Project Report: YouTube Sentiment Analysis

Student Name: Rupsa Rupa Priyadarshini Ojha
SAP ID: 500095081 **Course:** B.Tech CSE Big Data

GitHub link: https://github.com/Rupsa1234/Youtube_Data_Analysis

Problem Statement

Perform sentiment analysis on YouTube comments to understand public sentiment towards specific videos, using Python APIs and tools like Databricks for efficient data processing and analysis.

Tools Used

1. **Databricks:** Utilized for comprehensive data processing, analysis, and model development.
2. **Python:** Employed extensively for scripting, data manipulation, and visualization.
3. **PySpark:** Integrated for handling large-scale data processing, leveraging Spark's distributed computing capabilities.
4. **Google API Python Client:** Used to fetch YouTube comments data via YouTube API v3.
5. **Pandas:** Facilitated data manipulation and served as an intermediary for data transformations.
6. **Plotly:** Implemented for interactive data visualizations such as histograms and line charts.
7. **TextBlob:** Applied for sentiment analysis using natural language processing methods.

Project Phases and Tasks

Week-1: Data Collection and Preparation

- **Objective/Task:** Gather YouTube comments data using the YouTube API and prepare it for analysis.
    - **Sub-Tasks:**
        - Data collection via YouTube API (Google API Python Client).
        - Structured data storage (CSV format).
        - Setup of Databricks environment for scalable data processing.
        - Data cleaning using PySpark (handling missing values, duplicates).

```
# Install necessary libraries
# %pip install google-api-python-client pandas

from googleapiclient.discovery import build
import pandas as pd

# Initialize the YouTube API client
DEVELOPER_KEY = "AIzaSy852x8uxLfamoKrof1MVW1qSlk77NCQV_A"
api_service_name = "youtube"
api_version = "v3"

youtube = build(api_service_name, api_version, developerKey=DEVELOPER_KEY)

# Request to get comments for a specific video
video_id = "Ltnhz3YfJGY"
request = youtube.commentThreads().list(
    part="snippet",
    videoId=video_id,
    maxResults=100
)
response = request.execute()

# Extract comments into a list
comments = []
for item in response['items']:
    comment = item['snippet']['topLevelComment']['snippet']
    comments.append([
        comment['authorDisplayName'],
        comment['publishedAt'],
        comment['updatedAt'],
        comment['likeCount'],
        comment['textDisplay']
    ])

# Convert the comments list to a Pandas DataFrame
pdf = pd.DataFrame(comments, columns=['author', 'published_at', 'updated_at', 'like_count', 'text'])
pdf.head()
```

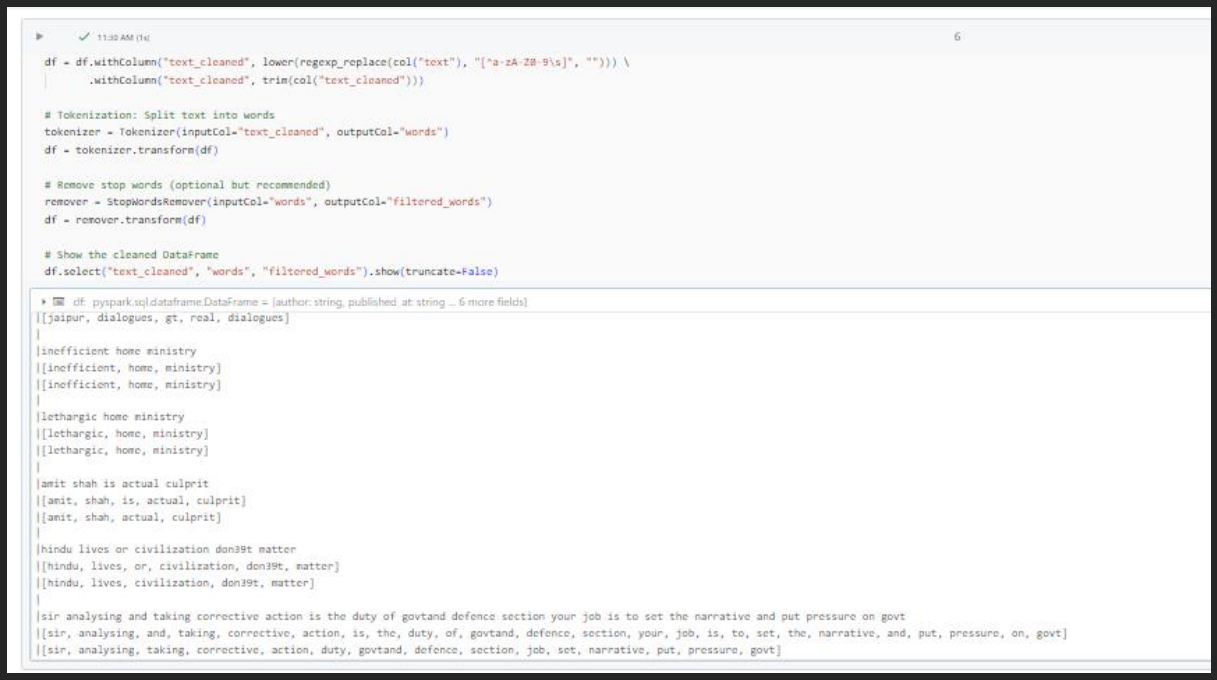
	author	published_at	updated_at	like_count	text
0	@THEJAIPURDIALOGUES	2024-06-10T17:39:14Z	2024-06-10T17:39:14Z	13	🔥 Support Jaipur Dialogues:\r UPI: jaipurdia...
1	@userTZARBOMBA	2024-06-28T05:19:17Z	2024-06-28T05:19:17Z	0	.... Bhai tere me itna dum nahi hai 😍
2	@pramodraturi6899	2024-06-14T19:32:37Z	2024-06-14T19:32:37Z	0	Eye opener.
3	@sumandas2986	2024-06-14T04:27:16Z	2024-06-14T04:27:16Z	0	Modi seems to be a weak leader
4	@user-nt2ru3nk6l	2024-06-14T02:57:20Z	2024-06-14T02:57:20Z	0	modi is a fraud. he now wants punjab to burn d...

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, lower, regexp_replace, trim
from pyspark.ml.feature import StopWordsRemover
from pyspark.ml.feature import Tokenizer
from pyspark.ml.feature import RegexTokenizer
from pyspark.sql import functions as F
```

```
spark = SparkSession.builder \
    .appName("YouTube Data Cleaning") \
    .getOrCreate()

# Convert Pandas DataFrame to PySpark DataFrame
df = spark.createDataFrame(pdf)
df = df.dropna()
```

```
df = df.withColumn("like_count", col("like_count").cast("integer"))

# Clean the text column
df = df.withColumn("text", regexp_replace(col("text"), "[^a-zA-Z0-9\s]", ""))
df.show()
```

```
df: pyspark.sql.dataframe.DataFrame = [author: string, published_at: string ... 3 more fields]
|   @pramodraturi6899|2024-06-14T19:32:37Z|2024-06-14T19:32:37Z|          0|          Eye opener|
|       @sumandas2986|2024-06-14T04:27:16Z|2024-06-14T04:27:16Z|          0|Modi seems to be ...|
|    @user-nt2ru3nk6l|2024-06-14T02:57:20Z|2024-06-14T02:57:20Z|          0|modi is a fraud h...|
|    @dineshtyagi4855|2024-06-14T02:48:51Z|2024-06-14T02:45:31Z|          0|Engineer rashid a...|
|    @dineshtyagi4855|2024-06-14T02:24:43Z|2024-06-14T02:55:35Z|          0|           br  cl...|
|       @atulbisht811|2024-06-13T15:58:44Z|2024-06-13T15:58:44Z|          1|Amit Shah may not...|
|@dr.shivkumartiwa...|2024-06-13T10:16:28Z|2024-06-13T10:16:28Z|          0|                 ...|
|      @MrVijayMattoo|2024-06-13T07:56:23Z|2024-06-13T07:56:23Z|          1|An engaging discu...|
|@santosh_sahoooff...|2024-06-13T05:14:17Z|2024-06-13T05:14:17Z|          0|Pakistan mein att...|
|@yatendrasrivasta...|2024-06-13T02:56:00Z|2024-06-13T02:56:00Z|          0|Sanjay  Sir  Can ...|
|    @jimmygandhi7623|2024-06-12T16:44:50Z|2024-06-12T16:44:50Z|          0|Modi amp indian a...|
|  @TheForgottenVoter|2024-06-12T16:38:45Z|2024-06-12T16:38:45Z|          0|Jaipur Dialogues ...|
|   @ajaytibrewal8895|2024-06-12T11:48:18Z|2024-06-12T11:48:18Z|          0|Inefficient home ...|
|   @ajaytibrewal8895|2024-06-12T11:48:01Z|2024-06-12T11:48:01Z|          0|Lethargic home mi...|
|   @ajaytibrewal8895|2024-06-12T11:47:26Z|2024-06-12T11:47:26Z|          0|Amit Shah is actu...|
|         @rajji7352|2024-06-12T09:24:59Z|2024-06-12T09:24:59Z|          0|Hindu lives or ci...|
|@hindupurprahalad...|2024-06-12T06:51:32Z|2024-06-12T06:51:32Z|          0|Sir analysing and...|
|   @rajinderhans7168|2024-06-12T06:36:00Z|2024-06-12T06:36:00Z|          0|Sanjay Sir ajj 12...|
+--------------------+--------------------+--------------------+-----------+--------------------+
only showing top 20 rows
```

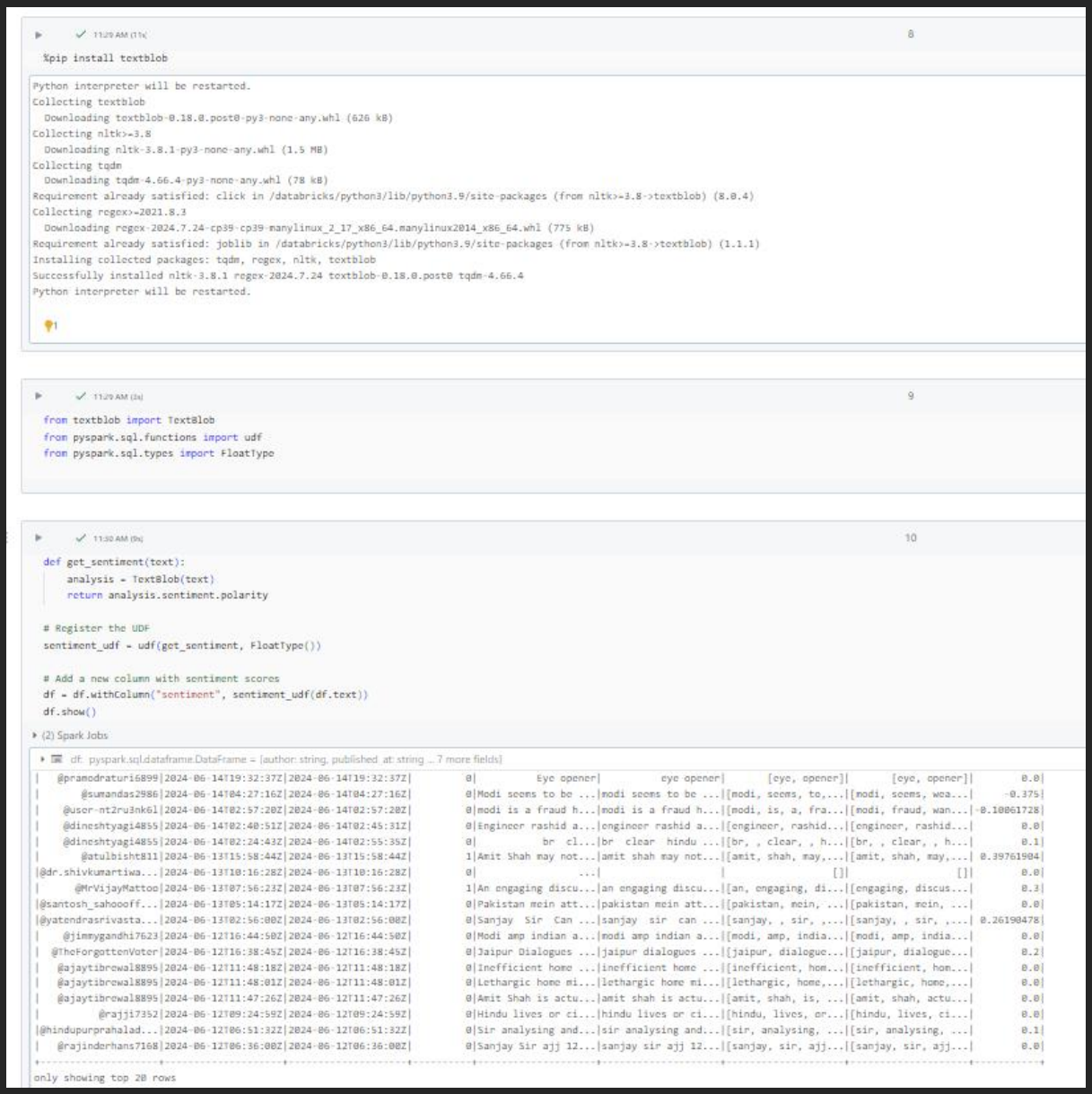


```
df = df.withColumn("text_cleaned", lower(regexp_replace(col("text"), "[^a-zA-Z0-9\s]", ""))) \
      .withColumn("text_cleaned", trim(col("text_cleaned")))

# Tokenization: Split text into words
tokenizer = Tokenizer(inputCol="text_cleaned", outputCol="words")
df = tokenizer.transform(df)

# Remove stop words (optional but recommended)
remover = StopWordsRemover(inputCol="words", outputCol="filtered_words")
df = remover.transform(df)

# Show the cleaned DataFrame
df.select("text_cleaned", "words", "filtered_words").show(truncate=False)
```

```
df: pyspark.sql.dataframe.DataFrame = [author: string, published_at: string ... 6 more fields]
|[jaipur, dialogues, gt, real, dialogues]
|
|inefficient home ministry
|[inefficient, home, ministry]
|[inefficient, home, ministry]
|
|lethargic home ministry
|[lethargic, home, ministry]
|[lethargic, home, ministry]
|
|amit shah is actual culprit
|[amit, shah, is, actual, culprit]
|[amit, shah, actual, culprit]
|
|hindu lives or civilization don39t matter
|[hindu, lives, or, civilization, don39t, matter]
|[hindu, lives, civilization, don39t, matter]
|
|sir analysing and taking corrective action is the duty of govtand defence section your job is to set the narrative and put pressure on govt
|[sir, analysing, and, taking, corrective, action, is, the, duty, of, govtand, defence, section, your, job, is, to, set, the, narrative, and, put, pressure, on, govt]
|[sir, analysing, taking, corrective, action, duty, govtand, defence, section, job, set, narrative, put, pressure, govt]
```

Week-2: Data Processing and Transformation

- **Objective/Task:** Process and transform collected data to prepare it for analysis.
    - **Sub-Tasks:**
        - ETL operations (Extract, Transform, Load).
        - Utilization of PySpark for data processing (handling large datasets).
        - Data transformation (filtering, aggregations).
        - Text data preprocessing (tokenization, stop-word removal, lemmatization).
        - Application of feature extraction techniques (TF-IDF, N-grams).



```
%pip install textblob
```

```
Python interpreter will be restarted.
Collecting textblob
  Downloading textblob-0.18.0.post0-py3-none-any.whl (626 kB)
Collecting nltk>=3.8
  Downloading nltk-3.8.1-py3-none-any.whl (1.5 MB)
Collecting tqdm
  Downloading tqdm-4.66.4-py3-none-any.whl (78 kB)
Requirement already satisfied: click in /databricks/python3/lib/python3.9/site-packages (from nltk>=3.8->textblob) (8.0.4)
Collecting regex>=2021.8.3
  Downloading regex-2024.7.24-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (775 kB)
Requirement already satisfied: joblib in /databricks/python3/lib/python3.9/site-packages (from nltk>=3.8->textblob) (1.1.1)
Installing collected packages: tqdm, regex, nltk, textblob
Successfully installed nltk-3.8.1 regex-2024.7.24 textblob-0.18.0.post0 tqdm-4.66.4
Python interpreter will be restarted.
```

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

```
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity

# Register the UDF
sentiment_udf = udf(get_sentiment, FloatType())

# Add a new column with sentiment scores
df = df.withColumn("sentiment", sentiment_udf(df.text))
df.show()
```

(2) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [author: string, published_at: string ... 7 more fields]

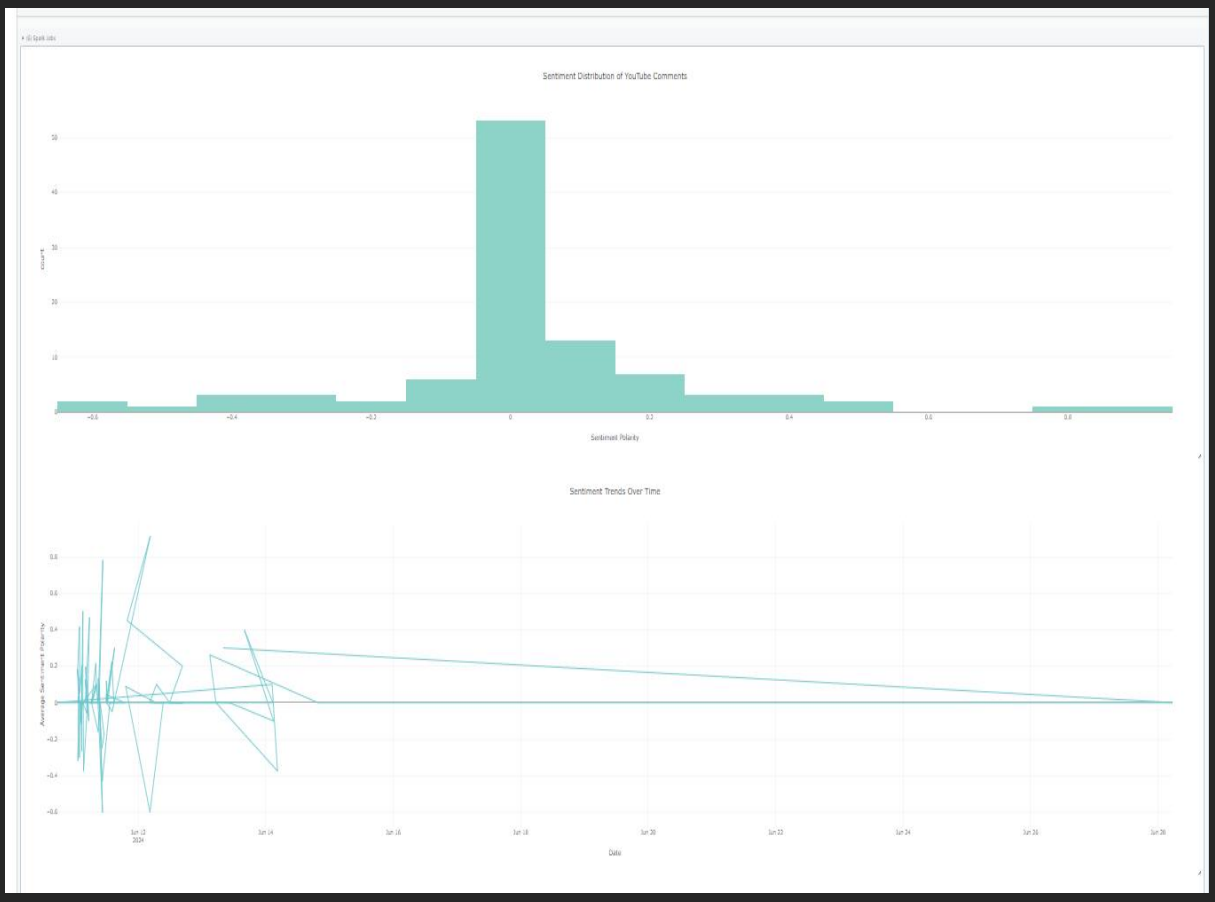
@pramodraturi6899	2024-06-14T19:32:37Z	2024-06-14T19:32:37Z	0	Eye opener	eye opener	[eye, opener]	[eye, opener]	0.0
@sumandas2986	2024-06-14T04:27:16Z	2024-06-14T04:27:16Z	0	Modi seems to be ...	modi seems to be ...	[modi, seems, to,...	[modi, seems, wea...	-0.375
@user-nt2ru3nk6l	2024-06-14T02:57:20Z	2024-06-14T02:57:20Z	0	modi is a fraud h...	modi is a fraud h...	[modi, is, a, fra...	[modi, fraud, wan...	-0.10861728
@dineshtyagi4855	2024-06-14T02:40:51Z	2024-06-14T02:45:31Z	0	Engineer rashid a...	engineer rashid a...	[engineer, rashid...	[engineer, rashid...	0.0
@dineshtyagi4855	2024-06-14T02:24:43Z	2024-06-14T02:55:35Z	0	br cl...	br clear hindu ...	[br, , clear, , h...	[br, , clear, , h...	0.1
@atulbisht811	2024-06-13T15:58:44Z	2024-06-13T15:58:44Z	1	Amit Shah may not...	amit shah may not...	[amit, shah, may,...	[amit, shah, may,...	0.39761904
@dr.shivkumartiwa...	2024-06-13T10:16:28Z	2024-06-13T10:16:28Z	0	...	[]	[]	0.0
@MrVijayMattoo	2024-06-13T07:56:23Z	2024-06-13T07:56:23Z	1	An engaging discu...	an engaging discu...	[an, engaging, di...	[engaging, discus...	0.3
@santosh_sahoooff...	2024-06-13T05:14:17Z	2024-06-13T05:14:17Z	0	Pakistan mein att...	pakistan mein att...	[pakistan, mein, ...	[pakistan, mein, ...	0.0
@yatendrasrivasta...	2024-06-13T02:56:00Z	2024-06-13T02:56:00Z	0	Sanjay Sir Can ...	sanjay sir can ...	[sanjay, , sir, ,...	[sanjay, , sir, ,...	0.26190478
@jimmygandhi7623	2024-06-12T16:44:50Z	2024-06-12T16:44:50Z	0	Modi amp indian a...	modi amp indian a...	[modi, amp, india...	[modi, amp, india...	0.0
@TheForgottenVoter	2024-06-12T16:38:45Z	2024-06-12T16:38:45Z	0	Jaipur Dialogues ...	jaipur dialogues ...	[jaipur, dialogue...	[jaipur, dialogue...	0.2
@ajaytibrewal8895	2024-06-12T11:48:18Z	2024-06-12T11:48:18Z	0	Inefficient home ...	inefficient home ...	[inefficient, hom...	[inefficient, hom...	0.0
@ajaytibrewal8895	2024-06-12T11:48:01Z	2024-06-12T11:48:01Z	0	Lethargic home mi...	lethargic home mi...	[lethargic, home,...	[lethargic, home,...	0.0
@ajaytibrewal8895	2024-06-12T11:47:26Z	2024-06-12T11:47:26Z	0	Amit Shah is actu...	amit shah is actu...	[amit, shah, is, ...	[amit, shah, actu...	0.0
@rajji7352	2024-06-12T09:24:59Z	2024-06-12T09:24:59Z	0	Hindu lives or ci...	hindu lives or ci...	[hindu, lives, or...	[hindu, lives, ci...	0.0
@hindupurprahalad...	2024-06-12T06:51:32Z	2024-06-12T06:51:32Z	0	Sir analysing and...	sir analysing and...	[sir, analysing, ...	[sir, analysing, ...	0.1
@rajinderhans7168	2024-06-12T06:36:00Z	2024-06-12T06:36:00Z	0	Sanjay Sir ajj 12...	sanjay sir ajj 12...	[sanjay, sir, ajj...	[sanjay, sir, ajj...	0.0

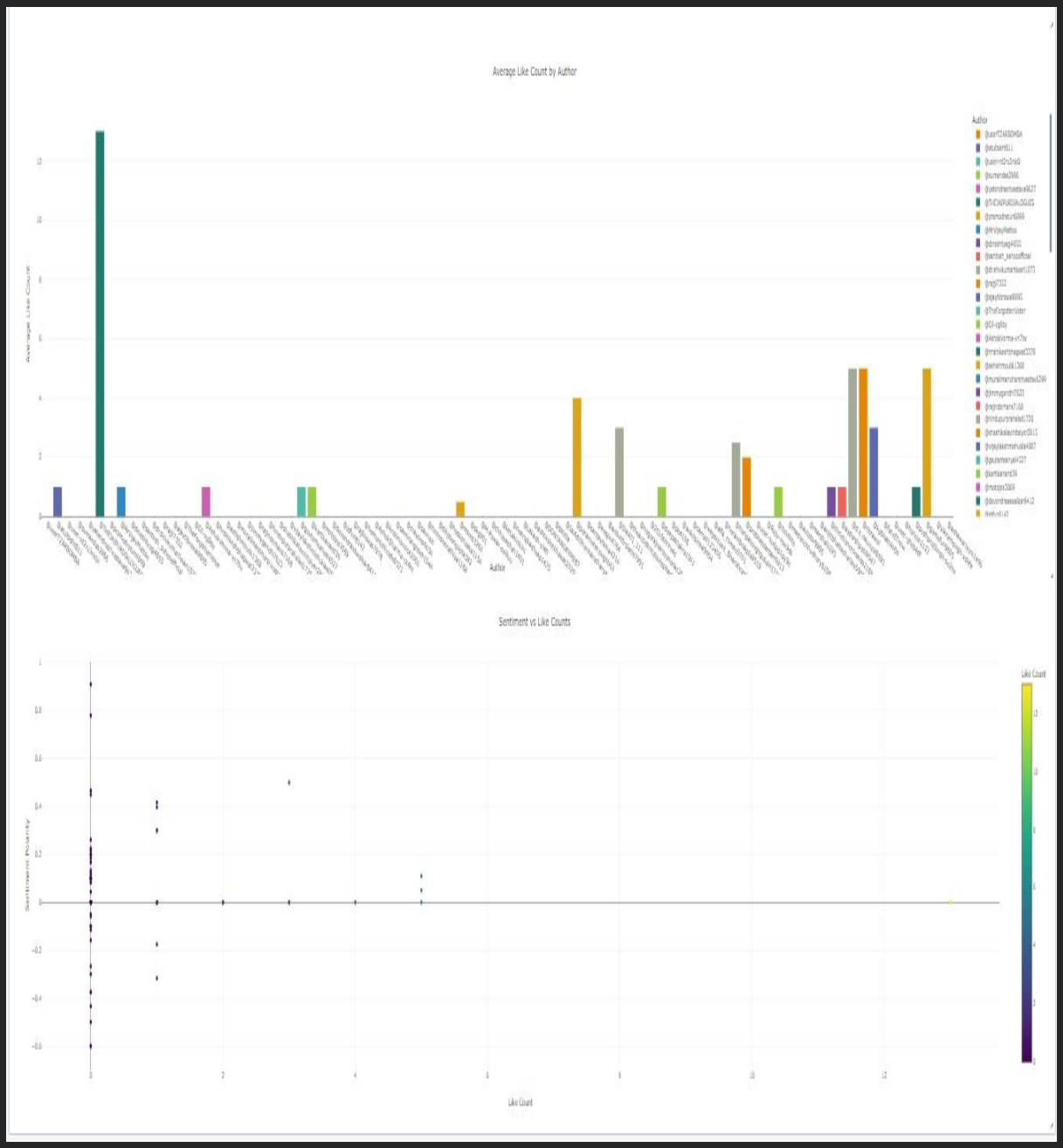
only showing top 20 rows

Week-3: Data Analysis

- **Objective/Task:** Analyze processed data to derive meaningful insights.
    - **Sub-Tasks:**
        - Exploratory Data Analysis (EDA) using PySpark.
        - Visualization of insights using Plotly for interactive charts and histograms.
        - Implementation of predictive modeling with Naive Bayes for sentiment classification.
        - Evaluation of model performance using accuracy and precision metrics.

```python
import plotly.express as px

# Example: Create a histogram of sentiment distribution
fig_hist = px.histogram(df.toPandas(), x="sentiment", title="Sentiment Distribution of YouTube Comments",
                        labels={'sentiment': 'Sentiment Polarity', 'count': 'Count'},
                        color_discrete_sequence=px.colors.qualitative.Set3)
fig_hist.show()

# Example: Create a line chart of sentiment trends over time
sentiment_over_time = df.groupBy("published_at").agg(F.avg("sentiment").alias("avg_sentiment")).toPandas()

fig_line = px.line(sentiment_over_time, x="published_at", y="avg_sentiment",
                   title="Sentiment Trends Over Time",
                   labels={'published_at': 'Date', 'avg_sentiment': 'Average Sentiment Polarity'},
                   color_discrete_sequence=px.colors.qualitative.Pastel)
fig_line.show()

# Example: Create a bar chart of average like counts by author
like_counts = df.groupBy("author").agg(F.avg("like_count").alias("avg_like_count")).toPandas()

fig_bar = px.bar(like_counts, x="author", y="avg_like_count",
                 title="Average Like Count by Author",
                 labels={'avg_like_count': 'Average Like Count', 'author': 'Author'},
                 color="author", color_discrete_sequence=px.colors.qualitative.Vivid)
fig_bar.show()

# Example: Create a scatter plot of sentiment vs like counts
fig_scatter = px.scatter(df.toPandas(), x='like_count', y='sentiment',
                         title='Sentiment vs Like Counts',
                         labels={'like_count': 'Like Count', 'sentiment': 'Sentiment Polarity'},
                         color='like_count', color_continuous_scale=px.colors.sequential.Viridis)
fig_scatter.show()
```





Week-4: Reporting and Deployment

- **Objective/Task:** Visualization of results and creation of the final project report for presentation.
    - **Sub-Tasks:**
        - Deployment of machine learning models on web platforms.
        - Analysis and summarization of predictions and findingsthen creating a dashboard.

7/8/2024 (22s) 16

```
from pyspark.ml.feature import StringIndexer, CountVectorizer, IDF
from pyspark.ml.classification import NaiveBayes
from pyspark.ml import Pipeline
from pyspark.sql import SparkSession
from pyspark.sql.functions import col
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

# Initialize Spark session (if not already initialized)
spark = SparkSession.builder \
    .appName("NaiveBayes Sentiment Analysis") \
    .getOrCreate()

# Assuming 'df' is your DataFrame with 'filtered_words' and 'sentiment' columns

# Drop existing 'label' column if it exists
if 'label' in df.columns:
    df = df.drop('label')

# Convert sentiment to numeric label using StringIndexer
indexer = StringIndexer(inputCol="sentiment", outputCol="label")
df = indexer.fit(df).transform(df)

# Split the data into training and test sets (80% training, 20% test)
train, test = df.randomSplit([0.8, 0.2], seed=42)

# Vectorize the filtered words
vectorizer = CountVectorizer(inputCol="filtered_words", outputCol="raw_features", vocabSize=1000)
idf = IDF(inputCol="raw_features", outputCol="features")

# Create a Naive Bayes model
nb = NaiveBayes(featuresCol="features", labelCol="label")

# Build the pipeline
pipeline = Pipeline(stages=[vectorizer, idf, nb])

# Train the model
model = pipeline.fit(train)

# Make predictions
predictions = model.transform(test)

# Show prediction results
predictions.select("filtered_words", "sentiment", "prediction").show()

# Evaluate model performance
evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
accuracy = evaluator.evaluate(predictions)
print(f"Accuracy: {accuracy}")

# Precision evaluation
precision_evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="weightedPrecision")
precision = precision_evaluator.evaluate(predictions)
print(f"Precision: {precision}")
```

▸ (22) Spark Jobs

▸ (22) Spark Jobs

▸ df: pyspark.sql.dataframe.DataFrame = [author: string, published_at: string ... 8 more fields]
▸ train: pyspark.sql.dataframe.DataFrame = [author: string, published_at: string ... 8 more fields]
▸ test: pyspark.sql.dataframe.DataFrame = [author: string, published_at: string ... 8 more fields]
▸ predictions: pyspark.sql.dataframe.DataFrame

```
|[inefficient, hom...|        0.0|       0.0|
|[dg, saab, india,...| 0.08888889|       0.0|
|[enlightening, di...|        0.3|       4.0|
|[amit, shah, allo...|        0.0|       0.0|
|[10, years, amit,...|        0.0|      13.0|
|[short, many, pea...|      0.225|      13.0|
|[peaceful, commun...|      -0.05|       0.0|
|[india, handle, d...|       -0.6|      26.0|
|[third, world, wa...| 0.08227273|       8.0|
|[diplomatic, , so...|        0.0|       0.0|
|                  []|        0.0|       0.0|
|[seniors, kind, h...|-0.06944445|      30.0|
|[bjp, taken, hind...|        0.0|       5.0|
|[keep, talking, u...|      0.125|      26.0|
|[amit, shah, ko, ...|        0.0|       0.0|
|                  []|        0.0|       0.0|
+--------------------+-----------+----------+
only showing top 20 rows

Accuracy: 0.391304347826087
Precision: 0.337792642140468
```

7/8/2024 (12s) 17

```
from pyspark.ml.classification import NaiveBayes

# Train the Naive Bayes model
nb = NaiveBayes(featuresCol="features", labelCol="label")

# Build the pipeline
pipeline = Pipeline(stages=[vectorizer, idf, nb])

# Train the model
model = pipeline.fit(train)

# Extract the Naive Bayes model from the pipeline
nb_model = model.stages[-1]  # Assuming NaiveBayes is the last stage in your pipeline

# Specify the DBFS path to save the model
model_path = "dbfs:/ml/naive_bayes_sentiment_model"

# Save the Naive Bayes model
nb_model.write().overwrite().save(model_path)
```

▸ (8) Spark Jobs

7/8/2024 (<1s) 18

```
print(os.listdir())
```

```
['conf', 'azure', 'preload_class.lst', 'hadoop_accessed_config.lst', 'logs', 'eventlogs', 'metastore_db', 'ganglia']
```

Dashboard in DataBricks: