Cloud Assignment

Global YouTube Trending Video Analytics using Apache Spark

1. Student Details

Student Name:

Jasnain Kaur Ajit Singh Banga , Sakshi Ram Wagh

Student ID:

A00052855 , A00052794

Email ID:

jasnainkaurajitsingh.banga2@mail.dcu.ie

sakshiram.wagh2@mail.dcu.ie

2. Link for the Dataset and Source Code Git Repository

Dataset (Kaggle): *Trending YouTube Videos – 113 Countries*

https://www.kaggle.com/datasets/asaniczka/trending-youtube-videos-113-countries

Source Code Repository (GitLab):

https://gitlab.computing.dcu.ie/sakshiram.wagh2/cloud_assignment/-/tree/main/Cloud_Assignment_Code

3. Link to Video Demonstration

https://drive.google.com/file/d/1sj3D5mLTqB3_krHI8UNWGYopSUjFr4to/view?usp=sharing

4. Introduction and Motivation

YouTube, one of the biggest internet video platforms that has become an essential part of today's digital world. Every day, billions of people watch videos, and millions of hours are uploaded. One of the most important features of YouTube is the Trending section. It shows videos that are becoming popular very quickly in every country. These popular movies are not only the entertainment choices but also reflect the cultural and societal interests as well as the patterns of public attention.

However, it is challenging to understand global trends in behaviour. With 113 countries, several categories, and long time periods, the dataset is large. Spreadsheets and basic Python scripts are examples of old single-machine solutions that quickly become slow or impractical when handling millions of data. Due to its complex links, diverse formats, and missing information, the raw data must be extensively preprocessed and aggregated before meaningful insights may be obtained.

This project's objective is to create and deploy a cloud data analytics pipeline using Apache Spark in order to analyse and evaluate global YouTube trending data. Apache Spark, a distributed big-data processing platform that enables parallel computations over large datasets, provides high-level Python APIs for data purification, transformation, and aggregation (PySpark).

Our specific focus is to study **country-level differences** in YouTube trending behaviour and audience engagement. By transforming the raw Kaggle data into a **cleaned, aggregated dataset** summarising, for each country:**(1)** the number of **unique trending videos ,(2)** the **average views** of those videos , **(3)** the **average engagement rate** (likes + comments per view).

We are able to systematically compare YouTube usage and consumption trends throughout the globe. The primary learning goals of the cloud technologies module include working with cloud technologies, large public datasets, and developing a non-trivial data analytics pipeline from intake to results.

5. Related Work

This project intersects large-scale social video analytics, engagement prediction on YouTube, and big-data processing frameworks. Below are three key related works and how our solution differs.

5.1 “Analysis of Comments on YouTube Videos using Hadoop”

Dabas, Kaur, Gulati, and Tilak (2019) created a Hadoop MapReduce-based pipeline to assess sentiment and trending patterns in millions of YouTube comments. They focus on comment-text data using Hadoop's batch processing method. Our method differs in three key ways: (1) we use metadata (views, likes, and comment counts) instead of comment text; (2) we operate in 113 countries worldwide rather than just one; and (3) we use Apache Spark, which is more flexible and in-memory than Hadoop MapReduce, enabling more complex transformations and aggregations.

5.2 “Examining Factors That Predict User Engagement on YouTube”

Qureshi (2019) studies user engagement on YouTube in the context of sports-brand videos, proposing a model that examines factors such as channel subscriber count, video age, video duration and channel type, and correlates them with engagement levels. While relevant in topic (YouTube engagement), our project differs by (1) using automatic metadata for over 100 countries rather than one domain; (2) focusing on country-level aggregated metrics (unique trending videos, average views, average engagement rate) rather than modelling individual video/predicting engagement; (3) using Apache Spark for large-scale pipeline processing rather than statistical modelling only.

5.3 “Optimizing Prediction of YouTube Video Popularity Using XGBoost”

Nisa et al. (2021) propose a machine-learning technique to evaluate YouTube video popularity using the XGBoost algorithm. Their approach looks at video data, such as views, likes, comments, category, and other features, to predict future trends in YouTube video popularity. While it is relevant due to its comparable engagement-based features and emphasis on YouTube statistics, our project differs in three ways: Instead of predicting the popularity of individual videos, it does large-scale country-level aggregation across 113 countries; (2) it prioritises descriptive analytics (such as unique trending videos, average views, and average engagement rate) over predictive modelling; and (3) it uses Apache Spark for scalable big-data processing in a cloud-oriented pipeline rather than a stand-alone machine-learning workflow.

Differences From Our Work

In conclusion, our approach is distinct in that it looks at a multi-country dataset that spans 113 countries rather than focussing on a particular area or region. Our study focuses on assessing engagement and trending behaviour using important indicators such as unique trending videos, average views, and engagement rate, in contrast to much previous research that highlights comment sentiment analysis or the prediction of individual video popularity. Additionally, the project uses an Apache Spark-based pipeline for scalable data transformation and aggregation, which makes it more appropriate for real-world cloud data analytics scenarios than more straightforward tools or standalone statistical modelling techniques. This allows for the efficient processing of large-scale datasets.

6. Description of the Dataset

6.a Source of the Data

The Trending YouTube Videos – 113 Countries dataset is used in this research and can be found on Kaggle. It gathers information on videos that have been on YouTube trending lists throughout time in 113 different nations. Each record is associated with a certain video, country, and trending date, along with associated information and engagement metrics.

The dataset includes millions of records and several key fields:

- Video identifiers (video_id)
- Titles and channel information
- Category identifiers (category_id)
- Temporal fields (published_at, trending_date)
- Engagement metrics (view_count, like_count, comment_count)
- Geographical attribute (country)

6.b Process of Extraction / Collection

We **downloaded** the dataset from Kaggle as a compressed CSV file and **stored** it in a local project folder for use within our Jupyter + Spark environment (e.g.data/trending_youtube_video_statistics.csv).After that we loaded the CSV into Spark using the PySpark DataFrame API with header recognition and schema inference enabled.

7. Description of Data Processing

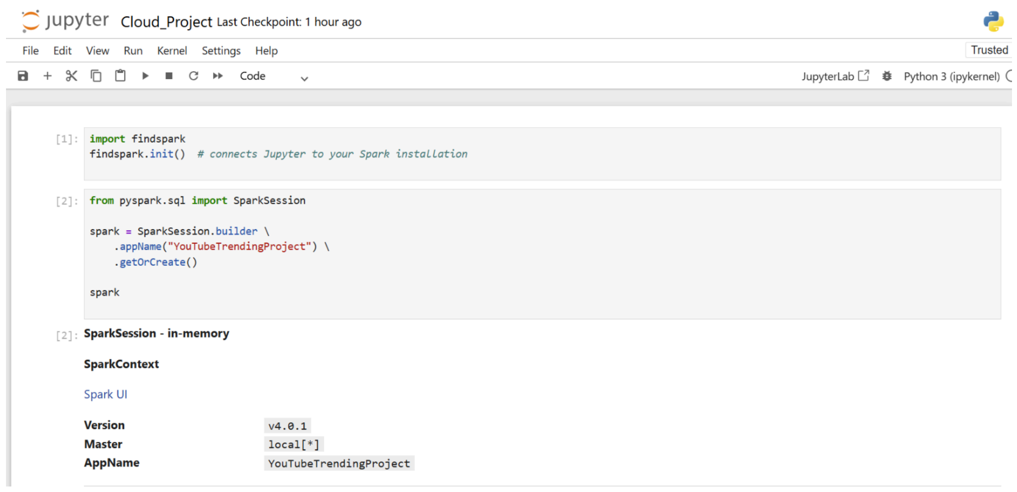
The data processing pipeline was implemented in **PySpark** inside a Jupyter notebook (Cloud_Project.ipynb).

7.a Overview

The major tasks of this data processing pipeline was to clean and standardize the raw YouTube trending dataset , feature engineer a meaningful engagement rate metric based on user interactions, and aggregate the data to present a country-level analytic dataset. The final dataset shows us the key metrics for every country: the number of unique trending videos, the average number of views per video, and the average engagement rate, in an easily comparable and structured cross-country comparison of YouTube trending behaviour.

7.b Loading the Dataset

We first created a Spark session and loaded the CSV file:

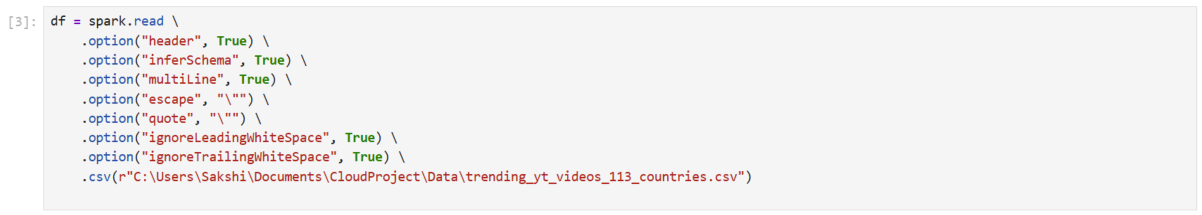


jupyter Cloud_Project Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[1]: import findspark
     findspark.init()  # connects Jupyter to your Spark installation
```

```
[2]: from pyspark.sql import SparkSession

     spark = SparkSession.builder \
         .appName("YouTubeTrendingProject") \
         .getOrCreate()

     spark
```

```
[2]: SparkSession - in-memory

     SparkContext

     Spark UI

     Version     v4.0.1
     Master      local[*]
     AppName     YouTubeTrendingProject
```



```
[3]: df = spark.read \
         .option("header", True) \
         .option("inferSchema", True) \
         .option("multiLine", True) \
         .option("escape", "\"") \
         .option("quote", "\"") \
         .option("ignoreLeadingWhiteSpace", True) \
         .option("ignoreTrailingWhiteSpace", True) \
         .csv(r"C:\Users\Sakshi\Documents\CloudProject\Data\trending_yt_videos_113_countries.csv")
```

This step reads the dataset into a distributed DataFrame, allowing us to perform parallel operations across the cluster (or local cores).

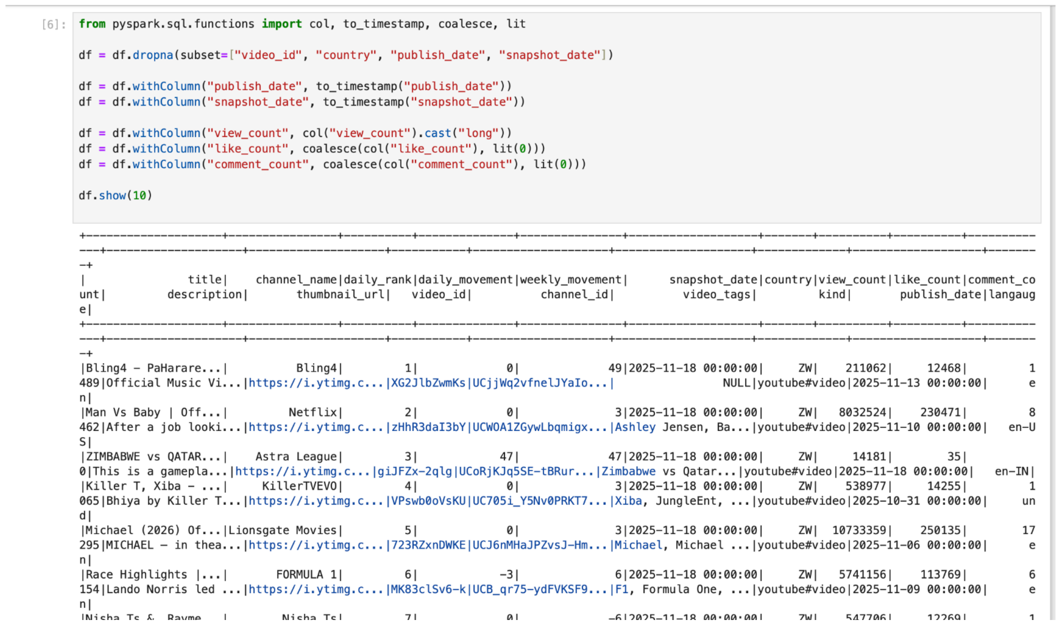
7.c Data Cleaning

The raw dataset contains missing and inconsistent values. We performed several cleaning operations:

1. **Dropping rows with critical nulls**

   Records without video_id, country, published_at, or trending_date were removed, since these fields are essential for grouping and time-based analysis.

2. **Type casting and timestamp conversion**

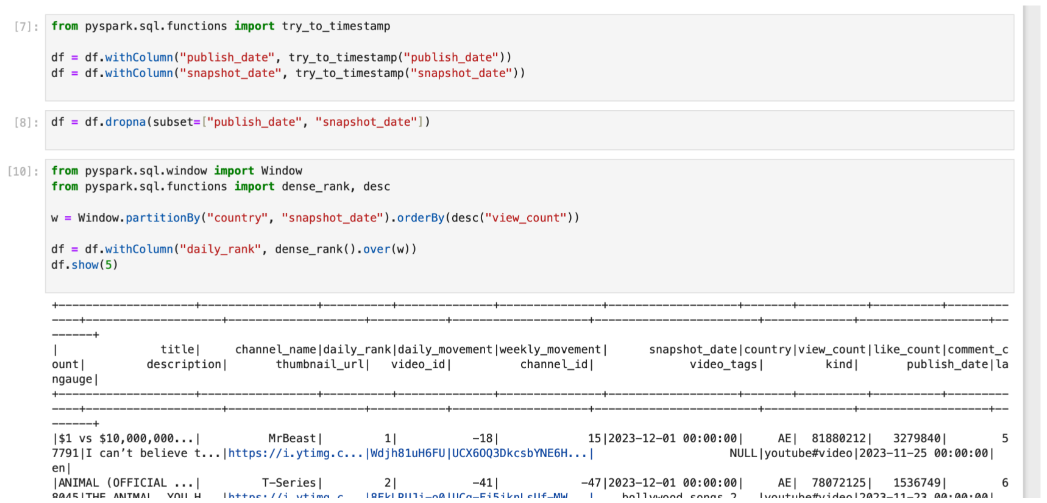


```
[6]: from pyspark.sql.functions import col, to_timestamp, coalesce, lit

     df = df.dropna(subset=["video_id", "country", "publish_date", "snapshot_date"])

     df = df.withColumn("publish_date", to_timestamp("publish_date"))
     df = df.withColumn("snapshot_date", to_timestamp("snapshot_date"))

     df = df.withColumn("view_count", col("view_count").cast("long"))
     df = df.withColumn("like_count", coalesce(col("like_count"), lit(0)))
     df = df.withColumn("comment_count", coalesce(col("comment_count"), lit(0)))

     df.show(10)
```

```
+--------------------+----------------+----------+--------------+---------------+-------------------+-------+----------+----------+----------
---+--------------------+--------------------+-----------+--------------------+--------------------+-------------+-------------------+-------
-+
|               title|    channel_name|daily_rank|daily_movement|weekly_movement|      snapshot_date|country|view_count|like_count|comment_co
unt|         description|       thumbnail_url|   video_id|          channel_id|          video_tags|         kind|       publish_date|langaug
e|
+--------------------+----------------+----------+--------------+---------------+-------------------+-------+----------+----------+----------
---+--------------------+--------------------+-----------+--------------------+--------------------+-------------+-------------------+-------
-+
|Bling4 - PaHarare...|          Bling4|         1|             0|             49|2025-11-18 00:00:00|     ZW|    211062|     12468|         1
489|Official Music Vi...|https://i.ytimg.c...|XG2JlbZwmKs|UCjjWq2vfnelJYaIo...|                NULL|youtube#video|2025-11-13 00:00:00|      e
n|
|Man Vs Baby | Off...|         Netflix|         2|             0|              3|2025-11-18 00:00:00|     ZW|   8032524|    230471|         8
462|After a job looki...|https://i.ytimg.c...|zHhR3daI3bY|UCWOA1ZGywLbqmigx...|Ashley Jensen, Ba...|youtube#video|2025-11-10 00:00:00|   en-U
S|
|ZIMBABWE vs QATAR...|    Astra League|         3|            47|             47|2025-11-18 00:00:00|     ZW|     14181|        35|
0|This is a gamepla...|https://i.ytimg.c...|giJFZx-2qlg|UCoRjKJq5SE-tBRur...|Zimbabwe vs Qatar...|youtube#video|2025-11-18 00:00:00|   en-IN|
|Killer T, Xiba - ...|     KillerTVEVO|         4|             0|              3|2025-11-18 00:00:00|     ZW|    538977|     14255|         1
065|Bhiya by Killer T...|https://i.ytimg.c...|VPswb0oVsKU|UC705i_Y5Nv0PRKT7...|Xiba, JungleEnt, ...|youtube#video|2025-10-31 00:00:00|     un
d|
|Michael (2026) Of...|Lionsgate Movies|         5|             0|              3|2025-11-18 00:00:00|     ZW|  10733359|    250135|        17
295|MICHAEL - in thea...|https://i.ytimg.c...|723RZxnDWKE|UCJ6nMHaJPZvsJ-Hm...|Michael, Michael ...|youtube#video|2025-11-06 00:00:00|      e
n|
|Race Highlights |...|       FORMULA 1|         6|            -3|              6|2025-11-18 00:00:00|     ZW|   5741156|    113769|         6
154|Lando Norris led ...|https://i.ytimg.c...|MK83clSv6-k|UCB_qr75-ydFVKSF9...|F1, Formula One, ...|youtube#video|2025-11-09 00:00:00|      e
n|
|Nisha Ts & Rayme...|        Nisha Ts|         7|             0|             -6|2025-11-18 00:00:00|     ZW|    547706|     12269|         1
```

This ensures that the numeric and time fields have correct types and that missing likes/comments are replaced with zero rather than remaining null.

3. **Basic sanity checks**



```
[7]: from pyspark.sql.functions import try_to_timestamp

     df = df.withColumn("publish_date", try_to_timestamp("publish_date"))
     df = df.withColumn("snapshot_date", try_to_timestamp("snapshot_date"))
```

```
[8]: df = df.dropna(subset=["publish_date", "snapshot_date"])
```

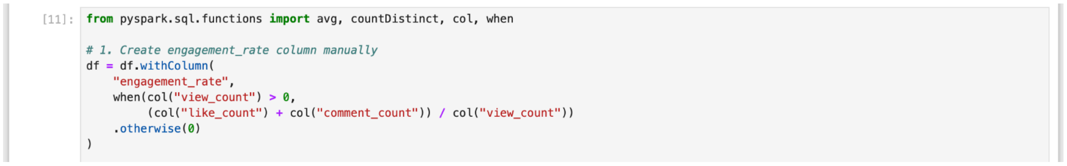
```
[10]: from pyspark.sql.window import Window
      from pyspark.sql.functions import dense_rank, desc

      w = Window.partitionBy("country", "snapshot_date").orderBy(desc("view_count"))

      df = df.withColumn("daily_rank", dense_rank().over(w))
      df.show(5)
```

```
+--------------------+----------------+----------+--------------+---------------+-------------------+-------+----------+----------+---------
----+--------------------+--------------------+-----------+--------------------+----------------------+-------------+-------------------+--
------+
|               title|    channel_name|daily_rank|daily_movement|weekly_movement|      snapshot_date|country|view_count|like_count|comment_c
ount|         description|       thumbnail_url|   video_id|          channel_id|            video_tags|         kind|       publish_date|la
ngauge|
+--------------------+----------------+----------+--------------+---------------+-------------------+-------+----------+----------+---------
----+--------------------+--------------------+-----------+--------------------+----------------------+-------------+-------------------+--
------+
|$1 vs $10,000,000...|        MrBeast|         1|           -18|             15|2023-12-01 00:00:00|     AE|  81880212|   3279840|        5
7791|I can’t believe t...|https://i.ytimg.c...|Wdjh81uH6FU|UCX6OQ3DkcsbYNE6H...|                  NULL|youtube#video|2023-11-25 00:00:00|
en|
|ANIMAL (OFFICIAL ...|        T-Series|         2|           -41|            -47|2023-12-01 00:00:00|     AE|  78072125|   1536749|        6
8045|THE ANIMAL  YOU H...|https://i.ytimg.c...|8FkLRUJi-o0|UCq-Fj5jknLsUf-MW...|     bollywood songs 2...|youtube#video|2023-11-23 00:00:00|
```

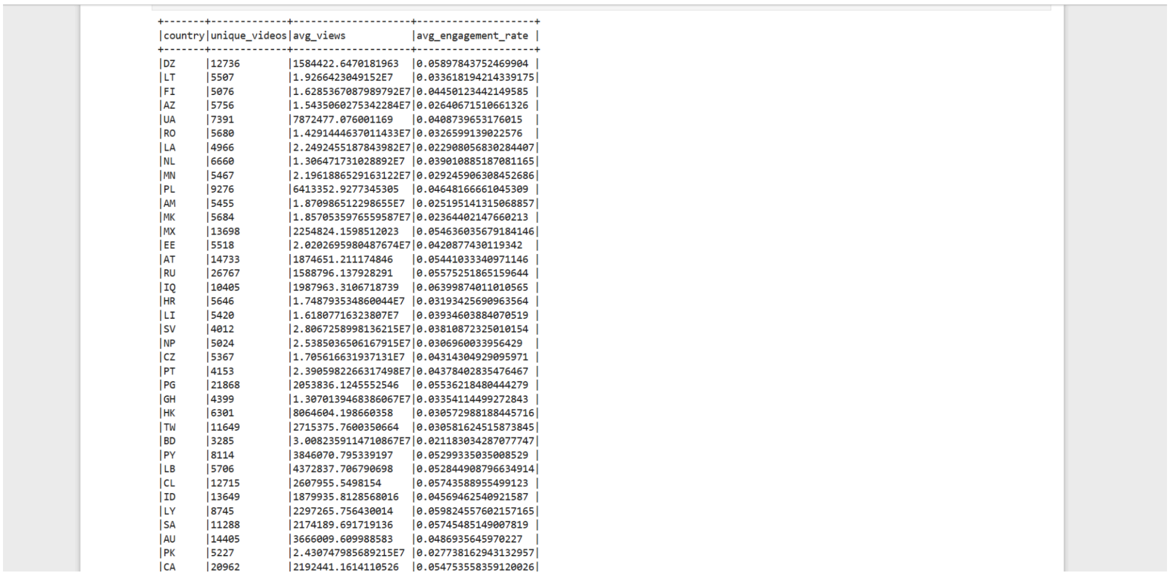
We inspected summary statistics and a few sample rows to confirm that conversions were correct and there were no obvious outliers or broken records.

7.d Feature Engineering

We then engineered an **engagement rate** metric to capture how interactive the audience is relative to views:



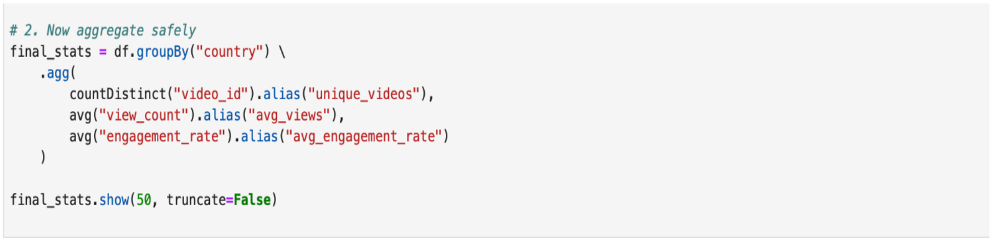
```
[11]: from pyspark.sql.functions import avg, countDistinct, col, when

      # 1. Create engagement_rate column manually
      df = df.withColumn(
          "engagement_rate",
          when(col("view_count") > 0,
               (col("like_count") + col("comment_count")) / col("view_count"))
          .otherwise(0)
      )
```



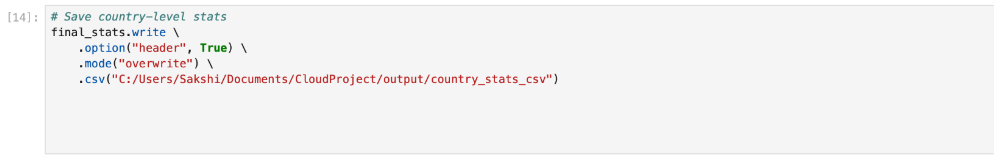
```
+-------+-------------+-------------------+--------------------+
|country|unique_videos|avg_views          |avg_engagement_rate |
+-------+-------------+-------------------+--------------------+
|DZ     |12736        |1584422.6470181963 |0.05897843752469904 |
|LT     |5507         |1.9266423049152E7  |0.033618194214339175|
|FI     |5076         |1.6285367087989792E7|0.04450123442149585 |
|AZ     |5756         |1.5435060275342284E7|0.02640671510661326 |
|UA     |7391         |7872477.076001169  |0.0408739653176015  |
|RO     |5680         |1.4291444637011433E7|0.0326599139022576  |
|LA     |4966         |2.2492455187843982E7|0.022908056830284407|
|NL     |6660         |1.306471731028892E7 |0.039010885187081165|
|MN     |5467         |2.1961886529163122E7|0.029245906308452686|
|PL     |9276         |6413352.9277345305 |0.04648166661045309 |
|AM     |5455         |1.870986512298655E7 |0.025195141315068857|
|MK     |5684         |1.8570535976559587E7|0.0236440214766021З |
|MX     |13698        |2254824.1598512023 |0.054636035679184146|
|EE     |5518         |2.0202695980487674E7|0.0420877430119342  |
|AT     |14733        |1874651.211174846  |0.05441033340971146 |
|RU     |26767        |1588796.137928291  |0.05575251865159644 |
|IQ     |10405        |1987963.3106718739 |0.06399874011010565 |
|HR     |5646         |1.748793534860044E7 |0.03193425690963564 |
|LI     |5420         |1.61807716323807E7  |0.03934603884070519 |
|SV     |4012         |2.8067258998136215E7|0.03810872325010154 |
|NP     |5024         |2.5385036506167915E7|0.0306960033956429  |
|CZ     |5367         |1.705616631937131E7 |0.04314304929095971 |
|PT     |4153         |2.3905982266317498E7|0.04378402835476467 |
|PG     |21868        |2053836.1245552546 |0.05536218480444279 |
|GH     |4399         |1.3070139468386067E7|0.03354114499272843 |
|HK     |6301         |8064604.198660358  |0.030572988188445716|
|TW     |11649        |2715375.7600350664 |0.030581624515873845|
|BD     |3285         |3.0082359114710867E7|0.021183034287077747|
|PY     |8114         |3846070.795339197  |0.05299335035008529 |
|LB     |5706         |4372837.706790698  |0.052844908796634914|
|CL     |12715        |2607955.5498154    |0.05743588955499123 |
|ID     |13649        |1879935.8128568016 |0.04569462540921587 |
|LY     |8745         |2297265.756430014  |0.059824557602157165|
|SA     |11288        |2174189.691719136  |0.05745485149007819 |
|AU     |14405        |3666009.609988583  |0.0486935645970227  |
|PK     |5227         |2.430747985689215E7 |0.027738162943132957|
|CA     |20962        |2192441.1614110526 |0.054753558359120026|
```

7.e Aggregation to Country-Level Dataset

The cleaned and feature-enhanced DataFrame was then aggregated by country:



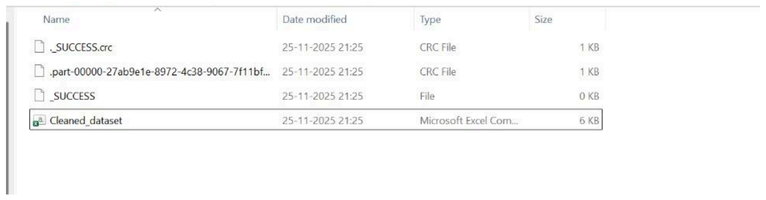
```
# 2. Now aggregate safely
final_stats = df.groupBy("country") \
    .agg(
        countDistinct("video_id").alias("unique_videos"),
        avg("view_count").alias("avg_views"),
        avg("engagement_rate").alias("avg_engagement_rate")
    )

final_stats.show(50, truncate=False)
```



```
[14]: # Save country-level stats
      final_stats.write \
          .option("header", True) \
          .mode("overwrite") \
          .csv("C:/Users/Sakshi/Documents/CloudProject/output/country_stats_csv")
```

7.f Exporting Results

After the country-level aggregation was complete, the resultant Spark DataFrame including the metrics unique_videos, avg_views, and avg_engagement_rate for each country was exported for further analysis and reporting. To enable efficient big-data storage and reuse in Spark-based applications, the results were first documented in Parquet format, which preserves schema information and enables fast column-based access for massive datasets. They were also exported as a CSV file to facilitate the integration of the same data with applications such as Microsoft Excel and Python pandas for post-processing, visualisation, and inclusion in the final report. This dual export approach ensures both scalability for future analytical procedures and accessibility for lightweight data exploration and display.



Name	Date modified	Type	Size
._SUCCESS.crc	25-11-2025 21:25	CRC File	1 KB
.part-00000-27ab9e1e-8972-4c38-9067-7f11bf...	25-11-2025 21:25	CRC File	1 KB
_SUCCESS	25-11-2025 21:25	File	0 KB
Cleaned_dataset	25-11-2025 21:25	Microsoft Excel Com...	6 KB

8. Development of the Pipeline

8.a Data Extraction Tool

The primary data extraction step was downloading the Kaggle dataset and loading it into the Spark environment. No custom web scraping was necessary.

- **Kaggle Web UI** was used to download the initial CSV files.
- The data was then placed in the project's data/ folder and accessed directly from the Jupyter notebook using Spark's CSV reader (as described in Section 7).

8.b Data Processing Tool

The core technology for data processing is **Apache Spark**, accessed through **PySpark** in Jupyter.

Spark Justification

Apache Spark is a unified analytics engine designed for large-scale data processing. It provides:

- In-memory computation for faster iterative operations than disk-based Hadoop MapReduce
- Support for high-level operations such as grouping, joins, aggregations, and user-defined functions
- APIs in Python, making it easy to integrate with data science workflows and Jupyter notebooks

These capabilities make Spark well suited to processing a dataset of millions of YouTube trending records and aggregating them by country.

Jupyter Notebook Environment

Using Jupyter Notebook (Cloud_Project.ipynb) provided the following advantages:

- Interactive experimentation with Spark transformations
- Immediate visual inspection of intermediate results (via df.show() and plots)
- Integration with matplotlib for visualisation of the final metrics

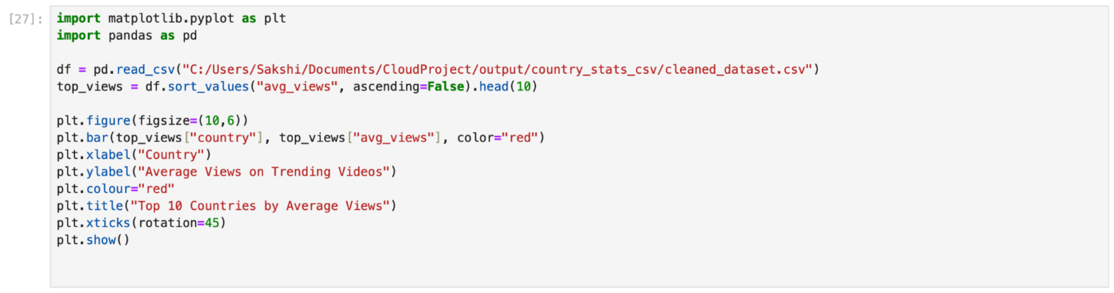
8.c Interface to View Results

The cleaned dataset was saved as a csv file and we also performed a basic interface through visualisations created in Jupyter using **matplotlib** and **pandas**.

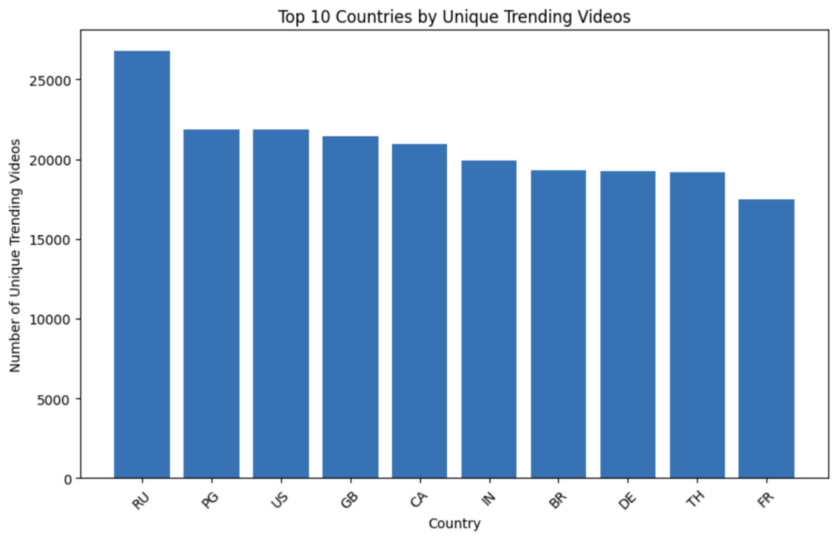
For example:

- A **bar chart** of the top 10 countries by number of unique trending videos

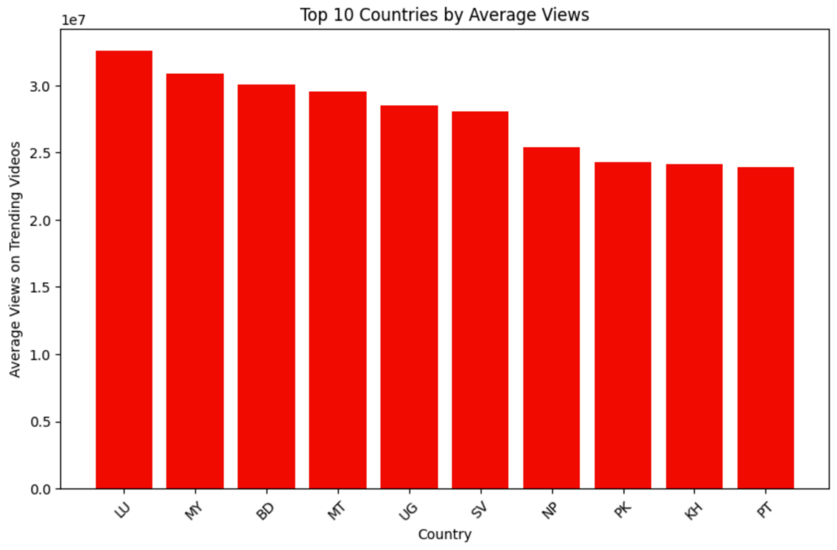
Example code:



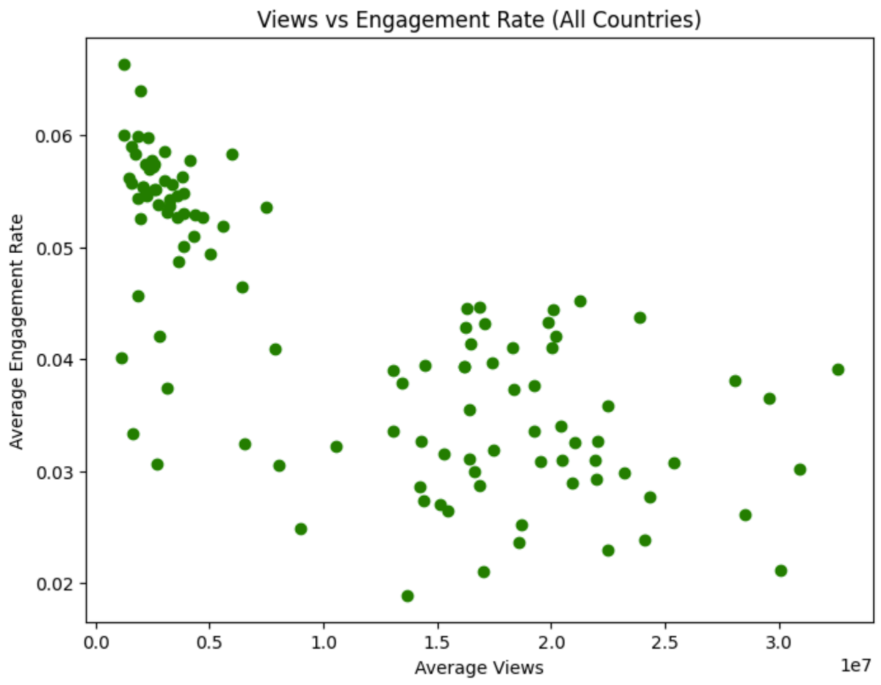
```
[27]: import matplotlib.pyplot as plt
      import pandas as pd

      df = pd.read_csv("C:/Users/Sakshi/Documents/CloudProject/output/country_stats_csv/cleaned_dataset.csv")
      top_views = df.sort_values("avg_views", ascending=False).head(10)

      plt.figure(figsize=(10,6))
      plt.bar(top_views["country"], top_views["avg_views"], color="red")
      plt.xlabel("Country")
      plt.ylabel("Average Views on Trending Videos")
      plt.colour="red"
      plt.title("Top 10 Countries by Average Views")
      plt.xticks(rotation=45)
      plt.show()
```



- A **bar chart** of the top 10 countries by average views



- A **scatter plot** of avg_views vs avg_engagement_rate to show how engagement changes with popularity



These visualisations act as a simple interface for exploring results.

9. Challenges and Lessons Learned

During the development of this project, several challenges were encountered:

1. **Environment and Dependency Issues:**

   Setting up Apache Spark and Java on Windows required careful configuration of JAVA_HOME, Spark versions, and PySpark dependencies. Misalignments between Java and Spark versions led to runtime errors that had to be debugged.

2. **Memory and Performance Constraints:**

   Although Spark is designed for distributed environments, running on a single laptop introduces memory limitations. We learned to avoid unnecessary caching, limit the number of columns when converting to pandas, and perform aggregations early to reduce data volume.

From these challenges, we **learned** that working with cloud frameworks requires us to be very careful with the environment setup and configuration to avoid technical issues during development. We also realize that cleaning the data is an integral part of the exercise before any meaningful analysis can be performed since missing or erroneous values might substantially affect the resulting relations. The combination of Apache Spark with Jupyter Notebook was a very powerful and flexible tool for iteratively building, testing, and validating a large-scale data processing pipeline. Finally, we noticed that engagement metrics show far more interesting and valuable insights than raw view counts alone, which confirms various findings from other domains related to research using YouTube data.

10. Conclusion

This project successfully illustrates the use of Apache Spark for cloud data analytics on a large, real-world YouTube dataset. A pipeline was created that begins with raw Kaggle CSV files, cleans and standardises the data, generates an engagement indicator, and aggregates results at the national level.

The ensuing analytical data indicated significant disparities in unique trending videos, views, and engagement rates between nations. These findings suggest that different regional audiences should be the focus of global content initiatives, and that YouTube trending behaviour is driven by both market and cultural variables.

The pipeline is reusable, and future work might expand the outputs to include time-series trends, category level analysis, and interaction with machine learning models.

11. References

1. C. Dabas, P. Kaur, N. Gulati, and M. Tilak, “Analysis of Comments on YouTube Videos using Hadoop,” 2019 Fifth International Conference on Image Information Processing (ICIIP), pp. 353–358, doi:10.1109/ICIIP47207.2019.8985907.
2. T. I. Qureshi, *Examining Factors That Predict User Engagement on YouTube*, Master’s thesis, Toronto Metropolitan University, 2019. Available: https://rshare.library.torontomu.ca/articles/thesis/Examining_factors_that_predict_user_engagement_on_YouTube/14652858?file=28134651
3. M. Nisa, “Optimizing Prediction of YouTube Video Popularity Using XGBoost,” *Electronics*, vol. 10, no. 23, 2962, 2021. doi:10.3390/electronics10232962.
4. Kaggle, “Trending YouTube Videos – 113 Countries,” accessed 2025.
5. Apache Spark Project, “Apache Spark – A unified analytics engine for large-scale data processing,” Official Documentation.

Contribution

Both team members collaborated together on every important job to conclude this project. As a team, we built the Apache Spark environment, used Jupyter to develop the PySpark data processing pipeline, executed feature engineering and data cleaning, and then created the visualizations. Furthermore we performed together on data gathering, literature evaluation, documentation, report creation, and structuring. All of the major choices, result interpretations, and final outputs have been discussed and executed together as a team.