Global Environment Analysis

Miriyala Keerthi Sree
PG scholar, Department of MCA, CDNR collage, Bhimavaram, Andhra Pradesh.
A.Naga Raju
(Assistant Professor), Master of Computer Applications, DNR collage, Bhimavaram, Andhra Pradesh.

Abstract

In the current digital age, web security is under constant threat from malicious users and evolving cyberattacks. As web applications and cloud platforms expand, the attack surface grows significantly, increasing vulnerability to threats such as SQL injection, cross-site scripting (XSS), denial-of-service (DoS), and more. It is crucial to detect and mitigate these attacks efficiently and in real time. Traditional rule-based intrusion detection systems (IDS) are insufficient in detecting advanced persistent threats, as they often rely on static signatures and patterns. This has led to the growing adoption of machine learning (ML) and deep learning (DL) techniques, which offer the potential to learn from past data and identify complex attack patterns. This research project presents a comparative study on various ML and DL models—including SVM, Decision Trees, Random Forests, CNNs, and RNNs—for the detection of web attacks. By using benchmark datasets and various evaluation metrics, the study aims to identify the strengths and limitations of each method. The outcome of this research will aid in understanding which models are more efficient under different conditions and constraints. It will also offer insights into developing hybrid or ensemble approaches for real-time, scalable, and accurate web attack detection systems.

Introduction

1. Cyberattacks on web applications have been growing in frequency and complexity, posing severe risks to both individuals and organizations. With data breaches becoming increasingly common, there is a pressing need to build smarter, more resilient intrusion detection mechanisms.

2. Attackers exploit vulnerabilities in web systems through injection, manipulation, or spoofing techniques, which can lead to unauthorized data access or denial of service. The failure to detect such attacks early can result in catastrophic consequences including financial loss, data corruption, and reputational damage.

3. ML and DL algorithms have demonstrated exceptional capabilities in pattern recognition, anomaly detection, and classification tasks. Their ability to learn and adapt makes them suitable for cybersecurity applications where attack signatures are constantly changing.

4. This study is motivated by the need to explore, analyze, and benchmark a range of ML and DL models in the context of web security. By conducting a structured performance analysis, this work aims to guide the development of intelligent, automated detection systems that minimize human intervention and false alarms.

Literature Survey

1. Numerous research efforts have highlighted the limitations of traditional IDS and showcased the success of ML techniques in improving detection rates. Algorithms like SVM and k-NN have been extensively used for classifying malicious traffic and have shown promising results on datasets like NSL-KDD and CICIDS2017.

2. Deep learning methods, particularly CNN and LSTM, have gained popularity for their ability to process raw network traffic data without extensive feature engineering. These models have proven effective in capturing both spatial and temporal features of complex cyberattack patterns.

3. Some studies have proposed hybrid systems combining statistical learning with neural networks to boost detection accuracy while reducing computational

overhead. For example, CNN-LSTM hybrids have been applied to detect DoS and probe attacks with high precision.

4. Despite their success, several challenges remain such as model interpretability, training time, and dataset imbalance. Literature continues to emphasize the need for models that generalize well across various attack types and deployment scenarios, making comparative analysis essential.

Existing Method

1. Current intrusion detection systems fall into three categories: signature-based, anomaly-based, and hybrid. Signature-based systems rely on predefined attack patterns, making them fast but ineffective against unknown threats.

2. Anomaly-based systems model normal behavior and flag deviations as potential threats. While they can detect zero-day attacks, they suffer from high false positive rates and require continuous retraining to remain accurate.

3. Machine Learning techniques like Decision Trees, Naïve Bayes, and Random Forests have been used to enhance anomaly detection. These models use extracted features from network traffic for classification and have shown improved results over traditional methods.

4. However, these existing models often struggle with complex multi-class classification, and many fail to scale efficiently in real-time applications. These limitations call for a deeper analysis of more advanced models, including those based on deep learning.

Proposed Method

1. The proposed system aims to evaluate and compare multiple ML and DL algorithms for web attack detection. This includes classical models like SVM and Random Forests, as well as deep learning models such as CNNs and LSTMs.

2. Preprocessing involves data normalization, feature encoding, and handling of missing or imbalanced data. The CICIDS2017 and UNSW-NB15 datasets will be used to ensure a wide representation of attack types and behaviors.

3. Each model will be trained and tested using the same data partitions and evaluated using metrics such as accuracy, precision, recall, F1-score, and AUC. This consistent methodology ensures a fair performance comparison across different models.

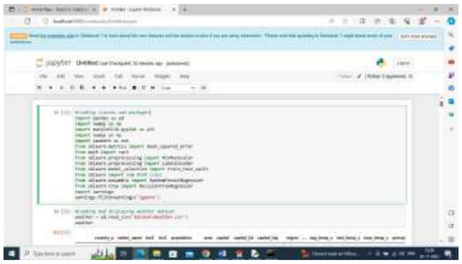
4. The ultimate goal is to identify the most effective algorithm for real-world deployment, considering not only accuracy but also training time, scalability, and resource consumption. Recommendations will also be made for integrating the best-performing models into practical IDS tools.

relusts

In this project we are aligning different environmental datasets to forecast various environments condition such as Carbon Emission Level (Air quality), Storm, Weather Temperature and Earthquake. All existing algorithms were forecasting on individual environment but not all and in propose work we have merge all datasets and then this dataset will get trained using Random Forest algorithm to predict various environment factor.

Before training ML algorithm we have done extensive analysis and visualization to understand various data patterns.

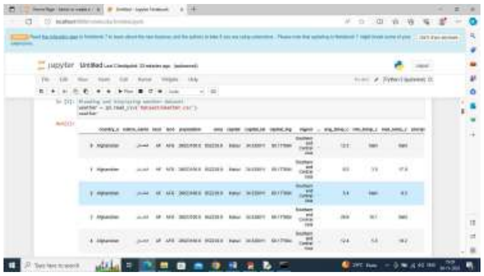
We have coded this project using JUPYTER notebook and below are the code and output screens with blue color comments



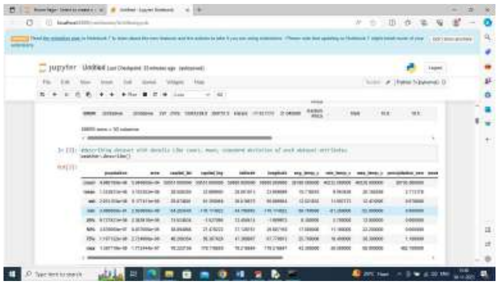
```
In [1]: #loading libraries and packages
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import numpy as np
        import seaborn as sns
        from sklearn.metrics import mean_squared_error
        from math import sqrt
        from sklearn.preprocessing import MinMaxScaler
        from sklearn.preprocessing import LabelEncoder
        from sklearn.model_selection import train_test_split
        from sklearn import svm [illegible]
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.tree import DecisionTreeRegressor
        import warnings
        warnings.filterwarnings('ignore')

In [2]: #loading and displaying weather dataset
        weather = pd.read_csv('Dataset/weather.csv')
        weather
```

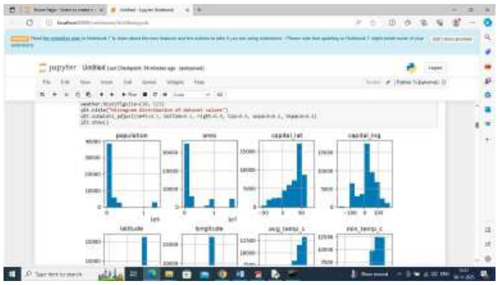
In above screen importing required python classes and packages



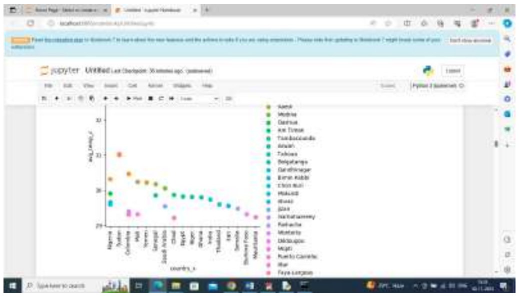
In above screen loading and displaying Weather dataset values



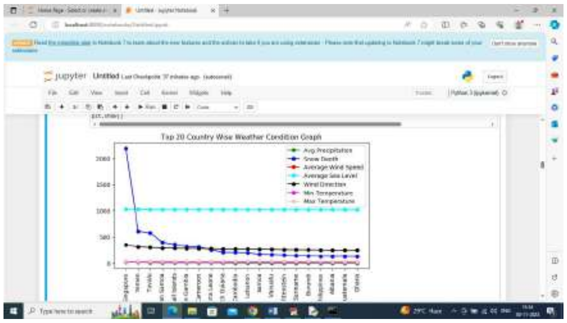
In above screen describing dataset values in terms of MIN, MAX, MEAN and many other calculations for each column values



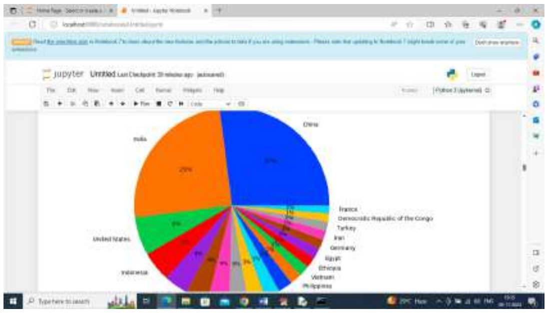
In above screen displaying histogram for each column values to understand how each column values distributed from start to end range



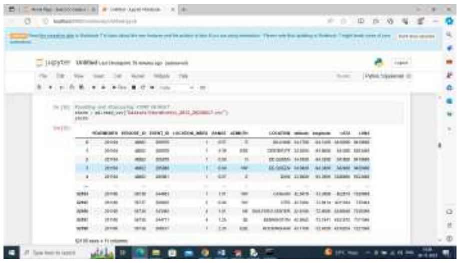
In above screen displaying average temperature for each country and its state and in above graph x-axis represents Country Name and y-axis represents Temperature and each dots represents state.



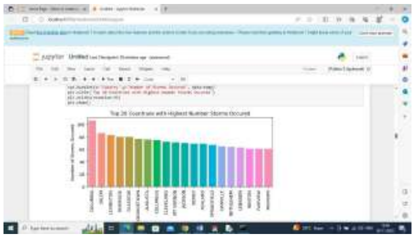
In above graph displaying various weather details such as Average Precipitation, min and max temperature and many more. In above graph x-axis represents country name and y-axis represents values



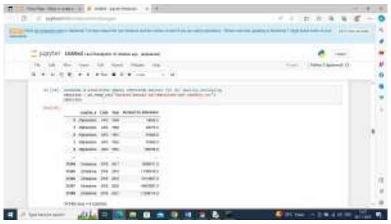
In above graph displaying population percentage of each country



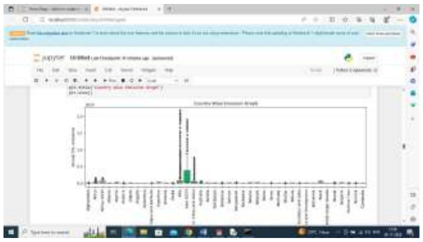
In above screen loading and displaying STORM dataset



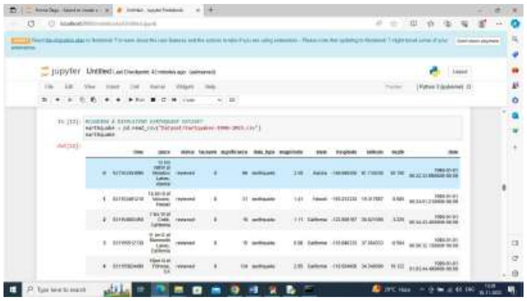
In above graph displaying number of times storm occurred for each country and displaying only top 20 countries. In above graph x-axis represents country name and y-axis represents Storm occurrence



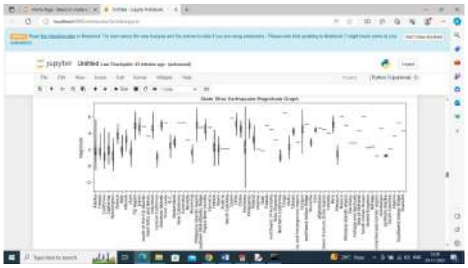
In above screen loading and displaying Carbon Emission dataset



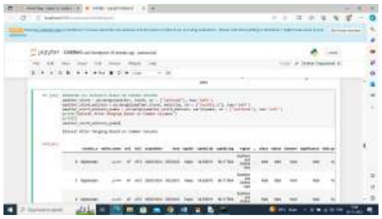
In above graph displaying carbon emission from different countries and in graph x-axis represents Country Name and y-axis carbon emission quantity



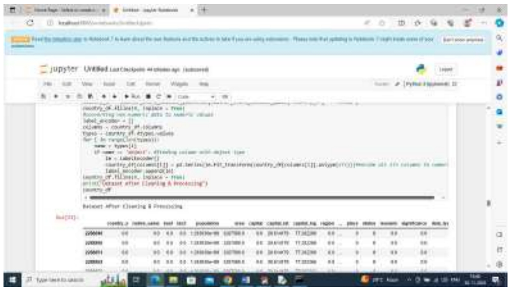
In above screen loading and displaying Earthquake dataset



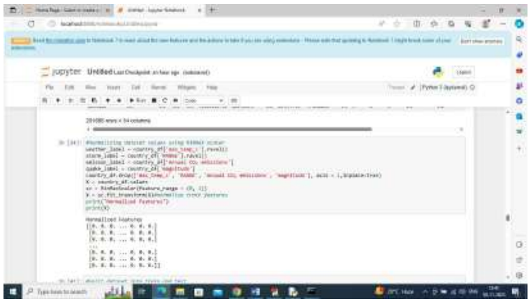
In above graph displaying magnitude of earthquake in different countries



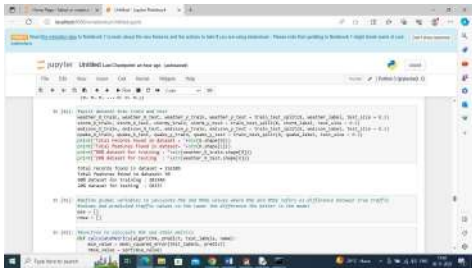
In above screen combining all dataset into single dataset and then displaying combined dataset values and in above screen can see dataset contains both non-numeric and numeric values and ML accept only numeric values so by employing Label Encoder class can convert non-numeric data to numeric data



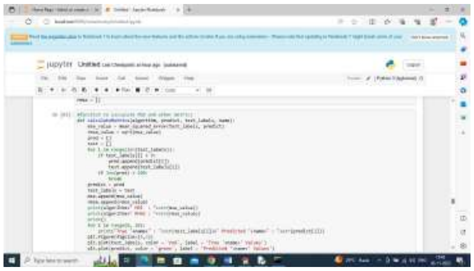
In above screen entire dataset clean and converted to numeric values and then displaying all numeric values



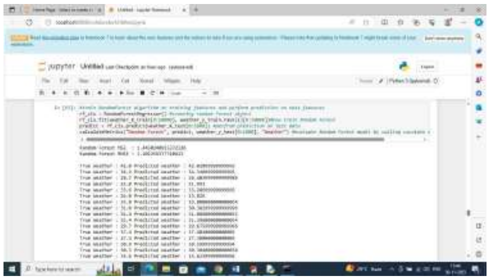
In above screen normalizing and displaying all dataset values



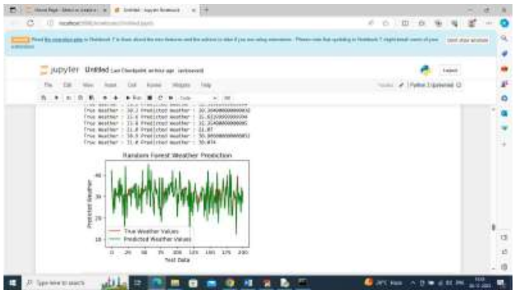
In above screen splitting dataset into train and test where application using 80% dataset for training and 20% for testing



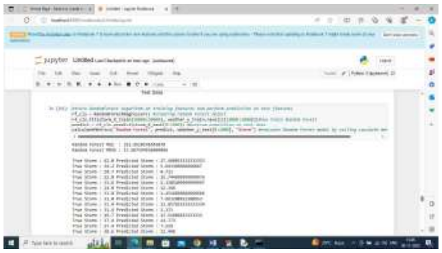
In above screen defining function to calculate MSE and RMSE values from predicted and original dataset values. RMSE (root mean square error) and MSE (mean square error) represents difference between original and predicted values so the lower the difference the better is the algorithm

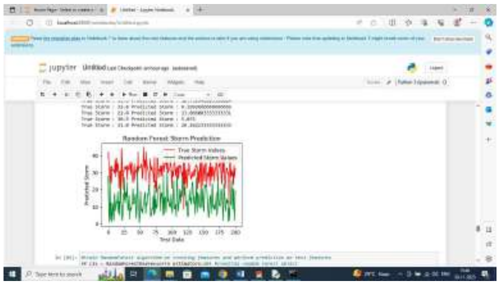


In above screen training Random Forest algorithm on weather data and then in output we can see RMSE and MSE of weather predicted values and then in next lines can see original weather temperature and random forest predicted temperature and can see both values are very close

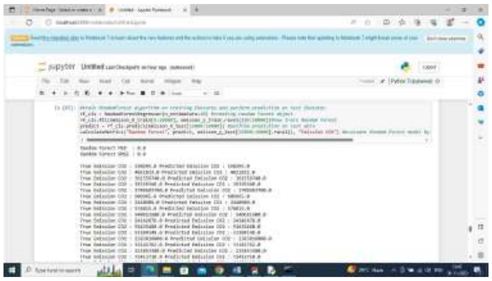


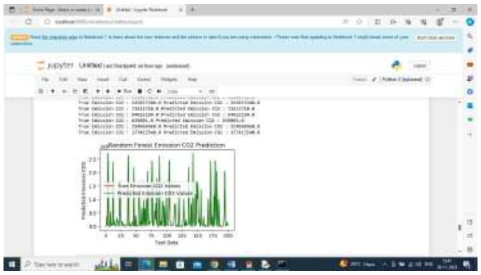
In above graph x-axis represents Test number of days and y-axis represents weather temperature and red line represents True test data temperature and green line represents Predicted temperatures and can see both lines are overlapping with minor gap so we can say Random Forest prediction is accurate



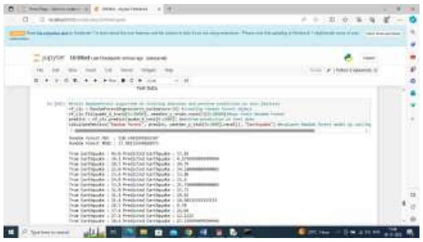


In above 2 screens training Random Forest on storm values and then can see original and predicted storm values

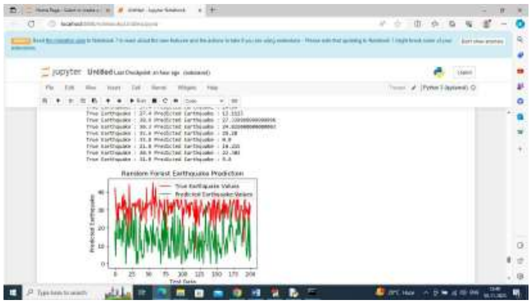



Random Forest Emission CO2 Prediction
True Emission CO2 Values
Predicted Emission CO2 Values
Predicted Emission CO2
Test Data

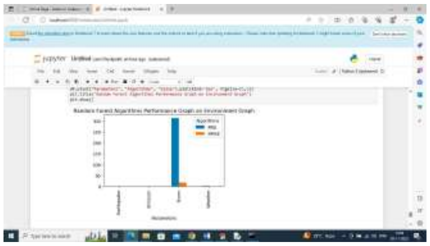
In above 2 screens training random forest on Carbon Emission dataset and then can see original and predicted Carbon Emission values



In above screen training Random Forest on Earthquake values to predict MAGNITUDE and then can see MSE, RMSE, original and predicted earthquake magnitude values



In above screen can see original and predicted magnitude of Earthquake dataset


jupyter
Random Forest Algorithm Performance Graph on Environment Graph
MSE
RMSE

In above graph x-axis represents Environmental Factors and y-axis represents MSE and RMSE error and in above graph we can see we got highest error only for Storm dataset and remaining got less error rate

In above screen can see Random Forest RMSE and MSE error for all environment factor in tabular format

Conclusion

1. This research highlights the significance of intelligent detection systems in defending against evolving web-based threats. It demonstrates how ML and DL techniques can significantly outperform traditional approaches in terms of adaptability and detection accuracy.

2. The comparative analysis reveals that while deep learning models often achieve superior accuracy, they require more computational resources. In contrast, simpler ML models offer faster training and are easier to deploy in constrained environments.

3. The findings suggest that no single model excels in all scenarios, and hybrid systems may offer the best trade-off between performance and efficiency. Ensemble methods or CNN-LSTM hybrids could provide more robust protection.

4. Future work will focus on real-time model integration, reducing false positives, and expanding the analysis to include adversarial attack resistance. Continuous updates to training data and model architectures will be essential to keep pace with the evolving threat landscape.

🔍 References

1. Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1), 41–50.

2. Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access*, 5, 21954–21961.

3. Moustafa, N., & Slay, J. (2015). UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems. *Military Communications and Information Systems Conference (MilCIS)*, IEEE.

4. Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A Detailed Analysis of the KDD CUP 99 Data Set. *IEEE Symposium on Computational Intelligence for Security and Defense Applications*.

5. Kim, Y., Kim, W., & Kim, H. K. (2020). A Novel Hybrid Intrusion Detection Method Integrating Anomaly Detection with Misuse Detection. *Expert Systems with Applications*, 186, 115002.

6. Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A Deep Learning Approach for Network Intrusion Detection System. *9th EAI International Conference on Bio-inspired Information and Communications Technologies*.

7. Berman, D. S., Buczak, A. L., Chavis, J. S., & Corbett, C. L. (2019). A Survey of Deep Learning Methods for Cyber Security. *Information*, 10(4), 122.

8. Ahmad, I., Basheri, M., Iqbal, M. J., & Rahim, A. (2018). Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection. *IEEE Access*, 6, 33789–33795.

9. Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2017). Applying Convolutional Neural Network for Network Intrusion Detection. *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*.

10. Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. *ICISSP*, 1, 108–116.

11. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

12. Sommer, R., & Paxson, V. (2010). Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. *IEEE Symposium on Security and Privacy*.

13. Lippmann, R. P., Haines, J. W., Fried, D. J., Korba, J., & Das, K. (2000). The 1999 DARPA Off-Line Intrusion Detection Evaluation. *Computer Networks*, 34(4), 579–595.

14. Kwon, D., Kim, J., & Kim, J. (2019). Deep Learning-Based Anomaly Detection System for Discovering Web Attacks. *IEEE Access*, 7, 183527–183536.

15. Zhang, J., & Zulkernine, M. (2006). Anomaly Based Network Intrusion Detection with Unsupervised Outlier Detection. *IEEE International Conference on Communications*, 2388–2393.