“Credit Card Fraud Detection Pipeline Using Apache Spark and Hadoop”

Introduction:

Credit card fraud is one of the most serious challenges faced by banks, financial institutions, and online payment systems. As the number of digital transactions increases every day, detecting fraudulent activity in real time becomes extremely difficult using traditional systems. Millions of transactions occur within minutes, and fraud patterns often change quickly, making manual detection impossible.

In this project, we aim to build a scalable fraud detection pipeline that can analyze large volumes of credit card transactions and identify suspicious behavior efficiently. Using Apache Spark for distributed processing and Hadoop/HDFS for scalable storage, our system will simulate how modern financial companies detect fraud at scale. We will analyze transaction patterns, engineer risk-related features, and train machine learning models that can classify transactions as fraudulent or legitimate. This project demonstrates a complete end-to-end workflow: data ingestion, ETL, analytics, machine learning, and visualization all designed to operate on high-volume datasets and support real-world fraud detection scenarios.

Problem Statement

Credit card fraud is increasing rapidly as digital payments grow, and financial institutions must process millions of transactions each day. Traditional detection systems are not fast or intelligent enough to analyze large-scale data and identify patterns of fraudulent activity. Because fraudsters constantly change their behavior, it becomes challenging to detect fraud using manual rules or small datasets. There is a need for a scalable, data-driven pipeline that can process high-volume transactions, analyze patterns, and accurately classify fraudulent behavior.

Dataset Description:

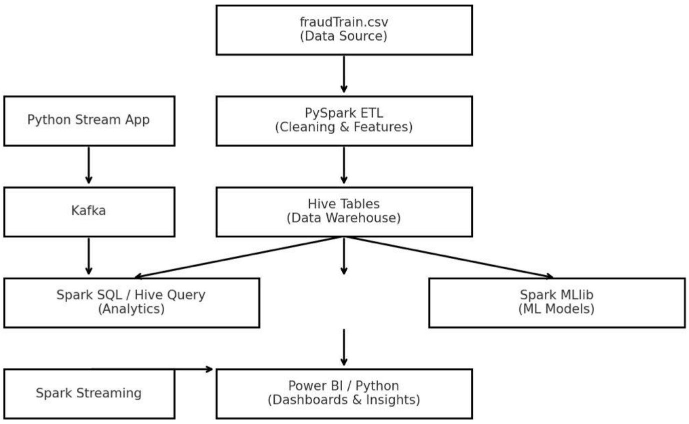
Link : https://www.kaggle.com/datasets/kartik2112/fraud-detection?select=fraudTrain.csv

We will use the “Credit Card Transactions Fraud Detection” dataset from Kaggle, which contains labeled records for both legitimate and fraudulent transactions. The primary file we will use is fraudTrain.csv, which includes fields such as transaction time, merchant category, transaction amount, customer demographics, and a binary fraud label. This dataset is suitable for building a fraud detection system because it captures realistic transaction patterns and provides enough attributes for feature engineering and machine learning. It also allows us to simulate a high-volume environment by scaling the data within a distributed system.

Project Objective:

The goal of this project is to build a **scalable fraud detection pipeline** that performs distributed ETL, fraud pattern analysis, and machine learning classification. We aim to preprocess data, engineer features, run analytics, and train models (such as Random Forest or Gradient Boosted Trees) using Spark MLlib. The pipeline will simulate real-world fraud detection used by banks.

Architecture Overview:



1. Batch ETL and Storage:

The fraudTrain.csv dataset is processed using PySpark for cleaning and feature engineering, and the curated data is stored in Hive tables on HDFS, which act as the central data warehouse.

2. Streaming and Real-Time Processing:

A Python-based transaction simulator sends events to Kafka, which are consumed by Spark Streaming to process real-time transactions and update fraud-related data back into Hive.

3. Analytics, Machine Learning, and Visualization:

Hive Queries and Spark SQL are used for analytical insights, while PySpark with Spark MLlib trains distributed fraud detection models. Power BI connects to Hive to generate dashboards and visualize fraud trends and model outcomes.

Big Data Technologies:

To build a scalable fraud detection system, we will use several big data technologies:

- Apache Spark – for distributed ETL, feature engineering, and machine learning
- Spark SQL / Hive – for large-scale querying and analytics
- Hadoop HDFS – for distributed data storage
- Kafka (optional) – to simulate streaming credit card transactions
- Spark MLlib – to build scalable fraud detection models
- Python + Jupyter Notebook – for experimentation and visualization
- Power BI or Tableau – for interactive dashboards and final insights

Pipeline Overview:

1. Load the fraudTrain.csv dataset into HDFS.
2. Perform distributed ETL using Apache Spark (cleaning, preprocessing, feature engineering).
3. Use Spark SQL / Hive to analyze fraud patterns and transaction behavior.
4. Train machine learning models using Spark MLlib to classify fraudulent transactions.
5. Generate visualizations and summary insights to support fraud detection.

(Optional) Simulate real-time transactions using Kafka and process them with Spark Structured Streaming.

Team Members:

1.Vrb9112 - Baireddy Venkata Devendhar Reddy

2.Rm7020- Rahul Mallidi

3.Sk12176- Sai krishna Kommineni

4.cn2507 -Hemanth Reddy