Big Data Project

-Miguel Cordeiro 88043
-Pietro Tellarini 114167
-Otávio Martins

Introduction for the dataset:

The rapid proliferation of Internet of Things (IoT) devices has led to a significant increase in network traffic, creating both opportunities and challenges for network management and security. The NF-ToN-IoT-v2 dataset provides a rich source of data for analyzing IoT network traffic, with features extracted from various types of IoT devices. This project aims to leverage big data technologies and machine learning algorithms to analyze, model, and enhance the understanding of IoT network traffic patterns.

Dataset: https://huggingface.co/datasets/Nora9029/NF-ToN-IoT-v2/tree/main

Decisions made:

Data Preprocessing and Exploration:

Spark Session Initialization:
Utilizing Apache Spark for efficient large-scale data processing.

Data Loading and Inspection:
Loading the NF-ToN-IoT-v2 dataset, inspecting the initial rows and schema to understand the data structure.

Feature Engineering:

Encoding Categorical Variables:

IP Address Encoding: Applying feature hashing for source subnets (IPV4_SRC_ADDR) and one-hot encoding for destination subnets (IPV4_DST_ADDR).

Target Variable Encoding:

Encoding the categorical Attack variable to numeric format for classification tasks.

Combining and Scaling Numerical Features: Using VectorAssembler and StandardScaler to create and normalize feature vectors for model training.

Model Training and Evaluation:

Model Selection:

Training machine learning models (e.g., RandomForestClassifier) to classify network traffic and detect anomalies.

Performance Metrics:

Evaluating models using accuracy, precision, recall, and F1-score to ensure robust performance.

Big Data Processing Pipeline:

Pipeline Implementation: Creating a data processing pipeline in Spark to streamline preprocessing steps and model training.

Data Visualization:

Histograms:

Plotting histograms for all features using Pandas and Matplotlib to understand the distribution of each feature.

Correlation Heatmap:

Creating a heatmap with Seaborn to visualize correlations between features, helping to identify key relationships and dependencies.

Categorical Data Analysis:

Frequency Analysis:

Calculating and displaying the frequency of categories for important variables such as Attack, IPV4_SRC_ADDR_Subnet, and IPV4_DST_ADDR_Subnet.

Dimensionality Reduction:

Principal Component Analysis (PCA):

Implementing PCA to reduce the dimensionality of selected numerical features (IN_BYTES, OUT_BYTES, IN_PKTS, OUT_PKTS), aiding in data visualization and understanding variance.

Model Selection:

Random Forest Classifier:

Chosen for its robustness and ability to handle large datasets with many features.

Logistic Regression:

Selected for its simplicity and interpretability.

Multilayer Perceptron Classifier (MLP):

Implemented to leverage neural networks for potentially higher accuracy.

Model Training:

Training Models:

Fitting the Random Forest, Logistic Regression, and Multilayer Perceptron models to the assembled dataset.

Layer Configuration for MLP:

Defining the input, hidden, and output layers for the MLP based on the feature count and number of distinct classes.

Model Evaluation:

Accuracy Evaluation:

Using MulticlassClassificationEvaluator to assess the accuracy of each trained model on the dataset.

Comparison of Results:

Comparing the accuracy of the Random Forest, Logistic Regression, and Multilayer Perceptron models to determine the most effective approach.

Experiences and tests made

Execution and Resource Utilization:

Execution Times:

Data Loading: Loading the Parquet dataset took approximately 1 min, leveraging Spark's parallel data reading capabilities.

Feature Engineering: Processing features, including encoding and scaling, was completed in about 3 minutes.

Model Training:

RandomForest: Training completed in approximately 2 minutes.
LogisticRegression: Training completed in 3 minutes.
MultilayerPerceptron: Training took around 4 minutes due to the model's complexity.

(We were having some issues when running on other machines, connection refused by pyspark, we were not able to solve this problem in the ML and the PCA part, so we reduced the dataset when working with this)

Results Analysis and Conclusions

Performance Analysis

- RandomForest Classifier:
- Accuracy: 88%
- Precision: 87%
- Recall: 88%
- F1-score: 87%
- Conclusion: The RandomForest model performed robustly with high accuracy and balanced precision and recall, making it suitable for detecting a wide range of attacks without missing many anomalies. However, this result was obtained by reducing the dataset by approximately 90% due to connection issues, which might affect the generalizability of the model.
- Logistic Regression:

- **Accuracy**: 70%
- **Precision**: 68%
- **Recall**: 70%
- **F1-score**: 67%
- **Conclusion**: Logistic Regression, while simpler, provided decent performance and can be used for initial anomaly detection tasks. Its interpretability makes it valuable for understanding feature importance. This model might not be robust enough for complex attack detection but serves as a good baseline.
- **Multilayer Perceptron (MLP)**:
- **Accuracy**: 82%
- **Precision**: 77%
- **Recall**: 82%
- **F1-score**: 79%
- **Conclusion**: The MLP model showed higher accuracy and F1-score compared to Logistic Regression, indicating its potential in capturing complex patterns in the data. However, the longer training time and higher computational resources required must be considered. This model can be effective for more nuanced attack detection but requires careful resource management.

Problem Perspective

The formulated problem focuses on enhancing network security by accurately classifying and detecting anomalies in IoT network traffic. The results indicate:

1. **Effectiveness of Machine Learning Models**:
    - The RandomForest model demonstrated robust performance, making it suitable for a wide range of attack detections with balanced sensitivity and specificity.
    - Logistic Regression, while less complex, provided a baseline performance that is easier to interpret and quicker to implement.
    - The MLP model, despite requiring more resources and time, showed promise in handling more complex patterns within the data, useful for detecting sophisticated attacks.
2. **Importance of Feature Engineering**:
    - The preprocessing steps, including feature encoding and scaling, significantly contributed to the models' performance, highlighting the need for thorough feature engineering in similar projects.
3. **Scalability and Efficiency**:
    - The use of Apache Spark for distributed processing proved essential in handling large-scale data efficiently, suggesting that similar big data frameworks should be employed in real-world applications to manage and process extensive network traffic data.

Conclusions

The project successfully applied big data analysis and machine learning techniques to the NF-ToN-IoT-v2 dataset, providing valuable insights and effective models for network traffic classification and anomaly detection. The results suggest that implementing such models in IoT networks can significantly enhance security by enabling real-time monitoring and rapid detection of malicious activities. Future work could focus on optimizing model deployment and integrating these solutions into existing network management systems for continuous, real-time protection. Additionally, addressing the dataset reduction due to connection issues and ensuring the robustness of the models on the full dataset will be crucial for real-world applicability.