Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	03 November 2022
Team ID	Team
Project Name	Online Payments Fraud Detection Using MI
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table

2 **Example: Order processing during pandemics for offline mode**

Reference: https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/

Guidelines:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructural demarcation (Local / Cloud)
3. Indicate external interfaces (third party API's etc.)
4. Indicate Data Storage components / services
5. Indicate interface to machine learning models (if applicable)

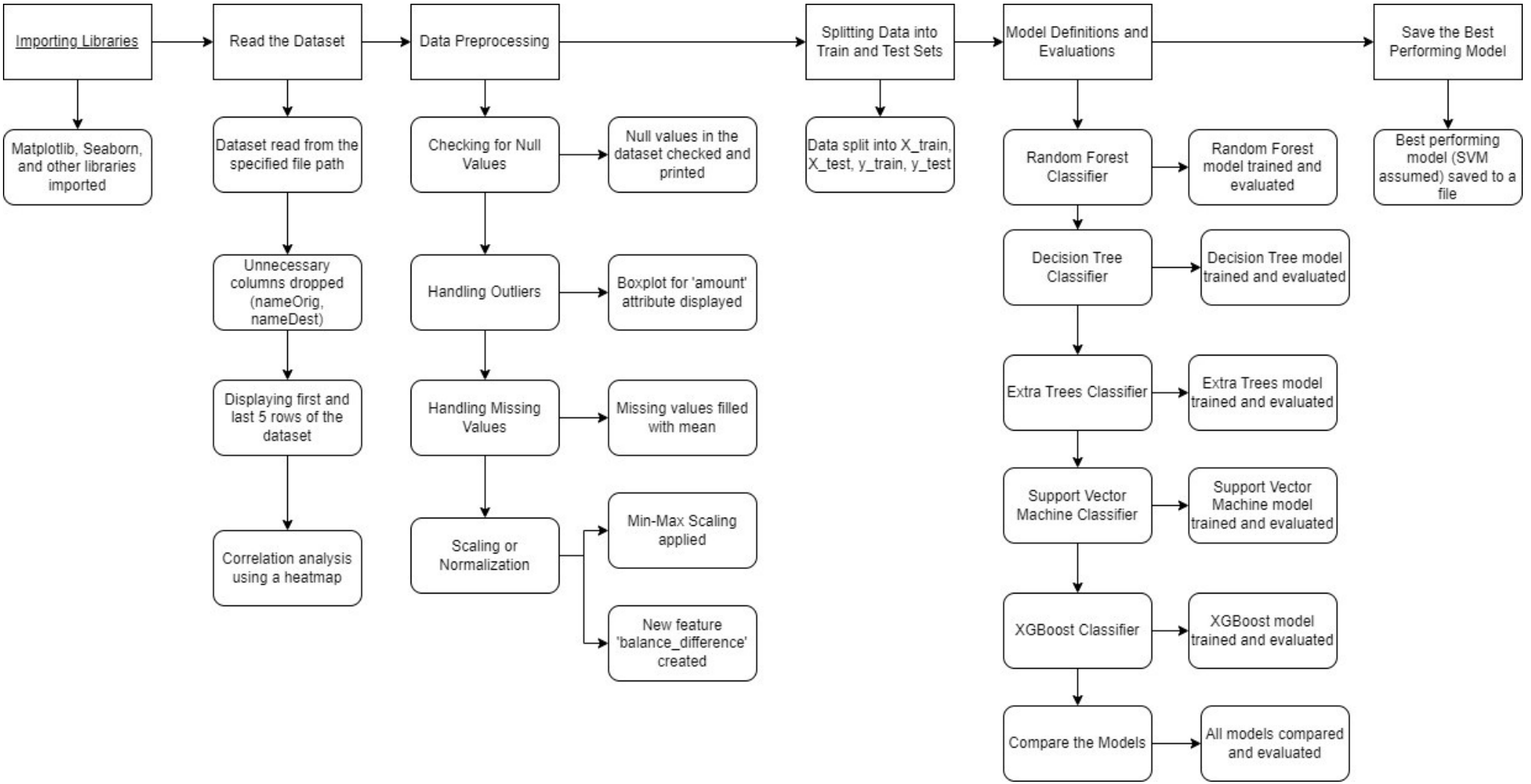
Importing Libraries
Matplotlib, Seaborn, and other libraries imported
Read the Dataset
Dataset read from the specified file path
Unnecessary columns dropped (nameOrig, nameDest)
Displaying first and last 5 rows of the dataset
Correlation analysis using a heatmap
Data Preprocessing
Checking for Null Values
Null values in the dataset checked and printed
Handling Outliers
Boxplot for 'amount' attribute displayed
Handling Missing Values
Missing values filled with mean
Scaling or Normalization
Min-Max Scaling applied
New feature 'balance_difference' created
Splitting Data into Train and Test Sets
Data split into X_train, X_test, y_train, y_test
Model Definitions and Evaluations
Random Forest Classifier
Random Forest model trained and evaluated
Decision Tree Classifier
Decision Tree model trained and evaluated
Extra Trees Classifier
Extra Trees model trained and evaluated
Support Vector Machine Classifier
Support Vector Machine model trained and evaluated
XGBoost Classifier
XGBoost model trained and evaluated
Compare the Models
All models compared and evaluated
Save the Best Performing Model
Best performing model (SVM assumed) saved to a file

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	Data Sources	Gather transaction data from online payment sources, including details such as transaction amount, user information, and device details.	APIs, Data Warehouses, Database ConnectorsETL Tools: Apache Nifi, Talend
2.	Data Cleaning and Transformation	Handle missing data, outliers, and normalize or standardize features. Prepare the data for machine learning model training.	Python (Pandas), Apache Spark
3.	Feature Extraction	Create new features that provide valuable information for fraud detection, such as transaction frequency, velocity checks, and anomaly scores..	Python (Scikit-Learn)
4.	Fraud Detection Models	Train machine learning models on historical data. Common algorithms include decision trees, random forests, logistic regression, and neural networks.	Scikit-Learn, TensorFlow, PyTorchS
5.	Real-time Transaction Monitoring	Implement a system for real-time monitoring of incoming transactions. Flag transactions that exceed a certain risk threshold.	Apache Kafka, Apache Flink, Streaming APIs
6.	Fraud Detection Dashboard	Develop a user interface for fraud analysts to review and analyze detected fraud cases. Provide visualizations and tools for investigation.	React.js, Angular, Vue.js

7.	Alerting System	Send alerts or notifications to relevant stakeholders (fraud analysts, users, etc.) for further investigation when suspicious activity is detected.	Email, SMS, Push Notifications
8.	Data Storage	Store transaction data, model parameters, and other relevant information. Enable efficient retrieval for analysis and reporting.	PostgreSQL, MongoDB, Cassandra
9.	Security Protocols	Implement security measures to protect sensitive information and secure communication channels	SSL/TLS, Encryption Algorithms
10.	Incident Response System	Define and automate steps to be taken in case of a confirmed fraud. This may include freezing accounts, notifying users, and collaborating with law enforcement.	Workflow Automation Tools
11.	Model Feedback Mechanism	Implement a feedback loop to continuously improve the model. Collect feedback from fraud analysts and use it to retrain the machine learning model.	Data Collection Tools
12.	Documentation and Logging	Document the entire process, including requirements, designs, technology choices, and logging of system activities for auditing purposes.	Logging Frameworks

Table-2: Application Characteristics:

S.NO	Characteristics	Description	Technology
1.	Real-time Processing	Process transactions in real-time.	Apache Kafka, Apache Flink, Stream Processing Frameworks
2.	Scalability	Scale the system to handle increasing transaction volumes.	Cloud Computing (e.g., AWS, Azure, Google Cloud), Containerization (e.g., Docker, Kubernetes).

3.	Accuracy and Precision:	Use machine learning models with high accuracy and precision.	Scikit-Learn, TensorFlow, PyTorch, XGBoost for model development and training.
4.	Adaptability and Learning	Regularly update machine learning models to adapt to evolving fraud patterns.	Automated Machine Learning (AutoML) tools, Model Versioning and Deployment Platforms.
5.	User-Friendly Interface	Intuitive interface for fraud analysts.	Web Development Frameworks (e.g., React.js, Angular, Vue.js), Data Visualization Libraries
6.	Cross-Channel Analysis	Correlate data from different channels and devices.	Data Integration Platforms (e.g., Apache Nifi, Talend), Cross-Channel Analytics Tools.
7.	Integration with Existing Systems	Seamless integration with payment systems and databases.	API Integrations, Middleware Solutions.
8.	Regulatory Compliance	Adherence to regulatory requirements.	Encryption Algorithms, Access Control Mechanisms, Compliance Monitoring Tools.
9.	Continuous Monitoring and Alerting	Implement continuous monitoring and alerting mechanisms	Real-time Monitoring Tools, Alerting Systems (e.g., PagerDuty).
10.	Automation of Routine Tasks	Automate routine tasks to improve efficiency.	Workflow Automation Tools, Robotic Process Automation (RPA).
11.	Explainability and Transparency	Ensure machine learning models are interpretable.	Explainable AI (XAI) Libraries (e.g., SHAP, LIME), Model Interpretability Tools.

References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d