# SENTIMENTAL ANALYSIS

PINISINGI SAIKARTHIK

VU21CSEN0100398

## Abstract

This report presents the development of a sentiment analysis model aimed at classifying text into predefined sentiment categories. The model leverages machine learning techniques, specifically a RandomForestClassifier, to predict sentiments based on textual input.

## Introduction

Sentiment analysis is a natural language processing (NLP) task that involves identifying and classifying the sentiment expressed in a piece of text. Sentiments can typically be categorized as positive, negative, or neutral. This project involves the development of a sentiment analysis model using a RandomForestClassifier. The model is trained on a dataset consisting of text labeled with sentiments and aims to predict the sentiment of new textual inputs accurately.

## Methodology

### Dataset:

| textID | text | selected_text | sentiment | | Time of Tv | Age of Use | Country | Populatior | Land Area | Density (P/Kr |
|---|---|---|---|---|---|---|---|---|---|---|
| cb774db0 | I`d have responded, if I were going | I`d have responded, if I were going | neutral | | morning | 0-20 | Afghanista | 38928346 | 652860 | 60 |
| 549e992a | Sooo SAD I will miss you here in San Diego!!! | Sooo SAD | negative | | noon | 21-30 | Albania | 2877797 | 27400 | 105 |
| 088c60f13 | my boss is bullying me... | bullying me | negative | | night | 31-45 | Algeria | 43851044 | 2381740 | 18 |
| 9642c003e | what interview! leave me alone | leave me alone | negative | | morning | 46-60 | Andorra | 77265 | 470 | 164 |
| 358bd9e8 | Sons of ****, why couldn`t they put them on the r | Sons of ****, | negative | | noon | 60-70 | Angola | 32866272 | 1246700 | 26 |
| 28b57f39 | http://www.dothebouncy.com/smf - some shame | http://www.dothebouncy.com/sm | neutral | | night | 70-100 | Antigua ar | 97929 | 440 | 223 |
| 6e0c6d75 | 2am feedings for the baby are fun when he is all sr | fun | positive | | morning | 0-20 | Argentina | 45195774 | 2736690 | 17 |
| 50e14c0bl | Soooo high | Soooo high | neutral | | noon | 21-30 | Armenia | 2963243 | 28470 | 104 |
| e050245fb | Both of you | Both of you | neutral | | night | 31-45 | Australia | 25499884 | 7682300 | 3 |
| fc2cbefa9 | Journey!? Wow... u just became cooler. hehe... (i | Wow... u just became cooler. | positive | | morning | 46-60 | Austria | 9006398 | 8.24E+04 | 109 |
| 2339a9b0 | as much as i love to be hopeful, i reckon the chanc | as much as i love to be hopeful, i re | neutral | | noon | 60-70 | Azerbaijar | 10139177 | 82658 | 123 |
| 16fab9f95 | I really really like the song Love Story by Taylor Swi | like | positive | | night | 70-100 | Bahamas | 393244 | 10010 | 39 |
| 74a76f6e0 | My Sharpie is running DANGERously low on ink | DANGERously | negative | | morning | 0-20 | Bahrain | 1701575 | 760 | 2239 |
| 04dd1d2e | i want to go to music tonight but i lost my voice. | lost | negative | | noon | 21-30 | Banglades | 1.65E+08 | 130170 | 1265 |
| bbe3cbf62 | test test from the LG enV2 | test test from the LG enV2 | neutral | | night | 31-45 | Barbados | 287375 | 430 | 668 |
| 8a939bfb5 | Uh oh, I am sunburned | Uh oh, I am sunburned | negative | | morning | 46-60 | Belarus | 9449323 | 202910 | 47 |
| 3440297f8 | S`ok, trying to plot alternatives as we speak *sigh* | *sigh* | negative | | noon | 60-70 | Belgium | 11589623 | 30280 | 383 |
| 919fa9339 | i`ve been sick for the past few days and thus, my h | sick | negative | | night | 70-100 | Belize | 397628 | 22810 | 17 |
| af3fed7fc | is back home now gonna miss every one | onna | negative | | morning | 0-20 | Benin | 12123200 | 112760 | 108 |
| 40e7becal | Hes just not that into you | Hes just not that into you | neutral | | noon | 21-30 | Bhutan | 771608 | 3.81E+04 | 20 |

The dataset used in this project is read from a CSV file (train.csv). It contains text data under the column selected_text and corresponding sentiment labels in the column sentiment.

## Data Preprocessing:

```
[53]: x = df['selected_text'].str.lower()
      y = df['sentiment']

      if len(x) > len(y):
          x = x[:len(y)]
      elif len(y) > len(x):
          y = y[:len(x)]

      df_cleaned = df.dropna(subset=['selected_text', 'sentiment'])

      x = df_cleaned['selected_text'].str.lower()
      y = df_cleaned['sentiment']
```

Text data is converted to lowercase to ensure uniformity.

Any null values in the selected_text and sentiment columns are removed.

The lengths of the feature (x) and target (y) variables are matched to avoid indexing errors.

## Label Encoding:

```
]: labelencoder = LabelEncoder()
   y = labelencoder.fit_transform(y)
```

Sentiment labels are converted to numerical format using LabelEncoder, which is necessary for training the machine learning model.

**Data Splitting:**

```
: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
```

The dataset is split into training and testing sets using an 80-20 ratio to evaluate the model's performance.

**Text Vectorization:**

The TfidfVectorizer is employed to transform the text data into a numerical format that can be fed into the RandomForestClassifier. The vectorizer also removes common English stop words.

**Model Training:**

A RandomForestClassifier is trained on the TF-IDF transformed training data.

**Model Evaluation:**

The trained model is evaluated on the test set, and its performance is measured using accuracy score.

**Prediction:**

The model is also capable of predicting the sentiment of new text inputs provided by the user.

## IMPLEMENTATION

[61]:
```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

[4]:
```
df = pd.read_csv("train.csv",encoding="latin1")
df.head(2)
```

[4]:

| | textID | text | selected_text | sentiment | Time of Tweet | Age of User | Country | Population -2020 | Land Area (Km²) | Density (P/Km²) |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | cb774db0d1 | I`d have responded, if I were going | I`d have responded, if I were going | neutral | morning | 0-20 | Afghanistan | 38928346 | 652860.0 | 60 |
| 1 | 549e992a42 | Sooo SAD I will miss you here in San Diego!!! | Sooo SAD | negative | noon | 21-30 | Albania | 2877797 | 27400.0 | 105 |

[53]:
```
x = df['selected_text'].str.lower()
y = df['sentiment']

if len(x) > len(y):
```

```
elif len(y) > len(x):
    y = y[:len(x)]

df_cleaned = df.dropna(subset=['selected_text', 'sentiment'])

x = df_cleaned['selected_text'].str.lower()
y = df_cleaned['sentiment']
```

```
[54]: from sklearn.model_selection import train_test_split
      from sklearn.preprocessing import StandardScaler,LabelEncoder
```

```
[55]: labelencoder = LabelEncoder()
      y = labelencoder.fit_transform(y)
```

```
[56]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
```

```
[57]: tfi = TfidfVectorizer(stop_words="english")
      x_train = tfi.fit_transform(x_train)
      x_test  = tfi.transform(x_test)
```

```
[58]: clr = RandomForestClassifier()
      clr.fit(x_train,y_train)
```



```
clr.fit(x_train,y_train)
```



```
[58]: RandomForestClassifier
      RandomForestClassifier()
```

```
[59]: y_pred = clr.predict(x_test)
```

```
[62]: print(accuracy_score(y_test,y_pred))

      0.7954876273653566
```

```
[78]: st = input()
      st = st.lower()
      x_t = tfi.transform([st])

       alright
```

```
[79]: y_pre = clr.predict(x_t)
```

```
[80]: print(labelencoder.inverse_transform(y_pre))

      ['neutral']
```

```
[ ]:
```

## Results

The model achieved an accuracy score on the test set, which provides an insight into its effectiveness at classifying sentiments.

The ability to input new text and receive a sentiment prediction demonstrates the model's practical utility.

```
[81]: st = input()
      st = st.lower()
      x_t = tfi.transform([st])
```

```
 you are nice
```

```
[82]: y_pre = clr.predict(x_t)
```

```
[83]: print(labelencoder.inverse_transform(y_pre))
```

```
['positive']
```

```
[84]: st = input()
      st = st.lower()
      x_t = tfi.transform([st])
```

```
 its really bad
```

```
[85]: y_pre = clr.predict(x_t)
```

```
[86]: print(labelencoder.inverse_transform(y_pre))
```

```
['negative']
```

```
[87]: st = input()
      st = st.lower()
      x_t = tfi.transform([st])
```

```
 its alright
```

```
[88]: y_pre = clr.predict(x_t)
```

```
[89]: print(labelencoder.inverse_transform(y_pre))
```

```
['neutral']
```

**Conclusion**

The sentiment analysis model developed in this project demonstrates a robust performance, as evidenced by its accuracy score. It effectively transforms text into a numerical format using TF-IDF vectorization and classifies sentiments with the help of a

RandomForestClassifier. The model's ability to predict sentiments for new inputs makes it a valuable tool for analyzing customer feedback, social media posts, and other text data where understanding sentiment is crucial.

**Future Work**

Potential improvements could include:

Experimenting with other machine learning algorithms such as Support Vector Machines (SVM) or deep learning models like LSTM.

Incorporating more advanced text preprocessing techniques.

Expanding the dataset for training to improve accuracy and robustness.

This report captures the essence of building and evaluating a sentiment analysis model using machine learning techniques, providing a foundational understanding of its implementation and effectiveness.