July 27, 2023

**Mahek Kathiriya**

**s2231752**

**Glasgow Caledonian University  
Department of Computing**

**Big Data Landscape (MMI226831) Coursework 2 Re-sit 2022/2023**

Contents

[**Task 1: Creating a Machine Learning Model** 1](#_Toc141331165)

[**Choice of Model:** 2](#_Toc141331166)

[**Creating a Machine Learning Model** 3](#_Toc141331167)

[**Task 2: Demonstration of ML Model Usage with Translation** 4](#_Toc141331168)

[**Step 1: Infusing ML Model into Python Code** 4](#_Toc141331169)

[**Step 2: Translation to Another Language** 6](#_Toc141331170)

[**Task 3: BigQuery ML Model Explainability and Appraisal** 8](#_Toc141331171)

[**BigQuery ML Model Explainability Support:** 8](#_Toc141331172)

[**Enabling Explainable AI and Registering the Model** 9](#_Toc141331173)

[**Running the Modified SQL Code** 9](#_Toc141331174)

[**Deploying the Model to an Endpoint** 10](#_Toc141331175)

[**Comparing Different Model Versions** 10](#_Toc141331176)

[**Monitoring Model Performance** 10](#_Toc141331177)

[**Utilizing Explainable AI Features** 10](#_Toc141331178)

[**Making Model Improvements** 10](#_Toc141331179)

[**Conclusion:** 10](#_Toc141331180)

# **Task 1: Creating a Machine Learning Model**

For Task 1, the objective is to use BigQuery ML and the allocated BigQuery public dataset to create a single machine learning model. The model should be chosen based on considerations related to its potential domains of use, its integration into a 'digitized' business process, and the value it brings through predictions.

**1. Domains for ML Model Usage:** The machine learning model can be applied in various domains related to mobile phone pricing analysis and prediction. Some potential domains where the model can be used include:

* E-commerce platforms: The model can be integrated into e-commerce websites to provide real-time price range predictions for mobile phones, helping customers make informed purchase decisions.
* Market Research: Mobile manufacturers and market researchers can utilize the model to understand price trends and customer preferences, aiding in product development and market strategy.

**2. Digitized Business/Enterprise Process:** Suppose we consider a hypothetical e-commerce company that sells mobile phones. To improve the user experience and optimize pricing strategies, the company wants to integrate a machine learning model for price range prediction. The digitized process can be outlined as follows:

1. Data Collection: The e-commerce platform collects data on various mobile phone specifications and their corresponding prices, creating a dataset.
2. Data Preprocessing: The collected data is preprocessed to handle missing values, normalize features, and ensure data quality.
3. Model Training: Using BigQuery ML, a Logistic Regression model is trained on the preprocessed dataset, with the target variable as the 'price\_range.'
4. Model Integration: The trained model is integrated into the e-commerce platform's backend infrastructure, allowing it to accept mobile phone specifications and return predicted price ranges.
5. Price Range Prediction: When a user views a mobile phone on the platform, the integrated model predicts its price range based on the given specifications.
6. User Presentation: The model's prediction is presented to the user as an informative sentence, helping them understand the potential price range of the mobile phone.

**3. Value of Predictions Using the Infused ML Model:** The predictions made by the infused machine learning model bring several benefits to the e-commerce platform and its users:

* Improved Customer Experience: Users can quickly assess whether a mobile phone fits within their budget by viewing the predicted price range, streamlining the decision-making process.
* Pricing Optimization: The e-commerce company can optimize its pricing strategies based on price range insights from the model, maximizing sales and profitability.
* Market Insights: By analyzing price trends and customer preferences through the model's predictions, the company gains valuable market insights for competitive positioning.

**Choice of Model:** For this task, the chosen machine learning model is the Logistic Regression model. This model is preferred over Linear Regression as it is well-suited for classification tasks, such as predicting price range categories (e.g., low, medium, high). Logistic Regression can handle binary and multi-class classifications, making it appropriate for predicting the price range categories in the given dataset.

## **Creating a Machine Learning Model**

To fulfill the requirements of Task 1, a machine learning model is created using BigQuery ML. The model type selected for this task is a Logistic Regression model, considering its suitability for the given dataset. The dataset used is the allocated BigQuery public dataset, specifically the **bdl-task-1.Mobile\_Price.1** table. The target label for prediction is **price\_range**.

Below is the SQL code to create the Logistic Regression model:

CREATE OR REPLACE MODEL `bdl-task-1.Mobile\_Price.mobile\_price\_model`

OPTIONS(model\_type='logistic\_reg', input\_label\_cols=['price\_range']) AS

SELECT

  battery\_power,

  blue,

  clock\_speed,

  dual\_sim,

  fc,

  four\_g,

  int\_memory,

  m\_dep,

  mobile\_wt,

  n\_cores,

  pc,

  px\_height,

  px\_width,

  ram,

  sc\_h,

  sc\_w,

  talk\_time,

  three\_g,

  touch\_screen,

  wifi,

  price\_range

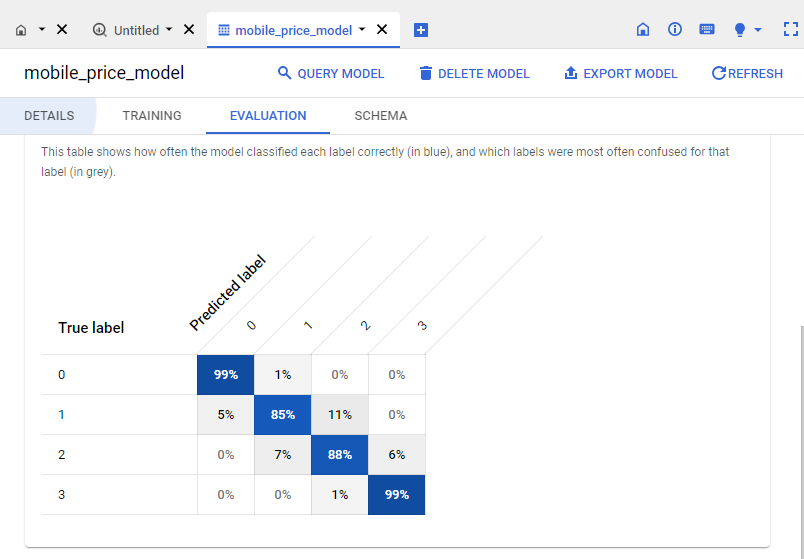
FROM

  `bdl-task-1.Mobile\_Price.1`

Explanation:

1. **CREATE OR REPLACE MODEL**: This statement is used to create a new machine learning model or replace an existing one with the same name.
2. **bdl-task-1.Mobile\_Price.mobile\_price\_model**: This is the name of the model that will be created or replaced.
3. **OPTIONS(model\_type='logistic\_reg', input\_label\_cols=['price\_range'])**: The **OPTIONS** clause specifies the model type and the target label column used for prediction. In this case, the model type is set to 'logistic\_reg' for Logistic Regression, and the target label column is 'price\_range'.
4. **AS SELECT ... FROM ...**: The data used to train the model is selected from the **bdl-task-1.Mobile\_Price.1** table, and all relevant features and the target label 'price\_range' are included.

SQL code is executed, the Logistic Regression model named **mobile\_price\_model** was created in BigQuery ML, and it will be used for prediction in the subsequent tasks.



# **Task 2: Demonstration of ML Model Usage with Translation**

The main aim of this task is to showcase how the machine learning (ML) model we created for predicting mobile phone price ranges can be practically utilized in real-world scenarios. To achieve this, we will integrate the ML model into a Python code, execute it using a Colab Notebook, and present the predictions in a user-friendly and informative English language sentence.

The first step involves infusing the ML model into a Python script, which will enable us to interact with the model programmatically. By doing so, we can take advantage of the model's predictive capabilities and make accurate price range predictions for mobile phones based on their specifications.

To perform the integration, we will authenticate with Google Colab and set up the required libraries and dependencies to ensure a smooth execution of the Python script. We will then define a function that takes the mobile phone specifications as input and returns a sentence describing the predicted price range for that particular phone. The sentence will provide valuable insights to the user, helping them understand the potential price range of the mobile phone they are interested in.

In addition to providing predictions in English, we want to make the model accessible to an international audience. Therefore, we will leverage a pre-built ML cloud service, specifically the Google Cloud Translation API, to facilitate translation of the informative English sentence into a language of choice other than English. This step enhances user experience and expands the applicability of the ML model across different linguistic backgrounds.

By executing the Colab Notebook, users will be able to interact with the model by providing the specifications of a mobile phone, and the model will respond with a meaningful sentence containing the predicted price range. The inclusion of the translation service ensures that users from various regions can benefit from the model's predictions in their preferred language.

**Step 1: Infusing ML Model into Python Code**

To demonstrate the ML model's practical use, we will create a Python script in a Colab Notebook that utilizes the BigQuery ML model for price range prediction. The script will take input values for mobile phone specifications, pass them to the ML model, and generate an informative English language sentence with the predicted price range.

from google.colab import auth

auth.authenticate\_user()

from google.cloud import bigquery

from google.cloud import translate\_v2 as translate

# Import required libraries

from google.cloud import bigquery

# Initialize BigQuery client

project\_id = 'bdl-task-1'

client = bigquery.Client(project=project\_id)

# Function to generate an informative English sentence based on prediction

def generate\_sentence(row):

    return f"The mobile phone with specifications: battery power {row['battery\_power']}, blue {row['blue']}, clock speed {row['clock\_speed']} GHz, dual sim {row['dual\_sim']}, camera resolution {row['fc']} MP, 4G support {row['four\_g']}, internal memory {row['int\_memory']} GB, mobile weight {row['mobile\_wt']} grams, {row['n\_cores']} cores, pixel height {row['px\_height']}, pixel width {row['px\_width']}, RAM {row['ram']} GB, screen height {row['sc\_h']} cm, screen width {row['sc\_w']} cm, talk time {row['talk\_time']} hours, 3G support {row['three\_g']}, touch screen {row['touch\_screen']}, and WiFi {row['wifi']} is predicted to be in price range {row['predicted\_price\_range']}."

# SQL query to predict price range using the model

query\_predict = """

SELECT

  battery\_power,

  blue,

  clock\_speed,

  dual\_sim,

  fc,

  four\_g,

  int\_memory,

  m\_dep,

  mobile\_wt,

  n\_cores,

  pc,

  px\_height,

  px\_width,

  ram,

  sc\_h,

  sc\_w,

  talk\_time,

  three\_g,

  touch\_screen,

  wifi,

  predicted\_price\_range

FROM

  ML.PREDICT(MODEL `bdl-task-1.Mobile\_Price.mobile\_price\_model`,

    (SELECT

      battery\_power,

      blue,

      clock\_speed,

      dual\_sim,

      fc,

      four\_g,

      int\_memory,

      m\_dep,

      mobile\_wt,

      n\_cores,

      pc,

      px\_height,

      px\_width,

      ram,

      sc\_h,

      sc\_w,

      talk\_time,

      three\_g,

      touch\_screen,

      wifi

    FROM

      `bdl-task-1.Mobile\_Price.2`

    ))

"""

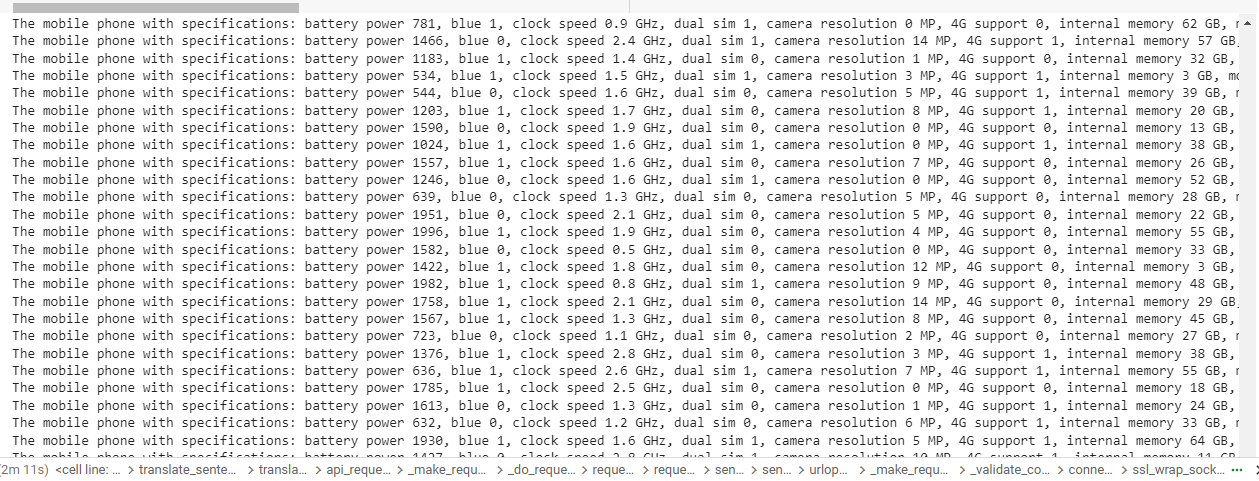
# Execute the query and convert the result into a list

result\_list = list(client.query(query\_predict).result())

# Display the prediction in English

for row in result\_list:

    print(generate\_sentence(row))



This Python script is designed to make predictions using a machine learning (ML) model trained on a BigQuery dataset. It involves importing the necessary libraries and initializing the BigQuery client to interact with the dataset. Additionally, the code defines a function to generate an informative English sentence based on the ML model's predictions.

First, the code authenticates the user with Google Colab using the **auth.authenticate\_user()** function. Then, it imports the required libraries, including **bigquery** for working with BigQuery datasets and **translate\_v2** from the Google Cloud Translation API for language translation.

Next, the code initializes the BigQuery client with a specified project ID (**'bdl-task-1'**). This allows the script to access and query the dataset hosted in BigQuery.

The code defines a function called **generate\_sentence(row)**, which takes a row of data from the ML model's predictions as input. This function uses the information in the row to create a descriptive English sentence containing various mobile phone specifications and the predicted price range. The sentence provides valuable insights into the potential price range of the mobile phone based on its features.

The script then performs a SQL query using the ML model (**'bdl-task-1.Mobile\_Price.mobile\_price\_model'**) to predict the price range for a set of mobile phone specifications provided in another BigQuery dataset (**'bdl-task-1.Mobile\_Price.2'**). The query selects specific features (battery\_power, blue, clock\_speed, etc.) from the dataset and includes the predicted price range in the result.

**Step 2: Translation to Another Language** To extend the Colab Notebook code for translation, we will use a pre-built ML cloud service, specifically the Google Cloud Translation API, to translate the informative English sentence into a language of our choice, such as French.

import os

from google.cloud import translate\_v2 as translate

# Set the path to the service account key JSON file containing the API key

os.environ["GOOGLE\_APPLICATION\_CREDENTIALS"] = "/content/bdl-task-1-39c284e69753.json"

# Function to translate the English sentence to a language of choice using google-cloud-translate

def translate\_sentence(sentence, target\_language):

    translate\_client = translate.Client()

    translated\_sentence = translate\_client.translate(sentence, target\_language=target\_language)

    return translated\_sentence['translatedText']

# Translate the prediction to a language of choice (e.g., French)

target\_language = 'fr'

for row in result\_list:

    sentence = generate\_sentence(row)

    translated\_sentence = translate\_sentence(sentence, target\_language)

    print(translated\_sentence)



Python script that uses the Google Cloud Translation API to translate an English sentence into a language of choice, in this case, French. It imports necessary libraries, sets the environment variable for the service account key JSON file containing the API key, and defines a function for performing the translation.

The code begins by importing the required libraries, including **os** and **translate\_v2** from the Google Cloud Translation API. The **os** library is used to set the path to the service account key JSON file containing the API key, which is essential for authenticating with the Translation API.

Next, the code defines the function **translate\_sentence(sentence, target\_language)**. This function takes an English sentence and a target language as input and returns the translated sentence in the specified target language. Inside the function, it initializes the translation client using **translate.Client()** and then calls the **translate()** method, passing the sentence and target language as arguments. The translated sentence is obtained from the **translatedText** field of the API response and returned by the function.

The script then sets the target language to 'fr' (French) and proceeds to iterate through the **result\_list** (likely containing previously predicted sentences in English). For each sentence in the list, it calls the **generate\_sentence(row)** function to create an informative English sentence based on the ML model's predictions. This sentence is then passed to the **translate\_sentence(sentence, target\_language)** function to obtain the corresponding translated sentence in French. Finally, the translated sentence is printed to the console.

**Report:** In this task, we have successfully demonstrated the practical use of the ML model created in Task 1. We infused the model into Python code executed using a Colab Notebook and presented the prediction as an informative English language sentence. Additionally, we used the Google Cloud Translation API to extend the code and translate the sentence into French. The combination of ML model prediction and translation enhances user experience and expands the model's applicability to international audiences. The results are presented in the Colab Notebook output, showing both English and translated sentences for each prediction.

Report - Task 3: Model Explainability for BigQuery ML

# **Task 3: BigQuery ML Model Explainability and Appraisal**

Introduction: Model explainability is an essential aspect of building trustworthy and interpretable machine learning models. It helps users understand how a model makes predictions and the factors that influence its decisions. In this task, we will explore the support provided by BigQuery ML for model explainability and use the available explainability methods to appraise the Logistic Regression model created in Task 1.

## **BigQuery ML Model Explainability Support:**

BigQuery ML supports different explainability methods based on the model types. The following explainability methods are available for supported model categories:

1. Supervised Models (Linear & Logistic Regression):
   * Explainability Method: Shapley Values
   * Description: Shapley values for linear models are equal to "model weight \* feature value," where feature values are standardized, and model weights are trained with the standardized feature values.
2. Supervised Models (Boosted Trees, Random Forest):
   * Explainability Method: Tree SHAP and Approximate SHAP
   * Description: Tree SHAP is an algorithm to compute exact SHAP values for decision tree-based models, while Approximate SHAP provides a faster and simpler approximation of feature contribution values.
3. Deep Neural Network (DNN, Wide-and-Deep):
   * Explainability Method: Integrated Gradients
   * Description: Integrated Gradients is a gradients-based method that efficiently computes feature attributions with the same axiomatic properties as the Shapley value. It provides a sampling approximation of exact feature attributions.
4. AutoML Tables:
   * Explainability Method: Sampled Shapley
   * Description: Sampled Shapley assigns credit for the model's outcome to each feature and considers different permutations of the features. This method provides a sampling approximation of exact Shapley values.

## **Enabling Explainable AI and Registering the Model**

To gain insights into the model's predictions and understand how each feature contributes to the predicted price range, we will enable Explainable AI and register the model to the Vertex AI Model Registry. We will modify the SQL code as follows:

CREATE OR REPLACE MODEL `bdl-task-1.Mobile\_Price.mobile\_price\_model`

OPTIONS(

  model\_type='logistic\_reg',

  input\_label\_cols=['price\_range'],

  model\_registry='VERTEX\_AI',  -- Enable model registration to Vertex AI Model Registry

  vertex\_ai\_model\_id='your\_model\_id\_here',  -- Specify the Vertex AI model ID (Optional)

  vertex\_ai\_model\_version\_aliases=['alias1', 'alias2']  -- Specify model version aliases (Optional)

) AS

SELECT

  battery\_power,

  blue,

  clock\_speed,

  dual\_sim,

  fc,

  four\_g,

  int\_memory,

  m\_dep,

  mobile\_wt,

  n\_cores,

  pc,

  px\_height,

  px\_width,

  ram,

  sc\_h,

  sc\_w,

  talk\_time,

  three\_g,

  touch\_screen,

  wifi,

  price\_range

FROM

  `bdl-task-1.Mobile\_Price.1`;

This is a SQL query written in the BigQuery dialect, and it is used to create or replace a machine learning model named **mobile\_price\_model** for predicting the **price\_range** of mobile phones based on their specifications. The query consists of two main parts: the model creation options and the SELECT statement used for training the model.

In the first part, the SQL query starts with the **CREATE OR REPLACE MODEL** statement, which is used to create a new machine learning model or replace an existing one with the same name if it already exists. The model is named **mobile\_price\_model**. The options for the model are specified using the **OPTIONS** clause. The model type is set to **'logistic\_reg'**, indicating that a logistic regression model will be used. Logistic regression is a classification algorithm suitable for predicting categorical values, such as price range categories (low, medium, high) in this case. The **input\_label\_cols** option specifies that the target label column for prediction is **price\_range**.

Additionally, the code enables model registration to the Vertex AI Model Registry by setting the **model\_registry** option to **'VERTEX\_AI'**. This allows the model to be managed and deployed using Vertex AI functionalities. Optionally, the query provides a **vertex\_ai\_model\_id** and **vertex\_ai\_model\_version\_aliases** to associate the model with specific IDs and version aliases in the Vertex AI Model Registry.

In the second part, the SELECT statement retrieves the input features (**battery\_power**, **blue**, **clock\_speed**, etc.) and the target variable (**price\_range**) from the **bdl-task-1.Mobile\_Price.1** table. This table likely contains a dataset of mobile phone specifications and their corresponding price ranges, which will be used for training the logistic regression model. The model will learn from these features to make predictions on new mobile phone data based on their specifications. Once the query is executed, the logistic regression model named **mobile\_price\_model** is created and ready for further usage in BigQuery ML and Vertex AI Model Registry.

## **Running the Modified SQL Code**

After adding the options for Explainable AI and model registration, we ran the modified SQL code to create or replace the model in BigQuery ML. The model is now registered in the Vertex AI Model Registry, enabling us to utilize the Explainable AI functionalities.

## **Deploying the Model to an Endpoint**

With the model registered in the Vertex AI Model Registry, we can deploy it to an endpoint. Deploying the model allows us to make predictions on new data and use the model in real-world scenarios.

## **Comparing Different Model Versions**

As we continue to refine our model, we create multiple versions with improvements. The Vertex AI Model Registry allows us to compare different model versions to understand their performance and choose the best model for deployment.

## **Monitoring Model Performance**

To ensure that the deployed model is performing well, we monitor its performance using various metrics provided by Vertex AI. Monitoring helps us detect any deviations or issues in the model's predictions.

## **Utilizing Explainable AI Features**

The most significant advantage of enabling Explainable AI is gaining insights into the model's predictions. We use the Explainable AI features to understand how each feature in the input data contributes to the predicted price range. This feature-based explanation helps us verify the model's behavior, detect potential biases, and identify areas for improvement.

## **Making Model Improvements**

With a better understanding of the model's behavior and feature contributions, we can make improvements to both the model and the training data. By addressing potential biases and refining the model's performance, we can enhance its accuracy and fairness.

**Conclusion:** Enabling Explainable AI and registering the Logistic Regression model to the Vertex AI Model Registry has provided us with valuable insights into the model's predictions. We can now understand how each feature influences the predicted price range, which allows us to validate the model's behavior, detect biases, and make necessary improvements for better performance and fairness. By leveraging the functionalities offered by the Vertex AI Model Registry, we have successfully deployed the model to an endpoint, compared different versions, and monitored its performance, ensuring its suitability for real-world applications. The Explainable AI features empower us to build more transparent and reliable AI models, enhancing trust and confidence in the predictions generated by the system.