# Evaluating Machine Learning Algorithms on Google BigQuery ML

**Submitted by**

Ayush Sharma (0774583)

Jasmeet Singh (0758054)

## Introduction

This project aims to evaluate the performance of two machine learning algorithms, Logistic Regression and K-nearest Neighbours (KNN), utilising Google BigQuery[1] as the primary tool. The primary focus is to assess the capabilities of BigQuery ML in a machine learning context, specifically understanding its functionalities for training, evaluating, and comparing models.

This project endeavours to comprehensively assess BigQuery ML's efficacy by juxtaposing the performances of Logistic Regression and KNN algorithms. Leveraging the human resources dataset[5], encompassing both 10,000 and 1 million entries, the aim is to delve into classification tasks within BigQuery. This dataset is a practical sample to scrutinise how effectively BigQuery handles the intricacies of machine learning processes. The primary ambition lies in deriving significant insights into the platform's efficiency concerning model training, evaluation, and comparative analysis. Through this exploration, the goal is to unearth valuable observations that delineate the strengths, weaknesses, and optimal use cases of BigQuery ML in various machine learning scenarios.

The project delves into a comprehensive evaluation of machine learning algorithms—specifically, Logistic Regression and K-nearest Neighbours (KNN)—within the framework of Google BigQuery. The overarching goal is to appraise the efficacy of BigQuery ML as a robust tool for machine learning endeavours. The focal point involves understanding its capabilities in training, assessing, and comparing models. The project utilises a human resources dataset featuring subsets of 10,000 and 1 million entries, serving as foundational data for conducting classification tasks. These datasets offer a practical lens through which to explore the functionalities of BigQuery in the context of machine learning processes.

Results from the analysis of the 10,000-entry dataset indicate distinct performances. Logistic Regression, when targeting gender, achieved high precision (0.978) while targeting salaries above $100,000 showed lower precision (0.619). KNN exhibited a Davies Bouldin Index of 0.684, indicating decent clustering for this dataset. Meanwhile, the 1 million-entry dataset displayed similar trends but with longer execution times and increased resource consumption, typical with larger datasets.

These findings provide a nuanced understanding of BigQuery ML's strengths and limitations. Logistic Regression showcased varying predictive accuracy based on target variables, while KNN demonstrated consistent performance in clustering tasks. The comparison elucidates resource utilisation and model evaluation metrics, offering valuable insights for determining BigQuery ML's suitability across diverse machine learning scenarios[6]. Ultimately, these results emphasise the platform's potential in handling different scales of data and the nuances of machine learning tasks within it.

## Description

In this project, the primary tools utilised were BigQuery Studio, Looker Studio[3], and Jupyter Notebook, complemented by machine learning algorithms such as Logistic Regression and K-means. BigQuery Studio, powered by BigQuery SQL and BigQuery ML, served as the core platform for conducting machine learning tasks. BigQuery SQL facilitated seamless querying and analysis of large datasets, while BigQuery ML, an integrated machine learning service, provided a versatile environment for building and evaluating machine learning models directly within the BigQuery framework.

Looker Studio played a pivotal role in the project by providing powerful visualisation capabilities[3]. As a business intelligence and data exploration platform, Looker Studio enabled the team to create compelling visual representations of the data, fostering a deeper understanding of the machine learning results and insights generated from BigQuery.

Jupyter Notebook further enhanced the project's analytical capabilities, offering a flexible and interactive environment for data visualisation. This open-source tool allowed for creating dynamic, shareable documents that combined code execution, rich text, and visualisations, facilitating the exploration and communication of findings.

Regarding machine learning algorithms, Logistic Regression and Kmeans were the focal points. Logistic Regression, a widely used classification algorithm, was employed for predictive modelling, particularly in scenarios where the target variable involved binary outcomes. Kmeans, on the other hand, served as a clustering algorithm, contributing to exploring patterns and groupings within the dataset. Together, these tools and algorithms provided a comprehensive framework for assessing the capabilities of BigQuery ML and extracting valuable insights from the human resources dataset.

## Design of the experiment

The experiment was meticulously designed to comprehensively evaluate the capabilities of BigQuery ML by employing two primary machine learning algorithms—Logistic Regression and Kmeans—using the human resources dataset in two scales: 10,000 entries and 1 million entries. The primary objective was to assess how these algorithms perform under varying dataset sizes, focusing on their efficiency, resource utilisation, and performance metrics across different target variables within BigQuery ML.

Initially, the dataset underwent a selection process, ensuring representative samples at varying scales. Preprocessing steps were applied to handle missing values, categorical variables, and feature engineering to ensure the dataset's readiness for analysis. The experiment then delved into implementing Logistic Regression and Kmeans, each with specific configurations and evaluation metrics tailored to the chosen algorithms. Logistic Regression was applied to two target variables—Gender and Salary (>

$100,000)—evaluating performance using precision, recall, accuracy, F1 score, log loss, and ROC AUC metrics. Kmeans, on the other hand, was assessed based on the Davies Bouldin Index and Mean Squared Distance.

The experiment relied on the integrated functionalities of BigQuery Studio, utilising BigQuery SQL[5] for data manipulation and BigQuery ML for model creation, training, and evaluation. Additionally, Looker Studio facilitated the visualisation and interpretation of the results derived from BigQuery ML, enhancing the understanding of the experiment's outcomes. Jupyter Notebook was a flexible and interactive platform for data visualisation, exploratory data analysis, and documentation of the experiment's progress.

Performance evaluation was multifaceted, focusing on the algorithms' predictive capabilities and resource consumption within the BigQuery environment. Iterative refinement was integral, allowing for adjustments in queries, algorithm parameters, and visualisation techniques based on initial findings to optimise model performance and glean deeper insights from the dataset.

Every step, from data preprocessing to algorithm implementation and result interpretation, was methodically documented within the Jupyter Notebooks. This comprehensive documentation allowed for a thorough analysis, enabling comparisons between algorithm performances, resource utilisation, and insights gained, ultimately informing the assessment of BigQuery ML's effectiveness for various machine learning tasks across different dataset scales.

## Result

In examining the results derived from the 10,000-entry dataset, the Logistic Regression models exhibited distinctive performances based on target variables. When considering gender as the target, the model showcased impressive metrics with a precision of 0.978, a recall of 0.746, and an accuracy of 0.863. However, when focusing on predicting salaries greater than $100,000, the precision dropped to 0.619 while achieving full recall. This variation in performance underlines the model's sensitivity to different target variables and their inherent nuances.

Concurrently, the KNN algorithm, configured with parameters like the number of clusters and distance type, demonstrated its own set of metrics, featuring a Davies Bouldin Index of 0.684 and a Mean Squared Distance of 0.174. These values, coupled with execution details such as model creation time and resource consumption, provided a holistic understanding of KNN's performance within the BigQuery ecosystem.

Expanding the analysis to the one million-entry dataset unveiled further insights. Logistic Regression, particularly in predicting salaries exceeding $100,000, maintained a precision of 0.624 and perfect recall while exhibiting an increased execution time and resource utilisation. Simultaneously, the KNN model, fine-tuned with distinct parameters for enhanced clustering, delivered consistent metrics with slightly varying execution times and substantial resource consumption, notably in bytes shuffled.

These observations collectively underscore the variability in algorithm performance concerning dataset sizes and target variables, illuminating the complexities and resource demands associated with model training and evaluation within the BigQuery ML environment. Ultimately, these findings serve as pivotal inputs for gauging the suitability of BigQuery ML for diverse machine learning scenarios, emphasising the trade-offs between execution efficiency and algorithmic performance in a data-rich context.

## Conclusion

Based on the comprehensive evaluation conducted through the experiment, several recommendations and conclusions arise regarding using BigQuery ML and its associated tools for machine learning tasks. The experiment's structured design enabled a nuanced understanding of the platform's capabilities.

Firstly, BigQuery ML showcases significant strengths in its seamless integration within the BigQuery environment. Its ability to conduct machine learning tasks directly within a robust data warehousing platform simplifies the workflow, particularly for organisations heavily reliant on Google Cloud services. Integrating BigQuery SQL streamlines data manipulation and model creation, offering a unified ecosystem for data analysis and machine learning.

Moreover, the experiment demonstrated BigQuery ML's efficiency in handling large-scale datasets, providing valuable insights into resource consumption and execution times. However, it's important to note that with larger datasets, resource consumption escalates, impacting both time and potential costs. Organisations considering BigQuery ML should weigh the trade-offs between computational resources, execution times, and associated expenses, especially when dealing with substantial volumes of data.

Regarding specific algorithms, Logistic Regression exhibited varying performance based on target variables. While its precision for certain classifications was high, its effectiveness fluctuated, especially in scenarios targeting specific salary thresholds. Kmeans showcased consistency in clustering tasks, presenting a robust option for pattern identification within datasets.

The combination of Looker Studio for visualisation and Jupyter Notebook for exploratory analysis enriched the experimentation process, offering robust tools for interpreting and documenting results. This combination enhanced the understanding of machine learning outcomes and facilitated the iterative refinement of models.

Ultimately, adopting BigQuery ML hinges on the organisation's reliance on Google Cloud services, the scale of data to be analysed, and the specific machine learning tasks at hand. For businesses already invested in Google Cloud, leveraging BigQuery ML offers a streamlined approach to machine learning tasks within a familiar ecosystem. However, for smaller-scale operations or those with limited cloud infrastructure, the potential cost implications of resource consumption should be carefully considered before opting for this platform.

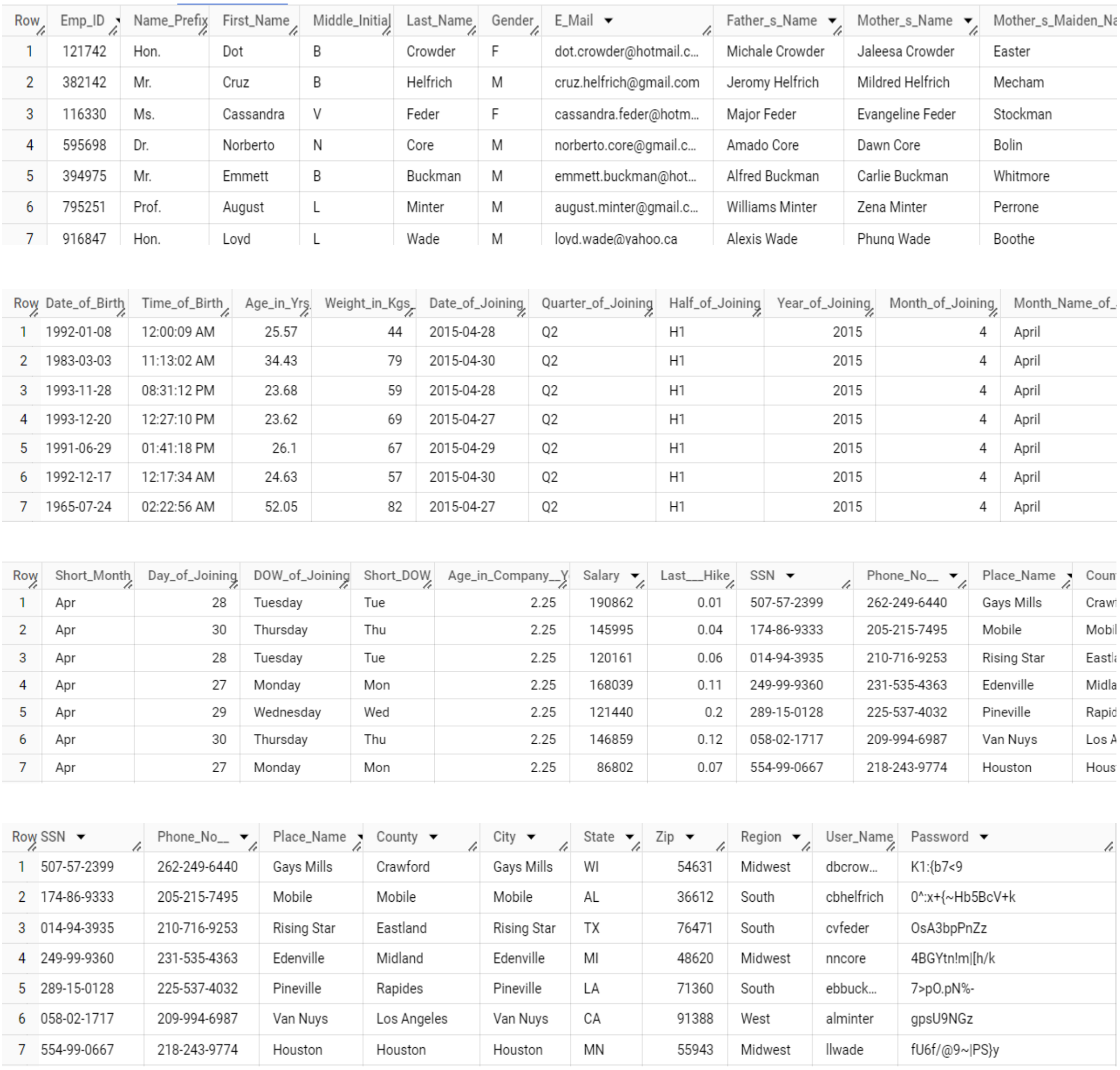
## References

1. BigQuery documentation. (n.d.). Google Cloud. https://cloud.google.com/bigquery/docs/
2. Introduction to BigQuery ML. (n.d.). Google Cloud. https://cloud.google.com/bigquery/docs/bqml-introduction
3. Looker | Google Cloud. (n.d.). Google Cloud. https://cloud.google.com/looker/docs/
4. Verma, Vijay A. “Downloads 16 - Sample CSV Files / Data Sets for Testing - Human Resources (5 Million Records).” *Excel BI Analytics*, 12 Aug. 2017, https://excelbianalytics.com/wp/downloads-16-sample-csv-files-data-sets-for-testing/ Accessed 15 Dec. 2023.
5. Introduction to SQL in BigQuery. (n.d.-b). Google Cloud. https://cloud.google.com/bigquery/docs/introduction-sql/
6. Nayyar, Anand & Puri, Vikram. (2017). Comprehensive Analysis & Performance Comparison of Clustering Algorithms for Big Data. REVIEW OF COMPUTER ENGINEERING RESEARCH. 4. 54-80. 10.18488/journal.76.2017.42.54.80.

# Appendix

The appendix includes essential supplementary materials, such as code snippets utilised in the experiment and a sample dataset or relevant excerpts showcasing the data used for analysis.

## Dataset



| Row | Emp_ID | Name_Prefix | First_Name | Middle_Initial | Last_Name | Gender | E_Mail | Father_s_Name | Mother_s_Name | Mother_s_Maiden_Na |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 121742 | Hon. | Dot | B | Crowder | F | dot.crowder@hotmail.c... | Michale Crowder | Jaleesa Crowder | Easter |
| 2 | 382142 | Mr. | Cruz | B | Helfrich | M | cruz.helfrich@gmail.com | Jeromy Helfrich | Mildred Helfrich | Mecham |
| 3 | 116330 | Ms. | Cassandra | V | Feder | F | cassandra.feder@hotm... | Major Feder | Evangeline Feder | Stockman |
| 4 | 595698 | Dr. | Norberto | N | Core | M | norberto.core@gmail.c... | Amado Core | Dawn Core | Bolin |
| 5 | 394975 | Mr. | Emmett | B | Buckman | M | emmett.buckman@hot... | Alfred Buckman | Carlie Buckman | Whitmore |
| 6 | 795251 | Prof. | August | L | Minter | M | august.minter@gmail.c... | Williams Minter | Zena Minter | Perrone |
| 7 | 916847 | Hon. | Loyd | L | Wade | M | loyd.wade@yahoo.ca | Alexis Wade | Phung Wade | Boothe |

| Row | Date_of_Birth | Time_of_Birth | Age_in_Yrs | Weight_in_Kgs | Date_of_Joining | Quarter_of_Joining | Half_of_Joining | Year_of_Joining | Month_of_Joining | Month_Name_of_ |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1992-01-08 | 12:00:09 AM | 25.57 | 44 | 2015-04-28 | Q2 | H1 | 2015 | 4 | April |
| 2 | 1983-03-03 | 11:13:02 AM | 34.43 | 79 | 2015-04-30 | Q2 | H1 | 2015 | 4 | April |
| 3 | 1993-11-28 | 08:31:12 PM | 23.68 | 59 | 2015-04-28 | Q2 | H1 | 2015 | 4 | April |
| 4 | 1993-12-20 | 12:27:10 PM | 23.62 | 69 | 2015-04-27 | Q2 | H1 | 2015 | 4 | April |
| 5 | 1991-06-29 | 01:41:18 PM | 26.1 | 67 | 2015-04-29 | Q2 | H1 | 2015 | 4 | April |
| 6 | 1992-12-17 | 12:17:34 AM | 24.63 | 57 | 2015-04-30 | Q2 | H1 | 2015 | 4 | April |
| 7 | 1965-07-24 | 02:22:56 AM | 52.05 | 82 | 2015-04-27 | Q2 | H1 | 2015 | 4 | April |

| Row | Short_Month | Day_of_Joining | DOW_of_Joining | Short_DOW | Age_in_Company__Y | Salary | Last__Hike | SSN | Phone_No__ | Place_Name | Coun |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Apr | 28 | Tuesday | Tue | 2.25 | 190862 | 0.01 | 507-57-2399 | 262-249-6440 | Gays Mills | Crawf |
| 2 | Apr | 30 | Thursday | Thu | 2.25 | 145995 | 0.04 | 174-86-9333 | 205-215-7495 | Mobile | Mobil |
| 3 | Apr | 28 | Tuesday | Tue | 2.25 | 120161 | 0.06 | 014-94-3935 | 210-716-9253 | Rising Star | Eastl |
| 4 | Apr | 27 | Monday | Mon | 2.25 | 168039 | 0.11 | 249-99-9360 | 231-535-4363 | Edenville | Midla |
| 5 | Apr | 29 | Wednesday | Wed | 2.25 | 121440 | 0.2 | 289-15-0128 | 225-537-4032 | Pineville | Rapid |
| 6 | Apr | 30 | Thursday | Thu | 2.25 | 146859 | 0.12 | 058-02-1717 | 209-994-6987 | Van Nuys | Los A |
| 7 | Apr | 27 | Monday | Mon | 2.25 | 86802 | 0.07 | 554-99-0667 | 218-243-9774 | Houston | Hous |

| Row | SSN | Phone_No__ | Place_Name | County | City | State | Zip | Region | User_Name | Password |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 507-57-2399 | 262-249-6440 | Gays Mills | Crawford | Gays Mills | WI | 54631 | Midwest | dbcrow... | K1:{b7<9 |
| 2 | 174-86-9333 | 205-215-7495 | Mobile | Mobile | Mobile | AL | 36612 | South | cbhelfrich | 0^:x+{~Hb5BcV+k |
| 3 | 014-94-3935 | 210-716-9253 | Rising Star | Eastland | Rising Star | TX | 76471 | South | cvfeder | OsA3bpPnZz |
| 4 | 249-99-9360 | 231-535-4363 | Edenville | Midland | Edenville | MI | 48620 | Midwest | nncore | 4BGYtn!m\|[h/k |
| 5 | 289-15-0128 | 225-537-4032 | Pineville | Rapides | Pineville | LA | 71360 | South | ebbuck... | 7>pO.pN%- |
| 6 | 058-02-1717 | 209-994-6987 | Van Nuys | Los Angeles | Van Nuys | CA | 91388 | West | alminter | gpsU9NGz |
| 7 | 554-99-0667 | 218-243-9774 | Houston | Houston | Houston | MN | 55943 | Midwest | llwade | fU6f/@9~\|PS}y |

***Fig.1.*** *Screenshots of the sample dataset* (37 columns and 10k/1 million rows)

## Code: BigQuery SQL and BigQuery ML

Below is the query that returns the first ten rows with all columns from the “employees_10000”.

```
-- Show the records
SELECT * FROM `big-data-project-408020.hr_db.employees_10000` LIMIT 10;
```

Below is the BigQuery SQL code chunk to check for missing values in each column.

```
-- Check for missing values in each column
SELECT
  COUNTIF(Emp_ID IS NULL) AS missing_emp_ID,
  COUNTIF(Name_Prefix IS NULL) AS missing_name_prefix,
  COUNTIF(First_Name IS NULL) AS missing_first_name,
  COUNTIF(Middle_Initial IS NULL) AS missing_middle_initial,
  COUNTIF(Last_Name IS NULL) AS missing_last_name,
  COUNTIF(Gender IS NULL) AS missing_gender,
  COUNTIF(E_Mail IS NULL) AS missing_e_mail,
  COUNTIF(Father_s_Name IS NULL) AS missing_father_s_name,
  COUNTIF(Mother_s_Name IS NULL) AS missing_mother_s_name,
  COUNTIF(Mother_s_Maiden_Name IS NULL) AS missing_mother_s_maiden_name,
  COUNTIF(Date_of_Birth IS NULL) AS missing_date_of_birth,
  COUNTIF(Time_of_Birth IS NULL) AS missing_time_of_birth,
  COUNTIF(Age_in_Yrs_ IS NULL) AS missing_age_in_yrs,
  COUNTIF(Weight_in_Kgs_ IS NULL) AS missing_weight_in_kgs,
  COUNTIF(Date_of_Joining IS NULL) AS missing_date_of_joining,
  COUNTIF(Quarter_of_Joining IS NULL) AS missing_quarter_of_joining,
  COUNTIF(Half_of_Joining IS NULL) AS missing_half_of_joining,
  COUNTIF(Year_of_Joining IS NULL) AS missing_year_of_joining,
  COUNTIF(Month_of_Joining IS NULL) AS missing_month_of_joining,
  COUNTIF(Month_Name_of_Joining IS NULL) AS missing_month_name_of_joining,
  COUNTIF(Short_Month IS NULL) AS missing_short_month,
  COUNTIF(Day_of_Joining IS NULL) AS missing_day_of_joining,
  COUNTIF(DOW_of_Joining IS NULL) AS missing_DOW_of_joining,
  COUNTIF(Short_DOW IS NULL) AS missing_short_DOW,
  COUNTIF(Age_in_Company__Years_ IS NULL) AS missing_age_in_company_years,
  COUNTIF(Salary IS NULL) AS missing_salary,
  COUNTIF(Last___Hike IS NULL) AS missing_last_hike,
  COUNTIF(SSN IS NULL) AS missing_SSN,
  COUNTIF(Phone_No__ IS NULL) AS missing_phone_no,
  COUNTIF(Place_Name IS NULL) AS missing_place_name,
  COUNTIF(County IS NULL) AS missing_county,
  COUNTIF(City IS NULL) AS missing_city,
  COUNTIF(State IS NULL) AS missing_state,
  COUNTIF(Zip IS NULL) AS missing_zip,
  COUNTIF(Region IS NULL) AS missing_region,
  COUNTIF(User_Name IS NULL) AS missing_user_name,
  COUNTIF(Password IS NULL) AS missing_password
FROM
  `big-data-project-part-2.hr_db.employees_1m`;
```

The given SQL query checks for duplicates in the dataset, which was found to be 50 duplicate values:-

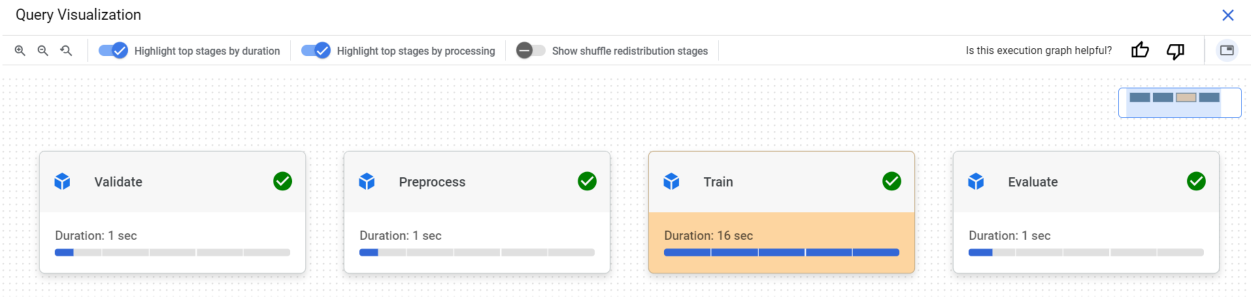
```
-- Check for duplicates
SELECT
  Emp_ID,
  COUNT(*) AS duplicate_count
FROM
  `big-data-project-part-2.hr_db.employees_1m`
GROUP BY
  Emp_ID
HAVING
  COUNT(*) > 1;
```

The duplicates are identified using a subquery that groups by Emp_ID and filters out entries with more than one occurrence. The resulting table contains unique records with 9,886 number of rows.

```
-- Create a new table without duplicates
CREATE OR REPLACE TABLE big-data-project-408020.hr_db.employees_10000 AS
SELECT
  *
FROM
  big-data-project-408020.hr_db.employees_10000
WHERE
  Emp_ID NOT IN (
    SELECT
      Emp_ID
    FROM
      big-data-project-408020.hr_db.employees_10000
    GROUP BY
      Emp_ID
    HAVING
      COUNT(*) > 1
);
```

The provided BigQuery creates a K-means (KNN) clustering model using the BigQuery ML (Machine Learning) service. The clustering uses the K-means algorithm with cosine distance to measure the similarity between data points.

```
-- Create a KNN model
CREATE OR REPLACE MODEL `big-data-project-part-2.hr_db.knn_model`
OPTIONS(
  model_type           = 'kmeans',
  num_clusters         = 3,        -- Number of clusters, adjust as needed
  distance_type        = 'cosine', -- Distance metric, can be 'euclidean' or 'cosine'
  standardize_features = TRUE      -- Standardize input features
) AS
SELECT
  Salary,
  Age_in_Yrs_
FROM
  `big-data-project-part-2.hr_db.employees_1m`;
```



Query Visualization

Highlight top stages by duration

Highlight top stages by processing

Show shuffle redistribution stages

Is this execution graph helpful?

Validate ✓
Duration: 1 sec

Preprocess ✓
Duration: 1 sec

Train ✓
Duration: 16 sec

Evaluate ✓
Duration: 1 sec

***Fig.2.*** *Screenshots of the execution graph while creating a K-means model in BigQuery studio*

Evaluate the performance of the KNN (K-means) model using the BigQuery ML (ML.EVALUATE) functionality. It assesses the model's accuracy, precision, recall, and other metrics by comparing its predictions to the actual values of the specified features.

```
-- Evaluate the KNN (K-means) model
SELECT
  *
FROM
  ML.EVALUATE(MODEL `big-data-project-408020.hr_db.knn_model`,
    (
    SELECT
        Salary,
        Age_in_Yrs_
    FROM
      `big-data-project-408020.hr_db.employees_10000`
    )
  );
```

The given code predicts the cluster assignment using the KNN (K-means) model for a specific employee with a salary of 105125 and an age in years.

```
-- Predict using the KNN (K-means) model
SELECT
  *
FROM
  ML.PREDICT(MODEL `big-data-project-408020.hr_db.knn_model`,
    (
    SELECT
      Salary,
      Age_in_Yrs_
    FROM
      `big-data-project-408020.hr_db.employees_10000`
    WHERE
      Salary = 105125
    LIMIT 1
    )
  );
```

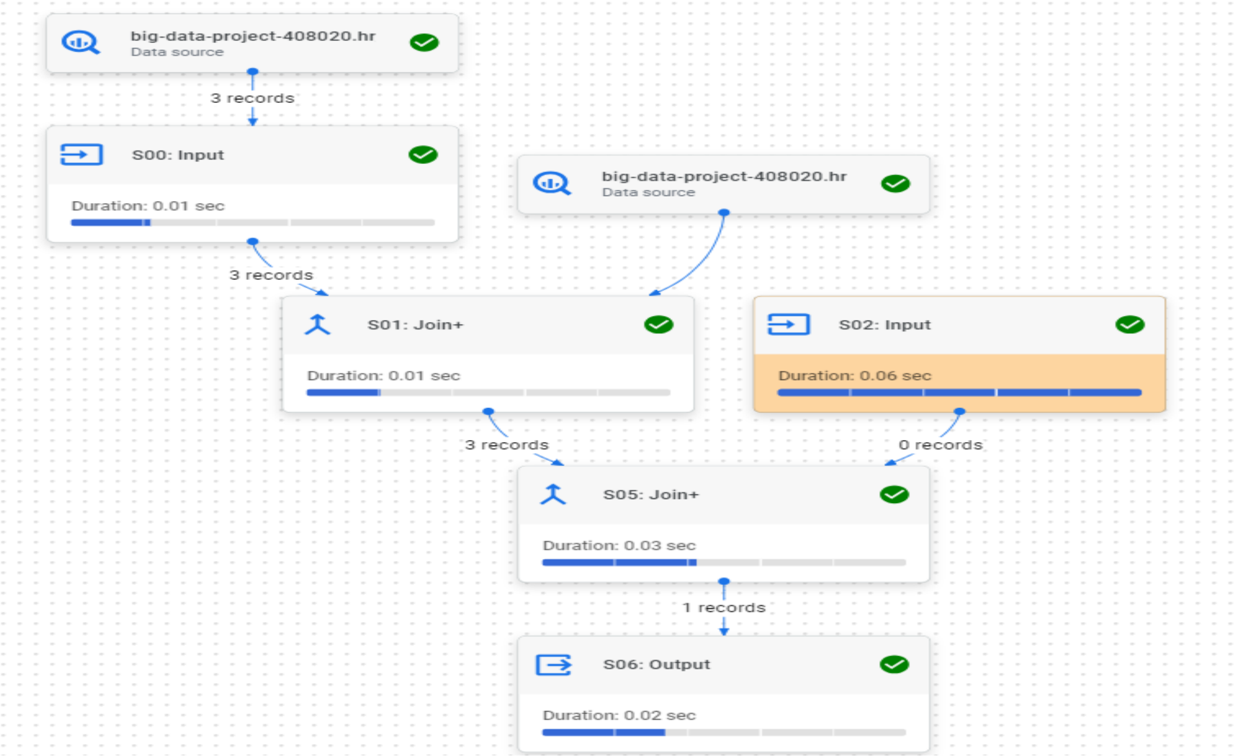
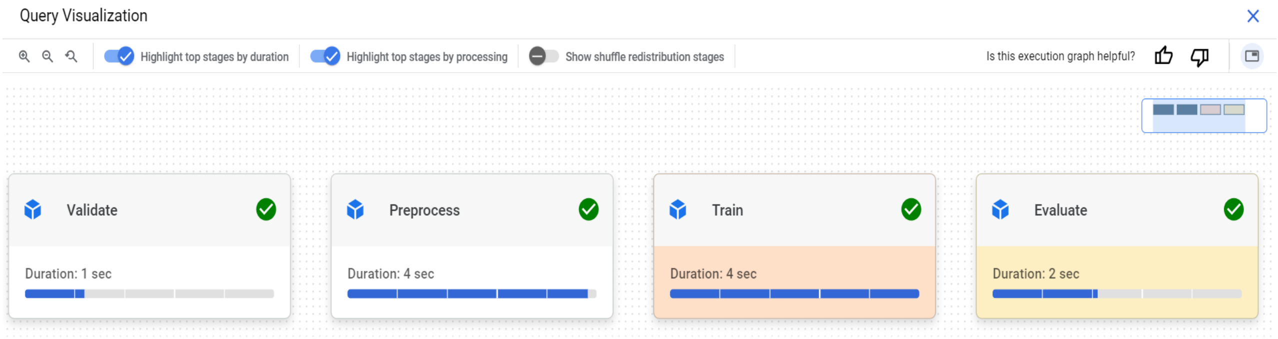


***Fig.3.*** *Screenshots show the execution graph while evaluating a K-means model in BigQuery studio.*

The provided code creates a logistic regression model named logistic_regression_for_salary and predicts whether an individual's salary is above $100,000.

```
-- Create a logistic regression model
CREATE OR REPLACE MODEL `big-data-project-408020.hr_db.logistic_regression_for_salary`
OPTIONS(
  model_type        = 'logistic_reg',
  Input_label_cols = ['Above_100K'],    -- Binary target variable
  l1_reg            = 0.1,              -- L1 regularization parameter
  l2_reg            = 0.1,              -- L2 regularization parameter
  learn_rate_strategy = 'line_search', -- Learning rate strategy
  ls_init_learn_rate  = 0.01,           -- Initial learning rate for optimization
  min_rel_progress    = 0.01,           -- Minimum relative progress for early stopping
  early_stop          = TRUE            -- Enable early stopping
) AS
SELECT
   IF(Salary > 100000, 1, 0) AS Above_100K, -- Create a binary column indicating if
Salary is above $100,000
  Age_in_Yrs_,
  Weight_in_Kgs_
FROM
  `big-data-project-408020.hr_db.employees_10000`;
```


Query Visualization
Highlight top stages by duration
Highlight top stages by processing
Show shuffle redistribution stages
Is this execution graph helpful?

Validate ✓ Duration: 1 sec

Preprocess ✓ Duration: 4 sec

Train ✓ Duration: 4 sec

Evaluate ✓ Duration: 2 sec

***Fig.4.*** *Screenshots of the BigQuery visualisation while creating a logistic regression model in BigQuery studio*

Using the human resources dataset, the provided code evaluates the performance of the logistic regression model named logistic_regression_for_salary.

```
-- Evaluate the logistic regression model
SELECT
  *
FROM
  ML.EVALUATE(MODEL `big-data-project-408020.hr_db.logistic_regression_for_salary`,
    (
    SELECT
      IF(Salary > 100000, 1, 0) AS Above_100K,
      Age_in_Yrs_,
      Weight_in_Kgs_
    FROM
      `big-data-project-408020.hr_db.employees_10000`
    )
  );
```

The provided code predicts the target variable, 'Above_100K' (indicating whether the salary is above $100,000), using the logistic regression model

```
-- Predict using the logistic regression model
SELECT
  *
FROM
  ML.PREDICT(MODEL `big-data-project-408020.hr_db.logistic_regression_for_salary`,
    (
    SELECT
      Age_in_Yrs_,
      Weight_in_Kgs_
    FROM
      `big-data-project-408020.hr_db.employees_10000`
    )
  );
```

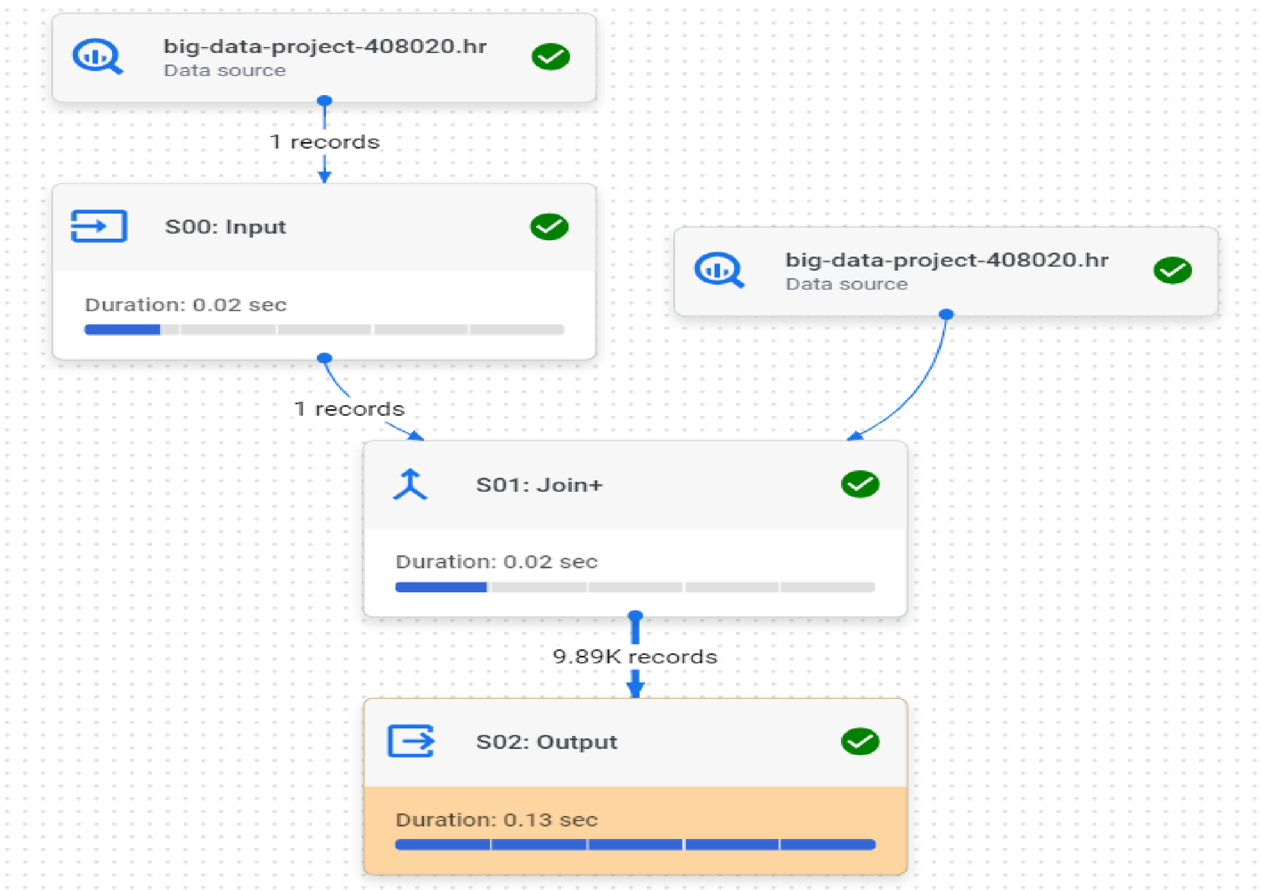


***Fig.5.*** *Screenshots show the execution graph while evaluating a logistic regression model in BigQuery studio.*

The given code creates a logistic regression model using the BigQuery ML service. The model is designed to predict the target variable ('Gender') based on input features such as 'Age_in_Yrs_', 'Weight_in_Kgs_', and 'Salary'.

```
-- Create a logistic regression model
CREATE OR REPLACE MODEL `big-data-project-408020.hr_db.logistic_regression`
OPTIONS(
  model_type          = 'logistic_reg',
  input_label_cols    = ['Gender'],
  l1_reg              = 0.1,           -- L1 regularization parameter
  l2_reg              = 0.1,           -- L2 regularization parameter
  learn_rate_strategy = 'line_search', -- Learning rate strategy
  ls_init_learn_rate  = 0.01,          -- Initial learning rate for optimization
  min_rel_progress    = 0.01,          -- Minimum relative progress for early stopping
  early_stop          = TRUE           -- Enable early stopping
) AS
SELECT
  Gender,
  Age_in_Yrs_,
  Weight_in_Kgs_,
  Salary
FROM
  `big-data-project-408020.hr_db.employees_10000`;
```

Evaluate the performance of a logistic regression model named “logistic_regression”.

```
-- Evaluate the model
SELECT * FROM ML.EVALUATE(MODEL `big-data-project-408020.hr_db.logistic_regression`,
    (
    SELECT
      Salary AS predicted_label,
      Gender,
      Age_in_Yrs_,
      Weight_in_Kgs_,
      Salary
    FROM
      `big-data-project-408020.hr_db.employees_10000`
    )
  );
```

The provided code uses a logistic regression model trained on the human resources dataset to predict the ‘Gender’, ‘Age’ and ‘Salary’ columns.

```
-- Predict using the logistic regression model
SELECT
  * EXCEPT(Gender, Age_in_Yrs_, Weight_in_Kgs_, Salary),
  predicted_Gender
FROM
```

```
ML.PREDICT(MODEL `big-data-project-408020.hr_db.logistic_regression`,
  (
  SELECT
    Gender,
    Age_in_Yrs_,
    Weight_in_Kgs_,
    Salary
  FROM
    `big-data-project-408020.hr_db.employees_10000`
  )
;
```

## Screenshots

Below is the screenshot of **Looker Studio**, which shows the data exploration and visualisation.

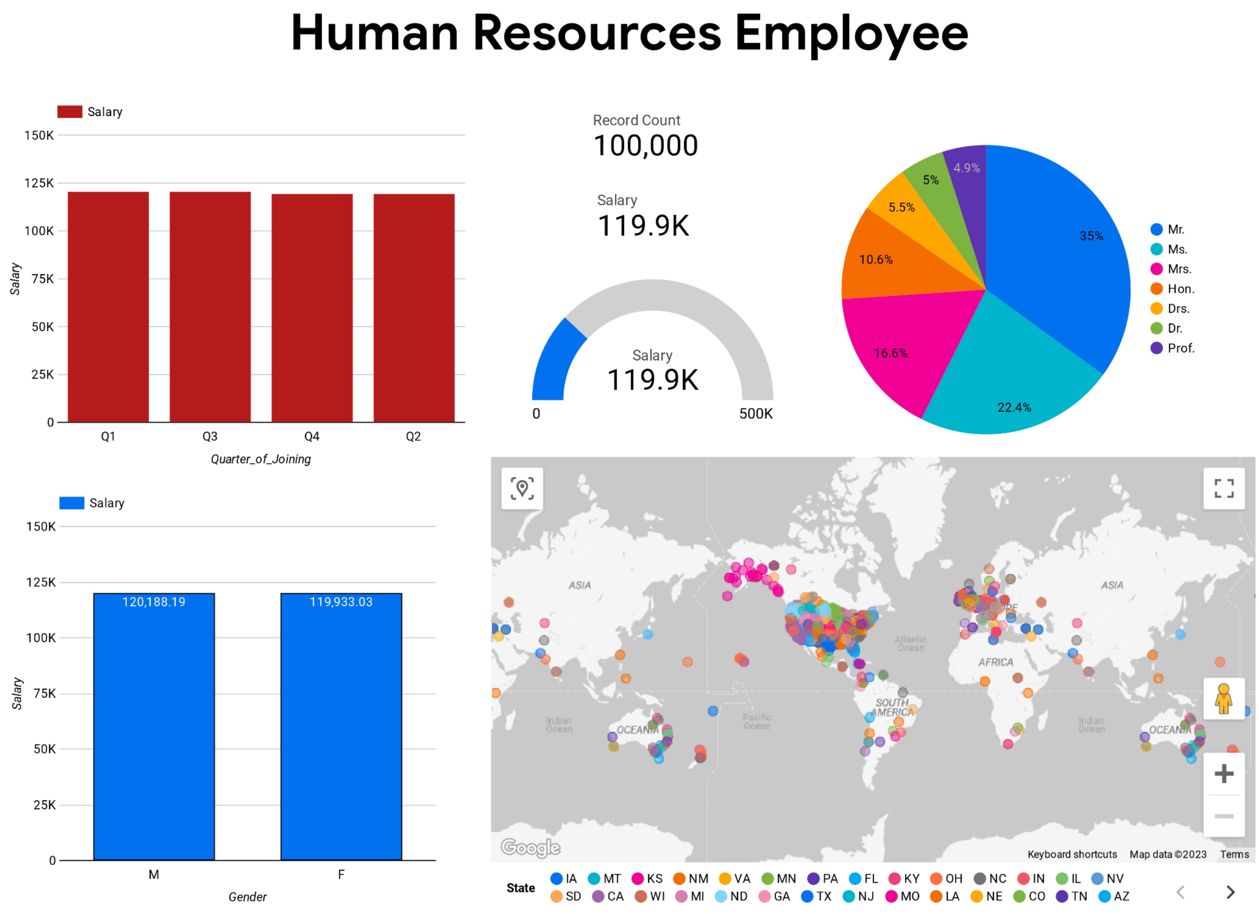
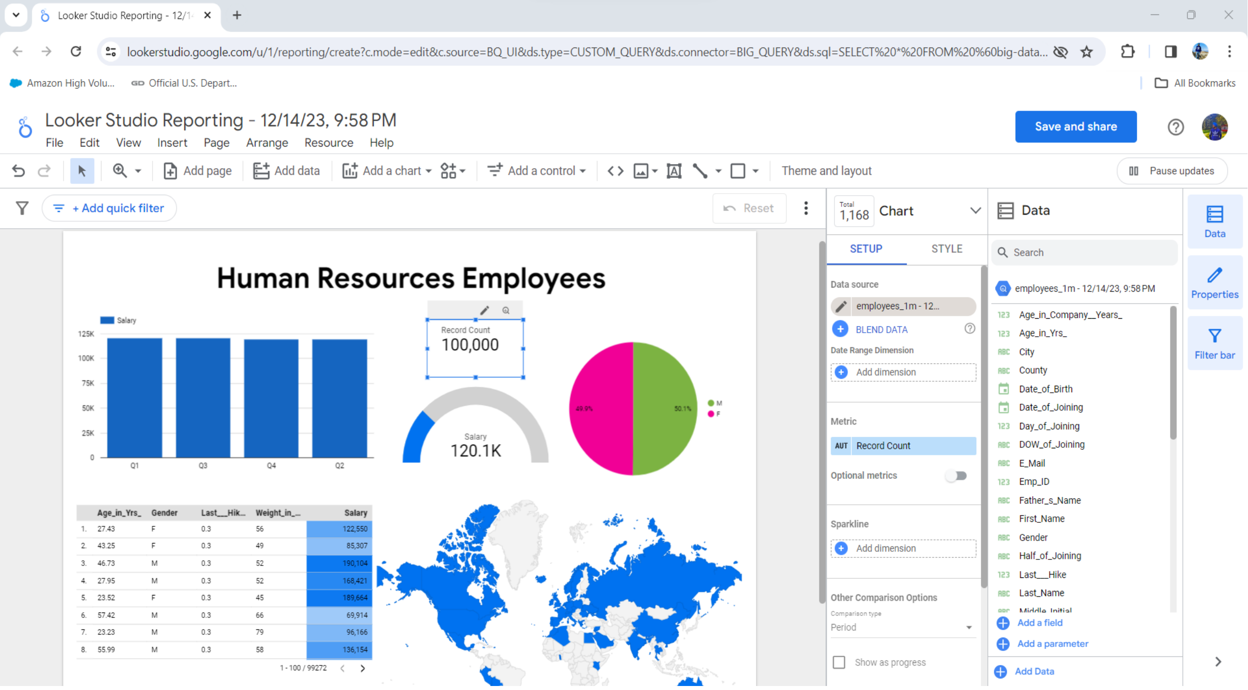


***Fig.6.*** *Screenshots show the* ***interactive dashboards*** *created in the* ***Google Looker Studio****.*

It allows users to create and share interactive dashboards, reports, and visualisations based on the underlying data in their databases.

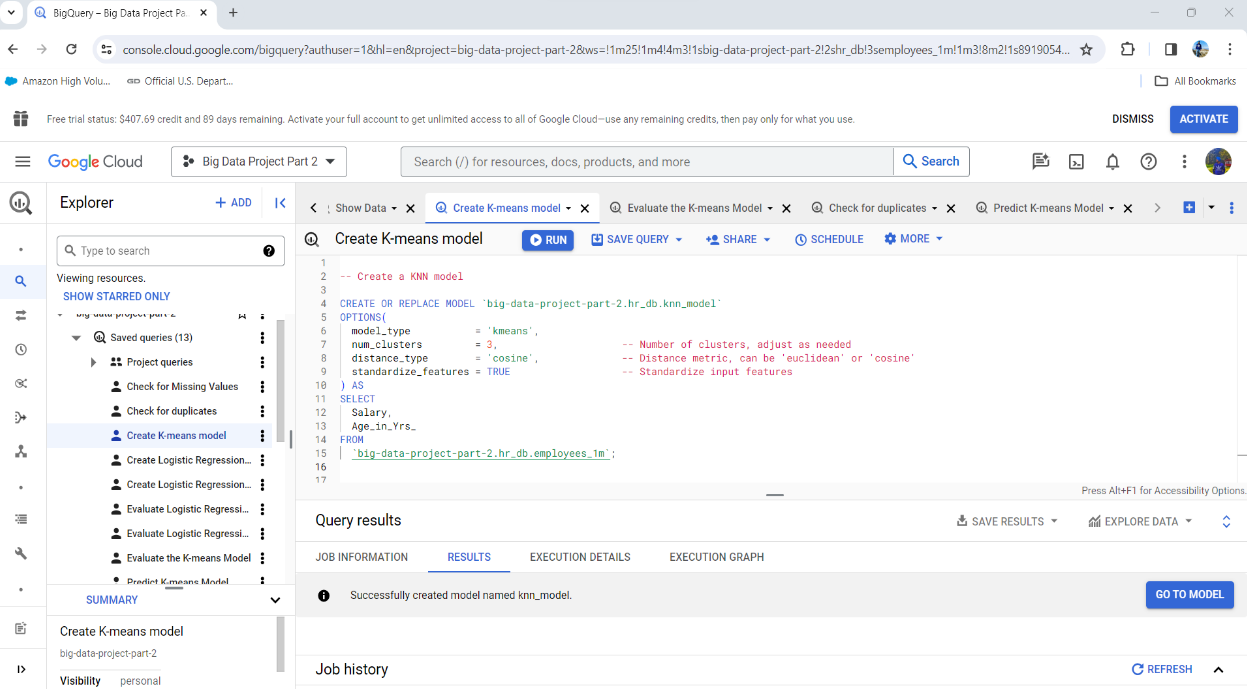


| | Age_in_Yrs_ | Gender | Last__Hik... | Weight_in_... | Salary |
|---|---|---|---|---|---|
| 1. | 27.43 | F | 0.3 | 56 | 122,550 |
| 2. | 43.25 | F | 0.3 | 49 | 85,307 |
| 3. | 46.73 | M | 0.3 | 52 | 190,104 |
| 4. | 27.95 | M | 0.3 | 52 | 168,421 |
| 5. | 23.52 | F | 0.3 | 45 | 189,664 |
| 6. | 57.42 | M | 0.3 | 66 | 69,914 |
| 7. | 23.23 | M | 0.3 | 79 | 96,166 |
| 8. | 55.99 | M | 0.3 | 58 | 136,154 |

1 - 100 / 99272

***Fig.7.*** *Screenshots show the Google Looker Studio platform while visualising the data.*

The screenshot shows the **Google BigQuery Studio's** interface, which provides a visual environment for querying, exploring, and analysing data in BigQuery.



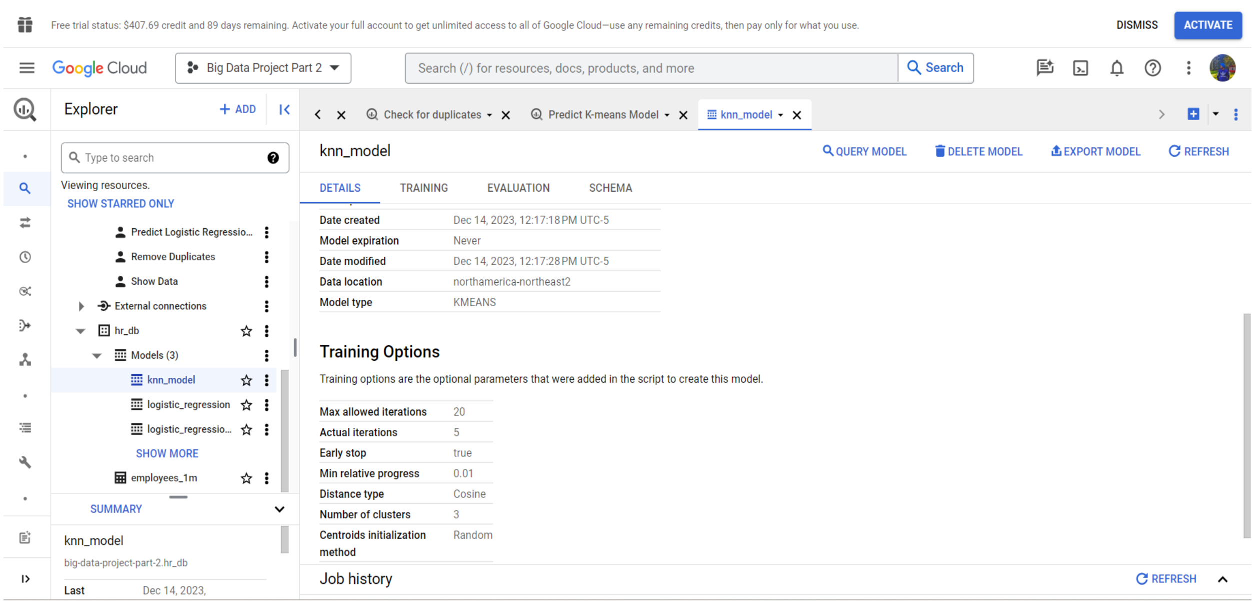
Create K-means model

```

-- Create a KNN model

CREATE OR REPLACE MODEL `big-data-project-part-2.hr_db.knn_model`
OPTIONS(
  model_type          = 'kmeans',
  num_clusters        = 3,                    -- Number of clusters, adjust as needed
  distance_type       = 'cosine',             -- Distance metric, can be 'euclidean' or 'cosine'
  standardize_features = TRUE                 -- Standardize input features
) AS
SELECT
  Salary,
  Age_in_Yrs_
FROM
  `big-data-project-part-2.hr_db.employees_1m`;

```

Successfully created model named knn_model.

***Fig.8.*** *Screenshots show the Google BigQuery studio platform while creating a K-means model.*

The screenshot shows the KNN model created with different **option parameters**.



Free trial status: $407.69 credit and 89 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.

knn_model

| | |
|---|---|
| Date created | Dec 14, 2023, 12:17:18 PM UTC-5 |
| Model expiration | Never |
| Date modified | Dec 14, 2023, 12:17:28 PM UTC-5 |
| Data location | northamerica-northeast2 |
| Model type | KMEANS |

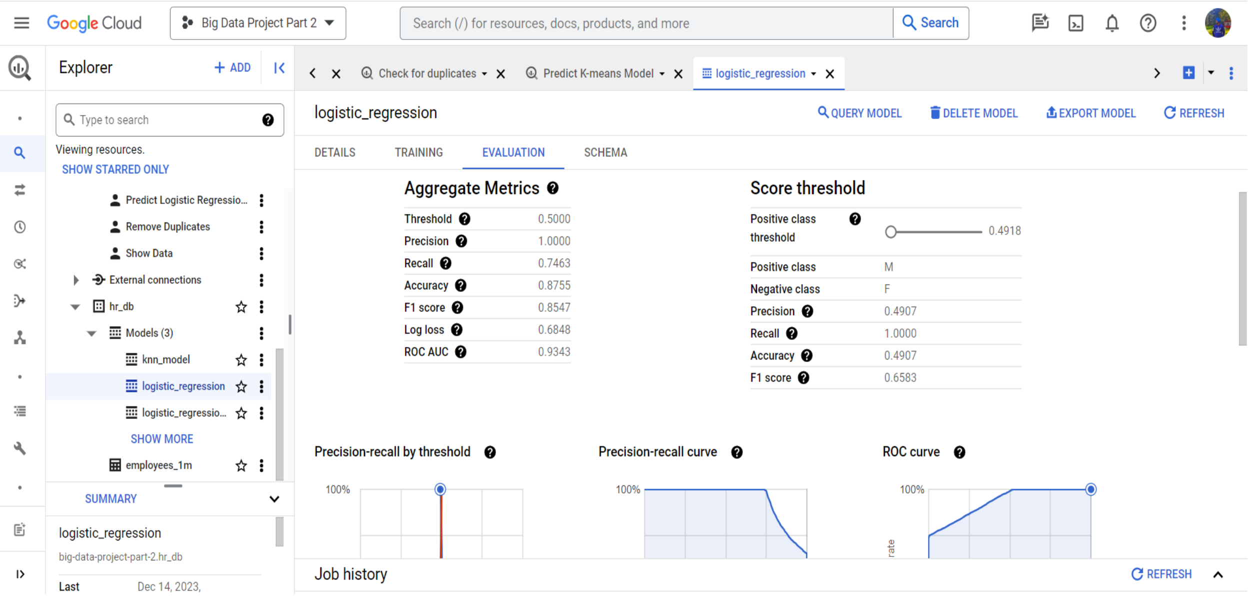
Training Options

Training options are the optional parameters that were added in the script to create this model.

| | |
|---|---|
| Max allowed iterations | 20 |
| Actual iterations | 5 |
| Early stop | true |
| Min relative progress | 0.01 |
| Distance type | Cosine |
| Number of clusters | 3 |
| Centroids initialization method | Random |

***Fig.9.*** *Screenshots show the details of the K-means model when it was created in Google BigQuery studio.*

Below is the logical regression screenshot, showing the **evaluation aggregate metrics and score threshold**.



logistic_regression

Aggregate Metrics

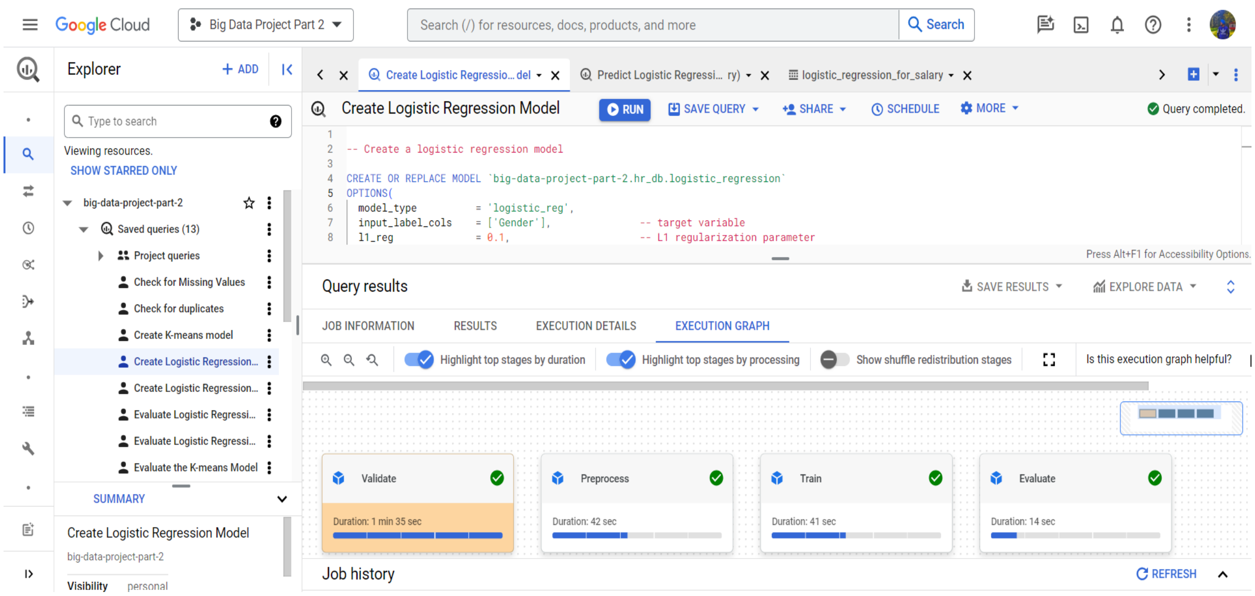
| | |
|---|---|
| Threshold | 0.5000 |
| Precision | 1.0000 |
| Recall | 0.7463 |
| Accuracy | 0.8755 |
| F1 score | 0.8547 |
| Log loss | 0.6848 |
| ROC AUC | 0.9343 |

Score threshold

| | |
|---|---|
| Positive class threshold | 0.4918 |
| Positive class | M |
| Negative class | F |
| Precision | 0.4907 |
| Recall | 1.0000 |
| Accuracy | 0.4907 |
| F1 score | 0.6583 |

***Fig.10.*** *Screenshots show the evaluation metrics such as aggregate metrics, threshold score and ROC curve of the K-means model in Google BigQuery studio.*

Below is the screenshot of the **execution graph** in Google BigQuery Studio showing the creation of the logistic regression model in the Human resource dataset:-



Google Cloud
Big Data Project Part 2
Search (/) for resources, docs, products, and more
Search
Explorer
+ ADD
Type to search
Viewing resources.
SHOW STARRED ONLY
big-data-project-part-2
Saved queries (13)
Project queries
Check for Missing Values
Check for duplicates
Create K-means model
Create Logistic Regression...
Create Logistic Regression...
Evaluate Logistic Regressi...
Evaluate Logistic Regressi...
Evaluate the K-means Model
SUMMARY
Create Logistic Regression Model
big-data-project-part-2
Visibility personal
Create Logistic Regressio...del
Predict Logistic Regressi... ry)
logistic_regression_for_salary
Create Logistic Regression Model
RUN
SAVE QUERY
SHARE
SCHEDULE
MORE
Query completed.
Press Alt+F1 for Accessibility Options.
Query results
SAVE RESULTS
EXPLORE DATA
JOB INFORMATION
RESULTS
EXECUTION DETAILS
EXECUTION GRAPH
Highlight top stages by duration
Highlight top stages by processing
Show shuffle redistribution stages
Is this execution graph helpful?
Job history
REFRESH

```
-- Create a logistic regression model

CREATE OR REPLACE MODEL `big-data-project-part-2.hr_db.logistic_regression`
OPTIONS(
  model_type          = 'logistic_reg',
  input_label_cols    = ['Gender'],              -- target variable
  l1_reg              = 0.1,                     -- L1 regularization parameter
```

Validate
Duration: 1 min 35 sec

Preprocess
Duration: 42 sec

Train
Duration: 41 sec

Evaluate
Duration: 14 sec

***Fig.11.*** *Screenshots show the Logistic regression model evaluation graph in Google BigQuery studio.*