Demographically-Enhanced Movie Recommendation System for Personalized Book Suggestions in Big Data

A Project Report
Presented to
DATA 228
Fall, 2023

Priya Varahan
Priyanka Akula
Pooja Manjunatha
Shreya Chikatmarla
Nandini Sreekumaran Nair

Abstract

The Project focuses on the development of a personalized recommendation system for movies and books by the use of Big Data Tools and Machine Learning algorithms. The important features were identified by analyzing the dataset. The data is analyzed and cleaned in AWS Glue, PySpark, DataBrew and Redshift. To this, machine Learning algorithms (K-Nearest-Neighbor) are applied for the prediction of books and movies using SageMaker and lambda functions which are AWS services. The system uses visualizations to understand the data. The UI allows the user to input their details which enables the system to recommend movies and books which were built using Amplify, Cognito and Amazon API Gateway. This system suggests movies based on user demographics and recommends books based on the genres of the suggested movies which provides a holistic and personalized entertainment experience in today's data driven world.

Keywords: machine learning, big data, AWS services

Acknowledgements

We would like to thank Professor Andrew H. Bond for his constant guidance and assistance throughout the course work and for the successful development of our project.

Table of Contents

Chapter 1. Introduction

1.1 Project goals and objectives

In today's data driven world, where there is an influx of data being produced, personalized recommendations have become significant. This project uses big data to give personalized suggestions for both movies and books using both collaborative filtering and content-based filtering algorithms.

The project is built on AWS(Amazon Web Services). The aim of this project is to offer recommendations based on the users tastes and preferences. By utilizing AWS's tools and technology, a recommendation engine is built that suggests movies and books. The movies are recommended based on the user's demographics and based on the genres of the movies recommended, a book is suggested as well.

1.2 Problem and Motivation

In the current digital world, the overflow of data poses a challenge for users to scroll through so much content, which leads to information overload. There are very few recommendation engines that are tailored to the user's preferences, especially in the combination of movies and books. Traditional systems do not use the demographics to provide suggestions.

Our motivation to build this project comes from recognizing that personalized recommendations are more significant now than ever. User's need help to navigate through the content available to them that aligns with their interests. By leveraging Big data and AWS tools, we aim to address the above mentioned problem by giving recommendations for both movies and books. This improves user experience and gives a solution to the problem of "too much information". We want to bridge the gap between the overflow of data and personalized recommendations.

1.3 Project application and impact

The application of this project is to enhance the user's experience in the field of movies and books recommendations. The applications include:

1. Personalized Recommendation: User’s have to log in and will be recommended movies and a book which is based on the demographics and genres of movies recommended respectively.
2. Demographic Data: The data that is collected and analyzed for this project is demographic data which takes into consideration the age, gender and location of the user.
3. AWS Infrastructure; Using AWS’s infrastructure makes sure that the platform is both reliable and scalable which improves the performance of the project.

The impact of this project includes:

1. Data Analytics and Big Data : The use of Big Data Analytics to process and analyze large volumes of data.This ensures that the recommendations given are accurate and also the framework can handle expanding data sets.
2. Cross- media Recommendations: By going a step ahead and recommending movies and books, this project crosses standard boundaries. The user can discover content in the field of movies and books.
3. Demographic insights: This project involves collecting and analyzing demographic data. This can provide insights into what the customer wants.
4. Versatile Application: This project can be used across various industries, not only in entertainment but it can be used in advertising, marketing and content creation.

1.4 Data Description

There are two dataset used for Movies and Books

The movie dataset is collected from

https://www.tensorflow.org/datasets/catalog/movie_lens#movie_lens1m-ratings

Movie dataset consists of:

Each user has rated at least 20 movies. It consists of 1 million rows. The dataset size is 308.42 MiB.

The dataset is collected from GroupLens research Group. The attributes are:

1. Bucketized user age: The age values of users who rated the movie. The age is grouped into buckets.
2. Movie_genres: The genres the rated movie belongs to.
3. Movie_id : A unique ID for every movie that is rated by the individual user.
4. Movie_title: The title of the rated movie and the year is mentioned in the parentheses.
5. Raw_user_age: The exact age of users
6. Timestamp: The time at which the user rated the movie.
7. User_gender: Gender of the user who rated the movie
8. User_id : A unique Identifier given to a user who made the rating
9. User_occupation_label: The occupation of the user in the form of a number
10. User_occupation_text: The original string text
11. User_rating: The rating that ranges from 1-5 given to the user.
12. User_zip_code: Zip code of the user who made the rating.

The Book Dataset is collected from:

https://www.kaggle.com/datasets/justinnguyen0x0x/best-books-of-the-21st-century-dataset

The book dataset is collected from kaggle. The source is Goodreads. It is a platform for readers and book recommendations. The attributes are:

1. Title :The title of each book
2. Author: The author or multiple authors of the books
3. Genre: The genres every book belongs to
4. Ratings: The rating given to the book by users
5. Reviews: The reviews given by the users to the book
6. Metadata: Additional information about books.

1.5 Project results and deliverables

Table 1

Project Deliverables

Phases	**Deliverables**	**Scheduled Date**
Planning and Designing	Topic	Sep 1
	Choosing Dataset	Sep 3
System Implementation	Creating S3 bucket	Oct 15
	Adding Dataset	Oct 15
	Cleaning	Oct 15
	ETL in AWS Glue	Oct 15
	Analysis	Oct 20
	Training Model	Nov 1

	Deploying Model in sagemaker	Nov 21
	Creating Lambda function	Nov 23
	Creation DynamoDB	Nov 23
	Front end	Nov 25
Documentation	Presentation	Nov 29
	Report	Dec 10

Chapter 2. Literature Survey

[1]The research paper “Movie Recommender System using K-Means Clustering and K-Nearest Neighbor” proposed a recommendation system that consists of three modules namely, input module, processing module and output module. The user enters his details in the user module and the details are age, gender and pincode. The processing module takes this input and creates separate data frames for movies and users. A utility matrix is built and K-means clustering is used to show the genre the movie belongs to. The output module shows the predicted movies that the user might like. The output contains movie names and predicted ratings for those movies. It is observed that the RMSE of the proposed technique is better than the existing technique. Future work includes Sentimental Analysis to enhance the efficiency of the movie recommendation system.

[2] The research paper “Cloud Big Data Decision Support System for Machine Learning on AWS” proposed a recommendation system that chooses the best dataset and a machine learning algorithm to optimize time and money spent on AWS instances. The motivation of the project is to select the best Amazon instances for general purpose, GPU instances,FPGA instances, Memory Optimized, Storage Optimized, and Compute Optimized. The datasets chosen are from different sources and are in different formats like .csv and .json. The datasets used are from Chicago Data Portal, Maryland Open Data Portal and NYC Open Data. All the AWS instances are tested first. AWS CloudFormation and boto3 client is used to access AWS API and run the tests for the instances. AWSDynamDB is used to store test results. Python is used for machine learning as it works seamlessly with AWS tools used. The code is first run on local machines for

debugging and then run on AWS instances. The aim of this paper is to predict execution time and cost for the machine learning methods implemented and tested for every AWS instance. Future work includes testing AWS lambda functions and AWS Step Functions and to use many more Machine Learning techniques.

[3] The research paper “Anomaly Detection in Intrusion Detection System using Amazon SageMaker” applies AI and Ml to analyze network traffic that could prevent cyber threats. Amazon SageMaker is used as a cloud platform to implement Machine Learning Models. The project flow is data collection, processing, model creation and evaluation. The dataset used is the UNSW-15 dataset. Two models are created- Random Cut Forest and XGBoost out of which the latter performed better with an accuracy of 61.83%. The recall for this model is 96.49% and f-1 score is 73.24% These two algorithms are already pre-built on SageMaker. Further work includes applying these two algorithms on streaming data.

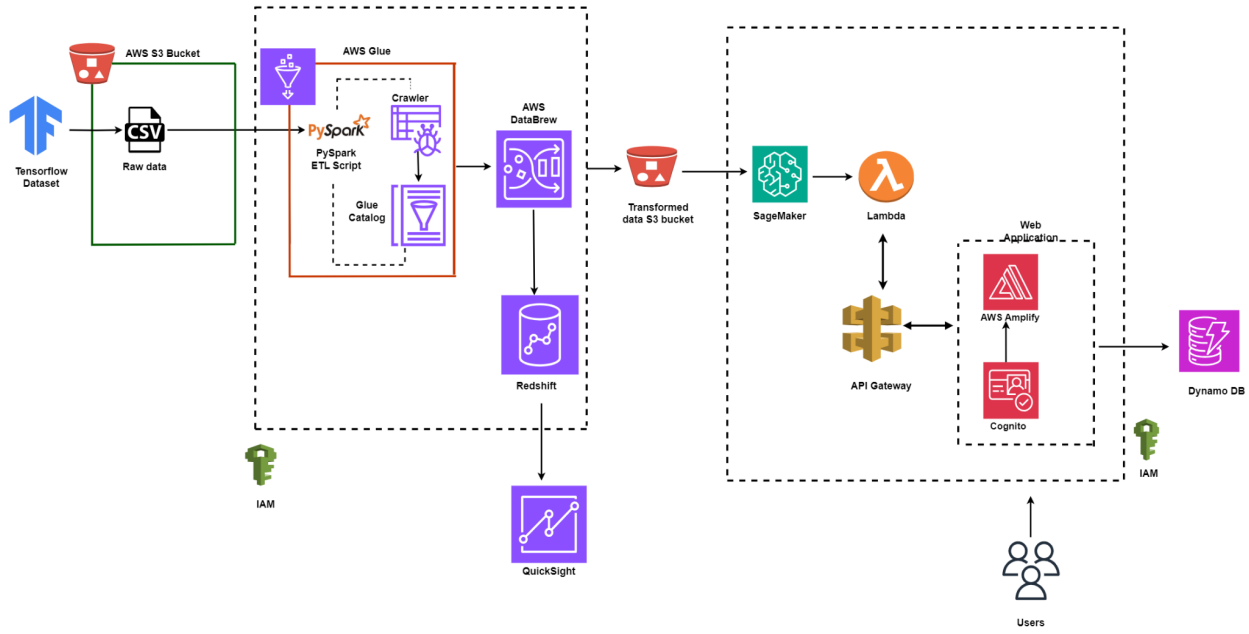
Chapter 3. System Design

3.1 System Architecture

The system design incorporates AWS-based data and application workflow services for ETL procedures, analytics, and user-facing web application features to easily transform, analyze, and use data in a safe, scalable environment. Figure 1 depicts an AWS cloud architecture for data processing and web application deployment. It illustrates the flow from raw data in an S3 bucket, processed by AWS Glue using PySpark, to transformed data for analytics in Redshift and QuickSight, and finally to application interaction through AWS Amplify, API Gateway, Lambda, and DynamoDB, managed by IAM roles for security.

Figure 1

System Architecture



The following are the detailed system architecture flow steps for the project:

Data Storage and Ingestion

The raw data for the TensorFlow MovieLens and Book datasets is kept in an AWS S3 bucket as a CSV file.

Data Processing

Data processing with AWS Glue includes a Glue Crawler that deduces the raw data's schema.An extract, transform, load (ETL) script written in PySpark handles the unprocessed data.Metadata regarding the formats, sources, and structures of the data are stored in the Glue Catalog.

Data Transformation

After processing, the data is sent to AWS DataBrew for further transformation operations, producing transformed data that is once more kept in an S3 bucket.

Machine Learning

Using book and movie dataset, Amazon SageMaker uses the modified data to create and train a scalable movie recommendation model that offers tailored recommendations.

API and Serverless

The role of an AWS Lambda function suggests a serverless computing model that might be utilized for a variety of purposes, such as managing requests from the API Gateway or initiating processing stages.AWS API Gateway is a service that may be used to create, publish, manage, monitor, and secure REST, HTTP, and WebSocket APIs.

Web Application and Authentication

AWS Amplify is utilized to create web applications that interface with AWS Cognito for identity management and user authentication.Given that the web application is linked to the API Gateway.

Database and Visualization

DynamoDB, a NoSQL database service, shows that data may be stored in or retrieved by the web application from this database.The Redshift data warehouse is linked to AWS QuickSight, a business intelligence tool for data visualization, indicating that Redshift data is utilized for reporting and analytics.

Access Management and Users

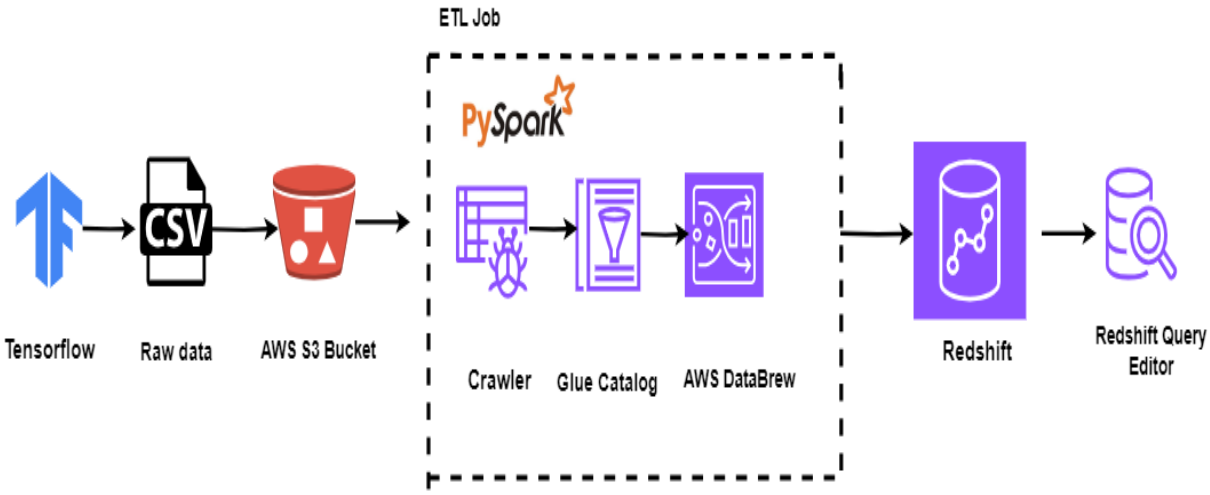
AWS Identity and Access Management, or IAM, is displayed twice, signifying that it controls access to both the web application/database and the data processing/storage components. The end users of the system are represented by the users at the bottom, who are probably interacting with the web application and consuming the analytics reports or machine learning insights.

3.2 ETL Architecture

Visualizing the ETL architecture allows raw data extraction, PySpark transformation in an AWS Glue task, and Redshift loading for sophisticated querying and data analysis. This simplified technique handles and interprets big cloud datasets efficiently.Figure 3 shows the ETL architecture of the project.

Figure 2

ETL Architecture



Data Source

The raw data for the TensorFlow MovieLens and Book datasets is kept in an AWS S3 bucket as a CSV file.

Raw Data

TensorFlow stores its data in CSV files, a widely used format for tabular data storage. AWS S3 Bucket.

AWS S3 Bucket

Following that, the CSV files are uploaded to an AWS S3 bucket, a scalable storage solution.

ETL Job

A PySpark job initiates the ETL (Extract, Transform, Load) procedure. PySpark is an Apache Spark-based framework for Python big data processing.

Data Cataloging and Transformation

In order to populate the AWS Glue Catalog, a Glue Crawler is utilized to identify ,classify various data formats and use them for data cleaning.

Data Warehouse

The prepared data is then loaded into Redshift, which is AWS's data warehousing service, allowing for complex queries and analysis.

Query and Analysis

Finally, the Redshift Query Editor is used to execute SQL queries on the data within Redshift for further analysis or to generate reports.

3.3 Tools Used

For a Demographically-Enhanced Movie Recommendation System aimed at providing personalized book suggestions in a Big Data environment, an array of AWS tools forms the backbone of the architecture. AWS S3 Buckets are utilized for massive data storage, capturing user demographics and interactions. AWS Glue, with its PySpark ETL capabilities, is employed to cleanse and structure this vast data trove, while the Glue Catalog indexes it for efficient retrieval. Amazon Redshift serves as the data warehousing solution, where complex analytical queries are executed to understand user preferences deeply. Amazon SageMaker spearheads the

machine learning aspect, crafting a recommendation algorithm that adapts to demographic nuances. The orchestration of these processes could be managed by AWS Lambda for efficient resource utilization. Finally, Amazon QuickSight taps into Redshift to visualize insights and enhance decision-making, ensuring the recommendation system remains intuitive and user-centric. Table 2 lists the tools used for the project.

Table 2

Table listing the tools and its usage

Tools	**Usage**
AWS S3 Bucket	Stores data with high scalability and availability
AWS Glue	Integrates and prepares data for analysis
Glue Crawler	Automates data schema discovery
PySpark ETL Script	Processes large-scale data transformations
Glue Catalog	Stores and organizes data metadata
AWS DataBrew	Visually cleans and normalizes data
AWS Redshift	Analyzes large volumes of data
Amazon SageMaker	Builds, trains, and deploys ML models
AWS Lambda	Runs code without server management

AWS API Gateway	Creates and manages APIs
AWS Amplify	Develops full stack applications
Amazon Cognito	Manages user authentication and identity
Amazon DynamoDB	Provides a fast NoSQL database service
IAM	Secures access to AWS resources
Amazon QuickSight	Delivers business intelligence and data visualization

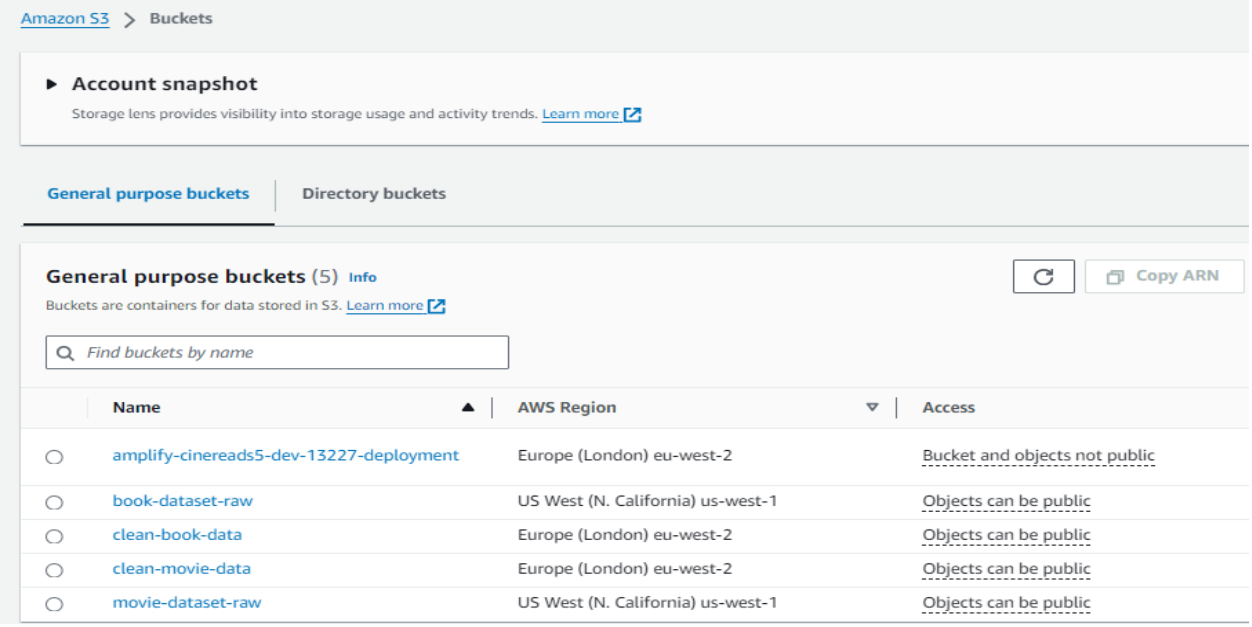
Chapter 4. System Implementation

4.1 ETL Process

The ETL (Extract, Transform, Load) process for a Demographically-Enhanced Movie Recommendation System for Personalized Book Suggestions using the specified AWS tools would involve several steps.The raw data consisted of two files, in csv format extracted from MovieLens Tensor File dataset .This dataset was first loaded into the S3 bucket. ‘book-dataset-raw’ and ‘movie-dataset-raw’ these buckets contain the raw data which needs to be transformed and cleaned.Figure 4 depicts the Storage of Raw Data in S3 Bucket.

Figure 3

Storage of Raw Data in S3 Bucket



Data cleaning for Book Dataset

The dataset has 10,018 entries with four columns: `id`, `title`, `book_link`, and `genre`. The unique `id` and 9,098 unique titles and 9,121 unique book links indicate that some books may have several entries. The most common title appears four times. The 'genre' column, with a peak of 'Fiction' at 40, has 999 null entries, suggesting missing data.Figure 5 shows the image of the book dataset with Null values. Below are the steps used for cleaning the Book Dataset AWS DataBrew and Python :

- Whitespace in string fields was reduced where extra spaces were found.

- Duplicate entries were identified and removed to preserve data integrity.

- The `genre` field, which contained multiple genres in a single string, was separated into distinct components or restructured for analysis, possibly using one-hot encoding or a genre mapping.

- Data type validation ensured that `title` and `book_link` were stored as strings, and `id` as an integer.

- The `book_link` URLs were verified for legitimacy and correctness.

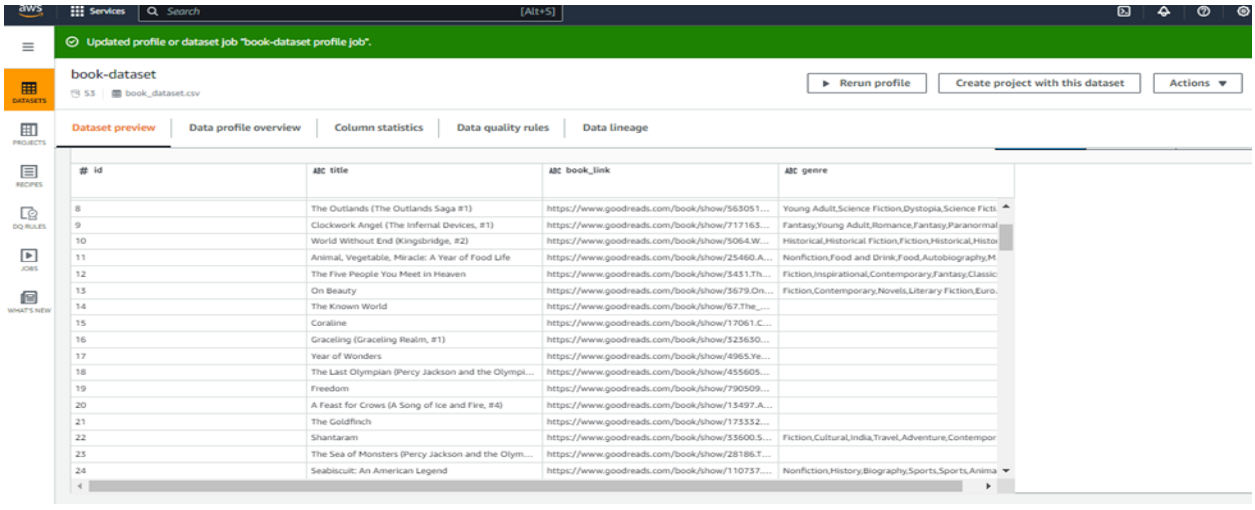
- Missing data was addressed, with strategies like imputation or removal, depending on its importance and volume.

- Text normalization, such as converting text to lowercase, was applied to the `title` and `genre` fields for consistency.

- The uniqueness of the `id` field was confirmed, and the dataset was scrutinized for structural issues to ensure it was ready for further analysis or processing.

Figure 4

Book dataset with Null values



Data cleaning for Movie Dataset

The dataset contains 100,000 records and is free of duplicate entries. It is well-structured, with complete data across all fields, no null values, and a variety of unique entries in categorical fields, ensuring it is primed for analysis. There are 216 unique categories in one of the list-encoded string fields and 21 unique descriptors in another byte string field, with the rest of the data similarly diverse and comprehensive.Figure 6 shows the image of the AWS Brew interface for cleaning the dataset.Below are the steps used for cleaning the Movie Dataset in AWS DataBrew and Python:

- Fields such as `movie_id`, `movie_title`, `user_id`, `user_occupation_text`, and `user_zip_code`, initially in byte string format (indicated by `b'...'`), were converted to standard strings.

- Numeric columns including `bucketized_user_age`, `raw_user_age`, `timestamp`, `user_occupation_label`, and `user_rating` were scrutinized for anomalies.

- The `movie_genres` column, containing string-encoded lists, was parsed into list objects or separated by individual genres.

- The dataset underwent checks for missing values and duplicates, which were then appropriately addressed.

- Correct data types for each column were ensured to maintain data integrity.

- Text fields like `movie_title` and `user_occupation_text` were standardized for uniformity.

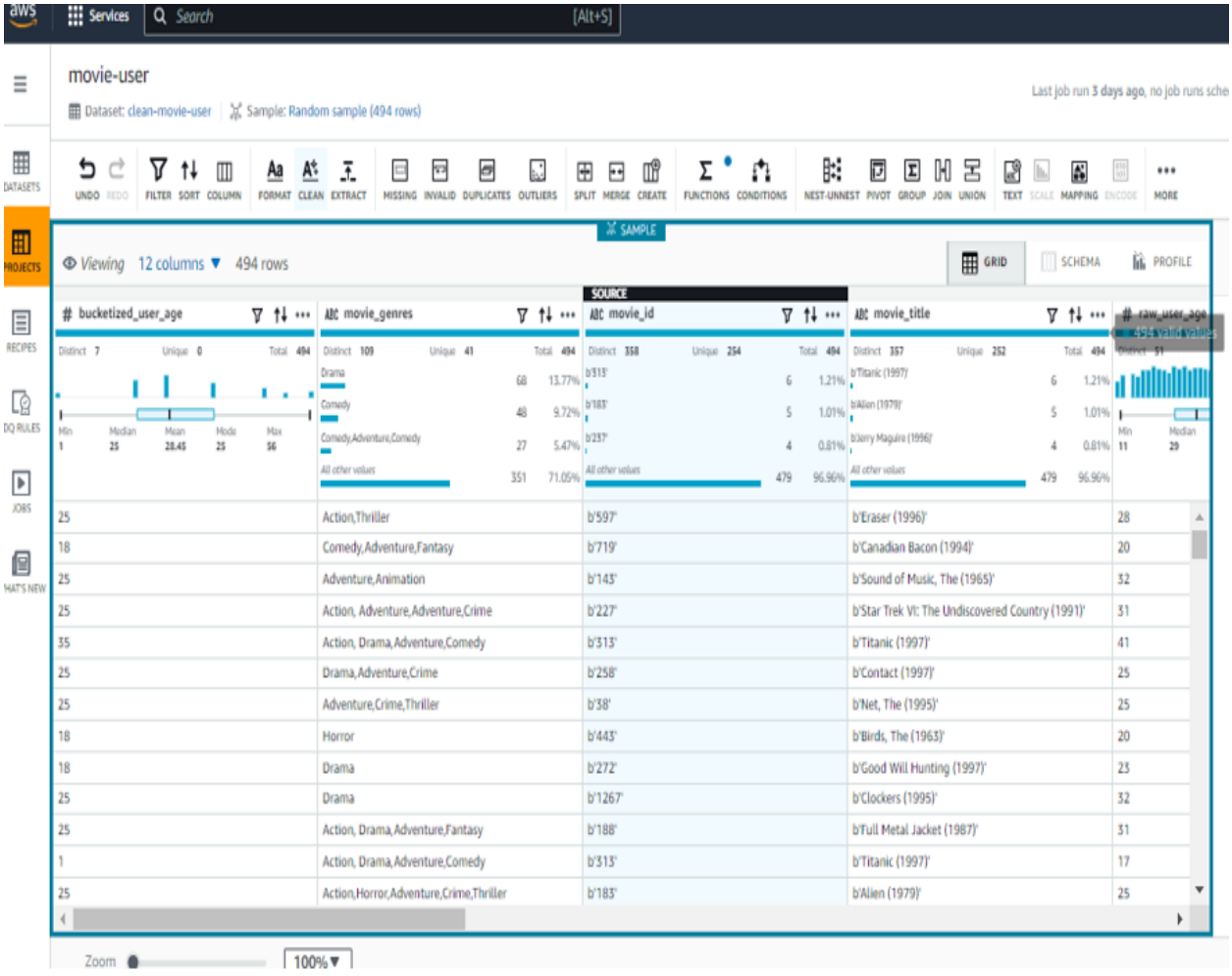
- Consistency checks, particularly for categorical fields like `user_gender`, were conducted, and any data anomalies or errors were corrected.

- A review of the dataset's structure was carried out to rectify any inconsistencies impacting its suitability for analysis or processing.

Figure 5

AWS brew interface

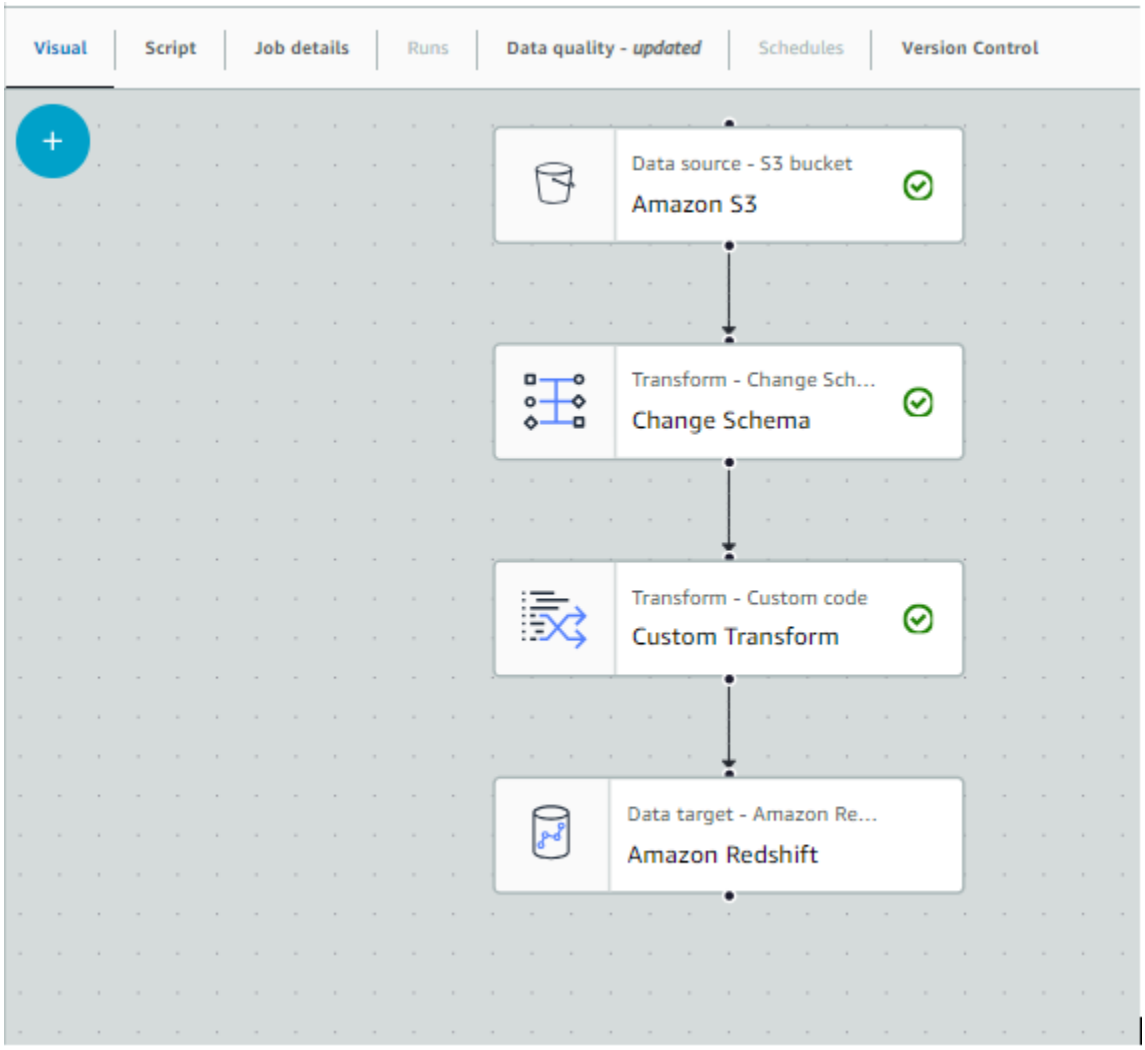
Schema Transformation and custom processing



A completed data pipeline had extracted movie and book datasets from an Amazon S3 bucket, transformed their schema, processed them with custom code, and ultimately loaded them into an Amazon Redshift data warehouse for analysis.The below figure 7 shows the ETL process

Figure 6

ETL Process Schema Transformation and custom processing



Below are the steps followed for the ETL process:

Data source - S3 bucket (Amazon S3): The movie and book datasets were initially stored in an Amazon S3 bucket from which the data was first pulled. These datasets were stored in a scalable cloud storage environment made possible by the S3 service.

Transform - Change Schema:The data was transformed to alter its schema after it was extracted. In order to prepare the data for further processing, this stage required changing the datasets' structure, potentially by renaming columns, changing data types, or eliminating unneeded columns.

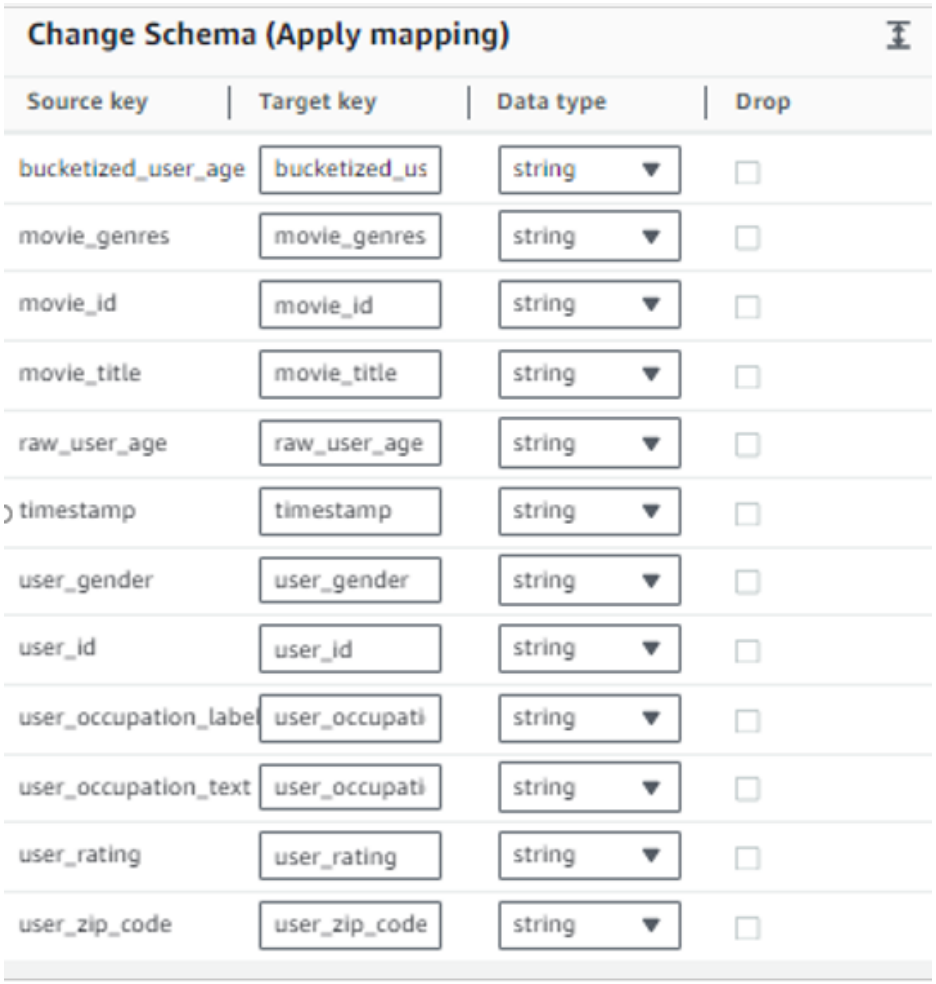
Transform - Custom code (Custom Transform):After that, custom code was used to process the data. The specific requirements of the movie and book datasets were taken into account when creating this transformation; these may have included cleaning the data, aggregating it, or performing other operations required for the data's intended use.

Data target - Amazon Redshift:Amazon Redshift was the ultimate destination for the transformed data. This cloud-based data warehouse service was used to load the movie and book datasets for further analysis or reporting.

ETL (Extract, Transform, Load) process for movie and book datasets is done . A mapping configuration where the user had defined the renaming and re-typing of each source field (source key) to a corresponding target field (target key) in the target system or dataset.A data schema mapping interface showed field names from a source dataset being mapped to target field names, all retained as string data types and none marked for deletion, indicative of a 'Change Schema' operation in ETL processes.Figure 8 shows the change schema operation in ETL.

Figure 7

Change Schema mapping configuration



The script shown in the below image outlines steps in a data transformation workflow for handling movie and book datasets in an ETL process. It first describes the generation of a dynamic frame from a CSV file in an Amazon S3 bucket. Then, it details a 'Change Schema' operation, mapping source data fields to new names and data types, indicating standardization and preparation for analysis.Figure 9 shows the code snippet for the change schema operation.

Figure 8

change schema operation script

```
33
34  # Script generated for node Change Schema
35  ChangeSchema_node1701324004477 = ApplyMapping.apply(
36      frame=AmazonS3_node1701323994859,
37      mappings=[
38          ("bucketized_user_age", "string", "bucketized_user_age", "int"),
39          ("movie_genres", "string", "movie_genres", "string"),
40          ("movie_id", "string", "movie_id", "int"),
41          ("movie_title", "string", "movie_title", "string"),
42          ("raw_user_age", "string", "raw_user_age", "int"),
43          ("timestamp", "string", "timestamp", "timestamp"),
44          ("user_gender", "string", "user_gender", "char"),
45          ("user_id", "string", "user_id", "int"),
46          ("user_occupation_label", "string", "user_occupation_label", "string"),
47          ("user_occupation_text", "string", "user_occupation_text", "string"),
48          ("user_rating", "string", "user_rating", "int"),
49          ("user_zip_code", "string", "user_zip_code", "int"),
50      ],
51      transformation_ctx="ChangeSchema_node1701324004477",
52  )
53
```

The script below in figure 10 uses header inclusion and a separator to construct a dynamic frame from a CSV file in an S3 bucket. Next, a Drop Duplicates script eliminates duplicates. The altered data is written back to S3 in columnar CSV format .

Figure 9

Drop Duplicates Script

```
# Script generated for node Drop Duplicates
DropDuplicates_node1702245963627 = DynamicFrame.fromDF(
    AmazonS3_node1702245123389.toDF().dropDuplicates(),
    glueContext,
    "DropDuplicates_node1702245963627",
)

# Script generated for node Amazon S3
AmazonS3_node1702245970152 = glueContext.write_dynamic_frame.from_options(
    frame=DropDuplicates_node1702245963627,
    connection_type="s3",
    format="glueparquet",
    connection_options={"path": "", "partitionKeys": []},
    format_options={"compression": "snappy"},
    transformation_ctx="AmazonS3_node1702245970152",
)

job.commit()
```

AWS Crawler and Glue Catalog

The below are the steps for AWS Glue crawler for the ETL process in the recommendation system Figure 11 and 12 shows the AWS crawler and Glue Catalog:

1) Crawler State:A crawler, designated as "newcrawler," was listed in the AWS Glue interface.

2) Successful Last Run: On its last activation, "newcrawler" processed new entries including movie listings, book publishing, and user-generated information like ratings and reviews without errors.

3)Schema Identification and Cataloging: The crawler detected the data schema for titles, genres, publication dates, and ratings during its operation and updated or created matching tables in the AWS Glue Data Catalog.

4)Preparation for Data Analytics: With an updated data catalog, the workflow may analyze processed and structured data and update movie and book recommendation algorithms.

Figure 10

AWS crawler

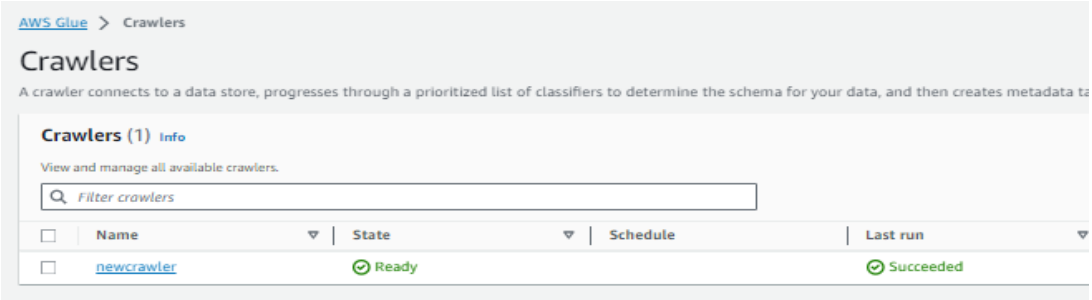
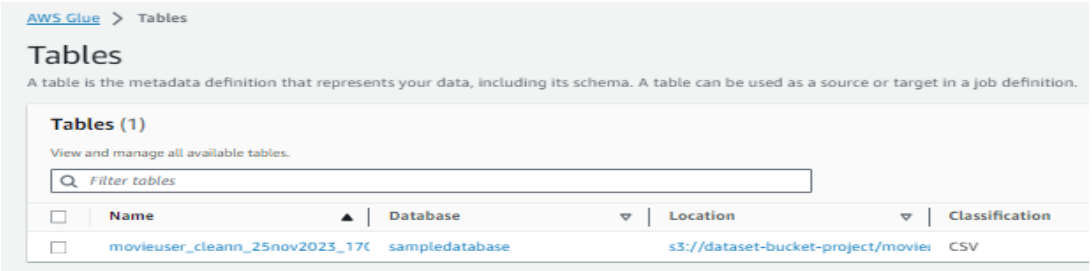


Figure 11

AWS Glue Catalog

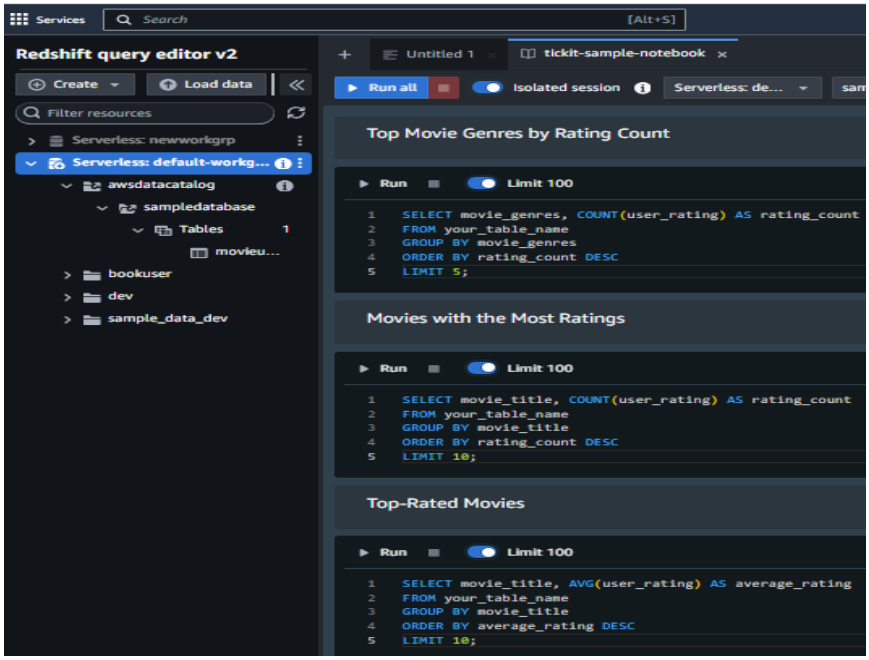


4.2 Data Analysis

Amazon Redshift Query Editor v2 interface, which had been used to execute SQL queries on Amazon Redshift, AWS's data warehouse service.The Figure 12 shows the queries performed in Redshift query editor for data analysis.

Figure 12

Redshift query editor interface



These queries had been utilized to derive insights from the data, such as identifying popular movie genres, viewer preferences, or content performance based on user ratings. If this method were applied to a book dataset, the queries would have been tailored to the specific fields and tables relevant to book data.

1. Top Movie Genres by Rating Count:This query had selected genres from a placeholder table your_table_name and counted the user ratings for each genre, ordering the results by the count in descending order. It was set to return only the top 5 genres, indicating an analysis to understand the popularity or frequency of ratings by genre.

2. Movies with the Most Ratings: This query had aggregated ratings by movie titles, listing the movies with the most user ratings and set to display the top 10 most-rated movies.

3. Top-Rated Movies: The third tab showed a query that calculated the average user rating for each movie, ordered the results by this average in descending order, and limited the output to the top 10 movies.

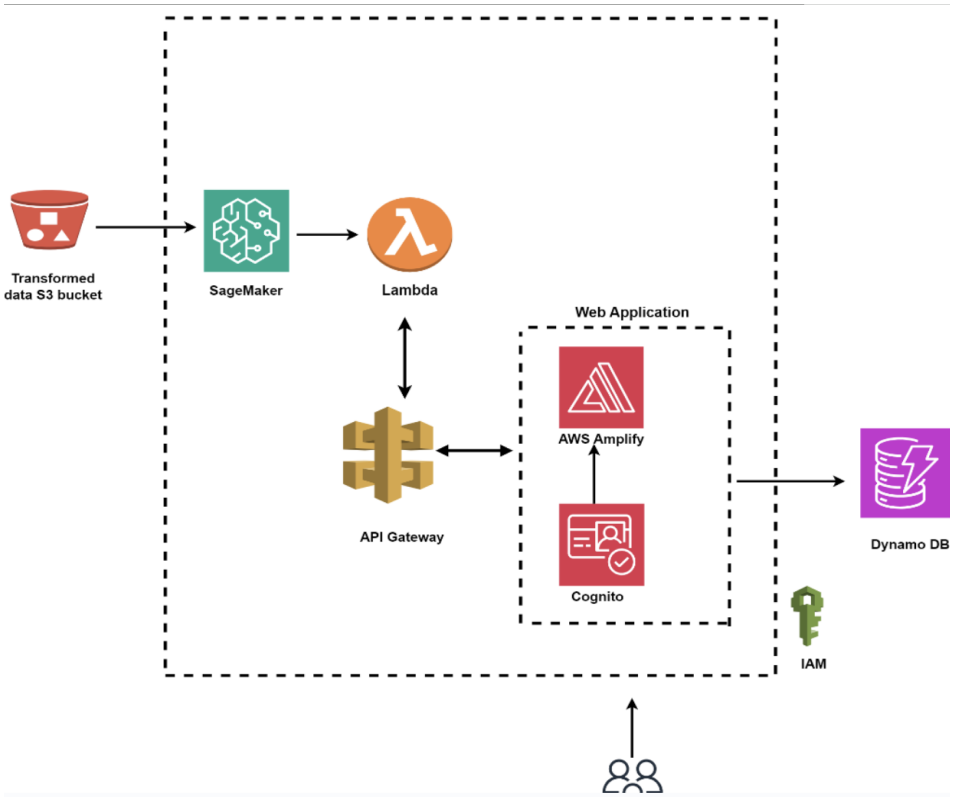
4.3 Web Integration

For web application development, integrating multiple AWS services creates a dynamic, scalable, and secure architecture. User authentication is provided by AWS Cognito, which scales to millions of users and integrates with social identity providers to streamline sign-in procedures. Amazon Amplify manages backend services effectively while streamlining frontend development. It is compatible with the React framework. AWS Lambda provides serverless computing, running code in response to events, while AWS API Gateway handles backend API calls, data processing, and request routing. With Amazon SageMaker, you can quickly design, train, and deploy models in a production-ready environment, streamlining the machine learning process.

By establishing certain access restrictions, AWS Identity and Access Management (IAM) controls access to AWS services and ensures safe interactions between application components. Low latency and scalability make Amazon DynamoDB, a NoSQL database service, ideal for managing large-scale applications. This service supports both key-value and document data structures for a variety of applications, adapting automatically to the needs of the application. Combining these many AWS services offers a complete platform for building reliable, effective, and safe cloud-based applications that meet a variety of needs and features.

Figure 13

Workflow Of The Web Application

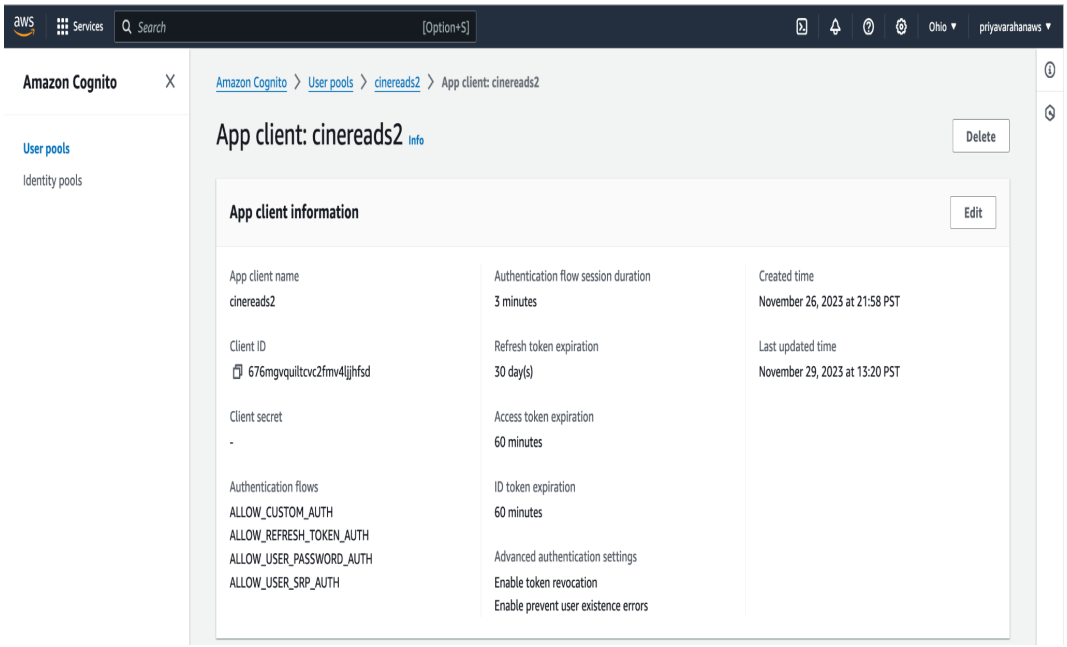


4.4.1 Setting up Authentication with AWS Cognito

An all-inclusive solution for user authorization and authentication is AWS Cognito. It makes it easier to build a user directory that can accommodate hundreds of millions of users. Simplifying the sign-in process is made possible by Cognito's seamless integration with social identity providers like Google, Facebook, and Amazon. Moreover, it enables multi-factor authentication, email verification, and password recovery as user management capabilities. The purpose of Cognito in the architecture is to give consumers a scalable, safe means of accessing the application.

Figure 14

Setting up Authentication with AWS Cognito

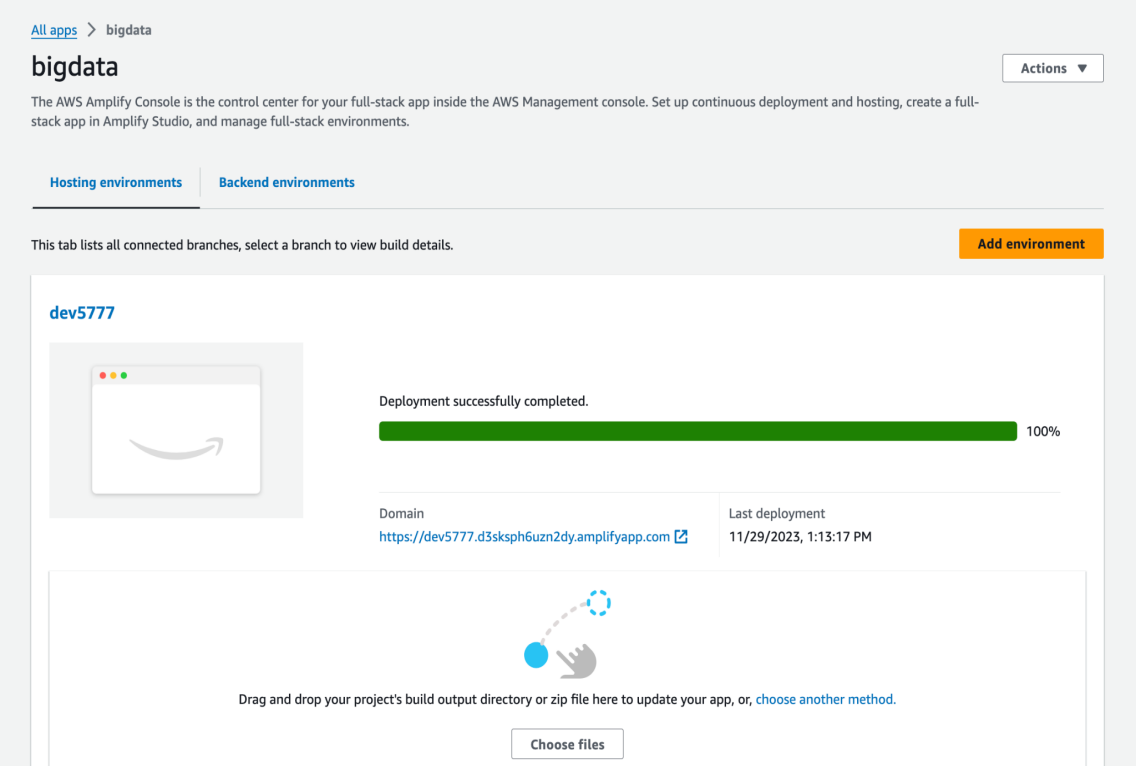


4.4.2 Frontend Development with AWS Amplify

Building, setting up, and launching mobile and online apps is made easier with AWS Amplify. It offers a selection of services and tools that are compatible with well-known frameworks like as Angular, Vue, and React. A lot of functions are covered by Amplify, such as analytics, RESTful APIs, and authentication. It also makes it simpler to integrate these features with the front end. Amplify's ease of use stems from its capacity to oversee backend services and offer a simple route for integrating intricate features.

Figure 15

Frontend Development with AWS Amplify



4.4.3 Backend API with API Gateway and AWS Lambda

Applications use AWS API Gateway as their front door since it manages all API calls, data processing, and request routing to different services. AWS Lambda, a serverless computing service that executes code in response to events like HTTP requests sent through API Gateway, is a complement to it. Building scalable, serverless microservices and backends is made possible by this combination. Lambda functions are an adaptable and effective solution to run backend code without having to manage servers. They can be triggered by a variety of AWS services.

Figure 16

Backend API with API Gateway and AWS Lambda

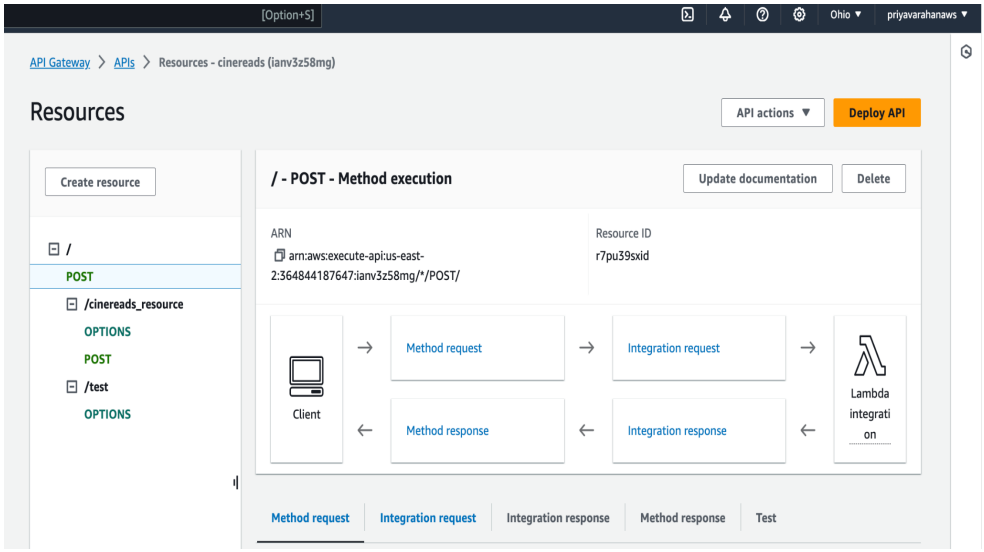
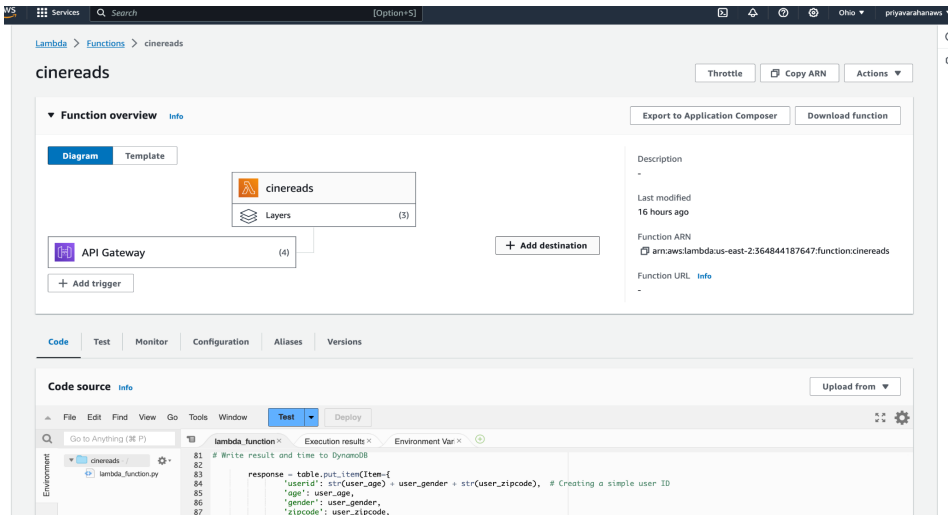


Figure 17

Lambda functions to handle API requests

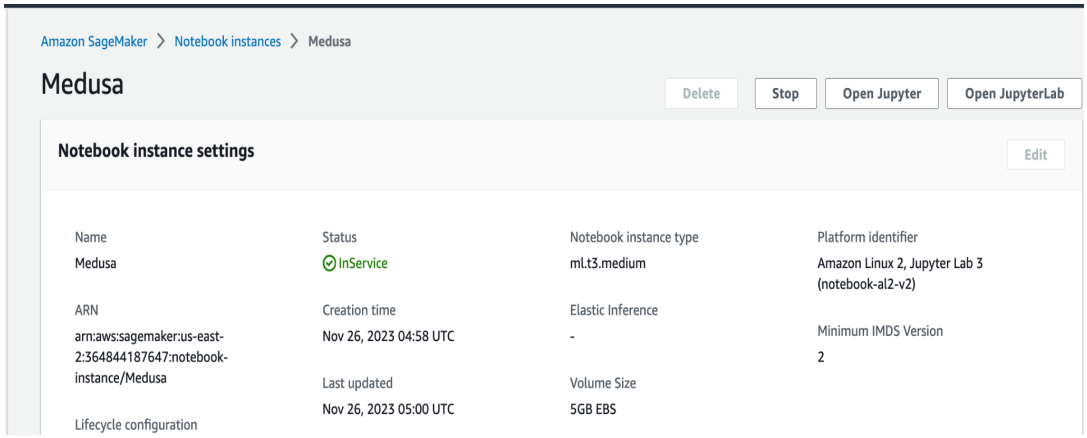


4.4.4 Machine Learning with SageMaker

Offering tools for each stage of the machine learning process, Amazon SageMaker is a fully managed service. Tasks like model creation, training, and deployment are made easier by it. Developers and data scientists may easily create and train machine learning models, then immediately implement them in a hosted environment that is prepared for production use. Popular machine learning frameworks are supported by SageMaker, providing model developers with flexibility. It plays a critical role in incorporating AI and ML capabilities into the application, improving features like analytics, customization, and recommendation engines.

Figure 18

Machine Learning with SageMaker



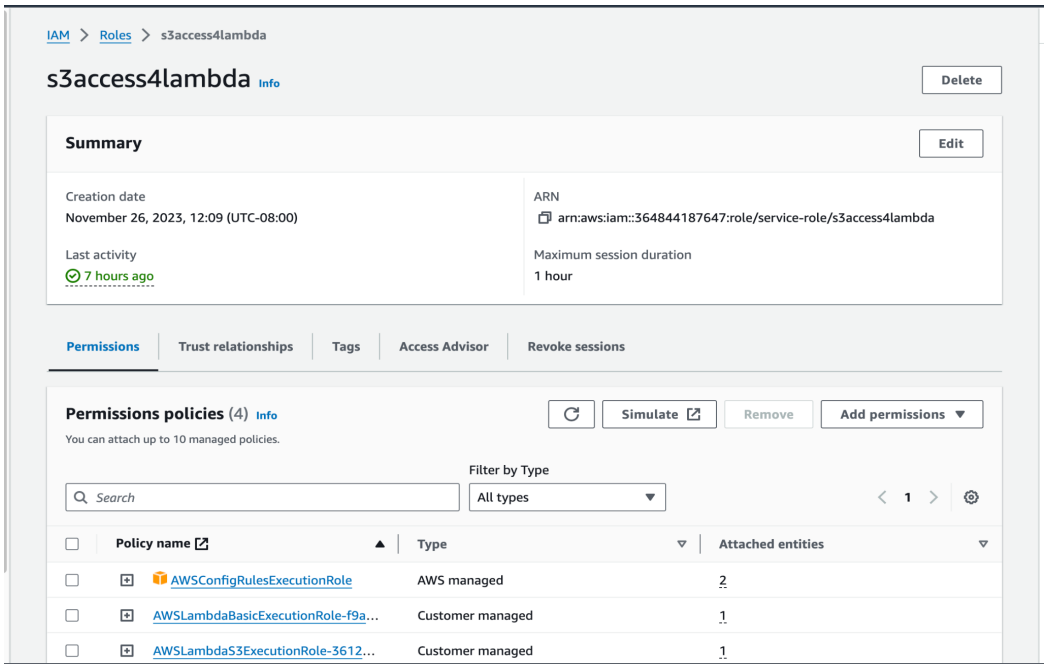
4.4.5 Connecting the Components using IAM

In order to control access to AWS services and resources, AWS Identity and Access Management, or IAM, is essential. It enables the development of roles, groups, and users with certain access rights to resources. By specifying exact access controls and roles, IAM makes sure that every part of the application may communicate with one another securely. For safe access to

other AWS services like DynamoDB or S3, Lambda functions can be allocated IAM roles. The integrity and security of the application depend on this secure access control.

Figure 19

Connecting the Components using IAM

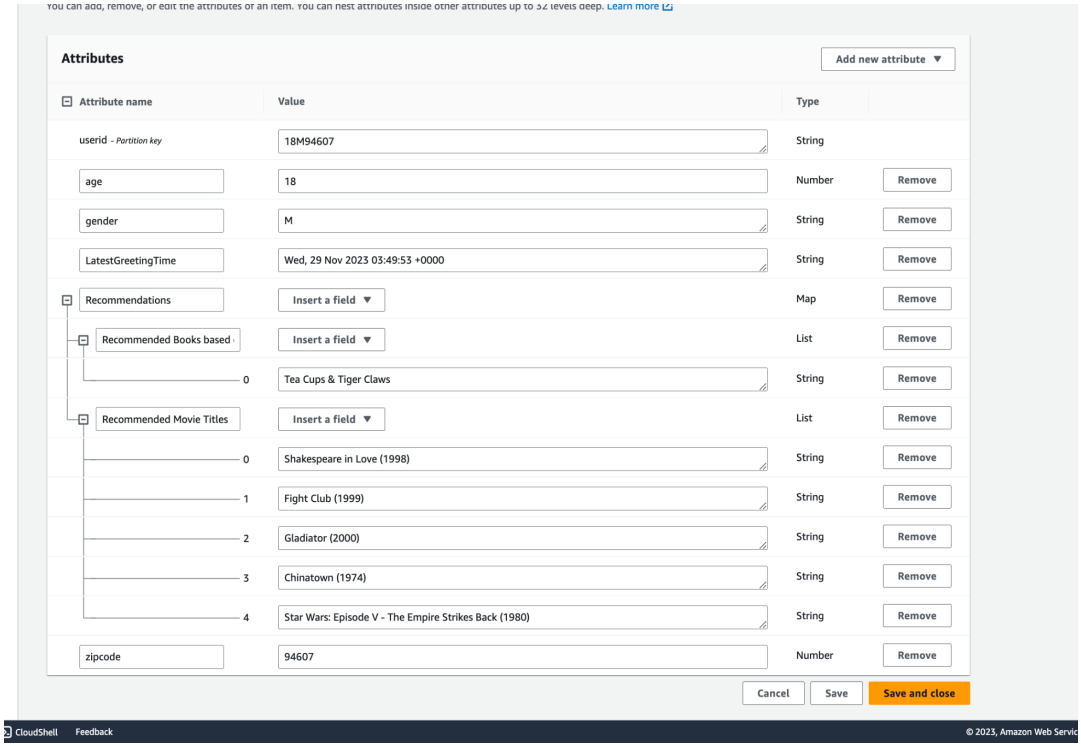


4.4.6 Storing the data in DynamoDB

NoSQL database services like Amazon DynamoDB are renowned for their scalability and low latency. It's the perfect option for online and mobile applications that need reliable, single-digit millisecond response times because it can manage high-traffic, large-scale applications. Because DynamoDB is serverless, it scales up and down automatically based on the demands of the application without the need for human involvement. Key-value and document data structures are supported, providing flexibility in the storing and retrieving of data. Because of this, it can be used for a variety of purposes, including gaming, IoT, and mobile applications.

Figure 20

Storing the data in DynamoDB



Chapter 5. Data Visualization and analysis

The next step involves analyzing the cleaned and transformed data through visualizations. The creation of visualizations is used to analyze the user ratings by the users with movie genres, movie titles, users' occupations, gender, etc. So, for this, we have used Amazon QuickSight for our analysis. This service empowers users to create and share interactive dashboards that include machine learning insights. With Amazon QuickSight, these dashboards can be accessed from any device and seamlessly embedded into various applications, portals, and websites. Leveraging the speed and scalability of the cloud, Amazon QuickSight enables businesses to gain valuable insights and make data-driven decisions in a user-friendly and accessible manner.

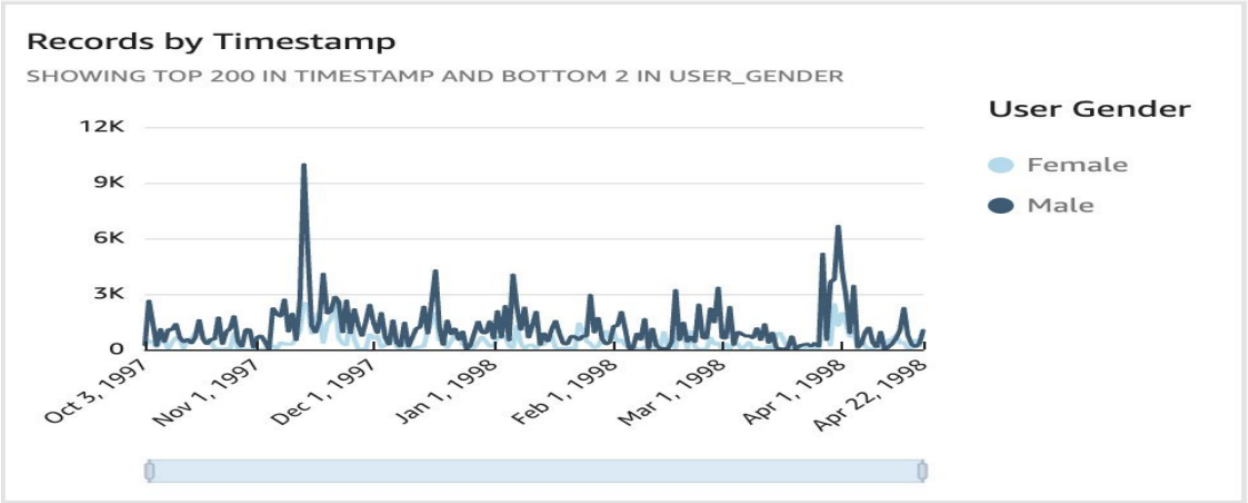
To establish a seamless connection between QuickSight and AWS Athena, we configure QuickSight with the necessary information. This includes specifying the Server, Port, and the S3 Staging Directory, which serves as the storage location for the queried data in S3. Furthermore, we provide the user's Access Key ID to ensure secure and authorized access to the data. By setting up this connection, QuickSight gains direct access to the transformed and queried data, enabling users to create insightful visualizations and dashboards.

5 Visualizations

5.1.1 Records By TimeStamp

Figure 21

Line Chart Showing Analysis of Records by TimeStamp

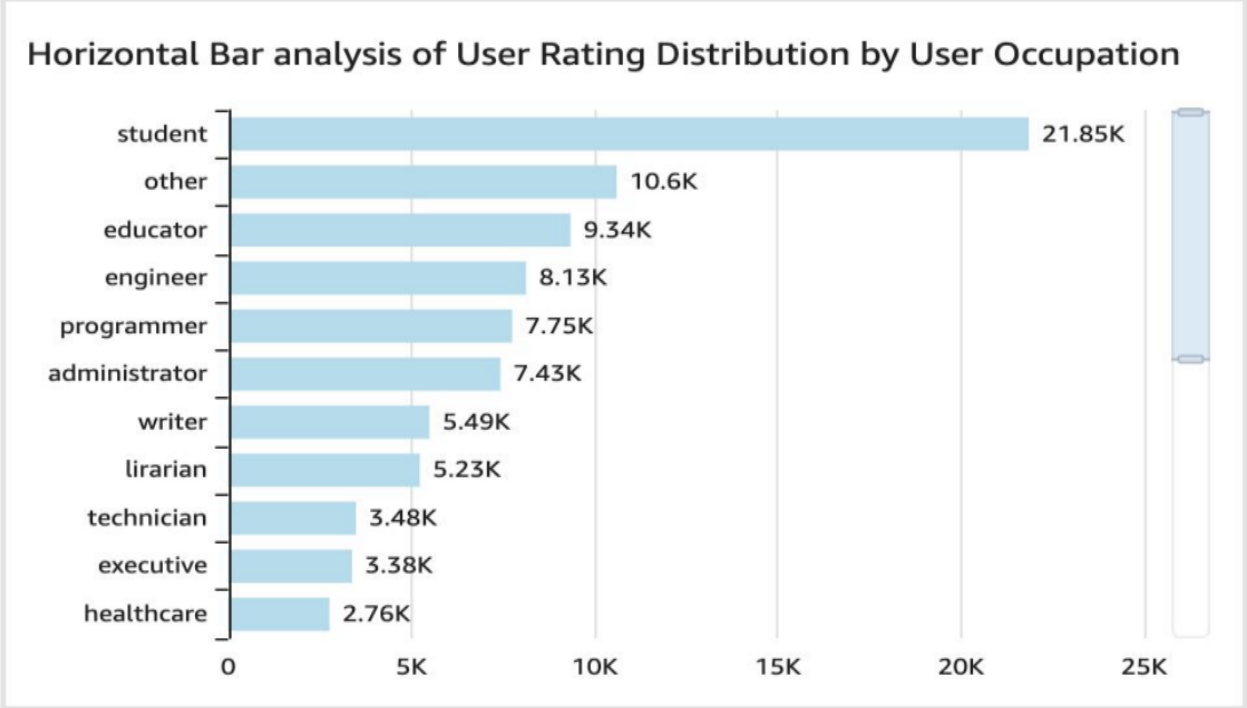


The provided Figure displays a Line Chart showing the timestamp records of the top 200 users, categorized by gender. On Nov 15, 1997, there was a significant peak, with a user rating of 214 for females and 357 for males. Conversely, the lowest peak occurred on April 20, 1998, with a rating of 6 for female users and 39 for male users.

5.1.2 Analysis of User Rating Distribution by User Occupation

Figure 22

Horizontal Bar Chart: Analysis of User Occupation Rating Distribution

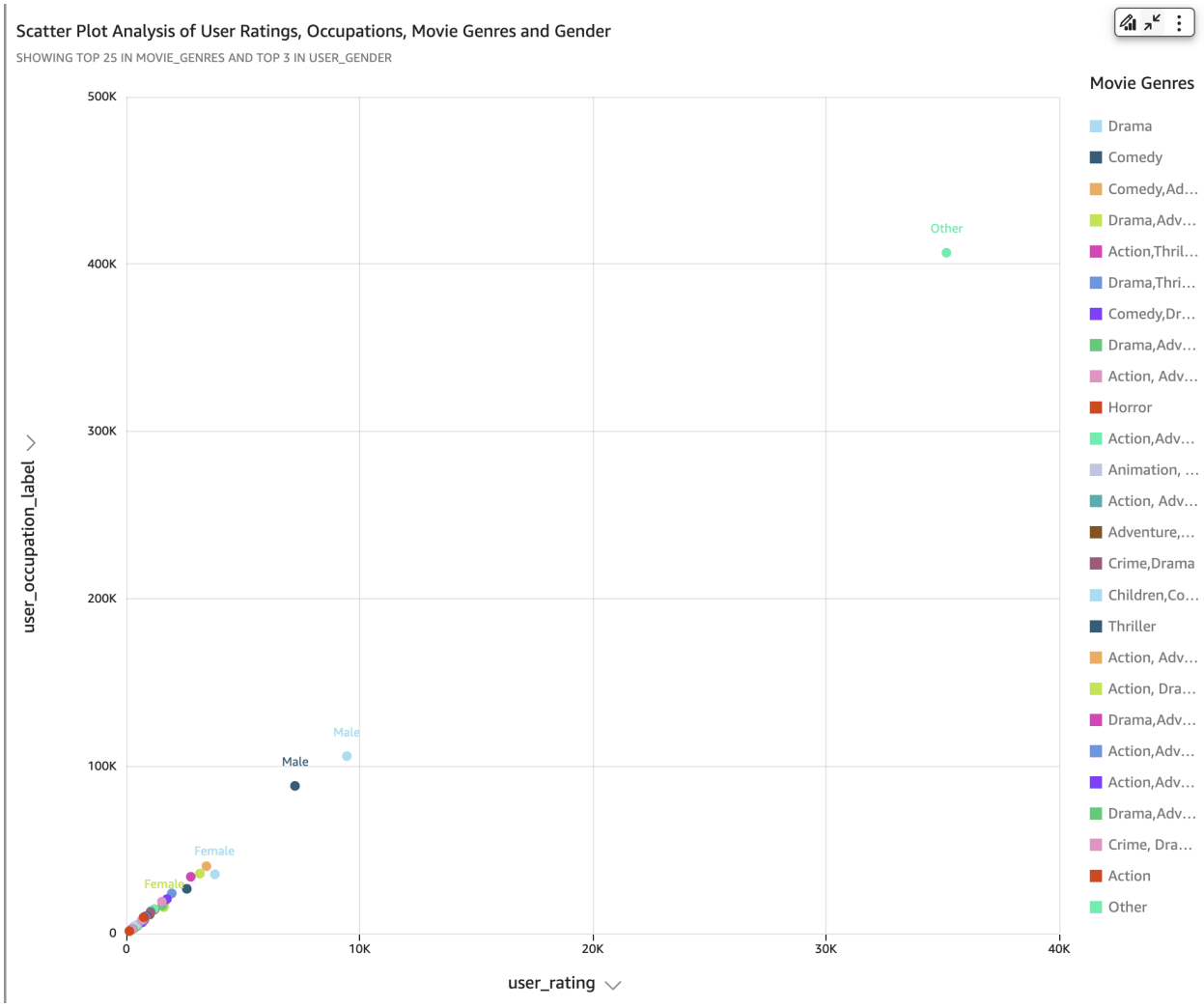


The figure above illustrates a Horizontal Bar Chart presenting the user occupation rating distribution analysis. The chart encompasses 25 different occupation categories. Among these categories, "Student" stands out with the highest rating of 21.85K, while "Homemaker" has the lowest rating of 0.3K. This visual representation provides an overview of the relative ratings assigned to each occupation category, allowing for easy comparison and identification of the highest and lowest-rated occupations.

5.1.3 Analysis of User Ratings, Occupation, Movie Genre and Gender

Figure 23

Scatter Plot: Multi-Factor User Rating Analysis



The provided Figure presents a Scatter Plot depicting user occupation ratings across 26 movie genres. The category "Other" exhibits the highest count, with ratings reaching 35,167 and

an associated occupation label of 4,06,811. Conversely, the lowest rating range falls within the 0-10K range. This visualization allows for a clear understanding of the distribution of ratings across different movie genres and provides insights into the occupation preferences of users.

5.1.4 Zip Code-Based User Rating Distribution

Figure 24

Geographic User Rating Distribution

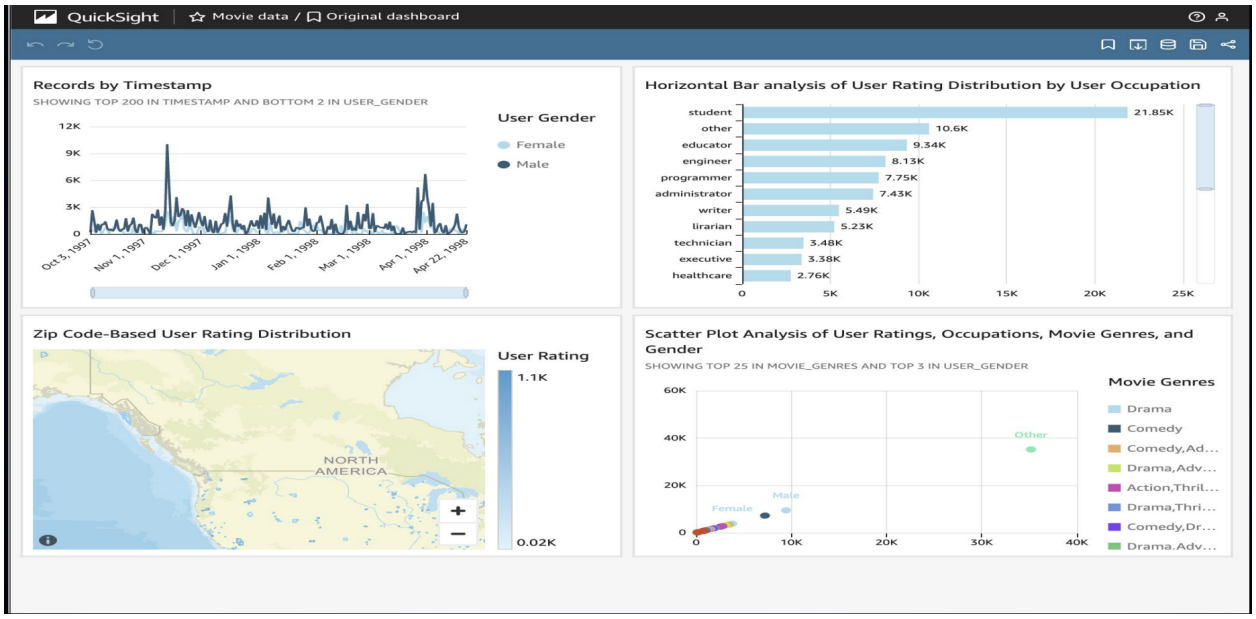
The Figure displayed above represents a geospatial map illustrating the distribution of zip codes based on user ratings. Each state is labeled with its corresponding zip codes and their respective counts. In total, there are approximately 794 zip codes included in the map, each with

its own user rating count. For instance, the zip code 93555 has a user rating count of 297. Similar counts are provided for each zip code, allowing for an overview of the distribution and concentration of user ratings across different geographic areas.

5.1.5 DashBoard

Figure 25

Overall DashBoard for Personalized Movie and Book Recommendations



Welcome to the dashboard! from Figure, you will find insightful visualizations that provide valuable information about user behavior, preferences, and geographic patterns. The timestamp records visualization shows peak user ratings by gender, allowing us to identify the most and least active periods. The horizontal bar chart analyzes user occupation ratings across 25 categories, highlighting the occupations with the highest and lowest ratings. The scatter plot explores user ratings across 26 movie genres, revealing preferences and popular categories.

Lastly, the geospatial map displays user ratings across 794 zip codes, giving insights into geographic distribution.

Chapter 6. Conclusion and Future Work

6.1 Project summary

Conducted an analysis of the dataset and identified the key descriptive features necessary for the book and movie recommendation system. Then, we identified and implemented suitable algorithms to predict movie and book recommendations based on user inputs. To enhance the understanding of the available data, we also created visualizations. Additionally, developed a user-friendly system that allows users to input their details, enabling us to provide personalized recommendations for movies and books. Through these efforts, we aimed to create an effective and user-centric recommendation system that enhances the overall experience for our users.

6.2 Future work

The project aims to provide personalized movie recommendations based on user demographics. The system tailors recommendations to each user's unique profile by utilizing big data analytics and machine learning. Integrating emerging technologies such as virtual and augmented reality, voice assistants, and others can enhance user interaction and engagement. The system prioritizes chance discoveries in its recommendations to mitigate filter bubbles by considering factors beyond the user's previous behavior. Additionally, the system incorporates multi-faceted profiling, considering variables like mood, recent news, and time of day, in addition to historical preferences. Ethical considerations are given due importance, ensuring responsible and unbiased recommendation practices.

Collaboration

The Project report, Project Plan, Presentation, source code, static website embed code, Tableau Workbook will be uploaded on GitHub repository.

GitHub link: https://github.com/Priyankaakula/Medusa/new/main

References

A. Kaplunovich and Y. Yesha, "Cloud big data decision support system for machine learning on AWS: Analytics of analytics," 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 2017, pp. 3508-3516, doi: 10.1109/BigData.2017.8258340.

R. Ahuja, A. Solanki and A. Nayyar, "Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor," 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 2019, pp. 263-268, doi: 10.1109/CONFLUENCE.2019.8776969.

Trawinski, H. Wimmer and J. Kim, "Anomaly Detection in Intrusion Detection System using Amazon SageMaker," 2023 IEEE/ACIS 21st International Conference on Software Engineering Research, Management and Applications (SERA), Orlando, FL, USA, 2023, pp. 210-217, doi: 10.1109/SERA57763.2023.10197735.

Movie_lens dataset:

https://www.tensorflow.org/datasets/catalog/movie_lens#movie_lens1m-ratings

User Dataset:

https://www.kaggle.com/datasets/justinnguyen0x0x/best-books-of-the-21st-century-dataset