

## **BDA Mini Project**

### 1. Introduction:

This comprehensive data analytics project focuses on processing Amazon's Digital Music review dataset([Amazon Reviews'23](#)), which contains 130.4K reviews from 101K users across 70.5K items, totaling 11.4M review tokens. The pipeline implements a sophisticated architecture that combines AWS services with distributed computing frameworks: AWS S3 handles scalable storage, PySpark manages distributed data processing, AWS SageMaker powers machine learning capabilities, and QuickSight delivers data visualization. The dataset captures detailed customer interactions through a rich set of features including numerical ratings (1.0-5.0), timestamped reviews, purchase verification status, helpfulness votes, and review text. This structured approach enables efficient large-scale data processing while providing valuable insights into customer behavior and preferences in the digital music marketplace.

Example entry of Dataset: Digital\_Music.jsonl

```
{ "rating": 5.0, "title": "Nice", "text": "If i had a dollar for how many times I have played this cd and how many times I have asked Alexa to play it, I would be rich. Love this singer along with the Black Pumas. Finding a lot of new music that I like a lot on amazon. Try new things.", "images": [], "asin": "B004RQ2IRG", "parent_asin": "B004RQ2IRG", "user_id": "AFUOYIZBU3MTBOLYKOJE5Z35MBDA", "timestamp": 1618972613292, "helpful_vote": 0, "verified_purchase": true }
```

### 2. Environment Setup

1. AWS S3 for Data Storage:
  - a. Step 1 : Create an S3 bucket to store data.

Name: Harsh Chauhan  
Username: hchauhan

General purpose buckets (1) Info All AWS Regions

Copy ARN

Empty

Delete

Create bucket

Buckets are containers for data stored in S3.

Find buckets by name

< 1 >

	Name	AWS Region	IAM Access Analyzer	Creation date
<input type="radio"/>	<a href="#">bda-miniproject-hchauhan</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	December 5, 2024, 23:09:07 (UTC-05:00)

b. Step 2: Upload the raw dataset to the S3 bucket.

Uploaded Digital\_music.jsonl to bda-miniproject-hchauhan S3 bucket

Summary

Destination

s3://bda-miniproject-hchauhan

Succeeded

1 file, 75.2 MB (100.00%)

Failed

0 files, 0 B (0%)

Files and folders

Configuration

Files and folders (1 total, 75.2 MB)

Find by name

< 1 >

Name	Folder	Type	Size	Status	Error
<a href="#">Digital_Music.jsonl</a>	-	-	75.2 MB	Succeeded	-

Object overview

Owner

harshchauhan02001

AWS Region

US East (N. Virginia) us-east-1

Last modified

December 5, 2024, 23:11:36 (UTC-05:00)

Size

75.2 MB

Type

jsonl

Key

Digital\_Music.jsonl

S3 URI

s3://bda-miniproject-hchauhan/Digital\_Music.jsonl

Amazon Resource Name (ARN)

arn:aws:s3:::bda-miniproject-hchauhan/Digital\_Music.jsonl

Entity tag (Etag)

7200a874b8088a784552d85c2c483098-5

Object URL

[https://bda-miniproject-hchauhan.s3.us-east-1.amazonaws.com/Digital\\_Music.jsonl](https://bda-miniproject-hchauhan.s3.us-east-1.amazonaws.com/Digital_Music.jsonl)

Name: Harsh Chauhan  
Username: hchauhan

## 2. Linux Environment with PySpark

### a. Step 1 : Setting up Ubuntu EC2 Instance.

The screenshot displays the AWS Management Console interface for an EC2 instance. At the top, the 'Instances' section shows a list of instances, with the selected instance 'bda\_hchauhan\_ec2' (ID: i-093d04adeb2c38d4e) in a 'Running' state. Below this, the 'Details' tab is active, providing a comprehensive overview of the instance's configuration and status.

**Instance summary**

Property	Value
Instance ID	i-093d04adeb2c38d4e
Public IPv4 address	34.205.140.233   <a href="#">open address</a>
Private IPv4 addresses	172.31.90.160
IPv6 address	—
Instance state	Running
Public IPv4 DNS	ec2-34-205-140-233.compute-1.amazonaws.com   <a href="#">open address</a>
Hostname type	IP name: ip-172-31-90-160.ec2.internal
Private IP DNS name (IPv4 only)	ip-172-31-90-160.ec2.internal
Answer private resource DNS name	IPV4 (A)
Instance type	t2.micro
Auto-assigned IP address	34.205.140.233 [Public IP]
VPC ID	vpc-0e536a0b7a0681127
IAM Role	—
Subnet ID	subnet-007b94994d97e0a9e
IMDSv2	Required
Instance ARN	arn:aws:ec2:us-east-1:183631311803:instance/i-093d04adeb2c38d4e
Operator	—
Elastic IP addresses	—
AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations.   <a href="#">Learn more</a>
Auto Scaling Group name	—
Managed	false

Name: Harsh Chauhan  
Username: hchauhan

## Established SSH connection:

```
ubuntu@ip-172-31-90-160: ~  
PS D:\Fall 2024\BDA\mini project> icaccls ".\bda_hchauhan_key.pem" /reset  
processed file: .\bda_hchauhan_key.pem  
Successfully processed 1 files; Failed processing 0 files  
PS D:\Fall 2024\BDA\mini project> icaccls ".\bda_hchauhan_key.pem" /inheritance:r /grant:r "$($env:USERNAME):(R)"  
processed file: .\bda_hchauhan_key.pem  
Successfully processed 1 files; Failed processing 0 files  
PS D:\Fall 2024\BDA\mini project> ssh -i ".\bda_hchauhan_key.pem" ubuntu@ec2-34-205-140-233.compute-1.amazonaws.com  
The authenticity of host 'ec2-34-205-140-233.compute-1.amazonaws.com (34.205.140.233)' can't be established.  
ED25519 key fingerprint is SHA256:rE9IZoQSW1eY2W/n+77VNnJbC8nhoK3zCDwsTiswJtI.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? Yes  
Warning: Permanently added 'ec2-34-205-140-233.compute-1.amazonaws.com' (ED25519) to the list of known hosts.  
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1018-aws x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/pro  
  
System information as of Fri Dec  6 04:50:15 UTC 2024  
  
System load:  0.0      Processes:      106  
Usage of /:   24.9% of 6.71GB   Users logged in:  0  
Memory usage: 21%      IPv4 address for enX0: 172.31.90.160  
Swap usage:   0%  
  
Expanded Security Maintenance for Applications is not enabled.  
  
0 updates can be applied immediately.  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
Last login: Fri Dec  6 04:39:31 2024 from 18.206.107.28  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-90-160: $
```

b. Step 2: Install PySpark for distributed data processing.

Installing Python on EC2 instance:

Name: Harsh Chauhan  
Username: hchauhan

```
ubuntu@ip-172-31-90-160: ~  
ubuntu@ip-172-31-90-160: $ sudo apt install python3-venv python3-full  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  2to3 blt fontconfig-config fonts-dejavu-core fonts-dejavu-mono fonts-mathjax idle idle-python3.12 javascript-common  
  libfontconfig1 libjs-jquery libjs-mathjax libjs-underscore libpython3.12-minimal libpython3.12-stdlib  
  libpython3.12-testsuite libpython3.12t64 libtk8.6 libxft2 libxrender1 libxss1 net-tools python3-doc python3-examples  
  python3-lib2to3 python3-pip-whl python3-setuptools-whl python3-tk python3.12 python3.12-doc python3.12-examples  
  python3.12-full python3.12-minimal python3.12-venv tk8.6-blt2.5 x11-common  
Suggested packages:  
  blt-demo apache2 | lighttpd | httpd fonts-mathjax-extras fonts-stix libjs-mathjax-doc tk8.6 python3-dev tix  
  python3-tk-dbg binutils python3.12-dev binfmt-support  
The following NEW packages will be installed:  
  2to3 blt fontconfig-config fonts-dejavu-core fonts-dejavu-mono fonts-mathjax idle idle-python3.12 javascript-common  
  libfontconfig1 libjs-jquery libjs-mathjax libjs-underscore libpython3.12-testsuite libtk8.6 libxft2 libxrender1  
  libxss1 net-tools python3-doc python3-examples python3-full python3-lib2to3 python3-pip-whl python3-setuptools-whl  
  python3-tk python3-venv python3.12-doc python3.12-examples python3.12-full python3.12-venv tk8.6-blt2.5 x11-common  
The following packages will be upgraded:  
  libpython3.12-minimal libpython3.12-stdlib libpython3.12t64 python3.12 python3.12-minimal  
5 upgraded, 33 newly installed, 0 to remove and 35 not upgraded.  
Need to get 40.3 MB of archives.  
After this operation, 163 MB of additional disk space will be used.  
Do you want to continue? [Y/n] Y  
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libpython3.12t64 amd64 3.12.3-1ubuntu0.3 [2333 kB]  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 python3.12 amd64 3.12.3-1ubuntu0.3 [651 kB]  
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libpython3.12-stdlib amd64 3.12.3-1ubuntu0.3 [2068 kB]  
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 python3.12-minimal amd64 3.12.3-1ubuntu0.3 [2333 kB]  
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libpython3.12-minimal amd64 3.12.3-1ubuntu0.3 [834 kB]  
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 python3-lib2to3 all 3.12.3-0ubuntu1 [78.0 kB]  
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 2to3 all 3.12.3-0ubuntu2 [11.0 kB]  
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 fonts-dejavu-mono all 2.37-8 [502 kB]  
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 fonts-dejavu-core all 2.37-8 [835 kB]  
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 fontconfig-config amd64 2.15.0-1.1ubuntu2 [37.3 kB]  
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libfontconfig1 amd64 2.15.0-1.1ubuntu2 [139 kB]  
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libxrender1 amd64 1:0.9.10-1.1build1 [19.0 kB]  
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libxft2 amd64 2.3.6-1build1 [45.3 kB]  
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 x11-common all 1:7.7+23ubuntu3 [21.7 kB]  
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libxss1 amd64 1:1.2.3-1build3 [7204 B]  
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libtk8.6 amd64 8.6.14-1build1 [779 kB]  
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 tk8.6-blt2.5 amd64 2.5.3+dfsg-7build1 [630 kB]  
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 blt amd64 2.5.3+dfsg-7build1 [4840 B]
```

Installing PySpark on EC2 instance:

Name: Harsh Chauhan

Username: hchauhan

```
ubuntu@ip-172-31-90-160: ~$ python3 -m venv ~/pyspark_env
ubuntu@ip-172-31-90-160: ~$ source ~/pyspark_env/bin/activate
(pyspark_env) ubuntu@ip-172-31-90-160: ~$ pip3 install pyspark
Collecting pyspark
  Downloading pyspark-3.5.3.tar.gz (317.3 MB)
    Installing build dependencies ... done
    Getting requirements to build wheel ... done
    Preparing metadata (pyproject.toml) ... done
Collecting py4j==0.10.9.7 (from pyspark)
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl.metadata (1.5 kB)
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
    Building wheels for collected packages: pyspark
  Building wheel for pyspark (pyproject.toml) ... done
  Created wheel for pyspark: filename=pyspark-3.5.3-py2.py3-none-any.whl size=317840629 sha256=545a7981a8c9ae25c9a952879
  Stored in directory: /home/ubuntu/.cache/pip/wheels/07/a0/a3/d24c94bf043ab5c7e38c30491199a2a11fef8d2584e6df7fb7
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.7 pyspark-3.5.3
(pyspark_env) ubuntu@ip-172-31-90-160: ~$
```

c. Step 3: Configure AWS CLI to interact with S3 buckets.

Installation of AWS CLI:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\USER> aws --version
aws-cli/2.22.12 Python/3.12.6 Windows/10 exe/AMD64
PS C:\Users\USER>
```

Name: Harsh Chauhan  
Username: hchauhan

### 3. Data Pipeline Implementation:

#### Task 1: Data Ingestion From S3: data-ingestion.py

##### Code:

```
ubuntu@ip-172-31-90-160: ~  
GNU nano 7.2 data-ingestion.py  
from pyspark.sql import SparkSession  
  
# Initialize Spark Session with S3 configurations  
spark = SparkSession.builder \  
    .appName("BDA_MiniProject") \  
    .config("spark.jars.packages", "org.apache.hadoop:hadoop-aws:3.3.4") \  
    .config("spark.hadoop.fs.s3a.access.key", "AKIASVQKH005U2ILKIAE") \  
    .config("spark.hadoop.fs.s3a.secret.key", "vNS6JjPH6y8Y49hXNIVYJhkH1d8g8tRgYQ1T1Uap0") \  
    .config("spark.hadoop.fs.s3a.impl", "org.apache.hadoop.fs.s3a.S3AFileSystem") \  
    .config("spark.hadoop.fs.s3a.aws.credentials.provider", "org.apache.hadoop.fs.s3a.SimpleAWSCredentialsProvider") \  
    .getOrCreate()  
  
# Read data from S3  
s3_path = "s3a://bda-miniproject-hchauhan/Digital_Music.jsonl"  
df = spark.read.json(s3_path)  
  
# Basic Dataset Information  
print("Dataset Overview:")  
print(f"Number of rows: {df.count()}")  
print(f"Number of columns: {len(df.columns)}")  
  
# Display schema  
print("\nDataset Schema:")  
df.printSchema()  
  
# Show sample data  
print("\nSample Data:")  
df.show(5)
```

##### Output:

```
ubuntu@ip-172-31-90-160: ~  
(pyspark_env) ubuntu@ip-172-31-90-160: $ python3 data-ingestion.py  
:: loading settings :: url = jar:file:/home/ubuntu/pyspark_env/lib/python3.12/site-packages/pyspark/jars/ivy-2.5.1.jar!org/apache/ivy/core/settings/ivysettings.xml  
Ivy Default Cache set to: /home/ubuntu/.ivy2/cache  
The jars for the packages stored in: /home/ubuntu/.ivy2/jars  
org.apache.hadoop#hadoop-aws added as a dependency  
:: resolving dependencies :: org.apache.spark#spark-submit-parent-47582e15-e102-445c-bdb5-939231e122e5;1.0  
   confs: [default]  
   found org.apache.hadoop#hadoop-aws;3.3.4 in central  
   found com.amazonaws#aws-java-sdk-bundle;1.12.262 in central  
   found org.wildfly.openssl#wildfly-openssl;1.0.7.Final in central  
:: resolution report :: resolve 548ms :: artifacts dl 20ms  
   :: modules in use:  
   com.amazonaws#aws-java-sdk-bundle;1.12.262 from central in [default]  
   org.apache.hadoop#hadoop-aws;3.3.4 from central in [default]  
   org.wildfly.openssl#wildfly-openssl;1.0.7.Final from central in [default]  
-----  
| conf | modules | artifacts |  
| conf | number | search | dwnlded | evicted | number | dwnlded |  
-----  
| default | 3 | 0 | 0 | 0 | 3 | 0 |  
-----  
:: retrieving :: org.apache.spark#spark-submit-parent-47582e15-e102-445c-bdb5-939231e122e5  
   confs: [default]  
   0 artifacts copied, 3 already retrieved (0kB/14ms)  
24/12/06 23:03:47 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable  
Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  
24/12/06 23:04:05 ERROR Influx: Ignoring error  
org.apache.spark.SparkException: Exception thrown in awaitResult:  
   at org.apache.spark.util.ThreadUtils$.awaitResult(ThreadUtils.scala:56)  
   at org.apache.spark.util.ThreadUtils$.awaitResult(ThreadUtils.scala:310)
```

Name: Harsh Chauhan  
Username: hchauhan

```
24/12/06 23:04:08 WARN MetricsConfig: Cannot locate configuration: tried hadoop-metrics2-s3a-file-system.properties,hadoop-metrics2.properties
Dataset Overview:
Number of rows: 130434
Number of columns: 10

Dataset Schema:
root
|-- asin: string (nullable = true)
|-- helpful_vote: long (nullable = true)
|-- images: array (nullable = true)
|   |-- element: struct (containsNull = true)
|   |   |-- attachment_type: string (nullable = true)
|   |   |-- large_image_url: string (nullable = true)
|   |   |-- medium_image_url: string (nullable = true)
|   |   |-- small_image_url: string (nullable = true)
|-- parent_asin: string (nullable = true)
|-- rating: double (nullable = true)
|-- text: string (nullable = true)
|-- timestamp: long (nullable = true)
|-- title: string (nullable = true)
|-- user_id: string (nullable = true)
|-- verified_purchase: boolean (nullable = true)

Sample Data:
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| asin|helpful_vote|images|parent_asin|rating|text|timestamp|title|user_id|verified_purchase|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|B004RQ2IRG|0|[[]|B004RQ2IRG|5.0|If i had a dollar...|1618972613292|Nice|AFUOYIZBU3MTBOLYK...|true|
|B0026UZEI0|0|[[]|B0026UZEI0|5.0|awesome sound - c...|1308167525000|Excellent|AHGAOIZVODNHMYMNCB...|true|
|B0055JSYHC|0|[[]|B0055JSYHC|5.0|This is a great c...|1615838793006|Great service|AFGEM6BXCYHUILEOA...|true|
|B000F9SMUQ|0|[[]|B000F9SMUQ|1.0|These are not rea...|1405219741000|No good|AH30G6QD6EDJGZRV...|true|
|B0049D1WVK|0|[[]|B0049D1WVK|3.0|I first heard thi...|1309029595000|Cool concept, so-...|AFW2PDT3AMT4X3PYQ...|false|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows

(pyspark_env) ubuntu@ip-172-31-90-160: $
```

## Task 2: Data Processing With PySpark.

### 1) Data Transformation: Year , Month Transformation year\_month\_transformation.py

Code:

```
ubuntu@ip-172-31-90-160: ~
GNU nano 7.2 year_month_transformation.py
from pyspark.sql import SparkSession
from pyspark.sql.functions import from_unixtime, year, month

# Initialize Spark Session with S3 configurations
spark = SparkSession.builder \
    .appName("BDA_Miniproject") \
    .config("spark.jars.packages", "org.apache.hadoop:hadoop-aws:3.3.4") \
    .config("spark.hadoop.fs.s3a.access.key", "AKIASVQKH005U2ILKIAE") \
    .config("spark.hadoop.fs.s3a.secret.key", "vMS6jPH6y8Y49hXNVYJhkH1d8g8tRgYQ1T1Uap0") \
    .config("spark.hadoop.fs.s3a.impl", "org.apache.hadoop.fs.s3a.S3AFileSystem") \
    .config("spark.hadoop.fs.s3a.aws.credentials.provider", "org.apache.hadoop.fs.s3a.SimpleAWSCredentialsProvider") \
    .getOrCreate()

# Read data from S3
s3_path = "s3a://bda-miniproject-hchauhan/Digital_Music.jsonl"
df = spark.read.json(s3_path)

# Transform timestamp to year and month
df_transformed = df \
    .withColumn("review_date", from_unixtime(df.timestamp/1000)) \
    .withColumn("year", year("review_date")) \
    .withColumn("month", month("review_date"))

# Select columns excluding 'images'
df_transformed_simple = df_transformed.select(
    'asin', 'helpful_vote', 'parent_asin', 'rating', 'text',
    'timestamp', 'title', 'user_id', 'verified_purchase',
    'review_date', 'year', 'month'
)

# Show transformed data
print("Transformed Data Sample:")
df_transformed_simple.show(5)

# Write to CSV
df_transformed_simple.write.option("header", "true").csv("transformed_data", mode="overwrite")
```



Name: Harsh Chauhan  
Username: hchauhan

## Output:

```
(pyspark_env) ubuntu@ip-172-31-90-160: $ (pyspark_env) ubuntu@ip-172-31-90-160: $ python3 year_month_transformation.py
:: loading settings :: url = jar:file:/home/ubuntu/pyspark_env/lib/python3.12/site-packages/pyspark/jars/ivy-2.5.1-jar!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /home/ubuntu/.ivy2/cache
The jars for the packages stored in: /home/ubuntu/.ivy2/jars
org.apache.hadoop#hadoop-aws added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-d5fa02bf-d638-481d-92da-b3ed6b6ec70d;1.0
  confs: [default]
    found org.apache.hadoop#hadoop-aws;3.3.4 in central
    found com.amazonaws#aws-java-sdk-bundle;1.12.262 in central
    found org.wildfly.openssl#wildfly-openssl;1.0.7.Final in central
:: resolution report :: resolve 571ms :: artifacts dl 25ms
  :: modules in use:
    com.amazonaws#aws-java-sdk-bundle;1.12.262 from central in [default]
    org.apache.hadoop#hadoop-aws;3.3.4 from central in [default]
    org.wildfly.openssl#wildfly-openssl;1.0.7.Final from central in [default]
  -----
  | conf | number | modules | search | dwnlded | evicted | artifacts |
  -----+-----+-----+-----+-----+-----+-----
  | default | 3 | 0 | 0 | 0 | 0 | 3 | 0 |
  -----
:: retrieving :: org.apache.spark#spark-submit-parent-d5fa02bf-d638-481d-92da-b3ed6b6ec70d
  confs: [default]
  0 artifacts copied, 3 already retrieved (0kB/13ms)
24/12/06 23:24:59 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/12/06 23:25:19 WARN MetricsConfig: Cannot locate configuration: tried hadoop-metrics2-s3a-file-system.properties,hadoop-metrics2.properties
Transformed Data Sample:
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| asin|helpful_vote|parent_asin|rating| text | timestamp | title | user_id|verified_purchase| review_date|year|month|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|B004RQ2IRG|0|B004RQ2IRG|5.0|If i had a dollar...|1618972613292|Nice|AFUOYIZBU3MTBOLYK...|true|2021-04-21 02:36:53|2021|4|
|B0026UZEI0|0|B0026UZEI0|5.0|awesome sound - c...|1308167525000|Excellent|AHGAOTZVODNHMYMICB...|true|2011-06-15 19:52:05|2011|6|
|B00553SYHC|0|B00553SYHC|5.0|This is a great c...|1615838793006|Great service|AFGEM68XCVHUILEOA...|true|2021-03-15 20:06:33|2021|3|
|B000F9SMUQ|0|B000F9SMUQ|1.0|These are not rea...|1405219741000|No good|AH30G6QD6ED1GZRV...|true|2014-07-13 02:49:01|2014|7|
|B0049D1WK|0|B0049D1WK|3.0|I first heard thi...|1309029595000|Cool concept, so-...|AFW2PDT3AMT4X3PVQ...|false|2011-06-25 19:19:55|2011|6|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows
```

## 2) Data Aggregation:

The 5 key Metric performed are as follows:

- **Rating Distribution by Year**  
Shows the average ratings and total review count for each year, helping track how product satisfaction has evolved over time. This metric reveals long-term trends in customer satisfaction.
- **Helpful Vote Analysis by Rating**  
Analyzes the relationship between rating scores (1-5) and helpful votes, showing which ratings tend to be considered most helpful by other users. This helps understand community engagement with different types of reviews.
- **Verified Purchase Impact**  
Compares ratings between verified and non-verified purchases across years, helping assess the credibility and potential bias in reviews. The sample shows both true/false verification status.
- **Monthly Review Volume**

Name: Harsh Chauhan  
Username: hchauhan

Tracks the number of reviews submitted each month across different years, revealing seasonal patterns and overall review activity trends.

- Top Reviewers Analysis

Identifies the most active reviewers based on review count, average rating, and helpful votes, helping understand user engagement patterns.

## Code:

```
ubuntu@ip-172-31-90-160: ~  
GNU nano 7.2 data_aggregation.py  
from pyspark.sql import SparkSession  
from pyspark.sql.functions import from_unixtime, year, month, avg, count, desc  
  
spark = SparkSession.builder \  
    .appName("BDA_MiniProject") \  
    .config("spark.jars.packages", "org.apache.hadoop:hadoop-aws:3.3.4") \  
    .config("spark.hadoop.fs.s3a.access.key", "AKIASVQKH005U2ILKIAE") \  
    .config("spark.hadoop.fs.s3a.secret.key", "vMS63jPH6y8Y49hXNVYJhkH1d8g8tRgYQ1T1Uap0") \  
    .config("spark.hadoop.fs.s3a.impl", "org.apache.hadoop.fs.s3a.S3AFileSystem") \  
    .config("spark.hadoop.fs.s3a.aws.credentials.provider", "org.apache.hadoop.fs.s3a.SimpleAWSCredentialsProvider") \  
    .getOrCreate()  
  
# Read data from S3  
df = spark.read.json("s3a://bda-miniproject-hchauhan/Digital_Music.jsonl")  
df_transformed_simple = spark.read.csv("transformed_data", header=True)  
  
# 1. Rating Distribution by Year  
yearly_ratings = df_transformed_simple.groupBy("year") \  
    .agg(avg("rating").alias("avg_rating"), count("*").alias("review_count")) \  
    .orderBy("year")  
print("\n1. Rating Distribution by Year:")  
yearly_ratings.show()  
  
# 2. Helpful Vote Analysis by Rating  
helpful_votes_analysis = df_transformed_simple.groupBy("rating") \  
    .agg(avg("helpful_vote").alias("avg_helpful_votes"), count("*").alias("total_reviews")) \  
    .orderBy("rating")  
print("\n2. Helpful Vote Analysis by Rating:")  
helpful_votes_analysis.show()  
  
# 3. Verified Purchase Impact  
verified_impact = df_transformed_simple.groupBy("verified_purchase", "year") \  
    .agg(avg("rating").alias("avg_rating"), count("*").alias("review_count")) \  
    .orderBy("year", "verified_purchase")  
print("\n3. Verified Purchase Impact by Year:")  
verified_impact.show()  
  
# 4. Monthly Review Volume  
monthly_volume = df_transformed_simple.groupBy("year", "month") \  
    .agg(count("*").alias("review_count")) \  
    .orderBy("year", "month")  
print("\n4. Monthly Review Volume:")  
monthly_volume.show()  
  
# 5. Top Reviewers Analysis  
top_reviewers = df_transformed_simple.groupBy("user_id") \  
    .agg(count("*").alias("review_count"),  
         avg("rating").alias("avg_rating"),  
         avg("helpful_vote").alias("avg_helpful_votes")) \  
    .orderBy(desc("review_count")) \  
    .limit(10)  
  
# 5. Top Reviewers Analysis  
top_reviewers = df_transformed_simple.groupBy("user_id") \  
    .agg(count("*").alias("review_count"),  
         avg("rating").alias("avg_rating"),  
         avg("helpful_vote").alias("avg_helpful_votes")) \  
    .orderBy(desc("review_count")) \  
    .limit(10)  
print("\n5. Top Reviewers Analysis:")  
top_reviewers.show()
```

Name: Harsh Chauhan  
Username: hchauhan

Output:

### 1) Rating Distribution By Year:

```
1. Rating Distribution by Year:
+-----+-----+-----+
|year|      avg_rating|review_count|
+-----+-----+-----+
|1997|          5.0|         2|
|1998| 4.66666666666667|        21|
|1999| 4.5256410256410255|        78|
|2000| 4.346153846153846|       416|
|2001| 4.275773195876289|       388|
|2002| 3.9575757575757575|       660|
|2003| 4.368852459016393|       244|
|2004| 4.4534005037783375|       397|
|2005| 4.404150197628459|      1012|
|2006| 4.477292965271594|      1123|
|2007| 4.454377311960543|      1622|
|2008| 4.434092112228693|      1889|
|2009| 4.4973776223776225|      2288|
|2010| 4.508384146341464|      2624|
|2011| 4.412309530923215|      3347|
|2012| 4.518949771689497|      4380|
|2013| 4.526229315808004|      8521|
|2014| 4.526788741946422|     11796|
|2015| 4.614016736401673|     14340|
|2016| 4.578699840703937|     13183|
+-----+-----+-----+
only showing top 20 rows
```

### 2) Helpful Vote Analysis by Rating:

```
2. Helpful Vote Analysis by Rating:
+-----+-----+-----+
|rating| avg_helpful_votes|total_reviews|
+-----+-----+-----+
| 1.0|1.6851368970013039|        6136|
| 2.0|1.4713516935739157|        3159|
| 3.0|1.1760012515644556|        6392|
| 4.0|1.3267039422464435|       14129|
| 5.0|0.9447613747043273|      100618|
+-----+-----+-----+
```

### 3) Verified Purchase Impact by Year:

Name: Harsh Chauhan  
Username: hchauhan

### 3. Verified Purchase Impact by Year:

verified_purchase	year	avg_rating	review_count
false	1997	5.0	2
false	1998	4.65	20
true	1998	5.0	1
false	1999	4.51948051948052	77
true	1999	5.0	1
false	2000	4.338383838383838	396
true	2000	4.5	20
false	2001	4.254794520547946	365
true	2001	4.608695652173913	23
false	2002	3.949675324675325	616
true	2002	4.068181818181818	44
false	2003	4.358078602620087	229
true	2003	4.533333333333333	15
false	2004	4.510695187165775	374
true	2004	3.5217391304347827	23
false	2005	4.41044776119403	938
true	2005	4.324324324324325	74
false	2006	4.503913894324853	1022
true	2006	4.207920792079208	101
false	2007	4.495633187772926	1374

only showing top 20 rows

### 4) Monthly Review Volume:

#### 4. Monthly Review Volume:

year	month	review_count
1997	12	1
1997	9	1
1998	10	2
1998	11	2
1998	12	5
1998	4	1
1998	6	2
1998	7	2
1998	8	3
1998	9	4
1999	1	1
1999	10	11
1999	11	13
1999	12	14
1999	2	3
1999	3	4
1999	4	4
1999	5	6
1999	6	4
1999	7	3

only showing top 20 rows

Name: Harsh Chauhan  
Username: hchauhan

## 5) Top Review Analysis:

```
5. Top Reviewers Analysis:
```

user_id	review_count	avg_rating	avg_helpful_votes
AGAFM74L2RIJ5036N...	341	3.865102639296188	1.6070381231671553
AEDFM4VDH2MKYVBKG...	182	1.043956043956044	1.4725274725274726
AH3FC6V3IUJIN2Y7B...	175	4.9314285714285715	1.4
AEFLICXXHRBMNT4HA...	136	3.8676470588235294	0.8235294117647058
AFMUUMXTKB6C52CCE...	130	4.553846153846154	0.36153846153846153
AEIOEJHOTKH6PKPVP...	126	4.325396825396825	0.5476190476190477
AEBMZ7Q2F2EHZ4CTO...	113	4.584070796460177	0.4336283185840708
AGS3YH5GKIZCIPYFF...	111	5.0	2.9279279279279278
AEKGXJDXUVQKHP55J...	95	4.4526315789473685	0.30526315789473685
AE6CHAOE2U5GBCM34...	83	4.506024096385542	1.1325301204819278

## Task 3: Store Processed Data back to S3:

Step 1: Saving metrics to csv  
file\_to\_csv.py

Code:

Name: Harsh Chauhan  
Username: hchauhan

```
ubuntu@ip-172-31-90-160: ~  
GNU nano 7.2 file_to_csv.py  
from pyspark.sql import SparkSession  
from pyspark.sql.functions import avg, count, desc  
  
# Initialize Spark session  
spark = SparkSession.builder \\\n    .appName("BDA_MiniProject") \\\n    .config("spark.jars.packages", "org.apache.hadoop:hadoop-aws:3.3.4") \\\n    .config("spark.hadoop.fs.s3a.access.key", "AKIASVQKH005U2ILKIAE") \\\n    .config("spark.hadoop.fs.s3a.secret.key", "vNS6JjPH6y8Y49hXNVYJhkH1d8g8tRgYQ1T1Uap0") \\\n    .config("spark.hadoop.fs.s3a.impl", "org.apache.hadoop.fs.s3a.S3AFileSystem") \\\n    .config("spark.hadoop.fs.s3a.aws.credentials.provider", "org.apache.hadoop.fs.s3a.SimpleAWSCredentialsProvider") \\\n    .getOrCreate()  
  
# Read data from S3  
df = spark.read.json("s3a://bda-miniproject-hchauhan/Digital_Music.jsonl")  
df_transformed_simple = spark.read.csv("transformed_data", header=True)  
  
# 1. Rating Distribution by Year  
yearly_ratings = df_transformed_simple.groupBy("year") \\\n    .agg(avg("rating").alias("avg_rating"), count("*").alias("review_count")) \\\n    .orderBy("year")  
yearly_ratings.write.csv('yearly_ratings', header=True, mode='overwrite')  
print("Successfully saved yearly_ratings.csv")  
  
# 2. Helpful Vote Analysis by Rating  
helpful_votes_analysis = df_transformed_simple.groupBy("rating") \\\n    .agg(avg("helpful_vote").alias("avg_helpful_votes"), count("*").alias("total_reviews")) \\\n    .orderBy("rating")  
helpful_votes_analysis.write.csv('helpful_votes_analysis', header=True, mode='overwrite')  
print("Successfully saved helpful_votes_analysis.csv")  
  
# 3. Verified Purchase Impact by Year  
verified_impact = df_transformed_simple.groupBy("verified_purchase", "year") \\\n    .agg(avg("rating").alias("avg_rating"), count("*").alias("review_count")) \\\n    .orderBy("year", "verified_purchase")  
verified_impact.write.csv('verified_impact', header=True, mode='overwrite')  
print("Successfully saved verified_impact.csv")  
  
# 4. Monthly Review Volume  
monthly_volume = df_transformed_simple.groupBy("year", "month") \\\n    .agg(count("*").alias("review_count")) \\\n    .orderBy("year", "month")  
monthly_volume.write.csv('monthly_volume', header=True, mode='overwrite')  
print("Successfully saved monthly_volume.csv")  
  
# 5. Top Reviewers Analysis  
top_reviewers = df_transformed_simple.groupBy("user_id") \\\n    .agg(count("*").alias("review_count"),  
        avg("rating").alias("avg_rating"),  
        avg("helpful_vote").alias("avg_helpful_votes")) \\\n    .orderBy(desc("review_count")) \\\n    .limit(10)  
top_reviewers.write.csv('top_reviewers', header=True, mode='overwrite')  
print("Successfully saved top_reviewers.csv")
```

Output:

Name: Harsh Chauhan  
Username: hchauhan

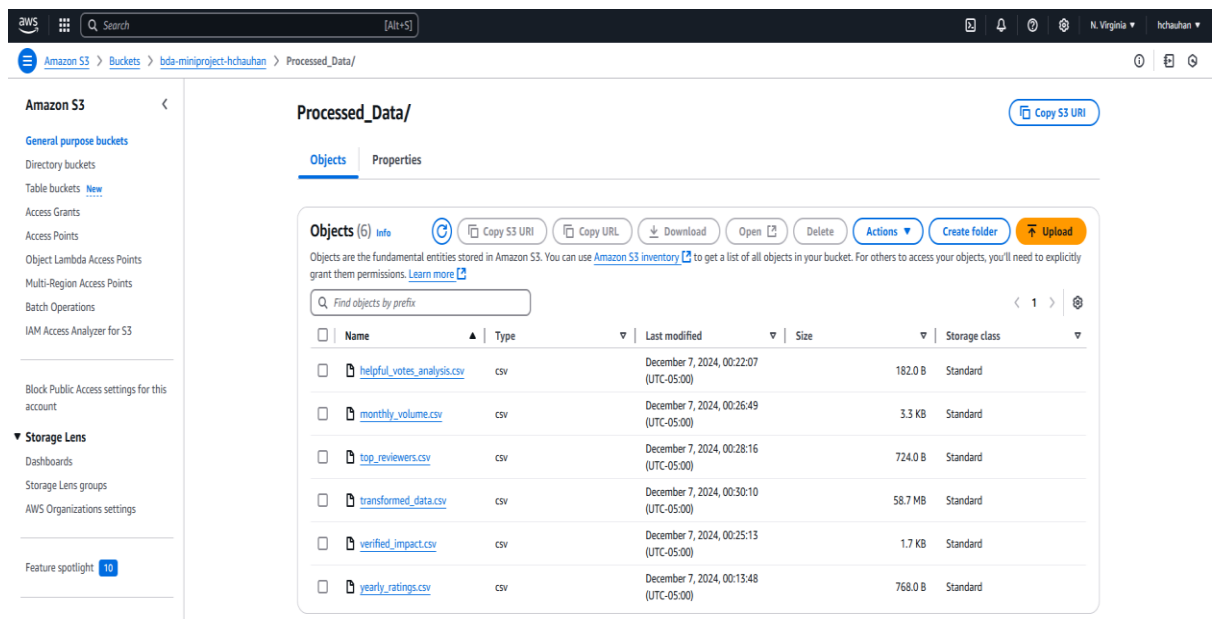
```
(pyspark_env) ubuntu@ip-172-31-90-160: $ (pyspark_env) ubuntu@ip-172-31-90-160:~$ python3 file_to_csv.py
:: loading settings :: url = jar:file:/home/ubuntu/pyspark_env/lib/python3.12/site-packages/pyspark/jars/ivy-2.5.1.jar!
r!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /home/ubuntu/.ivy2/cache
The jars for the packages stored in: /home/ubuntu/.ivy2/jars
org.apache.hadoop#hadoop-aws added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-90048e76-6770-4403-9270-187b9b6f65f9;1.0
  confs: [default]
    found org.apache.hadoop#hadoop-aws;3.3.4 in central
    found com.amazonaws#aws-java-sdk-bundle;1.12.262 in central
    found org.wildfly.openssl#wildfly-openssl;1.0.7.Final in central
:: resolution report :: resolve 609ms :: artifacts dl 23ms
  :: modules in use:
    com.amazonaws#aws-java-sdk-bundle;1.12.262 from central in [default]
    org.apache.hadoop#hadoop-aws;3.3.4 from central in [default]
    org.wildfly.openssl#wildfly-openssl;1.0.7.Final from central in [default]
-----
|               |             modules             || artifacts |
|               | number| search|dwnlded|evicted|| number|dwnlded|
|-----|-----|-----|-----|-----|
| default      | 3    | 0    | 0    | 0    || 3    | 0    |
|-----|-----|-----|-----|-----|
:: retrieving :: org.apache.spark#spark-submit-parent-90048e76-6770-4403-9270-187b9b6f65f9
  confs: [default]
  0 artifacts copied, 3 already retrieved (0kB/12ms)
24/12/07 05:10:35 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/12/07 05:10:57 WARN MetricsConfig: Cannot locate configuration: tried hadoop-metrics2-s3a-file-system.properties,h
adoop-metrics2.properties
Successfully saved yearly_ratings.csv
Successfully saved helpful_votes_analysis.csv
Successfully saved verified_impact.csv
Successfully saved monthly_volume.csv
Successfully saved top_reviewers.csv
```

Moving these csv files to new S3 folder along with transformed\_data:

```
(pyspark_env) ubuntu@ip-172-31-90-160:~$ ls yearly_ratings
SUCCESS part-00000-dd51cbe0-dc1e-4016-a29d-15d202c350e5-c000.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ mv yearly_ratings/part-00000-*.csv yearly_ratings.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ aws s3 cp yearly_ratings.csv s3://bda-miniproject-hchauhan/Processed_Data/
upload: ./yearly_ratings.csv to s3://bda-miniproject-hchauhan/Processed_Data/yearly_ratings.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ mv helpful_votes_analysis/part-00000-*.csv helpful_votes_analysis.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ aws s3 cp helpful_votes_analysis.csv s3://bda-miniproject-hchauhan/Processed
_Data/
upload: ./helpful_votes_analysis.csv to s3://bda-miniproject-hchauhan/Processed_Data/helpful_votes_analysis.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ mv verified_impact/part-00000-*.csv verified_impact.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ aws s3 cp verified_impact.csv s3://bda-miniproject-hchauhan/Processed_Data/
upload: ./verified_impact.csv to s3://bda-miniproject-hchauhan/Processed_Data/verified_impact.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ mv monthly_volume/part-00000-*.csv monthly_volume.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ aws s3 cp monthly_volume.csv s3://bda-miniproject-hchauhan/Processed_Data/
upload: ./monthly_volume.csv to s3://bda-miniproject-hchauhan/Processed_Data/monthly_volume.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ mv top_reviewers/part-00000-*.csv top_reviewers.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ aws s3 cp top_reviewers.csv s3://bda-miniproject-hchauhan/Processed_Data/
upload: ./top_reviewers.csv to s3://bda-miniproject-hchauhan/Processed_Data/top_reviewers.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ mv transformed_data/part-00000-*.csv transformed_data.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$ aws s3 cp transformed_data.csv s3://bda-miniproject-hchauhan/Processed_Data/
upload: ./transformed_data.csv to s3://bda-miniproject-hchauhan/Processed_Data/transformed_data.csv
(pyspark_env) ubuntu@ip-172-31-90-160:~$
```



Name: Harsh Chauhan  
Username: hchauhan



## Task 4 : Data Analysis With Spark SQL:

Code:

```
ubuntu@ip-172-31-90-160: ~  
GNU nano 7.2 sql_analysis.py  
from pyspark.sql import SparkSession  
from pyspark.sql.functions import from_unixtime, year, month  
  
# Initialize Spark Session  
spark = SparkSession.builder \\\n    .appName("BDA_MiniProject") \\\n    .config("spark.jars.packages", "org.apache.hadoop:hadoop-aws:3.3.4") \\\n    .config("spark.hadoop.fs.s3a.access.key", "AKIASVQKH005U2ILKIAE") \\\n    .config("spark.hadoop.fs.s3a.secret.key", "vNS6JjPH6y8Y49hXNVYJhkH1d8g8tRgYQ1T1Uap0") \\\n    .config("spark.hadoop.fs.s3a.impl", "org.apache.hadoop.fs.s3a.S3AFileSystem") \\\n    .config("spark.hadoop.fs.s3a.aws.credentials.provider", "org.apache.hadoop.fs.s3a.SimpleAWSCredentialsProvider") \\\n    .getOrCreate()  
  
# Load the transformed data  
df = spark.read.csv("s3a://bda-miniproject-hchauhan/Processed_Data/transformed_data.csv", header=True)  
  
# Create temporary view for SQL queries  
df.createOrReplaceTempView("digital_music")  
  
# 1. Top-Rated Products Analysis  
query1 = """  
SELECT asin, title,\\\n       AVG(CAST(rating AS DOUBLE)) as avg_rating,\\\n       COUNT(*) as review_count  
FROM digital_music  
GROUP BY asin, title  
HAVING COUNT(*) > 10  
ORDER BY avg_rating DESC  
LIMIT 5  
"""  
  
# 2. Monthly Review Growth Analysis  
query2 = """  
SELECT year, month,\\\n       COUNT(*) as review_count,\\\n       LAG(COUNT(*)) OVER (PARTITION BY year ORDER BY month) as prev_month_count  
FROM digital_music  
GROUP BY year, month  
ORDER BY year, month  
"""
```



Name: Harsh Chauhan  
Username: hchauhan

```
ubuntu@ip-172-31-90-160: ~
GNU nano 7.2 sql_analysis.py
# 3. Verified vs Non-Verified Purchase Impact
query3 = """
SELECT verified_purchase,
       AVG(CAST(rating AS DOUBLE)) as avg_rating,
       COUNT(*) as total_reviews,
       AVG(CAST(helpful_vote AS DOUBLE)) as avg_helpful_votes
FROM digital_music
GROUP BY verified_purchase
"""

# 4. Most Helpful Reviews Analysis
query4 = """
SELECT user_id, rating, title, helpful_vote
FROM digital_music
WHERE CAST(helpful_vote AS INT) > 0
ORDER BY CAST(helpful_vote AS INT) DESC
LIMIT 10
"""

# 5. Yearly Rating Distribution
query5 = """
SELECT year,
       COUNT(*) as total_reviews,
       AVG(CAST(rating AS DOUBLE)) as avg_rating,
       COUNT(CASE WHEN CAST(rating AS DOUBLE) >= 4 THEN 1 END) as positive_reviews
FROM digital_music
GROUP BY year
ORDER BY year
"""

# Execute and display results
print("\nExecuting SQL Queries for Digital Music Analysis:")

queries = [query1, query2, query3, query4, query5]
titles = [
    "Top-Rated Products Analysis",
    "Monthly Review Growth Analysis",
    "Verified vs Non-Verified Purchase Impact",
    "Most Helpful Reviews Analysis",
    "Yearly Rating Distribution"
]

for i, (query, title) in enumerate(zip(queries, titles), 1):
    print(f"\n{i}. {title}:")
    spark.sql(query).show(truncate=False)
```

Name: Harsh Chauhan  
Username: hchauhan

## Output:

```
(pyspark_env) ubuntu@ip-172-31-90-160: $ (pyspark_env) ubuntu@ip-172-31-90-160: $ python3 sql_analysis.py
:: loading settings :: url = jar:file:/home/ubuntu/pyspark_env/lib/python3.12/site-packages/pyspark/jars/ivy-2.5.1.jar!/
org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /home/ubuntu/.ivy2/cache
The jars for the packages stored in: /home/ubuntu/.ivy2/jars
org.apache.hadoop#hadoop-aws added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-ac91b3e4-309f-4293-b835-0ff69073592f;1.0
  confs: [default]
    found org.apache.hadoop#hadoop-aws;3.3.4 in central
    found com.amazonaws#aws-java-sdk-bundle;1.12.262 in central
    found org.wildfly.openssl#wildfly-openssl;1.0.7.Final in central
:: resolution report :: resolve 550ms :: artifacts dl 14ms
  :: modules in use:
    com.amazonaws#aws-java-sdk-bundle;1.12.262 from central in [default]
    org.apache.hadoop#hadoop-aws;3.3.4 from central in [default]
    org.wildfly.openssl#wildfly-openssl;1.0.7.Final from central in [default]
-----
|               | modules                || artifacts |
|   conf        | number| search|dwnlded|evicted|| number|dwnlded|
|-----|-----|-----|-----|-----|
|   default     |    3  |    0  |    0  |    0  ||    3  |    0  |
|-----|-----|-----|-----|-----|
:: retrieving :: org.apache.spark#spark-submit-parent-ac91b3e4-309f-4293-b835-0ff69073592f
  confs: [default]
    0 artifacts copied, 3 already retrieved (0kB/14ms)
24/12/07 06:54:04 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/12/07 06:54:26 WARN MetricsConfig: Cannot locate configuration: tried hadoop-metrics2-s3a-file-system.properties,hado
op-metrics2.properties

Executing SQL Queries for Digital Music Analysis:

1. Top-Rated Products Analysis:
+-----+-----+-----+-----+
|asin      |title      |avg_rating|review_count|
+-----+-----+-----+-----+
|B001EJH4SW|Five Stars|5.0       |30          |
|B0009PUCUE|Five Stars|5.0       |34          |
|B00ZUPPH5S|Five Stars|5.0       |17          |
|B00PG5G5QC|Five Stars|5.0       |15          |
|B00IKM5NZC|Five Stars|5.0       |15          |
+-----+-----+-----+-----+
```

Name: Harsh Chauhan  
Username: hchauhan

## 2. Monthly Review Growth Analysis:

year	month	review_count	prev_month_count
1997	12	1	NULL
1997	9	1	1
1998	10	2	NULL
1998	11	2	2
1998	12	5	2
1998	4	1	5
1998	6	2	1
1998	7	2	2
1998	8	3	2
1998	9	4	3
1999	1	1	NULL
1999	10	11	1
1999	11	13	11
1999	12	14	13
1999	2	3	14
1999	3	4	3
1999	4	4	4
1999	5	6	4
1999	6	4	6
1999	7	3	4

only showing top 20 rows

## 3. Verified vs Non-Verified Purchase Impact:

verified_purchase	avg_rating	total_reviews	avg_helpful_votes
false	4.411412458940147	34401	1.904072556030348
true	4.576333135484677	96033	0.7373298761883935

## 4. Most Helpful Reviews Analysis:

user_id	rating	title	helpful_vote
AFCRT2KE2N5P4VSA7MY74SGRV6AA	2.0	Overpriced for what you get.	259
AGAWJK3Z57FRGCVR5TNMIQIDQ2SA	5.0	Let's set the record straight...	213
AHVIMKECEZDXK5QIGX7KWJPN6DQ	4.0	A Grimm, Grimm tale	191
AGK5K3EAMWHSIPWG2VXJP7W32M7Q	4.0	1991 Gramophone Award Winner	162
AGY3BQJXYGHK7OVCFHCXBCIWPNA	5.0	ANTHOLOGY 1 - MAINLY FOR BEATLES FANS AND COLLECTORS ONLY	160
AFPRDHJPDFDTVOSZKAESF2U56LBQ	5.0	The Clash of Titans	157
AH64C6DXEZN3IMX4W2SDHXXJP5GA	1.0	Where are the "Unleashing" techniques?	148
AH2NK6SZGQXS6NSD4VVP20U4HTWA	5.0	Beyond Awesome!	141
AHSF3JIVAMNDRI7PLZ7373VXISHA	5.0	The Two Origins	140
AFIV4IC7VU6FRI364KGJKBPPXXJFA	4.0	Bodes well for Clooney's future behind the camera!	130

## 5. Yearly Rating Distribution:

year	total_reviews	avg_rating	positive_reviews
1997	2	5.0	2
1998	21	4.666666666666667	20
1999	78	4.5256410256410255	70
2000	416	4.346153846153846	349
2001	388	4.275773195876289	317
2002	660	3.9575757575757575	473
2003	244	4.368852459016393	206
2004	397	4.4534005037783375	345
2005	1012	4.404150197628459	861
2006	1123	4.477292965271594	979
2007	1622	4.454377311960543	1422
2008	1889	4.434092112228693	1641
2009	2288	4.4973776223776225	2013
2010	2624	4.508384146341464	2331
2011	3347	4.412309530923215	2867
2012	4380	4.518949771689497	3868
2013	8521	4.526229315808004	7483
2014	11796	4.526788741946422	10369
2015	14340	4.614016736401673	12915
2016	13183	4.578699840703937	11781

only showing top 20 rows

(pyspark\_env) ubuntu@ip-172-31-90-160: \$

Name: Harsh Chauhan  
Username: hchauhan

## Task 5 : Machine Learning with AWS Sagemaker Autopilot

### Created Domain on SageMaker

[Amazon SageMaker AI](#) > Domains

### Domains Info

In SageMaker AI, a domain is an environment for your team to access SageMaker resources. A domain consists of a list of authorized users and users within a domain can share notebook files and other artifacts with each other. One account can have either one or multiple domains.

Domains (1) Info

Find domain name

< 1 >

Name	Id	Status	Created on	Modified on
<a href="#">QuickSetupDomain-20241207T152898</a>	d-rjefwrhyqwdo	<span>InService</span>	Dec 07, 2024 20:28 UTC	Dec 07, 2024 20:36 UTC

Services  [Alt+S]

N. Virginia hchauhan

### Domain details

Configure and manage the domain.

[Domain settings](#) [User profiles](#) [Space management](#) [App Configurations](#) [Environment](#) [Resources](#)

#### General settings Info

Name	Status	Domain ID
QuickSetupDomain-20241207T152898	<span>Ready</span>	d-rjefwrhyqwdo
Created	Last modified	VPC
Sat Dec 07 2024 15:28:13 GMT-0500 (Eastern Standard Time)	Sat Dec 07 2024 15:36:32 GMT-0500 (Eastern Standard Time)	vpc-0e536a0b7a0681127

#### Domain rules

Visibility of instance and image resources for this domain

Manage rules

Rule type	Application type	Rule action	Resource
Instance type	JupyterLab, Code Editor	Hide	ml.t3.medium

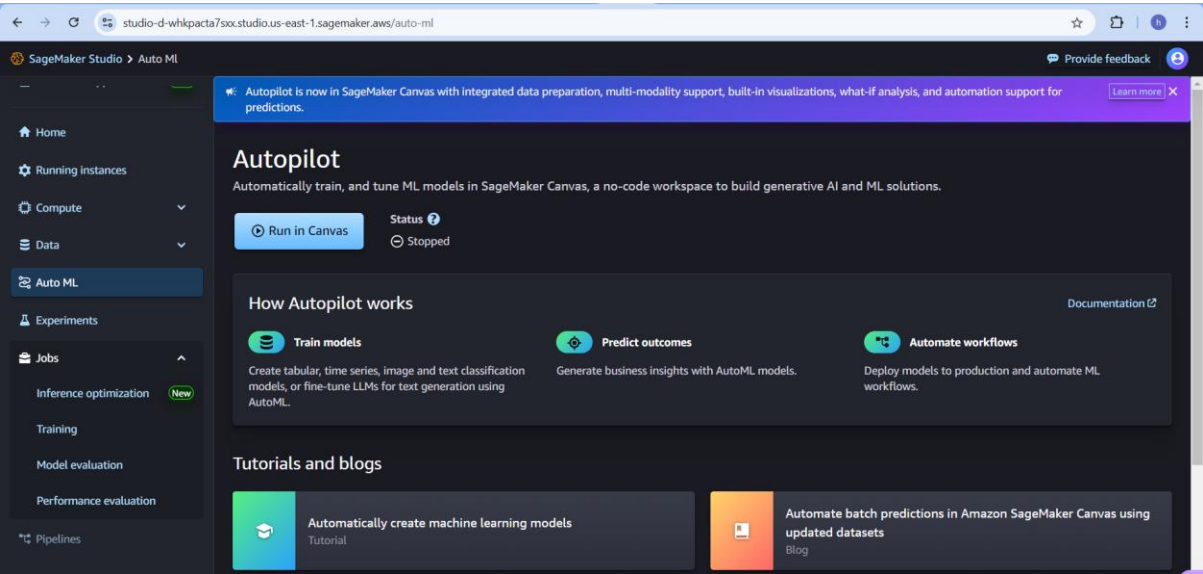
#### Authentication and permissions

Edit

Authentication method	AWS Identity and Access Management (IAM)		
Default execution role	am:awsiam::183631311803:role/service-role/AmazonSageMaker-ExecutionRole-20241207T152898		
Space execution role	am:awsiam::183631311803:role/service-role/AmazonSageMaker-ExecutionRole-20241207T152898		

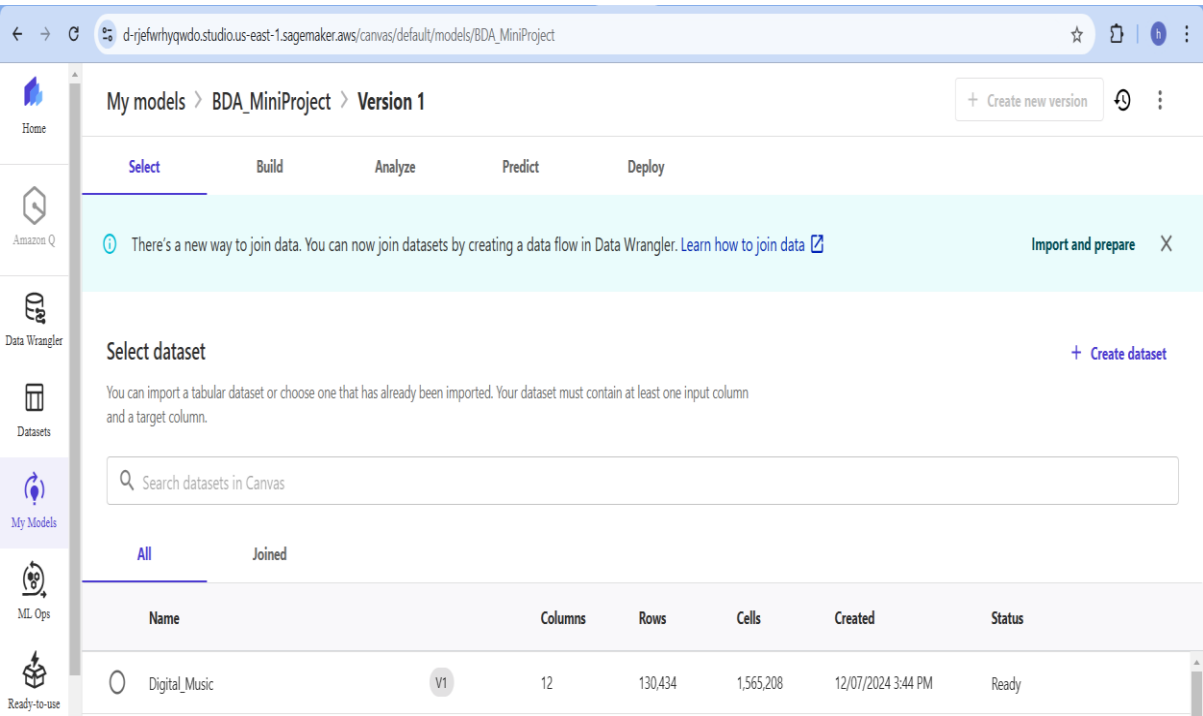
Name: Harsh Chauhan  
Username: hchauhan

Connected to AWS Studio:



Connected to Canvas for AutoML:

Imported Dataset to Canvas



Name: Harsh Chauhan  
Username: hchauhan

## Building Standard Model with target column as Rating

My models > BDA\_MiniProject > Version 1

+ Create new version

🔄

⋮

Select

Build

Analyze

Predict

Deploy

📘 Changes cannot be made while model is building.

View building

✕


Select a column to predict

Choose the target column. The model that you build predicts values for the column that you select.

Target column

rating

Value distribution



Model type

SageMaker Canvas automatically recommends the appropriate model type for your analysis.

3+ category prediction

Your model classifies rating into 3 or more categories.

Configure model

Standard build

Preview model

Digital\_Music

Random sample: 20.0k rows

📊

📈

🔍

🔍

📋

📊 Data visualizer

⬆

Column name ↓	Data type ⓘ	Feature type ⓘ	Missing ⓘ	Mismatched ⓘ	Unique ⓘ	Mode ⓘ
year	Numeric	-	0.00% (985)	0.00% (0)	92	2,015
verified_purchase	Text	Binary	0.00% (1050)	0.00% (0)	2	True

## Configuring Model Setting:

Configure model

🔄 Reset to default settings

✕

Basic

Model type

Advanced - Optional

Objective metric

Training method and algorithms

Data split

Max candidates and runtime

📘 Configuring the Ensemble or Hyperparameter optimization training method will default to Standard build.

☐ Auto 

Recommended

Canvas selects the algorithms that are most relevant to your dataset and the best range of hyperparameters to tune model candidates. The best-performing model candidate is chosen.

☐ Ensemble

☒ Hyperparameter optimization

Algorithms

Select the algorithms that you'd like to test for improving the model's prediction accuracy. 2/2 selected

☒ XGBoost

A supervised learning algorithm that attempts to accurately predict a target variable by combining an

☒ Multilayer Perceptron

A multilayer perceptron (MLP) and feedforward artificial neural network. This algorithm can handle data that is not

Cancel

Save

Name: Harsh Chauhan  
Username: hchauhan

Configure model

Reset to default settings

Basic

Model type

Advanced - Optional

Objective metric

Training method and algorithms

Data split

Max candidates and runtime

Configuring the max candidates and runtime will default to Standard build.

Max candidates

Set the maximum number of model candidates Canvas is allowed to generate. The larger the value, the longer it takes to run the build job.

Max candidates

5

Max job runtime

Set the maximum amount of time that Canvas is allowed to run a build job. If the job exceeds the maximum runtime, it stops automatically.

Hour

2

Min

✓ You are using the recommended max job runtime to optimize your model's accuracy.

Cancel

Save

## Model Training:

My models > BDA\_MiniProject > Version 1

Create new version

Select

Build

Analyze

Predict

Deploy

Model overview

Your model is being created. Standard build usually takes between 2-4 hours. You can now leave this view.

Time elapsed

6 min 4 sec

Expected build time

45 min

Build type

Standard build

Detailed progress

Training models

Digital\_Music

Total columns: 12

Total rows: 130,434

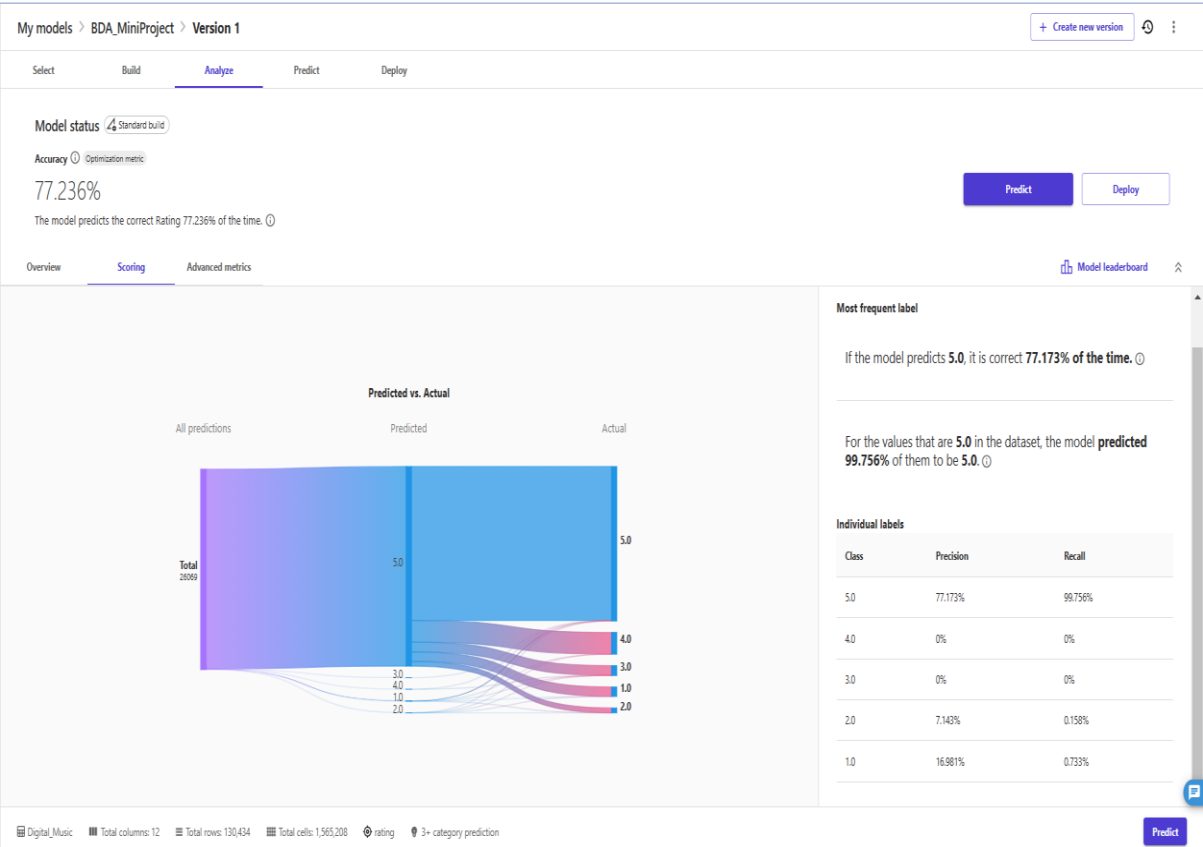
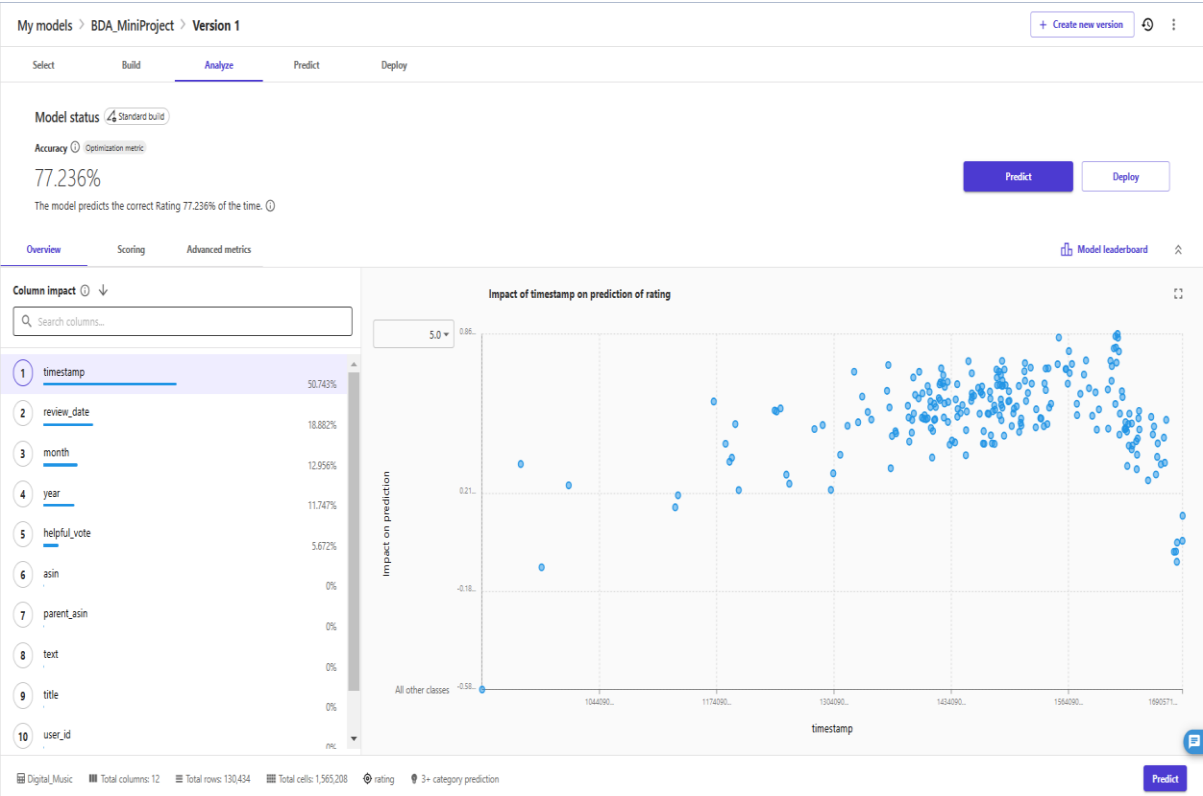
Total cells: 1,565,208

rating

3+ category prediction

Name: Harsh Chauhan  
Username: hchauhan

Model Results:





Name: Harsh Chauhan  
Username: hchauhan

My models > BDA\_MiniProject > Version 1

+ Create new version

Select

Build

Analyze

Predict

Deploy

Model status Standard build

Accuracy Optimization metric

77.236%

The model predicts the correct Rating 77.236% of the time.

Predict

Deploy

Overview

Scoring

Advanced metrics

Model leaderboard

Average accuracy Optimization metric

76.992%

Average f1

17.748%

Average precision

20.259%

Average recall

20.13%

Average AUC-ROC

Not available

Metrics table

Confusion matrix

Metrics table

Metric name	Value
accuracy	0.772
balancedAccuracy	0.205
f1Macro	0.184
precisionMacro	0.569
recallMacro	0.205
logLoss	0.806

Digital\_Music

Total columns: 12

Total rows: 130,434

Total cells: 1,565,208

rating

3+ category prediction

Predict

My models > BDA\_MiniProject > Version 1

+ Create new version

Select

Build

Analyze

Predict

Deploy

Model leaderboard

X

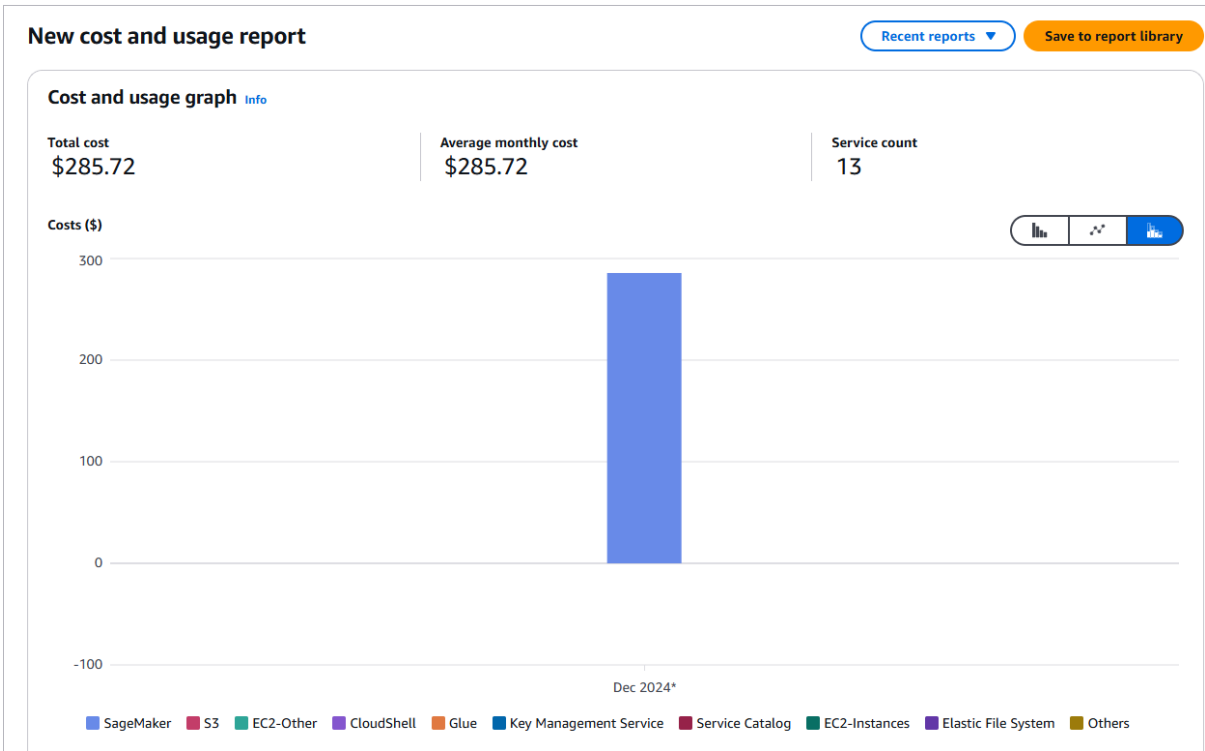
Search leaderboard

Model name	Accuracy	Balanced Accuracy	F1 Macro	Precision Macro	Recall Macro	Log Loss
eU922Qj11V1b-005-807786d5 <span>Default mode</span>	77.236%	20.505%	18.426%	56.905%	20.505%	0.806
eU922Qj11V1b-004-55d83e80	77.251%	20.504%	18.414%	34.820%	20.504%	0.806
eU922Qj11V1b-003-5f08e9fb	77.239%	20.470%	18.350%	34.154%	20.470%	0.806
eU922Qj11V1b-002-46e5cd29	77.239%	20.455%	18.320%	34.754%	20.455%	0.806
eU922Qj11V1b-001-ebc14d63	77.228%	20.406%	18.225%	34.673%	20.406%	0.806

The results show a strong bias towards predicting class 5.0, with 77.173% precision and 99.756% recall for this class. This indicates that the dataset likely has a significant imbalance, with class 5.0 being the dominant category. The model shows very poor performance on other classes (4.0, 3.0, 2.0, 1.0), with 0% precision and recall for most of

them. This further confirms the class imbalance issue and suggests the model struggles to identify less common categories. The advanced metrics show average accuracy (76.992%), F1 score (17.748%), precision (20.259%), and recall (20.13%), which are relatively low, indicating that while the model performs well on the dominant class, it struggles with overall balanced performance across all classes. These results suggest that the dataset has a significant class imbalance issue, and the model has overfit to the majority class (5.0). To improve results, we may need to address the class imbalance, possibly through resampling techniques, class weighting, or collecting more data for underrepresented classes.

Incurred charges for AWS Sagemaker:



Name: Harsh Chauhan  
Username: hchauhan

Delete app

Name

default

Status

InService

Type

Canvas

Created

Sat Dec 07 2024 15:33:42 GMT-0500 (Eastern Standard Time)

User name

default-20241207T152898

This will delete the app. Any data and work that the user has saved to their Studio notebooks and home directory will be unaffected. Unsaved work will be lost. Do you want to proceed?

Yes, delete app

To confirm deletion, type *delete* in the field.

delete

Cancel

Delete

Account snapshot - updated every 24 hours

All AWS Regions

View Storage Lens dashboard

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

General purpose buckets

Directory buckets

General purpose buckets (4)

Info

All AWS Regions

Refresh

Copy ARN

Empty

Delete

Create bucket

Buckets are containers for data stored in S3.

Find buckets by name

< 1 >

Settings

Name	AWS Region	IAM Access Analyzer	Creation date
<div><input type="radio"/></div> <a href="#">bda-miniproject-hchauhan</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	December 5, 2024, 23:09:07 (UTC-05:00)
<div><input type="radio"/></div> <a href="#">sagemaker-studio-183631311803-36mu6weay92</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	December 7, 2024, 14:20:20 (UTC-05:00)
<div><input type="radio"/></div> <a href="#">sagemaker-studio-183631311803-bp1dsrxugn</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	December 7, 2024, 15:28:13 (UTC-05:00)
<div><input type="radio"/></div> <a href="#">sagemaker-us-east-1-183631311803</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	December 7, 2024, 14:20:22 (UTC-05:00)

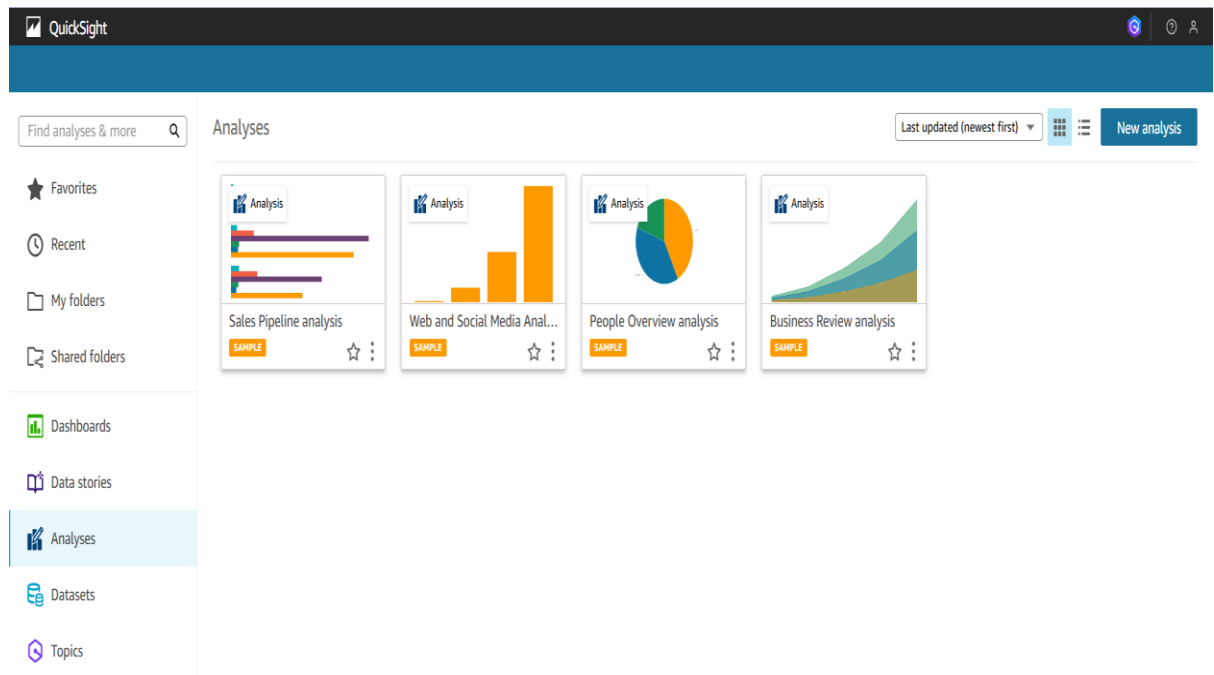
Even though I have set domain rule to ml.t3.medium and modified the model to only use hyperparameter optimization algorithm , setting the number of candidates to 5 and maximum run job time to 2 hours , using memory only till 5gb from 100 gb and still I incurred charges from AWS Sagemaker so I have deleted the Model, Domain, all the S3

Name: Harsh Chauhan  
Username: hchauhan

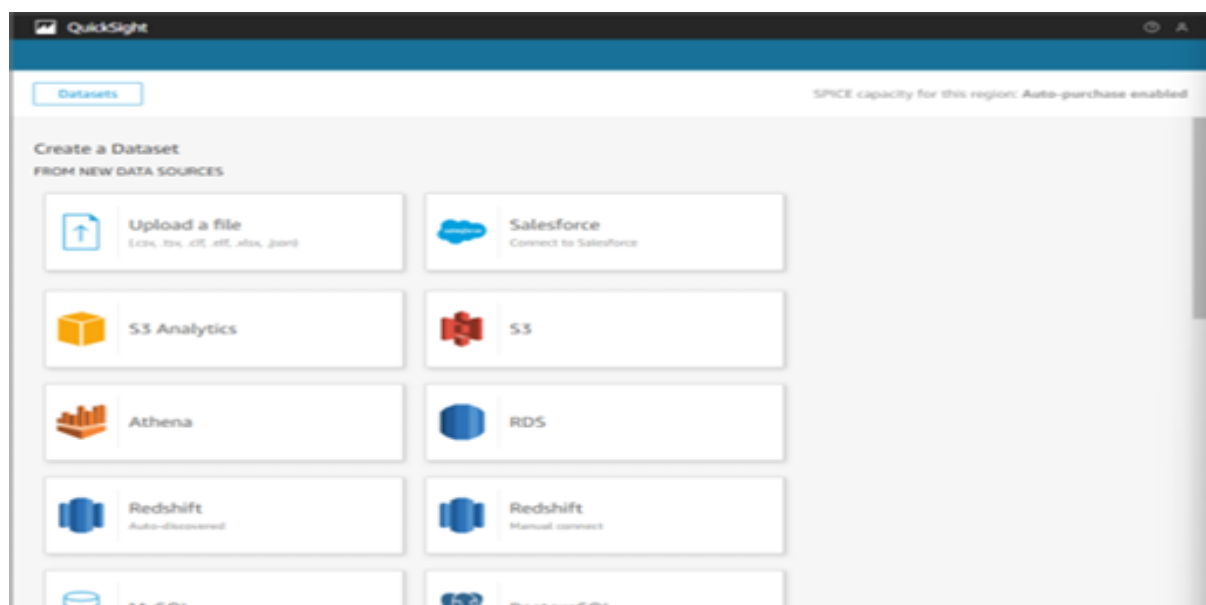
buckets along with EC2 instances.

#### 4. Visualizations:

Created Quicksight Account:



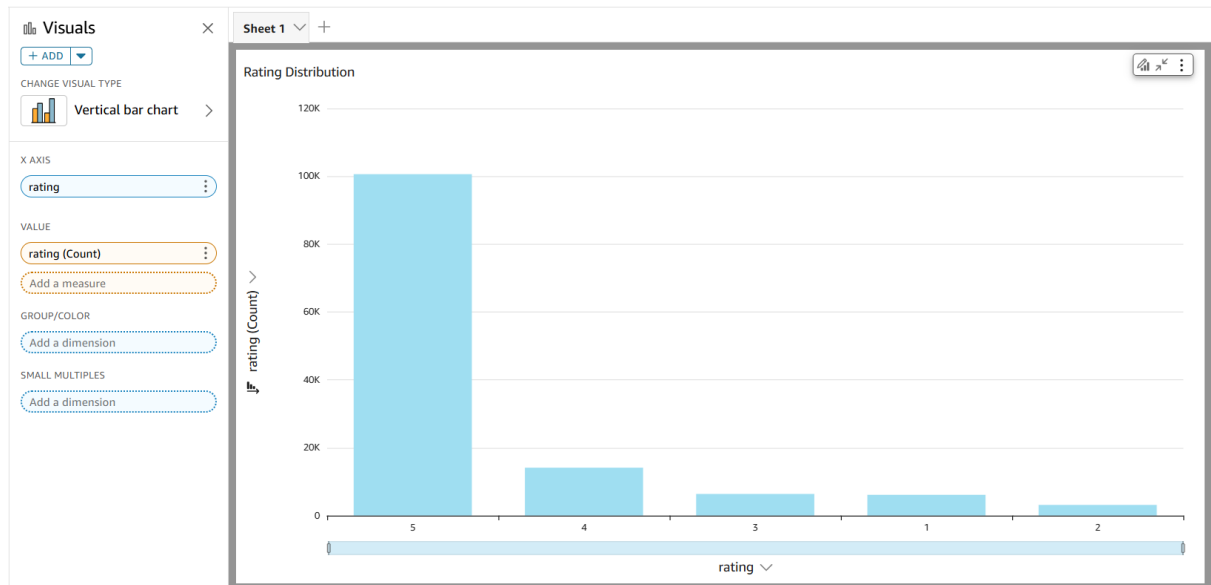
Importing Dataset From S3:



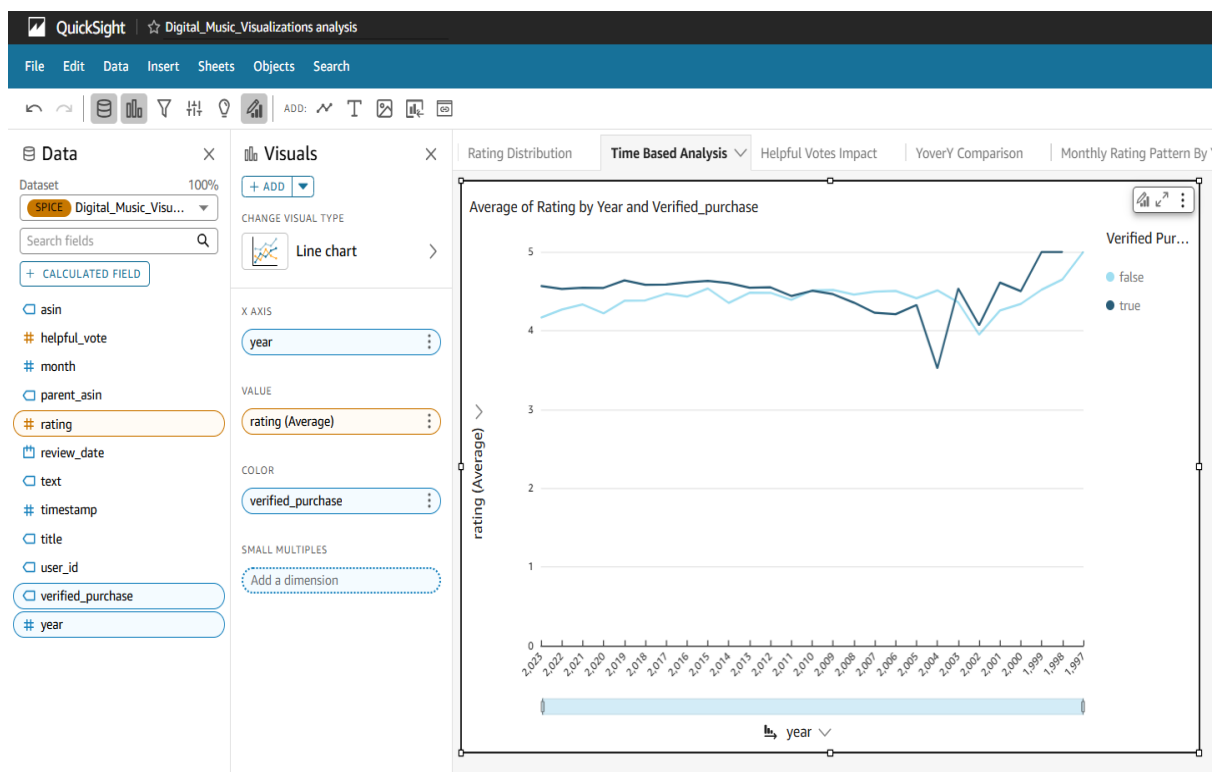
Name: Harsh Chauhan  
Username: hchauhan

## Visualizations:

### 1) Rating Distribution:

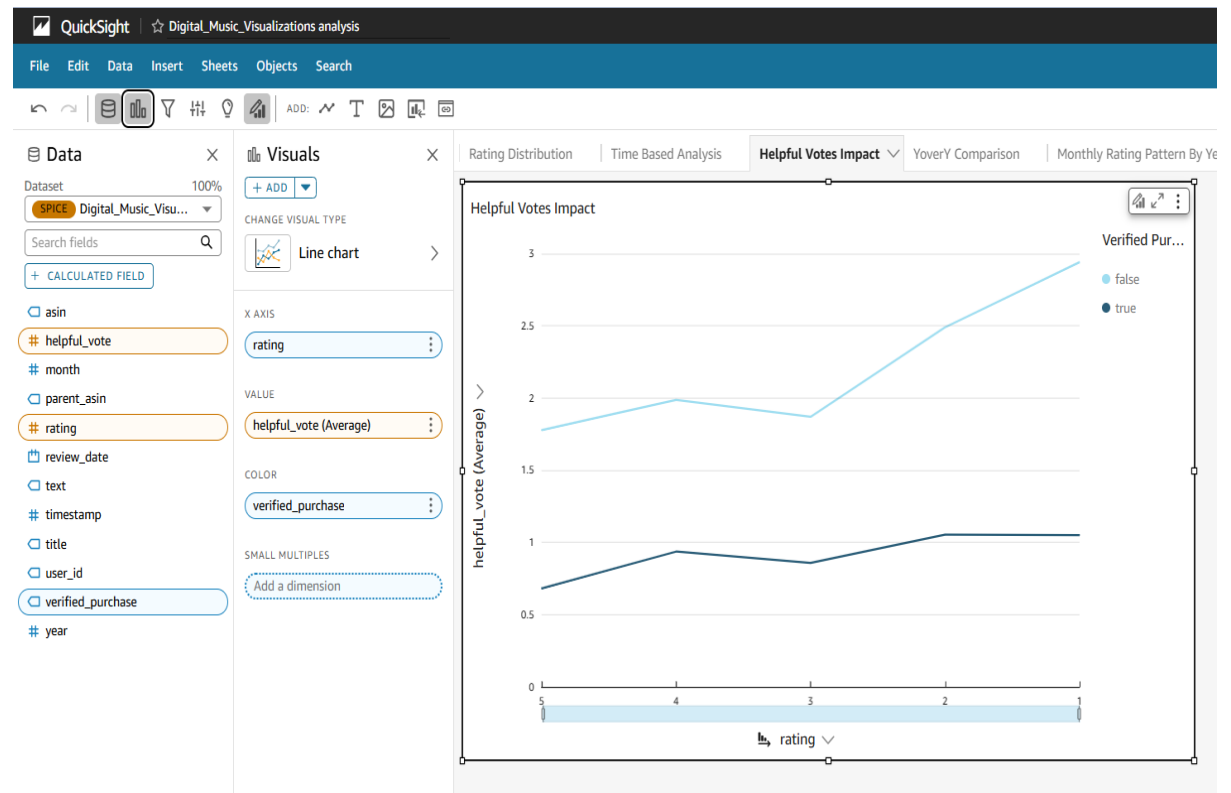


### 2) Time Based Analysis

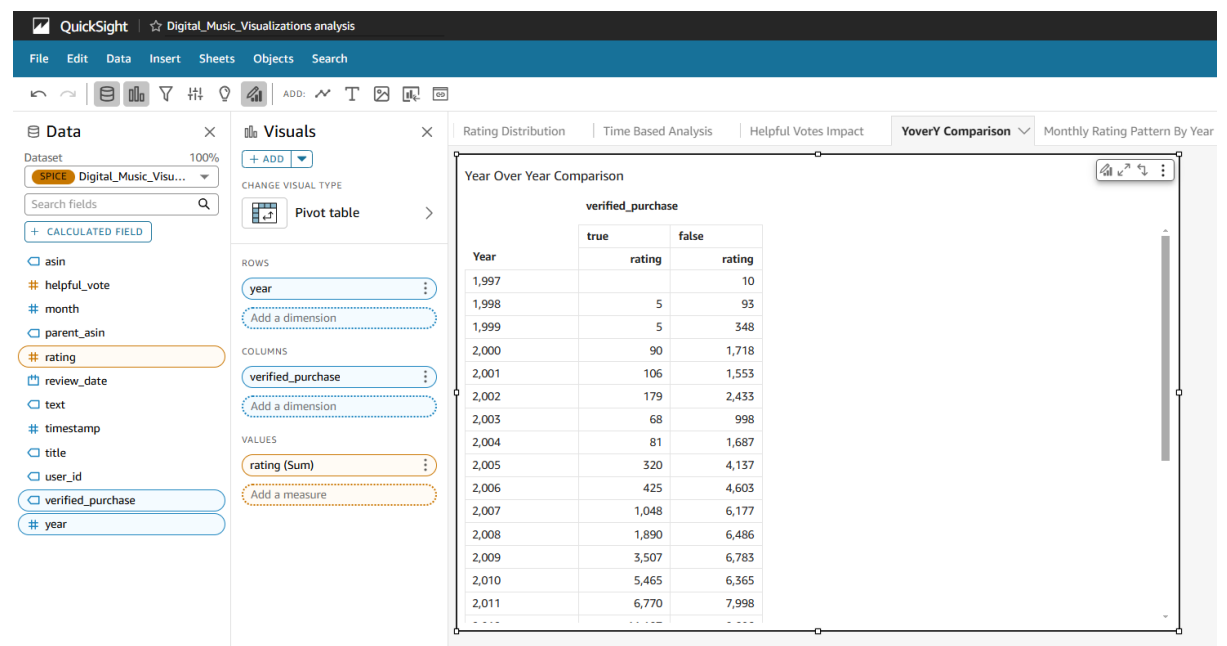


Name: Harsh Chauhan  
Username: hchauhan

### 3) Helpful Votes Impact

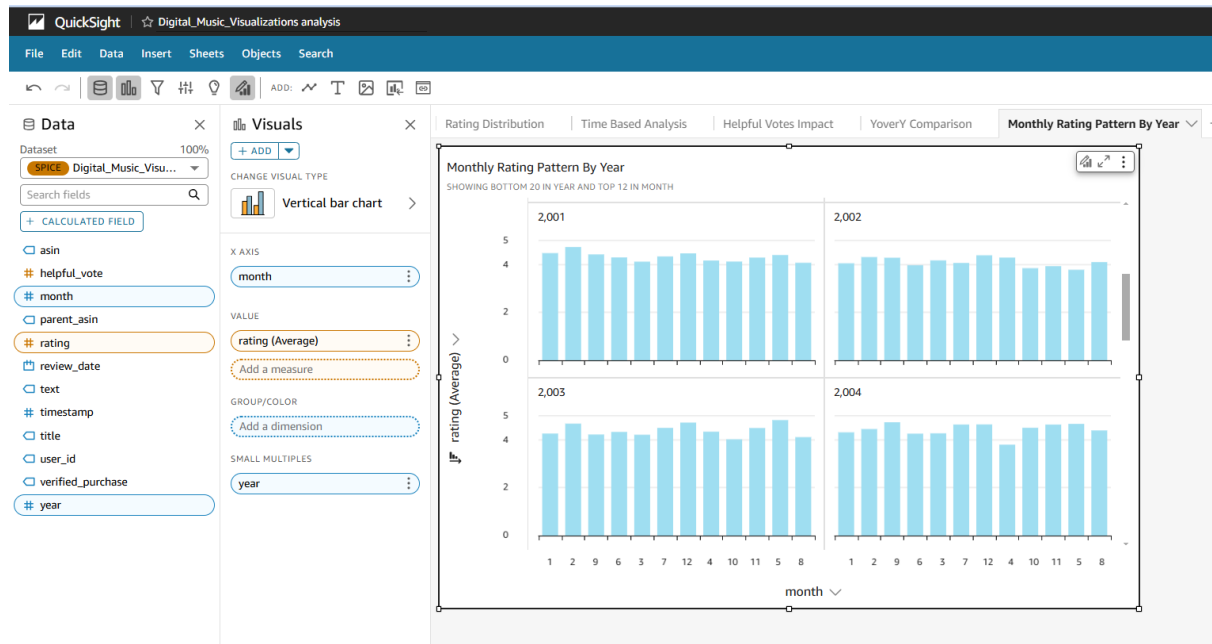


### 4) Year Over Year Comparison



Name: Harsh Chauhan  
Username: hchauhan

## 5) Monthly Rating Pattern by Year



Deleting the Quicksight account : In order to not incur any further charges , I have deleted the account.

The screenshot shows the 'Account termination' page in the Amazon QuickSight console. The account name is 'hchauhan'. The 'Account termination protection' is currently 'on'. A warning message states: 'Delete account. Deleting this account can't be undone and will permanently delete all users, dashboards, analyses, along with other related data.' Below this, there is a confirmation field with the text 'confirm' and two buttons: 'Cancel' and 'Delete account'.

## 5. Results and Analysis:

1) Machine Learning Results:  
Sagemaker Model Analysis:

Model achieved 77.236% accuracy.

Performance metrics:

- 1) F1 Macro: 17.748%
- 2) Precision: 20.259%
- 3) Recall: 20.13%

The results show a strong bias towards predicting class 5.0, with 77.173% precision and 99.756% recall for this class. This indicates that the dataset likely has a significant imbalance, with class 5.0 being the dominant category. The model shows very poor performance on other classes (4.0, 3.0, 2.0, 1.0), with 0% precision and recall for most of them. This further confirms the class imbalance issue and suggests the model struggles to identify less common categories. The advanced metrics show average accuracy (76.992%), F1 score (17.748%), precision (20.259%), and recall (20.13%), which are relatively low, indicating that while the model performs well on the dominant class, it struggles with overall balanced performance across all classes. These results suggest that the dataset has a significant class imbalance issue, and the model has overfit to the majority class (5.0). To improve results, we may need to address the class imbalance, possibly through resampling techniques, class weighting, or collecting more data for underrepresented classes.

## 2) Visualization Analysis:

### 1) Rating Distribution

The rating distribution shows a strong positive skew with 5-star ratings dominating at approximately 100,000 counts, while ratings 4 through 1 have significantly lower frequencies around 10,000-20,000 counts each.

### 2) Time Based Analysis:

The graph shows average ratings over time (1997-2023) with verified purchases consistently rating slightly higher than non-verified ones, both generally maintaining ratings between 4.0-4.5 stars.



### 3)Helpful Votes Impact

The graph shows that helpful votes increase as the rating increases, with verified purchases having a higher impact on helpfulness across all ratings.

### 4)Year over Year Comparison

The data shows a consistent year-over-year increase in both verified and non-verified purchase ratings from 1997 to 2011, with non-verified purchases maintaining higher volumes throughout the period.

### 5)Monthly rating pattern by Year

The monthly rating patterns across 1997-2023 show consistent average ratings between 4-5 stars with slight seasonal fluctuations but overall stable performance throughout each year.

## 6. Challenges:

- 1) Challenges faced with the AWS Sagemaker model configuration.
- 2) Incurred unexpected AWS charges (\$285.72) primarily from using Sagemaker.

## 7. Conclusion:

This project highlights the power of combining distributed computing frameworks and cloud-based tools to process and analyze large-scale datasets. By leveraging AWS S3, PySpark, AWS SageMaker, and QuickSight, the study successfully processed Amazon's Digital Music review dataset, providing valuable insights into customer behavior, preferences, and review patterns. Key findings include the dominance of 5-star ratings, seasonal trends in review activity, and the impact of verified purchases on helpfulness and satisfaction scores.

Name: Harsh Chauhan  
Username: hchauhan

The machine learning model developed using SageMaker achieved a 77.236% accuracy but revealed significant challenges related to class imbalance, overfitting, and limited generalization across all rating categories. This underlines the need for future improvements, such as balancing datasets through resampling or class weighting, to enhance model performance.

While the project faced challenges, including unexpected AWS charges and configuration issues with SageMaker, these experiences offered practical insights into the complexities of cloud-based analytics workflows. The visualizations created through QuickSight provided intuitive representations of trends and patterns, aiding in comprehending the dataset's dynamics.

This project underscores the importance of careful resource management and configuration in cloud services and demonstrates the potential of data analytics to derive actionable insights in the digital music domain. Future efforts could expand on this foundation by addressing identified limitations and exploring other feature sets within the dataset.

## **8. References:**

- a) AWS Documentation.
- b) Sagemaker Documention.
- c) PySpark Documentation.
- d) Amazon Review Dataset: [Amazon Reviews'23](#)
- e) Quicksight Documentation
- f) Referred Perplexity.ai for setup error and configuration doubts.