Outline

Initialization and Data Loading:

- Initialize a Spark session.
- Load data from HDFS using PySpark.

Exploratory Data Analysis (EDA):

- Check the schema of the DataFrame.
- Display the first few rows of the DataFrame.
- Perform basic EDA, including checking the number of rows, columns, and summary statistics.
- Handle null values by dropping rows with null values.

Data Cleaning:

- Perform additional data cleaning steps, including dropping rows with 'undefined' sentiment.
- Check and display null values again after cleaning.

Data Exploration:

- Explore the distribution of tweets among different companies.
- Visualize the count and percentage of company tweets.

Sentiment Distribution:

- Analyze and visualize the distribution of sentiment categories.
- Explore sentiment distribution across different companies.

Text Analysis and Feature Engineering:

- Convert the 'polarity' column to float and define a UDF for sentiment conversion.
- Apply the sentiment UDF to create a new 'sentiment' column.
- Explore and visualize text-related features such as character and token length distributions.

Text Cleaning and Feature Engineering:

- Use the neattext library for text cleaning.
- Tokenize, remove stop words, and apply TF-IDF to the text.

Classification Models:

- Train and evaluate Random Forest, Decision Tree, and Logistic Regression classifiers.
- Evaluate models based on accuracy, precision, recall, and F1-score.

Sentiment Analysis Using Vader Lexicon:

- Utilize the VADER and TextBlob SentimentIntensityAnalyzer for sentiment analysis.
- Define UDFs for sentiment score and category conversion.
- Apply sentiment analysis to the cleaned text.
- Visualize the distribution of sentiments using VADER VADER and TextBlob.

Location-Based Analysis:

- Analyze the count of unique location names.
- Group data by 'location', 'group_name', and 'SentimentCategory'.
- Pivot the table for better visualization of sentiment distribution across locations.

In [ ]:
```
pip install hdfs
```

In [ ]:
```
pip install pyspark
```

Initialization of Spark Session and Data Loading:

In [1]:
```
from pyspark.sql import SparkSession

# Initialize Spark session
spark = SparkSession.builder.appName("BigTechSentimentAnalysis").getOrCrea
```

```
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogL
evel(newLevel).
23/12/12 12:54:31 WARN NativeCodeLoader: Unable to load native-hadoop library
for your platform... using builtin-java classes where applicable
```

In [2]:

```
# HDFS path to your file
hdfs_path = 'hdfs://localhost:9000/user/saikumargandham/combined_bigtech.c

# Read the file into a PySpark DataFrame
df_spark = spark.read.csv(hdfs_path, header=True, inferSchema=True)

# Show the schema of the DataFrame
df_spark.printSchema()

# Display the first few rows of the DataFrame
df_spark.show()

```

```
[Stage 1:>                                                          (0 + 8) /
8]
```

```
root
 |-- created_at: timestamp (nullable = true)
 |-- file_name: string (nullable = true)
 |-- followers: integer (nullable = true)
 |-- friends: integer (nullable = true)
 |-- group_name: string (nullable = true)
 |-- location: string (nullable = true)
 |-- retweet_count: string (nullable = true)
 |-- screenname: string (nullable = true)
 |-- search_query: string (nullable = true)
 |-- text: string (nullable = true)
 |-- twitter_id: string (nullable = true)
 |-- username: string (nullable = true)
 |-- polarity: string (nullable = true)
 |-- partition_0: string (nullable = true)
 |-- partition_1: string (nullable = true)

+-------------------+---------+---------+-------+----------+--------------+--
-----------+-------------+------------+--------------------+---------------
----+--------------------+--------+-----------+-----------+
|         created_at|file_name|followers|friends|group_name|      location|re
tweet_count|    screenname|search_query|                text|          twitte
r_id|            username|polarity|partition_0|partition_1|
+-------------------+---------+---------+-------+----------+--------------+--
-----------+-------------+------------+--------------------+---------------
----+--------------------+--------+-----------+-----------+
|2020-07-12 09:24:26|      AMD|       25|    114|       AMD|United Kingdom|
0.0| moffphcgaming|        #AMD|Been on holiday s...|1.282244417466884...|
⛪MoffPHC Gaming⛪| -0.3102| Technology|        AMD|
|2020-07-12 09:09:36|      AMD|      159|   1144|       AMD|  digitalverse|
4.0|    ironparr0t|        #AMD|RT @NinjaParanoid...|1.282240682023719...|
ironparrot|     0.0| Technology|        AMD|
|2020-07-12 08:37:31|      AMD|     4931|      7|       AMD|          NULL|
0.0|   ASUS_ROG_IN|        #AMD|The Beast #StrixG...|1.282232609427226...|
ASUS ROG IN|     0.0| Technology|        AMD|
|2020-07-12 08:31:24|      AMD|        9|    188|       AMD|   HAHA no....|
0.0|   XApochrypha|        #AMD|Recently purchase...|1.282231068649652...|
XenosApochrypha|     0.0| Technology|        AMD|
|2020-07-12 08:16:45|      AMD|     1719|      1|       AMD|  digitalocean|
1.0|   LinuxDreams|        #AMD|"RT @LinuxReviews...|1.282227383932772...|
LinuxDreams| -0.3612| Technology|        AMD|
|2020-07-12 08:16:44|      AMD|      394|      4|       AMD|          NULL|
3.0|botxboxseriesx|        #AMD|RT @RedGamingTech...|1.282227377754583...|
botxboxseriesx|   -0.34| Technology|        AMD|
|2020-07-12 08:12:33|      AMD|       19|    146|       AMD|          NULL|
3.0|   Tippy_Power|        #AMD|RT @RedGamingTech...|1.282226328067072...|
Tippy|   -0.34| Technology|        AMD|
|2020-07-12 08:11:41|      AMD|       69|    135|       AMD|     Amsterdam|
1.0|  LinuxReviews|        #AMD|"#Linux architect...|1.282226107027271...|
LinuxReviews| -0.3612| Technology|        AMD|
|2020-07-12 07:50:29|      AMD|      520|    293|       AMD| Wolverhampton|
3.0|    Wolves_LOC|        #AMD|RT @CollegeOptomU...|1.282220772816310...|
WolvesLOC|     0.0| Technology|        AMD|
|2020-07-12 07:30:05|      AMD|      369|    899|       AMD|          NULL|
0.0|     cnxplayer|        #AMD|Cast videos from ...|1.282215640246313...|
CnX Player|  0.4019| Technology|        AMD|
|2020-07-12 07:09:54|      AMD|     1833|    144|       AMD|   New Zealand|
```

```
2.0|     __Langley|        #AMD|RT @Dewton1992: (...|1.282210558461177...|
don’t mind me|     0.0| Technology|        AMD|
|2020-07-12 07:05:09|      AMD|      864|      5|       AMD|          NULL|
2.0|    TwitchBuds|        #AMD|RT @Dewton1992: (...|1.282209363504103...|
Twitch Buds|     0.0| Technology|        AMD|
|2020-07-12 06:44:11|      AMD|      250|    739|       AMD|  Florida, USA|
2.0|    Dewton1992|        #AMD|(PS4) MOTORCYCLE ...|1.282204087476420...|Boo
Boo Kitty Fuc...|     0.0| Technology|        AMD|
|2020-07-12 05:52:48|      AMD|      221|      1|       AMD|          NULL|
1.0| cooltechrobot|        #AMD|RT @LamarMK: TP-L...|1.282191156865097...|
Cool Tech Bot|  0.7125| Technology|        AMD|
|2020-07-12 05:51:05|      AMD|       12|      9|       AMD|           USA|
1.0|       LamarMK|        #AMD|TP-Link AX3000 Wi...|1.282190723811627...|
Lamar MK|  0.7125| Technology|        AMD|
|2020-07-12 05:43:57|      AMD|     2735|      1|       AMD|          NULL|
1.0|   ReGamertron|        #AMD|RT @Radiorole: 🔥...|1.282188928448188...|
Re:Gamertron|  0.5106| Technology|        AMD|
|2020-07-12 04:48:39|      AMD|        6|      0|       AMD|       Kolkata|
0.0|    MbtTraders|        #AMD|AMD Ryzen™ proces...|1.282175014683373...|
MBT TRADERS|     0.0| Technology|        AMD|
|2020-07-12 04:42:26|      AMD|       24|    257|       AMD|          NULL|
0.0|    DhW_SKILLZ|        #AMD|Should I go with ...|1.282173448761028...|
DhW SKILLZ|     0.0| Technology|        AMD|
|2020-07-12 04:26:15|      AMD|      230|     11|       AMD| United States|
1.0|    DivinityRT|        #AMD|RT @PCgeekinfo: u...|1.282169376263856...|
DivinityRT|     0.0| Technology|        AMD|
|2020-07-12 04:25:45|      AMD|      961|    214|       AMD| Sunnyvale, CA|
1.0|    PCgeekinfo|        #AMD|u/tanvirsingh 😂 ...|1.282169249713328...|
PCgeek|     0.0| Technology|        AMD|
+------------------+---------+---------+-------+----------+--------------+--
-----------+--------------+------------+--------------------+---------------
----+--------------------+--------+-----------+-----------+
only showing top 20 rows
```

EDA

In [3]:
```
# Perform basic EDA
df_spark.printSchema()
print("Number of rows:", df_spark.count())
print("Number of columns:", len(df_spark.columns))
```

```
root
 |-- created_at: timestamp (nullable = true)
 |-- file_name: string (nullable = true)
 |-- followers: integer (nullable = true)
 |-- friends: integer (nullable = true)
 |-- group_name: string (nullable = true)
 |-- location: string (nullable = true)
 |-- retweet_count: string (nullable = true)
 |-- screenname: string (nullable = true)
 |-- search_query: string (nullable = true)
 |-- text: string (nullable = true)
 |-- twitter_id: string (nullable = true)
 |-- username: string (nullable = true)
 |-- polarity: string (nullable = true)
 |-- partition_0: string (nullable = true)
 |-- partition_1: string (nullable = true)

Number of rows: 1133004
Number of columns: 15
```

In [4]:
```
# Display summary statistics
df_spark.describe().show()
```

```
23/12/12 12:55:07 WARN SparkStringUtils: Truncated the string representation
of a plan since it was too large. This behavior can be adjusted by setting 's
park.sql.debug.maxToStringFields'.
[Stage 6:=================================================>       (7 + 1) /
8]

+-------+---------+-----------------+-----------------+----------+---------
--------+-----------------+-------------------+-----------+--------------
-----+-------------------+-------------------+--------------------+--------
-----------+--------------------+
|summary|file_name|        followers|          friends|group_name|
location|     retweet_count|         screenname|search_query|
text|          twitter_id|           username|            polarity|
partition_0|        partition_1|
+-------+---------+-----------------+-----------------+----------+---------
--------+-----------------+-------------------+-----------+--------------
-----+-------------------+-------------------+--------------------+--------
-----------+--------------------+
|  count|  1133004|          1133004|          1133004|   1133004|
810400|          1133004|             1133004|     1133004|             1133
002|            1132994|            1132964|            1132993|
1132995|             1132996|
|   mean|     NULL| 8228.402585516027|2111.7848648371937|      NULL|
Infinity|164.34559891119093|1.510735075758921...|        NULL|
NULL|1.297846426141480...|                NaN|3.833162450143584E15|2.2168174
28579606...|2.703740512522227...|
| stddev|     NULL|128252.27006513478|12448.919756692145|      NULL|
NaN|1925.7989485159526|1.464244989583906...|        NULL|                NULL
|1.355294143378008E16|                NaN|7.046665387633437...| 4.8882188808
1344E17|5.276605666374685...|
|    min|      AMD|                0|                0|       AMD|\t.ɘʌɐ uo
oɯʎəuoɥ |               MD"|                0.0|        #AMD| -- DIFFERENT W
AL...|   @xamarinhq  #t...|             simply|  #Netflix and… h...|      Wi
-Fi adapater|   USB (for RAM b...|
|    max|  Youtube|         14443441|          1168982|   Youtube|      🏐
- she/her|           9993.0|       zzzzzebra1224|      cutety|🫖 @otter_ai
punc...|😥😥😥😥😥 #Netfl...|    Berz|  👖🚀Flywheel🔋🔥🪫|
🦢 Mike|          🦎👴🍳☀|
+-------+---------+-----------------+-----------------+----------+---------
--------+-----------------+-------------------+-----------+--------------
-----+-------------------+-------------------+--------------------+--------
-----------+--------------------+
```

In [5]:
```
from pyspark.sql.functions import col, sum

# Check null values in each column
null_counts = df_spark.agg(*[sum(col(c).isNull().cast("int")).alias(c + '_
# Display the null counts
null_counts.show()

```

```
[Stage 9:>                                                          (0 + 8) /
8]

+--------------------+--------------------+--------------------+------------
------+--------------------+--------------------+-----------------------+---
------------------+----------------------+---------------+-----------------
---+--------------------+--------------------+----------------------+----------
------------+
|created_at_null_count|file_name_null_count|followers_null_count|friends_null
_count|group_name_null_count|location_null_count|retweet_count_null_count|scr
eenname_null_count|search_query_null_count|text_null_count|twitter_id_null_co
unt|username_null_count|polarity_null_count|partition_0_null_count|partition_
1_null_count|
+--------------------+--------------------+--------------------+------------
------+--------------------+--------------------+-----------------------+---
------------------+----------------------+---------------+-----------------
---+--------------------+--------------------+----------------------+----------
------------+
|                   0|                   0|                   0|
0|                   0|              322604|                   0|
0|                   0|                   2|                  10|
40|                  11|                   9|                   8|
+--------------------+--------------------+--------------------+------------
------+--------------------+--------------------+-----------------------+---
------------------+----------------------+---------------+-----------------
---+--------------------+--------------------+----------------------+----------
------------+
```

In [6]:

```
# Drop rows with any null values
df_spark= df_spark.na.drop()

# Show the cleaned DataFrame
df_spark.show()
```

```
+------------------+---------+---------+-------+----------+----------------
---+-------------+-------------+-----------+--------------------+----------
---------+--------------------+--------+-----------+-----------+
|         created_at|file_name|followers|friends|group_name|            locat
ion|retweet_count|   screenname|search_query|                text|          t
witter_id|            username|polarity|partition_0|partition_1|
+------------------+---------+---------+-------+----------+----------------
---+-------------+-------------+-----------+--------------------+----------
---------+--------------------+--------+-----------+-----------+
|2020-07-12 09:24:26|      AMD|       25|    114|       AMD|      United King
dom|          0.0|moffphcgaming|        #AMD|Been on holiday s...|1.282244417
466884...|  ⛪MoffPHC Gaming⛪| -0.3102| Technology|        AMD|
|2020-07-12 09:09:36|      AMD|      159|   1144|       AMD|        digitalve
rse|          4.0|   ironparr0t|        #AMD|RT @NinjaParanoid...|1.282240682
023719...|          ironparrot|     0.0| Technology|        AMD|
|2020-07-12 08:31:24|      AMD|        9|    188|       AMD|         HAHA n
o....|          0.0|  XApochrypha|        #AMD|Recently purchase...|1.2822310
68649652...|     XenosApochrypha|     0.0| Technology|        AMD|
|2020-07-12 08:16:45|      AMD|     1719|      1|       AMD|        digitaloc
ean|          1.0|  LinuxDreams|        #AMD|"RT @LinuxReviews...|1.282227383
932772...|         LinuxDreams| -0.3612| Technology|        AMD|
|2020-07-12 08:11:41|      AMD|       69|    135|       AMD|           Amster
dam|          1.0| LinuxReviews|        #AMD|"#Linux architect...|1.282226107
027271...|        LinuxReviews| -0.3612| Technology|        AMD|
|2020-07-12 07:50:29|      AMD|      520|    293|       AMD|       Wolverhamp
ton|          3.0|   Wolves_LOC|        #AMD|RT @CollegeOptomU...|1.282220772
816310...|           WolvesLOC|     0.0| Technology|        AMD|
|2020-07-12 07:09:54|      AMD|     1833|    144|       AMD|         New Zeal
and|          2.0|    __Langley|        #AMD|RT @Dewton1992: (...|1.282210558
461177...|       don’t mind me|     0.0| Technology|        AMD|
|2020-07-12 06:44:11|      AMD|      250|    739|       AMD|        Florida,
USA|          2.0|   Dewton1992|        #AMD|(PS4) MOTORCYCLE ...|1.282204087
476420...|Boo Boo Kitty Fuc...|     0.0| Technology|        AMD|
|2020-07-12 05:51:05|      AMD|       12|      9|       AMD|
USA|          1.0|      LamarMK|        #AMD|TP-Link AX3000 Wi...|1.282190723
811627...|            Lamar MK|  0.7125| Technology|        AMD|
|2020-07-12 04:48:39|      AMD|        6|      0|       AMD|             Kolk
ata|          0.0|   MbtTraders|        #AMD|AMD Ryzen™ proces...|1.282175014
683373...|         MBT TRADERS|     0.0| Technology|        AMD|
|2020-07-12 04:26:15|      AMD|      230|     11|       AMD|       United Sta
tes|          1.0|   DivinityRT|        #AMD|RT @PCgeekinfo: u...|1.282169376
263856...|          DivinityRT|     0.0| Technology|        AMD|
|2020-07-12 04:25:45|      AMD|      961|    214|       AMD|       Sunnyvale,
CA|          1.0|   PCgeekinfo|        #AMD|u/tanvirsingh 😂 ...|1.282169249
713328...|              PCgeek|     0.0| Technology|        AMD|
|2020-07-12 02:31:13|      AMD|       18|    113|       AMD|
Wa|          0.0|   Sixish6Six|        #AMD|I mean the 2700x ...|1.2821404279
57624...|              Sixish|  0.9177| Technology|        AMD|
|2020-07-12 02:22:50|      AMD|       34|    155|       AMD|       San Franci
sco|          0.0|  NdrewGarcia|        #AMD|#AMD stuck in a r...|1.282138318
252028...|          Encino_Man|   -0.25| Technology|        AMD|
|2020-07-12 02:01:04|      AMD|      143|    546|       AMD|         Seattle,
WA|          0.0|    Star_Barf|        #AMD|After HOURS of tr...|1.2821328407
51132...|         Rob n Pants|  0.3147| Technology|        AMD|
|2020-07-12 01:53:19|      AMD|      134|     34|       AMD|       Mumbai, In
dia|          0.0|  HardwareBBQ|        #AMD|.@AMD shipped @AM...|1.282130889
820913...|        Hardware BBQ|     0.0| Technology|        AMD|
```

```
|2020-07-12 00:55:03|      AMD|       85|    122|       AMD|                O
hio|          0.0|   specsifier|        #AMD|Didn't get the ri...|1.282116224
705536...|          Specsifier|   0.904| Technology|        AMD|
|2020-07-12 00:53:53|      AMD|     6390|   5844|       AMD|                #
PDX|          0.0|  RobShiveley|        #AMD|Leaked AMD EPYC M...|1.282115930
500333...|Marketing & Sales...| -0.1531| Technology|        AMD|
|2020-07-12 00:39:18|      AMD|      213|    433|       AMD|Santa Cruz Tacac
h...|          0.0|altatensionmx|        #AMD|Vendo #TarjetaGra...|1.28211226
2141104...|  Jonathan M. Hunter|     0.0| Technology|        AMD|
|2020-07-12 00:34:49|      AMD|       47|     53|       AMD|              愛
知県|          6.0|  PARZIVALX40|        #AMD|RT @hms1193: #AMD...|1.28211113
4506479...|          PARZIVAL40|     0.0| Technology|        AMD|
+-------------------+---------+---------+-------+----------+----------------
---+-------------+-------------+-----------+--------------------+-----------
---------+--------------------+--------+-----------+-----------+
only showing top 20 rows
```

In [7]:

```
1 # Check null values in each column and get the count
2 null_counts = df_spark.agg(*[sum(col(c).isNull().cast("int")).alias(c + '_
3
4 # Display the null counts
5 null_counts.show()
```

```
[Stage 13:=======>                                                 (1 + 7) /
8]

+--------------------+--------------------+--------------------+------------
------+---------------------+--------------------+-----------------------+---
-----------------+----------------------+---------------+-----------------
---+--------------------+--------------------+----------------------+----------
------------+
|created_at_null_count|file_name_null_count|followers_null_count|friends_null
_count|group_name_null_count|location_null_count|retweet_count_null_count|scr
eenname_null_count|search_query_null_count|text_null_count|twitter_id_null_co
unt|username_null_count|polarity_null_count|partition_0_null_count|partition_
1_null_count|
+--------------------+--------------------+--------------------+------------
------+---------------------+--------------------+-----------------------+---
-----------------+----------------------+---------------+-----------------
---+--------------------+--------------------+----------------------+----------
------------+
|                   0|                   0|                   0|
0|                   0|                   0|                   0|
0|                   0|                   0|                   0|
0|                   0|                   0|                   0|
+--------------------+--------------------+--------------------+------------
------+---------------------+--------------------+-----------------------+---
-----------------+----------------------+---------------+-----------------
---+--------------------+--------------------+----------------------+----------
------------+
```

In [8]:
```
# Assuming df_spark is your DataFrame
column_data_types = df_spark.dtypes

# Display the data types of each column
for col_name, col_type in column_data_types:
    print(f"Column: {col_name}, Data Type: {col_type}")

```

```
Column: created_at, Data Type: timestamp
Column: file_name, Data Type: string
Column: followers, Data Type: int
Column: friends, Data Type: int
Column: group_name, Data Type: string
Column: location, Data Type: string
Column: retweet_count, Data Type: string
Column: screenname, Data Type: string
Column: search_query, Data Type: string
Column: text, Data Type: string
Column: twitter_id, Data Type: string
Column: username, Data Type: string
Column: polarity, Data Type: string
Column: partition_0, Data Type: string
Column: partition_1, Data Type: string
```

In [10]:

```
from pyspark.sql.functions import col

def basic_eda(df_spark, row_limit=5, list_elements_limit=10):
    # Rows and columns
    print('Info : There are {} columns in the dataset'.format(len(df_spark
    print('Info : There are {} rows in the dataset'.format(df_spark.count(

    print("================================================")

    # Data types
    print("\nData type information of different columns")
    dtypes_df_spark = df_spark.dtypes
    cat_cols = [col_name for col_name, col_type in dtypes_df_spark if col_
    num_cols = [col_name for col_name, col_type in dtypes_df_spark  if col

    print('Info : There are {} categorical columns'.format(len(cat_cols)))
    print('Info : There are {} numerical columns'.format(len(num_cols)))

    if list_elements_limit >= len(cat_cols):
        print("Categorical columns : ", cat_cols)
    else:
        print("Categorical columns : ", cat_cols[:list_elements_limit])

    if list_elements_limit >= len(num_cols):
        print("Numerical columns : ", num_cols)
    else:
        print("Numerical columns : ", num_cols[:list_elements_limit])

    # Display data types
    for col_name, col_type in dtypes_df_spark :
        print("{}: {}".format(col_name, col_type))

    print("================================================")
    print("\nDescription of numerical variables")

    # Describing numerical columns
    df_spark .select(num_cols).describe().show(row_limit)

    print("================================================")
    print("\nDescription of categorical variables")

    # Describing categorical columns
    df_spark.select(cat_cols).describe().show(row_limit)

    return
```

In [11]:
```
1 basic_eda(df_spark)
```

```
Info : There are 15 columns in the dataset


Info : There are 810383 rows in the dataset
==================================================

Data type information of different columns
Info : There are 12 categorical columns
Info : There are 3 numerical columns
Categorical columns :  ['file_name', 'group_name', 'location', 'retweet_coun
t', 'screenname', 'search_query', 'text', 'twitter_id', 'username', 'polarit
y']
Numerical columns :  ['created_at', 'followers', 'friends']
created_at: timestamp
file_name: string
followers: int
friends: int
group_name: string
location: string
retweet_count: string
screenname: string
search_query: string
text: string
twitter_id: string
username: string
polarity: string
partition_0: string
partition_1: string
==================================================

Description of numerical variables


+-------+------------------+------------------+
|summary|         followers|           friends|
+-------+------------------+------------------+
|  count|            810383|            810383|
|   mean|10111.410045620405| 2565.079293371159|
| stddev| 143888.3708965397|14419.059802955742|
|    min|                 0|                 0|
|    max|          14443441|           1168982|
+-------+------------------+------------------+

==================================================

Description of categorical variables

[Stage 22:=====================================>                   (5 + 3) /
8]
```

```
+-------+---------+----------+-----------------+------------------+----------
---------+------------+--------------------+--------------------+-----------
--------+--------------------+--------------------+--------------------+
|summary|file_name|group_name|         location|     retweet_count|         s
creenname|search_query|                text|          twitter_id|
username|            polarity|         partition_0|         partition_1|
+-------+---------+----------+-----------------+------------------+----------
---------+------------+--------------------+--------------------+-----------
--------+--------------------+--------------------+--------------------+
|  count|   810383|    810383|           810383|            810383|
810383|      810383|              810383|              810383|              8
10383|              810383|              810383|              810383|
|   mean|     NULL|      NULL|         Infinity|107.63962436447999| 1.4055123
4421875E8|        NULL|                NULL|1.298188644110661...|
NaN|4.216454033222882...|2.282438386375511...|3.027867686873197...|
| stddev|     NULL|      NULL|              NaN|1501.5457522837353|5.69680519
4152116E8|        NULL|                NULL|1.439952594142635...|
NaN|7.391041965101361...|4.945305994706281...|5.495645479279455...|
|    min|      AMD|       AMD|\t.əʌɐ uoowʎəuoʇ |               MD"|
0.0|        #AMD| -- DIFFERENT WAL...|   @xamarinhq  #t...|  #Cybertruck.
#...|  #Netflix and… h...|      Wi-Fi adapater|   USB (for RAM b...|
|    max|  Youtube|   Youtube|      🏐 - she/her|            9993.0|      zzz
zzebra1224|      cutety|🫖 @otter_ai punc...|                …"|
🪔💫|  👗🚀Flywheel🔋🔥🪠|             🦢 Mike|          🦎👴🍳☀|
+-------+---------+----------+-----------------+------------------+----------
---------+------------+--------------------+--------------------+-----------
--------+--------------------+--------------------+--------------------+
```

In [12]:

```
from pyspark.sql.functions import col, udf
from pyspark.sql.types import StringType, FloatType

# Convert 'polarity' column to float
df_spark = df_spark.withColumn('polarity', col('polarity').cast(FloatType(

# Define UDF to convert polarity to sentiment
def polarity_to_sentiment_udf(polarity):
    if polarity is not None:
        return 'positive' if polarity >= 0.5 else 'negative'
    else:
        return None

# Create a UDF
polarity_to_sentiment = udf(polarity_to_sentiment_udf, StringType())

# Apply the UDF to create a new column 'sentiment'
df_spark = df_spark.withColumn('sentiment', polarity_to_sentiment(df_spark

# Drop rows with 'undefined' sentiment
df_spark = df_spark.dropna(subset=['sentiment'])

# Display the updated DataFrame
df_spark.select('polarity', 'sentiment').show(5)

# Get the count of each sentiment
sentiment_counts = df_spark.groupBy('sentiment').count()
sentiment_counts.show()

```

```
+--------+---------+
|polarity|sentiment|
+--------+---------+
| -0.3102| negative|
|     0.0| negative|
|     0.0| negative|
| -0.3612| negative|
| -0.3612| negative|
+--------+---------+
only showing top 5 rows


[Stage 26:=======>                                                 (1 + 7) /
8]

+---------+------+
|sentiment| count|
+---------+------+
| positive|197317|
| negative|601050|
+---------+------+
```

In [13]:
```
1 df_spark.show(5)
```

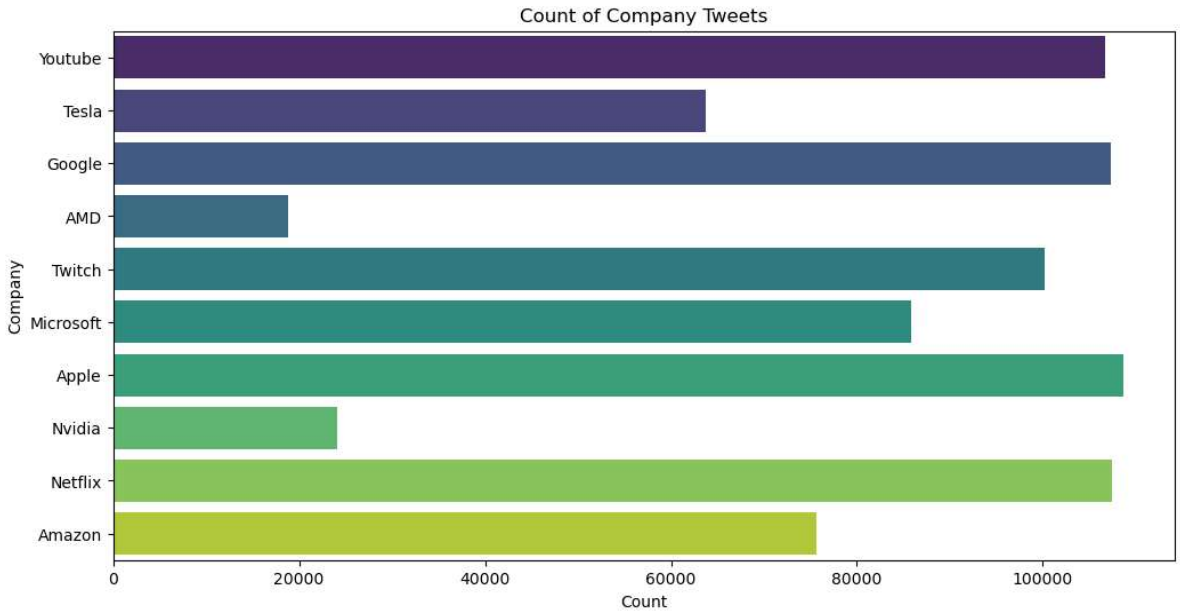
```
+------------------+---------+---------+-------+----------+--------------+--
----------+------------+-----------+--------------------+----------------
---+------------------+--------+----------+----------+---------+
|         created_at|file_name|followers|friends|group_name|      location|re
tweet_count|   screenname|search_query|                text|          twitter
_id|          username|polarity|partition_0|partition_1|sentiment|
+------------------+---------+---------+-------+----------+--------------+--
----------+------------+-----------+--------------------+----------------
---+------------------+--------+----------+----------+---------+
|2020-07-12 09:24:26|      AMD|       25|    114|       AMD|United Kingdom|
0.0|moffphcgaming|        #AMD|Been on holiday s...|1.282244417466884...|⛸Mo
ffPHC Gaming⛸| -0.3102| Technology|        AMD| negative|
|2020-07-12 09:09:36|      AMD|      159|   1144|       AMD|  digitalverse|
4.0|   ironparr0t|        #AMD|RT @NinjaParanoid...|1.282240682023719...|
ironparrot|     0.0| Technology|        AMD| negative|
|2020-07-12 08:31:24|      AMD|        9|    188|       AMD|   HAHA no....|
0.0|  XApochrypha|        #AMD|Recently purchase...|1.282231068649652...|   X
enosApochrypha|     0.0| Technology|        AMD| negative|
|2020-07-12 08:16:45|      AMD|     1719|      1|       AMD|  digitalocean|
1.0|  LinuxDreams|        #AMD|"RT @LinuxReviews...|1.282227383932772...|
LinuxDreams| -0.3612| Technology|        AMD| negative|
|2020-07-12 08:11:41|      AMD|       69|    135|       AMD|     Amsterdam|
1.0| LinuxReviews|        #AMD|"#Linux architect...|1.282226107027271...|
LinuxReviews| -0.3612| Technology|        AMD| negative|
+------------------+---------+---------+-------+----------+--------------+--
----------+------------+-----------+--------------------+----------------
---+------------------+--------+----------+----------+---------+
only showing top 5 rows
```

Companies Distribution in the Tweets

In [14]:
```
# Check unique values and counts in 'group_name' column
df_spark.groupBy('group_name').count().show()
```

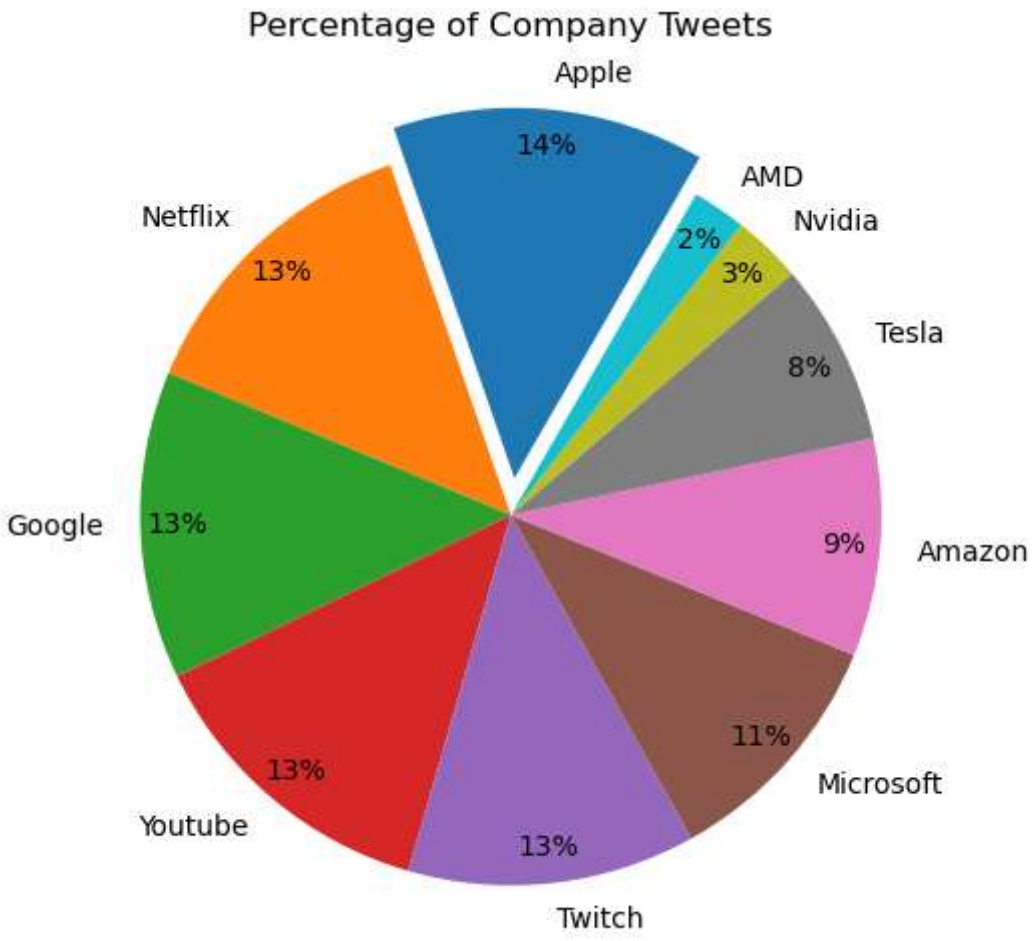
```
[Stage 30:>                                                         (0 + 8) /
8]

+----------+------+
|group_name| count|
+----------+------+
|   Youtube|106665|
|     Tesla| 63788|
|    Google|107307|
|       AMD| 18766|
|    Twitch|100227|
| Microsoft| 85793|
|     Apple|108710|
|    Nvidia| 24057|
|   Netflix|107382|
|    Amazon| 75672|
+----------+------+
```

In [15]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Plot the count of company tweets
group_name_counts = df_spark.groupBy('group_name').count().toPandas()

plt.figure(figsize=(12, 6))
sns.barplot(x='count', y='group_name', data=group_name_counts, palette='vi
plt.title('Count of Company Tweets')
plt.xlabel('Count')
plt.ylabel('Company')
plt.show()
```



In [16]:

```
from pyspark.sql import functions as F

# Calculate the percentage of company tweets
group_name_counts = df_spark.groupBy('group_name').count()
group_name_counts = group_name_counts.orderBy(F.desc('count')).limit(10)

# Plot the percentage of company tweets
explode = [0.1 if i == 0 else 0 for i in range(10)]
labels = group_name_counts.select('group_name').rdd.flatMap(lambda x: x).c
sizes = group_name_counts.select('count').rdd.flatMap(lambda x: x).collect

plt.figure(figsize=(12, 6))
plt.pie(sizes, explode=explode, labels=labels, autopct='%1.0f%%', startang
plt.title('Percentage of Company Tweets')
plt.show()

```

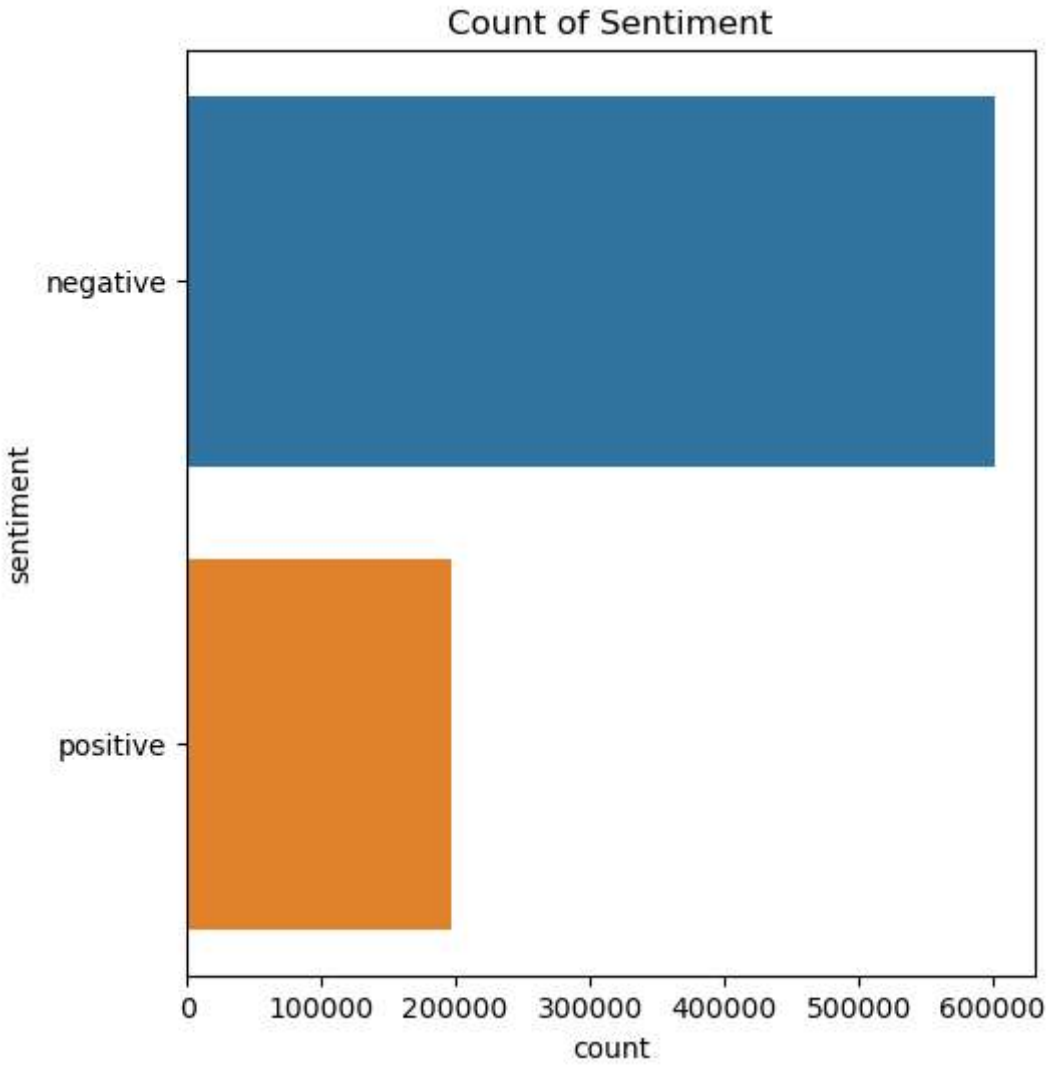


Sentiment Distribution

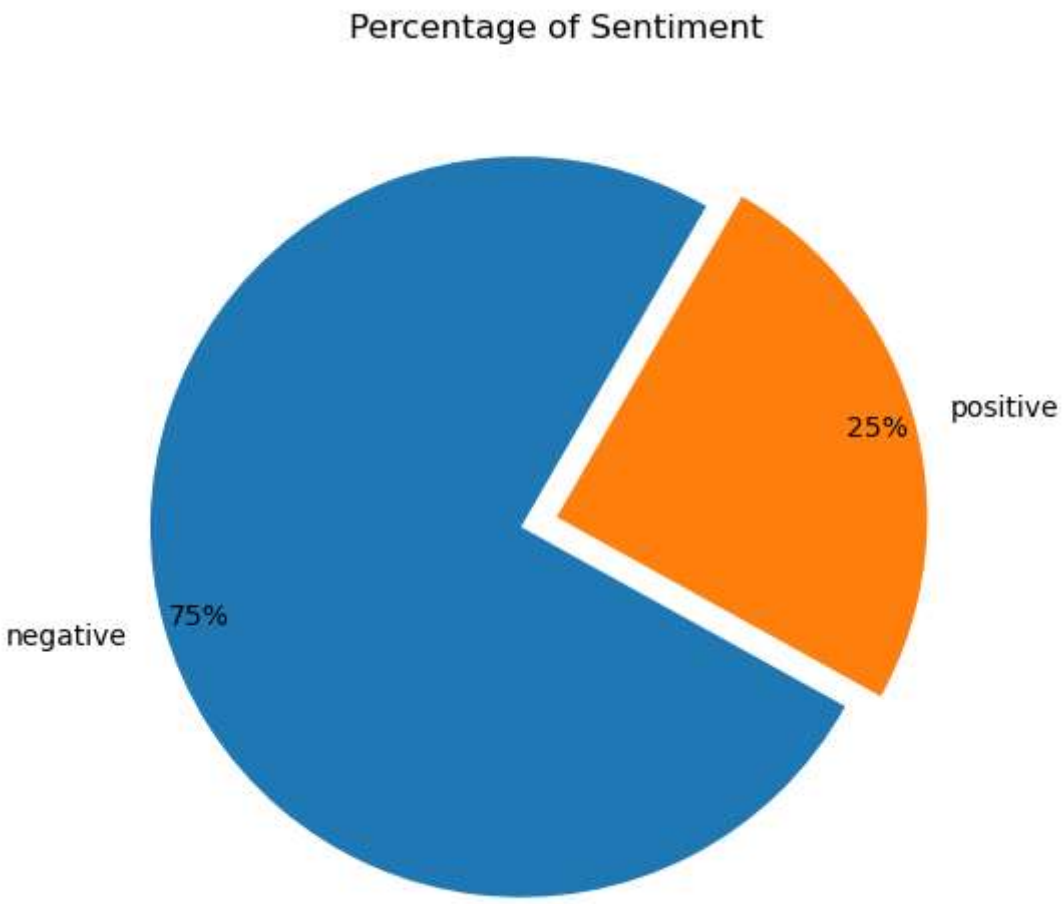
In [17]:
```
from pyspark.sql import functions as F

# Calculate the sentiment distribution
sentiment_counts = df_spark.groupBy('sentiment').count()
sentiment_counts = sentiment_counts.orderBy(F.desc('count'))

# Display the sentiment distribution
sentiment_counts.show()
```

```
[Stage 42:==============>                                          (2 + 6) /
8]

+---------+------+
|sentiment| count|
+---------+------+
| negative|601050|
| positive|197317|
+---------+------+
```

In [18]:
```
# Plot the count of sentiment
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 2)
sns.countplot(y='sentiment', data=df_spark.toPandas())
plt.title('Count of Sentiment')
plt.show()
```



In [19]:
```
# Plot the percentage of sentiment
explode = [0.1 if i == 0 else 0 for i in range(sentiment_counts.count())]
labels = sentiment_counts.select('sentiment').rdd.flatMap(lambda x: x).col
sizes = sentiment_counts.select('count').rdd.flatMap(lambda x: x).collect(

plt.figure(figsize=(12, 6))
plt.pie(sizes, explode=explode, labels=labels, autopct='%1.0f%%', startang
plt.title('Percentage of Sentiment')
plt.show()
```

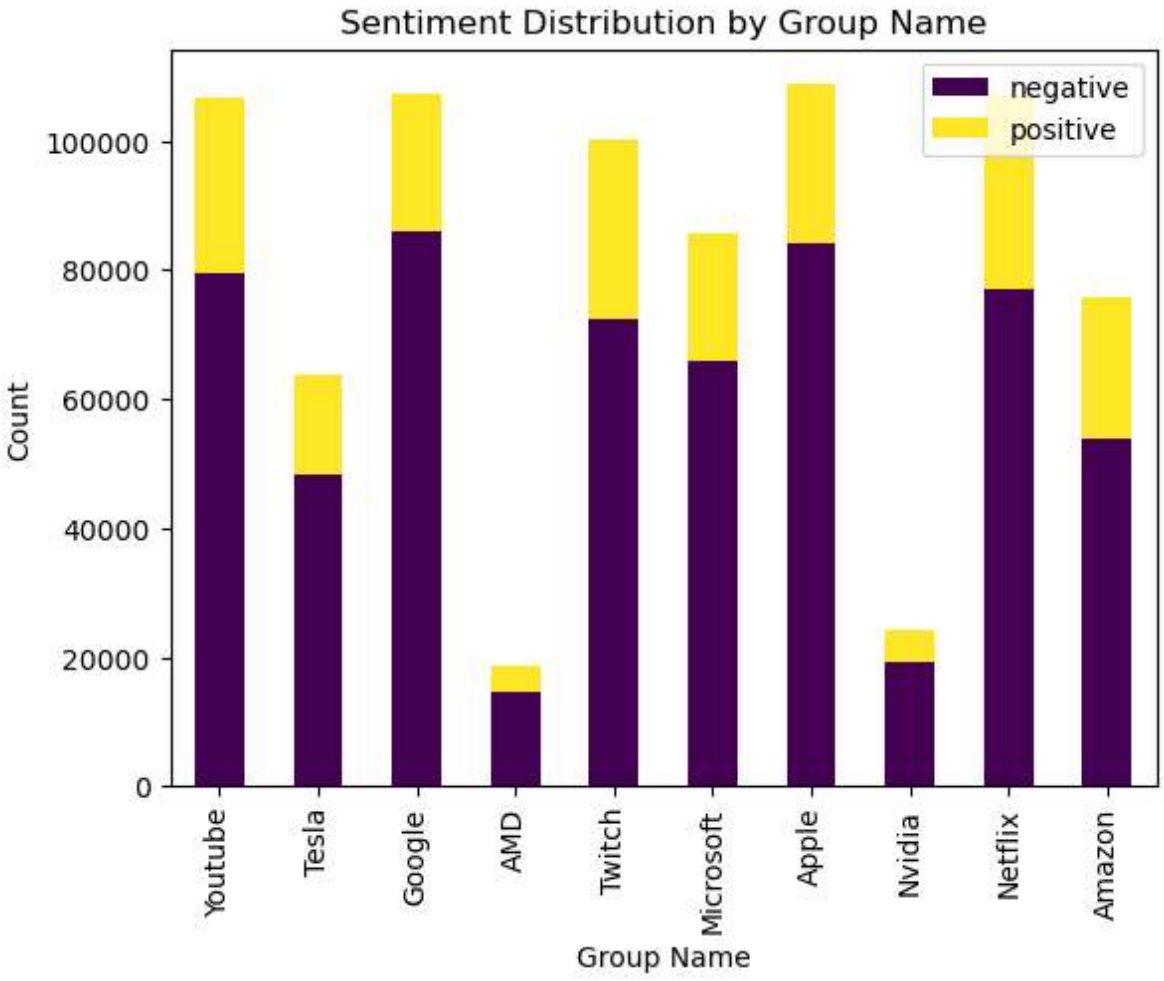


In [29]:
```
# Calculate the sentiment distribution
sentiment_counts = df_spark.groupBy('group_name', 'sentiment').count()

# Pivot the data to have 'positive' and 'negative' as columns
sentiment_pivot = sentiment_counts.groupBy('group_name').pivot('sentiment'

# Display the result
sentiment_pivot.show()
```

```
[Stage 188:============================>                            (4 + 4) /
8]

+----------+--------+--------+
|group_name|negative|positive|
+----------+--------+--------+
|   Youtube|   79586|   27079|
|     Tesla|   48169|   15619|
|    Google|   85978|   21329|
|       AMD|   14549|    4217|
|    Twitch|   72429|   27798|
| Microsoft|   65961|   19832|
|     Apple|   84042|   24668|
|    Nvidia|   19333|    4724|
|   Netflix|   77116|   30266|
|    Amazon|   53887|   21785|
+----------+--------+--------+
```

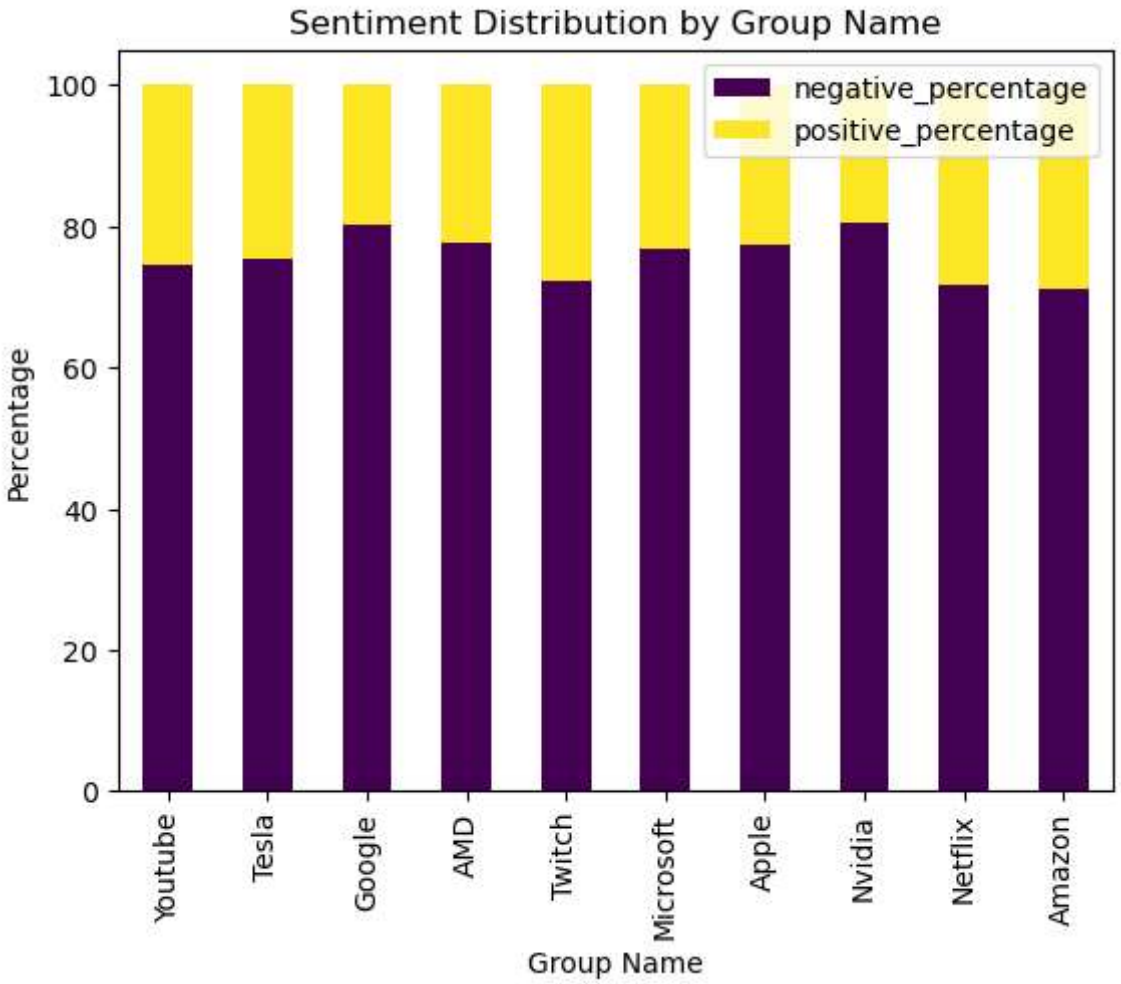
In [30]:

```
import matplotlib.pyplot as plt
import pandas as pd

# Convert the DataFrame to Pandas for easier plotting
sentiment_pandas = sentiment_pivot.toPandas()

# Plot the data
sentiment_pandas.plot(kind='bar', x='group_name', stacked=True, colormap=

# Set plot labels and title
plt.xlabel('Group Name')
plt.ylabel('Count')
plt.title('Sentiment Distribution by Group Name')

# Display the plot
plt.show()
```



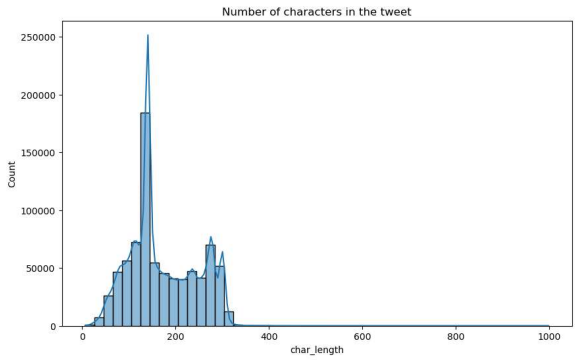
In [40]:

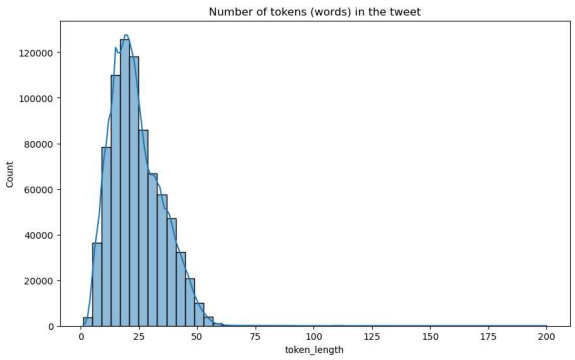
```
import matplotlib.pyplot as plt
import pandas as pd
from pyspark.sql.functions import col

# Assuming you have a DataFrame named sentiment_counts with columns 'group
sentiment_counts = spark.createDataFrame([
    ("Youtube", 79586, 27079),
    ("Tesla", 48169, 15619),
    ("Google", 85978, 21329),
    ("AMD", 14549, 4217),
    ("Twitch", 72429, 27798),
    ("Microsoft", 65961, 19832),
    ("Apple", 84042, 24668),
    ("Nvidia", 19333, 4724),
    ("Netflix", 77116, 30266),
    ("Amazon", 53887, 21785)
], ["group_name", "negative", "positive"])

# Calculate percentage columns
sentiment_counts = sentiment_counts.withColumn("total", col("negative") +
sentiment_percentages = sentiment_counts.withColumn("negative_percentage",
sentiment_percentages = sentiment_percentages.withColumn("positive_percent

# Pivot the table for better visualization
sentiment_pivot = sentiment_percentages.select("group_name", "negative_per

# Plot the data
sentiment_pivot.plot(kind='bar', x='group_name', y=['negative_percentage',

# Set plot labels and title
plt.xlabel('Group Name')
plt.ylabel('Percentage')
plt.title('Sentiment Distribution by Group Name')

# Display the plot
plt.show()

```


Sentiment Distribution by Group Name
negative_percentage
positive_percentage
Percentage
0
20
40
60
80
100
Youtube
Tesla
Google
AMD
Twitch
Microsoft
Apple
Nvidia
Netflix
Amazon
Group Name

In [20]:

```
from pyspark.sql.functions import length, size, split
from pyspark.sql.types import IntegerType

# Calculate the character length and token length
df_spark = df_spark.withColumn('char_length', length('text'))
df_spark = df_spark.withColumn('token_length', size(split('text', ' ')).ca

# Plot the distributions
char_length_dist = df_spark.select('char_length').toPandas()
token_length_dist = df_spark.select('token_length').toPandas()

fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(22, 6))
sns.histplot(char_length_dist['char_length'], ax=ax1, bins=50, kde=True)
sns.histplot(token_length_dist['token_length'], ax=ax2, bins=50, kde=True)

ax1.set_title('Number of characters in the tweet')
ax2.set_title('Number of tokens (words) in the tweet')
plt.show()

```

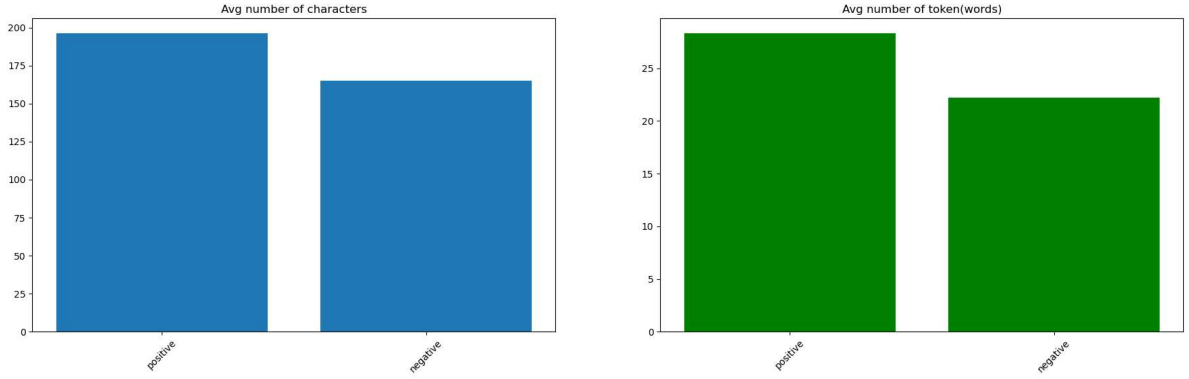




In [21]:

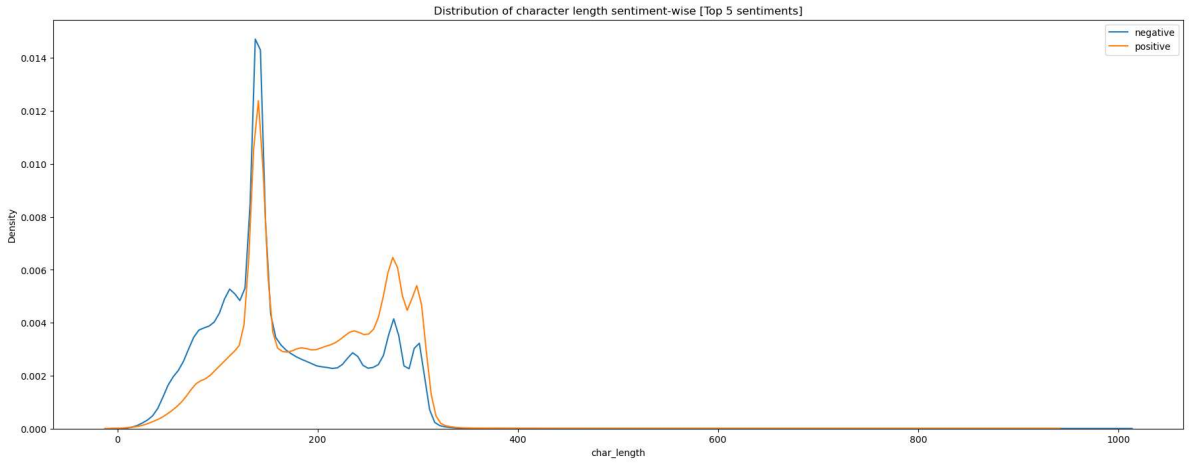
```
from pyspark.sql.functions import mean
from pyspark.sql.types import IntegerType

# Calculate average character length and token length
avg_df_spark = df_spark.groupBy('sentiment').agg(mean('char_length').alias

# Plot the averages
fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(22, 6))
avg_df_pandas = avg_df_spark.toPandas()
ax1.bar(avg_df_pandas['sentiment'], avg_df_pandas['avg_char_length'])
ax2.bar(avg_df_pandas['sentiment'], avg_df_pandas['avg_token_length'], col
ax1.set_title('Avg number of characters')
ax2.set_title('Avg number of token(words)')
ax1.set_xticklabels(avg_df_pandas['sentiment'], rotation=45)
ax2.set_xticklabels(avg_df_pandas['sentiment'], rotation=45)
plt.show()
```

```
/var/folders/08/bd2b2ypn66q0mdj491dlry4c0000gn/T/ipykernel_16102/1437006977.p
y:14: UserWarning: FixedFormatter should only be used together with FixedLoca
tor
  ax1.set_xticklabels(avg_df_pandas['sentiment'], rotation=45)
/var/folders/08/bd2b2ypn66q0mdj491dlry4c0000gn/T/ipykernel_16102/1437006977.p
y:15: UserWarning: FixedFormatter should only be used together with FixedLoca
tor
  ax2.set_xticklabels(avg_df_pandas['sentiment'], rotation=45)
```



In [22]:

```
from pyspark.sql.functions import col
from pyspark.sql.types import IntegerType

# Top 5 sentiments
top_sentiments = df_spark.groupBy('sentiment').count().orderBy('count', as

# Create a list of top 5 sentiments
top_sentiments_list = top_sentiments.select('sentiment').rdd.flatMap(lambd

# Plot the distribution of character length sentiment-wise for top 5 senti
fig, ax = plt.subplots(figsize=(22, 8))
for sentiment in top_sentiments_list:
    sentiment_df = df_spark.filter(col('sentiment') == sentiment).select(
    sns.kdeplot(sentiment_df['char_length'], ax=ax, label=sentiment)

ax.legend()
ax.set_title("Distribution of character length sentiment-wise [Top 5 senti
plt.show()
```



Text Cleaning and Feature Engineering:

In [23]:
```
from pyspark.sql import functions as F

# Assuming 'created_at' is a column that represents the order
last_n_rows = df_spark.orderBy(F.desc('created_at')).limit(5)

# Display the last N rows
last_n_rows.show()
```

```
[Stage 78:>                                                         (0 + 8) /
8]

+------------------+---------+---------+-------+----------+----------------
---+-------------+-------------+-----------+--------------------+----------
---------+--------------------+-------------+--------------------+----------
+---------+----------+------------+
|         created_at|file_name|followers|friends|group_name|            locat
ion|retweet_count|   screenname|search_query|                text|          t
witter_id|            username|     polarity|         partition_0|partition_1
|sentiment|char_length|token_length|
+------------------+---------+---------+-------+----------+----------------
---+-------------+-------------+-----------+--------------------+----------
---------+--------------------+-------------+--------------------+----------
+---------+----------+------------+
|2020-10-12 23:59:54|   Google|      739|    607|    Google|lilianne. she/he
r...|          1.0|  CllNDERFALL|     #Google|@bvbbline #GOOGLE...|1.31580442
0437835...|           WITCHLING|       0.3612|          Technology|     Googl
e| negative|         75|          12|
|2020-10-12 23:59:37|    Tesla|     3266|   4399|     Tesla|       Las Vegas,
NV|          2.0|  SparksRadio|      #Tesla|"Listen to ""Ass ...| Pike's Car
Got B...| Surf Ninjas Stor...|1.31580439E18|Sparks ⚡☠️⚡ dai...|    -0.877
9| positive|         33|           7|
|2020-10-12 23:59:11|    Tesla|      297|    319|     Tesla|     New Jersey,
USA|          0.0|goldentouch73|      #Tesla|@elonmusk @Teslar...|1.315804236
496744...|             Thinker|       0.0516|          Technology|      Tesla
| negative|        267|          52|
|2020-10-12 23:59:05|Microsoft|     2040|     96| Microsoft|
UK|          1.0|  TechyGeeks1|  #Microsoft|Get SID from User...|1.3158042110
03695...|A.J. Armstrong @ ...|       0.4019|          Technology|  Microsoft|
negative|        268|          43|
|2020-10-12 23:58:52|  Netflix|        2|      8|   Netflix| Playing Video Ga
mes|          0.0|   wierdoboy2|    #Netflix|I love how @netfl...|1.315804159
292182...|           wierdoboy|       0.6369|          Technology|    Netflix
| positive|        138|          18|
+------------------+---------+---------+-------+----------+----------------
---+-------------+-------------+-----------+--------------------+----------
---------+--------------------+-------------+--------------------+----------
+---------+----------+------------+
```

In [24]:
```
1 pip install neattext
```

```
Requirement already satisfied: neattext in /Users/saikumargandham/opt/anacond
a3/lib/python3.9/site-packages (0.1.3)
Note: you may need to restart the kernel to use updated packages.
```

In [25]:

```
from pyspark.ml.feature import RegexTokenizer, StopWordsRemover, CountVect
from pyspark.sql.functions import udf
from pyspark.sql.types import StringType
import neattext.functions as nfx

# Define UDFs for text cleaning using neattext
remove_userhandles_udf = udf(lambda x: nfx.remove_userhandles(str(x)) if x
remove_stopwords_udf = udf(lambda x: nfx.remove_stopwords(str(x)) if x is
remove_emails_udf = udf(lambda x: nfx.remove_emails(str(x)) if x is not No
remove_emojis_udf = udf(lambda x: nfx.remove_emojis(nfx.remove_special_cha
remove_special_characters_udf = udf(lambda x: nfx.remove_special_character
remove_urls_udf = udf(lambda x: nfx.remove_urls(str(x)) if x is not None e
lowercase_udf = udf(lambda x: str(x).lower() if x is not None else '', Str

# Apply text cleaning and feature engineering
df_spark = df_spark.withColumn('Clean_Text', remove_userhandles_udf('text'
df_spark = df_spark.withColumn('Clean_Text', remove_stopwords_udf('Clean_T
df_spark = df_spark.withColumn('Clean_Text', remove_emails_udf('Clean_Text
df_spark = df_spark.withColumn('Clean_Text', remove_emojis_udf('Clean_Text
df_spark = df_spark.withColumn('Clean_Text', remove_special_characters_udf
df_spark = df_spark.withColumn('Clean_Text', remove_urls_udf('Clean_Text')
df_spark = df_spark.withColumn('Clean_Text', lowercase_udf('Clean_Text'))

# Display the cleaned text
df_spark.select('text', 'Clean_Text').show(5, truncate=False)
```

```
+--------------------------------------------------------------------------------
--------------------------------------------------------------------+------
--------------------------------------------------------------------------------
------------------------------+
|text
|Clean_Text
|
+--------------------------------------------------------------------------------
--------------------------------------------------------------------+------
--------------------------------------------------------------------------------
------------------------------+
|Been on holiday so back now. Gonna try get some more sven coop horror footag
e and get a new video up also working o… https://t.co/X1qV2B0cGr (https://t.c
o/X1qV2B0cGr)      |holiday now gonna try sven coop horror footage new video
working o httpstcox1qv2b0cgr                                |
|RT @NinjaParanoid: #AMD #Ryzen #3900x #rtx2070 #nvidia 64 gigs, 144hz DP. Th
e #Hacker House is set! https://t.co/BclpeT6z7s (https://t.co/BclpeT6z7s)
|rt amd ryzen 3900x rtx2070 nvidia 64 gigs 144hz dp hacker house set httpstco
bclpet6z7s                              |
|Recently purchased everything for my first personal gaming pc build with an
#AMD Ryzen 7 3700X and a #EVGA GTX 2070… https://t.co/CLCWdnKjdl (https://t.c
o/CLCWdnKjdl)      |recently purchased personal gaming pc build amd ryzen 7 3
700x evga gtx 2070 httpstcoclcwdnkjdl                       |
|"RT @LinuxReviews: #Linux architect Linus Torvalds: AVX512 Is ""A Hot Mess""
""I hope AVX512 dies a painful death"". He elaborated that  it is a…"|rt linu
x architect linus torvalds avx512 a hot mess i hope avx512 dies painful death
elaborated a                |
|"#Linux architect Linus Torvalds: AVX512 Is ""A Hot Mess"" ""I hope AVX512 d
ies a painful death"". He elaborated that  i… https://t.co/K7F1Nw395g"|linux
(https://t.co/K7F1Nw395g"|linux) architect linus torvalds avx512 a hot mess i
hope avx512 dies painful death elaborated i httpstcok7f1nw395g|
+--------------------------------------------------------------------------------
--------------------------------------------------------------------+------
--------------------------------------------------------------------------------
------------------------------+
only showing top 5 rows
```

neattext library might not cover all cases of emojis

In [26]:

```
# Show the first 5 rows of the DataFrame
# Drop intermediate columns
df_spark = df_spark.drop("words", "filtered_words", "rawFeatures", "featur
df_spark.show(5, truncate=False)
```

```
+------------------+---------+---------+-------+----------+-------------+--
----------+------------+-----------+----------------------------------------
----------------------------------------------------------------------------
-------------------------------+---------------------+-----------------+---
-----+----------+----------+---------+----------+----------+-----------
----------------------------------------------------------------------------
-----------------------+
|created_at         |file_name|followers|friends|group_name|location      |re
tweet_count|screenname   |search_query|text
|twitter_id           |username          |polarity|partition_0|partition_1|s
entiment|char_length|token_length|Clean_Text
|
+------------------+---------+---------+-------+----------+-------------+--
----------+------------+-----------+----------------------------------------
----------------------------------------------------------------------------
-------------------------------+---------------------+-----------------+---
-----+----------+----------+---------+----------+----------+-----------
----------------------------------------------------------------------------
-----------------------+
|2020-07-12 09:24:26|AMD      |25       |114    |AMD       |United Kingdom|0.
0          |moffphcgaming|#AMD        |Been on holiday so back now. Gonna try
get some more sven coop horror footage and get a new video up also working o…
https://t.co/X1qV2B0cGr (https://t.co/X1qV2B0cGr)      |1.282244417466884e+18
|⛪MoffPHC Gaming⛪|-0.3102 |Technology |AMD        |negative |140        |25
|holiday now gonna try sven coop horror footage new video working o httpstcox
1qv2b0cgr                                |
|2020-07-12 09:09:36|AMD      |159      |1144   |AMD       |digitalverse  |4.
0          |ironparr0t   |#AMD        |RT @NinjaParanoid: #AMD #Ryzen #3900x
#rtx2070 #nvidia 64 gigs, 144hz DP. The #Hacker House is set! https://t.co/Bc
lpeT6z7s (https://t.co/BclpeT6z7s)                        |1.282240682023719e+
18 |ironparrot        |0.0     |Technology |AMD        |negative |123
|17          |rt amd ryzen 3900x rtx2070 nvidia 64 gigs 144hz dp hacker house
set httpstcobclpet6z7s                              |
|2020-07-12 08:31:24|AMD      |9        |188    |AMD       |HAHA no....   |0.
0          |XApochrypha  |#AMD        |Recently purchased everything for my f
irst personal gaming pc build with an #AMD Ryzen 7 3700X and a #EVGA GTX 207
0… https://t.co/CLCWdnKjdl (https://t.co/CLCWdnKjdl)      |1.2822310686496522
e+18|XenosApochrypha   |0.0     |Technology |AMD        |negative |140
|22          |recently purchased personal gaming pc build amd ryzen 7 3700x e
vga gtx 2070 httpstcoclcwdnkjdl                     |
|2020-07-12 08:16:45|AMD      |1719     |1      |AMD       |digitalocean  |1.
0          |LinuxDreams  |#AMD        |"RT @LinuxReviews: #Linux architect Li
nus Torvalds: AVX512 Is ""A Hot Mess"" ""I hope AVX512 dies a painful deat
h"". He elaborated that  it is a…"|1.2822273839327724e+18|LinuxDreams       |
-0.3612 |Technology |AMD        |negative |146        |25          |rt linux
architect linus torvalds avx512 a hot mess i hope avx512 dies painful death e
laborated a                  |
|2020-07-12 08:11:41|AMD      |69       |135    |AMD       |Amsterdam     |1.
0          |LinuxReviews |#AMD        |"#Linux architect Linus Torvalds: AVX5
12 Is ""A Hot Mess"" ""I hope AVX512 dies a painful death"". He elaborated th
at  i… https://t.co/K7F1Nw395g"|1.2822261070272717e+18|LinuxReviews (https://
t.co/K7F1Nw395g"|1.2822261070272717e+18|LinuxReviews)      |-0.3612 |Technolo
gy |AMD        |negative |146        |22          |linux architect linus torv
alds avx512 a hot mess i hope avx512 dies painful death elaborated i httpstco
k7f1nw395g|
+------------------+---------+---------+-------+----------+-------------+--
----------+------------+-----------+----------------------------------------
```

```
--------------------------------------------------------------------------------
--------------------------------+---------------------+------------------+---
-----+-----------+-----------+---------+-----------+------------+------------
--------------------------------------------------------------------------------
------------------------+
only showing top 5 rows
```

***Classification Models*:**

In [27]:

```
from pyspark.ml.feature import Tokenizer, StopWordsRemover, HashingTF, IDF
from pyspark.ml import Pipeline
from pyspark.ml.classification import RandomForestClassifier, DecisionTree
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

# Tokenize the text
tokenizer = Tokenizer(inputCol="Clean_Text", outputCol="words")
df_spark = tokenizer.transform(df_spark)

# Remove stop words
remover = StopWordsRemover(inputCol="words", outputCol="filtered_words")
df_spark = remover.transform(df_spark)

# Apply TF-IDF
hashingTF = HashingTF(inputCol="filtered_words", outputCol="rawFeatures",
idf = IDF(inputCol="rawFeatures", outputCol="features")
df_spark = hashingTF.transform(df_spark)
df_spark = idf.fit(df_spark).transform(df_spark)

# Convert 'sentiment' column to numeric
label_indexer = StringIndexer(inputCol="sentiment", outputCol="label")
df_spark = label_indexer.fit(df_spark).transform(df_spark)

# Split the data into training and testing sets
(training_data, testing_data) = df_spark.randomSplit([0.8, 0.2], seed=42)

# Define the classifiers
rf_classifier = RandomForestClassifier(labelCol="label", featuresCol="feat
dt_classifier = DecisionTreeClassifier(labelCol="label", featuresCol="feat
lr_classifier = LogisticRegression(labelCol="label", featuresCol="features

# Train the models
rf_model = rf_classifier.fit(training_data)
dt_model = dt_classifier.fit(training_data)
lr_model = lr_classifier.fit(training_data)

# Make predictions
rf_predictions = rf_model.transform(testing_data)
dt_predictions = dt_model.transform(testing_data)
lr_predictions = lr_model.transform(testing_data)

# Evaluate the models
evaluator = MulticlassClassificationEvaluator(labelCol="label", predictior

rf_accuracy = evaluator.evaluate(rf_predictions)
dt_accuracy = evaluator.evaluate(dt_predictions)
lr_accuracy = evaluator.evaluate(lr_predictions)

print("Random Forest Accuracy: {:.2%}".format(rf_accuracy))
print("Decision Tree Accuracy: {:.2%}".format(dt_accuracy))
print("Logistic Regression Accuracy: {:.2%}".format(lr_accuracy))

```

```
23/12/12 13:02:23 WARN MemoryStore: Not enough space to cache rdd_388_1 in me
mory! (computed 47.4 MiB so far)
23/12/12 13:02:23 WARN BlockManager: Persisting block rdd_388_1 to disk inste
ad.
23/12/12 13:02:23 WARN MemoryStore: Not enough space to cache rdd_388_0 in me
mory! (computed 73.6 MiB so far)
23/12/12 13:02:23 WARN BlockManager: Persisting block rdd_388_0 to disk inste
ad.
23/12/12 13:02:23 WARN MemoryStore: Not enough space to cache rdd_388_2 in me
mory! (computed 9.1 MiB so far)
23/12/12 13:02:23 WARN BlockManager: Persisting block rdd_388_2 to disk inste
ad.
23/12/12 13:02:24 WARN MemoryStore: Not enough space to cache rdd_388_6 in me
mory! (computed 9.1 MiB so far)
23/12/12 13:02:24 WARN BlockManager: Persisting block rdd_388_6 to disk inste
ad.
23/12/12 13:02:24 WARN MemoryStore: Not enough space to cache rdd_388_4 in me
mory! (computed 31.4 MiB so far)
23/12/12 13:02:24 WARN BlockManager: Persisting block rdd_388_4 to disk inste
ad.
23/12/12 13:02:24 WARN MemoryStore: Not enough space to cache rdd_388_3 in me
mory! (computed 112.4 MiB so far)
23/12/12 13:02:24 WARN BlockManager: Persisting block rdd_388_3 to disk inste
ad.
23/12/12 13:02:24 WARN MemoryStore: Not enough space to cache rdd_388_1 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:24 WARN MemoryStore: Not enough space to cache rdd_388_0 in me
mory! (computed 73.6 MiB so far)
23/12/12 13:02:25 WARN MemoryStore: Not enough space to cache rdd_388_2 in me
mory! (computed 47.4 MiB so far)
23/12/12 13:02:25 WARN MemoryStore: Not enough space to cache rdd_388_3 in me
mory! (computed 73.6 MiB so far)
23/12/12 13:02:25 WARN MemoryStore: Not enough space to cache rdd_388_4 in me
mory! (computed 73.6 MiB so far)
23/12/12 13:02:25 WARN MemoryStore: Not enough space to cache rdd_388_6 in me
mory! (computed 112.4 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_0 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_2 in me
mory! (computed 13.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_6 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_1 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_4 in me
mory! (computed 47.4 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_3 in me
mory! (computed 112.4 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_2 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_3 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_4 in me
mory! (computed 13.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_1 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_6 in me
```

```
mory! (computed 20.9 MiB so far)
23/12/12 13:02:26 WARN MemoryStore: Not enough space to cache rdd_388_0 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:27 WARN MemoryStore: Not enough space to cache rdd_388_6 in me
mory! (computed 13.9 MiB so far)
23/12/12 13:02:27 WARN MemoryStore: Not enough space to cache rdd_388_1 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:27 WARN MemoryStore: Not enough space to cache rdd_388_3 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:27 WARN MemoryStore: Not enough space to cache rdd_388_0 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:27 WARN MemoryStore: Not enough space to cache rdd_388_2 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:27 WARN MemoryStore: Not enough space to cache rdd_388_4 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:28 WARN MemoryStore: Not enough space to cache rdd_388_1 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:28 WARN MemoryStore: Not enough space to cache rdd_388_6 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:02:28 WARN MemoryStore: Not enough space to cache rdd_388_3 in me
mory! (computed 13.9 MiB so far)
23/12/12 13:02:28 WARN MemoryStore: Not enough space to cache rdd_388_0 in me
mory! (computed 31.4 MiB so far)
23/12/12 13:02:28 WARN MemoryStore: Not enough space to cache rdd_388_4 in me
mory! (computed 13.9 MiB so far)
23/12/12 13:02:28 WARN MemoryStore: Not enough space to cache rdd_388_2 in me
mory! (computed 20.9 MiB so far)
23/12/12 13:03:29 WARN MemoryStore: Not enough space to cache rdd_464_2 in me
mory! (computed 1628.6 KiB so far)
23/12/12 13:03:29 WARN BlockManager: Persisting block rdd_464_2 to disk inste
ad.
23/12/12 13:03:29 WARN MemoryStore: Not enough space to cache rdd_464_3 in me
mory! (computed 5.6 MiB so far)
23/12/12 13:03:29 WARN BlockManager: Persisting block rdd_464_3 to disk inste
ad.
23/12/12 13:03:29 WARN MemoryStore: Not enough space to cache rdd_464_1 in me
mory! (computed 20.4 MiB so far)
23/12/12 13:03:29 WARN BlockManager: Persisting block rdd_464_1 to disk inste
ad.
23/12/12 13:03:29 WARN MemoryStore: Not enough space to cache rdd_464_0 in me
mory! (computed 109.4 MiB so far)
23/12/12 13:03:29 WARN BlockManager: Persisting block rdd_464_0 to disk inste
ad.
23/12/12 13:03:31 WARN MemoryStore: Not enough space to cache rdd_464_4 in me
mory! (computed 71.7 MiB so far)
23/12/12 13:03:31 WARN BlockManager: Persisting block rdd_464_4 to disk inste
ad.
23/12/12 13:03:31 WARN MemoryStore: Not enough space to cache rdd_464_5 in me
mory! (computed 71.7 MiB so far)
23/12/12 13:03:31 WARN BlockManager: Persisting block rdd_464_5 to disk inste
ad.
23/12/12 13:03:31 WARN MemoryStore: Not enough space to cache rdd_464_0 in me
mory! (computed 30.7 MiB so far)
23/12/12 13:03:31 WARN MemoryStore: Not enough space to cache rdd_464_6 in me
mory! (computed 71.7 MiB so far)
23/12/12 13:03:31 WARN BlockManager: Persisting block rdd_464_6 to disk inste
ad.
```

```
23/12/12 13:03:31 WARN MemoryStore: Not enough space to cache rdd_464_2 in me
mory! (computed 110.6 MiB so far)
23/12/12 13:03:31 WARN MemoryStore: Not enough space to cache rdd_464_1 in me
mory! (computed 30.7 MiB so far)
23/12/12 13:03:31 WARN MemoryStore: Not enough space to cache rdd_464_3 in me
mory! (computed 8.9 MiB so far)
23/12/12 13:03:32 WARN MemoryStore: Not enough space to cache rdd_464_5 in me
mory! (computed 110.6 MiB so far)
23/12/12 13:03:32 WARN MemoryStore: Not enough space to cache rdd_464_4 in me
mory! (computed 110.6 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_1 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_5 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_0 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_3 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_2 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_4 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_4 in me
mory! (computed 13.6 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_0 in me
mory! (computed 13.6 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_2 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_3 in me
mory! (computed 30.7 MiB so far)
23/12/12 13:03:33 WARN MemoryStore: Not enough space to cache rdd_464_1 in me
mory! (computed 46.6 MiB so far)
23/12/12 13:03:34 WARN MemoryStore: Not enough space to cache rdd_464_5 in me
mory! (computed 110.6 MiB so far)
23/12/12 13:03:34 WARN MemoryStore: Not enough space to cache rdd_464_2 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:34 WARN MemoryStore: Not enough space to cache rdd_464_3 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:34 WARN MemoryStore: Not enough space to cache rdd_464_1 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:34 WARN MemoryStore: Not enough space to cache rdd_464_4 in me
mory! (computed 30.7 MiB so far)
23/12/12 13:03:34 WARN MemoryStore: Not enough space to cache rdd_464_0 in me
mory! (computed 13.6 MiB so far)
23/12/12 13:03:34 WARN MemoryStore: Not enough space to cache rdd_464_5 in me
mory! (computed 110.6 MiB so far)
23/12/12 13:03:35 WARN MemoryStore: Not enough space to cache rdd_464_1 in me
mory! (computed 13.6 MiB so far)
23/12/12 13:03:35 WARN MemoryStore: Not enough space to cache rdd_464_2 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:35 WARN MemoryStore: Not enough space to cache rdd_464_4 in me
mory! (computed 20.5 MiB so far)
23/12/12 13:03:35 WARN MemoryStore: Not enough space to cache rdd_464_0 in me
mory! (computed 30.7 MiB so far)
23/12/12 13:03:35 WARN MemoryStore: Not enough space to cache rdd_464_5 in me
mory! (computed 46.6 MiB so far)
23/12/12 13:03:35 WARN MemoryStore: Not enough space to cache rdd_464_3 in me
```

```
mory! (computed 110.6 MiB so far)
23/12/12 13:03:54 WARN InstanceBuilder: Failed to load implementation from:de
v.ludovic.netlib.blas.JNIBLAS
23/12/12 13:03:54 WARN InstanceBuilder: Failed to load implementation from:de
v.ludovic.netlib.blas.VectorBLAS
[Stage 160:=========================================>             (6 + 2) /
8]

Random Forest Accuracy: 75.63%
Decision Tree Accuracy: 77.04%
Logistic Regression Accuracy: 78.84%
```

In [28]:

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator


# Random Forest evaluation
rf_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
rf_precision = rf_evaluator.evaluate(rf_predictions)
rf_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
rf_recall = rf_evaluator.evaluate(rf_predictions)
rf_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
rf_f1 = rf_evaluator.evaluate(rf_predictions)

# Decision Tree evaluation
dt_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
dt_precision = dt_evaluator.evaluate(dt_predictions)
dt_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
dt_recall = dt_evaluator.evaluate(dt_predictions)
dt_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
dt_f1 = dt_evaluator.evaluate(dt_predictions)

# Logistic Regression evaluation
lr_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
lr_precision = lr_evaluator.evaluate(lr_predictions)
lr_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
lr_recall = lr_evaluator.evaluate(lr_predictions)
lr_evaluator = MulticlassClassificationEvaluator(labelCol="label", predict
lr_f1 = lr_evaluator.evaluate(lr_predictions)

# Print the results
print("Random Forest Precision: {:.2%}".format(rf_precision))
print("Random Forest Recall: {:.2%}".format(rf_recall))
print("Random Forest F1-score: {:.2%}".format(rf_f1))
print("\nDecision Tree Precision: {:.2%}".format(dt_precision))
print("Decision Tree Recall: {:.2%}".format(dt_recall))
print("Decision Tree F1-score: {:.2%}".format(dt_f1))
print("\nLogistic Regression Precision: {:.2%}".format(lr_precision))
print("Logistic Regression Recall: {:.2%}".format(lr_recall))
print("Logistic Regression F1-score: {:.2%}".format(lr_f1))
```

```
[Stage 178:==================================>                    (5 + 3) /
8]

Random Forest Precision: 81.50%
Random Forest Recall: 75.63%
Random Forest F1-score: 65.33%

Decision Tree Precision: 74.03%
Decision Tree Recall: 77.04%
Decision Tree F1-score: 72.16%

Logistic Regression Precision: 76.76%
Logistic Regression Recall: 78.84%
Logistic Regression F1-score: 75.93%
```

Random Forest:

Precision: 81.51% - Out of the instances predicted as positive, 81.51% were correct.

Recall: 75.63% - The model identified 75.63% of all actual positive instances.

F1-score: 65.35% - The harmonic mean of precision and recall. It provides a balance between precision and recall.

High precision but lower recall compared to Decision Tree. If precision is crucial and a slight decrease in recall is acceptable, Random Forest might be a good choice.

Decision Tree:

Precision: 74.03% - Out of the instances predicted as positive, 74.03% were correct.

Recall: 77.04% - The model identified 77.04% of all actual positive instances.

F1-score: 72.16% - The harmonic mean of precision and recall.

Balanced performance with high accuracy, precision, recall, and F1-score. No significant bias towards false positives or false negatives. May be a good choice if overall balanced performance is desired.

Logistic Regression:

Precision: 76.76% - Out of the instances predicted as positive, 76.76% were correct.

Recall: 78.84% - The model identified 78.84% of all actual positive instances.

F1-score: 75.93% - The harmonic mean of precision and recall.

Slightly higher accuracy compared to others. Balanced precision and recall with a good F1-score. May be a good choice if a slightly higher overall accuracy is a priority.

In summary, considering the balanced performance across multiple metrics, Logistic Regression appears to be a reasonable choice. It provides a good balance between precision and recall while achieving a high overall accuracy and F1-score.

Sentiment Analysis :

Vader Lexicon

In [41]:

```
from pyspark.sql.functions import udf
from pyspark.sql.types import StringType, FloatType
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import nltk

# Download Vader Lexicon
nltk.download('vader_lexicon')

# Initialize the VADER SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

# Define UDF for sentiment analysis using Vader
def vader_sentiment_to_categories(sentiment):
    if sentiment == 0:
        return 'Neutral'
    elif sentiment > 0:
        return 'Positive'
    else:
        return 'Negative'

# Define UDF for sentiment analysis scores using Vader
vader_sentiment_score_udf = udf(lambda x: sid.polarity_scores(x)['compound
vader_sentiment_category_udf = udf(lambda x: vader_sentiment_to_categories

# Apply sentiment analysis scores and categories
df_spark_vader = df_spark.withColumn('SentimentScore', vader_sentiment_sc
df_spark_vader = df_spark_vader.withColumn('SentimentCategory', vader_sent

# Display the sentiment analysis results
df_spark_vader.select('Clean_Text', 'SentimentScore', 'SentimentCategory'
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /Users/saikumargandham/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
23/12/12 13:38:50 WARN DAGScheduler: Broadcasting large task binary with size
1201.5 KiB
[Stage 226:>                                                        (0 + 1) /
1]
```

```
+--------------------------------------------------------------------------------
-----------------------------------+-------------+-----------------+
|Clean_Text
|SentimentScore|SentimentCategory|
+--------------------------------------------------------------------------------
-----------------------------------+-------------+-----------------+
|holiday now gonna try sven coop horror footage new video working o httpstcox
1qv2b0cgr                          |-0.25        |Negative         |
|rt amd ryzen 3900x rtx2070 nvidia 64 gigs 144hz dp hacker house set httpstco
bclpet6z7s                         |0.0          |Neutral          |
|recently purchased personal gaming pc build amd ryzen 7 3700x evga gtx 2070
httpstcoclcwdnkjdl                 |0.0          |Neutral          |
|rt linux architect linus torvalds avx512 a hot mess i hope avx512 dies painf
ul death elaborated a              |-0.7506      |Negative         |
|linux architect linus torvalds avx512 a hot mess i hope avx512 dies painful
death elaborated i httpstcok7f1nw395g|-0.7506      |Negative         |
+--------------------------------------------------------------------------------
-----------------------------------+-------------+-----------------+
only showing top 5 rows
```

In [42]:
```
1 df_spark_vader.show(5)
```

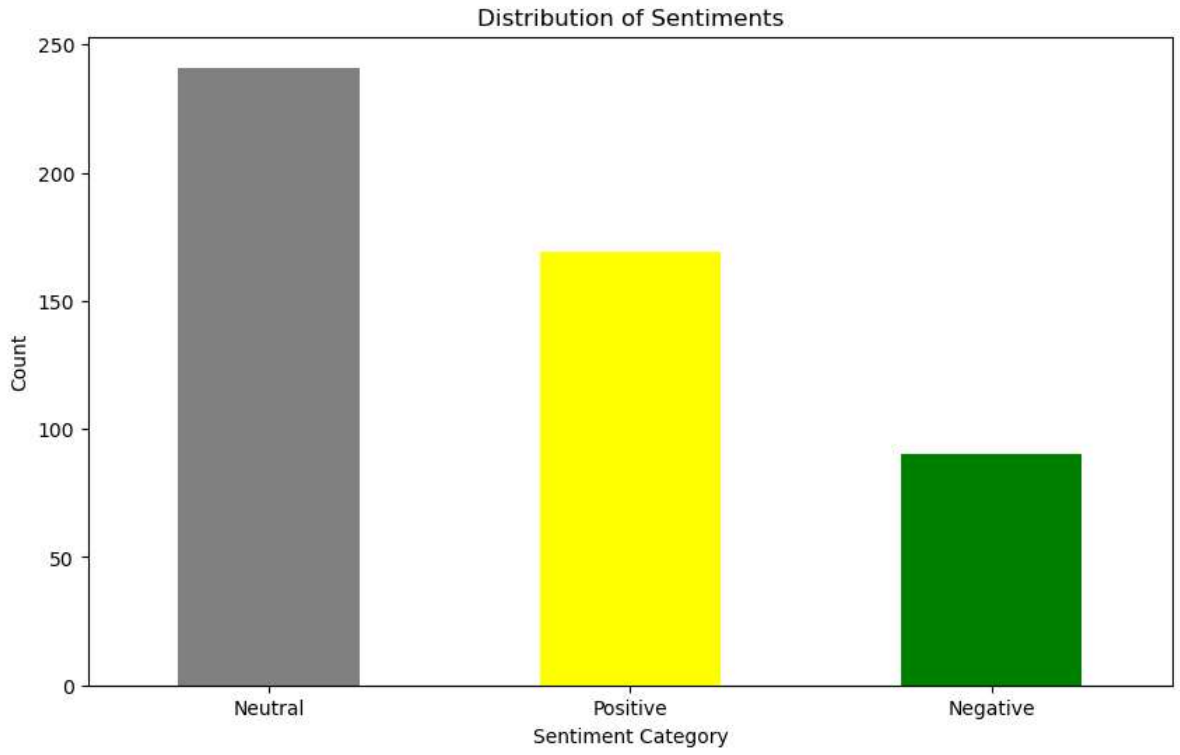
```
23/12/12 13:38:57 WARN DAGScheduler: Broadcasting large task binary with size
1289.8 KiB
[Stage 227:>                                                        (0 + 1) /
1]

+------------------+---------+---------+-------+----------+--------------+--
----------+------------+-----------+--------------------+----------------
---+------------------+--------+----------+----------+---------+----------
+-----------+--------------------+--------------------+--------------------+
--------------------+--------------------+-----+-------------+-------------
---+
|         created_at|file_name|followers|friends|group_name|      location|re
tweet_count|   screenname|search_query|                text|          twitter
_id|          username|polarity|partition_0|partition_1|sentiment|char_length
|token_length|          Clean_Text|               words|      filtered_words|
rawFeatures|            features|label|SentimentScore|SentimentCategory|
+------------------+---------+---------+-------+----------+--------------+--
----------+------------+-----------+--------------------+----------------
---+------------------+--------+----------+----------+---------+----------
+-----------+--------------------+--------------------+--------------------+
--------------------+--------------------+-----+-------------+-------------
---+
|2020-07-12 09:24:26|      AMD|       25|    114|       AMD|United Kingdom|
0.0|moffphcgaming|        #AMD|Been on holiday s...|1.282244417466884...|⛸Mo
ffPHC Gaming⛸| -0.3102| Technology|        AMD| negative|        140|
25|holiday now gonna...|[holiday, now, go...|[holiday, gonna, ...|(500,[56,6
6,85,96...|(500,[56,66,85,96...|  0.0|         -0.25|         Negative|
|2020-07-12 09:09:36|      AMD|      159|   1144|       AMD|  digitalverse|
4.0|   ironparr0t|        #AMD|RT @NinjaParanoid...|1.282240682023719...|
ironparrot|     0.0| Technology|        AMD| negative|        123|          1
7|rt amd ryzen 3900...|[rt, amd, ryzen, ...|[rt, amd, ryzen, ...|(500,[111,11
9,120...|(500,[111,119,120...|  0.0|           0.0|          Neutral|
|2020-07-12 08:31:24|      AMD|        9|    188|       AMD|   HAHA no....|
0.0|  XApochrypha|        #AMD|Recently purchase...|1.282231068649652...|   X
enosApochrypha|     0.0| Technology|        AMD| negative|        140|
22|recently purchase...|[recently, purcha...|[recently, purcha...|(500,[14,1
5,34,18...|(500,[14,15,34,18...|  0.0|           0.0|          Neutral|
|2020-07-12 08:16:45|      AMD|     1719|      1|       AMD|  digitalocean|
1.0|  LinuxDreams|        #AMD|"RT @LinuxReviews...|1.282227383932772...|
LinuxDreams| -0.3612| Technology|        AMD| negative|        146|
25|rt linux architec...|[rt, linux, archi...|[rt, linux, archi...|(500,[0,65,
108,19...|(500,[0,65,108,19...|  0.0|       -0.7506|         Negative|
|2020-07-12 08:11:41|      AMD|       69|    135|       AMD|     Amsterdam|
1.0| LinuxReviews|        #AMD|"#Linux architect...|1.282226107027271...|
LinuxReviews| -0.3612| Technology|        AMD| negative|        146|
22|linux architect l...|[linux, architect...|[linux, architect...|(500,[0,65,
108,12...|(500,[0,65,108,12...|  0.0|       -0.7506|         Negative|
+------------------+---------+---------+-------+----------+--------------+--
----------+------------+-----------+--------------------+----------------
---+------------------+--------+----------+----------+---------+----------
+-----------+--------------------+--------------------+--------------------+
--------------------+--------------------+-----+-------------+-------------
---+
only showing top 5 rows
```

```
23/12/12 13:39:01 WARN PythonUDFRunner: Detected deadlock while completing ta
sk 0.0 in stage 227 (TID 933): Attempting to kill Python Worker
```

In [43]:

```
import matplotlib.pyplot as plt

# Collect a sample of the data to visualize
sample_data = df_spark_vader.limit(500).toPandas()

# Plotting
plt.figure(figsize=(10, 6))
sample_data['SentimentCategory'].value_counts().plot(kind='bar', color=['g
plt.title('Distribution of Sentiments')
plt.xlabel('Sentiment Category')
plt.ylabel('Count')
plt.xticks(rotation=0)  # Rotate x-axis labels for better readability
plt.show()
```

```
23/12/12 13:40:56 WARN DAGScheduler: Broadcasting large task binary with size
1293.5 KiB
23/12/12 13:41:00 WARN PythonUDFRunner: Detected deadlock while completing ta
sk 0.0 in stage 228 (TID 934): Attempting to kill Python Worker
```



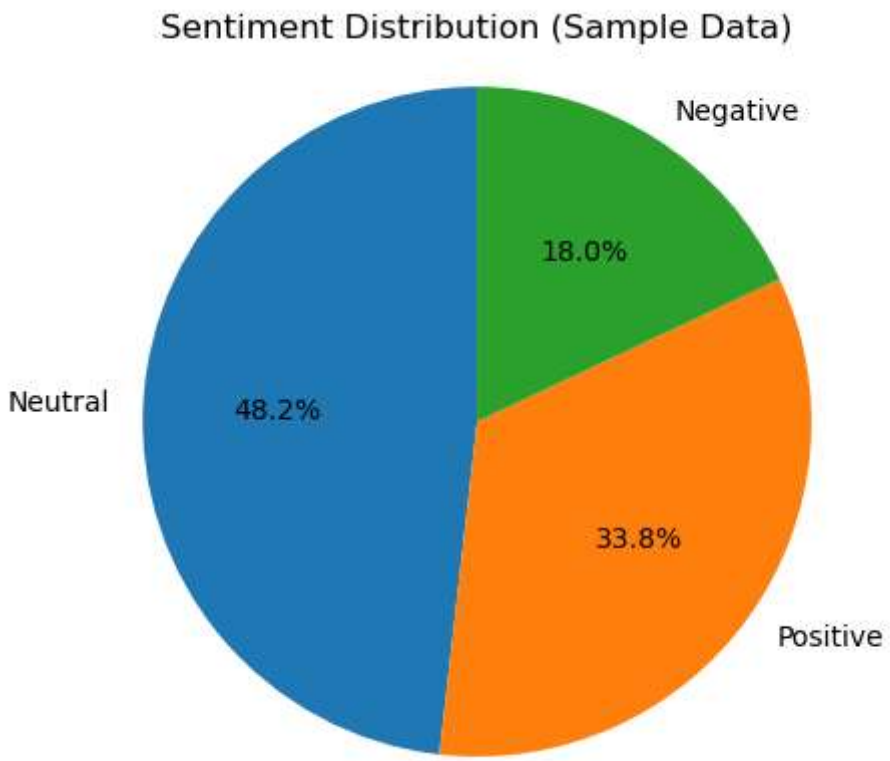
In [44]:
```
1 sample_data = df_spark_vader.limit(500).toPandas()
2 sample_data['SentimentCategory'].value_counts()
```

```
23/12/12 13:41:19 WARN DAGScheduler: Broadcasting large task binary with size
1293.5 KiB
23/12/12 13:41:23 WARN PythonUDFRunner: Detected deadlock while completing ta
sk 0.0 in stage 229 (TID 935): Attempting to kill Python Worker
```

Out[44]:
```
Neutral     241
Positive    169
Negative     90
Name: SentimentCategory, dtype: int64
```

In [45]:
```
import matplotlib.pyplot as plt

# Sample data for demonstration (replace this with your actual DataFrame)
sample_data = df_spark_vader.limit(500).toPandas()

# Calculate sentiment distribution for the sample data
sentiment_counts = sample_data['SentimentCategory'].value_counts()

# Plot pie chart
fig, ax = plt.subplots()
ax.pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%',
ax.axis('equal')  # Equal aspect ratio ensures that the pie is drawn as a

plt.title('Sentiment Distribution (Sample Data)')
plt.show()
```

```
23/12/12 13:41:43 WARN DAGScheduler: Broadcasting large task binary with size
1293.5 KiB
23/12/12 13:41:47 WARN PythonUDFRunner: Detected deadlock while completing ta
sk 0.0 in stage 230 (TID 936): Attempting to kill Python Worker
```



In [48]:
```
unique_locations_count = df_spark_vader.select('location').distinct().cour
print(f"Count of unique location names: {unique_locations_count}")
```

```
[Stage 253:============================>                            (4 + 4) /
8]

Count of unique location names: 86711
```

In [49]:
```
from pyspark.sql.functions import count, first

# Group by 'location', 'group_name', and 'SentimentCategory'
grouped_df = df_spark_vader.groupBy('location', 'group_name', 'SentimentCa

# Pivot the table for better visualization
sentiment_pivot_location = grouped_df.groupBy('location').pivot('Sentiment

# Display the results
sentiment_pivot_location.show(truncate=False)
```

```
[Stage 267:=================================================>       (7 +
1) / 8]

+------------------------------+--------+-------+--------+
|location                      |Negative|Neutral|Positive|
+------------------------------+--------+-------+--------+
|South Asia                    |1       |1      |1       |
|England- Scotland-Brazil-USA  |2       |0      |5       |
|Worcester                     |0       |2      |1       |
|Leicestershire, England, Earth|1       |0      |0       |
|Denver, CO / Dammam, KSA      |1       |0      |0       |
|Heading South                 |1       |0      |0       |
|Brisbane, QLD, Australia      |1       |1      |1       |
|St Petersburg, FL             |3       |1      |8       |
|New Delhi, India              |1       |1      |2       |
|Lakewood CA                   |1       |2      |4       |
|Utah                          |1       |2      |1       |
|International Shipping        |0       |0      |2       |
|England                       |5       |8      |5       |
|Pleasanton, CA                |1       |1      |2       |
```

In [56]:
```
# Checking the text and sentiment for a specific row (index 177)
selected_row = df_spark_vader.select('text', 'SentimentCategory').collect(

# Extracting text and sentiment from the selected row
text = selected_row['text']
sentiment = selected_row['SentimentCategory']

# Displaying the results
print(f'Text: {text}\nSentimentCategory: {sentiment}')

```

```
Text: RT @NeseMusic: As a producer and songwriter, I feel there's always a ma
ke or break moment to see the music video for the first time. Only 1…
SentimentCategory: Neutral
```

In [57]:
```
# Checking the text and sentiment for a specific row (index 127)
selected_row = df_spark_vader.select('text', 'SentimentCategory').collect(

# Extracting text and sentiment from the selected row
text = selected_row['text']
sentiment = selected_row['SentimentCategory']

# Displaying the results
print(f'Text: {text}\nSentimentCategory: {sentiment}')
```

```
Text: RT @sarhan_aashir: Had done my first Gaming&amp;amp;Streaming PC Build last
Sunday #AMD Ryzen 7 3700x...   wanted to stream PUBG but unfortunately…
SentimentCategory: Negative
```

In [58]:
```
# Checking the text and sentiment for a specific row (index 217)
selected_row = df_spark_vader.select('text', 'SentimentCategory').collect(

# Extracting text and sentiment from the selected row
text = selected_row['text']
sentiment = selected_row['SentimentCategory']

# Displaying the results
print(f'Text: {text}\nSentimentCategory: {sentiment}')
```

```
Text: RT @sarvagna_mehta: You only need to find yourself, Everything else can
be googled. . #google #ceo #startup #Entrepreneur  #100DaysOfCode #…
SentimentCategory: Neutral
```

In [59]:

```
from textblob import TextBlob
from pyspark.sql.functions import udf
from pyspark.sql.types import StringType

# Define a UDF (User Defined Function) for sentiment analysis using TextBl
def get_sentiment(text):
    analysis = TextBlob(str(text))
    # Classify as positive, negative, or neutral based on polarity
    if analysis.sentiment.polarity > 0:
        return 'positive'
    elif analysis.sentiment.polarity < 0:
        return 'negative'
    else:
        return 'neutral'

# Register the UDF with Spark
sentiment_udf = udf(get_sentiment, StringType())

# Apply the UDF to your DataFrame
df_spark_textblob = df_spark.withColumn('Sentiment_TextBlob', sentiment_uc

# Show the results
df_spark_textblob.select('Clean_Text', 'Sentiment_TextBlob').show(truncate

```

```
[Stage 295:>                                                        (0 + 1) /
1]
```

```
+--------------------------------------------------------------------------
-----------------------------------------------+-----------------+
|Clean_Text
|Sentiment_TextBlob|
+--------------------------------------------------------------------------
-----------------------------------------------+-----------------+
|holiday now gonna try sven coop horror footage new video working o httpstcox
1qv2b0cgr                                      |positive         |
|rt amd ryzen 3900x rtx2070 nvidia 64 gigs 144hz dp hacker house set httpstco
bclpet6z7s                                     |neutral          |
|recently purchased personal gaming pc build amd ryzen 7 3700x evga gtx 2070
httpstcoclcwdnkjdl                             |neutral          |
|rt linux architect linus torvalds avx512 a hot mess i hope avx512 dies painf
ul death elaborated a                          |negative         |
|linux architect linus torvalds avx512 a hot mess i hope avx512 dies painful
death elaborated i httpstcok7f1nw395g           |negative         |
|rt researchwednesday looking new read article face perception amd published
|positive         |
|rt ps4 motorcycle  zombies  daryl dixon fanfic httpstcon6pf3afsvz latenights
tream daysgone ps4 playstation4                |neutral          |
|ps4 motorcycle  zombies  daryl dixon fanfic httpstcon6pf3afsvz latenightstre
am daysgone ps4 httpstcotayr58ymvh             |neutral          |
|tplink ax3000 wifi 6  best router  archer ax50 httpstco3wq410sen6 intel pc p
cbuild pcgaming amd corsair rgb                |positive         |
|amd ryzen processors fly shelves worlds largest retailers fast perform amd r
yz httpstco4gq2ikdh9d                          |positive         |
|rt utanvirsingh  follow  twitter   youtube httpstcoek4wvdsdvo drdisrespect r
gblights pc                                    |neutral          |
|utanvirsingh  follow  twitter   youtube httpstcoek4wvdsdvo drdisrespect http
stcovp3ewpfkd0                                 |neutral          |
|mean 2700x awesome 570 mobo rx 5700 ryzen 5 3600 good d ryzen amd
|positive         |
|amd stuck range box chart httpstco5m2j685tij httpstcoahzmiuoed7
|neutral          |
|hours troubleshooting finally got working come hang build new pc pcgaming ht
tpstcozce0y85w3j                               |positive         |
| shipped 5 3600 cpus ryzen 3 3200g apus packaging httpstcomyx22i1fi7 technew
s pcmr amd china                               |neutral          |
|right water block running happy it came great computer httpstcopch15z8bpf
|positive         |
|leaked amd epyc milan specifications tease possible 64 zen 3 cores 3 ghz zhi
ye liu 6 hours ago  welcome t httpstcori9u3xdtrg|positive         |
|vendo tarjetagrafica amd rx590 sapphire rx590 8gb gold edition 11000 cdmx me
xico                                           |neutral          |
|rt amd eypc milan genesis cpus spotted  64 zen 3 cores 30 ghz clocks early a
0 samples httpstco7bvxjiaxeg                   |positive         |
+--------------------------------------------------------------------------
-----------------------------------------------+-----------------+
only showing top 20 rows
```

TextBlob

I used TextBlob's polarity values to classify sentiments. TextBlob calculates the sentiment polarity, which is a numerical value indicating the positivity or negativity of the text.

If the polarity is greater than 0, it's considered positive. If it's less than 0, it's considered negative. If the polarity is exactly 0, we consider it as neutral.

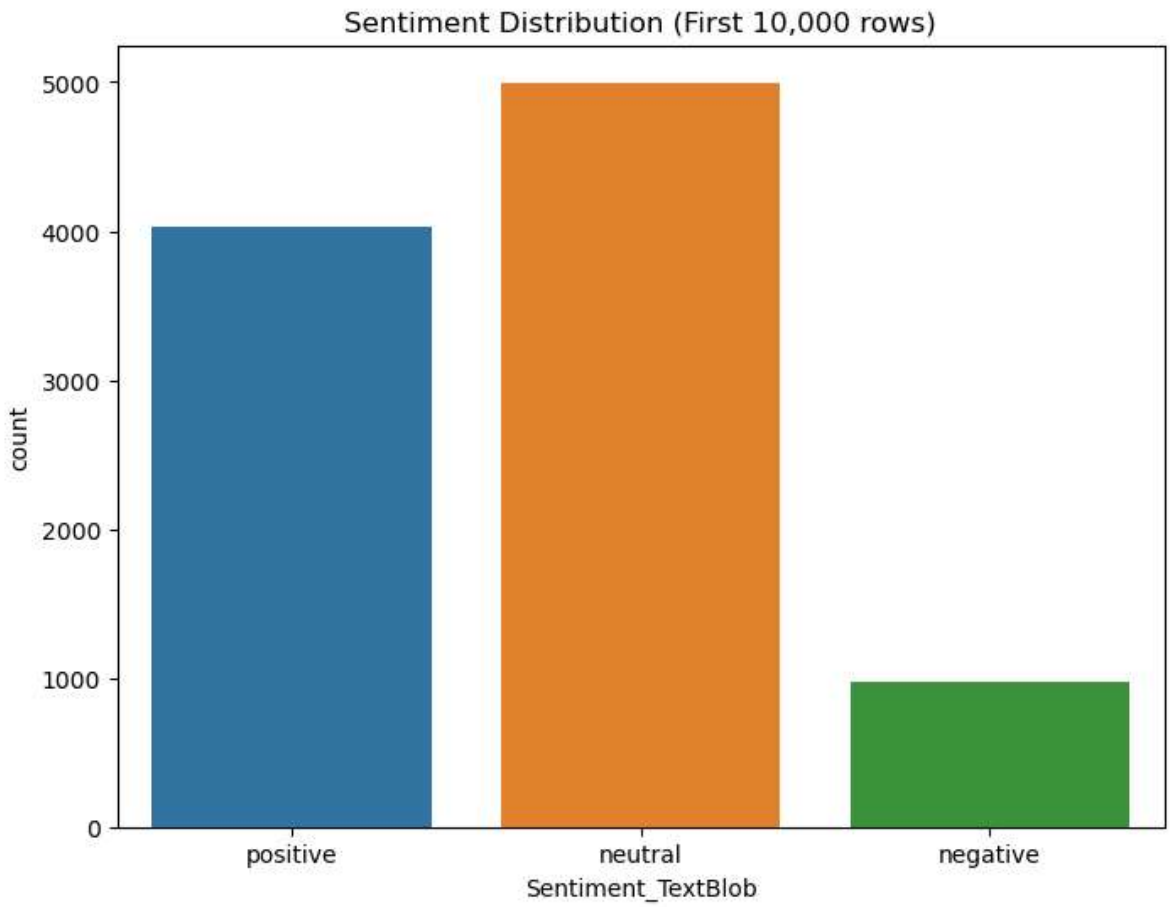
Here's a clarification for better understanding:

Polarity > 0: Positive sentiment

Polarity < 0: Negative sentiment

Polarity = 0: Neutral sentiment

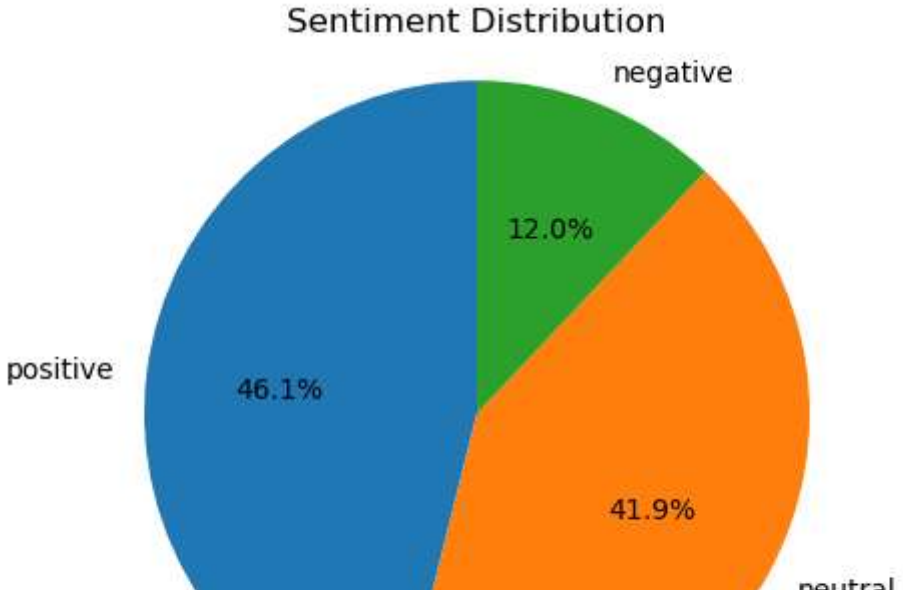
In [60]:
```
import seaborn as sns

# Convert PySpark DataFrame to Pandas DataFrame
df_pandas = df_spark_textblob.select('Sentiment_TextBlob').limit(10000).to

# Plot the sentiment distribution for the first 10,000 rows
plt.figure(figsize=(8, 6))
sns.countplot(x='Sentiment_TextBlob', data=df_pandas)
plt.title('Sentiment Distribution (First 10,000 rows)')
plt.show()
```



In [65]:
```
from pyspark.sql.functions import col

# Assuming your DataFrame is named df_spark
sentiment_counts = df_spark_textblob.groupBy('Sentiment_TextBlob').count()

# Show the counts
sentiment_counts.show()

```

```
[Stage 306:=========================================>               (6 + 2) / 
8]

+------------------+------+
|Sentiment_TextBlob| count|
+------------------+------+
|          positive|367956|
|           neutral|334240|
|          negative| 96171|
+------------------+------+
```

In [66]:
```
# Calculate sentiment distribution
sentiment_counts = df_spark_textblob.groupBy('Sentiment_TextBlob').count()

# Plot pie chart
fig, ax = plt.subplots()
ax.pie(sentiment_counts['count'], labels=sentiment_counts['Sentiment_TextB
ax.axis('equal')  # Equal aspect ratio ensures that the pie is drawn as a

plt.title('Sentiment Distribution')
plt.show()
```



In [68]:
```
# Checking the text and sentiment for a specific row (index 217)
selected_row = df_spark_textblob.select('text', 'Sentiment_TextBlob').coll

# Extracting text and sentiment from the selected row
text = selected_row['text']
sentiment = selected_row['Sentiment_TextBlob']

# Displaying the results
print(f'Text: {text}\nSentiment_TextBlob: {sentiment}')

```

```
Text: RT @sarvagna_mehta: You only need to find yourself, Everything else can
be googled. . #google #ceo #startup #Entrepreneur  #100DaysOfCode #…
Sentiment_TextBlob: neutral
```

In [70]:
```
# Checking the text and sentiment for a specific row (index 127)
selected_row = df_spark_textblob.select('text', 'Sentiment_TextBlob').coll

# Extracting text and sentiment from the selected row
text = selected_row['text']
sentiment = selected_row['Sentiment_TextBlob']

# Displaying the results
print(f'Text: {text}\nSentiment_TextBlob: {sentiment}')
```

```
Text: RT @sarhan_aashir: Had done my first Gaming&amp;amp;Streaming PC Build last
Sunday #AMD Ryzen 7 3700x...   wanted to stream PUBG but unfortunately…
Sentiment_TextBlob: negative
```

In [69]:
```
# Checking the text and sentiment for a specific row (index 177)
selected_row = df_spark_textblob.select('text', 'Sentiment_TextBlob').coll

# Extracting text and sentiment from the selected row
text = selected_row['text']
sentiment = selected_row['Sentiment_TextBlob']

# Displaying the results
print(f'Text: {text}\nSentiment_TextBlob: {sentiment}')

```

```
Text: RT @NeseMusic: As a producer and songwriter, I feel there's always a ma
ke or break moment to see the music video for the first time. Only 1…
Sentiment_TextBlob: neutral
```

As observed in the outcomes from both VADER and TextBlob analyses, it is evident that the sentiment predictions align closely, yielding similar results for the sentiment categorization of the given text data. The agreement in sentiment classifications between the VADER lexicon and TextBlob suggests a consistent interpretation of sentiment across both approaches, reinforcing the reliability and concordance of the sentiment analysis results obtained from these methods.

In [ ]:
```
1
```

Conclusion:

- *Model Evaluation:* After training and evaluating Random Forest, Decision Tree, and Logistic Regression classifiers, the models demonstrated varying performances in terms of accuracy, precision, recall, and F1-score. Logistic Regression emerged as a balanced choice, showcasing competitive results across multiple metrics.

- *Sentiment Analysis:*
  - *Polarity-Based Approach:* Analyzed the sentiment of tweets based on polarity scores.Visualized sentiment distribution and explored patterns.
  - Utilizing both VADER and TextBlob for sentiment analysis, the project observed a harmonious alignment in sentiment categorizations. This convergence reinforces the reliability of sentiment analysis results, affirming a consistent interpretation of sentiment across different lexicons.

- *Text Features and Exploration:* Exploring text-related features, including character and token lengths, provided insights into the structure of tweets. Additionally, the analysis of sentiment distributions across companies shed light on variations in public sentiment towards different big tech entities.

- *Location-Based Analysis:* Examining sentiment distribution across unique locations allowed for a more granular understanding of regional sentiments, contributing valuable insights for strategic decision-making.

- *Overall Insights:* The project contributes to understanding public sentiments towards big tech companies, offering a comprehensive analysis that incorporates both machine learning classifiers and lexicon-based sentiment analysis.

- *Recommendations:* Based on the findings, stakeholders can make informed decisions about corporate reputation management, brand strategies, and customer engagement approaches. The insights derived from this sentiment analysis project can be valuable for addressing public perceptions and adapting business strategies accordingly.

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```