Cloud Based Generalized Data Analysis Platform

Hymavathi Gummudala ,Shreyas Aswar, Siddhesh Chavan

Introduction

In the dynamic realm of data analysis, the preliminary step of descriptive analysis serves as the bedrock for informed decision-making and strategic planning. Traditional methods of conducting descriptive analysis, often a manual endeavor, impose significant time constraints, with the process typically demanding 4-5 minutes per dataset. Acknowledging the imperative for efficiency and accessibility in data analysis, our project pioneers a groundbreaking solution – a cloud-based, generalized data analysis platform.

The inherent challenge in manual descriptive analysis lies in its labor-intensive nature, limiting the pace at which valuable insights can be extracted from data. Recognizing this bottleneck, our project is poised to transform the landscape by introducing a platform designed to streamline and expedite the entire process. By harnessing the power of cloud computing, we aim to redefine how data analysis is approached, making it more efficient, accessible, and user-friendly.

Our solution revolves around the development of a cloud-based platform that caters to the needs of both tech-savvy and non-technical users. The user-friendly interface ensures that individuals with varying degrees of technical expertise can effortlessly engage with the platform. The core simplicity lies in the streamlined process – users merely need to upload a CSV file, and the platform takes care of the rest. This democratization of data analysis, accessible from anywhere, is a cornerstone of our project.

The technological backbone of our platform rests on Amazon Web Services (AWS), utilizing the secure and scalable S3 bucket for data storage. To process the data efficiently, we employ Spark and SparkML, leveraging their capabilities to deliver insightful descriptive statistics in a fraction of the time it takes through manual methods. This innovation not only accelerates the data analysis workflow but also marks a departure from the exclusive domain of data experts, welcoming a broader audience into the realm of data-driven insights.

Our cloud-based generalized data analysis platform represents a paradigm shift in the way descriptive analysis is conducted. By embracing cloud computing, we aim to transcend the limitations of manual approaches, ushering in an era of efficiency, accessibility, and accelerated insights. This project is not just a technological advancement; it is a step towards democratizing data analysis, empowering individuals across various domains to actively participate in the process of deriving meaningful insights from their data.

Problem Description

Description Of Data

In our project, we have integrated two user-defined functions designed to streamline the data analysis process. The first function handles descriptive statistics, offering three output types: values, data frames, and figures. Values are collected and appended to an empty JSON, while data frames are passed as arguments to subsequent functions. This function not only deals with encoding issues that may arise when reading files but also addresses columns containing special characters, ensuring smoother processing. It provides crucial information such as dataset details, quantitative and qualitative column names, and even generates figures like chi-square tests,

correlation matrices, histograms for quantitative variables, and bar graphs for qualitative ones. Ultimately, it consolidates all results into a final data frame and JSON.

The second function operates on data frames, addressing missing values, duplicate rows, encoding categorical variables, and performing normalization and scaling. Once all functions have been executed, the final data frame and JSON are returned. Based on the target variable's data type, the system intelligently selects between regression or classification tasks. If the target is a float, regression analysis is performed, while an integer with fewer than 10 categories triggers classification. For string/object targets, the system also engages in classification tasks. The evaluation metrics are then presented, offering a comprehensive and user-friendly approach to data analysis and machine learning.

We have harnessed a diverse set of datasets to conduct a comprehensive data analysis, encompassing both descriptive statistics and machine learning endeavors. During the descriptive statistics phase, we diligently extracted vital information, including the count of missing values, dataset length, the nomenclature of quantitative and categorical columns, and the identification of outliers. For our machine learning tasks, we judiciously selected distinct datasets tailored to different analytical purposes:

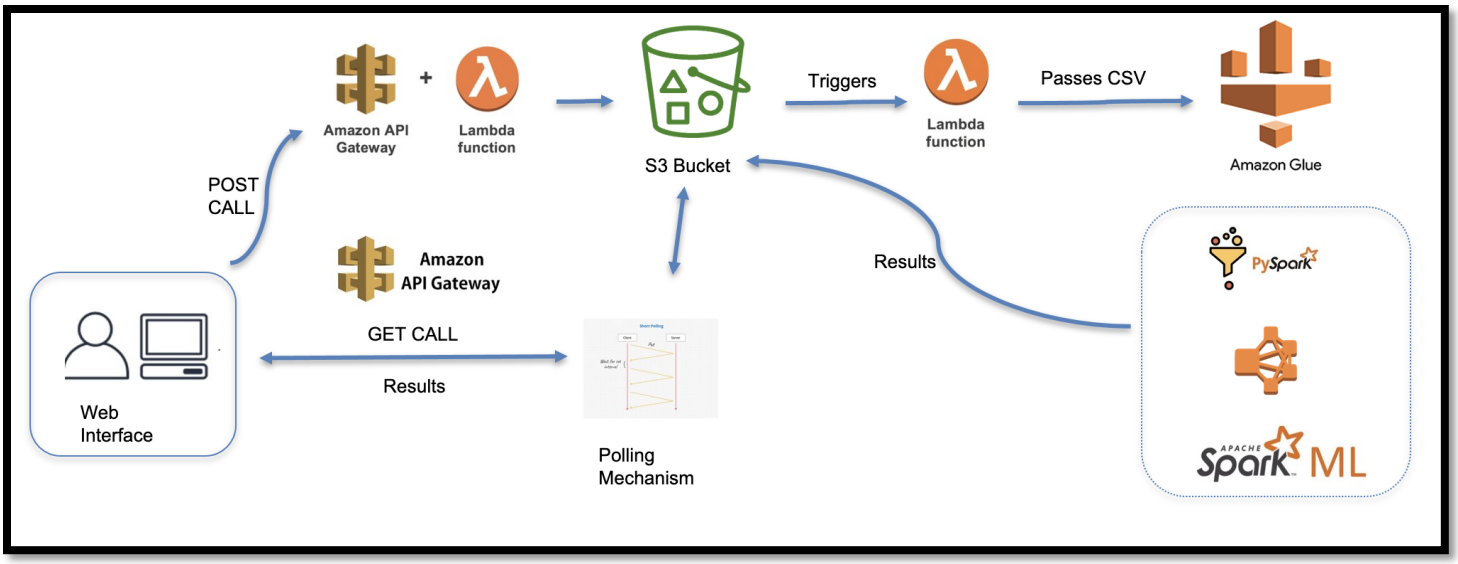
1. The Boston dataset served as the foundation for our regression analysis, allowing us to model and predict continuous numeric outcomes.
2. The Iris dataset, on the other hand, played a pivotal role in our multi-class classification tasks, enabling us to categorize data into multiple distinct classes or categories.
3. The Customer churn dataset was instrumental in our binary classification efforts, facilitating the prediction of customer attrition.
4. The Titanic survived is an example for binary classification in predicting whether a person will die or not.

Methodology

Our methodology revolves around the seamless integration of user-friendly web technologies, Amazon Web Services (AWS), and sophisticated data processing tools to create an efficient and accessible cloud-based data analysis platform. The step-by-step process is designed to optimize the entire workflow, from user input to result retrieval, ensuring a smooth and rapid analytical experience.

User Interface Development:

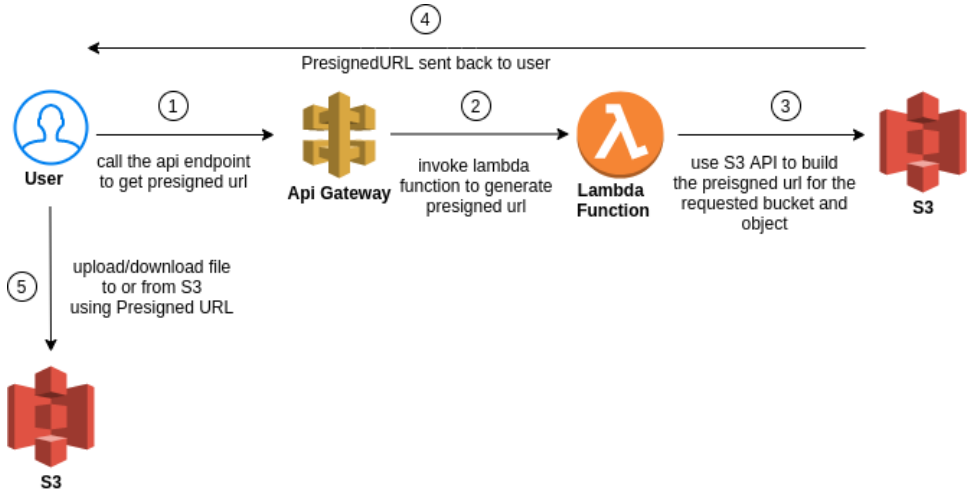
We initiated the project by developing a user-friendly web page using HTML and JavaScript. This interface serves as the entry point for users to upload their CSV files, ensuring simplicity and ease of use for both technical and non-technical individuals.



ARCHITECHTURAL DIAGARM

Data Upload and Storage:

Upon file submission through the web page, we leverage Amazon API Gateway to handle the POST call, triggering a pre-signed URL Lambda function. This Lambda function facilitates the secure storage of the CSV file in an S3 bucket. This approach ensures data integrity and accessibility within the AWS ecosystem.



Data Transformation with Amazon Glue:

After data upload, our platform employs a pivotal step - transforming data using Amazon Glue. This process is initiated when a Lambda function, triggered by new data in an S3 bucket, passes the CSV data to Amazon Glue. Amazon Glue, a serverless data integration service, automates the transformation, ensuring the data is analytics ready. This involves converting CSV formats, cleaning, and structurally organizing the data for further analysis.

The key advantages of using Amazon Glue in this stage include:

Efficiency: Automating the transformation process, reducing manual effort.

Scalability: Handling varying data sizes seamlessly, thanks to its serverless nature.

Resource Optimization: Efficient use of computing resources, scaling as needed, and reducing operational costs.

Preparation for Analysis: Ensuring the data is in the right format and structure for subsequent analysis stages.

In essence, Amazon Glue streamlines our data transformation, making it ready for sophisticated analysis with minimal manual intervention.

Statistical Analysis with AWS Py Spark and SparkML:

The transformed data is then processed using AWS Py Spark, a distributed data processing library, for descriptive statistics. Simultaneously, SparkML, a machine learning library, is employed for predictive analysis. This combination enables us to derive valuable insights, catering to both numeric and categorical data.

Result Storage and Retrieval:

The computed results are stored in the same S3 bucket, organized as JSON objects. For numeric data, the JSON includes descriptive statistics, while for categorical data, it contains RMSE (Root Mean Squared Error) for predictive analysis or Accuracy metrics. This organized structure facilitates clarity and ease of retrieval.

Web Interface Result Retrieval:

To ensure a dynamic and interactive user experience, a web polling mechanism continuously queries the S3 bucket for results using API calls. As soon as the results are available, the polling mechanism fetches the JSON object and delivers the outcomes back to the web interface through API calls. Results are presented to the user as a comprehensive and easily interpretable JSON object.

Result

Our cloud-based data analysis platform delivers rapid and comprehensive insights with a simple CSV upload. Users receive key descriptive statistics, including total rows, distinct count, missing values, mean, and standard deviation. The platform's predictive analysis adapts to the variable type, performing regression for numerical targets and classification for categorical ones. Results are presented in a structured JSON object, offering a quick snapshot of the dataset and, if applicable, Root Mean Squared Error (RMSE) or accuracy metrics. This approach streamlines data interpretation and decision-making. In just minutes, users gain a profound understanding of their data, ushering in a new era of efficient and accessible data analysis.

Option 1 - Choose a CSV/Excel file

Choose file No file chosen Upload

Results

statusCode": 200, "body": "{\"total_rows\": 506, \"column_stats\": {\"crim\": {\"distinct_count\": 504, \"missing_values\": 0, \"mean\": 3.6135235573122535, \"stddev\": 8.601545105332486}, \"zn\": {\"distinct_count\": 26, \"missing_values\": 0, \"mean\": 11.363636363636363, \"stddev\": 23.322452994515153}, \"indus\": {\"distinct_count\": 76, \"missing_values\": 0, \"mean\": 11.136778656126493, \"stddev\": 6.860352940897588}, \"chas\": {\"distinct_count\": 2, \"missing_values\": 0, \"mean\": 0.0691699604743083, \"stddev\": 0.2539940413404103}, \"nox\": {\"distinct_count\": 81, \"missing_values\": 0, \"mean\": 0.5546950592885377, \"stddev\": 0.115877675667556}, \"rm\": {\"distinct_count\": 446, \"missing_values\": 0, \"mean\": 6.284634387351784, \"stddev\": 0.7026171434153233}, \"age\": {\"distinct_count\": 356, \"missing_values\": 0, \"mean\": 68.57490118577073, \"stddev\": 28.148861406903613}, \"dis\": {\"distinct_count\": 412, \"missing_values\": 0, \"mean\": 3.795042687747034, \"stddev\": 2.1057101266276104}, \"rad\": {\"distinct_count\": 9, \"missing_values\": 0, \"mean\": 9.549407114624506, \"stddev\": 8.707259384239368}, \"tax\": {\"distinct_count\": 66, \"missing_values\": 0, \"mean\": 408.2371541501976, \"stddev\": 168.537116054959}, \"ptratio\": {\"distinct_count\": 46, \"missing_values\": 0, \"mean\": 18.455533596837935, \"stddev\": 2.1649455237144397}, \"b\": {\"distinct_count\": 357, \"missing_values\": 0, \"mean\": 356.67403162055217, \"stddev\": 91.29486438415785}, \"lstat\": {\"distinct_count\": 455, \"missing_values\": 0, \"mean\": 12.653063241106715, \"stddev\": 7.141061511348568}, \"medv\": {\"distinct_count\": 229, \"missing_values\": 0, \"mean\": 22.532806324110673, \"stddev\": 9.197104087379817}}, \"categorical_columns\": [], \"quantitative_columns\": [\"crim\", \"zn\", \"indus\", \"chas\", \"nox\", \"rm\", \"age\", \"dis\", \"rad\", \"tax\", \"ptratio\", \"b\", \"lstat\", \"medv\"], \"rows_after_handling_missing\": 506}

```
[ ]  #asking user to input to input
     user_input = input("Enter 'classification', 'regression' ").lower()
```

```
Enter 'classification', 'regression' regression
```

```
#target variable will choose which model to go for:
if "classification" in user_input:
    target = input("Enter the classification target column name: ")
    classification_accuracy = calculate_accuracy(encoded_df, target)
    print("Classification Accuracy:", classification_accuracy)

if "regression" in user_input:
    target = input("Enter the regression target column name: ")
    regression_rmse = calculate_rmse(encoded_df, target)
    print("Regression RMSE:", regression_rmse)
```

```
Enter the regression target column name: medv
Regression RMSE: 3.9524689809083613
```

Discussion:

Evaluated with two independent users (MS Students from IUPUI):

Two independent users, who are master's students from Indiana University–Purdue University Indianapolis (IUPUI), were observed while they performed a set of operations equivalent to those executed by our platform. The focus was to note the time difference between manual execution and platform performance.

For the three datasets evaluated, our platform consistently outperformed manual efforts.

Users manually performing these tasks encountered various issues, such as the absence of necessary libraries and syntax errors, leading to delays in their processes.

Conclusion: Based on the observed efficiency and effectiveness, our platform meets the set expectations and demonstrates its potential as a robust cloud-based solution.

Dataset	User time	Reason for user	Our Platform
BostonHousing	4 mins	Library issue	2 min 4 sec
Titanic	6mins	Column has . In it	2 min 13 sec
Iris	4 mins	No issues	2 min 3 sec

References:

https://www.kaggle.com/datasets/altavish/boston-housing-dataset

https://www.kaggle.com/datasets/himanshunakrani/iris-dataset

https://www.kaggle.com/datasets/hassanamin/customer-churn

https://www.kaggle.com/datasets/yasserh/titanic-dataset

https://docs.aws.amazon.com/AmazonS3/latest/userguide/ShareObjectPreSignedURL.html

https://aws.amazon.com/glue/

Appendix 1: Contributions from each member

Hymavathi Gummudala:

- Project Preliminary Presentation Creation
- Pyspark Code for Descriptive Statistics
- Pyspark Code for Machine Learning
- Report Writing
- Dataset collection

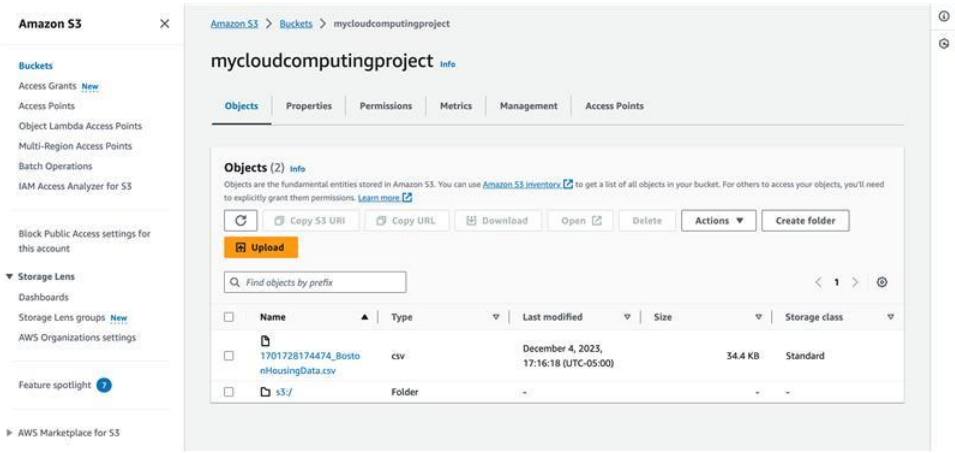
Shreyas Aswar

- AWS setup
- AWS S3 setup
- AWS Lambda Functions
- AWS Glue
- Final Project Presentation
- Project Report Writing
- Pyspark Code for Machine Learning

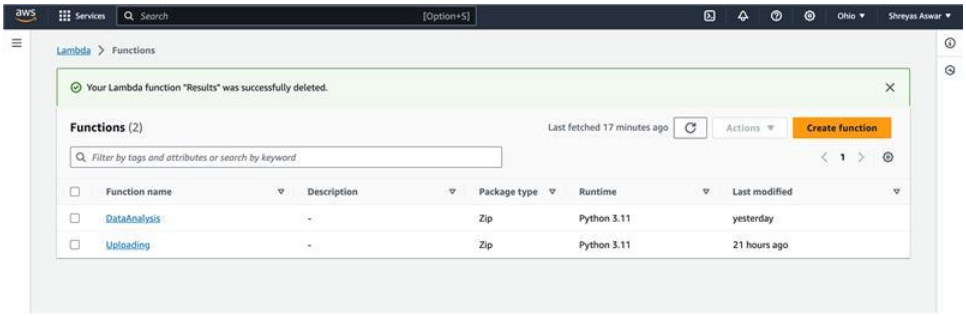
Siddhesh Chavan

- GUI Creation
- Project Report Writing
- Final Project Presentation
- Through Testing of the platform
- Pyspark Code for Machine Learning

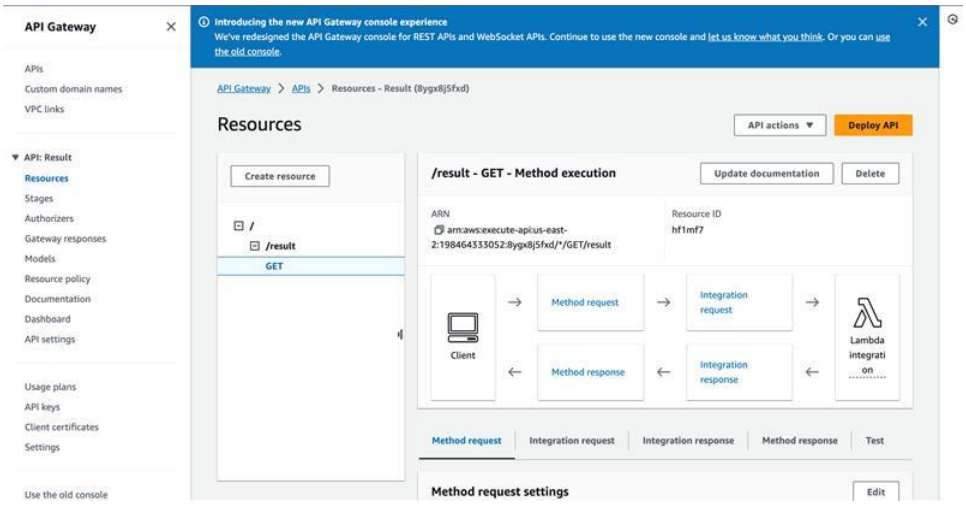
Appendix 2: Code and Configurations



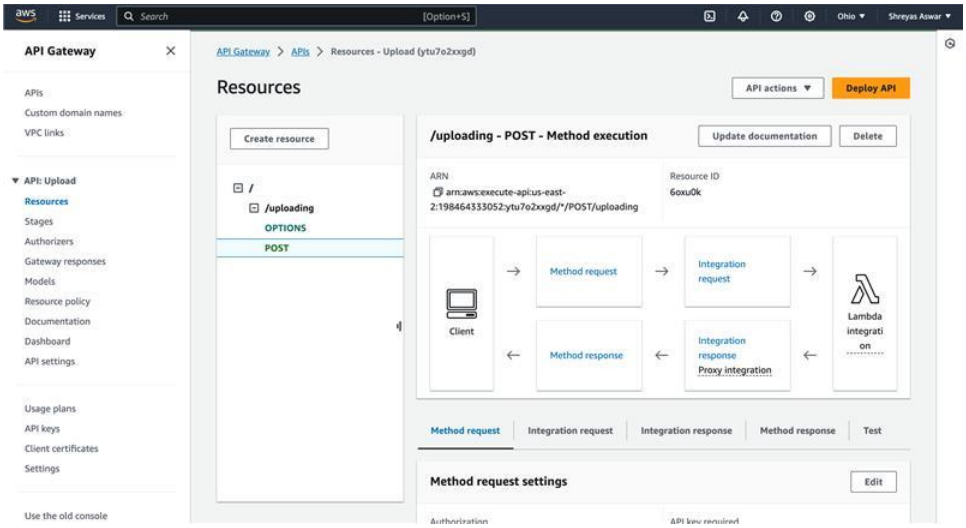
Successful Creation of S3 Bucket



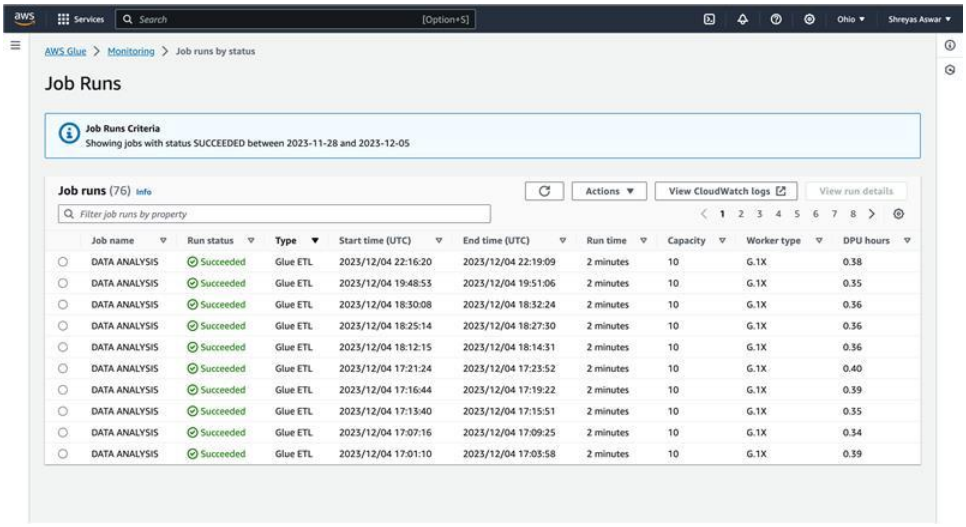
Lambda Functions



GET CALL ON AWS API GATEWAY



POST CALL FOR UPLOADING CSV FILE TO S3 BUCKET



SUCCESSFULL JOB RUNS ON AWS GLUE