**Blockchain Transactions Model Report**

**1. Data Cleaning and Preparation:**

1. Initial Data Shape: (500, 14)
2. Features: ['blockNumber', 'from', 'gasLimit', 'gasPrice', 'hash', 'input', 'maxFeePerGas', 'maxPriorityFeePerGas', 'nonce', 'r', 's', 'to', 'v', 'value']
3. Data Types: 7 integer columns, 7 object columns
4. Non-Numeric Features: ['from', 'hash', 'input', 'r', 's', 'to', 'value']

**2. Feature Engineering:**

1. Function Signature Extraction: Derived from 'input' column, converted to integer.
2. Contract Interaction Flag: Added a binary feature indicating contract interaction.
3. Input Length: Captured length of 'input' data.
4. Hex String Handling: Limited length to avoid overflow and converted to integers.
5. String to Numeric Conversion: Converted 'from', 'to', 'hash', 'r', 's', 'value' to numeric types.
6. Duplicate Removal: Dropped duplicate transactions.

**3. Correlation with blockNumber:**

1. Positive Correlation: 's' (0.0656), 'input\_length' (0.1005)
2. Negative Correlation: 'nonce' (-0.046), 'input' (-0.0683), 'gasPrice' (-0.1019)

**4. Label Encoding:**

Encoded 'blockNumber' using LabelEncoder for model compatibility.

**5. Model Training:**

1. Independent Features: ['hash', 'from', 'to', 'nonce', 'input', 'r', 's', 'v', 'gasPrice', 'maxPriorityFeePerGas', 'maxFeePerGas', 'gasLimit', 'value', 'function\_signature', 'is\_contract\_interaction', 'input\_length']
2. Dependent Feature: 'blockNumber'
3. Train-Test Split: 80% training, 20% testing (random\_state=42)
4. Standardization: Applied StandardScaler to features.

**6. Model Performance Metrics:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Training Time (s)** |
| **Random Forest Classifier** | 0.36 | 0.402 | 0.36 | 0.373 | 3.584 |
| **XGBoost Classifier** | 0.36 | 0.402 | 0.36 | 0.373 | 1.058 |
| **CatBoost Classifier** | 0.60 | 0.590 | 0.60 | 0.589 | 60.240 |
| **Light GBoost Classifier** | 0.64 | 0.633 | 0.64 | 0.629 | 5.614 |
| **LSTM** | 0.22 | 0.266 | 0.22 | 0.228 | 135.291 |
| **GCN** | 0.70 | 0.707 | 0.70 | 0.701 | 12.490 |

**7. Best Model:**

**Graph Convolutional Network (GCN)** achieved the highest performance with 70% accuracy, making it the optimal choice for blockNumber prediction.

**8. Model Saving:**

1. Trained Model: Saved as 'final\_gcn\_model.pth'
2. Label Encoder: Saved as 'label\_encoder.pkl'

**9. Steps to Run the Process:**

1. Execute "blockchain\_transactions\_data\_analysis.ipynb" for data analysis with raw data "TxnPool\_Dataset" to produce processed CSV output.
2. Run "blockchain\_transactions\_ModelTraining.ipynb" for training models using the processed data.
3. Use "blockchain\_transactions\_ModelPrediction.ipynb" to predict blockNumber for pending transactions using 'final\_gcn\_model.pth' and 'label\_encoder.pkl'.

**Conclusion**:

The process outlined ensures robust data preprocessing, feature engineering, and model selection, leading to a reliable prediction model for blockchain transaction analysis. The GCN model provides superior performance, demonstrating its ability to capture complex relationships within the transaction data.