|[managed, fractur…|(262144,[157,256,…|
|  0.0|tea lovely accide…|[tea, lovely, acc…|(262144,[176,317,…|
+-----+--------------------+--------------------+--------------------+
only showing top 20 rows
```

Applying Term Frequency - Inverse Document Frequency (TF-IDF)

```
[ ]: idf = IDF(inputCol="rawFeatures", outputCol="features")
     idfModel = idf.fit(featurizedData)
     rescaledData = idfModel.transform(featurizedData)

     rescaledData.select("label", "features").show()
```

```
+-----+--------------------+
|label|            features|
+-----+--------------------+
|  0.0|(262144,[7],[3.24…|
|  0.0|(262144,[2,26,30,…|
|  0.0|(262144,[25,33,10…|
|  0.0|(262144,[4,56,235…|
|  0.0|(262144,[12,18,35…|
|  0.0|(262144,[4,6,13,2…|
|  0.0|(262144,[5,32,61,…|
|  0.0|(262144,[1,33,48,…|
|  0.0|(262144,[7,39,121…|
|  0.0|(262144,[20,81,20…|
|  0.0|(262144,[0,10,35,…|
|  0.0|(262144,[24,39,40…|
|  0.0|(262144,[3,147,98…|
|  0.0|(262144,[15,49,29…|
|  0.0|(262144,[21,51,14…|
|  0.0|(262144,[8,31,78,…|
|  0.0|(262144,[66,201,3…|
|  0.0|(262144,[8,31,124…|
|  0.0|(262144,[157,256,…|
|  0.0|(262144,[176,317,…|
+-----+--------------------+
only showing top 20 rows
```

```
[ ]: df_final = rescaledData.select("label", "features")# We want only the label and␣
      ↪features columns for our machine learning models
```

Splitting the data into 70% training and 30% test dataset

```
[ ]: seed = 42  # set seed for reproducibility

     trainDF, testDF = df_final.randomSplit([0.7,0.3],seed)
```

2.6 Training the Model

```
[ ]: lr = LogisticRegression(labelCol = "label", featuresCol = "features",maxIter =␣
      ↪10)
     model = lr.fit(trainDF)
```

```
[ ]: predictions = model.transform(testDF)
```

```
[ ]: pred = predictions.toPandas()
     pred.head()
```

```
[ ]:   label                                         features  \
     0   0.0  (0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
     1   0.0  (0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
     2   0.0  (0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
     3   0.0  (0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
     4   0.0  (0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …

                                   rawPrediction  \
     0  [-0.27250436642405323, 0.27250436642405323]
     1  [-0.27250436642405323, 0.27250436642405323]
     2  [-0.27250436642405323, 0.27250436642405323]
     3  [-0.27250436642405323, 0.27250436642405323]
     4  [-0.27250436642405323, 0.27250436642405323]

                                  probability  prediction
     0  [0.4322923803374027, 0.5677076196625973]         1.0
     1  [0.4322923803374027, 0.5677076196625973]         1.0
     2  [0.4322923803374027, 0.5677076196625973]         1.0
     3  [0.4322923803374027, 0.5677076196625973]         1.0
     4  [0.4322923803374027, 0.5677076196625973]         1.0
```

2.7 Evaluating the Model

```
[ ]: evaluator = BinaryClassificationEvaluator(labelCol = "label",␣
      ↪metricName='areaUnderROC')
```

```
[ ]: areaUnderROC = evaluator.evaluate(predictions)
     print(f"The testing areaUnderROC of our Logistic Regression model is:␣
      ↪{areaUnderROC}")
```

```
The testing areaUnderROC of our Logistic Regression model is: 0.8165731721137901
```

2.7.1 Classification Report and Confusion Matrix

```
[ ]: y_true = pred['label'].astype('float')
     y_pred = pred['prediction']
```

```
[ ]: y_true.value_counts()
```

```
[ ]: 0.0    240100
     1.0    239582
     Name: label, dtype: int64
```

```
[ ]: y_pred.value_counts()
```

```
[ ]: 1.0    247858
     0.0    231824
     Name: prediction, dtype: int64
```

```
[ ]: print(classification_report(y_true, y_pred))
```

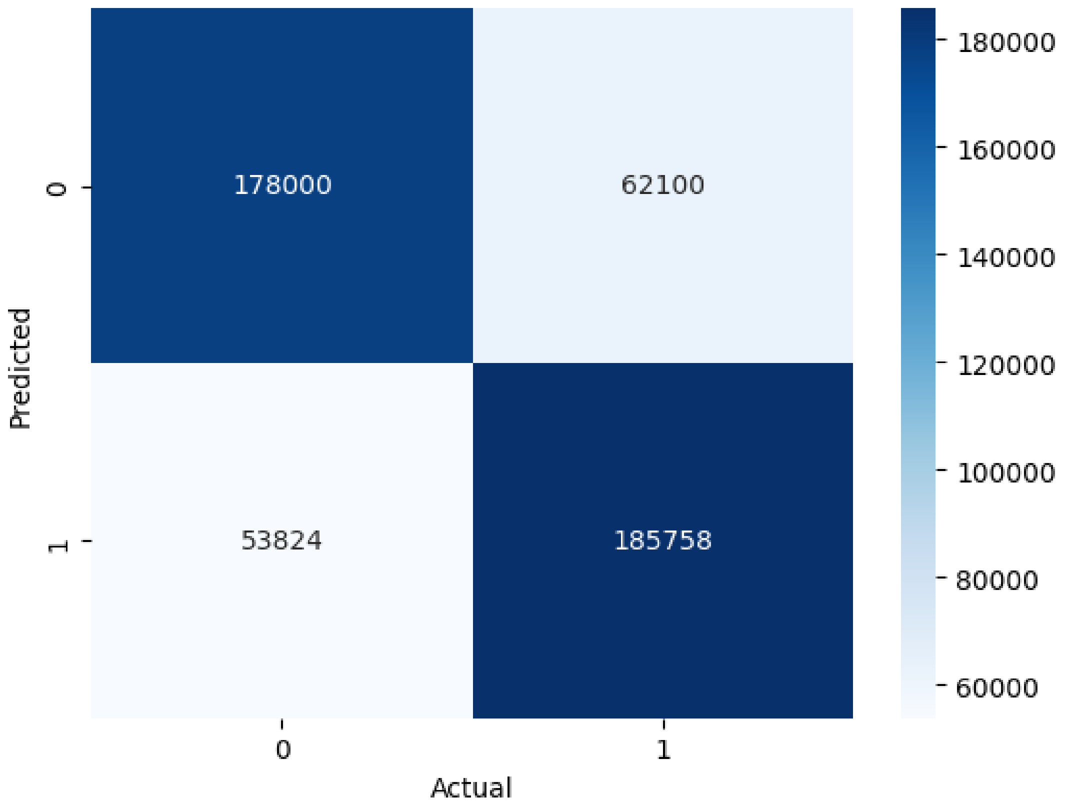
```
              precision    recall  f1-score   support

         0.0       0.77      0.74      0.75    240100
         1.0       0.75      0.78      0.76    239582

    accuracy                           0.76    479682
   macro avg       0.76      0.76      0.76    479682
weighted avg       0.76      0.76      0.76    479682
```

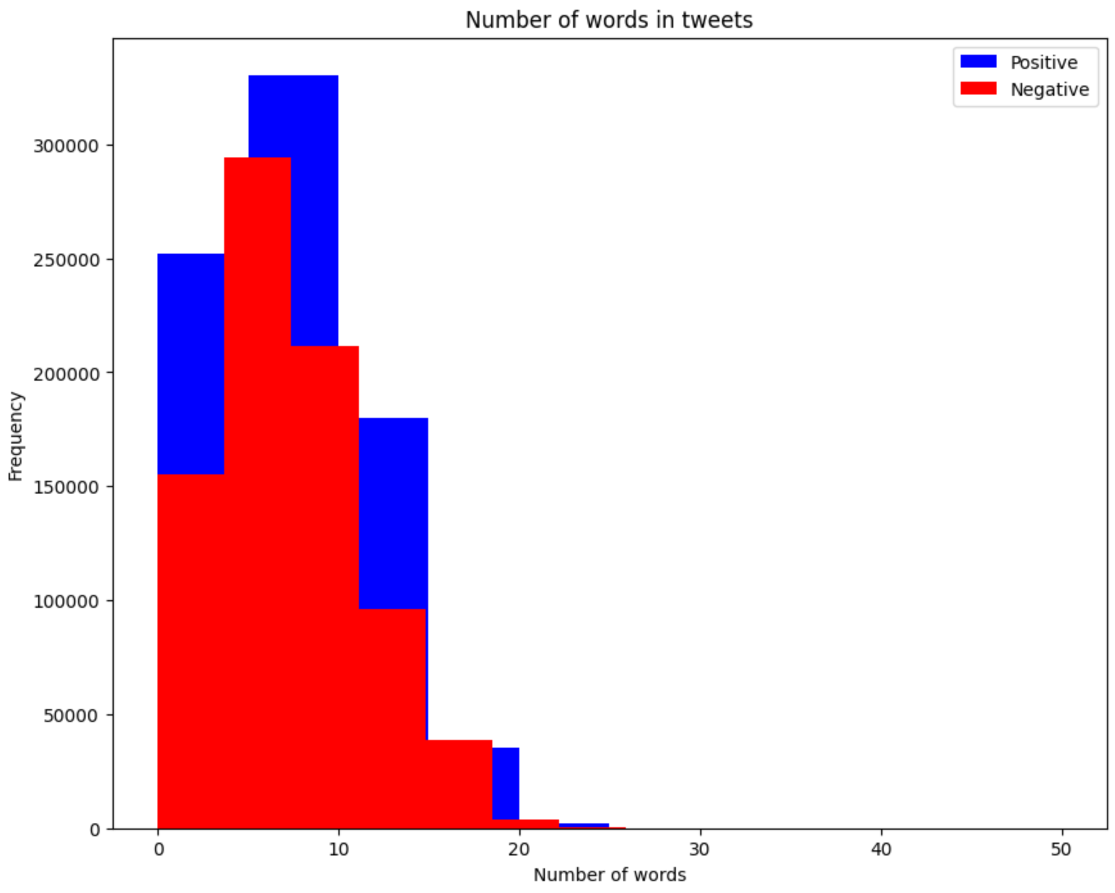
```
[ ]: sns.heatmap(confusion_matrix(y_true, y_pred), annot = True, fmt=  'd', cmap =␣
      ↪'Blues')
     plt.xlabel('Actual')
     plt.ylabel('Predicted')
```

```
[ ]: Text(50.72222222222214, 0.5, 'Predicted')
```



```
[ ]: # Histogram negative and positive words
     plt.figure(figsize = (10,8))
     plt.hist(pandas_df[pandas_df["label"]==1.0].text_cleaned.str.split().map(lambda␣
      ↪x: len(x)), color = 'blue', label = 'Positive')
     plt.hist(pandas_df[pandas_df["label"]==0.0].text_cleaned.str.split().map(lambda␣
      ↪x: len(x)), color = 'red', label = 'Negative')
     plt.legend()
     plt.title('Number of words in tweets')
     plt.xlabel('Number of words')
     plt.ylabel('Frequency')
```

```
[ ]: Text(0, 0.5, 'Frequency')
```


Number of words in tweets
Positive
Negative
Frequency
Number of words