CSE-3120 BIG DATA FRAMEWORKS

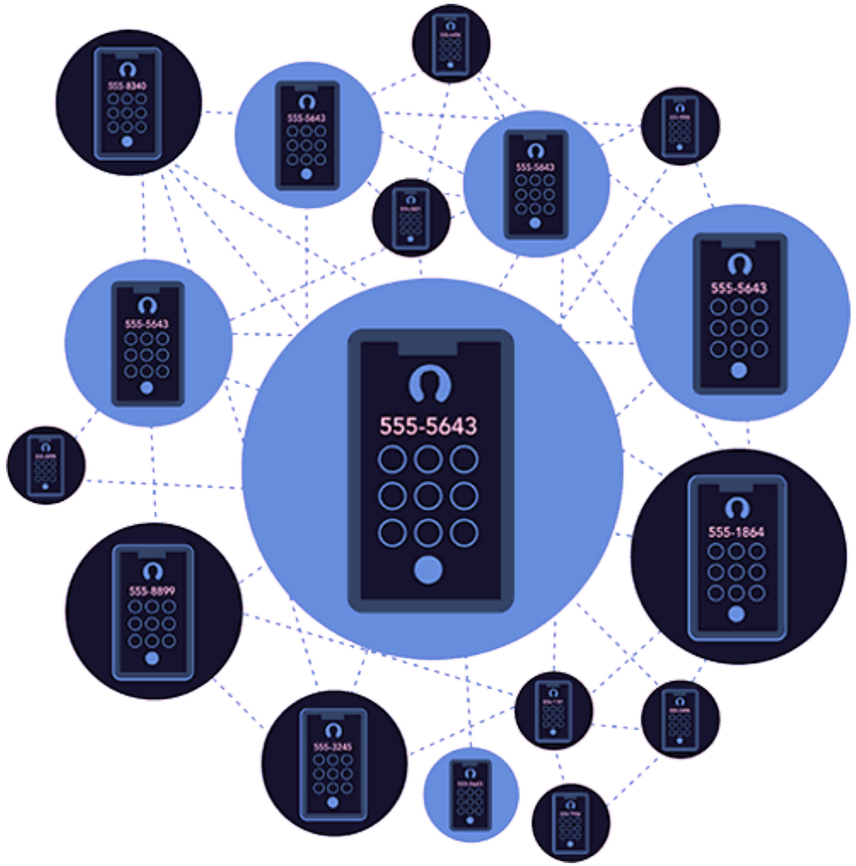
FRAUD DETECTION SYSTEM IN MOBILE TRANSACTIONS USING BIG DATA

REVIEW-3

ABSTRACT

- This project aims to develop a generalized model for fraud detection that can effectively handle big and imbalanced datasets, making it suitable for real-world applications.
- The analysis leverages the PySpark framework to handle the large-scale data processing requirements.
- The methodology involves various stages, including data understanding, conversion, feature engineering, data stratified splitting, model building, pipeline construction, cross-validation, and model evaluation.

The subject matter of "Fraud Detection inMobile Transactions using Big Data"

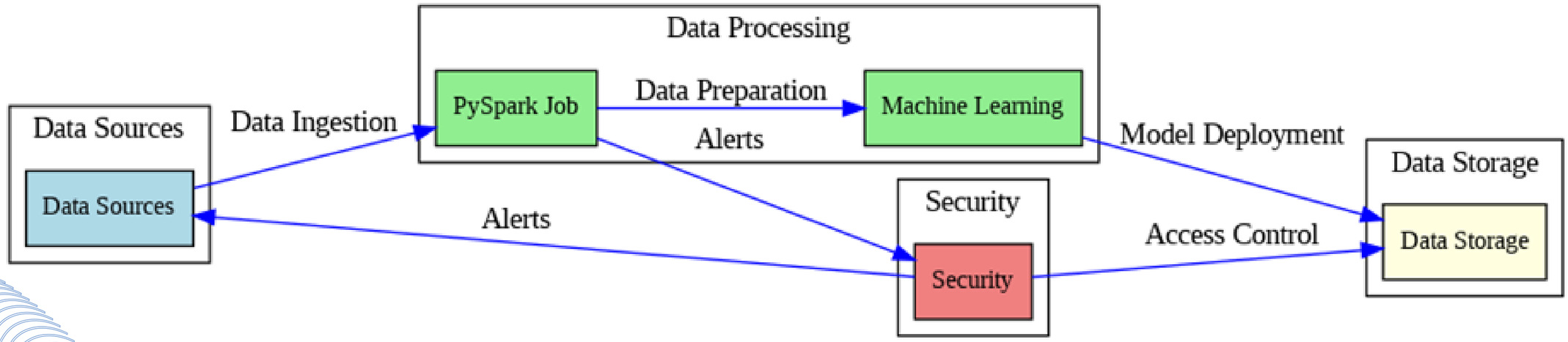


INTRODUCTION

- Fraud detection is a critical challenge in the financial services industry, particularly in the domain of emerging mobile money transactions.
- However, the lack of publicly available datasets on financial services poses a significant obstacle to researchers studying fraud detection.
- To address this issue, a synthetic dataset called PaySim was generated using aggregated data from private sources.
- PaySim simulates normal transaction operations while injecting malicious behavior, enabling the evaluation of fraud detection methods.

ARCHITECTURE

The proposed architecture for the fraud detection system consists of Data Sources, PySpark Job, Machine Learning (utilizing Support Vector Classification - SVC), Data Storage, and Security components. Transactional data is ingested from Data Sources into the PySpark Job for processing. The PySpark Job handles data preparation tasks, and the Machine Learning component utilizes SVC to train the model. Processed data and trained models are stored in Data Storage. The Security component monitors for alerts and ensures access control. This architecture enables efficient, scalable, and secure fraud detection using PySpark's distributed processing capabilities and SVC for accurate classification.



METHOD OVERVIEW

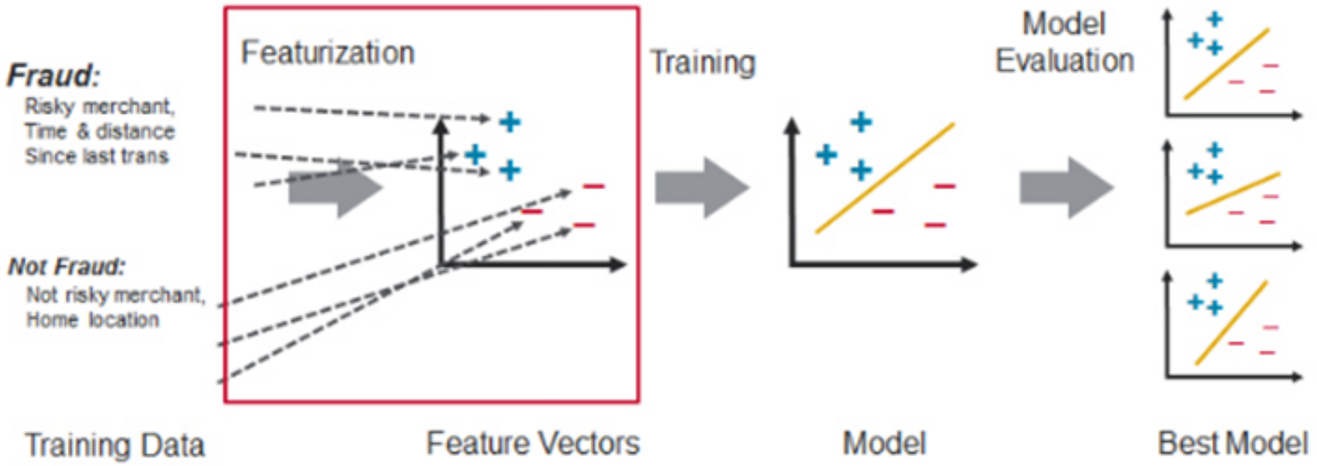
Data understanding using PySpark

Data conversion and Feature Engineering

Data stratified splitting and Model building

Pipeline and Cross Varaiation

Model evaluation



UNDERSTANDING THE DATA

step	maps a unit of time in the real world. In this case 1 step is 1 hour of time. Total steps 744 (30 days simulation).
type	CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER.
amount	amount of the transaction in local currency.
nameOrig	customer who started the transaction
oldbalanceOrg	initial balance before the transaction
newbalanceOrg	new balance after the transaction
nameDest	customer who is the recipient of the transaction
oldbalanceDest	initial balance recipient before the transaction. Note that there is not information for customers that start with M (Merchants).
newbalanceDest	new balance recipient after the transaction. Note that there is not information for customers that start with M (Merchants).
isFraud	Fraudulent agents' simulation transactions. In this dataset, agents steal customers' accounts and try to empty them by transferring funds to another account and cashing out.
isFlaggedFraud	Controlling huge account transfers and flagging unlawful efforts is the business model. Transferring more than 200,000 in one transaction is forbidden in this dataset.

MODULES

1.Data collection:

The simulator PaySim created a synthetic dataset. PaySim creates a synthetic dataset from aggregated data from the private dataset and injects fraudulent behaviors to test fraud detection techniques.

2.Tools used:

Python was used as the major programming language on this project, which was carried out at Google Colab using the PySpark framework. Some supplementary libraries, such findspark, were used to set up the Python Spark environment.

3.Data preprocessing:

The dataset underwent preprocessing steps to handle missing values. The presence of missing values in each column was evaluated, and null or NaN-containing rows were removed to ensure data integrity. Additionally, the dataset was filtered to separate records related to fraud (isFraud = 1) and non-fraud (isFraud = 0) transactions.

4.Feature extraction:

Relevant features were extracted from the dataset, focusing on specific columns such as 'type' and 'isFraud'. StringIndexer was used to convert categorical data into numerical representations for analysis.

MODULES

5.Visualizations:

Visualizations were generated to gain insights into the dataset using PySpark's capabilities. These visualizations helped explore the distribution of fraud instances and provide a better understanding of the data.

6.Model planning :

A LinearSVC (Support Vector Classification) model was created to classify fraud transactions. The model was trained using the training data and hyperparameter tuning was performed using TrainValidationSplit with a parameter

7.Model training:

The dataset was split into training and testing sets based on the 'label' column. The training set consisted of 80% of the data, and the testing set comprised the remaining 20%.

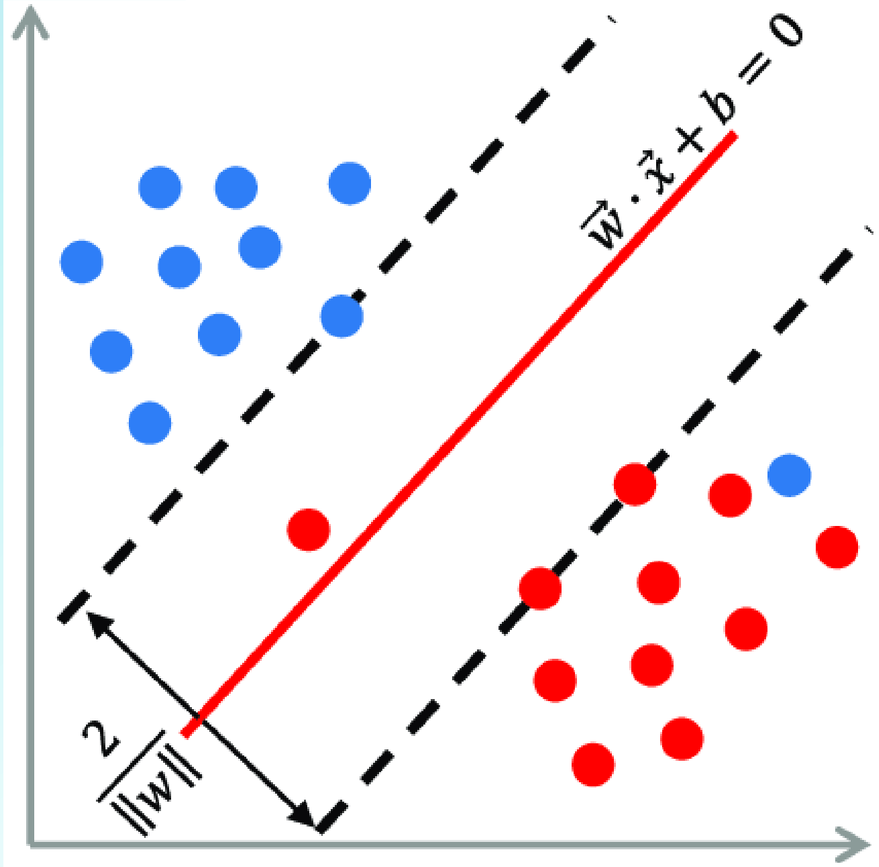
8.Model Evaluation:

The trained model was applied to the testing data to make predictions. The accuracy of the predictions was evaluated using a BinaryClassificationEvaluator, and the accuracy percentage was printed as output.

ALGORITHM USED

The model used in this project for fraud detection is LinearSVC, which stands for Linear Support Vector Classification. LinearSVC is a linear classifier and a variant of the Support Vector Machines (SVM) algorithm specifically designed for classification tasks.

LinearSVC works by finding a hyperplane that best separates the two classes in the dataset, in this case, fraudulent and non-fraudulent transactions. The goal is to maximize the margin between the classes while minimizing the classification errors. The hyperplane represents a decision boundary that separates the two classes in the feature space.



REVIEW OF STATUS

- The code sets up PySpark, a distributed data processing framework, by installing necessary packages and initializing SparkContext and SQLContext. It imports financial transaction data from a CSV file into a DataFrame and conducts basic data exploration, counting rows and examining data types. The code groups data based on the 'isFraud' column to analyze fraud occurrences, identifying 'TRANSFER' and 'CASH_OUT' types as more likely to be associated with fraud.

- For data cleaning, the DataFrame is filtered to retain relevant transaction types, and zero values in destination account columns are replaced with -1 for meaningful representation. The 'type' column is transformed into a numeric format for machine learning compatibility.

REVIEW OF STATUS

- . Feature engineering creates 'errorbalanceOrig' and 'errorbalanceDest' columns to capture balance value deviations. Data types are adjusted for further processing.

- The code splits data into training and test sets, utilizing Linear Support Vector Classification (LinearSVC) to classify transactions as fraudulent or not. Hyperparameter tuning optimizes model performance, and the model is evaluated with 92% accuracy on test data. Additional evaluation metrics like precision and recall are suggested for a comprehensive assessment. Overall, this code presents a promising approach to financial fraud detection using PySpark, with potential for further analysis and improvements.

ACCURACY

- Here we are implementing the Linear Support Vector Classification model as it will classify the whether it is fraud or not with the values as 0 or 1.
- Also creatin a parameter to hypertune it and creating a evaluator for validation.
- The accuracy of our model is 92 percent, shows that our model has been performed well in this Fraud Detection Checkpoint.

```
Accuracy: 93.76127685199252
```

GITHUB REPO AND COLAB FILE

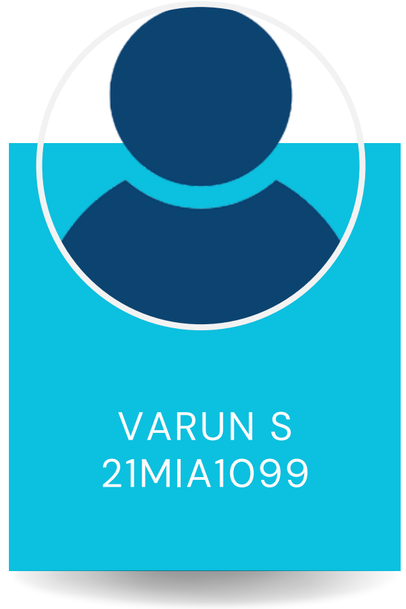
https://github.com/VarunS05/pyspark_implementation

https://colab.research.google.com/drive/13prj5KmMARb0E9_nmqr3ExtZvXsNDQJ8?usp=sharing

TEAM MEMBERS

HARISH A S
21MIA1021

THAYUMANAVAN T
21MIA1007



VARUN S
21MIA1099

Submitted to

Dr. Priyadarshini R

VIT SCOPE School

Vellore Institute of Technology, Chennai.

THANK YOU