

Dataathon '24

Team:CodeBytes

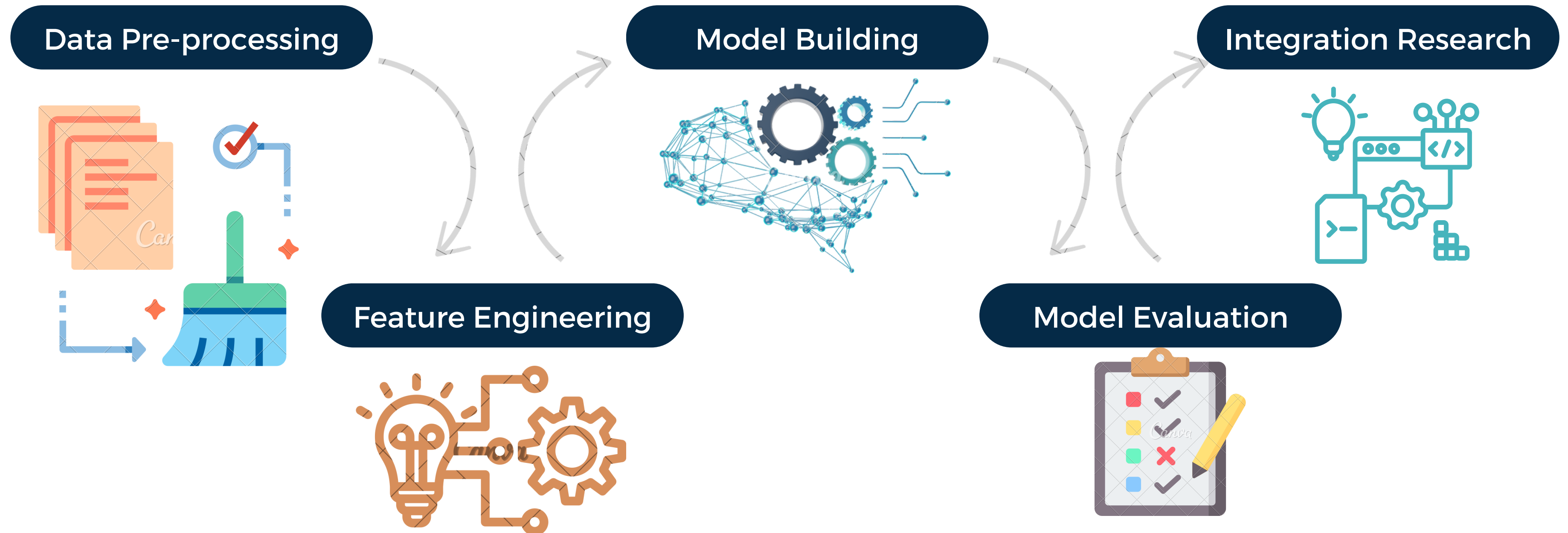
Problem Statement 1 - Fraud Detection

TY IT
CCOEW
Gauri Joshi
Kritika Dubey
Avani Ausekar
Akanksha Kale

Objective

- 01 Performing EDA on the dataset, gaining insights on trends, & finding patterns.
- 02 Pre-processing data, and performing feature engineering.
- 03 Developing an ensemble of ML models with high accuracy
Evaluating performance features using metrics such as precision, recall, F1 score, and AUC-ROC
- 04 Providing insights, and recommendations on integration of the fraud detection system into the Bank's existing infrastructure and process
- 05

Our Approach



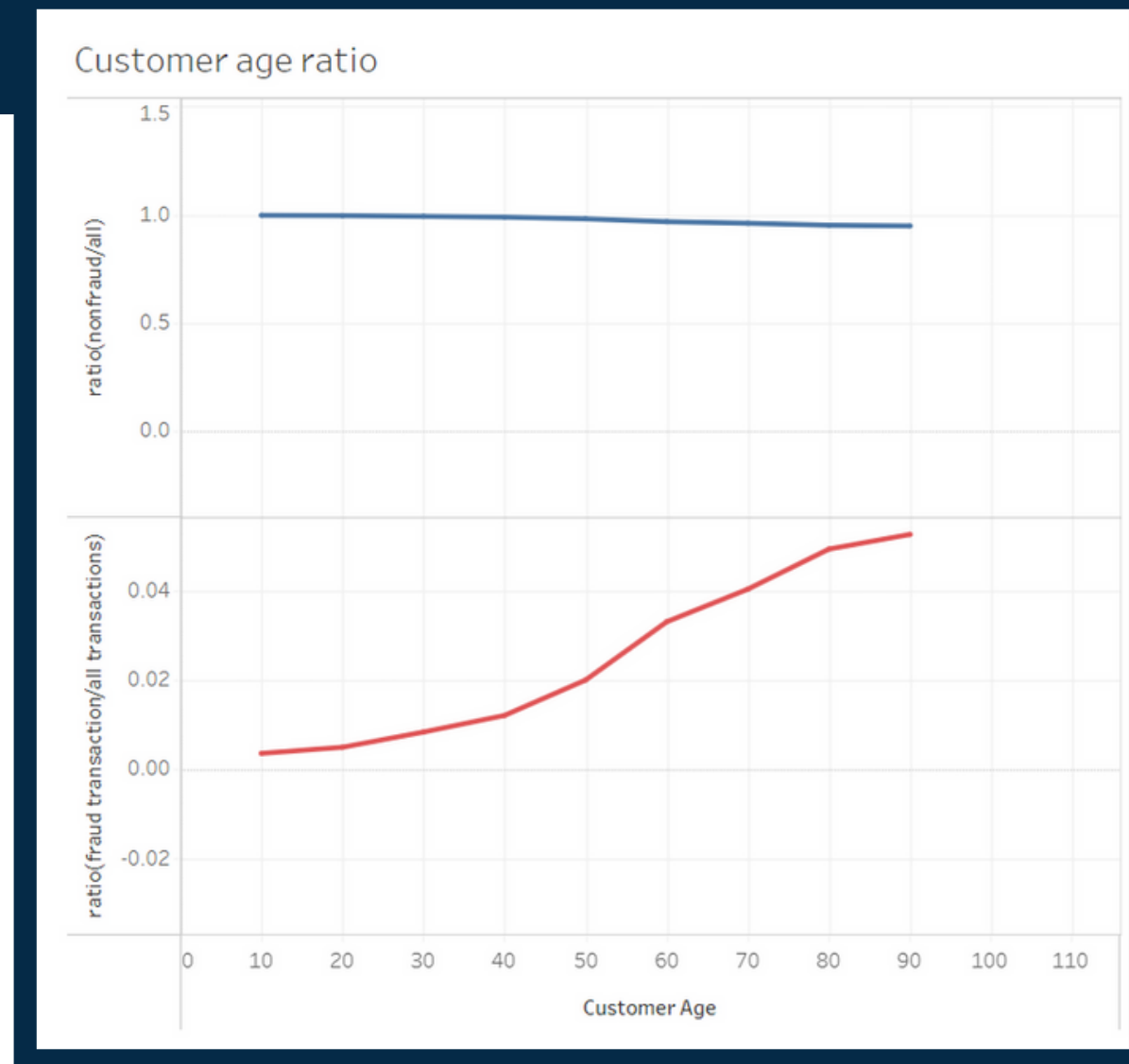
Data Pre-Processing

- Original dataset contained 32 columns
- Validated the range and inputs for all attributes.
- Data trends observation done using **Data Visualisation**
- Found **3 almost unrelated columns** to fraudulent transactions
- Total **29 columns** considered for processing by the model
- Missing values handled using **MODE**
- **MODE** used for handling missing values because as there are **high variations in banking data** **MODE** is more suitable
- **OneHotEncoder** used for encoding textual data into numeric data

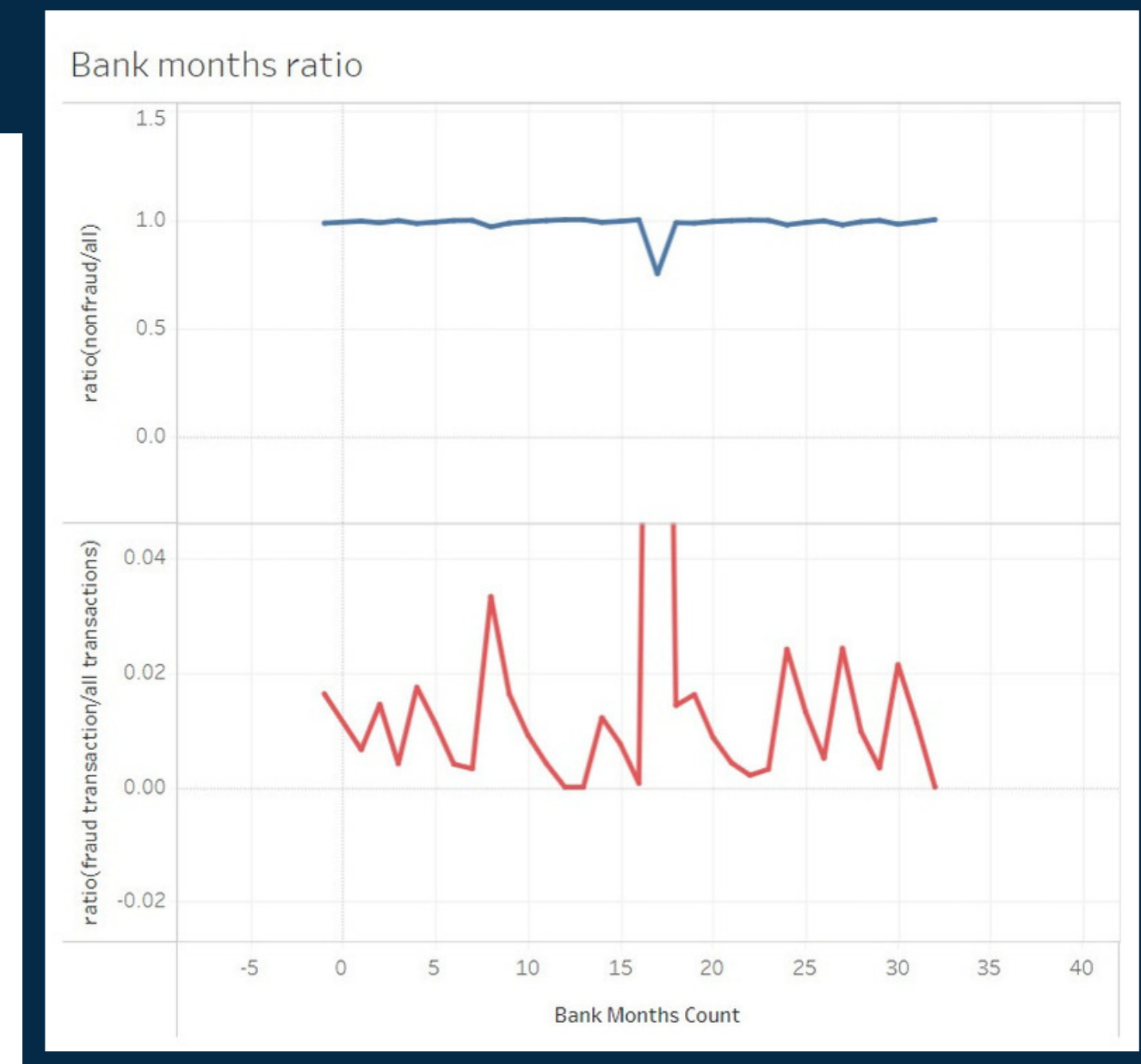
Feature Selection

We utilized scatter plots, kernel density estimation (KDE) plots, and other graphical tools on Tableau platform to explore factors associated with fraud.

For Example:



The fraudulent ratio increases with an increase in age. This suggests that age is a relevant factor and attackers have a tendency to target older customers.



A surge in fraudulent transactions occurred between 15 and 20 months, suggesting potential targeting by attackers during this timeframe.



Feature Selection

We initially attempted to perform visualizations using a simple count of fraud transactions but this produced visualizations that were difficult to interpret and evaluate. So we instead used a **ratio of fraud/total transactions** for each attribute to help distinguish between an increase in the number of fraud transactions for a category to just an increase in the number of transactions.

Fraud ratio = Fraudulent Transactions of category 'X' in feature 'A' / All Transactions containing category 'X'

Some of Our Insights:

- **Income:** Higher income correlates with **increased** fraudulent transaction ratios.
- **Device OS:** Fraud ratio highest with **Windows OS**
- **Housing Status:** Category BA exhibits the highest fraudulent ratio.
- **Employment Status:** Observed higher fraud for CG and CA. **Highest for CC.**
- **Payment Type:** The fraud ratio **higher** for **A B** and **AD** and **peaks** at payment type **AC.**

- 
- **Payment Type:** The fraud ratio **higher** for **A B** and **AD** and **peaks** at payment type **AC**.
 - **Bank Months Count:** A spike in fraudulent transactions between 15-20 months indicates potential targeting during this period.
 - **Customer Age:** The fraudulent transaction ratio **increases** with **age**.
 - **Credit Risk Score:** Fraud ratio **escalates** with **higher credit risk scores**.
 - **Device Distinct Emails:** A higher fraud ratio was observed for accounts with **2 distinct device emails**.
 - **Foreign Request:** **Higher** likelihood of fraud in **foreign requests**.
 - **Has Other Cards:** **A higher fraud** ratio is observed **when the user lacks other cards**.
 - Visualization was done for all other features, however, they didn't show much variability.
 - Due to their minimal variation compared to other features, "email_is_free", "month", and "device_fraud_count" were deemed less informative for fraud detection and were consequently omitted from the model.
- 

Model Building

Research Done

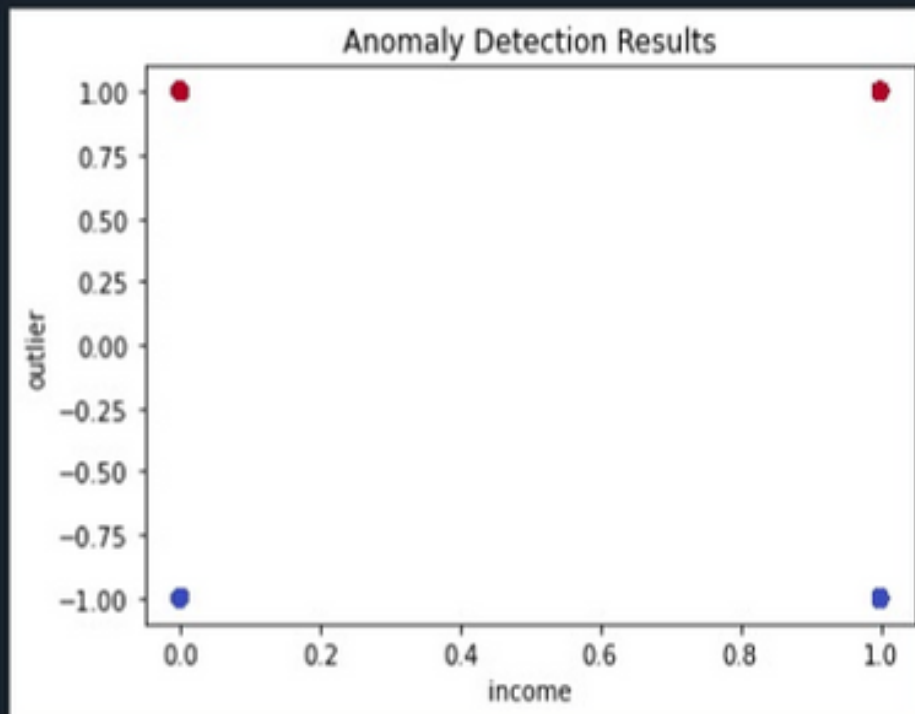
Researched about classification and anomaly detection algorithms that can be used for highly imbalanced dataset.

Models Tried

- LightGBM
- XGBoost
- Neural Network
- Bagging Classifier
- IsolationForest
- Random Forest (Too Slow thus dropped)
- KNN (Too slow thus dropped)

Model Evaluation

```
In [51]: runfile('C:/Users/avani/Desktop/Datathon,
Desktop/Datathon')
Accuracy: 0.892085
Number of outliers: 100000
AUC-ROC Score: 0.5208162172949278
Precision: 0.01557
Recall: 0.14117327046876416
F1 Score: 0.028046726530906337
```



Isolation Forest

```
In [42]: runfile('C:/Users/avani/Des
Datathon')
6250/6250 [=====]
AUC-ROC Score: 0.8684928694589554
Precision: 0.6666666666666666
Recall: 0.0009095043201455207
F1 Score: 0.0018165304268846503
Accuracy: 0.98901
```

Neural Network

```
PS D:\datathon> & C:/Python312/python.ex
Accuracy: 0.988905
Precision: 0.3717948717948718
Recall: 0.01318781264211005
F1 Score: 0.025472112428634164
AUC-ROC Score: 0.6887925464447034
PS D:\datathon> 
```

Bagging Classifier

Model Evaluation (continued)

```
In [47]: runfile('C:/Users/avani/Desktop/Datathon/lgb.py', wdir='C:/Users/avani/Desktop/Datathon')
Evaluation Metrics:
Accuracy: 0.9890
Precision: 0.2095
Recall: 0.0029
F1 Score: 0.0057
ROC AUC Score: 0.5014
Confusion Matrix:
[[692256    83]
 [ 7639    22]]
```

XG Boost

```
In [50]: runfile('C:/Users/avani/Desktop/Datathon/lgb.py', wdir='C:/Users/avani/Desktop/Datathon')
[LightGBM] [Info] Number of positive: 9898, number of negative: 890102
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.049321
seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 3202
[LightGBM] [Info] Number of data points in the train set: 900000, number of used features: 28
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.010998 -> initscore=-4.499003
[LightGBM] [Info] Start training from score -4.499003
LightGBM Accuracy: 0.98872
AUC-ROC Score: 0.5091927900845029
Precision: 0.5384615384615384
Recall: 0.01856763925729443
F1 Score: 0.035897435897435895
```

Light Gradient Boosting
Machine

Model Evaluation (continued)

Final Model Chosen- **Neural Networks** for Classification of a transaction as fraudulent or non-fraudulent due to high AUC-ROC score.

Other Model that can be preferred- **LightGBM** due to it's comparatively better performance metrics.

Tech Stack

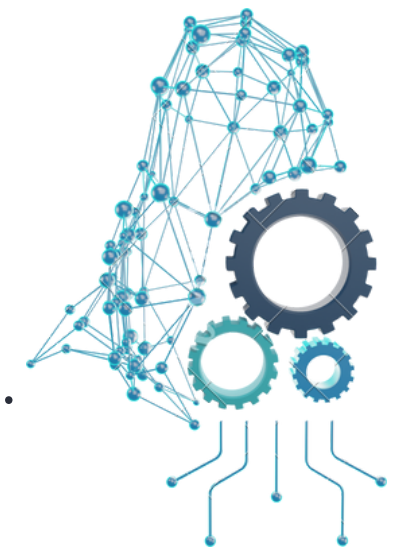
- Python - TensorFlow, Scikit-learn (or Sklearn), Pandas, NumPy, Flask, Matplotlib Data
- Visualization - Tableau, Power BI
- UI - HTML, CSS, JavaScript



DEMO Video

<https://drive.google.com/file/d/12ZA7JKyLVdiBqiJhCFXfJbgenJnK2I2P/view?usp=sharing>

Future Scope of our model



Continuous and Self-Learning Models:

- Real-Time Adaptation: Models adjust in real-time to new data, ensuring effectiveness against evolving fraud.
- Dynamic Updates: Automatic parameter updates based on incoming data streams.
- Anomaly Detection: Identify anomalies in data distributions for early fraud detection.
- Scalability: Handle large data volumes efficiently.
- Adaptive Thresholds: Adjust detection thresholds based on changing data.

Big Data Integration: Incorporate big data tools like Hadoop and Spark for efficient handling of time processing.

User Behavior Analysis:


- Pattern Recognition: Identify deviations from normal behavior.
- Profile Creation: Build user profiles for anomaly detection.
- Contextual Understanding: Consider transaction history and context for accurate detection.
- Risk Scoring: Assign risk scores to prioritize interventions.
- Behavioral Biometrics: Utilize unique user characteristics for authentication.
- Segmentation Analysis: Tailor detection strategies to different user groups.

Deep Learning Architectures:

- Feature Extraction: Automatically learn relevant features from raw data.
- Temporal Dependencies: Capture complex fraud patterns over time.
- Spatial Patterns: Detect spatial relationships in structured data.



Recommendations to integrate model in existing Banking infrastructure

1. **API Integration:** Develop an API for seamless integration of the fraud detection model with the bank's existing systems, allowing real-time fraud detection during transactions.
 2. **Alert Mechanism:** Implement an alert mechanism to notify relevant stakeholders, such as risk managers or fraud analysts, in real-time when fraudulent activity is detected.
 3. **Dashboard Integration:** Integrate the fraud detection results into existing dashboards used by bank employees for monitoring and decision-making, providing easy access to insights and trends.
 4. **Model Monitoring:** Set up continuous monitoring of the deployed model's performance metrics and retrain the model periodically to adapt to changing fraud patterns.
 5. **Compliance Considerations:** Ensure that the deployed model complies with regulatory requirements such as GDPR and financial regulations governing data privacy and security.
 6. **User Training:** Provide training sessions for bank staff involved in fraud detection and prevention to familiarize them with the capabilities and limitations of the new model.
 7. **Feedback Loop:** Establish a feedback loop to collect input from users and stakeholders, allowing continuous improvement of the model's accuracy and effectiveness.
 8. **Evaluation and Optimization:** Regularly evaluate the performance of the integrated model and optimize its parameters based on feedback and evolving fraud patterns.
- 



Thank You!