Bahria University,

Karachi Campus

A logo with text on it

Description automatically generated

LAB EXPERIMENT NO.

06

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | Using python implements VADER rules based classification algorithm to find the sentiments of different sentences. |
| 02 | Using python implements textBlob rules-based classification  algorithm to find the sentiments of different sentences and compare the results with task # 01. |
|  |  |

Submitted On:

27-03-2024

(Date: DD/MM/YYYY)

**Task No. 01**: Using python implements VADER rules based classification algorithm to find the sentiments of different sentences.

**Solution:**

def analyze\_sentiment(sentence):

sid\_obj = SentimentIntensityAnalyzer()

sentiment\_dict = sid\_obj.polarity\_scores(sentence)

print("Overall sentiment dictionary for the sentence is:", sentiment\_dict)

print("Sentence was rated as {:.2f}% Negative".format(sentiment\_dict['neg']\*100))

print("Sentence was rated as {:.2f}% Neutral".format(sentiment\_dict['neu']\*100))

print("Sentence was rated as {:.2f}% Positive".format(sentiment\_dict['pos']\*100))

print("Sentence Overall Rated As ", end="")

if sentiment\_dict['compound'] >= 0.05:

print("Positive")

elif sentiment\_dict['compound'] <= -0.05:

print("Negative")

else:

print("Neutral")

labels = ['Positive', 'Neutral', 'Negative']

sizes = [sentiment\_dict['pos'], sentiment\_dict['neu'], sentiment\_dict['neg']]

colors = ['green', 'gold', 'red']

explode = (0.1, 0, 0) # explode the 1st slice (Positive)

plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', shadow=True, startangle=140)

plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

plt.title("Sentiment Analysis")

plt.show()

sentences = [

"I love sunny days at the beach.",

"The movie was boring and uninteresting.",

"Eating ice cream makes me happy.",

"The traffic jam stressed me out.",

"Reading a good book relaxes me.",

"The customer service was excellent.",

"I feel lonely when it rains.",

"Winning the game was exhilarating.",

"The food at the restaurant was terrible.",

"Listening to music lifts my spirits."

]

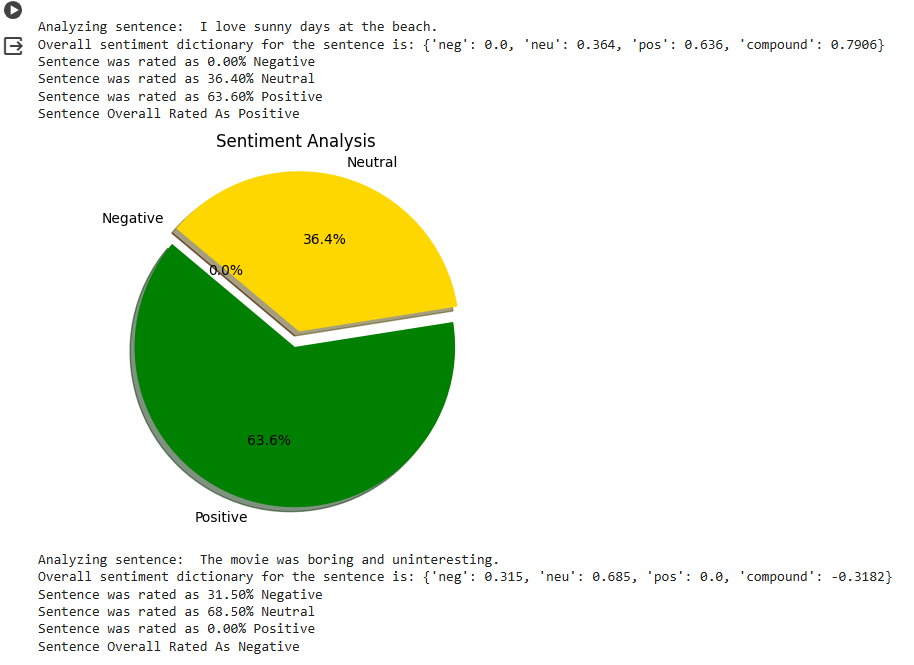
for sentence in sentences:

print("\nAnalyzing sentence: ", sentence)

analyze\_sentiment(sentence)

**Output:**

A screenshot of a computer

Description automatically generated****

**Task No. 02**: Using python implements textBlob rules-based classification algorithm to find the sentiments of different sentences and compare the results with task # 01.

**Solution:**

pip install textblob

from textblob import TextBlob

def analyze\_sentiment(sentence):

    # Create a TextBlob object

    blob = TextBlob(sentence)

    # Get the sentiment polarity

    sentiment\_polarity = blob.sentiment.polarity

    # Decide sentiment based on polarity

    if sentiment\_polarity > 0:

        sentiment = "Positive"

    elif sentiment\_polarity < 0:

        sentiment = "Negative"

    else:

        sentiment = "Neutral"

    return sentiment

sentences = [

    "I love sunny days at the beach.",

    "The movie was boring and uninteresting.",

    "Eating ice cream makes me happy.",

    "The traffic jam stressed me out.",

    "Reading a good book relaxes me.",

    "The customer service was excellent.",

    "I feel lonely when it rains.",

    "Winning the game was exhilarating.",

    "The food at the restaurant was terrible.",

    "Listening to music lifts my spirits."

]

for sentence in sentences:

    print("\nAnalyzing sentence: ", sentence)

    sentiment = analyze\_sentiment(sentence)

    print("Sentiment of the sentence '{}' is: {}".format(sentence, sentiment))

**Comparison**

for sentence in sentences:

# VaderSentiment analysis

sid\_obj = SentimentIntensityAnalyzer()

sentiment\_dict = sid\_obj.polarity\_scores(sentence)

vader\_sentiment = sentiment\_dict['compound']

# TextBlob analysis

blob = TextBlob(sentence)

textblob\_sentiment = blob.sentiment.polarity

# Print the results

print("\nSentence: ", sentence)

print("VaderSentiment: {:.2f}".format(vader\_sentiment))

print("TextBlob: {:.2f}".format(textblob\_sentiment))

# Compare the sentiments

if vader\_sentiment > 0 and textblob\_sentiment > 0:

print("Both VaderSentiment and TextBlob agree on positive sentiment.")

elif vader\_sentiment < 0 and textblob\_sentiment < 0:

print("Both VaderSentiment and TextBlob agree on negative sentiment.")

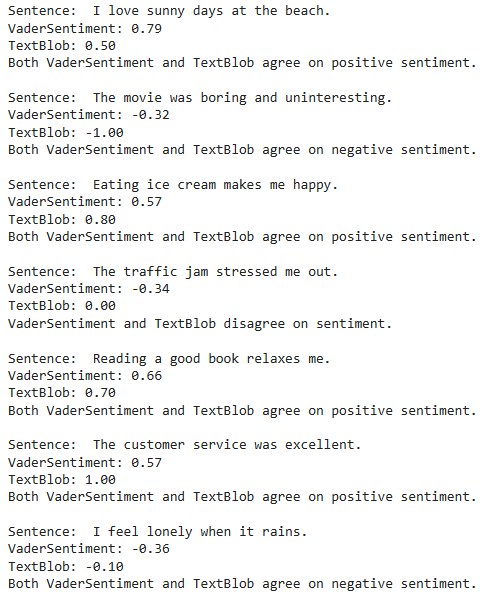
elif vader\_sentiment == 0 and textblob\_sentiment == 0:

print("Both VaderSentiment and TextBlob agree on neutral sentiment.")

else:

print("VaderSentiment and TextBlob disagree on sentiment.")

**Output:**

****

**Output:**

