AUTOMATED BI-LINGUAL COMPLAINT SYSTEM

PROJECT SCOPING REPORT

of

IE 7273: MACHINE LEARNING OPERATIONS

BY

GROUP NUMBER 6:

ARCHIT SINGH (002813253)

ANIRUDH HEGDE (002813268)

RAHUL NENAVATH (002866967)

RAJKUMAR RAJANIKANT PANDEY (002898966)

SAYAM KHATRI (002877613)

VENKATA MANI SIVASAI SHANMUKHA RAJENDRA GOPARAJU (002299776)

DEPARTMENT OF COLLEGE OF ENGINEERING

NORTHEASTERN UNIVERSITY

BOSTON, MASSACHUSETTS – 022115

OCTOBER 2024

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

1.0 INTRODUCTION

This project aims to build an automated system for handling customer complaints related to JPMorgan Chase products. The system will process each complaint, validate its content, and predict both the product and the department responsible for resolving it. Afterward, the system will assign the complaint to an available support agent in the correct department who would be able to provide support for the specific product. By automating these steps, the project will help reduce resolution times and improve customer satisfaction. Additionally, the system will support bilingual complaints and use scalable cloud infrastructure to ensure continuous improvement and efficiency.

CHAPTER 2: DATA SOURCING AND PLANNING

This project uses a dataset of customer complaints from JPMorgan Chase, covering products like credit cards, loans, mortgages, and savings accounts. Each complaint includes a text description of the issue, along with structured fields like product type, sub-product type, and the date the complaint was received. This dataset is essential for training models to classify complaints by product and department, helping route them to the correct team for resolution. The data will also be used to evaluate the accuracy of the predictive models.

Data Card:

Customer Complaints Dataset

• Size and Scope:

The primary dataset consists of approximately 43,000 customer complaints related to various financial products and services offered by JPMorgan Chase. The dataset encompasses complaints registered over the past five years, providing a comprehensive view of recurring customer issues and department-specific challenges.

• Dimensionality:

The dataset contains 18 features (columns), each capturing critical details about the complaints. A brief description of the features and their relevance is as follows:

Consumer Complaint Dataset		
Variable Name	**Data Type**	**Description**
Date received	Date	The date the complaint was received
Product	String	The type of financial product related to the complaint
Sub-product	String	Sub-category of the product
Issue	String	The specific issue described in the complain
Sub-issue	String	A more detailed description of the issue
Consumer complaint narrative	String	The narrative text of the consumer's complaint
Company public response	String	The public response provided by the company (if any)
Company	String	The name of the company involved
State	String	The state where the complaint originated
ZIP code	Numeric	The ZIP code associated with the complaint
Tags	String	Tags related to the complaint (if any)
Consumer consent provided	String	Whether the consumer provided consent to publish the complaint
Submitted via	String	The platform through which the complaint was submitted
Date sent to company	Date	The date the complaint was sent to the company

Company response to consumer	String	The company's response to the consumer's complaint
Timely response	String	Whether the company responded to the complaint in a timely manner
Consumer disputed	String	Whether the consumer disputed the company's response
Complaint ID	Numeric	A unique identifier for each complaint

Agent Dataset

• Size and Scope:

The synthetic agent dataset includes 1,000 records, each representing a unique support agent within JPMorgan Chase. The dataset is designed to simulate agent profiles for testing and development purposes.

• Dimensionality:

The agent dataset contains the following features:

Agent Dataset		
Variable Name	**Data Type**	**Description**
ID	Numeric	Unique identifier for the support agent
Name	String	Name of the support agent
Language	String	Language(s) the support agent is proficient in
Department	String	Department the support agent is assigned to
Product	String	Product the support agent specializes in
Availability	Boolean	Availability status of the support agent (True/False)

Data Source:

JPMorgan Chase Complaints Dataset: Historical complaints from customers of JPMorgan Chase derived from the Consumer Financial Protection Bureau (CFPB).

Support Agent Dataset: Created a synthetic collection of support agent profiles.

External Sources: Public datasets from regulatory bodies like the Consumer Financial Protection Bureau (CFPB).

Data Rights and Privacy:

To comply with data protection laws such as GDPR and CCPA, personally identifiable information (PII) will be anonymized. All data will be securely stored on cloud platforms with privacy measures in place. The system will handle only anonymized, non-sensitive information in line with legal requirements to safeguard customer privacy.

Data Planning and Splits

- **Preprocessing:**

1. We use the TF-IDF Vectorizer to convert the consumer complaint narrative into numerical features, capturing keywords in the complaints. A Multinomial Naive Bayes classifier is then trained on these features to predict the responsible department (The model is subjected to change). Domain knowledge is incorporated by understanding how specific products and sub-products map to departments, allowing for fine-tuning of predictions.
2. We calculate the time taken to resolve a complaint by using the date received column. To simulate the variability in real-world resolution times, we add a resolved time data column to have a mean time of 21 hours with a standard deviation of 15 hrs.

- **Loading:**

1. We will follow the ETL (Extract, Transform, Load) process, where the data is first extracted from its source (database), transformations and cleaning operations will take place, and then finally loaded into BigQuery.
2. Once the preprocessing steps are completed, the cleaned and enriched dataset will be stored in Google BigQuery. BigQuery provides scalable and efficient storage for large datasets, ensuring that data is easily accessible for further analysis and machine learning tasks.
3. This processed data will then be fed into the machine-learning model for training purposes. By storing the data in BigQuery, we ensure seamless integration with downstream tools for model training and development.

- **Splitting:**

1. The dataset will be split into training, validation, and testing sets using a time-based approach. This method involves dividing the data chronologically, ensuring that we train the model on older data while testing it on more recent complaints.
2. A time-based split ensures a real-world scenario, where models are typically trained on past data and expected to generalize to future cases. This split method provides a realistic assessment of the model's performance in predicting outcomes for newer complaints.

CHAPTER 3: GITHUB REPOSITORY

Repository link: GitHub Link

Folder structure:

- README.md: It will provide an overview of the project, installation instructions, and usage guidelines.
- /data: It will store the raw and processed datasets used for training and testing the model.
- /models: It will contain the saved machine learning models (e.g., trained models and model checkpoints).
- /notebooks: It will include Jupyter notebooks for exploratory data analysis (EDA), model training, and evaluation.
- /scripts: It will have Python scripts for data preprocessing, training models, and other core functionalities.
- /logs: It will store log files generated during model training, validation, and error tracking.
- /configs: It will have configuration files that manage parameters such as learning rate, batch size, and other hyperparameters.
- /results: It will be used for storing the output from models, such as predictions, evaluation metrics, and performance results.
- README: Include a README file with essential project information, installation instructions, and usage guidelines.

CHAPTER 4 : PROJECT SCOPE

A. Problems:

- The current systems have trouble understanding complaints and sending them to the right department and support agent. This causes delays in resolution and leads to customer dissatisfaction.
- Many times, complaints are not sent to the right department quickly. This makes the process slow and cost-inefficient for the bank.
- The current systems also struggle to handle complaints in different languages like English, Spanish, and Hindi using single UI.

B. Current Solutions:

- The current approach to the problems is below:
- Lengthy forms to fill the complaint.
- No proper handling of user input validation. For example, if a user picks the wrong option from a dropdown in the form, the complaint is assigned to the wrong department. This may lead to multiple ticket rerouting and delays.
- Unable to prevent harmful and unethical inputs. The current solution does not implement stricter checks to prevent unethical inputs.
- Incorrect routing results in poor customer experience and increased operational costs for the company due to inefficient handling.
- According to Clutch, answering services can cost up to $25 per hour, adding to the overall expense due to inefficient handling.
- No bi-lingual input support.

C. Proposed Solutions:

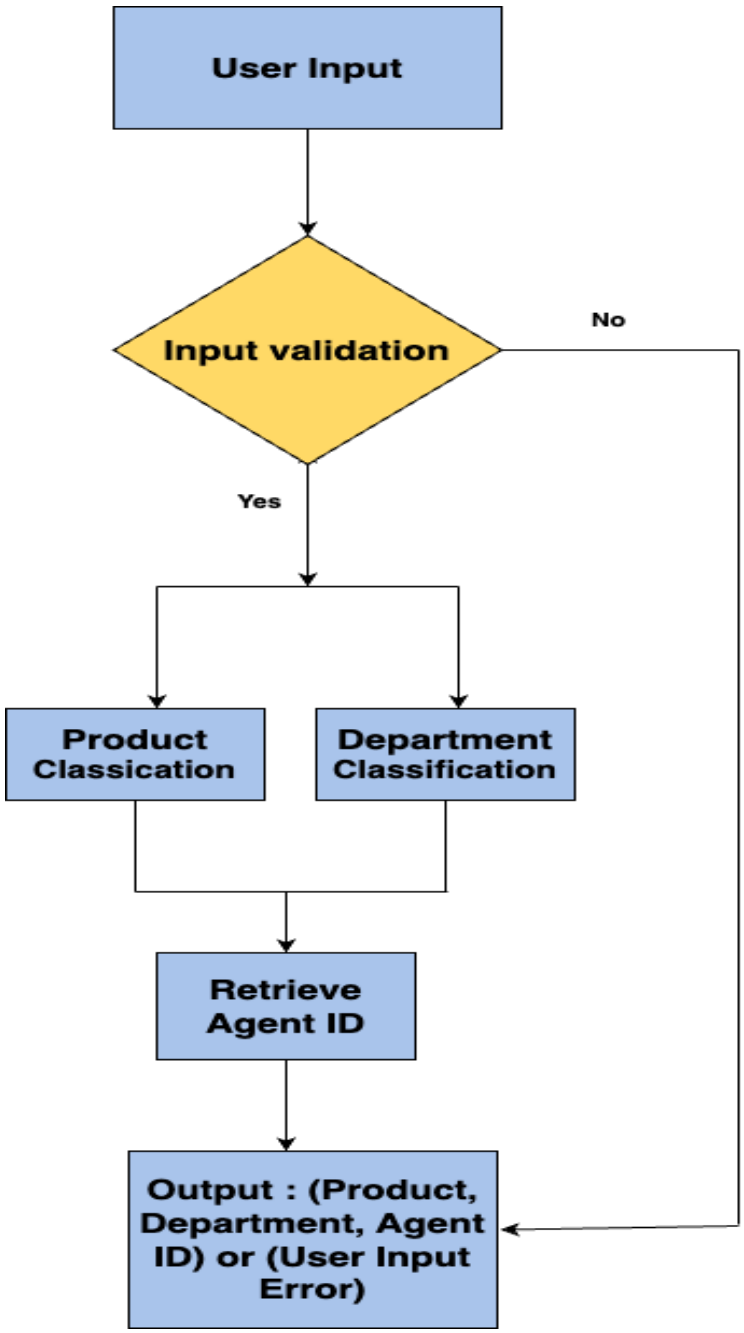
- Below is the proposed solution for the problems:
- A simplified UI where users can simply write their complaint in a text box and submit, eliminating the need for filling out lengthy forms.
- Use a single interface to process complaints in multiple languages (e.g., English, Hindi).
- The system will reject complaints that contain inappropriate or offensive language. Valid complaints will be automatically routed to the correct support agent, based on the product, and department extracted from the complaint.

- The proposed system will use a machine learning model to process bi-lingual complaints and extract key information to find the right product & department. It will then assign the complaint to a support agent proficient in that language. This ensures efficient resource use and faster resolution.
- Reduce the time to resolve tickets, which will improve user satisfaction and reduce the bank's operational cost and cost to resolve per ticket.

CHAPTER 5: APPROACH AND BOTTLENECK DETECTION

A. Potential Bottleneck:

The input validation stage can slow down the process as the system needs to review each customer complaint to ensure there's no inappropriate, abusive, or irrelevant content. This can be particularly time-consuming if complaints are submitted in multiple languages. Handling real-time validation becomes more challenging when complaint volumes increase or when bi-lingual complaints require extra processing.



B. Improvement Suggestions:

Parallel Processing: Process multiple complaints simultaneously to speed up validation.

Use Faster Algorithms: Implement faster, more efficient text processing methods or simple filtering techniques to quickly identify inappropriate content without requiring heavy computation.

CHAPTER 6: METRICS, OBJECTIVES & BUSINESS GOALS

A. Metrics:

1) Business Metrics: Reduce ticket resolution time.
2) Data Science Metrics: Improve F1-score, Precision, and Recall of product, department, and harmful/unethical model predictions.

B. Objectives:

1) Build a scalable, bi-lingual complaint processing pipeline.
2) Reduce complaint resolution time by efficient routing.
3) Improve accuracy in predicting the right product and department to assign customer service agents.

C. Business Goals:

1) Improve user experience by providing bi-lingual complaint registration support.
2) Automating routing of customer complaints to reduce complaint resolution time and improve customer satisfaction.

CHAPTER 7: FAILURE ANALYSIS

A. Risks During the Project:

1) Inaccurate department predictions due to poorly trained models

Risk: The machine learning model may not predict the correct department for complaint resolution, leading to inefficient routing.

Impact on Users: Users may experience significant delays in having their complaints resolved. This could result in frustration, reduced trust in the system, and a negative customer experience.

Mitigation: Continuously retrain the model using fresh data to improve accuracy.

2) Abusive Content Detection model might miss harmful language

Risk: The system may fail to catch all abusive content, which could escalate complaints that waste the support agent's time or result in negative customer interactions.

Impact on Users: Genuine complaints might be deprioritized while agents handle inappropriate submissions. Additionally, users may feel unsafe or unsupported if abusive complaints are processed improperly.

Mitigation: Implement a combination of keyword-based filtering and machine learning models for detecting abusive content. Regularly update the model to include emerging forms of harmful language.

3) Data Quality Issues

Risk: Poor data quality, including incomplete or incorrect complaints and mislabelled instances, can lead to model training issues and inaccurate predictions.

Impact on Users: Users may face incorrect or delayed resolutions due to misclassified complaints. This can result in dissatisfaction and reduced confidence in the system's reliability.

Mitigation: Implement data quality validation techniques and pre-processing steps to clean and verify data before it is fed into the model.

B. Post-Deployment Risks:

1) System Downtime or API Failures

Risk: Unexpected system downtime or API failures could delay the assignment of complaints to agents, delay complaint resolution, and ultimately impact customer satisfaction.

Impact on Users: Users may not be able to register complaints or receive timely responses. This could cause frustration and dissatisfaction, especially for urgent complaints.
Mitigation: Implement failover systems and redundancy, ensuring that the system continues to operate during outages. Use real-time monitoring API health checks to quickly detect and fix issues.

2) Bi-Lingual Complaint Misclassification

Risk: The model might incorrectly classify complaints in different languages, leading to delays or incorrect routing.
Impact on Users: Non-English-speaking users may feel excluded or underserved, resulting in poor user experience and potential dissatisfaction with the service.
Mitigation: Use bi-lingual models and regularly retrain them to handle the nuances of different languages effectively.

3) Legal and Privacy Risks

Risk: Failure to comply with data privacy regulations could lead to legal consequences or customer mistrust.
Impact on Users: Users may lose trust in the system if their personal data is mishandled, leading to reputational damage for the organization and reduced adoption of the service.
Mitigation: Ensure sensitive data is anonymized and that data handling practices comply with privacy regulations. Regularly audit the system for compliance.

C. Pipeline Failures:

1) Data Pipeline Failures During ETL

Risk: Issues in the ETL pipeline cause incomplete or corrupt data, which would lead to inaccurate model predictions.
Impact on Users: Users might experience misrouted complaints or delayed responses, leading to frustration and a lack of faith in the system.
Mitigation: Implement error-handling mechanisms to detect and resolve issues in the ETL process. Use automated quality checks to ensure data quality before it enters the system.

2) Model Performance Degradation

Risk: Over time, the model's accuracy might degrade as customer complaints evolve, causing inefficiencies.

Impact on Users: Users may see declining response quality over time, leading to repeated escalations and dissatisfaction with the system.

Mitigation: Regularly monitor data distribution & model drifts to monitor model quality, and ensure regular retraining to adapt to changing complaint patterns.

D. Scalability Issues:

1) High Complaint Volume Overload

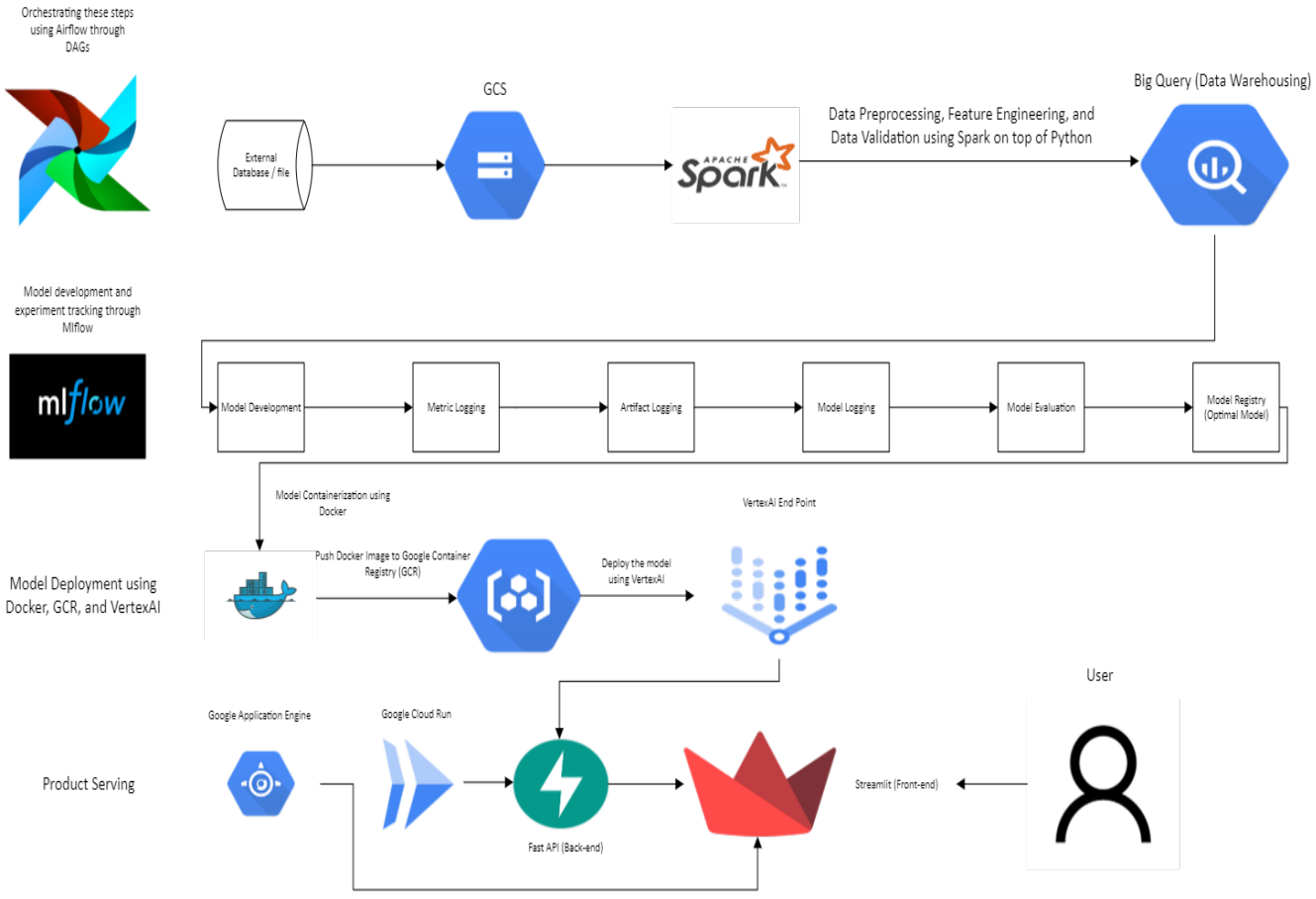
Risk: A surge in complaint volume could overwhelm the system, leading to delays or crashes.

Impact on Users: During high-demand periods, users may experience delays in submitting complaints or receiving resolutions, which could create dissatisfaction and impact the overall user experience.

Mitigation: Use cloud-based infrastructure with auto-scaling capabilities to dynamically adjust resources based on workload.

CHAPTER 8: DEPLOYMENT INFRASTRUCTURE

- **Cloud Composer:** Orchestrating the entire process with Airflow on Google Cloud Platform (GCP).
- **Cloud RUN / Vertex AI:** Serverless infrastructure for deploying machine learning models and APIs.
- **Dataproc / Apache Spark:** Data processing, Feature Engineering, and Data validation.
- **BigQuery:** For storing and querying transformed data and logs.
- **Google Cloud Storage:** For storing raw/unprocessed data.
- **GitHub & GitHub Actions:** Code repository and CI/CD pipelines to build pipelines and test new changes.
- **Docker:** Code Containerization
- **MLFlow:** Experiment tracking and model version control.
- **Google Container Registry:** A registry of docker images built from CI/CD pipelines that has packaged ML models and APIs.
- **App Engine:** Serving UI application server.



CHAPTER 9: MONITORING PLAN

We will use Airflow to compute monitoring metrics from different systems and store in Big Query for Analytics / Visualization.

A. Data Collection

1) Data Volume: Measure the number of complaints received.
2) Data Quality Metrics:
    i. Completeness: Percentage of missing values in the dataset.
    ii. Validity: Compliance of data against predefined formats.

B. Data Processing

1) Processing Time: Time is taken for data preprocessing pipeline for one record.

C. Orchestrating Workflows

1) Workflow Execution Time: Total time for a complete pipeline run.
2) Error Rate: Frequency of failures or errors in workflows.
3) Throughput: Number of jobs completed per time unit.

D. Model Training

1) Training Time: Duration taken to train the model.
2) Resource Utilization: CPU, GPU, and memory usage during training. Model
3) Performance Metrics:
    i. Precision: Ratio of true positive predictions to the sum of true positive and false positive predictions.
    ii. Recall: Ratio of true positive predictions to the sum of true positive and false negative predictions.
    iii. F1 Score: Harmonic mean of precision and recall.

E. Model Prediction Inference

1) Inference Time: Total time taken for inference pipeline.
2) Model Latency: Time delay between request and response from the Models.
3) Throughput: Number of predictions processed per second.

F. Model Monitoring

1) Drift Detection Metrics: Monitoring changes in data distributions over time (e.g., population stability index).
2) Performance Degradation Metrics: Regularly checking model precision, recall, and F1-score over time.
3) Alerts: Frequency of alerts triggered by performance degradation.

G. Model Logging

1) Log Completeness: Percentage of events successfully logged during training and inference.
2) Error Logging: Number of errors captured during model training and prediction phases.

CHAPTER 10: SUCCESS AND ACCEPTANCE CRITERIA

The success of this project will be determined by the system's ability to accurately classify customer complaints to the appropriate product, and department. A classification accuracy of at least 90% is required for the system to be considered effective. Additionally, the automated system should reduce complaint resolution time approximately by 40% compared to the current manual process.

In addition to it, the system must also handle complaints in multiple languages. And the system uptime should remain above 99% with acceptance latency.

Furthermore, it should be fully scalable using cloud infrastructure and ensure data privacy compliance with GDPR, ISO 27001, and CCPA standards.

CHAPTER 11: TIMELINE PLANNING

Week 1-2: Data Preprocessing and Setup

- Collect and clean the complaint data.
- Add a department column based on product/sub-product and narrative.
- Create the support agent dataset with availability simulation.

Week 3-4: Model Training

- Train machine learning models for product, department, and abuse classification.
- Evaluate models and adjust based on feedback.

Week 5: Agent Assignment Logic

- Implement and test the logic for assigning support agents to complaints based on department and language.

Week 6-7: Logging and Retraining Pipeline

- Set up Apache Airflow for logging and automating model retraining monthly.
- Ensure logs are tracked for complaints and assignments.

Week 8: System Deployment

- Deploy models and APIs using Google Cloud (Vertex AI, Cloud RUN and Compose).
- Test the system in a controlled environment.

Week 9-10: Documentation and Presentation Preparation

- Finalize documentation and validate the system against performance metrics.
- Prepare for the final project presentation.

CHAPTER 12: ADDITIONAL INFORMATION

We will provide an email of the general complaints department of the company, on the UI, in case the ML model does not allow the user to submit a complaint.
A thumbs-down button will be provided to send us a signal that the user was not able to submit a valid complaint (when the abuse detection model mispredicts).

Assumptions:

1) It is assumed that the user is already authenticated (logged in) while filing the complaint.
2) The complaint is being submitted by a valid and verified customer.

Cost and Resource Estimation:

- Google Cloud Storage: $0.02/month for storing pre-processed data and model artifacts.
- Big Query : $0.02/month for querying and storing data
- Vertex AI Workbench: $10.00/month for preprocessing and exploration of 200 MB of data.
- Vertex AI Training: $4.00/month for training models on 200 MB of combined data (used for 11 months).
- Model Deployment: $50.00/month for serving models trained on 75 MB of processed data.
- Monitoring and Logging: $30.00/month for logging preprocessing steps, model metrics, and selecting the best-performing model.
- Total Monthly Estimate: $100/month