Hackathon Problem Statement

SabPaisa

TABLE OF CONTENTS

SabPaisa

Context

We are a Payment Gateway. We receive a payment transaction request from a payer to a payee. We authorise or reject the detection based on various parameters like enough funds in the payer account, approval from payer's bank, etc.

As part of the authorisation we also need to check if the transaction is fraudulent or not. Now labeling a transaction as fraudulent is not just a rule based deterministic process, it is a probabilistic dynamic process.

And this is where AI comes in. We want to combine the power of expert rules with that of AI models to detect frauds better.

We also want to monitor how our detection is faring against actually reported errors

What you need to Build

Build a Fraud Detection, Alert, and Monitoring (FDAM) system

The FDAM system comprises the below mentioned components. Build all of them.

SabPaisa

Fraud Detection API: Real time

- **Input**: A single transaction data in JSON format in the request body
- **Output**:

  {

  “transaction_id”: <string>,

  “is_fraud”: <boolean>,

  “fraud_source”: <string: “rule”/”model”>,

  “fraud_reason”: <string>,

  “fraud_score”: <float>

  }
- It should have a front end to configure the rules of the rule engine

SabPaisa

- It should have an AI model to predict `is_fraud`
  - The model can output just a boolean/binary value, or a score which then can be converted into a binary decision. In case of latter return the score in `fraud_score` field

- It should store all the input and output data to an external database in the table named `fraud_detection` with field `is_fraud` recorded as `is_fraud_predicted`

- It should be real time with average latency < 300 ms

Fraud Detection API: Batch

- **Input**: Data of multiple transactions in JSON format in the request body

- It should be using the Fraud Detection Real Time API in the backend

- Its should process multiple transactions in parallel but latency can vary depending on underlying server's memory and compute

- **Output**:

  {

  “<transaction id>”: {

  “is_fraud”: <boolean>,

  “fraud_reason”: <string>,

  “fraud_score”: <float>

  },

  ...

  }

Fraud Reporting API

- **Input:**

{

“transaction_id”: <string>,

“reporting_entity_id”: <string>“,

“fraud_details”: <string>,

}

- It too should store all the input and output data to the same external database as used by the Fraud Detection API in the table named `fraud_reporting` with field a `is_fraud_reported` set to True

SabPaisa

- **Output**:

  {

  “transaction_id”: <string>,

  “reporting_acknowledged”: <boolean>“,

  “failure_code”: <int>

  }

Transaction and Fraud Monitoring Dashboard

It should have the following

- A table to show raw transaction data including `is_fraud_predicted` and `is_fraud_reported` columns
  - Data should be filterable by
    - Dates
    - Payer ID
    - Payee ID
  - Data should be searchable by
    - Transaction ID
- A dynamic graph to compare the no. of predicted and reported frauds on the following dimensions:
  - Transaction Channel
  - Transaction Payment Mode
  - Payment Gateway Bank
  - Payer ID
  - Payee ID
- A time series graph to compare the trend of predicted and reported frauds
  - The time frame should be selectable
  - The granularity of time on x axis should dynamically vary as per the selected time frame and zoom levels
- A evaluation section showing confusion matrix, precision, and recall over a selectable time period

What will be provided to you

- A data file where each row will be a transaction. In columns you will have transaction details as received by the Fraud Detection API as input and also a column `is_fraud_reported` which will be a boolean telling if the transaction was actually a fraud or not. Below are the column names and their meaning:

- A sample request each for each of the APIs

SabPaisa

Key	Value
transaction_id	Unique transaction Id
transaction_date	Date on which txn is happening
transaction_amount	Amount of the transaction
transaction_channel	Like, web, mobile
transaction_payment_mode	Card, UPI, NEFT...
payment_gateway_bank	Service Bank
payer_email	Sender email
payer_mobile	Sender mobile
payer_card_brand	Sender Card like
payer_device	Device Id
payer_browser	Web browser
payee_id	Unique payee Id

How will you be evaluated

Deliverable	Weightage
Fraud Detection API: Real Time 1.Rule Engine with frontend 2.AI model 3.API responding or not? 4.API latency 5.API storing the data in a DB or not?	50% 30% 40% 10% 5% 15%
Fraud Detection API: Batch 1.API working or not? 2.Is processing transactions in parallel?	5% 40% 60%
Fraud Reporting API	5%
Database	10%
Dashboard Raw transaction table Graph to see groupings by dimensions Time series graph Evaluation section	30% 30% 20% 20% 30%

SabPaisa

Thank You

www.sabpaisa.in

SabPaisa