comments-generator-tool

June 23, 2024

#COMMENTS GENERATOR

[1]:
```
# Install necessary libraries
!pip install nltk textblob vaderSentiment scikit-learn
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages
(3.8.1)
Requirement already satisfied: textblob in /usr/local/lib/python3.10/dist-
packages (0.17.1)
Collecting vaderSentiment
  Downloading vaderSentiment-3.3.2-py2.py3-none-any.whl (125 kB)
                             126.0/126.0

kB 2.1 MB/s eta 0:00:00
Requirement already satisfied: scikit-learn in
/usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages
(from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages
(from nltk) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in
/usr/local/lib/python3.10/dist-packages (from nltk) (2024.5.15)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages
(from nltk) (4.66.4)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from vaderSentiment) (2.31.0)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.25.2)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.11.4)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->vaderSentiment) (3.7)
```

```
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment)
(2024.6.2)
Installing collected packages: vaderSentiment
Successfully installed vaderSentiment-3.3.2
```

[2]:
```python
# Import necessary libraries
import nltk
from textblob import TextBlob
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

# Download the 'punkt' package from NLTK, necessary for TextBlob
nltk.download('punkt')

def analyze_content(text):
    """
    Analyze the sentiment of the given text using both TextBlob and VADER.

    Args:
    text (str): The text to analyze.

    Returns:
    tuple: A tuple containing the sentiment analysis results from TextBlob and␣
 ↪VADER.
    """

    # Analyze using TextBlob
    blob = TextBlob(text)
    textblob_sentiment = blob.sentiment  # Get the sentiment from TextBlob

    # Analyze using VADER
    analyzer = SentimentIntensityAnalyzer()
    vader_sentiment = analyzer.polarity_scores(text)  # Get the sentiment from␣
 ↪VADER

    return textblob_sentiment, vader_sentiment

# Example usage
text = "Your article on AI was incredibly insightful and well-written!"
textblob_sentiment, vader_sentiment = analyze_content(text)  # Analyze the␣
 ↪example text
print(textblob_sentiment)  # Print TextBlob sentiment result
print(vader_sentiment)  # Print VADER sentiment result
```

```
[nltk_data] Downloading package punkt to /root/nltk_data…
```

```
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

```
Sentiment(polarity=1.0, subjectivity=0.9)
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
```

[3]:
```python
# Import the random library for selecting random comments
import random

def generate_friendly_comment():
    """
    Generate a random friendly comment from a predefined list.

    Returns:
    str: A friendly comment.
    """
    # List of friendly comments
    friendly_comments = [
        "Great job on this article!",
        "I really enjoyed reading this!",
        "This is very insightful, thank you for sharing!",
        "You did an excellent job explaining this topic.",
        "This is a well-written and thoughtful piece!"
    ]
    # Return a random friendly comment
    return random.choice(friendly_comments)

def generate_funny_comment():
    """
    Generate a random funny comment from a predefined list.

    Returns:
    str: A funny comment.
    """
    # List of funny comments
    funny_comments = [
        "This article is as good as a hot cup of coffee in the morning!",
        "I was reading this and my cat looked interested too!",
        "Your article is so good, it should come with a laugh track!",
        "This made my day, seriously!",
        "Reading this was like finding a ten-dollar bill in my pocket!"
    ]
    # Return a random funny comment
    return random.choice(funny_comments)

def generate_congratulating_comment():
    """
    Generate a random congratulating comment from a predefined list.
```

```
    Returns:
    str: A congratulating comment.
    """
    # List of congratulating comments
    congratulating_comments = [
        "Congratulations on a well-written piece!",
        "You should be proud of this work!",
        "Kudos to you for this excellent article!",
        "Hats off to you for this insightful write-up!",
        "Great accomplishment with this article!"
    ]
    # Return a random congratulating comment
    return random.choice(congratulating_comments)

def generate_questioning_comment():
    """
    Generate a random questioning comment from a predefined list.

    Returns:
    str: A questioning comment.
    """
    # List of questioning comments
    questioning_comments = [
        "Can you elaborate more on this topic?",
        "I have a question about this part of your article...",
        "Could you explain how you came to this conclusion?",
        "What are your thoughts on the opposite viewpoint?",
        "Can you provide more details on this?"
    ]
    # Return a random questioning comment
    return random.choice(questioning_comments)

def generate_disagreement_comment():
    """
    Generate a random disagreement comment from a predefined list.

    Returns:
    str: A disagreement comment.
    """
    # List of disagreement comments
    disagreement_comments = [
        "I see things differently on this topic.",
        "I don't quite agree with this perspective.",
        "This point doesn't seem accurate to me.",
        "I have a different opinion on this matter.",
        "I respectfully disagree with this viewpoint."
    ]
```

```
    # Return a random disagreement comment
    return random.choice(disagreement_comments)

# Example usage
print(generate_friendly_comment())  # Print a random friendly comment
print(generate_funny_comment())  # Print a random funny comment
print(generate_congratulating_comment())  # Print a random congratulating␣
 ↪comment
print(generate_questioning_comment())  # Print a random questioning comment
print(generate_disagreement_comment())  # Print a random disagreement comment
```

```
You did an excellent job explaining this topic.
Your article is so good, it should come with a laugh track!
You should be proud of this work!
What are your thoughts on the opposite viewpoint?
I see things differently on this topic.
```

[4]:
```
def generate_comments(text):
    """
    Generate a set of different types of comments based on the provided text.

    Args:
    text (str): The text to analyze and generate comments for.

    Returns:
    dict: A dictionary containing various types of comments.
    """
    # Analyze the sentiment of the text using TextBlob and VADER
    textblob_sentiment, vader_sentiment = analyze_content(text)

    # Generate different types of comments
    comments = {
        "friendly": generate_friendly_comment(),  # Generate a friendly comment
        "funny": generate_funny_comment(),  # Generate a funny comment
        "congratulating": generate_congratulating_comment(),  # Generate a␣
 ↪congratulating comment
        "questioning": generate_questioning_comment(),  # Generate a␣
 ↪questioning comment
        "disagreement": generate_disagreement_comment()  # Generate a␣
 ↪disagreement comment
    }

    return comments  # Return the dictionary of comments

# Example usage
text = "Your article on AI was incredibly insightful and well-written!"
```

```
comments = generate_comments(text)  # Generate comments based on the example text
print(comments)  # Print the generated comments
```

```
{'friendly': 'I really enjoyed reading this!', 'funny': 'This article is as good as a hot cup of coffee in the morning!', 'congratulating': 'Hats off to you for this insightful write-up!', 'questioning': 'Can you provide more details on this?', 'disagreement': 'I have a different opinion on this matter.'}
```

[5]:
```
def main():
    """
    Main function to interactively generate and display comments based on user-provided text.
    """
    # Prompt the user to enter the text or article content
    text = input("Enter the text or article content: ")

    # Generate comments based on the entered text
    comments = generate_comments(text)

    # Print the generated comments
    print("\nGenerated Comments:")
    for comment_type, comment in comments.items():
        print(f"{comment_type.capitalize()} Comment: {comment}")

# If the script is run directly (not imported as a module), execute the main function
if __name__ == "__main__":
    main()
```

```
Enter the text or article content: Artificial intelligence is the science of making machines that can think like humans. It can do things that are considered "smart." AI technology can process large amounts of data in ways, unlike humans. The goal for AI is to be able to do things such as recognize patterns, make decisions, and judge like humans.

Generated Comments:
Friendly Comment: You did an excellent job explaining this topic.
Funny Comment: This made my day, seriously!
Congratulating Comment: Kudos to you for this excellent article!
Questioning Comment: What are your thoughts on the opposite viewpoint?
Disagreement Comment: I see things differently on this topic.
```