High-Level Design (HLD) Document

Fraudulent Transaction Detector High-Level Design

Date: April 30, 2025

Author: Sudhanva Satish DA24M023

Overview

The Fraudulent Transaction Detector is a distributed application designed to predict fraudulent transactions using Machine Learning. It consists of a frontend interface, a backend API, and a data pipeline managed by DVC. The deployed web application can be monitored using a Grafana dashboard.

The model chosen is a Random Forest model trained on the Credit Card Fraud Dataset on Kaggle. Model training is tracked using MLFlow

System Components

➢ Frontend:

- Built with Streamlit, providing a web-based UI for users to upload CSV files, view predictions, and provide feedback.
- Runs in a Docker container, mounted to a local directory for feedback data storage.

➢ Backend:

- Implements a FastAPI service to handle prediction requests.
- Loads a pre-trained model and scaler, processes input data, and returns predictions with probabilities.
- Exposes Prometheus metrics for monitoring (e.g., request count, latency, system resources).
- Runs in a Docker container, communicating with the frontend via HTTP.

➢ Model Pipeline (DVC):

- Manages data ingestion, transformation, model training and feedback loop.
- Includes stages to ingest raw data, transform feedback data and combine data into processed_data.csv.
- Ensures version control and reproducibility of data processing.

Data Flow

- Input: Users upload a CSV file to the frontend.
- Processing: The frontend sends the file to the backend, which applies the model and returns predictions.
- Feedback: Users mark incorrect predictions, and the frontend saves feedback to

raw_feedback.csv.
• Transformation: The DVC pipeline transforms feedback into transformed_feedback.csv.
• Ingestion: The DVC pipeline combines creditcard.csv and transformed_feedback.csv into processed_data.csv for retraining.
• Model training/retraining: The train, test and validation data is loaded in and the model is trained by sweeping over a set of hyperparameters using MLFLow.