CAPSTONE PROJECT

NETWORK INTRUSION DETECTION

Presented By:
1. **Student Name:- Yash Umesh Patil**
   College Name:- Government College of Engineering, Jalgaon(M.
   Department:- Computer Engineering

edunet foundation

OUTLINE

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

PROBLEM STATEMENT

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.

PROPOSED SOLUTION

- The proposed system aims to detect malicious network activity by analyzing traffic data and classifying it as either normal or intrusive. The approach leverages machine learning via IBM Watsonx.ai and includes the following components:
- **Data Collection:**
  - Used Kaggle dataset: *Network Intrusion Detection*
  - Collected network traffic logs with 41 features and a target label (class**).**
- **Data Preprocessing:**
  - Uploaded CSV to IBM Watsonx.ai
  - No manual preprocessing required — AutoAI handled:
    1. Missing values
    2. Feature transformations
    3. Data normalization
- **Machine Learning Algorithm:**
  - Used **IBM Watsonx AutoAI** to automatically generate pipelines
  - Selected model: **Snap Decision Tree Classifier**
  - Enhancements applied:
    - Feature Engineering (FE)
    - Hyperparameter Optimization (HPO1, HPO2)

PROPOSED SOLUTION

Deployment:

- Promoted model to **IBM Cloud Deployment Space**
- Deployed as an **online REST API**
- Supports real-time predictions using JSON input

- **Evaluation:**
  - AutoAI evaluated models using **accuracy** as the main metric
  - 8 pipelines compared; selected the best-performing one
  - Deployment tested using sample network input

- **Result:**
  - Model accurately classified traffic as normal or intrusive
  - Deployment is live and accessible via API
  - System is scalable, reliable, and ready for further integration

SYSTEM APPROACH

The "System Approach" outlines the architecture and methodology used for building and deploying the Network Intrusion Detection System using IBM Cloud. Below is the structured breakdown:

- **System requirements:**
  - **Platform:** IBM Watsonx.ai on IBM Cloud Lite
  - **Access:** IBM Cloud account with AutoAI and Deployment Space enabled
  - **Input Format:** CSV file (42 columns including target label)
  - **Output:** Predicted class (normal or attack)
  - **Deployment Type:** REST API (Online deployment)

- **Library required to build the model:**
  - Since the model was developed using **IBM Watsonx AutoAI**, coding libraries were abstracted. However, under the hood, the following technologies were involved:
    - **AutoAI (automated ML pipeline generator)**
    - **Snap Decision Tree Classifier**
    - **Python Runtime (auto-managed by IBM)**
    - **Model Enhancements:**
      - Feature Engineering (FE)
      - Hyperparameter Optimization (HPO)
- **Note:** *Manual Python coding was not required as AutoAI handled all preprocessing and model building internally.*

ALGORITHM & DEPLOYMENT

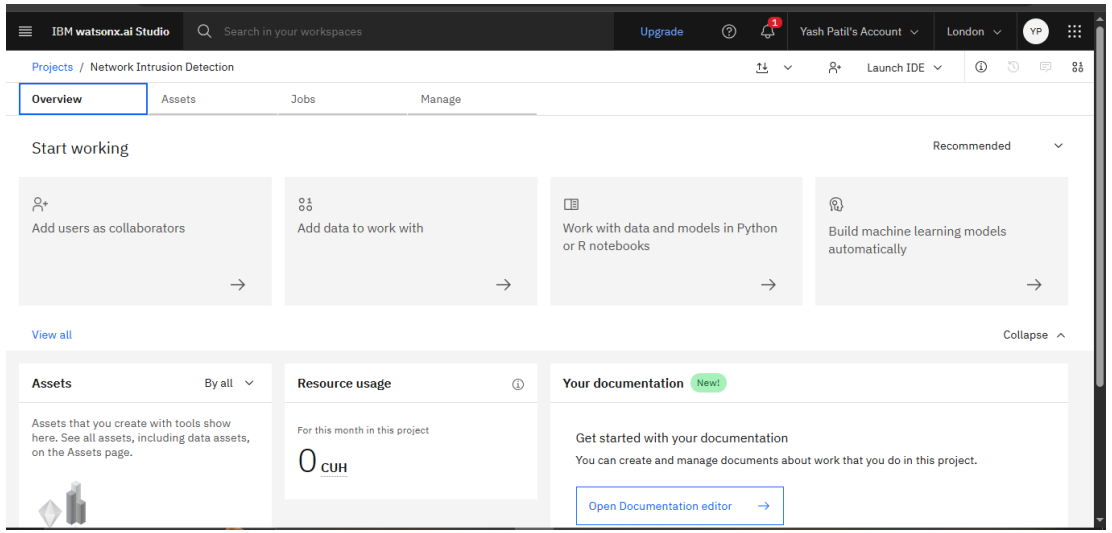
- **Algorithm Selection:**
  - Chosen Algorithm: **Snap Decision Tree Classifier**
  - Type: **Supervised Classification**
  - Justification:
    - Fast training and prediction time
    - High accuracy and interpretability
    - Works well on structured datasets like intrusion detection logs
  - AutoAI tested multiple algorithms; this model outperformed others during evaluation
- **Data Input:**
  - The input dataset included **41 features**, such as:
    - duration, protocol_type, service, flag
    - src_bytes, dst_bytes, wrong_fragment, hot, etc.
    - Target column: class (normal or attack)
  - IBM Watsonx AutoAI handled:
    - Feature selection and encoding
    - Automatic data transformation

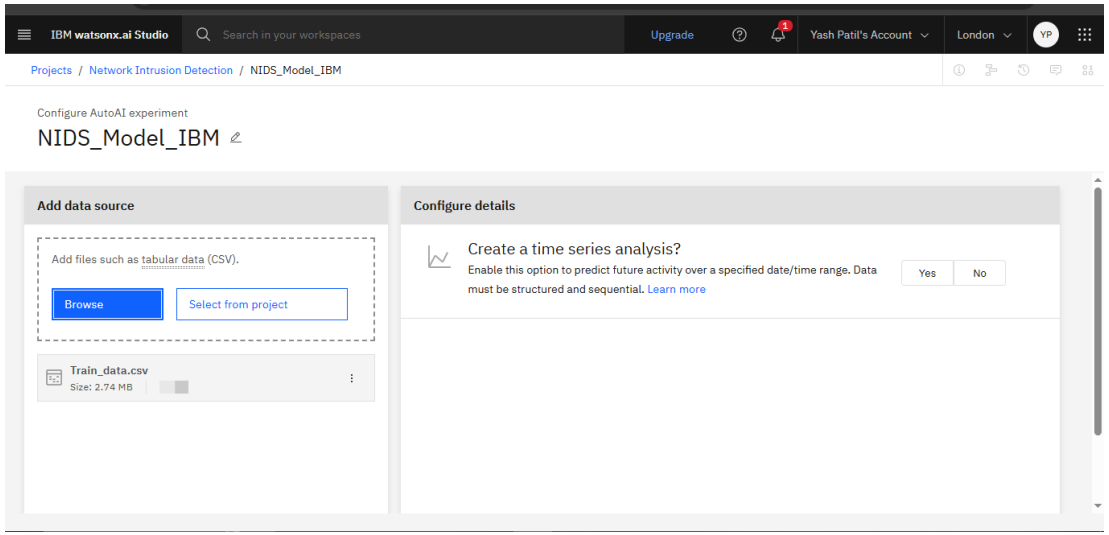
ALGORITHM & DEPLOYMENT

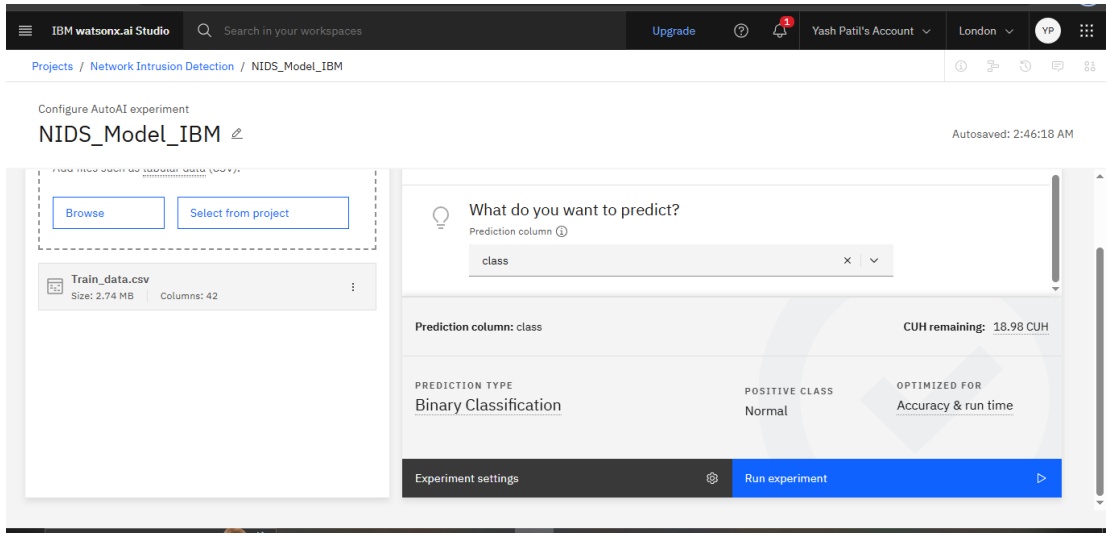
- **Training Process:**
  - Dataset uploaded to IBM Watsonx.ai
  - AutoAI split data into training and hold-out sets
  - Multiple pipelines generated with:
    - Feature Engineering (FE)
    - Hyperparameter Optimization (HPO1, HPO2)
  - Best pipeline selected using **cross-validation accuracy**

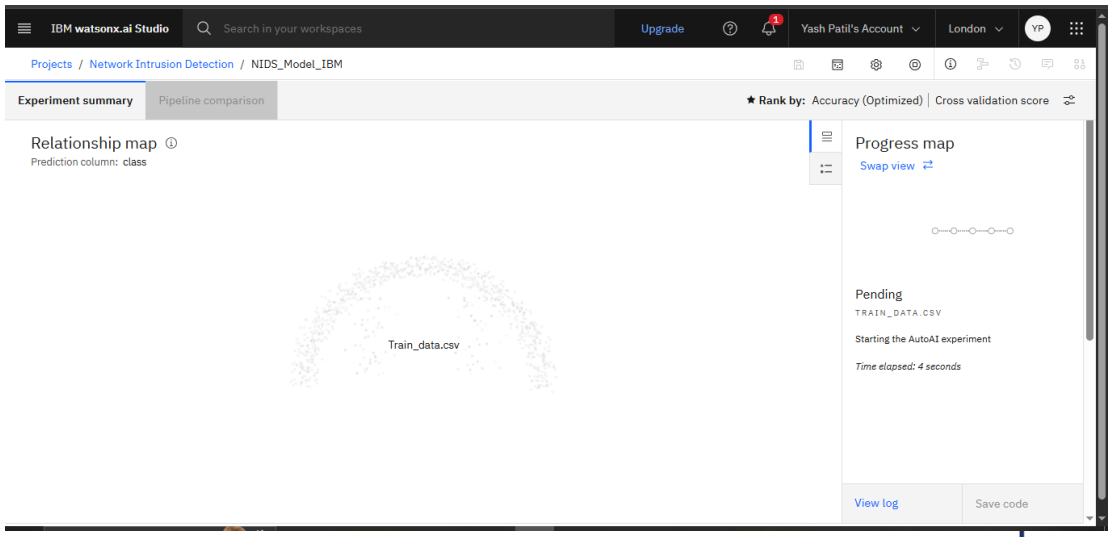
- **Prediction Process:**
  - Model deployed as an **online REST API** via IBM Cloud
  - Prediction made by sending **JSON-formatted inputs** with 41 features
  - Output: Predicted network status (normal or attack)
  - Real-time testing done via Watsonx.ai Test UI

RESULT



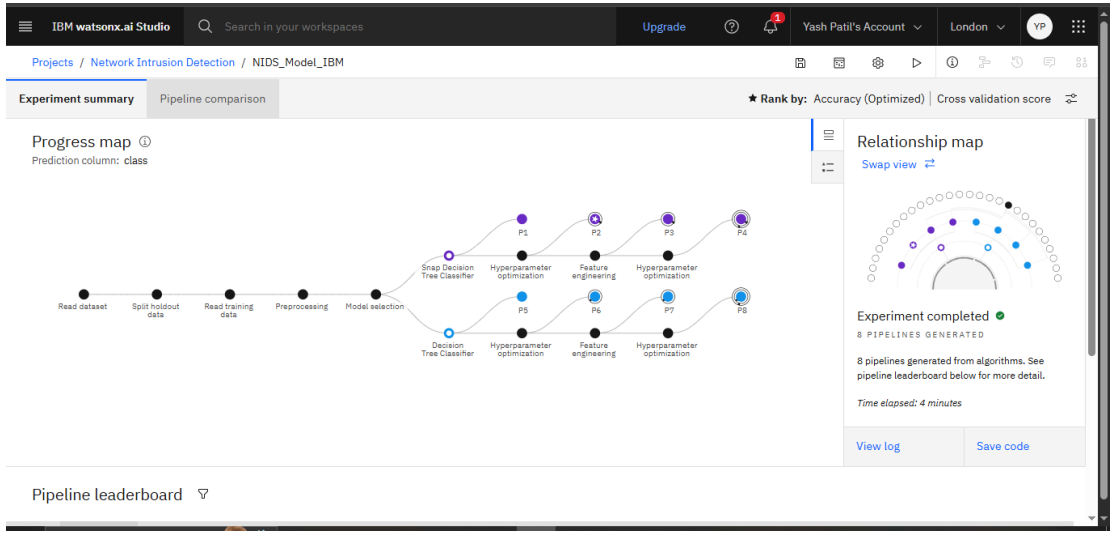


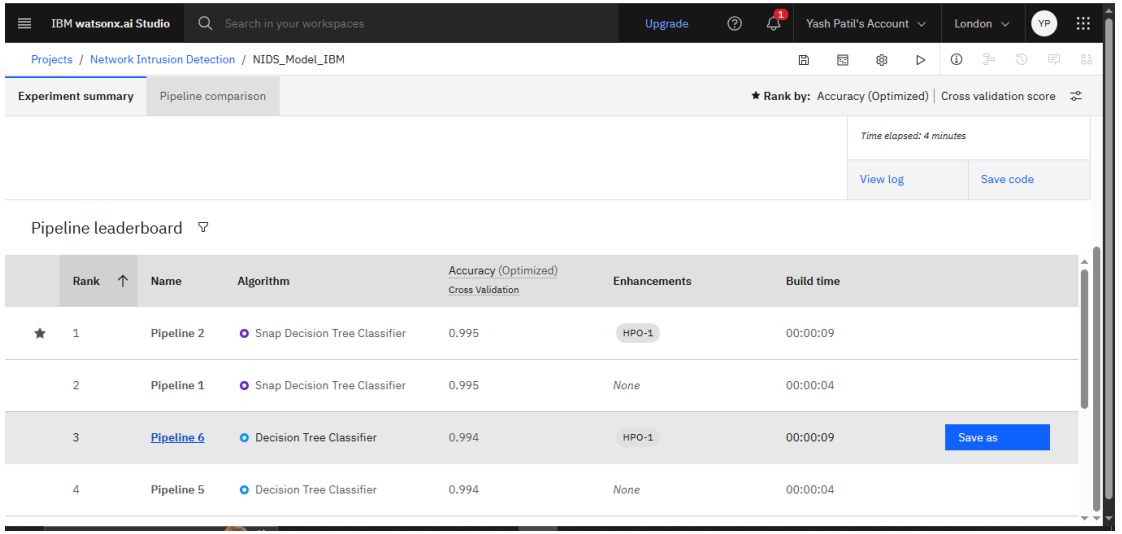


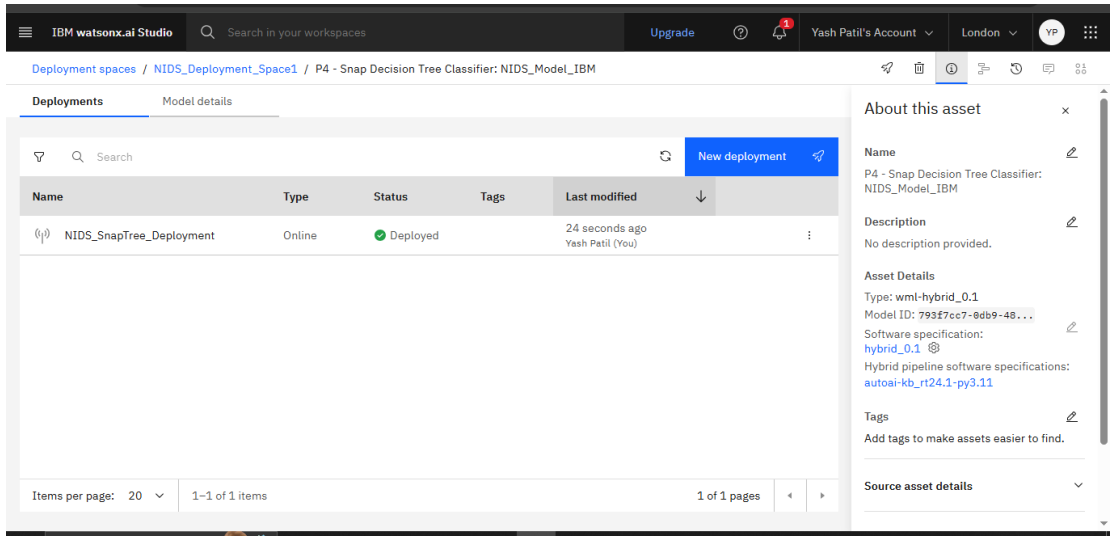


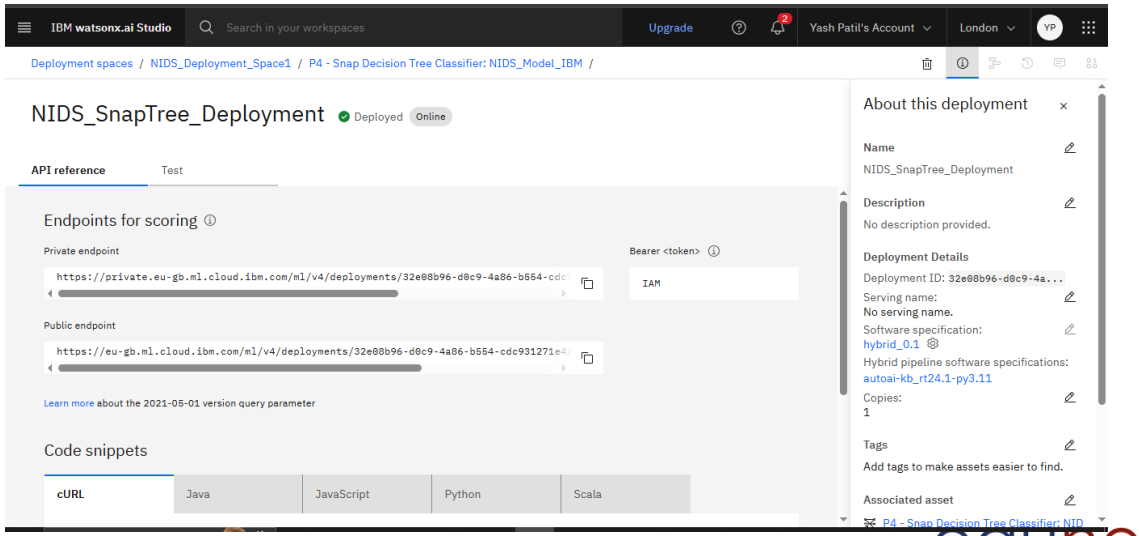
edunet foundation

RESULT



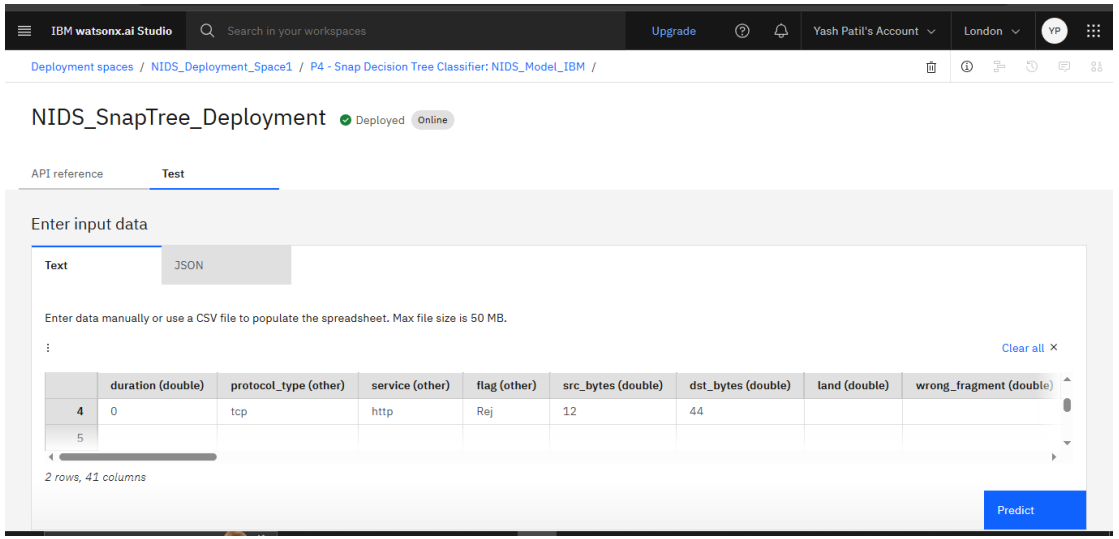


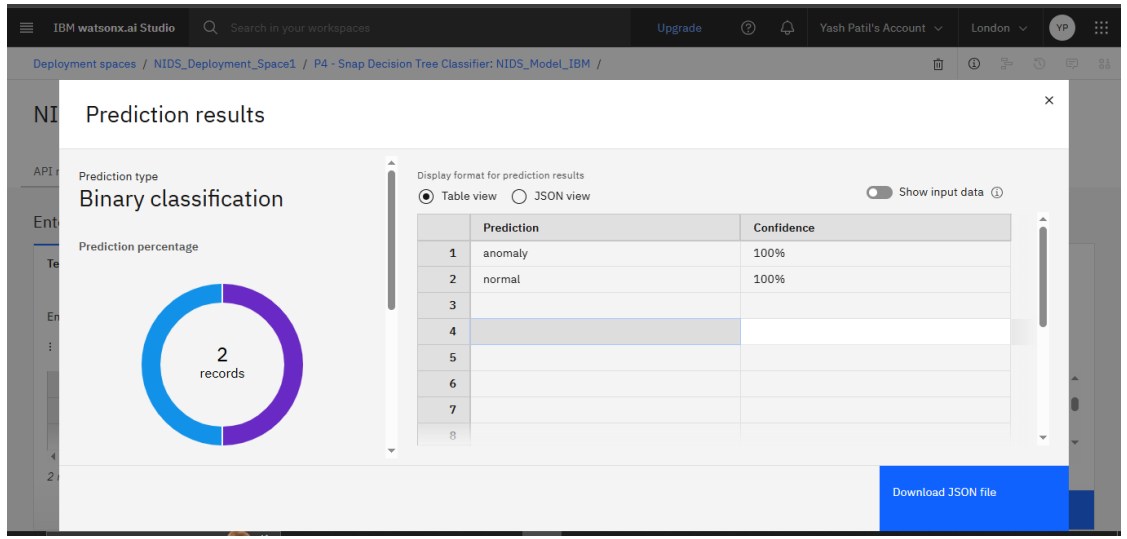


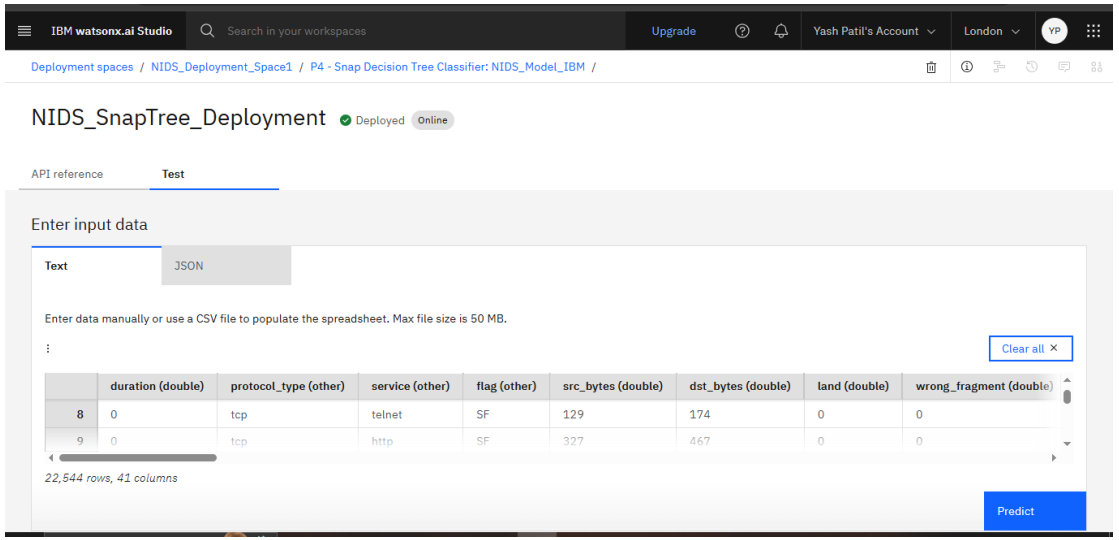


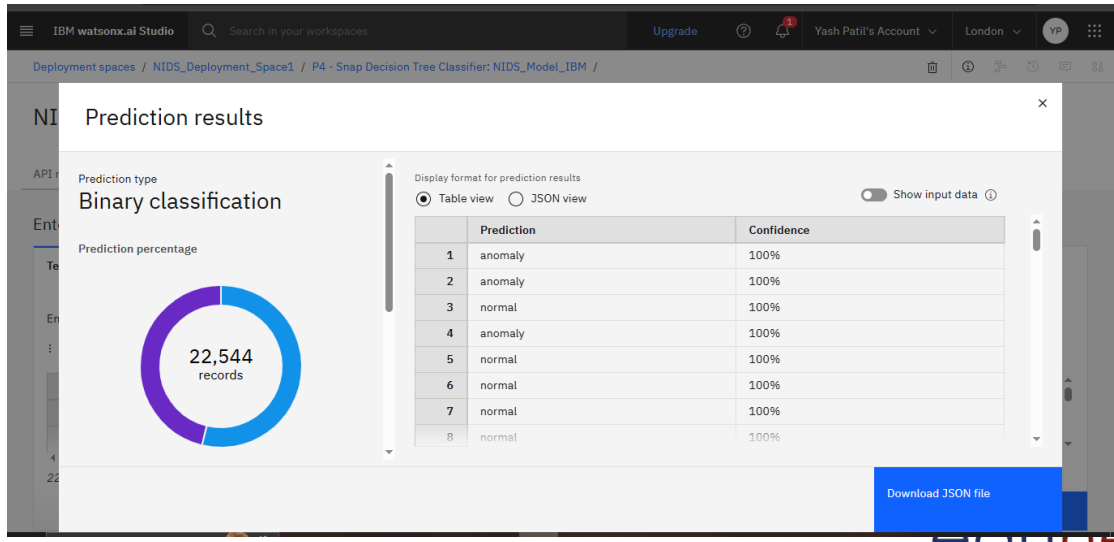
edunet foundation

RESULT


IBM watsonx.ai Studio
Search in your workspaces
Upgrade
Yash Patil's Account
London
Deployment spaces / NIDS_Deployment_Space1 / P4 - Snap Decision Tree Classifier: NIDS_Model_IBM /
NIDS_SnapTree_Deployment
Deployed
Online
API reference
Test
Enter input data
Text
JSON
Enter data manually or use a CSV file to populate the spreadsheet. Max file size is 50 MB.
Clear all
duration (double) | protocol_type (other) | service (other) | flag (other) | src_bytes (double) | dst_bytes (double) | land (double) | wrong_fragment (double)
4 | 0 | tcp | http | Rej | 12 | 44
5
2 rows, 41 columns
Predict


IBM watsonx.ai Studio
Deployment spaces / NIDS_Deployment_Space1 / P4 - Snap Decision Tree Classifier: NIDS_Model_IBM /
Prediction results
Prediction type
Binary classification
Prediction percentage
2
records
Display format for prediction results
Table view
JSON view
Show input data
Prediction | Confidence
1 | anomaly | 100%
2 | normal | 100%
3
4
5
6
7
Download JSON file


IBM watsonx.ai Studio
Search in your workspaces
Upgrade
Yash Patil's Account
London
Deployment spaces / NIDS_Deployment_Space1 / P4 - Snap Decision Tree Classifier: NIDS_Model_IBM /
NIDS_SnapTree_Deployment
Deployed
Online
API reference
Test
Enter input data
Text
JSON
Enter data manually or use a CSV file to populate the spreadsheet. Max file size is 50 MB.
Clear all
duration (double) | protocol_type (other) | service (other) | flag (other) | src_bytes (double) | dst_bytes (double) | land (double) | wrong_fragment (double)
8 | 0 | tcp | telnet | SF | 129 | 174 | 0 | 0
9 | 0 | tcp | http | SF | 327 | 467 | 0 | 0
22,544 rows, 41 columns
Predict


IBM watsonx.ai Studio
Deployment spaces / NIDS_Deployment_Space1 / P4 - Snap Decision Tree Classifier: NIDS_Model_IBM /
Prediction results
Prediction type
Binary classification
Prediction percentage
22,544
records
Display format for prediction results
Table view
JSON view
Show input data
Prediction | Confidence
1 | anomaly | 100%
2 | anomaly | 100%
3 | normal | 100%
4 | anomaly | 100%
5 | normal | 100%
6 | normal | 100%
7 | normal | 100%
Download JSON file

CONCLUSION

- The Network Intrusion Detection System developed using IBM Watsonx.ai successfully demonstrates the use of machine learning to classify network traffic as normal or intrusive. By leveraging AutoAI, the Snap Decision Tree Classifier was trained with automatic feature engineering and hyperparameter optimization, resulting in a high-performing model deployed as a REST API. The system enables real-time predictions and provides a scalable solution for network security. Challenges included understanding complex network traffic patterns and configuring deployment inputs. Future improvements could involve multiclass classification, real-time data integration, and a user interface for monitoring. Overall, the solution proves effective in enhancing cybersecurity through intelligent threat detection.

FUTURE SCOPE

- **Incorporate real-time data:**
  Integrate live network traffic and external threat intelligence feeds for dynamic and up-to-date intrusion detection.
- **Advanced machine learning models:**
  Explore ensemble models, deep learning (e.g., LSTM, CNN), or anomaly detection techniques to improve classification accuracy.
- **Scalability across regions:**
  Expand the system to monitor multiple networks across cities or enterprise branches for broader security coverage.
- **Edge computing integration:**
  Deploy NIDS on edge devices to enable faster, decentralized threat detection with reduced response time.
- **Continuous learning:**
  Implement online learning techniques for the model to adapt to new and evolving cyber-attack patterns.
- **User interface enhancement:**
  Add dashboards or real-time alerting systems to support security teams with actionable insights.

REFERENCES

1. Kaggle Dataset – Network Intrusion Detection
   https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection
2. IBM Watsonx.ai – AutoAI
   https://www.ibm.com/cloud/watsonx/autoai
3. IBM Cloud Deployment Space Documentation
   https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=models-deploying

IBM CERTIFICATIONS

- Screenshot/ credly certificate(getting started with AI)

IBM CERTIFICATIONS

- Screenshot/ credly certificate(Journey to Cloud)

IBM CERTIFICATIONS

- Screenshot/ certificate(RAG Lab)

IBM **SkillsBuild** Completion Certificate

This certificate is presented to

YASH PATIL

for the completion of

Lab: Retrieval Augmented Generation with LangChain

(ALM-COURSE_3824998)

According to the Adobe Learning Manager system of record

Completion date: 15 Jul 2025 (GMT)

Learning hours: 20 mins

THANK YOU

edunet foundation