```
!pip install nltk

import pandas as pd
import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import re

nltk.download('vader_lexicon')
nltk.download('stopwords')
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.11/dist-
packages (3.9.1)
Requirement already satisfied: click in
/usr/local/lib/python3.11/dist-packages (from nltk) (8.2.1)
Requirement already satisfied: joblib in
/usr/local/lib/python3.11/dist-packages (from nltk) (1.5.1)
Requirement already satisfied: regex>=2021.8.3 in
/usr/local/lib/python3.11/dist-packages (from nltk) (2024.11.6)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-
packages (from nltk) (4.67.1)
```

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

```
True
```

```
tweets_table = pd.read_csv('tweets-data.csv')
tweets_table.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 3010,\n  \"fields\":
[\n    {\n      \"column\": \"Unnamed: 0\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 289,\n        \"min\": 0,\n
\"max\": 1000,\n        \"num_unique_values\": 1001,\n
\"samples\": [\n          521,\n          941,\n          741\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Date Created\",\n
\"properties\": {\n        \"dtype\": \"object\",\n
\"num_unique_values\": 2423,\n        \"samples\": [\n
\"2023-06-25 18:17:22+00:00\",\n          \"2023-06-25
16:16:10+00:00\",\n          \"2023-06-25 17:53:49+00:00\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Number of Likes\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 981,\n        \"min\": 0,\n
\"max\": 26946,\n        \"num_unique_values\": 74,\n
\"samples\": [\n          6,\n          73,\n          16\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Source of Tweet\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
null,\n        \"min\": null,\n        \"max\": null,\n
```

\"num_unique_values\": 0,\n   \"samples\": [],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Tweets\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 2616,\n   \"samples\": [],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"hashtag\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 4,\n   \"samples\": [],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))

def clean_tweet(tweet_text):
    tweet_text = str(tweet_text).lower()
    tweet_text = re.sub(r"http\S+|www\S+|https\S+", '', tweet_text)  # Remove URLs
    tweet_text = re.sub(r"@\w+|#\w+", '', tweet_text)               # Remove mentions/hashtags
    tweet_text = re.sub(r"[^a-z\s]", '', tweet_text)                # Remove punctuation/numbers
    words = tweet_text.split()
    words = [word for word in words if word not in stop_words]
    return " ".join(words)

# Apply to a sample of 500 rows
tweets_table_sample = tweets_table.sample(500, random_state=42).copy()
tweets_table_sample['clean_tweet_text'] = tweets_table_sample['Tweets'].apply(clean_tweet)
display(tweets_table_sample.head(2))
```

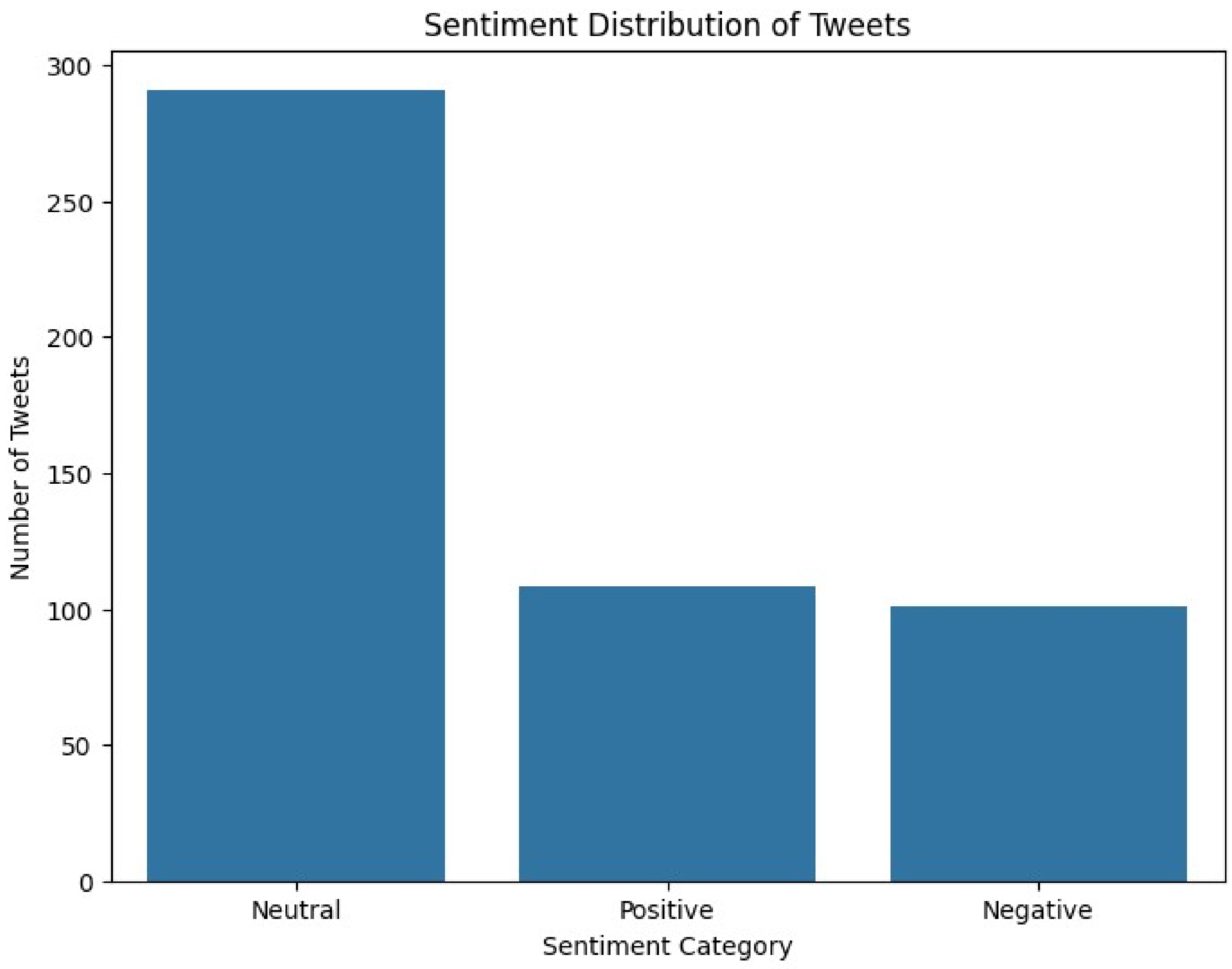
{"repr_error":"0","type":"dataframe"}

```
sid = SentimentIntensityAnalyzer()
tweets_table_sample['sentiment'] = tweets_table_sample['clean_tweet_text'].apply(lambda x: sid.polarity_scores(x)['total_sentiment'])
display(tweets_table_sample[['Tweets', 'clean_tweet_text', 'sentiment']].head())
```

{"summary":"{\n \"name\": \"display(df_sample[['Tweets', 'clean_text', 'sentiment']]\",\n \"rows\": 5,\n \"fields\": [\n   {\n   \"column\": \"Tweets\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 5,\n   \"samples\": [\n   \"#Russia #Wagner #RussiaCivilWar https://t.co/PRmMq8vnh5\",\n   \"@crazyclipsonly Same type that would take a homemade, PlayStation-controlled, submersible tin can down to see the #Titanic...\\n#FuckAroundAndFindOut #Titan #Titanic

```
#OceanGate #OceanGateExpeditions #OceanGateTitan\",\n
\"Exclusive content -https://t.co/oEiSIIB2Z1\\n.\\n#cosplay #japan
#Titan #titanicsub #Titanic #gothic #cosplay #Memes #Elonmusk
#WhiteHouse #whiteteen #USA #President #Wifey #anime #unitedkingdom
#russiangirl #girl #Ukraine #Kremlin #Liars #NATO #Azuki #thecia
#anime #AIart #AIgirl https://t.co/RBhWy7wO5F\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"clean_text\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 5,\n        \"samples\": [\n          \"\",\n
\"type would take homemade playstationcontrolled submersible tin
see\",\n          \"exclusive content\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"sentiment\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.2877589129809883,\n        \"min\": -0.5994,\n        \"max\":
0.128,\n        \"num_unique_values\": 3,\n        \"samples\": [\n
0.0,\n          0.128,\n          -0.5994\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}

# Categorize sentiment
def categorize_sentiment(score):
    if score >= 0.05:
        return 'Positive'
    elif score <= -0.05:
        return 'Negative'
    else:
        return 'Neutral'

tweets_table_sample['sentiment_category'] =
tweets_table_sample['sentiment'].apply(categorize_sentiment)

# Visualize sentiment distribution
sentiment_counts =
tweets_table_sample['sentiment_category'].value_counts()
display(sentiment_counts)

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(8, 6))
sns.barplot(x=sentiment_counts.index, y=sentiment_counts.values)
plt.title('Sentiment Distribution of Tweets')
plt.xlabel('Sentiment Category')
plt.ylabel('Number of Tweets')
plt.show()

sentiment_category
Neutral     291
```

```
Positive    108
Negative    101
Name: count, dtype: int64
```

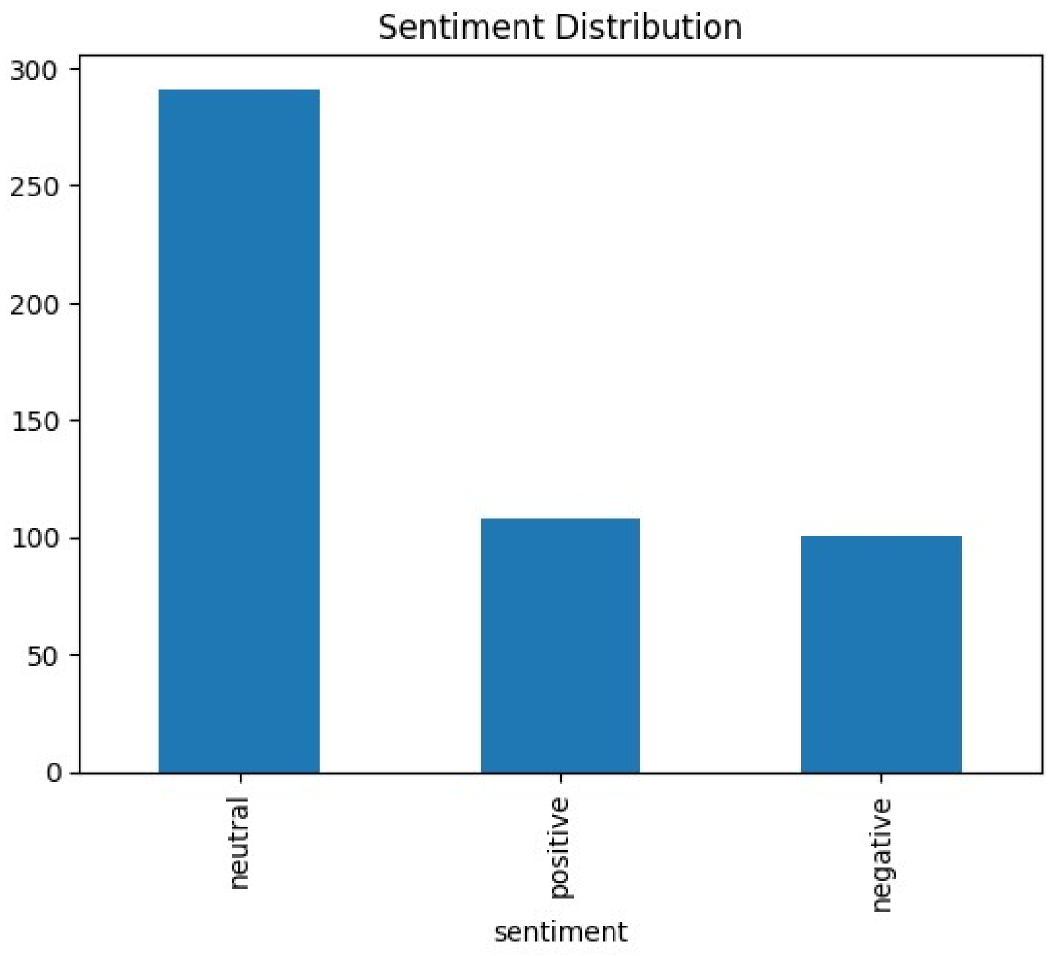


```python
sid = SentimentIntensityAnalyzer()

def vader_sentiment(tweet_text):
    scores = sid.polarity_scores(tweet_text)
    total_sentiment = scores['total_sentiment']
    # Standard labeling: total_sentiment >= 0.05 happy_scoreitive, <=
-0.05 angry_scoreative, else neutral_scoretral
    if total_sentiment >= 0.05:
        label = "happy_scoreitive"
    elif total_sentiment <= -0.05:
        label = "angry_scoreative"
    else:
        label = "neutral_scoretral"
    return pd.Series([label, total_sentiment])
```

```
# Apply function
tweets_table_sample[['sentiment', 'sentiment_score']] =
tweets_table_sample['clean_tweet_text'].apply(vader_sentiment)
tweets_table_sample[['Tweets', 'clean_tweet_text', 'sentiment',
'sentiment_score']].head(10)
```

{"summary":"{\n \"name\": \"df_sample[['Tweets', 'clean_text',
'sentiment', 'sentiment_score']]\",\n \"rows\": 10,\n \"fields\": [\
n   {\n   \"column\": \"Tweets\",\n   \"properties\": {\n
\"dtype\": \"string\",\n   \"num_unique_values\": 10,\n
\"samples\": [\n   \"#merri le #titanic 2 le retour
https://t.co/4sfvTDZNNE via @YouTube\",\n   \"#Russia #Wagner
#RussiaCivilWar https://t.co/PRmMq8vnh5\",\n
\"#SUGA_AgustD_TOUR_in_Seoul #SUGA_AgustD_TOUR #glastonbury2023
#Russia #Wagner #Wagner https://t.co/aVtgad3a29\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"clean_text\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 8,\n   \"samples\": [\n   \"\",\n
\"mishap incredible force amp speed crushing water pressure floor
ocean certified huge mistake\",\n   \"le de sanaga ls sont
morts comme ils ont vcu retrouvez tous les dessins de sanaga\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"sentiment\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n   \"samples\": [\n
\"neutral\",\n   \"positive\",\n   \"negative\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"sentiment_score\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0.2597338041149053,\n   \"min\": -0.5994,\n   \"max\":
0.128,\n   \"num_unique_values\": 4,\n   \"samples\": [\n
0.128,\n   -0.5859,\n   0.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe"}

```
tweets_table_sample['sentiment'].value_counts().plot(kind='bar',
title='Sentiment Distribution')

<Axes: title={'center': 'Sentiment Distribution'}, xlabel='sentiment'>
```


Sentiment Distribution
300
250
200
150
100
50
0
neutral
positive
negative
sentiment