CREDIT CARD FRAUD DETECTION

UNDER GUIDANCE OF
MR. GOKUL YENDURI

TEAM MEMBERS

21MIC7171- MOHAMMAD NUZHAT KULSUM

21MIC7191- SHAIK AFIFA ALIYA

21MIC7147- GUNDA KARTHEEK

FIRST REVIEW

SUMMER INTERNSHIP

5- YEAR INTEGRATED MTECH

SOFTWARE ENGINEERING

21ST BATCH

DEPARTMENT OF SCOPE

CREDIT CARD FRAUD DETECTION

PURPOSE OF THE SYSTEM :

The purpose of a credit card fraud detection system is to identify and prevent fraudulent transactions in real-time or near-real-time to protect both the credit card holder and the issuing financial institution. This system aims to detect and block unauthorized or suspicious transactions before they are completed, thereby preventing financial losses. A key objective is to ensure accuracy and precision, minimizing false positives (legitimate transactions flagged as fraudulent) and false negatives (fraudulent transactions that go undetected). Real-time monitoring is essential for continuously tracking transaction activity to identify anomalies and unusual patterns that may indicate fraud. By safeguarding the credit card holder's financial assets and personal information, the system enhances user protection and boosts customer trust in the financial institution. Additionally, it helps financial institutions reduce the risk of financial losses and meet regulatory requirements related to fraud prevention and data protection. Ultimately, the credit card fraud detection system aims to ensure the security and integrity of financial transactions, protect users' funds, and maintain the trust and reputation of financial institutions, all while being cost-efficient.

SYSTEM REQUIREMENT SPECIFICATION:

A detailed explanation of the system requirements specifications is provided below:

HARDWARE REQUIREMENTS PROCESSOR:

Intel i5 or later To efficiently handle the computational demands of machine learning and deep learning algorithms, the system requires a processor with at least Intel i5 capabilities. This ensures sufficient processing power for training models and performing complex data analyses.

RAM: 512GB RAM :

The system demands a substantial amount of RAM, specifically 512GB, to manage large datasets and support extensive parallel processing tasks during model training and inference phases, ensuring smooth and efficient operation.

Hard Disk:

PC with 20GB or more A minimum of 20GB of hard disk space is necessary to store essential software, datasets, and temporary files generated during data

preprocessing and model training. This ensures adequate storage capacity for smooth system operation.

SOFTWARE REQUIREMENTS OPERATING SYSTEM:

Windows XP or later :

The system is compatible with Windows XP or later versions, providing flexibility and ease of use for users operating in a Windows environment. This ensures broad compatibility and user accessibility.

Tools: Google Colab, Kaggle, VS Code :

Google Colab: An online platform that allows users to write and execute Python code in a web-based notebook environment, providing access to free GPU resources for training machine learning & deep learning models.

Kaggle: A platform for data science and machine learning competitions, offering a vast repository of datasets and pre-built notebooks for experimentation and collaboration.

Visual Studio Code : (VS Code) is a free, open-source code editor developed by Microsoft. It is popular among developers for its versatility and a wide range of features. Here are some key aspects of VS Code:

Languages Used:

Python Python is the primary programming language used for developing the system, due to its extensive libraries and frameworks for machine learning and deep learning, such as TensorFlow, Keras, and PyTorch. Python's readability and ease of use make it ideal for rapid prototyping and development

PROBLEMS IN THE EXISTING SYSTEM

Imbalanced Data: Fraudulent transactions are rare compared to legitimate ones, leading to biased models that may miss detecting fraud.

Data Quality: Poor data quality (incomplete, noisy data) can reduce the accuracy of fraud detection models.

Limited Adaptability: Logistic reasoning methods may not adapt well to evolving fraud patterns or new types of fraud that emerge over time. They rely heavily on predefined rules and may struggle to adjust to novel fraud techniques.

Scalability Issues: As the volume of credit card transactions increases, logistic reasoning methods might face challenges in processing large amounts of data efficiently. This could impact real-time fraud detection capabilities.

SOLUTION OF THESE PROBLEMS :

To address imbalanced datasets in fraud detection, techniques like resampling adjusting class weights, or employing algorithms like penalized models or ensembles are effective strategies to balance class representation and improve model performance.

Improve data preprocessing by cleaning and validating data to reduce noise and ensure accuracy. Incorporate automated checks to flag and correct data inconsistencies.

Maintain constant observation of data streams and model performance. When new fraud patterns appear, use adaptable methods of learning to update the model instantly For efficiency.

SCOPE OF THE PROJECT :

The project's goal is to build an effective system that can identify credit card fraud. it monitor transactions and look for patterns to identify suspicious activity using machine learning.

Possible fraud will be identified through pattern, and reports will help investigators quickly look into incidents. It providing a dependable solution that improves transaction security and lowers financial losses for clients and banks is the ultimate objective.

FUNCTIONAL COMPONENTS OF THE PROJECT :

1. Data Collection Module
    - Transaction Data Acquisition:
        - Gather transaction data from financial institutions or public datasets (Kaggle Credit Card Fraud Detection dataset).
        - Include attributes such as transaction amount, time, and anonymized features.
2. Data Preprocessing Module
    - Data Cleaning and Normalization:
        - Handle missing values by imputation or removal.
        - Remove duplicates to ensure data quality.

- Normalize features using techniques such as Min-Max scaling or Standardization.

- Class Imbalance Handling:
  - Apply techniques like SMOTE (Synthetic Minority Over-sampling Technique) to balance the dataset.

3. Exploratory Data Analysis (EDA) Module
   - Descriptive Statistics:
     - Calculate and display summary statistics for the dataset.
   - Data Visualization:
     - Use histograms, box plots, pair plots, and heatmaps to understand feature distributions and correlations.
4. Feature Engineering Module
   - Feature Selection:
     - Identify and select relevant features for fraud detection.
   - Feature Creation:
     - Develop new features if necessary to enhance model performance.
5. Model Development Module
   - Machine Learning Algorithms:
     - Implement models such as Logistic Regression, Decision Trees, Random Forest
6. Result Visualization Module
   - Diagnostic Results Display:
     - Visualize transaction data and detection results using dashboards.
   - Alert System:
     - Implement alert systems to notify stakeholders of potential fraudulent transactions.
   - Monitoring:
     - Continuously monitor model performance and update it as needed.
7. Documentation and Reporting Module

- Project Documentation:
  - Document the entire process, including data preprocessing, model development, evaluation, and deployment.
- Results Presentation:
  - Prepare presentations to showcase the project's results and insights.

These components ensure that the credit card fraud detection system is comprehensive, accurate, and effective in identifying fraudulent activities.

STUDY OF THE SYSTEM :

1. Introduction

- **Objective:** To develop a robust and reliable system for detecting fraudulent credit card transactions using machine learning and deep learning techniques.
- **Importance:** With the rise in online transactions, credit card fraud has become a major concern for financial institutions and customers. An effective detection system can minimize financial losses and enhance transaction security.

2. System Overview

- **Data Source:** The system utilizes transaction data, including attributes such as transaction amount, time, and anonymized features, from financial institutions or publicly available datasets like the Kaggle Credit Card Fraud Detection dataset.
- **Architecture:** The system includes modules for data collection, preprocessing, model development, evaluation, deployment, result visualization, and user recommendations.

3. Technical Feasibility Assessment

- **Hardware Requirements:** Ensure the system has adequate hardware, including at least Intel i5 or later processors, 16GB RAM, and 512GB of hard disk space, to handle intensive computational tasks and large datasets.
- **Operating Systems:** The system runs on Windows 10 or later, Linux, or macOS.
- **Development Tools:** Utilize tools such as Jupyter Notebook, Google Colab, Kaggle, and Visual Studio Code for development and deployment, with Python as the primary programming language.

- **Cloud Platforms:** Leverage cloud platforms like AWS, Azure, or Google Cloud for scalable resources for model training and deployment.

4. Validation and Testing

- **Model Validation:** Validate models against a large dataset of annotated transactions to ensure accuracy and reliability.
- **Performance Review:** Continuously review and refine models based on feedback and performance metrics.
- **Integration:** Ensure the system integrates seamlessly with existing transaction processing systems and fraud detection workflows.

5. Conclusion

- **Impact:** An effective credit card fraud detection system can significantly reduce financial losses and enhance security.
- **Future Work:** Continuously improve the model with new data and advanced techniques. Explore more sophisticated algorithms and real-time detection capabilities.

This comprehensive approach ensures the credit card fraud detection system is accurate, reliable, and user-friendly, ultimately improving transaction security and reducing financial losses.

THE MODULES THAT ARE INVOLVED:

1. Data Collection
2. Data Preprocessing
3. Model selection and Training
4. Logistic regression
5. XG boost
6. Decision tree
7. Random forest
8. SHAP
9. Model Evaluation and Validation
10. Evaluation Metrics

DATA COLLECTION :

Data collection is a crucial step in any research process, as it involves gathering information that will be used to answer research questions, test hypotheses, and evaluate outcomes. There are several methods of data collection, each with its own advantages and disadvantages.

DATA PREPROCESSING :

Data preprocessing involves cleaning and transforming raw data to prepare it for analysis. This includes handling missing values, normalizing data, encoding categorical variables, and feature extraction. The goal is to improve data quality and ensure it is suitable for training machine learning models, ultimately enhancing model performance and accuracy.

MODEL SELECTION AND TRAINING :

Model selection and training are fundamental steps in the development of machine learning systems. These steps involve choosing the appropriate model for your task, training the model on your data, and evaluating its performance.

LOGISTIC REGRESSION :

Logistic Regression is a statistical method used for binary classification problems, where the outcome is a binary variable it has two possible outcomes. Despite its name, logistic regression is actually a linear model for classification rather than regression.

XG BOOST :

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. It implements machine learning algorithms under the Gradient Boosting framework.

SHAP :

SHAP is a game-theoretic approach to explain the output of any machine learning model. It connects optimal credit allocation with local explanations using the classic Shapley values from cooperative game theory and their related extensions.

DECISION TREE :

A decision tree is a supervised learning algorithm used for both classification and regression tasks. It is a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.

RANDOM FOREST :

Random Forest is an ensemble learning method used for classification, regression, and other tasks. It operates by constructing multiple decision trees during training and outputting the mode of the classes (classification) or mean prediction (regression) of the individual trees.

MODEL EVALUATION AND VALIDATION :

Model evaluation and validation are essential steps in the machine learning workflow to ensure that a model performs well on unseen data and is not overfitting or underfitting.

EVALUATION METRICS :

Evaluation metrics are used to assess the performance of machine learning models. The choice of metric depends on the type of problem classification, regression and the specific requirements of the application.

PERFORMANCE REQUIREMENTS :

- **Accuracy and Detection Metrics:**
  - Achieve an overall accuracy rate of 95% or higher.
  - Maintain precision and recall rates of 90% or higher.
  - Ensure an F1-score of 90% or above.
  - Attain a ROC-AUC score of 0.95 or higher.
- **Performance Efficiency:**
  - Process transactions with a maximum latency of 500 milliseconds.
  - Handle high throughput, processing thousands of transactions per second.
- **Scalability and Robustness:**
  - Scale effectively to accommodate increasing transaction volumes and user demands.
  - Remain robust against variations in data and anomalies.

FEASIBILITY REPORT :

1. **Technical Feasibility**

Hardware:

- Processor: Intel i5 or later.

- RAM: Minimum 16GB.
- Storage: At least 512GB SSD. Optional GPU for deep learning tasks.

Software:

- Operating Systems: Windows 10 or later, Linux, macOS.
- Development Tools: Jupyter Notebook, Google Colab, Kaggle, Spyder.
- Programming Languages: Python.
- Libraries and Frameworks: Scikit-learn, TensorFlow, PyTorch, Pandas, NumPy, Matplotlib, Seaborn.

Cloud Platforms:

- Utilize AWS, Azure, or Google Cloud for scalable resources for model training and deployment.

2. Operational Feasiibility

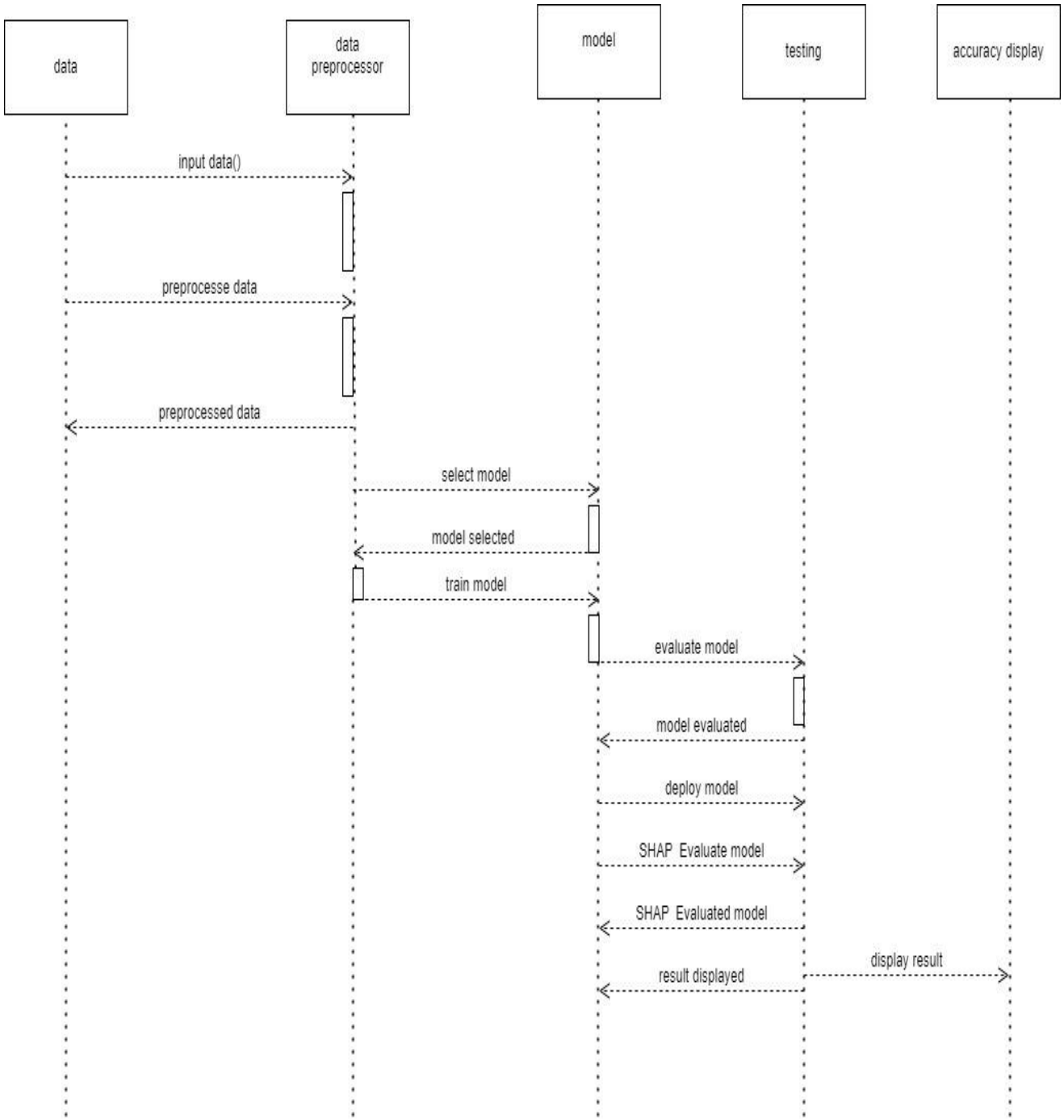
- **Implementation Plan:**
  - **Data Collection and Preprocessing:** Acquire and prepare transaction data.
  - **Model Development and Evaluation:** Build, validate, and test ML models.
  - **Deployment and Integration:** Deploy the model and integrate it with existing systems.
  - **Monitoring and Maintenance:** Continuously monitor and update the system.
- **Scalability and Robustness:**
  - Design for scalability to handle increasing data and user volumes.
  - Ensure system robustness against data variations and anomalies.

3. Economic Feasibility

- **Cost Analysis:**
  - **Development Costs:** Includes expenses for hardware, software, and development tools.
  - **Operational Costs:** Includes cloud services, maintenance, and support.
  - **Training Costs:** Includes resources and materials for training users.
- **Return on Investment (ROI):**

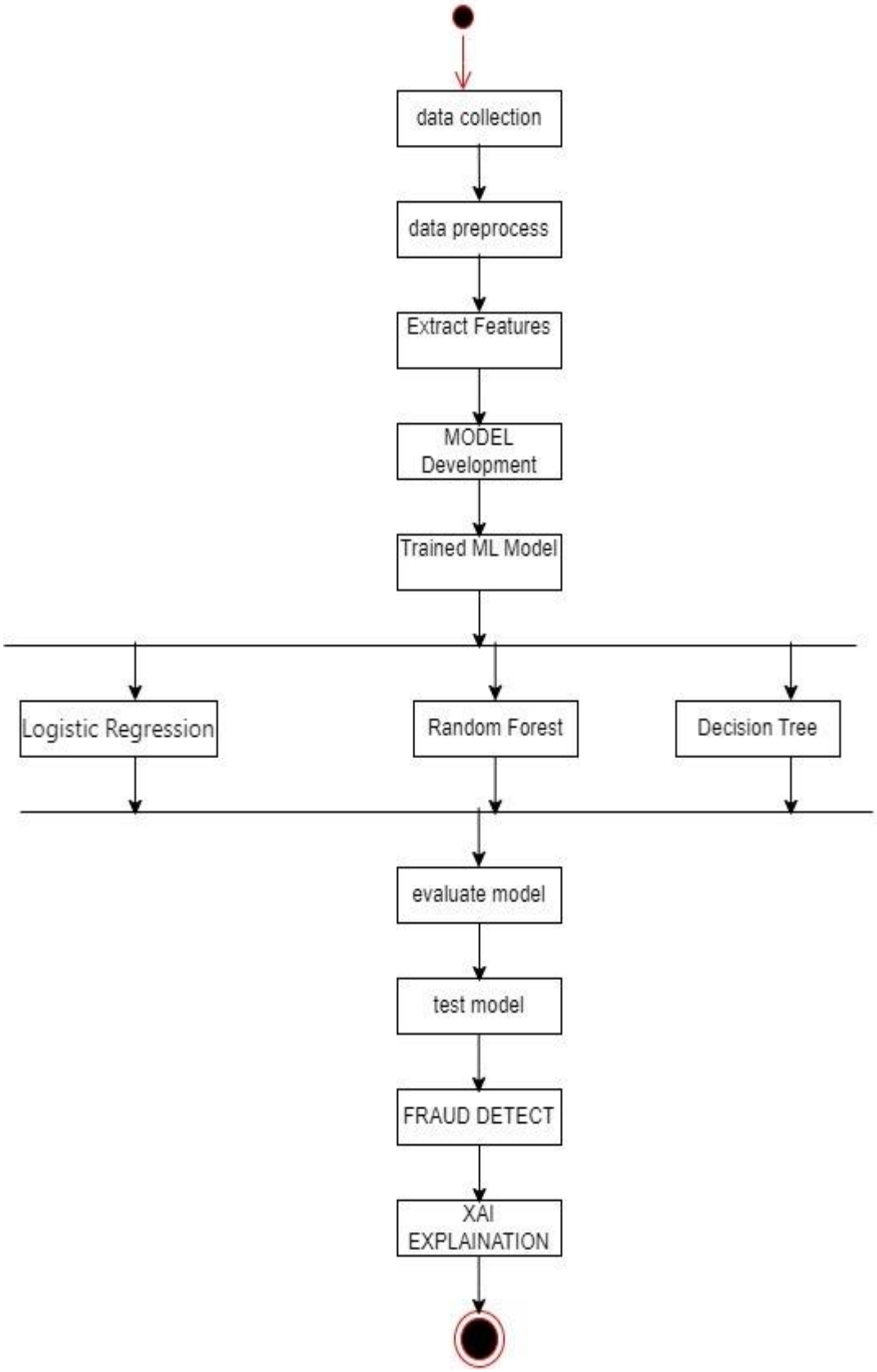
- **Cost Savings:** Reduction in financial losses due to fraudulent transactions.
- **Efficiency Gains:** Improved transaction security and processing efficiency.
- **Customer Trust:** Enhanced customer confidence and satisfaction due to better fraud detection.

SEQUENCE DIAGRAM



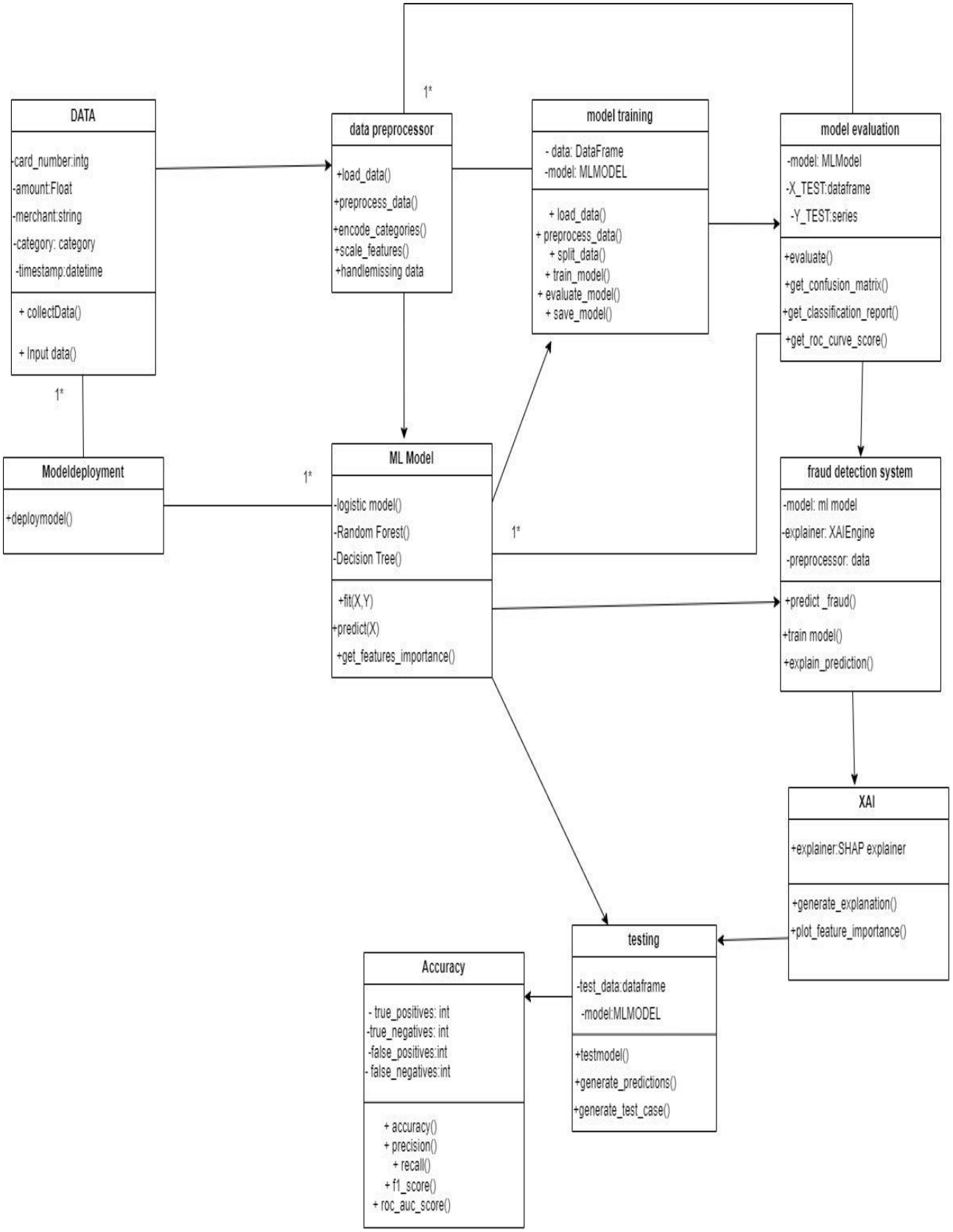
FIRST REVIEW

ACTIVITY DIAGRAM



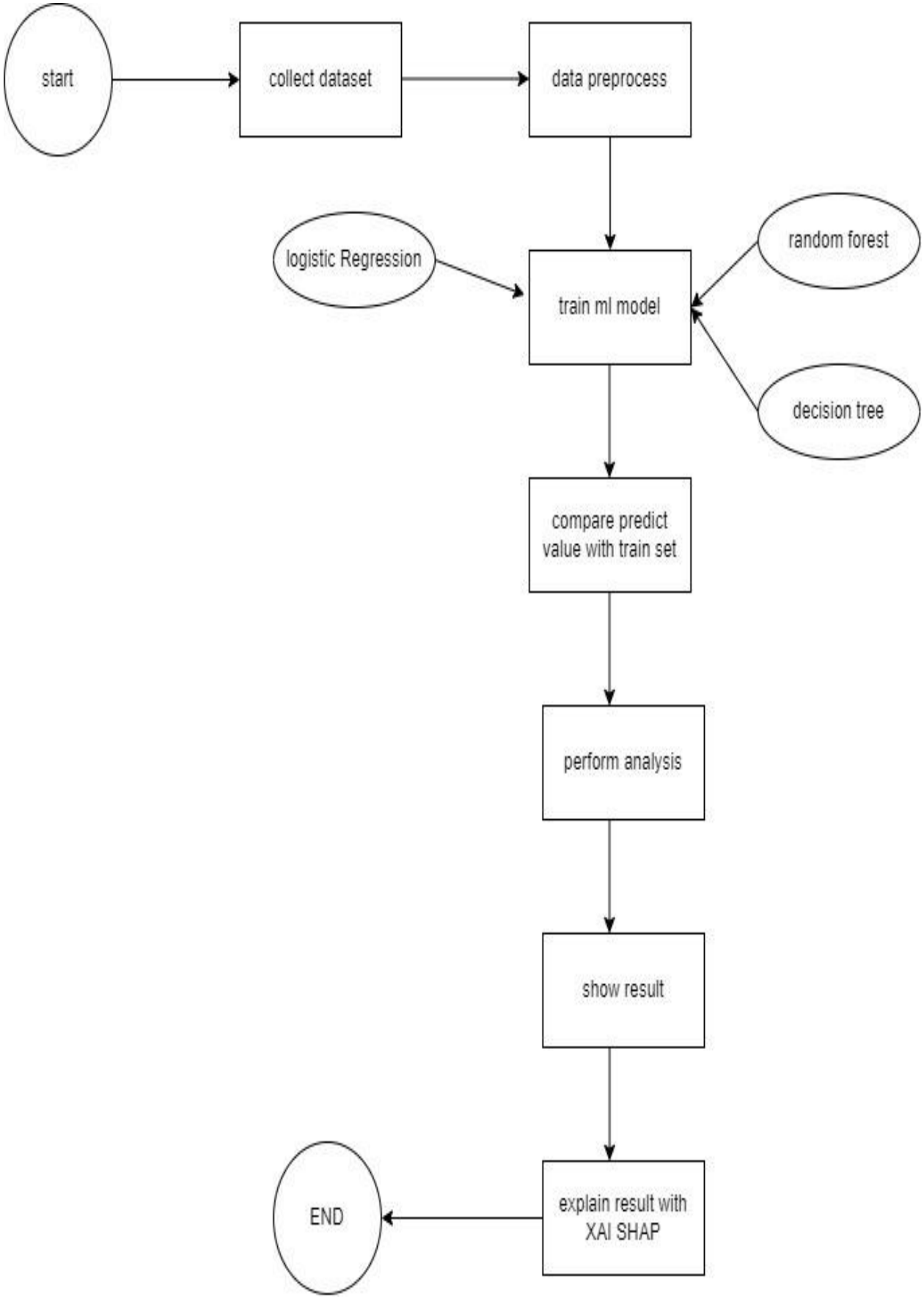
FIRST REVIEW

CLASS DIAGRAM



FIRST REVIEW

METHODLOGY DIAGRAM



FIRST REVIEW