CAPSTONE PROJECT

NETWORK INTRUSION DETECTION

Presented By:
1. Suraj Singh Rawat-GRAPHIC ERA DEEMED TO BE UNIVERSITY-B.TECH CSE

edunet foundation

OUTLINE

- Problem Statement
- Proposed System/Solution
- System Development Approach
- Algorithm & Deployment
- Result (Output Image)
- Conclusion
- Future Scope
- References

PROBLEM STATEMENT

- The exponential growth of **networked systems**, **IoT devices**, and **digital communications** has dramatically increased the surface area for cyber threats. Attackers now use **sophisticated techniques** that easily bypass traditional security mechanisms.
- Manual monitoring is **time-consuming** and **inefficient**, while static rule-based systems are **rigid** and often **fail to adapt** to new or evolving threats.

PROPOSED SOLUTION

- Data Collection
  - Source: Kaggle "**Network Intrusion Detection**" dataset.
  - The dataset contains simulated network connection records, generated by mimicking US Air Force LAN activity under normal and attack conditions. Each connection is defined by 41 features, with labels specifying "**Normal**" or various "**Anormal**" (attack) classes.
  - Content: Features encapsulate connection metadata (e.g., duration, protocol, service), traffic statistics, and error indicators, enabling multi-dimensional analysis of network behavior.
- Data Preprocessing
  - Data Cleaning: Address missing or inconsistent values, remove duplicates, and handle outliers. Proper normalization is performed to ensure all features contribute equally to learning algorithms.
- Feature Engineering
  - Extract and encode protocol types, services, and flags.
  - Aggregate session-based statistics.
  - Use correlation analysis and dimensionality reduction methods (e.g., PCA) to identify the most relevant features, helping mitigate issues with high dimensionality and improve model efficiency.
- Handling Class Imbalance: Employ strategies such as oversampling (SMOTE), under-sampling, or class weight adjustments, as the dataset can be skewed toward normal vs. attack samples.

- **Model Selection**
  - **AutoAI Pipelines (in Watsonx):** Automatically test ML pipelines.
  - **Algorithms:** Random Forest, Decision Tree, XGBoosting.
  - **Evaluation Metrics:** Train a model to detect intrusions in real-time using IBM Watsonx.ai AutoAI pipelines
- **Deployment**
  - **Platform:** Deploy best-performing model using **Watsonx.ai Deployment Space**.
  - **Endpoint:** REST API generated for real-time intrusion prediction.
  - **Monitoring:** Use **IBM Cloud Monitoring** to track model usage and logs.

- **Evaluation**
  - **Test Interface:** Predict on new traffic data via Watsonx UI.
  - **Reporting:** Display prediction probabilities and logs.
  - **Confidence Score:** Provided by the deployed model API (e.g., 98% intrusion confidence).

SYSTEM APPROACH

- System requirements
- IBM Cloud Setup
  - **IBM Cloud Account** – Register at **cloud.ibm.com**
  - **Watsonx.ai Studio –** For building and training the model (AutoAI + Notebook).
  - **IBM Cloud Object Storage –** To upload CSV datasets and share across projects.
  - **Watson Machine Learning Runtime** – Required to run AutoAI experiments and deploy models.

- **Development Environment**
  - Browser: Chrome/Edge
  - Internet: Stable connection for IBM Cloud access
  - Python version: 3.8+ (for notebook-based extensions)
  - Jupyter Notebooks: Used inside Watsonx.ai Studio

- Library required to build the model
  - pandas, numpy, sklearn, matplotlib, seaborn
  - scikit-Learn,xgboost, imbalanced-learn
  - IBM SDKs: ibm-watson-machine-learning

ALGORITHM & DEPLOYMENT

- Algorithm Selection
  We selected **XGBoost Classifier**, a powerful ensemble method known for its high performance on structured data. It was chosen based on its ability to handle class imbalance, noisy features, and its success in classification challenges involving tabular datasets. IBM Watsonx.ai AutoAI pipelines automatically compared XGBoost against models like Random Forest, Decision Tree .

- **Data Input**
  The input features include:
  • duration – length of the network connection
  • protocol_type, service, flag – encoded network protocol details
  • Statistical features – src_bytes, dst_bytes, count, srv_count, etc.
  • Attack indicators – land, wrong_fragment, urgent

- All 41 features from the Kaggle dataset were used as inputs, with the target being class (Normal/Abnormal type).

- Training Process:

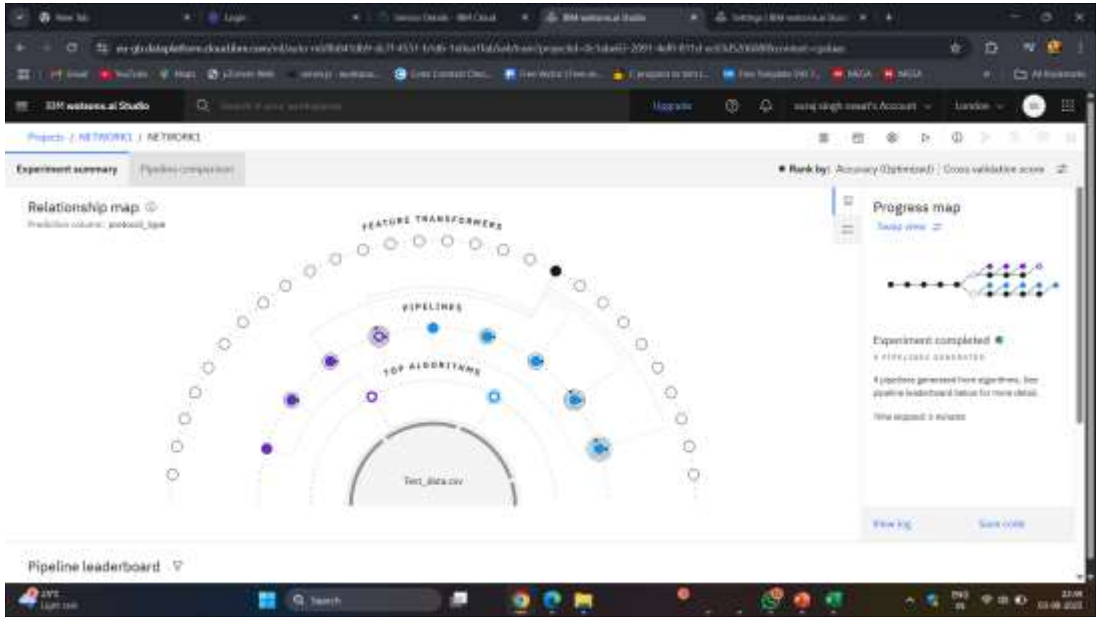
  - Dataset split into training and test sets using **AutoAI's internal CV mechanism**
    • AutoAI performed **automated preprocessing, feature selection**, and **hyperparameter tuning**
    • Evaluation metrics used: **Accuracy, Precision, Recall, F1-score**
    • XGBoost with tuned parameters emerged as the top performer

- Prediction Process:

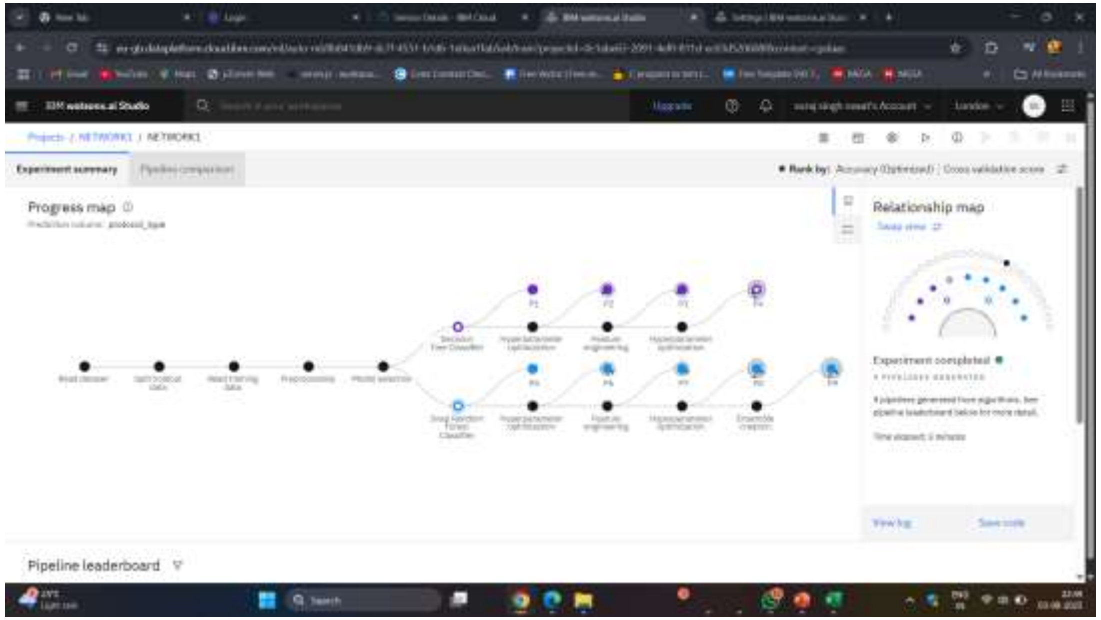
  - The trained model predicts whether a new network connection is "normal" or an "intrusion."
    • Inputs: Real-time connection attributes streamed or uploaded via UI
    • The model returns a class prediction along with a **confidence score**
    • Deployment is done via **IBM Watsonx.ai REST API**, allowing real-time detection in practical environments

RESULT

Experimental Summary



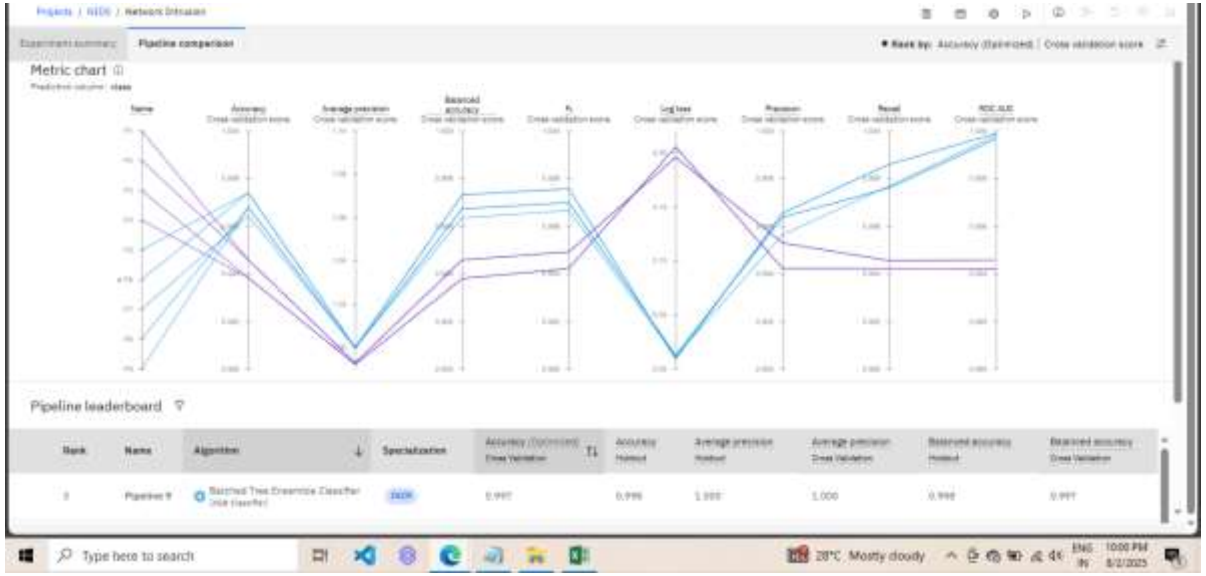
Pipeline Map



edunet foundation

RESULT (METRIC CHART)

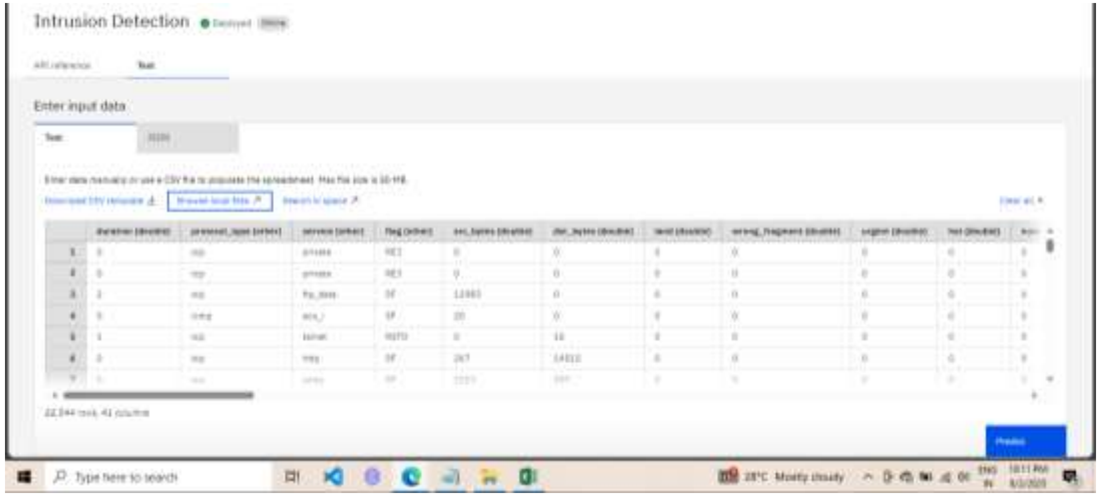
- Pipeline 6 had highest cross-validation accuracy: 0.997
- Holdout accuracy: 0.998
- Average precision, recall, and F1-score: ~1.00

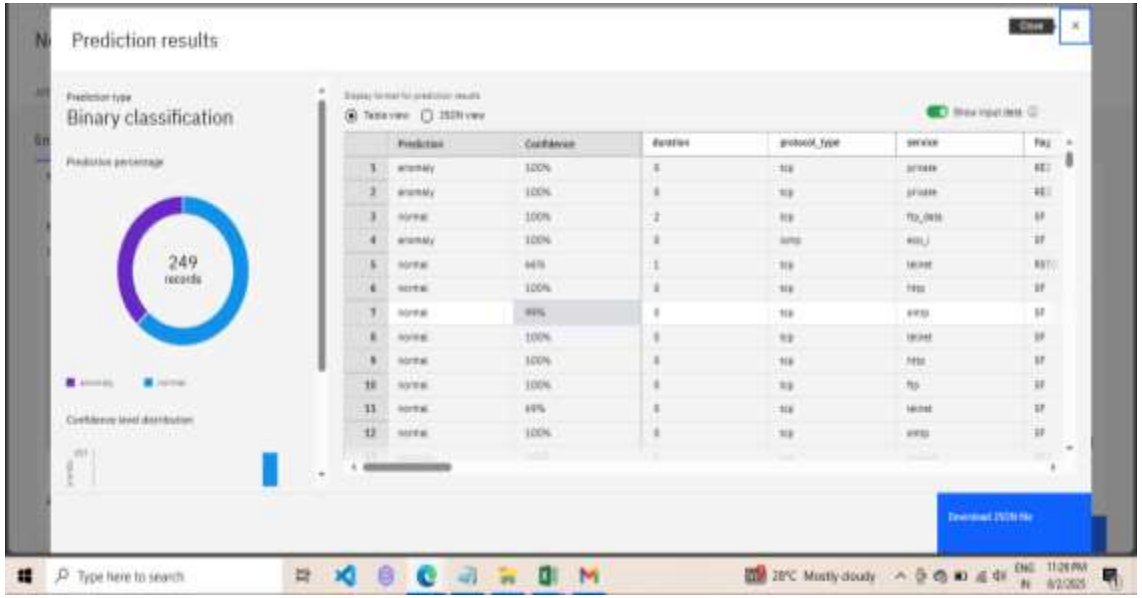


RESULT

Fig:Testing

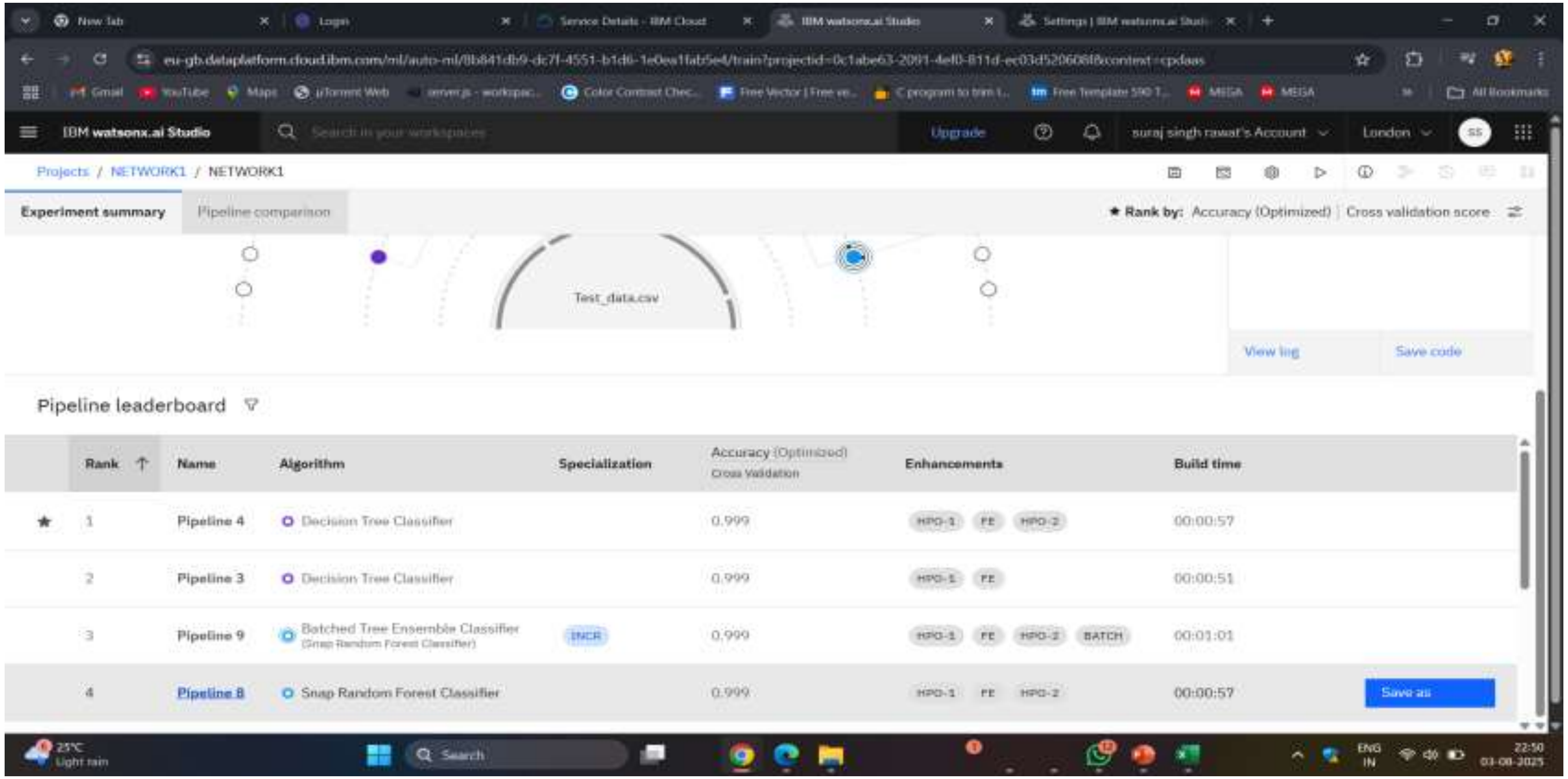
FIG:Prediction





PIPELINE LEADERBOARD

- Top 4 pipelines were all variations of XGBoost Classifier
- Shows AutoAI robustness and consistency in performance
- Final selected: Pipeline 6



CONCLUSION

- AutoAI efficiently selected the best pipeline with ~99.8% accuracy
- Minimal manual effort due to AutoAI automation
- Effective in detecting normal vs anomalous traffic
- Model deployed and ready for real-time detection

FUTURE SCOPE

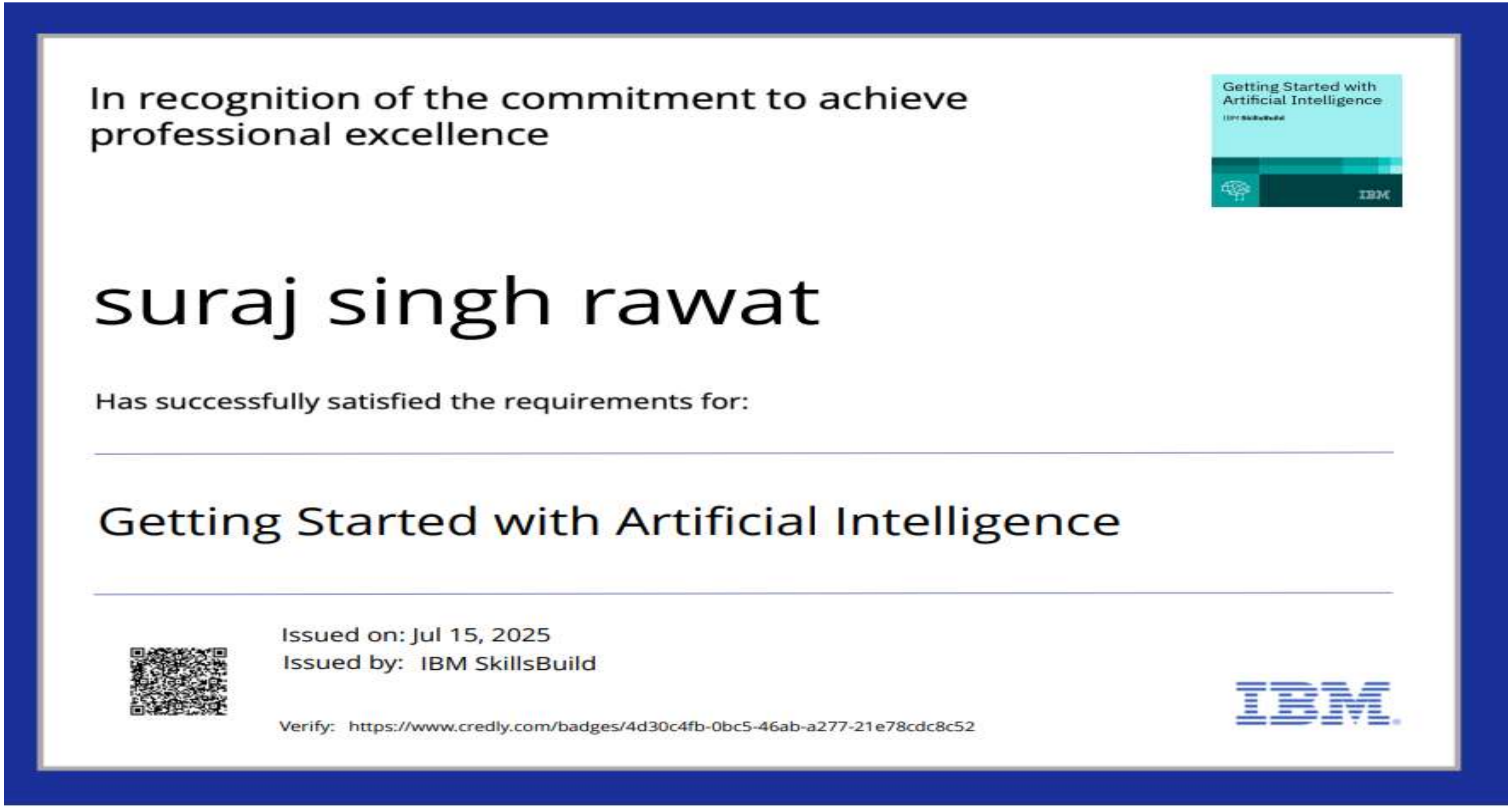
- Use real-time packet sniffers to feed live data
- Integrate advanced deep learning models like CNN-LSTM
- Enhance system with feedback-based retraining
- Extend to detect zero-day attacks using anomaly detection

REFERENCES

- Dataset: https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection
- IBM Cloud: https://cloud.ibm.com
- Tools: IBM Watsonx.ai Studio, AutoAI

IBM CERTIFICATIONS

- Screenshot/ credly certificate(getting started with AI)



Getting Started with Artificial Intelligence

In recognition of the commitment to achieve professional excellence

suraj singh rawat

Has successfully satisfied the requirements for:

Getting Started with Artificial Intelligence

Issued on: Jul 15, 2025
Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/4d30c4fb-0bc5-46ab-a277-21e78cdc8c52

IBM

IBM CERTIFICATIONS

- Screenshot/ credly certificate(Journey to Cloud)


In recognition of the commitment to achieve professional excellence

Journey to Cloud: Envisioning Your Solution

IBM

suraj singh rawat

Has successfully satisfied the requirements for:

Journey to Cloud: Envisioning Your Solution

Issued on: Jul 16, 2025
Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/df938733-03ed-4a71-9cbe-c01d3e7d004d

IBM

IBM CERTIFICATIONS

- Screenshot/ credly certificate(RAG Lab)

IBM **SkillsBuild** Completion Certificate

This certificate is presented to

Suraj Singh Rawat

for the completion of

Lab: Retrieval Augmented Generation with LangChain

(ALM-COURSE_3824998)

According to the Adobe Learning Manager system of record

Completion date: 24 Jul 2025 (GMT)

Learning hours: 20 mins

THANK YOU