UMBC

CYBER SECURITY THREAT DETECTION USING BIG DATA

TEAM MEMBERS:
MITAALI PATEL
MRUNAL JADHAV
SHRANYA GANDHAM
JAYANTH RACHURI

INTRODUCTION

- **Cybersecurity is crucial** in the age of IoT and connected devices.
- **Threats are rapidly evolving** and increasingly sophisticated.
- **Traditional security systems** struggle to handle massive volumes of network data.
- This project **leverages Big Data platforms** for real-time threat analysis.
- **Machine learning models** are integrated to enhance detection accuracy.
- **Scalable data processing tools** improve system performance and efficiency.

PROJECT OBJECTIVES

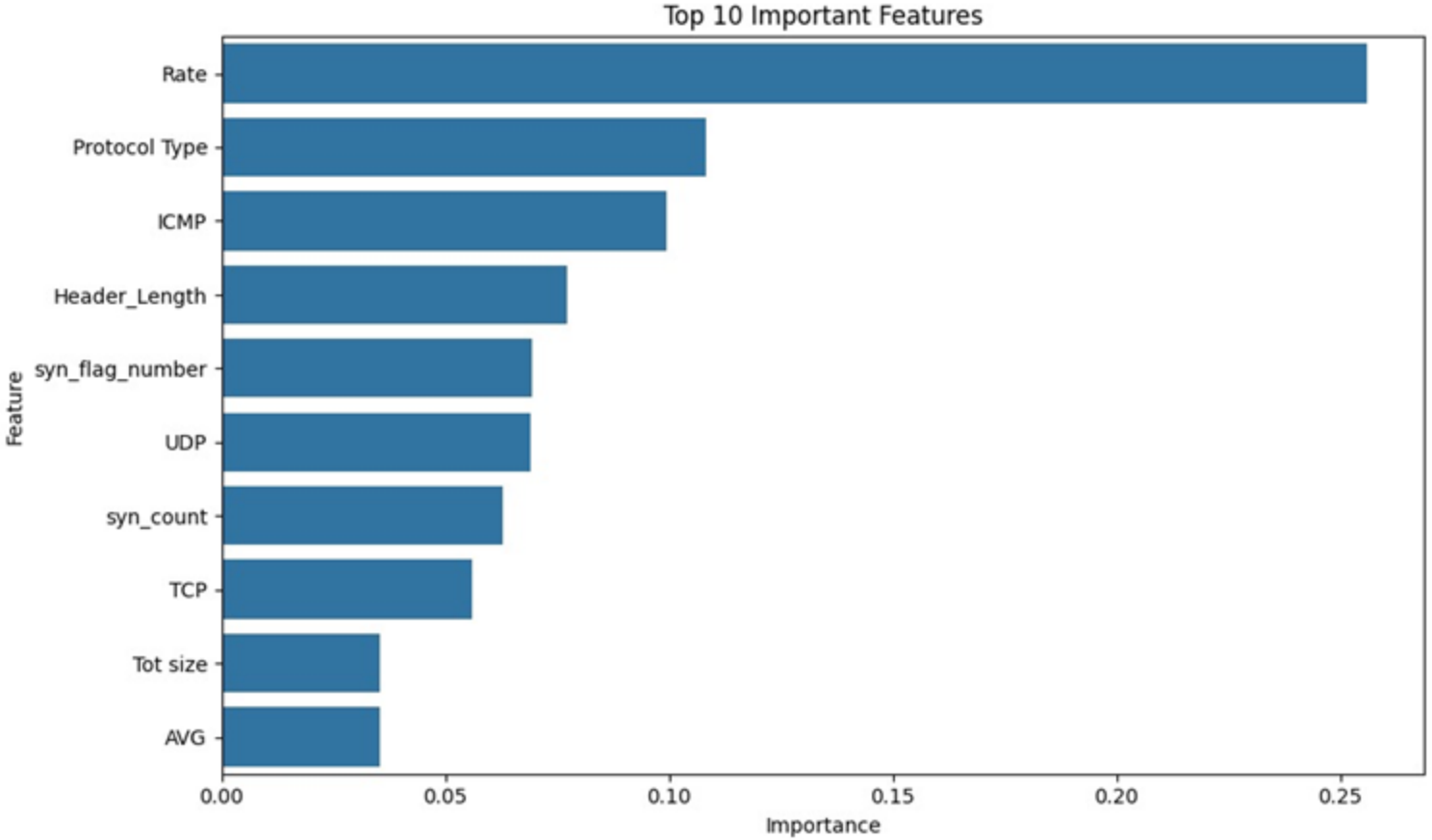
- To detect cybersecurity threats using Big Data techniques.
- Utilized PySpark MLlib for scalable machine learning on large datasets.
- Classified network traffic as normal or attack types using XGBoost and other models.

DATA PREPROCESSING

- Dropped missing and infinite values.
- Encoded categorical variables using LabelEncoder.
- Standardized numerical features using StandardScaler.
- Removed outliers using Z-score thresholding.
- Dropped ~131,000 duplicate records.

FEATURE ENGINEERING

- Used VarianceThreshold to remove low-variance features.
- Selected top 21 features for modeling.
- RandomForestClassifier used to assess feature importance.


Top 10 Important Features
Rate
Protocol Type
ICMP
Header_Length
syn_flag_number
UDP
syn_count
TCP
Tot size
AVG
Feature
0.00
0.05
0.10
0.15
0.20
0.25
Importance

TOOLS AND TECHNOLOGIES

- Processing: Apache Spark for large-scale data processing, PySpark for transformations.
- Modeling: XGBoost, Random Forest, Logistic regression

DATASET OVERVIEW

- Original Dataset: ~712,000 rows, 40 features.
- Data collected from IoT network traffic, including protocol, flags, packet sizes.
- Labelled as normal or various types of attacks.

Modeling with PySpark MLlib

- Applied XGBoost for multi-class classification.
- Achieved ~80% accuracy on test data.
- Used PySpark MLlib for scalable feature processing and training.

STRATEGIC RECOMMENDATIONS

- Implement behavior-based intrusion detection alongside signature-based systems.
- Use AI-enhanced monitoring tools that continuously learn from new threats.
- Automate incident response pipelines for faster mitigation.
- Invest in edge-computing models for decentralized security checks on IoT endpoints.

FUTURE OPPORTUNITIES

- Integration of federated learning to preserve data privacy while training global models.
- Adoption of edge analytics for faster detection at the device level.
- Use of blockchain for secure device identity and access control.
- Emerging AI models like Transformers for sequence-based intrusion analysis.

CHALLENGES

- **Feature Dominance:** Some features overpower others, biasing the model.
- **Low-Variance Removal Ineffective:** Doesn’t improve performance.
- **High-Variance Noise:** Some high-variance features reduce model quality.
- **Class Imbalance:** Requires SMOTE to balance data.
- **Feature Dependence:** Model accuracy heavily tied to selected features.
- **Too Many Key Features:** Makes selection and interpretation hard.
- **High Computational Load:** More features = longer training times.
- **Low Interpretability:** Complex models harder to explain.

REFERENCES

Apache Spark Documentation
https://spark.apache.org/docs/latest/

PySpark MLlib Guide
https://spark.apache.org/mllib/

XGBoost Documentation
https://xgboost.readthedocs.io/

Matplotlib and Seaborn Visualization Docs

https://matplotlib.org/

THANK YOU