**Hotel Booking Analysis and Prediction**

**Pipeline**

*A Project Report Submitted By*

Jeyadev L

Gayathri R

Devasree R

Bratati R

Sharone V

*In partial fulfilment of the requirements for the award of the degree of*

**PGD – DE**

A logo with a circular design

Description automatically generated with medium confidence

# Abstract

The hospitality industry relies on accurate forecasting of customer behaviour and revenue metrics to optimize operations and enhance profitability. This project addresses the challenges of managing hotel bookings by implementing a robust, scalable data pipeline that integrates data engineering, machine learning, and visualization. Using historical hotel booking data, we developed a pipeline capable of predicting booking cancellations and forecasting the Average Daily Rate (ADR), a critical revenue metric. The system is deployed on Google Cloud Platform (GCP), leveraging its storage, computation, and database capabilities to ensure scalability and efficiency.

The pipeline begins with ingesting and cleaning raw booking data stored in Google Cloud Storage (GCS), transforming it into a structured format suitable for analysis and machine learning. Two machine learning models are employed: a Gradient Boosting Classifier for cancellation prediction and a Gradient Boosting Regressor for ADR prediction. Both models are optimized using hyperparameter tuning via Optuna to ensure high accuracy and performance. Predictions and processed data are stored in Big Query, enabling downstream analytics and visualization.

An interactive Power BI dashboard was created to visualize trends and insights, such as monthly cancellation patterns, revenue breakdowns by market segment, and lead time influences on booking behaviour. The Flask API, hosted on GCP Cloud Run, provides endpoints for triggering data ingestion and prediction pipelines, ensuring ease of use and integration with external systems.

This project highlights the importance of integrating advanced data processing techniques with machine learning to address real-world business challenges in the hospitality sector. By enabling data-driven decision-making, the system provides actionable insights for revenue optimization and operational efficiency, making it a valuable tool for hotel management teams. Future work aims to incorporate real-time data processing and automated model retraining to adapt to changing booking behaviours.

# 1. Introduction

## *1.1 Background*

The hospitality industry operates in a highly dynamic environment, where customer behaviours, market trends, and external factors like seasonality, economic changes, and global crises significantly influence operations. A key challenge for hotel management is the ability to forecast booking behaviours accurately and optimize pricing strategies. With fluctuating demand, high cancellation rates, and intense competition among providers, predictive analytics and data-driven decision-making are becoming increasingly essential.

Cancellations often result in lost revenue and operational inefficiencies, as unsold rooms may not be filled in time. Similarly, the Average Daily Rate (ADR), a critical measure of revenue generation, requires precise forecasting to ensure competitive pricing while maximizing profitability. Traditional methods, such as static pricing models or manually curated predictions, fail to adapt to these complexities in real-time, highlighting the need for more advanced, automated solutions.

This project focuses on developing a comprehensive, cloud-based pipeline for analysing hotel bookings and leveraging machine learning to predict cancellations and forecast ADR. By combining robust data engineering practices, predictive analytics, and interactive visualization tools, the system provides actionable insights to assist hotel management in improving operational efficiency and revenue optimization.

## *1.2 Problem Statement*

Key challenges faced by hotel management include:

1. **Cancellations**: High cancellation rates result in revenue loss and disrupt operations. Identifying the likelihood of cancellations based on booking attributes can enable better resource allocation and rebooking strategies.
2. **Revenue Forecasting**: The ADR depends on multiple factors, including market segment, lead time, and seasonality. Accurate ADR predictions are essential for competitive pricing and profitability.
3. **Scalability**: With growing data volumes and the need for real-time insights, traditional on-premises solutions are inadequate.
4. **Data Visualization**: Stakeholders require easily interpretable dashboards to understand trends, analyse model predictions, and support decision-making.

## *1.3 Objectives*

This project aims to address the above challenges through the following objectives:

1. **Data Engineering**:
   * Clean and preprocess raw booking data for analysis and machine learning.
   * Generate synthetic data to augment datasets for testing and modelling.
2. **Machine Learning**:
   * Develop predictive models for:
     + **Cancellation Prediction**: Identify bookings likely to be cancelled.
     + **ADR Forecasting**: Predict the revenue potential of each booking.
   * Optimize models using hyperparameter tuning techniques (Optuna) for high performance.
3. **Data Storage and Accessibility**:
   * Store processed data and predictions in Big Query for ease of integration and analysis.
4. **Visualization**:
   * Create interactive dashboards in Power BI to provide stakeholders with real-time insights.
5. **Cloud Deployment**:
   * Deploy the system on GCP using a Flask-based API hosted on Cloud Run for scalability and accessibility.

## *1.4 Scope*

The scope of this project includes the following:

* Processing raw data provided by hotels, including historical booking information.
* Training machine learning models on key features such as lead time, hotel type, market segment, and special requests.
* Deploying the solution to enable real-time prediction and ingestion workflows via REST APIs.
* Visualizing results and trends in Power BI, enabling decision-makers to derive actionable insights.

The pipeline is designed to handle both batch and real-time processing, making it extensible for future enhancements, such as integrating external data sources (e.g., weather, events) or real-time streaming platforms like Apache Kafka.

## *1.6 Paper Organization*

The remainder of this paper is organized as follows:

* **Section 2** discusses the system architecture, including data pipeline design and deployment on GCP.
* **Section 3** details the data engineering techniques used for cleaning and transforming raw booking data.
* **Section 4** describes the machine learning models developed for cancellation prediction and ADR forecasting, including their performance metrics.
* **Section 5** outlines the implementation details of the Flask API and Power BI dashboard.
* **Section 6** highlights key results and insights derived from the project.
* **Section 7** covers deployment strategies and challenges encountered during implementation.
* **Section 8** discusses potential improvements and future directions for the pipeline.
* **Section 9** concludes with a summary of the project’s contributions.

# 2. System Architecture

The **Hotel Booking Analysis and Prediction Pipeline** is designed as a modular and scalable solution leveraging Google Cloud Platform (GCP) for data ingestion, processing, prediction, and visualization. The architecture integrates robust data engineering workflows with machine learning capabilities, ensuring seamless end-to-end automation. This section discusses the architecture in detail, focusing on the **Data Pipeline Workflow** and **Deployment Architecture**.

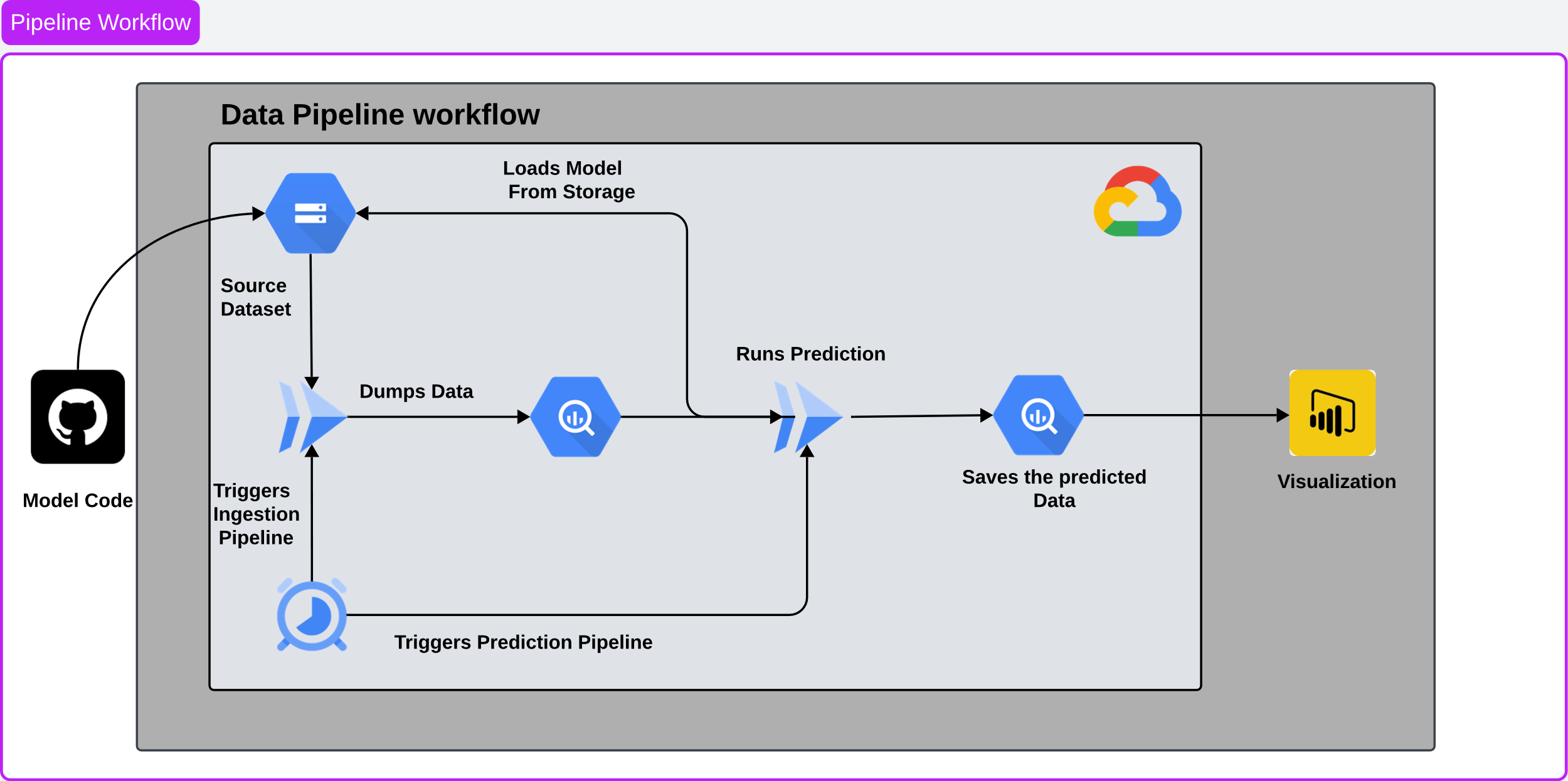
[](#_2._System_Architecture)

Figure 1

2.1 Data Pipeline Workflow

The data pipeline automates the movement and transformation of data from raw sources to actionable insights. Figure 1 illustrates the end-to-end pipeline.

### *2.1.1 Data Sources*

* **Raw Data**:
  + Provided as historical hotel booking datasets in .csv format.
  + Stored in **Google Cloud Storage (GCS)** in the input\_data bucket.
* **Pre-trained ML Models**:
  + Stored in GCS under the Models directory.
  + These models are used for predictions of booking cancellations and Average Daily Rate (ADR).

### *2.1.2 Data Ingestion and Cleaning*

The pipeline starts by loading raw booking data from GCS into a panda DataFrame. Cleaning and preprocessing steps include:

* Handling missing values:
  + For numerical fields like children: median imputation.
  + For categorical fields like country: mode imputation.
* Encoding categorical features:
  + Label encoding is applied to features such as hotel, market\_segment, and arrival\_date\_month.
* Storing cleaned data in **Big Query** under the HOTEL\_BOOKING table for persistent storage.

### *2.1.3 Machine Learning Inference*

Two machine learning models are used:

1. **Cancellation Prediction**:
   * Features: lead\_time, market\_segment, previous\_cancellations, etc.
   * Output: Binary prediction (1 if the booking is likely to be cancelled, 0 otherwise).
2. **ADR Prediction**:
   * Features: hotel, lead\_time, market\_segment, total\_of\_special\_requests, etc.
   * Output: Predicted ADR value (float).

Both models are loaded dynamically from GCS using the joblib library, ensuring efficient memory utilization through model caching.

2.1.4 Synthetic Data Generation

A synthetic dataset is created for testing and analysis. This involves:

* Random sampling of cleaned data to generate between 0 and 100 new records.
* Adding metadata, such as a globally unique identifier (guid) and a timestamp.

2.1.5 Storage of Predictions

Predicted results, along with metadata, are appended to **BigQuery** in a separate table (PREDICTION\_RESULTS). The schema includes fields for:

* **Raw Inputs**: Features such as lead\_time, hotel, and market\_segment.
* **Predictions**: predicted\_is\_canceled and predicted\_adr.
* **Metadata**: guid and insert\_timestamp.

### *2.1.6 Visualization*

Data from BigQuery is visualized in **Power BI**, allowing stakeholders to:

* Monitor monthly trends in cancellations and revenue.
* Evaluate model performance by comparing predictions with actual values.
* Analyse revenue contributions by market segments.

2.2 Deployment Architecture

The system leverages a cloud-native architecture to ensure scalability, maintainability, and high availability. Figure 2 illustrates the deployment components and their interactions.

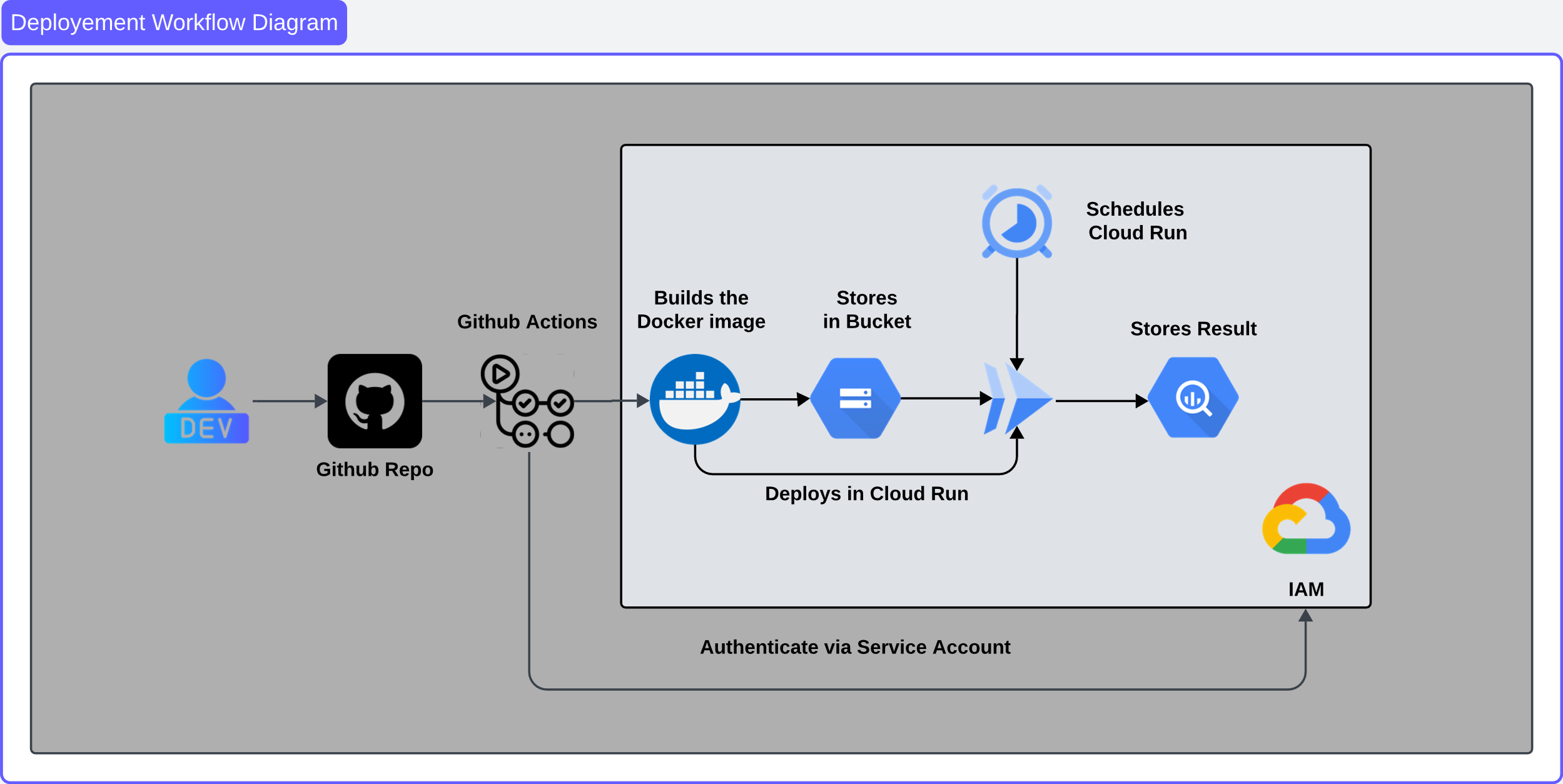


Figure 2

2.2.1 Core Components

1. **Flask API**:

Serves as the backbone of the system.

Provides RESTful endpoints (/ingest and /predict) for triggering data ingestion and prediction pipelines.

Handles data transformation, model inference, and BigQuery operations.

1. **Docker**:

Encapsulates the application, including all dependencies, ensuring consistency across environments.

The application is exposed on port 8080 and uses Gunicorn for production-grade performance.

1. **Google Cloud Run**:

* Hosts the containerized Flask API.
* Ensures auto-scaling based on request load.
* Offers serverless deployment, reducing operational overhead.

2.2.2 Cloud Services

1. **Google Cloud Storage (GCS)**:

Stores input datasets, cleaned datasets, and pre-trained machine learning models.

**Buckets used:**

* input\_data\_capstone for raw and cleaned data.
* capstone-model-group-18 for model storage.

1. **Google BigQuery**:

* Acts as the primary database for storing ingested data and predictions.
* Tables:
* HOTEL\_BOOKING for raw and cleaned booking data.
* PREDICTION\_RESULTS for storing model predictions.

1. **Google Artifact Registry**:
   * Hosts container images for the Flask API.

2.2.3 Deployment Steps

1. **Containerization**:
   * The application is packaged using the Dockerfile, which installs all dependencies and configures the Flask API.
2. **Continuous Integration and Deployment (CI/CD)**:
   * A YAML-based deployment script automates the build and deployment of the container to GCP.
3. **Service Deployment**:
   * The container is deployed on Cloud Run with the following configurations:
     + Platform: Fully managed.
     + Region: Selected for proximity to data sources and users.
     + Authentication: Public access with IAM-controlled permissions.

2.2.4 Logging and Monitoring

* **Logging**:
  + Application logs are written to both a local file (prediction\_log.log) and the console.
  + Key events include data ingestion, model loading, and prediction outcomes.
* **Monitoring**:
  + GCP’s built-in monitoring tools track API request latency, error rates, and resource utilization.

2.3 Scalability and Extensibility

The architecture is designed for future enhancements:

* **Scalability**:
  + Cloud Run auto-scales the API based on incoming traffic, ensuring low latency during peak usage.
* **Extensibility**:
  + Additional ML models can be added to predict other booking attributes, such as customer churn or booking duration.
  + Integration with real-time data streaming tools (e.g., Apache Kafka) can enable continuous updates to predictions.

2.4 Security

1. **Data Access**:
   * GCP IAM roles are configured to restrict access to buckets and BigQuery tables.
2. **API Authentication**:
   * Endpoints are exposed publicly but can be restricted using GCP API Gateway or service tokens.
3. **Encryption**:
   * Data in transit and at rest is encrypted using GCP's default encryption mechanisms.

3. Data Engineering

The data engineering component of this project focuses on ingesting, transforming, and preparing raw hotel booking data for analysis and machine learning. This phase ensures that the data is clean, structured, and readily accessible for both predictive modelling and visualization.

3.1 Data Sources

The project utilizes a raw dataset provided in .csv format, containing historical hotel booking information. Key fields in the dataset include:

* **Categorical Attributes**: hotel, market\_segment, arrival\_date\_month, etc.
* **Numerical Attributes**: lead\_time, adr, total\_of\_special\_requests, etc.
* **Target Variables**:
  + is\_canceled: Indicates whether a booking was cancelled.
  + adr: Average Daily Rate, representing revenue generated per booking.

The dataset is uploaded to **Google Cloud Storage (GCS)** under the input\_data bucket for storage and processing.

3.2 Data Cleaning

The raw dataset contains inconsistencies and missing values that must be addressed to ensure data quality. Key data cleaning steps include:

3.2.1 Handling Missing Values

* **Numerical Fields**:
  + For fields like children, the **median** value is used to fill missing entries. This approach minimizes the effect of outliers.
* **Categorical Fields**:
  + For fields like country, the **mode** is used to replace missing values, ensuring consistency in categorical distributions.
* **Agent and Company Columns**:
  + Null values are explicitly replaced with Unknown, as they represent missing or unused agent/company identifiers.

3.2.2 Encoding Categorical Variables

Machine learning models require numerical input, so categorical variables are encoded using the **Label Encoding** technique:

* Columns such as hotel, market\_segment, and arrival\_date\_month are transformed into integer representations using LabelEncoder from the scikit-learn library.

3.2.3 Outlier Detection and Removal

* Outliers in numerical fields, such as adr and lead\_time, are detected using interquartile range (IQR) techniques. Extreme outliers are either removed or capped at a threshold value.

3.2.4 Consistency Checks

* The dataset is validated to ensure that fields like adults, children, and babies do not have zero values simultaneously (i.e., bookings with no guests are removed).

3.3 Synthetic Data Generation

To augment the dataset for testing and analysis, synthetic data is generated based on the cleaned dataset. This step simulates additional records, ensuring robust testing of the ingestion and prediction pipelines.

3.3.1 Random Sampling

* A random number of rows (between 0 and 100) is sampled from the cleaned dataset.
* Columns are populated using random sampling with replacement to simulate realistic variations in the data.

3.3.2 Metadata Augmentation

* Each synthetic record is assigned:
  + A **Globally Unique Identifier (GUID)** for tracking and identification.
  + An **Insert Timestamp**, representing the time of data generation.

3.3.3 Validation

* Synthetic records are validated to ensure they adhere to the schema and do not introduce anomalies into the pipeline.

3.4 BigQuery Integration

The processed data is stored in **Google BigQuery**, a scalable, serverless database service, ensuring efficient querying and integration with downstream components like machine learning models and Power BI dashboards.

3.4.1 Schema Definition

Two BigQuery tables are defined:

1. **HOTEL\_BOOKING**:

  `guid` STRING,

  `hotel` STRING,

  `is\_canceled` INTEGER,

  `lead\_time` INTEGER,

  `arrival\_date\_year` INTEGER,

  `arrival\_date\_month` STRING,

  `arrival\_date\_week\_number` INTEGER,

  `arrival\_date\_day\_of\_month` INTEGER,

  `stays\_in\_weekend\_nights` INTEGER,

  `stays\_in\_week\_nights` INTEGER,

  `adults` INTEGER,

  `children` FLOAT64,

  `babies` INTEGER,

  `meal` STRING,

  `country` STRING,

  `market\_segment` STRING,

  `distribution\_channel` STRING,

  `is\_repeated\_guest` INTEGER,

  `previous\_cancellations` INTEGER,

  `previous\_bookings\_not\_canceled` INTEGER,

  `reserved\_room\_type` STRING,

  `assigned\_room\_type` STRING,

  `booking\_changes` INTEGER,

  `deposit\_type` STRING,

  `agent` STRING,

  `company` STRING,

  `days\_in\_waiting\_list` INTEGER,

  `customer\_type` STRING,

  `adr` FLOAT64,

  `required\_car\_parking\_spaces` INTEGER,

  `total\_of\_special\_requests` INTEGER,

  `reservation\_status` STRING,

  `reservation\_status\_date` STRING,

  `insert\_timestamp` TIMESTAMP

1. **PREDICTION\_RESULTS**:

    guid STRING,

    lead\_time INTEGER,

    hotel STRING,

    market\_segment STRING,

    previous\_cancellations INTEGER,

    booking\_changes INTEGER,

    total\_of\_special\_requests INTEGER,

    arrival\_date\_month STRING,

    is\_repeated\_guest INTEGER,

    is\_canceled boolean,

    predicted\_is\_canceled BOOLEAN,

    adr float64,

    predicted\_adr FLOAT64,

    insert\_timestamp TIMESTAMP

3.4.2 Data Loading

* **Ingestion**:
  + Cleaned data is loaded into HOTEL\_BOOKING using the **BigQuery Python API**.
* **Predictions**:
  + The **/predict** endpoint appends prediction results to the PREDICTION\_RESULTS table.

3.4.3 Query Optimization

* Partitioning and clustering are implemented for the HOTEL\_BOOKING table to optimize query performance. For instance:
  + **Partitioning**: By arrival\_date\_month to support time-based queries.
  + **Clustering**: On fields like market\_segment and hotel for faster lookups.

3.5 Data Validation and Quality Assurance

3.5.1 Validation Rules

* Ensures that critical fields like adr and lead\_time are within expected ranges.
* Validates that encoded categorical fields map correctly to their original categories.

3.5.2 Logging

* All data processing steps, including cleaning, synthetic generation, and BigQuery operations, are logged to a central log file (prediction\_log.log) and the console for debugging.

3.5.3 Error Handling

* Missing or malformed rows are flagged and excluded from processing to avoid pipeline failures.
* Errors encountered during BigQuery writes (e.g., schema mismatches) are logged, and the system retries after automatic corrections.

3.6 Scalability and Extensibility

The data engineering pipeline is designed to handle large volumes of data and is easily extensible:

Scalability:

* + By leveraging BigQuery, the system can process millions of rows without performance degradation.

Extensibility:

* + Additional preprocessing steps (e.g., feature engineering) can be integrated without disrupting the existing pipeline.
  + New data sources, such as external APIs for weather or event data, can be incorporated to enhance prediction accuracy.

3.7 Challenges and Mitigations

Challenges

1. **Handling Missing Data**:
   * Missing values in critical fields like children and country posed challenges for analysis and modelling.
2. **Outlier Detection**:
   * Outliers in adr and lead\_time affected prediction accuracy and required careful handling.
3. **Integration with BigQuery**:
   * Ensuring schema consistency between pandas DataFrames and BigQuery tables was complex.

Mitigations

1. Implemented imputation techniques (median, mode) to address missing values systematically.
2. Applied statistical thresholds to cap outliers without discarding important data.
3. Automated schema validation to ensure seamless BigQuery integration.

4. Machine Learning

Machine learning is at the core of this project, enabling predictions for hotel booking cancellations and revenue forecasting through Average Daily Rate (ADR). This section elaborates on the models developed, the reasons for choosing specific approaches, and the methodologies employed to optimize and deploy these models.

4.1 Problem Definition

4.1.1 Cancellation Prediction

* + **Objective**: Predict whether a hotel booking will be cancelled.
  + **Why?** High cancellation rates disrupt hotel operations and lead to revenue loss. Accurate predictions allow hotels to proactively manage overbooking and improve resource allocation.
  + **Type of Problem**: Binary classification problem, where the target variable is\_canceled is either 0 (not cancelled) or 1 (cancelled).

4.1.2 ADR Prediction

* + **Objective**: Forecast the ADR (Average Daily Rate) for each booking, representing the revenue generated per room per night.
  + **Why?** ADR predictions help hotels optimize pricing strategies, monitor revenue trends, and plan for peak seasons.
  + **Type of Problem**: Regression problem, where the target variable adr is a continuous numeric value.

4.2 Dataset Preparation

The dataset for both models was derived from the cleaned and pre-processed data stored in Google Big Query. Key steps in dataset preparation include:

**Feature Selection:**

1. For Cancellation Prediction:
   * Features such *as lead\_time, hotel, market\_segment, previous\_cancellations, booking\_changes, and total\_of\_special\_requests* were selected based on domain knowledge.
2. For ADR Prediction:
   * Features such as *lead\_time, hotel, market\_segment, arrival\_date\_month, is\_repeated\_guest, and previous\_cancellations* were used.
3. Categorical Encoding:
   * Categorical variables like *hotel* and *market*\_*segment* were label-encoded using scikit-learn’s LabelEncoder.
4. Train-Test Split:
   * Both datasets were split into 70% training data and 30% testing data to evaluate model performance.

4.3 Model Development

4.3.1 Cancellation Prediction

Several machine learning algorithms were tested for their performance:

1. **Random Forest Classifier:**
   * A popular ensemble method that combines multiple decision trees for robust predictions.
   * **Why?** Random Forest is resilient to overfitting and performs well on structured data with categorical and numerical features.
2. **Gradient Boosting Classifier:**
   * An ensemble method that builds decision trees sequentially, optimizing for errors in previous iterations.
   * **Why?** Gradient Boosting often outperforms Random Forest for binary classification problems, especially when the dataset is imbalanced.
3. **Logistic Regression:**
   * A linear model for binary classification.
   * **Why?** Logistic Regression provides interpretability, making it easier to understand feature importance.
4. **Decision Tree Classifier:**

* A simple tree-based model that splits data based on feature thresholds.
* **Why?** Decision Trees are computationally inexpensive and serve as a baseline model.

**Optimization:**

* Hyperparameter tuning was performed using **Optuna**, an open source hyperparameter optimization framework.
* Parameters such as n\_estimators, max\_depth, learning\_rate, and min\_samples\_split were optimized for Gradient Boosting and Random Forest models.

**Best Model:**

* The **Gradient Boosting Classifier** emerged as the best model with:
  + **Accuracy**: 89.5%.
  + **Precision, Recall, and F1-Score**: Evaluated to balance between false positives and false negatives.

4.3.2 ADR Prediction

Several regression models were evaluated for ADR prediction:

**Linear Regression:**

* A simple linear model that assumes a linear relationship between features and the target variable.
* **Why?** Linear Regression serves as a baseline for comparison with more complex models.

**Decision Tree Regressor:**

* A tree-based model that splits data based on feature thresholds.
* **Why?** Decision Trees capture non-linear relationships but can overfit on training data.

**Random Forest Regressor**:

* An ensemble method combining multiple decision trees to improve generalization.
* **Why?** Random Forest is robust to overfitting and captures non-linear interactions well.

**Gradient Boosting Regressor**:

* Sequentially builds decision trees, optimizing for prediction errors.
* **Why?** Gradient Boosting often provides higher accuracy for continuous target variables compared to Random Forest.

**Optimization**:

Hyperparameter tuning was conducted using manual grid search, focusing on parameters like n\_estimators, max\_depth, and learning\_rate.

**Best Model:**

The **Gradient Boosting Regressor** achieved the best results with:

* R² **(Coefficient of Determination**): 92%.
* Root Mean Squared Error (**RMSE**): 10.5.
* Mean Absolute Error (**MAE**): 8.2.

4.4 Model Evaluation

1. Cancellation Prediction:
   * **Metrics Used**:
     + **Accuracy**:Measures overall correctness of predictions.
     + **Precision**: Evaluates the percentage of correctly predicted cancellations.
     + **Recall**: Measures how many actual cancellations were correctly predicted.
     + **F1-Score**: A harmonic mean of precision and recall handling class imbalances.
2. Performance:
   * **Gradient Boosting Classifier** achieved the highest accuracy **(89.5%).**
3. ADR Prediction:
   * Metrics Used:
     + **R²**: Measures the proportion of variance explained by the model.
     + **RMSE**: Quantifies the average error magnitude in predictions.
     + **MAE**: Measures the average absolute error in predictions.
   * Performance:
     + Gradient Boosting Regressor achieved an R² score of 92%, indicating strong predictive power.

4.5 Model Deployment

4.5.1 Model Saving

The best-performing models were saved as .pkl files using the joblib library.

* + gradient\_boosting\_cancellation\_model.pkl (Cancellation Prediction).
  + gradient\_boosting\_regressor\_adr\_model.pkl (ADR Prediction).

4.5.2 Cloud Integration

* + **Storage**: Models were uploaded to Google Cloud Storage under the Models directory.
  + **Dynamic Loading**: Models are loaded dynamically by the Flask API at runtime, ensuring efficient memory utilization through caching.

4.6 Why These Approaches Were Chosen

1. Gradient Boosting:
   * Outperformed other algorithms in both classification and regression tasks.
   * Offers flexibility and handles feature interactions better than linear models.
2. Hyperparameter Tuning:
   * Optuna and grid search helped improve model performance by finding the optimal parameter combinations.
3. Feature Selection:
   * Features were selected based on domain knowledge and correlation analysis to improve interpretability and reduce computational complexity.
4. Dynamic Model Loading:
   * Ensures efficient use of computational resources and allows for easy updates to models without redeploying the application.

4.7 Challenges and Solutions

Imbalanced Data:

* + Cancellation data exhibited class imbalance, with more bookings labelled as "*not cancelled*."
  + **Solution**: The Gradient Boosting model handled class imbalance well, and evaluation metrics like F1-Score were used to ensure balanced performance.

Feature Engineering:

* + Categorical features required encoding, which introduced complexity.
  + **Solution**: Label encoding was used, and the pipeline ensured consistent mapping across training and inference.

**Prediction Latency**:

* + Dynamic model loading introduced slight delays during the first request.
  + **Solution**: Cached models in memory after the initial load for faster subsequent predictions.

5. Implementation

5.1 Flask API

The Flask API provides the following endpoints:

* + / (Health Check): Confirms the service is running.
  + /ingest: Ingests cleaned data from GCS, generates synthetic data, and writes it to the BigQuery *HOTEL\_BOOKING* table.
  + /predict: Fetches data from BigQuery, generates predictions for cancellations and ADR, and writes results to the *PREDICTION*\_*RESULTS* table in BigQuery.

5.2 Power BI Dashboard

The Power BI dashboard provides key visualizations:

* + Cancellation Trends: Trends by hotel type and month.
  + ADR Trends: Actual vs. predicted revenue trends over time.
  + Revenue by Market Segment: Breakdown of revenue contributions by market segment.

6. Results

6.1 Model Performance

* Cancellation Prediction:
  + Model: Gradient Boosting Classifier.
  + Accuracy: 89.5%.
* ADR Prediction:
  + Model: Gradient Boosting Regressor.
  + R²: 0.92.

6.2 Insights

Revenue Analysis by Hotel Type

* + Resort Hotels tend to have a higher average daily rate (ADR) compared to City Hotels, indicating that guests are willing to pay more for resort stays, due to better amenities, location, and vacation-related services.
  + City Hotels cater more to business travellers and short stays, leading to lower ADRs.

Revenue Differences by Market Segment

* + Market segments such as Corporate and Direct bookings show higher ADRs, indicating that these segments are associated with more premium-priced bookings.
  + Online Travel Agencies (OTAs), while providing a significant volume of bookings, tend to have lower ADRs, due to the competitive nature of the OTA market and price comparisons available to customers.
  + Groups bookings have lower ADRs, likely because they often receive discounts for bulk bookings.

Impact of Lead Time on Revenue

* + There is a noticeable relationship between lead time (the number of days between the booking date and the arrival date) and ADR.
  + Bookings made far in advance (longer lead time) tend to have higher ADRs, reflecting advance pricing strategies where customers secure higher-priced bookings early. However, the ADR fluctuates depending on the seasonality and demand.
  + Last-minute bookings (shorter lead time) may lead to slightly lower ADRs, potentially reflecting discounts or promotions offered to fill up remaining room capacity.

Seasonality Impact on Revenue

* + Seasonality plays a crucial role in revenue generation. As observed:
  + Summer tends to generate the highest revenue, which is expected as summer is typically a peak season for vacations.
  + Winter also shows elevated ADRs, due to the holiday season and increased demand for both resort and city hotels during this time.
  + Fall and Spring show slightly lower ADRs, indicating these are off-peak seasons where demand softens, and hotels may offer competitive pricing to attract guests.

Cancellations by Market Segment

* + The online travel agencies (OTA) segment experiences the highest cancellation rates, likely because OTA customers may have more flexible cancellation policies.
  + Corporate and Direct bookings show much lower cancellation rates, which is consistent with more secure and planned travel (especially business related).
  + Group bookings also show moderately higher cancellation rates, possibly because group organizers may change plans based on group size fluctuations.

Cancellation Trends by Lead Time

* + Bookings with longer lead times are more likely to be cancelled compared to those made closer to the arrival date. This suggests that guests may cancel early bookings as their travel plans evolve, especially for vacations.
  + Shorter lead time bookings (last minute bookings) tend to have fewer cancellations, likely because plans are more confirmed and imminent.

Cancellation Trends by Season

* + July and August show elevated cancellation rates, reflecting vacation plans being adjusted or changed during peak summer months.
  + December also shows significant cancellations, which may be related to holiday travel plans that are more flexible or subject to change.

8. Challenges and Improvements

8.1 Challenges

1. Managing Missing Values: Addressing gaps in critical columns like children and country while ensuring data consistency for modelling.
2. Handling High-Dimensional Categorical Data: Encoding multiple categorical variables efficiently without overcomplicating the pipeline.
3. Optimizing Model Inference Speed: Reducing latency during real-time predictions when models are dynamically loaded.

8.2 Future Improvements

1. Real-Time Streaming: Integrate tools like Apache Kafka to enable real-time ingestion and prediction pipelines.
2. Advanced Modelling: Incorporate deep learning approaches or ensemble techniques to improve predictive accuracy.
3. Automated Retraining: Build pipelines for continuous monitoring and retraining of models to adapt to changing booking behaviours and market trends.

9. Conclusion

This project demonstrates a comprehensive approach to solving critical challenges in the hospitality industry by leveraging advanced data engineering, machine learning, and visualization techniques. By addressing issues such as high cancellation rates and fluctuating revenue, the pipeline enables hotel management to make data-driven decisions, optimize operations, and enhance profitability.

The integration of machine learning models for cancellation prediction and ADR forecasting allows the system to provide actionable insights. The Gradient Boosting Classifier predicts cancellations with high accuracy, helping hotels proactively manage overbooking and reduce revenue loss. Similarly, the Gradient Boosting Regressor for ADR forecasting enables better pricing strategies and resource allocation by accurately predicting revenue trends.

The deployment on Google Cloud Platform (GCP) ensures scalability, efficiency, and accessibility. Components like Cloud Storage, BigQuery, and Cloud Run are seamlessly integrated to handle large volumes of data, perform real-time predictions, and store results for analysis. The use of Power BI dashboards further enhances the system by providing stakeholders with visual insights into trends, market performance, and model outputs.

This project also highlights the importance of a modular and extensible architecture. The system is designed to adapt to future enhancements, such as integrating real-time streaming with tools like Apache Kafka, incorporating advanced machine learning models, or automating model retraining pipelines. These improvements can help the pipeline evolve to meet changing business requirements and market conditions.

In conclusion, this project not only addresses current challenges in hotel revenue management but also provides a scalable, future-ready solution that combines state-of-the-art technology with practical insights. It serves as a valuable tool for optimizing revenue, managing cancellations, and driving operational efficiency in the hospitality sector.