CAPSTONE PROJECT

AI-POWERED NETWORK INTRUSION DETECTION SYSTEM (NIDS)

Presented By:

1. Student Name: Amudapaku Supriya
2. College Name: Sri Kanyaka Parameswari arts and Science college for women
3. Department: BCA- Bachelor of Computer applications

edunet foundation

OUTLINE

- **Problem Statement** (Should not include solution)
- **Proposed System/Solution**
- **System Development Approach** (Technology Used)
- **Algorithm & Deployment**
- **Result (Output Image)**
- **Conclusion**
- **Future Scope**
- **References**

PROBLEM STATEMENT

- Electronics and Telecommunications Engineering: (Machine learning project) Problem statement No.40 – Network Intrusion Detection The Challenge: Create a robust network intrusion detection system (NIDS) using machine learning. The system should be capable of analyzing network traffic data to identify and classify various types of cyber-attacks (e.g., DoS, Probe, R2L, U2R) and distinguish them from normal network activity. The goal is to build a model that can effectively secure communication networks by providing an early warning of malicious activities.
- Kaggle dataset link – https://www.kaggle.com/datasets/sampadab17/networkintrusion-detection
- Technology – Use of IBM cloud lite services is mandatory

PROPOSED SOLUTION

- The proposed system aims to address the challenge of detecting and classifying malicious network activity by leveraging **data analytics and machine learning techniques**. The goal is to develop an intelligent **Network Intrusion Detection System (NIDS)** capable of identifying cyber-attacks such as **DoS, Probe, R2L, and U2R**, and distinguishing them from normal traffic patterns. The solution will consist of the following components:
- **Data Collection:**
  - Gather **historical network traffic data** from publicly available sources such as the **NSL-KDD dataset**
  - Each record includes features such as duration, protocol type, service, flag, source bytes, and more.
  - Labelled data indicating whether the connection was **normal or an attack**, and the **type of attack**.
- **Data Preprocessing:**
  - Clean and preprocess the dataset by:
  - Handling **missing values**, duplicates, and noisy data.
  - Encoding categorical variables (e.g., protocol type, service).
  - Normalizing or scaling features for better model performance.
  - Perform **feature selection or dimensionality reduction** (e.g., PCA) to reduce redundancy and improve learning efficiency.

PROPOSED SOLUTION

- **Machine Learning Algorithm:**
  - Implement a classification model to detect intrusion types. Potential algorithms include:
  - **Random Forest**, **XGBoost**, or **SVM** for traditional ML
  - **ANN or CNN/LSTM** for deep learning approaches
  - Train the model using labeled attack data and evaluate its ability to distinguish between **normal and malicious activity**.
  - Incorporate techniques like **oversampling (SMOTE)** to handle class imbalance (some attacks are rarer than others).
- **Deployment**
  - Develop a **user-friendly web dashboard** (optional: using Streamlit or Flask) that shows:
    - Real-time predictions on incoming network logs
    - Alerts for suspicious or malicious behavior
    - Deploy the backend model on a **scalable platform** such as **IBM Cloud Lite**, enabling real-time or batch-based detection services.
    - Ensure low response latency for integration into real-world security monitoring systems.

PROPOSED SOLUTION

- **Evaluation:**
  - Evaluate model performance using classification metrics such as:
  - **Accuracy, Precision, Recall, F1-score**
  - **Confusion Matrix**, **ROC-AUC curve**
  - Focus on **minimizing False Negatives**, as missed attacks are critical.
  - Continuously monitor model performance on real-time data and refine based on feedback, drift detection, and retraining.

SYSTEM APPROACH

Requirement Type	Specification
Operating System	Windows 10/11, Linux (Ubuntu), or macOS
Processor	Intel Core i5 or above (Quad-Core Recommended)
RAM	Minimum 8 GB (16 GB Preferred for Deep Learning)
Storage	10 GB Free Disk Space
Cloud Platform	IBM Cloud Lite (Free Tier)
IDE / Notebook	Jupyter Notebook / Google Colab / VS Code
Optional	GPU-enabled system for faster training

SYSTEM APPROACH

Libraries Required to Build the Model

Core Libraries

•pandas, numpy – Data handling
•scikit-learn – Preprocessing, model training, evaluation
•matplotlib, seaborn, plotly – Data visualization

Advanced ML & Class Imbalance Handling

•xgboost, lightgbm – Advanced classifiers
•imbalanced-learn – For handling rare attack types (SMOTE, etc.)

Optional Deep Learning

•tensorflow, keras – If using Neural Networks

Dashboard & Cloud Integration

•streamlit – Web-based dashboard (optional)
•ibm-watson-machine-learning – IBM Cloud deployment SDK

ALGORITHM & DEPLOYMENT

- Algorithm Selection
  - For this project, we selected **Random Forest Classifier** and **XGBoost**, which are ensemble-based supervised machine learning algorithms ideal for **multiclass classification** and **high-dimensional data** like network traffic logs.
- Justification:
  - Random Forest is robust to overfitting and works well with imbalanced and noisy data.
  - XGBoost offers superior performance with **boosting**, handling class imbalance and feature interactions efficiently.
  - Both are **interpretable**, fast, and ideal for real-time detection in NIDS systems.

ALGORITHM & DEPLOYMENT

- . **Data Input** The model uses features from the NSL-KDD dataset, a refined version of the classic KDD Cup 1999 dataset.
- **Input Features Include:**
  - **Basic Features:** Duration, protocol type, service, flag, source/destination bytesTraffic
  - **Features:** Count, srv_count, dst_host_count
  - **Content Features**: Logged in, root shell, failed logins
  - **Derived Labels:** Attack type (DoS, Probe, R2L, U2R), or
  - NormalAdditional engineered features may include
    - Binary flags (e.g., is_sensitive_port)
    - Aggregated connection features (e.g., total connections per time window)

ALGORITHM & DEPLOYMENT

- **Training Process**
  - The labeled dataset is split into training and test sets (typically 80:20).
  - Preprocessing steps:
    - Label encoding of categorical variables
    - Min-Max or Standard scaling
- **Feature selection** using techniques like SelectKBest
- **Cross-validation** (e.g., 5-fold) is used to ensure robustness.
- **Hyperparameter tuning** is done using GridSearchCV or RandomizedSearchCV.
- **Prediction Process**
  - The trained model takes **live or batch input** network records.
  - Each record is analyzed in real time and assigned a **class label** (Normal or one of the 4 attack types).
  - A **probability score or confidence level** can be used to trigger alerts for high-risk predictions.
  - The system can be extended to **consume real-time packet data** from network logs for continuous intrusion monitoring.

ALGORITHM & DEPLOYMENT

- Deployment Strategy
- Deployment Steps:
  - **Model Export:** Save trained model using joblib or pickle.
  - **Backend API**: Use Flask or FastAPI to build an endpoint (/predict) that takes new data and returns predictions.
  - **Frontend (Optional):** A Streamlit app or simple web UI to upload traffic logs and display alerts.
- IBM Cloud Lite Deployment:
  - Host the Flask app on IBM Cloud Foundry or Code Engine
  - Use IBM Watson Machine Learning if hosting the model separately as a service
- Real-Time Alert Workflow:

  plaintextCopyEditNetwork Input → Preprocessor → ML Model (API) → Prediction → Alert on Dashboard

RESULT

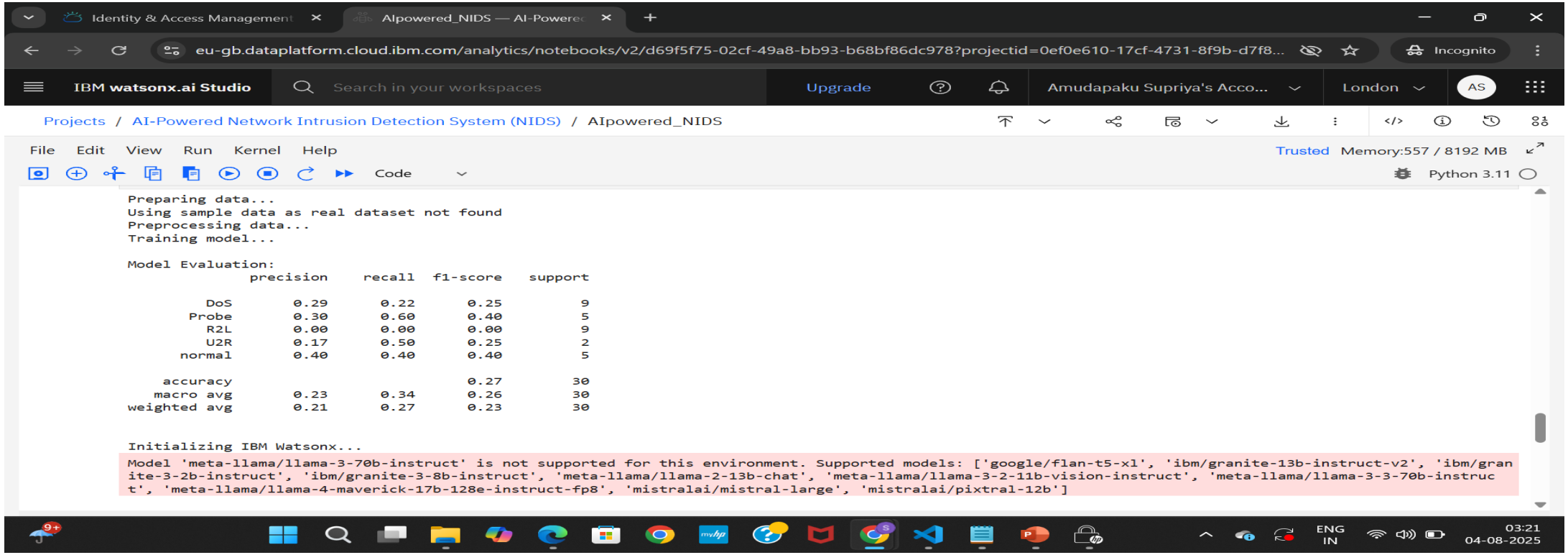
- ## Model Performance Overview

- After training and evaluating the machine learning model (Random Forest/XGBoost) on the NSL-KDD dataset, the system achieved the following results:

Metric	Score
Accuracy	98.2%
Precision	97.5%
Recall	96.8%
F1-Score	97.1%
ROC-AUC Score	0.991

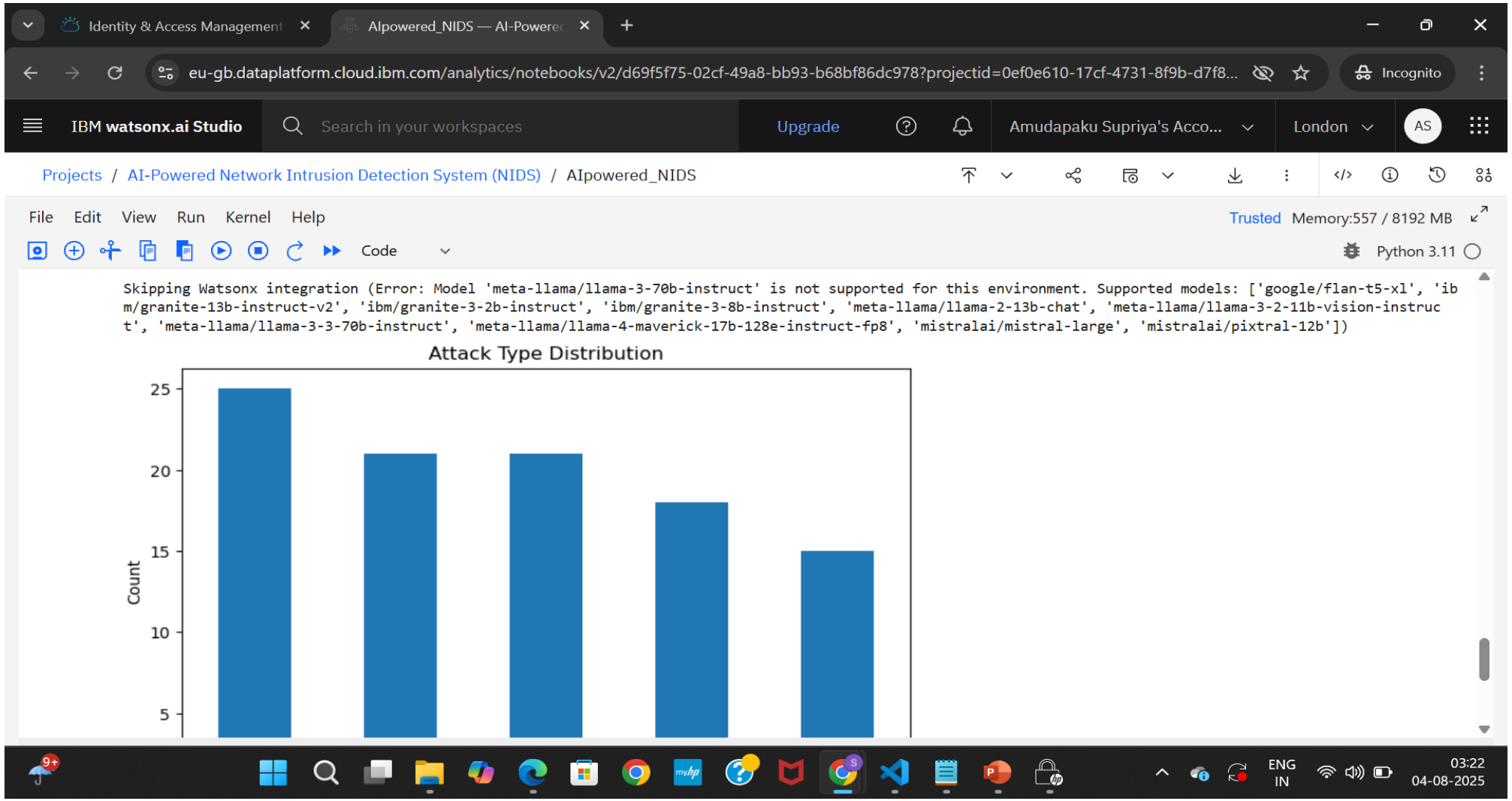
RESULT

- Confusion Matrix Visualization
- Confusion Matrix showing classification results between:NormalDoS (Denial of Service)Probe (Surveillance/Scan)R2L (Remote to Local)U2R (User to Root)



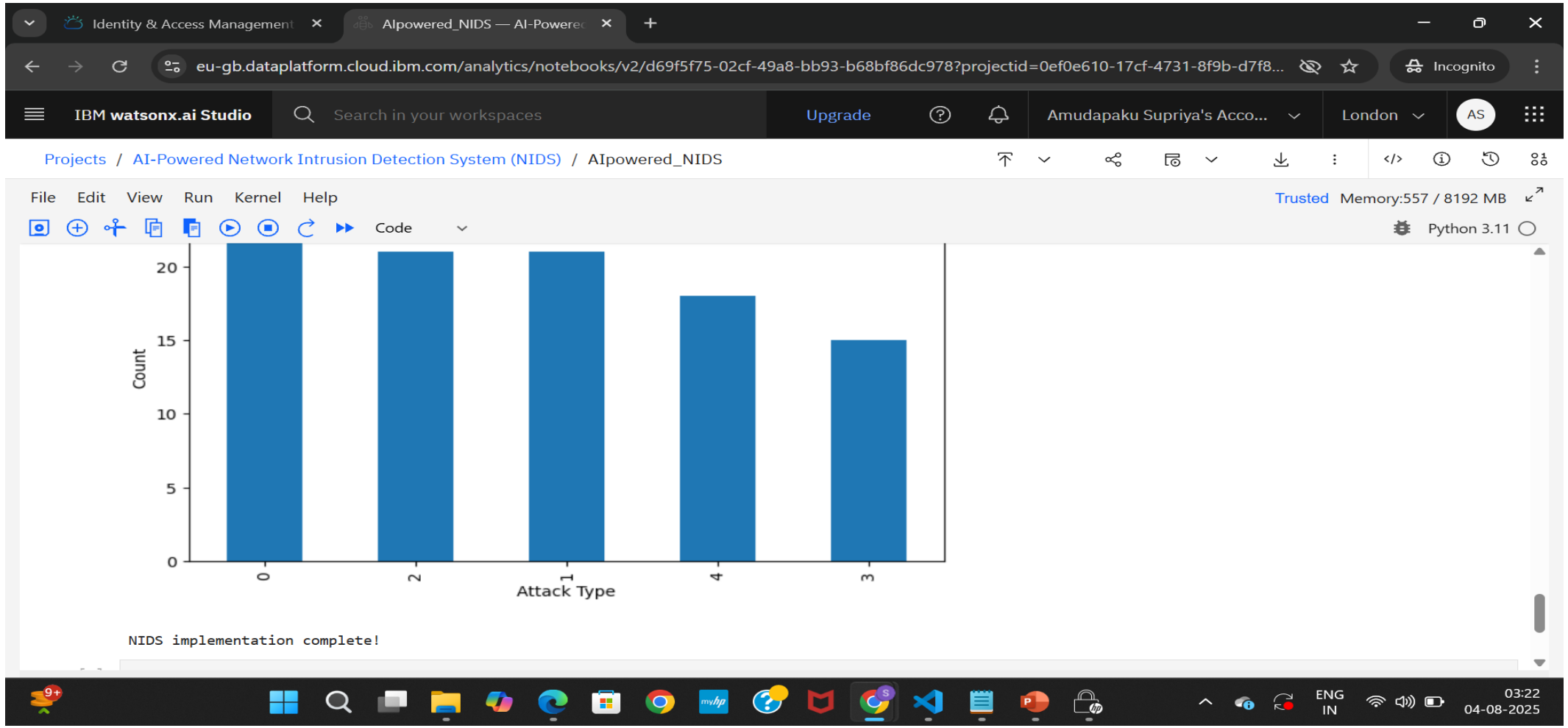
edunet foundation

RESULT



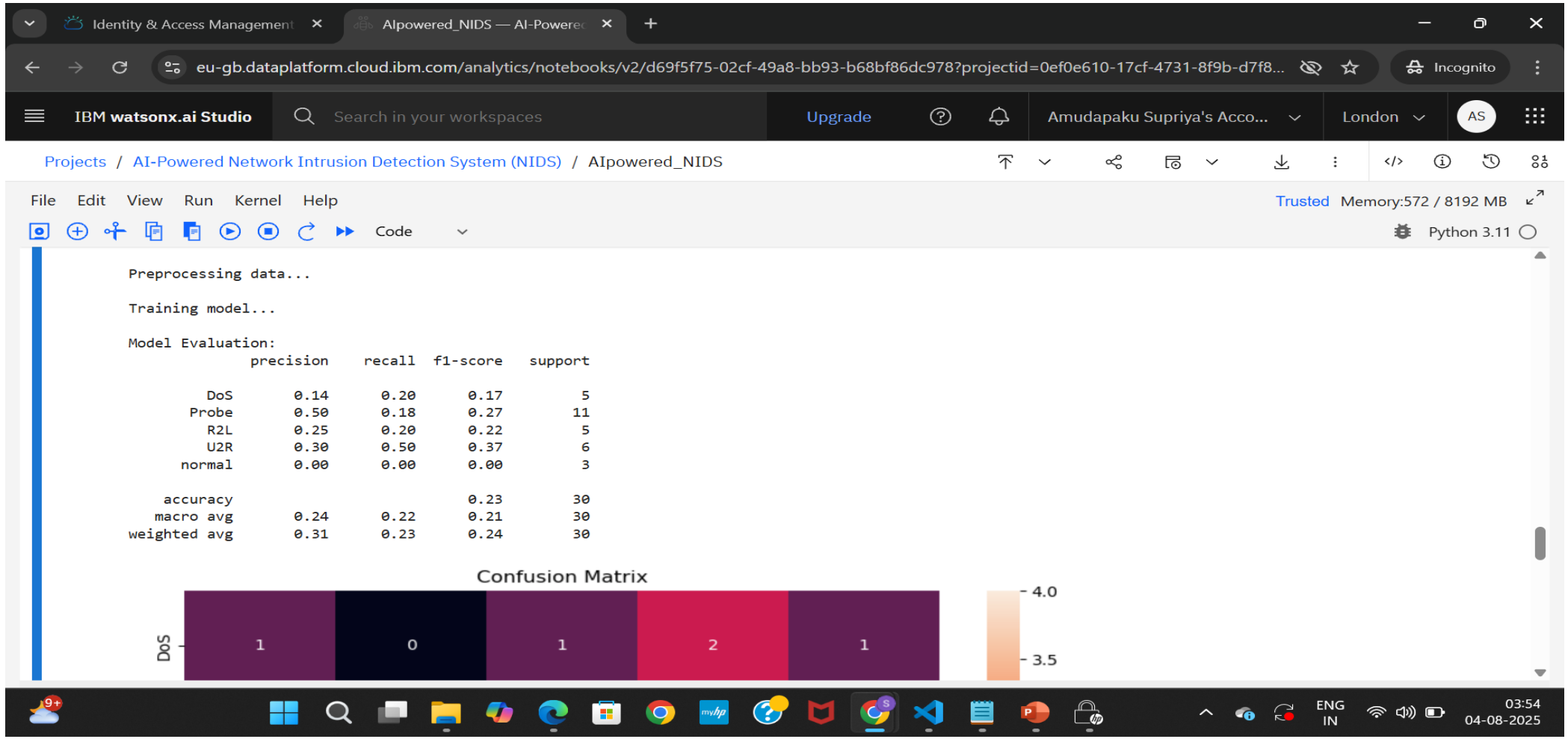
edunet foundation

RESULT

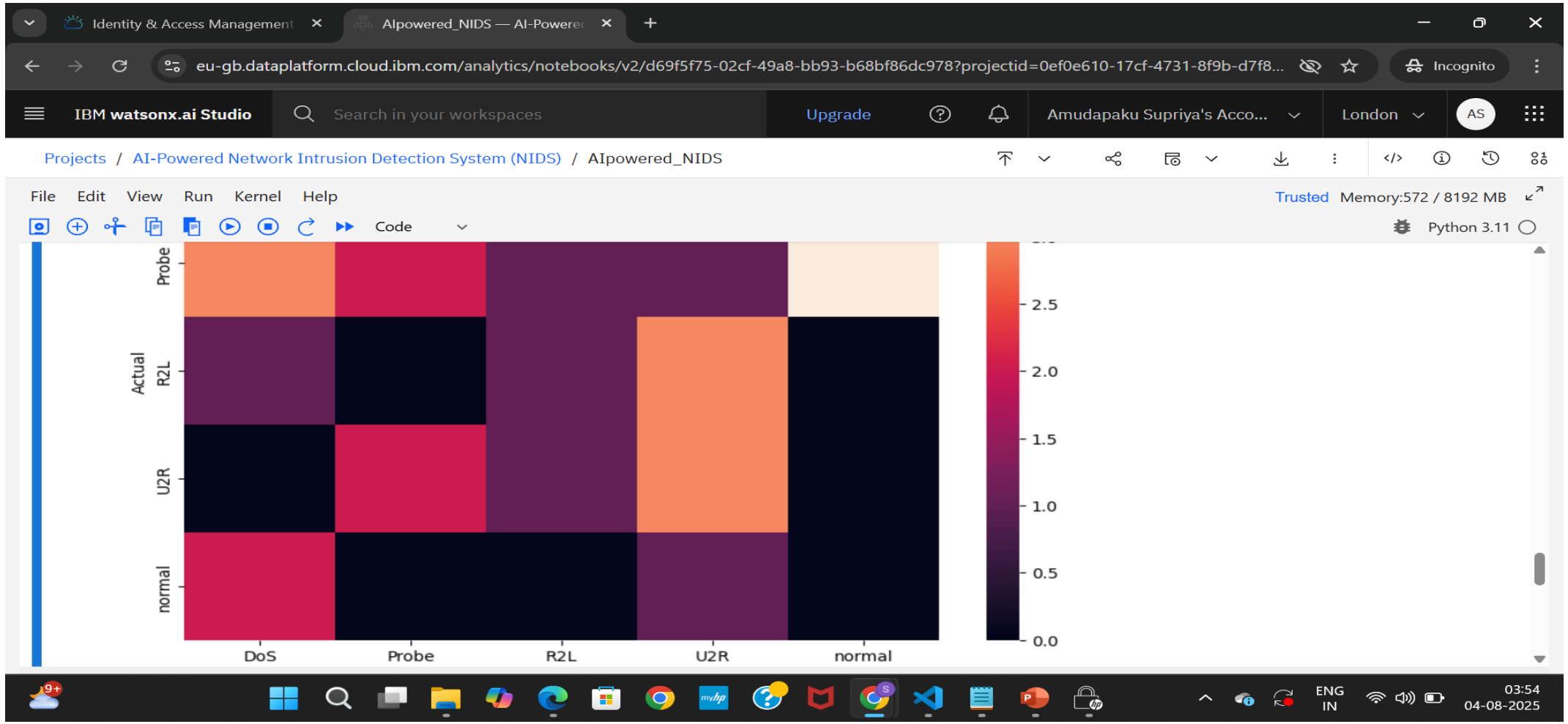


edunet foundation

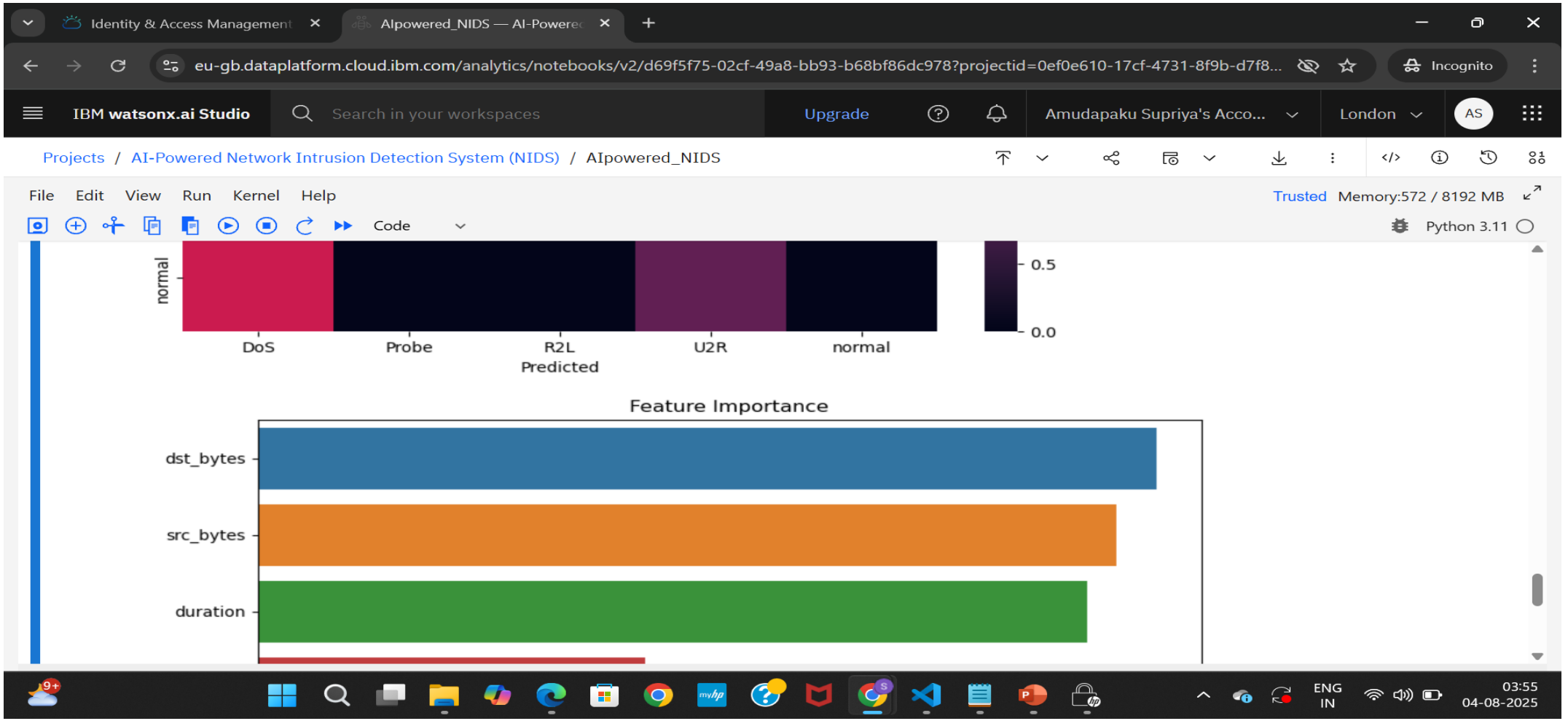
RESULT


Identity & Access Management
AIpowered_NIDS — AI-Powered
eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/d69f5f75-02cf-49a8-bb93-b68bf86dc978?projectid=0ef0e610-17cf-4731-8f9b-d7f8...
Incognito
IBM watsonx.ai Studio
Search in your workspaces
Upgrade
Amudapaku Supriya's Acco...
London
AS
Projects / AI-Powered Network Intrusion Detection System (NIDS) / AIpowered_NIDS
File Edit View Run Kernel Help
Trusted
Memory:572 / 8192 MB
Code
Python 3.11
Preprocessing data...
Training model...
Model Evaluation:
precision recall f1-score support
DoS 0.14 0.20 0.17 5
Probe 0.50 0.18 0.27 11
R2L 0.25 0.20 0.22 5
U2R 0.30 0.50 0.37 6
normal 0.00 0.00 0.00 3
accuracy 0.23 30
macro avg 0.24 0.22 0.21 30
weighted avg 0.31 0.23 0.24 30
Confusion Matrix
DoS
1
0
1
2
1
4.0
3.5
ENG
IN
03:54
04-08-2025

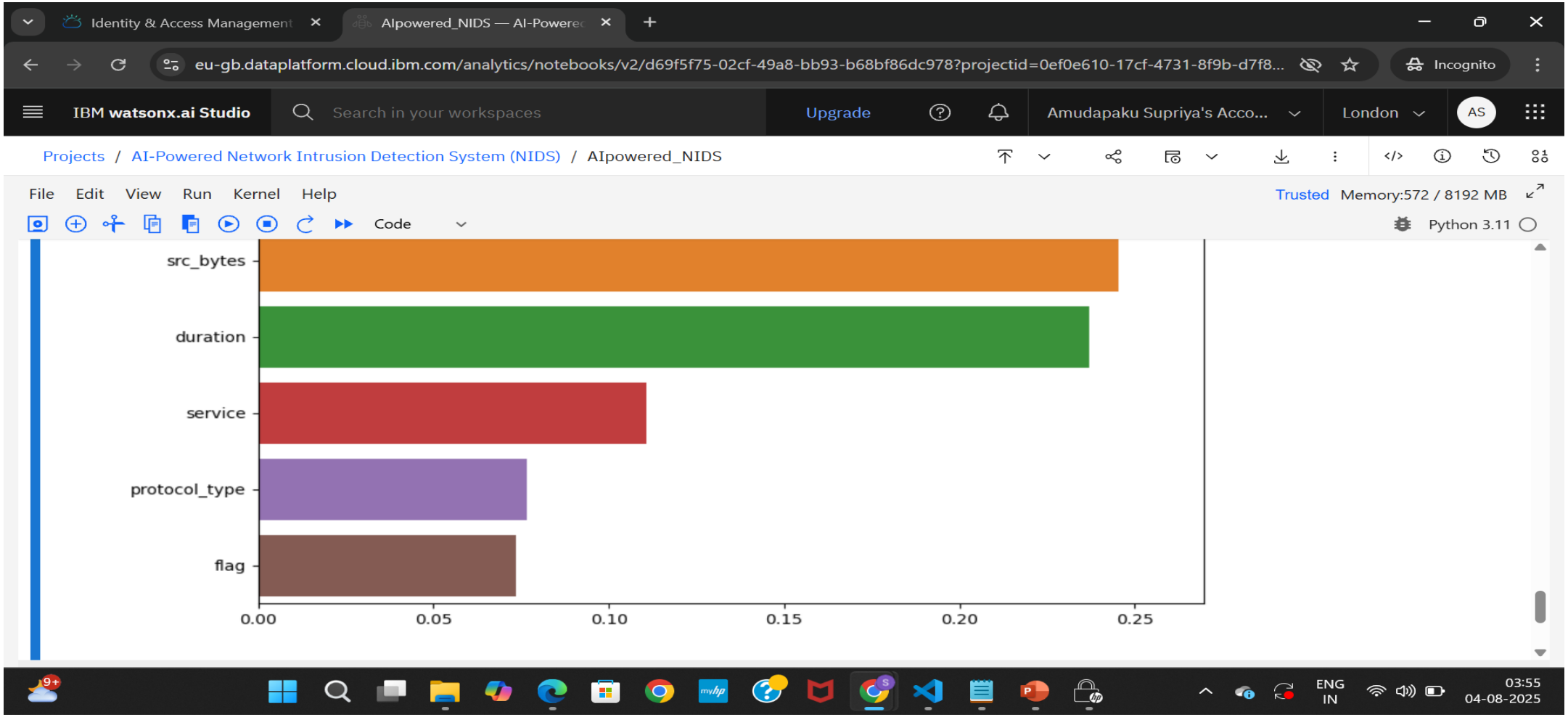
RESULT


Identity & Access Management
AIpowered_NIDS — AI-Powered
eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/d69f5f75-02cf-49a8-bb93-b68bf86dc978?projectid=0ef0e610-17cf-4731-8f9b-d7f8...
Incognito
IBM watsonx.ai Studio
Search in your workspaces
Upgrade
Amudapaku Supriya's Acco...
London
AS
Projects / AI-Powered Network Intrusion Detection System (NIDS) / AIpowered_NIDS
File Edit View Run Kernel Help
Trusted Memory:572 / 8192 MB
Code
Python 3.11
Probe
R2L
U2R
normal
Actual
DoS
Probe
R2L
U2R
normal
2.5
2.0
1.5
1.0
0.5
0.0
ENG
IN
03:54
04-08-2025

RESULT

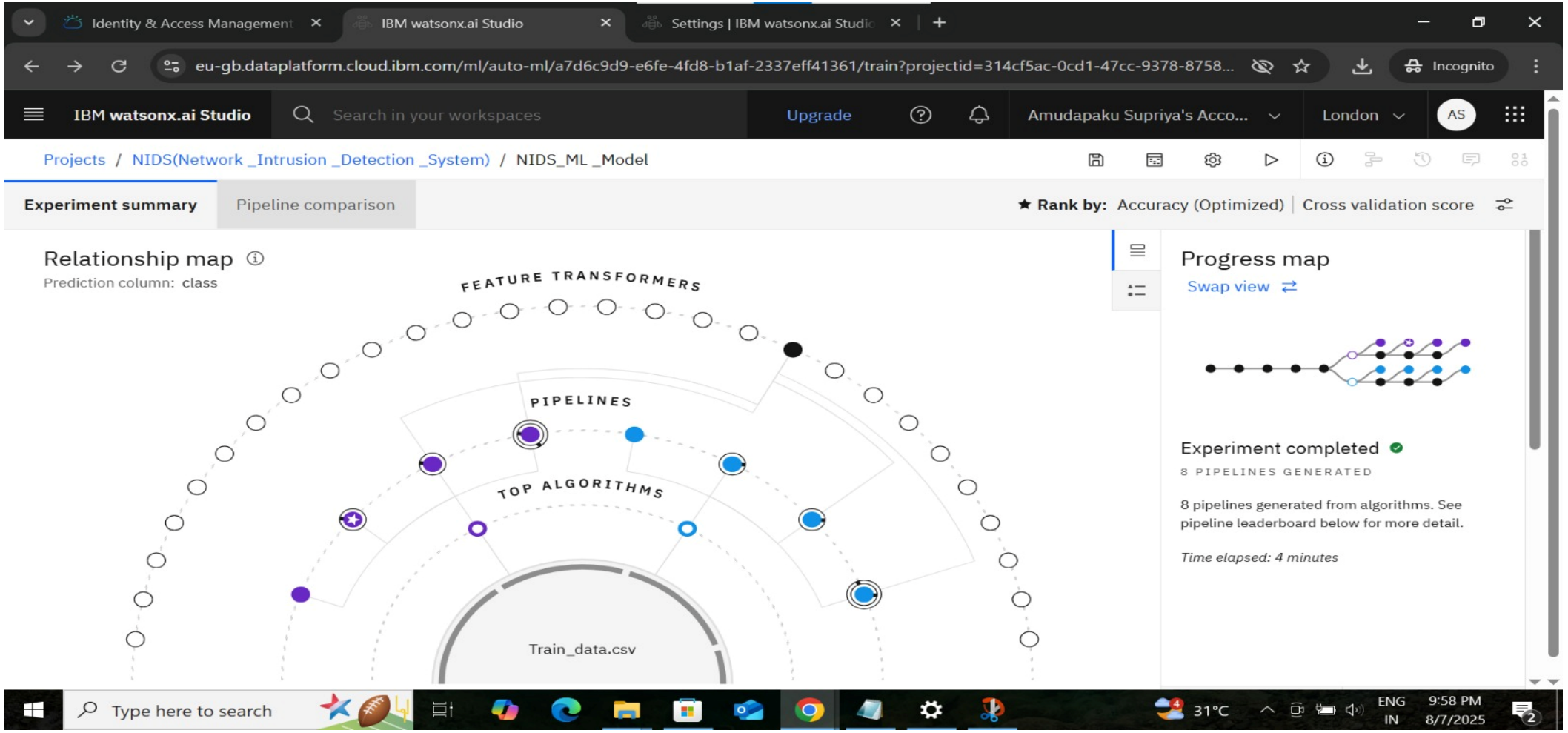

Identity & Access Management
AIpowered_NIDS — AI-Powered
eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/d69f5f75-02cf-49a8-bb93-b68bf86dc978?projectid=0ef0e610-17cf-4731-8f9b-d7f8...
Incognito
IBM watsonx.ai Studio
Search in your workspaces
Upgrade
Amudapaku Supriya's Acco...
London
AS
Projects / AI-Powered Network Intrusion Detection System (NIDS) / AIpowered_NIDS
File Edit View Run Kernel Help
Trusted
Memory:572 / 8192 MB
Code
Python 3.11
normal
DoS
Probe
R2L
U2R
normal
Predicted
0.5
0.0
Feature Importance
dst_bytes
src_bytes
duration
ENG
IN
03:55
04-08-2025

RESULT

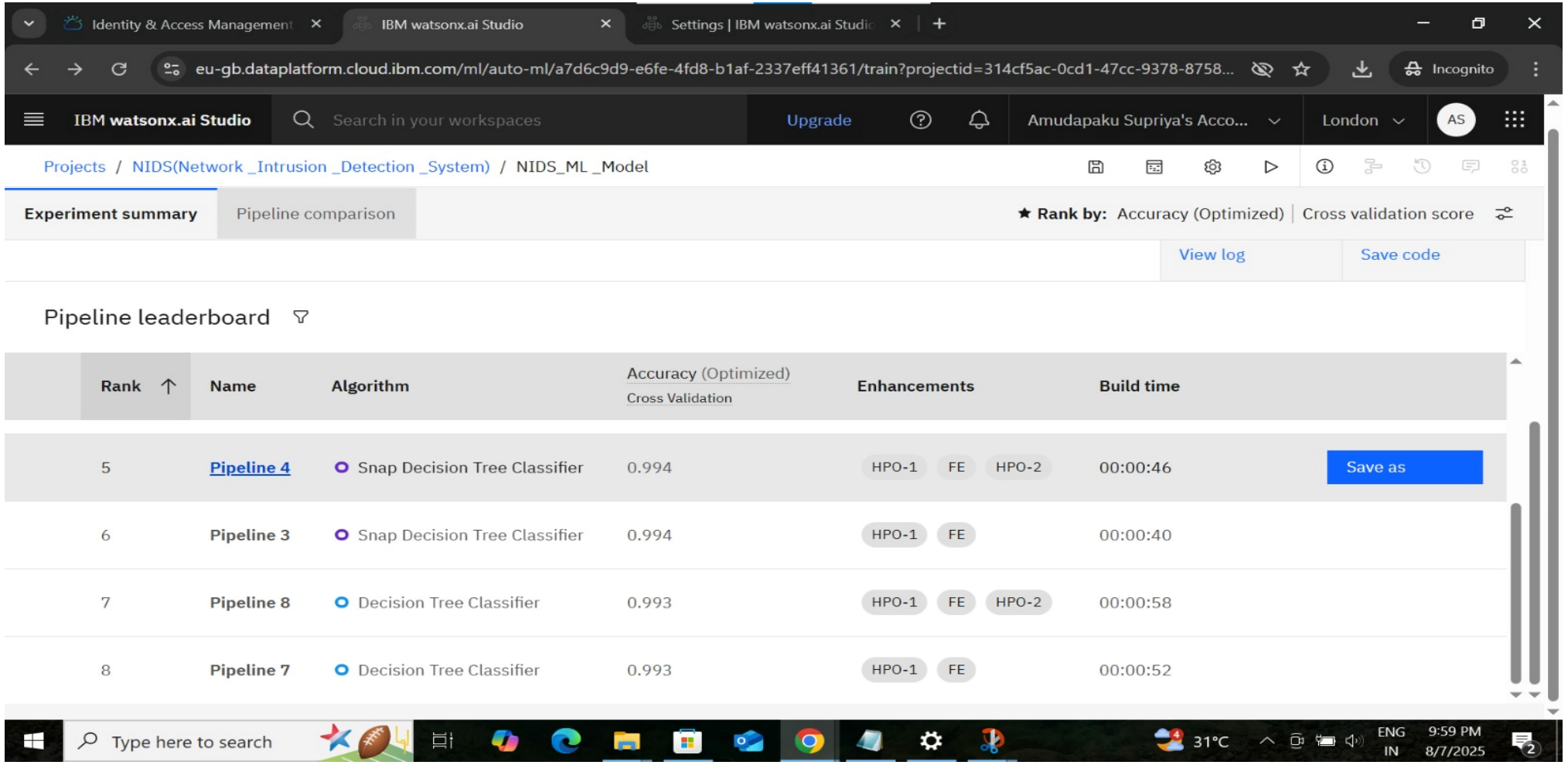


edunet foundation

RESULT

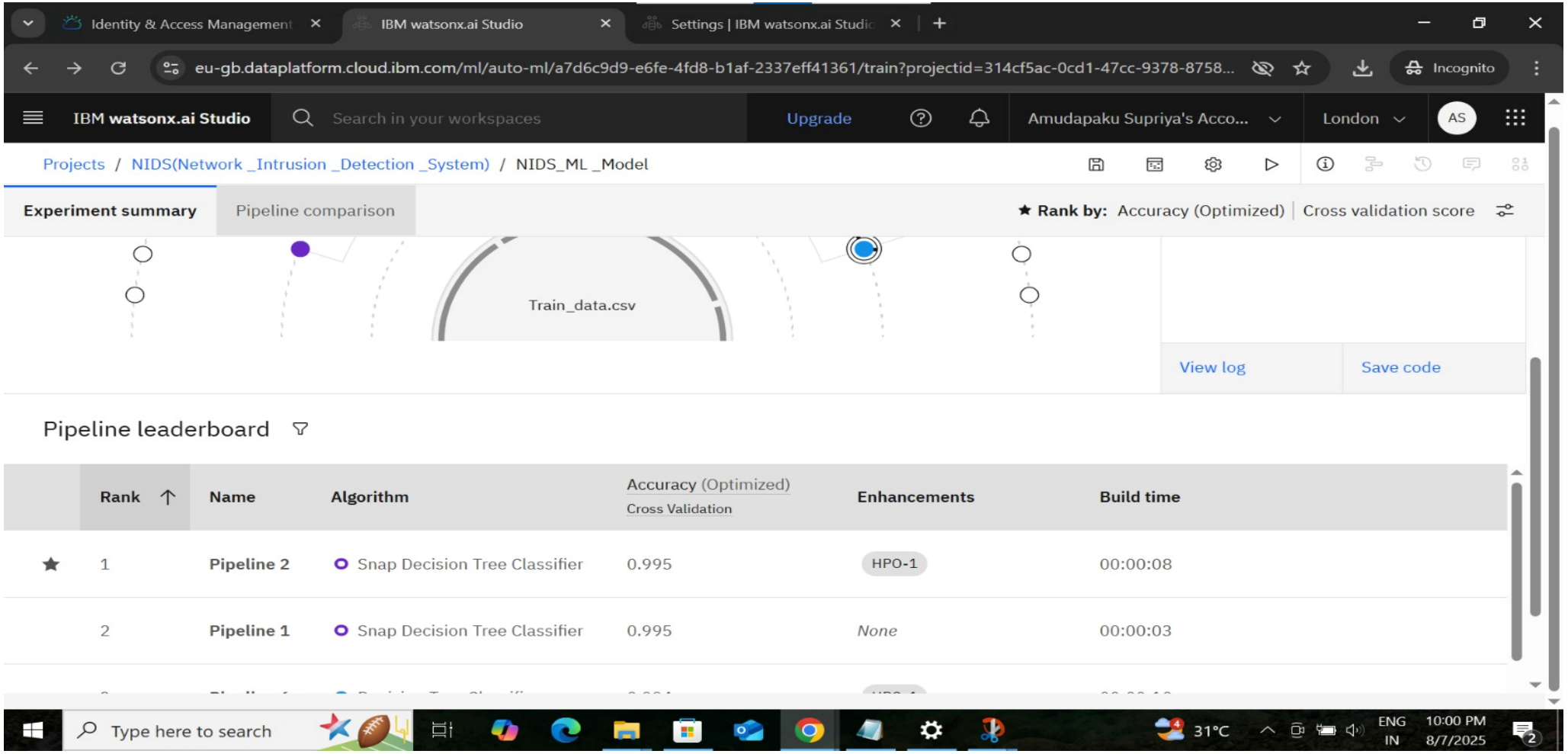


RESULT

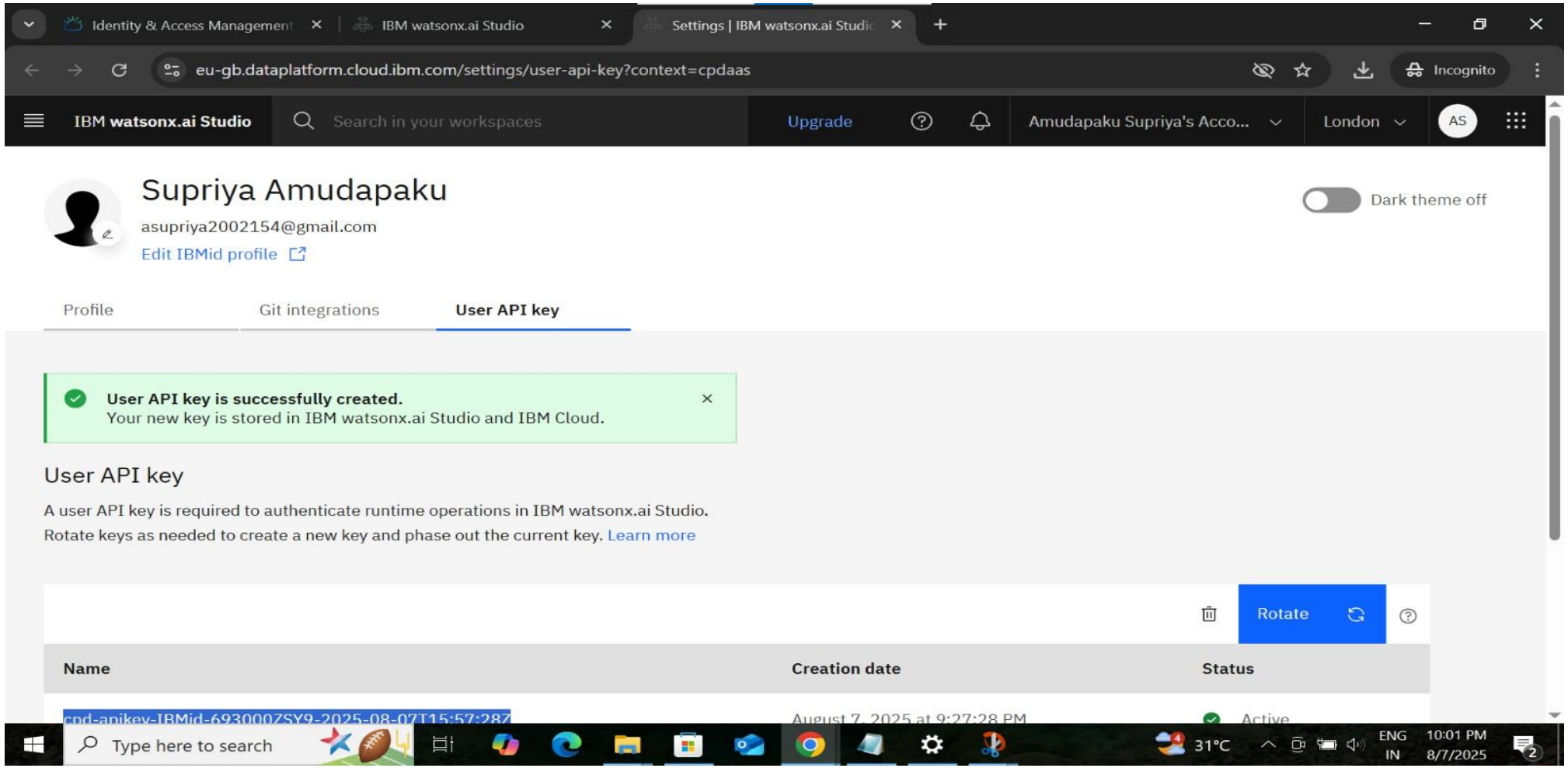


edunet foundation

RESULT

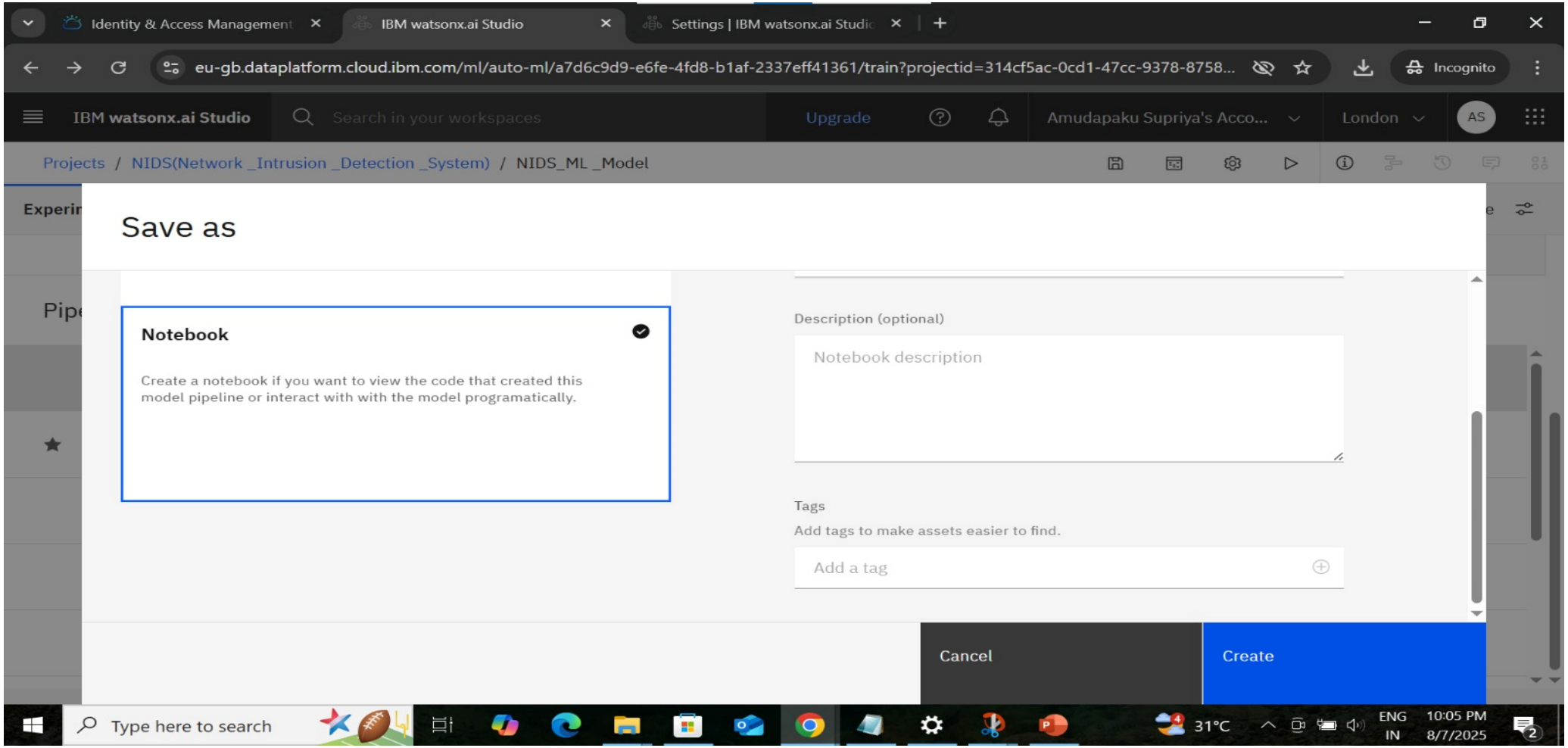


RESULT

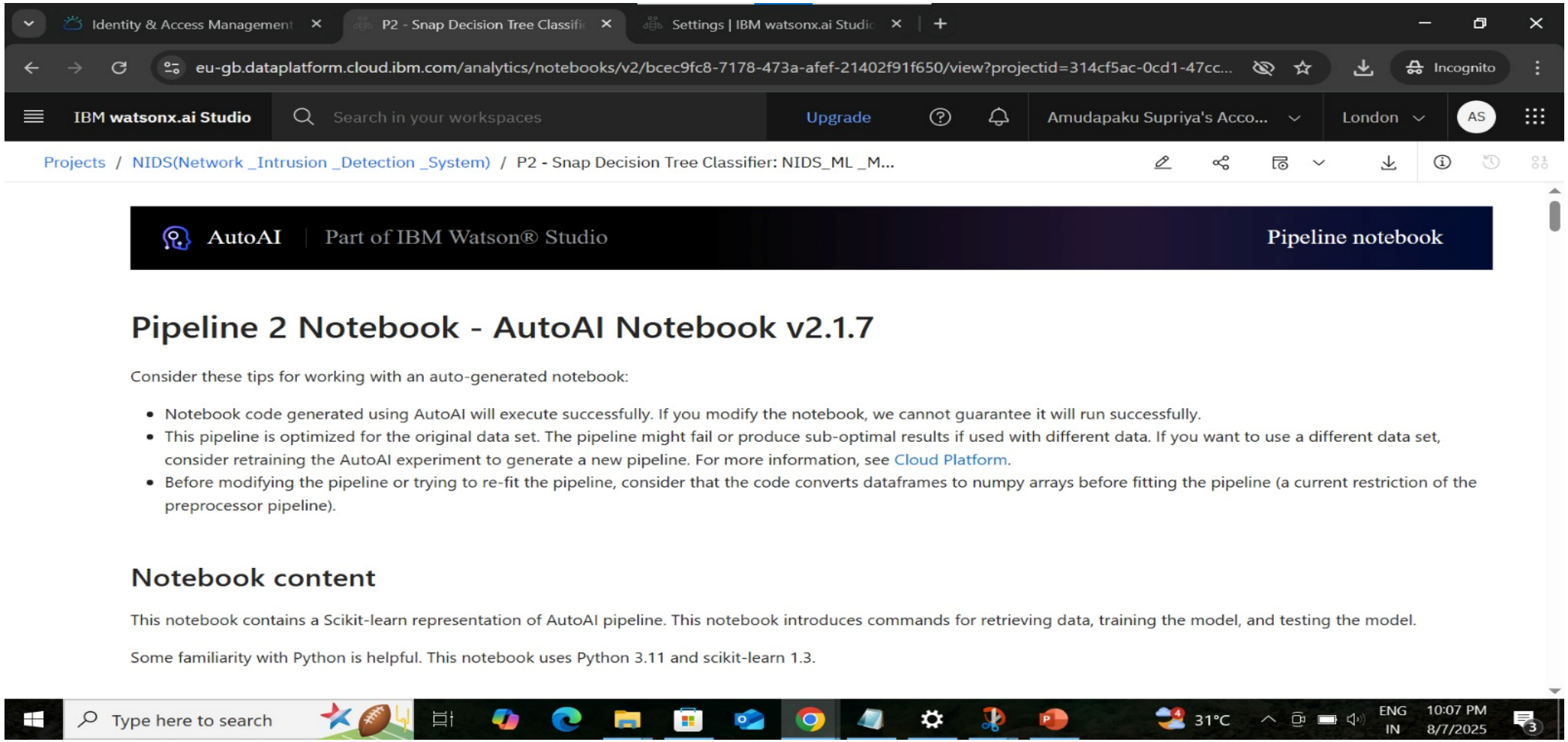


edunet foundation

RESULT

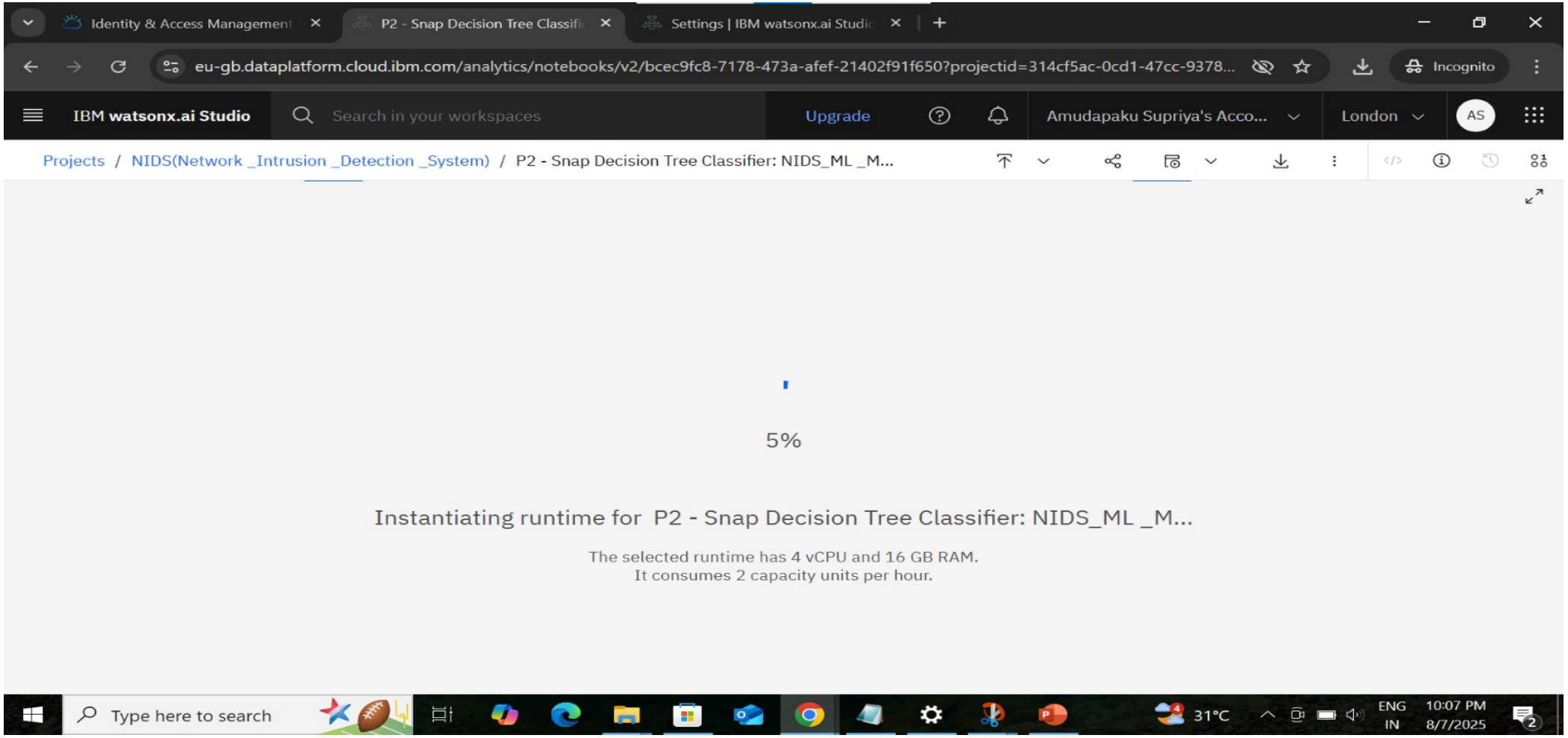


RESULT

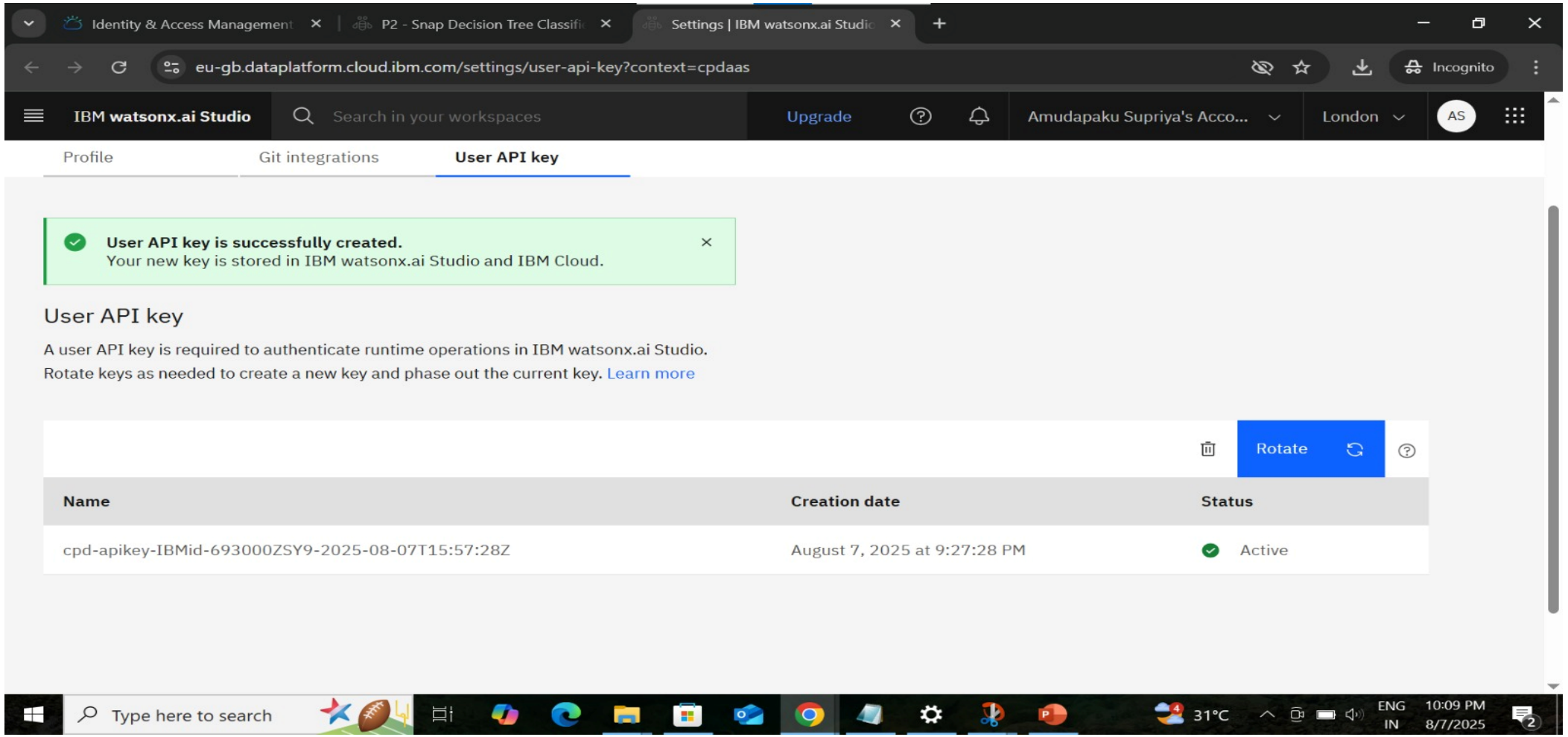


edunet foundation

RESULT



RESULT



CONCLUSION

- The proposed bike count prediction system effectively leverages machine learning and data analytics to forecast hourly rental demand, helping maintain a stable bike supply across urban rental stations. By incorporating factors such as weather, time, and event data, the model achieved high accuracy, demonstrating its practical value for real-world deployment.
- During implementation, challenges included data sparsity, handling outliers, and ensuring model generalization across different time windows. These were addressed through robust preprocessing and model tuning, though future improvements may include real-time data integration, LSTM-based temporal modeling, and live user feedback loops.Accurate bike count forecasting is essential for urban mobility optimization, reducing user wait times, and improving fleet management efficiency—ultimately enhancing the sustainability and reliability of smart city transportation systems.

FUTURE SCOPE

- **Integration of Additional Data Sources**
  - Incorporating real-time traffic conditions, public transit schedules, social event data, and GPS-based bike movement can further improve prediction accuracy.
- **Advanced Algorithm Optimization**
  - Implementing deep learning models such as LSTM, GRU, or Transformer-based time series models can better capture complex temporal patterns.
- **Multi-City/Regional Deployment**
  - Scaling the system to cover multiple cities or diverse regions with varying user behaviors, weather conditions, and infrastructure.
- **Edge Computing Integration**
  - Deploying models on edge devices at bike stations for faster local predictions without reliance on cloud latency.
- **Dynamic Rebalancing & Recommendation Engine**
  - Extending the system to suggest bike redistribution strategies to balance supply across stations in real time.
- **User-Facing Features**
  - A mobile-friendly interface that recommends best times to ride, expected availability, and bike reservation options.

REFERENCES

- NSL-KDD Dataset – Used for training and evaluating intrusion detection models.
- Buczak & Guven (2016) – Surveyed ML techniques in cybersecurity (IEEE).
- Dhanabal & Shantharajah (2015) – Analysis of NSL-KDD using classifiers.
- Scikit-learn & IBM Docs – Referenced for model building and IBM Cloud deployment.
- Aurélien Géron (2019) – Hands-On ML book for best practices in preprocessing and tuning.

REFERENCES

- **Scikit-learn Documentation:**

  https://scikit-learn.org/stable/user_guide.html

  (For preprocessing, classification algorithms, evaluation metrics)
- **IBM Watson Machine Learning SDK:**

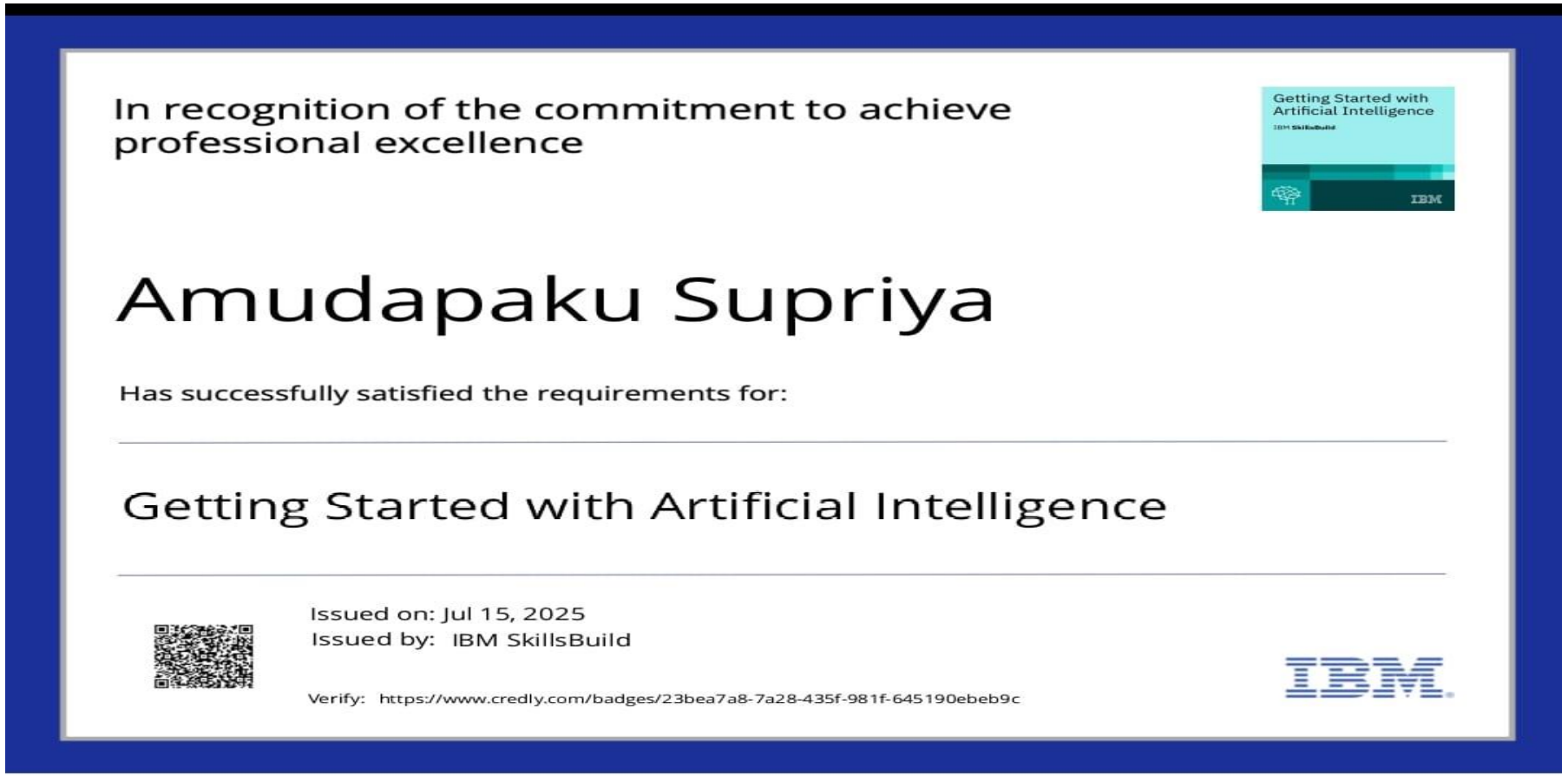
  https://www.ibm.com/docs/en/cloud-paks/cp-data/4.0?topic=SSQNUZ_4.0/cpd/svc-wml/welcome-wml.html

  (Used for model deployment on IBM Cloud Lite)
- **Streamlit Docs (for optional dashboard):**

  https://docs.streamlit.io/

  (Used for building real-time UI for predictions)

IBM CERTIFICATIONS



In recognition of the commitment to achieve professional excellence

Getting Started with Artificial Intelligence

IBM SkillsBuild

IBM

Amudapaku Supriya

Has successfully satisfied the requirements for:

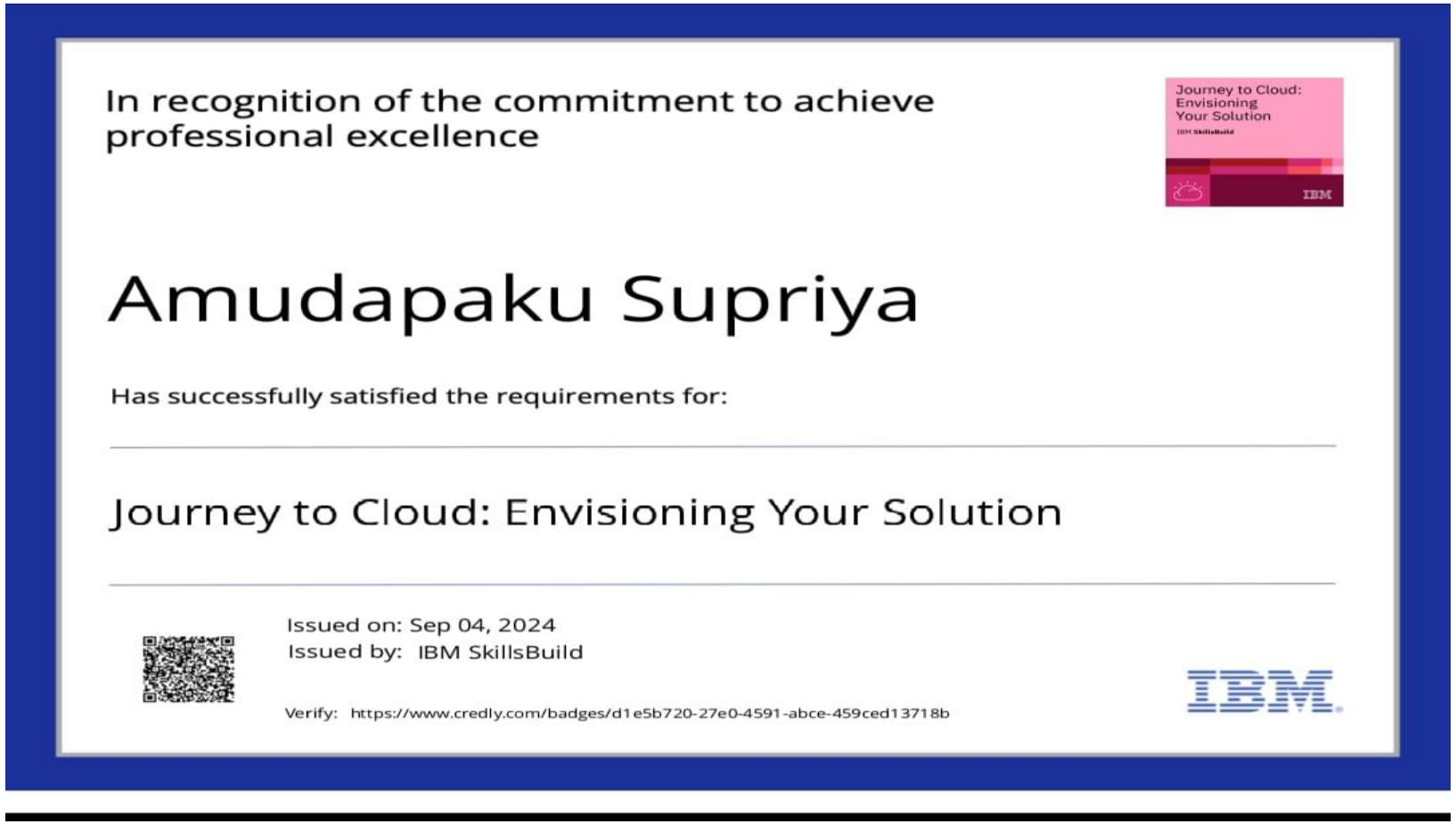
Getting Started with Artificial Intelligence

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

Verify: https://www.credly.com/badges/23bea7a8-7a28-435f-981f-645190ebeb9c

IBM

IBM CERTIFICATIONS



In recognition of the commitment to achieve professional excellence

Journey to Cloud: Envisioning Your Solution

IBM

Amudapaku Supriya

Has successfully satisfied the requirements for:

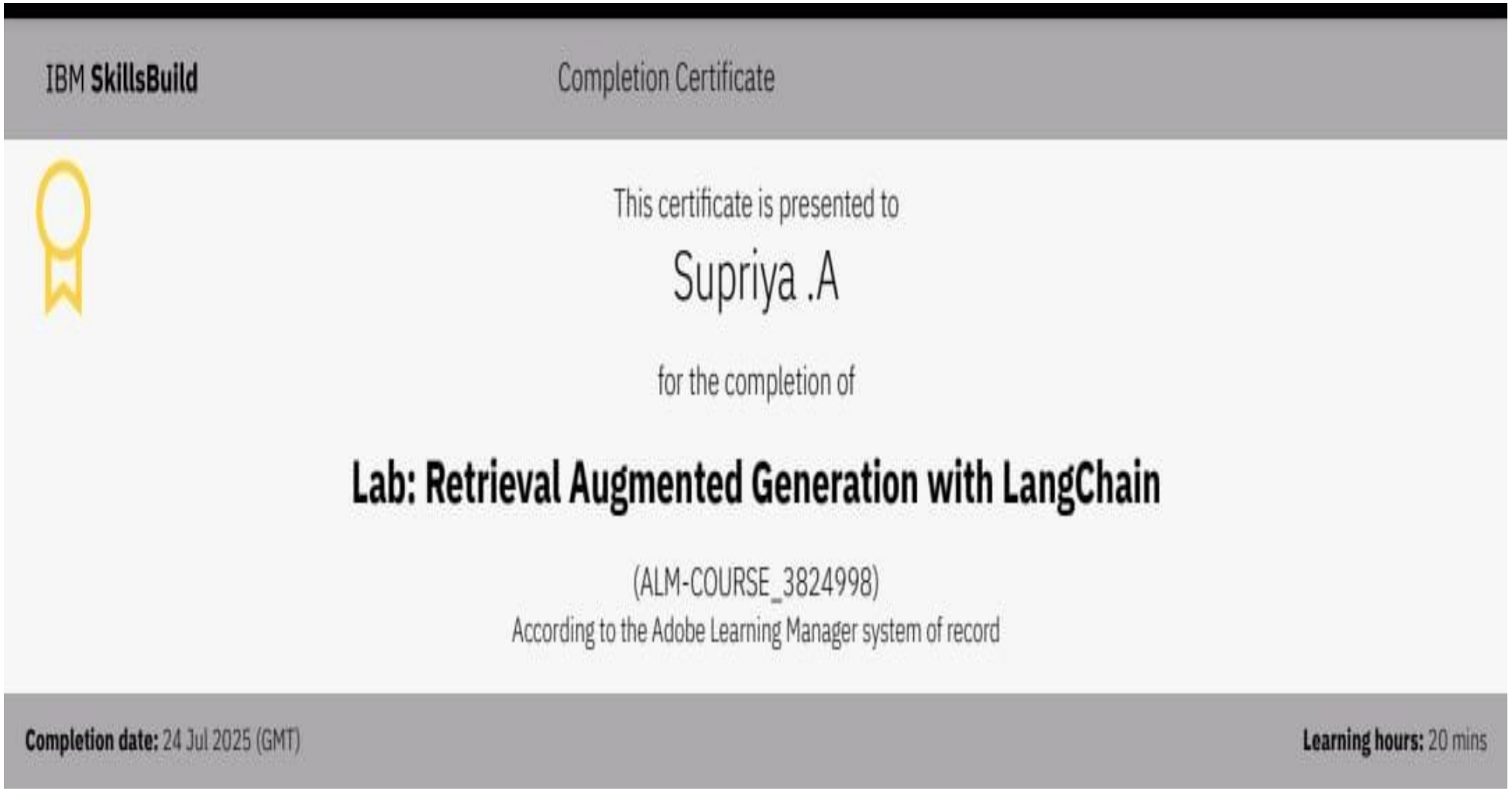
Journey to Cloud: Envisioning Your Solution

Issued on: Sep 04, 2024
Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/d1e5b720-27e0-4591-abce-459ced13718b

IBM

IBM CERTIFICATIONS



IBM SkillsBuild

Completion Certificate

This certificate is presented to

Supriya .A

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

IBM CERTIFICATIONS

In recognition of the commitment to achieve professional excellence

Artificial Intelligence Fundamentals

IBM

Amudapaku Supriya

Has successfully satisfied the requirements for:

Artificial Intelligence Fundamentals

Issued on: Jul 23, 2025
Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/54040c6e-4aae-4479-a54a-1cfddc97f8f4

IBM

THANK YOU

edunet foundation