Low-Level Design (LLD) Document

Fraudulent Transaction Detector Low-Level Design

Date: April 30, 2025

Author: Sudhanva Satish DA24M023

Overview

This document details the low-level design of the Fraudulent Transaction Detector, focusing on module interactions, API endpoints, and I/O specifications.

Software Paradigm

Object-Oriented (OO): The application uses classes (e.g., FastAPI app, Prometheus metrics) and objects (e.g., model, scaler). Functional elements are present in data transformations (e.g., pandas operations).

Module Breakdown

➢ Frontend (app.py):

- Language: Python with Streamlit
- Functionality: Handles UI, file upload, prediction display, and feedback collection.
- Key Functions:
  - st.file_uploader: Uploads CSV files.
  - st.data_editor: Displays predictions with checkboxes.
  - requests.post: Sends data to backend.

➢ Backend (main.py):

- Language: Python with FastAPI
- Functionality: Processes predictions and exposes metrics.
- Key Functions:
  - predict: Handles prediction requests.
  - health_check: Provides health status.
  - update_system_metrics: Updates Prometheus metrics.

➢ DVC Pipeline (ingest.py, transform_feedback.py):

- Language: Python with pandas
- Functionality: Manages data ingestion and transformation.
- Key Functions:
  - ingest_data: Combines raw and feedback data.
  - transform_feedback: Parses and transforms feedback.
  - preprocess: Preprocesses the saved and collated data
  - train: Trains a Random Forest model on the data

API Endpoint Definitions

➢ localhost:8000/predict

- Method: POST
- Input:
  - file: UploadFile (CSV file with columns Time, V1 to V28, Amount)
  - Content-Type: multipart/form-data
- Output:
  - Status: 200 OK
  - Response Body: JSON

```
{
  "predictions": [
    {
      "row": int,
      "data": object (dictionary of original row),
      "prediction": string ("Fraud" or "Not Fraud"),
      "fraud_probability": float (0 to 1)
    }
  ]
}
```

  - Status: 400 Bad Request
    - Detail: "File must be a CSV" or "CSV missing required columns. Missing: [...], Expected: [...]"
  - Status: 500 Internal Server Error
    - Detail: "Prediction failed: [error message]"
- Description: Processes the uploaded CSV and returns predictions.

➢ localhost:8000/health

- Method: GET
- Input: None
- Output:
  - Status: 200 OK
  - Response Body: JSON

```
{
  "status": "healthy"
}
```

- Description: Returns the health status of the backend.

Data Structures

- Prediction Response: Dictionary per row with row, data, prediction, fraud_probability.
- Feedback CSV: Columns: row, prediction, fraud_probability, original_data, feedback_correct.