oject Planning Phase

**Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)**

Date

18 October 2022

Team ID

Team-591868

Project Name

**Online Payments Fraud Detection Using ML**

Maximum Marks

8 Marks

TABLE I. Previous Findings for Other Studies

**Paper**

**Used dataset**

**Techniques used**

**Performance**

**Limits**

[13]

Credit card dataset

SVM, LR, and neural networks

The support vector machine beats the others

The precision of the ANN is around twelve percent less than that of both of the models

[14]

Online transactions

XGBoost, and Fully Connected Neural Network (FCNN)

XGBoost reaches 0.912 and FCNN 0.969

The system can’t identify malicious transactions in real-time as they occur

[15]

Online transactions

artificial bee colony model and k-means

Results showed up to 100% True positive and less than 2% False Positive

Quadratic Discriminant Analysis give the fewer accuracy

[16]

Online transactions

Tailored alert model for detecting fraud in online transactions

The suggested approach beats the rule- based paradigm and the Markov chain method.

The suggested methodology detects fraud by using regular patterns; however, it will only identify scams when individuals display considerably different trading habits than typical.

[17]

real-world e-commerce transaction data

temporal attention-based Bi- LSTM, pHDBSCAN

Results show that the proposed method successfully detects lacking suspicious transactions having excellent business value.

Unable to identify low frequency of fraudulent transaction

[18]

internet-based e- transactions (credit card details data and trading)

Spark streaming and Kafka. DT, support vector machine, and CNN

The findings show that the proposed strategy produces satisfactory results.

The outcomes need to be improved

[8]

Credit card dataset

Spark GraphX, Hadoop, and graph embedding technique Node2Vec

The findings indicate that the proposed strategy enhances the precision and accuracy of Online fraudulent transaction detection systems.

The suggested model will be enhanced to successfully learn the newly generated features, resulting in better identification of fraud.

[19]

Credit card dataset

Microsoft Azure, Extreme Random Trees, and Stochastic Gradient Descent.

Good accuracy

Does not handle the class imbalance problem

[20]

Credit card dataset

IForest and LOF

The findings show that Isolation Forest beats the local outlier factor within 0.99774 of accuracy. The fraud detection percentage is about 0.27, whereas the LOF discovery rate is scarcely 0.02.

The LOF learner yield low performance

[21]

Credit card dataset

IForest and LOF

The experiments provide good results.

LOF give the worst results

**Layer**

**Description & prerequisites**

**Implementation**

**choices**

Events streaming

Refers to events streaming from digital banking applications.

This component must publish events as soon as they occur.

Kafka-connect. Kafka producer API

Data capture

Refers to events captured in a resilient way as well as making them available to different consumers.

Apache Kafka

Fraud prevention

Refers to real-time fraud prevention while transactions are in motion. This step must respond with a significantly reduced latency, given that the end-user would be blocked until this prevention is performed.

Apache Kafka Streams

Fraud detection

Refers to detection of fraud in a non-deterministic way, affecting a score to each transaction and persisting information about suspicious transactions.

Apache Spark Spark Streaming H2O

PostgreSQL

Monitoring

Refers to making potential fraud alerts available to human supervisors that could analyze and eventually contact end- users and perform curative actions accordingly.

React NodeJS

Alerting

Refers to raising alerts once a suspicious alert is confirmed to be fraudulent. These alerts could be consumed afterward by third-party consumers for actions such as account blocking and SMS notifications...

Kafka-connect. Kafka producer API

**Component**

**Servers / Characteristics**

Spark streaming / H2O

Driver : CPU: 1 core

RAM: 4 Go

Storage: 50 Go

Worker 1 :

CPU: 2 cores

RAM: 8 Go

Storage: 50 Go

Worker 2 :

CPU : 2 cores RAM : 8 Go

Storage : 50 Go

Worker 3 :

CPU : 2 cores RAM : 8 Go

Storage : 50 Go

Kafka

Broker 1 :

CPU : 2 cores RAM : 8 Go

Storage : 50 Go

Broker 2 :

CPU : 2 cores RAM : 8 Go

Storage : 50 Go

Broker 3 :

CPU : 2 cores RAM : 8 Go

Storage : 50 Go

Monitoring application

Application server / Database: CPU : 2 cores

RAM : 8 Go

Storage : 50 Go

**Performance metrics**

**Formulas**

Precision:

TP⁄TP + 𝐅P (3)

Recall:

TP⁄(TP + 𝐅N) (4)

Accuracy:

((TP + TN)⁄(TP + TN + 𝐅P + 𝐅N)) (5)

F1 score:

(precision × recall)

2 × (6)

(precision + recall)