# CAPSTONE PROJECT

# POWER SYSTEM FAULT DETECTION AND CLASSIFICATION


**Presented By:**

**Punit Verma-Guru Jambheshwar University of Science and Technology-Computer Science and Engineering**


edunet foundation

# OUTLINE

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

# PROBLEM STATEMENT

The goal is to design a machine learning model that can identify and classify different types of faults in a power distribution network using voltage and current phasor data. By distinguishing between normal operations and faults such as line-to-ground, line-to-line, and three-phase, the model will support faster fault detection and improve the overall stability and reliability of the power grid.

GitHub Link :- https://github.com/PunitVerma0009/IBM-Cloud-Project

# PROPOSED SOLUTION

- The proposed system addresses the challenge of detecting and classifying faults in a power distribution system using machine learning techniques. The solution leverages electrical measurement data to accurately identify fault types and support rapid grid response. The following components make up the solution:

- **Data Collection:**
  - Used a publicly available dataset from Kaggle consisting of voltage and current phasor readings under various conditions.
  - Focused on identifying different fault types such as line-to-ground, line-to-line, and three-phase faults.

- **Data Preprocessing:**
  - Removed the 'Fault ID' column to avoid data leakage during training.
  - Cleaned the dataset by standardizing features to prepare for model training.

- **Machine Learning Algorithm:**
  - Implemented a Snap Logistic Regression model to classify the system state as normal or one of several fault types.
  - The model was trained using 506 rows of cleaned data.

- **Deployment:**
  - Developed and deployed the model entirely on IBM Cloud for accessible and scalable use.
  - Ensured the system supports cloud-based testing and future integration with real-time data sources.

- **Evaluation:**
  - The model achieved an accuracy of **41%**, limited by the small dataset size.
  - Performance can be improved with a larger and more balanced dataset.

# SYSTEM APPROACH

- The "System Approach" section outlines the strategy and resources used to design and implement the fault detection model in a power distribution system. The system leverages cloud-based resources and essential Python libraries to build, train, and evaluate the model.
- **System Requirements:**
  - **IBM Cloud environment** with **8 CPUs** and **32 GB RAM** for training and deployment.
  - High-performance computing resources were essential due to the data preprocessing and model training phases.
- **Libraries Required to Build the Model:**
  - **Pandas** – for data loading, cleaning, and manipulation.
  - **NumPy** – for numerical operations.
  - **Scikit-learn** – for implementing Snap Logistic Regression and evaluation metrics.
  - **Matplotlib / Seaborn** – for visualizing fault distributions and model performance.
  - **Joblib / Pickle** – for saving and deploying the trained model.

# ALGORITHM & DEPLOYMENT

- In this section, we describe the machine learning algorithm used to detect and classify faults in the power distribution system.

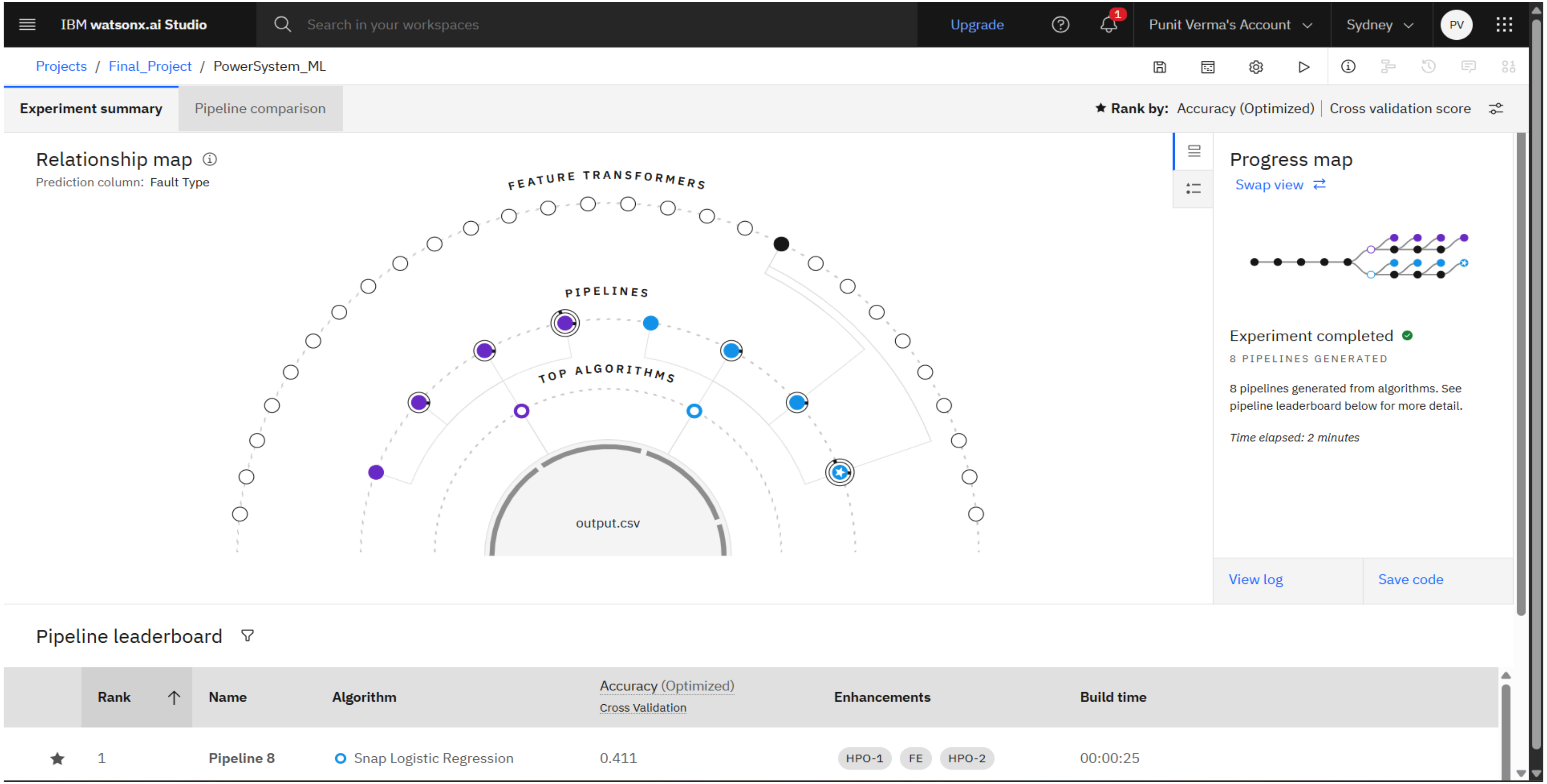
- **Algorithm Selection**:
  - **Snap Logistic Regression** was chosen due to its simplicity, interpretability, and ability to handle binary and multiclass classification. It is suitable for small datasets and performs well in identifying relationships between electrical measurements and fault types.

- **Data Input**:
  - The model uses features derived from **voltage and current phasors** under different system conditions.
  - The target variable indicates whether the system is operating normally or experiencing a specific type of fault (e.g., line-to-ground, line-to-line, three-phase).

- **Training Process**:
  - The dataset was cleaned and preprocessed, including removal of the Fault ID column to avoid label leakage.
  - The model was trained on 506 rows of data, using IBM Cloud resources (8 CPUs, 32 GB RAM).
  - Given the limited data, the model was trained using default parameters without hyperparameter tuning to avoid overfitting.

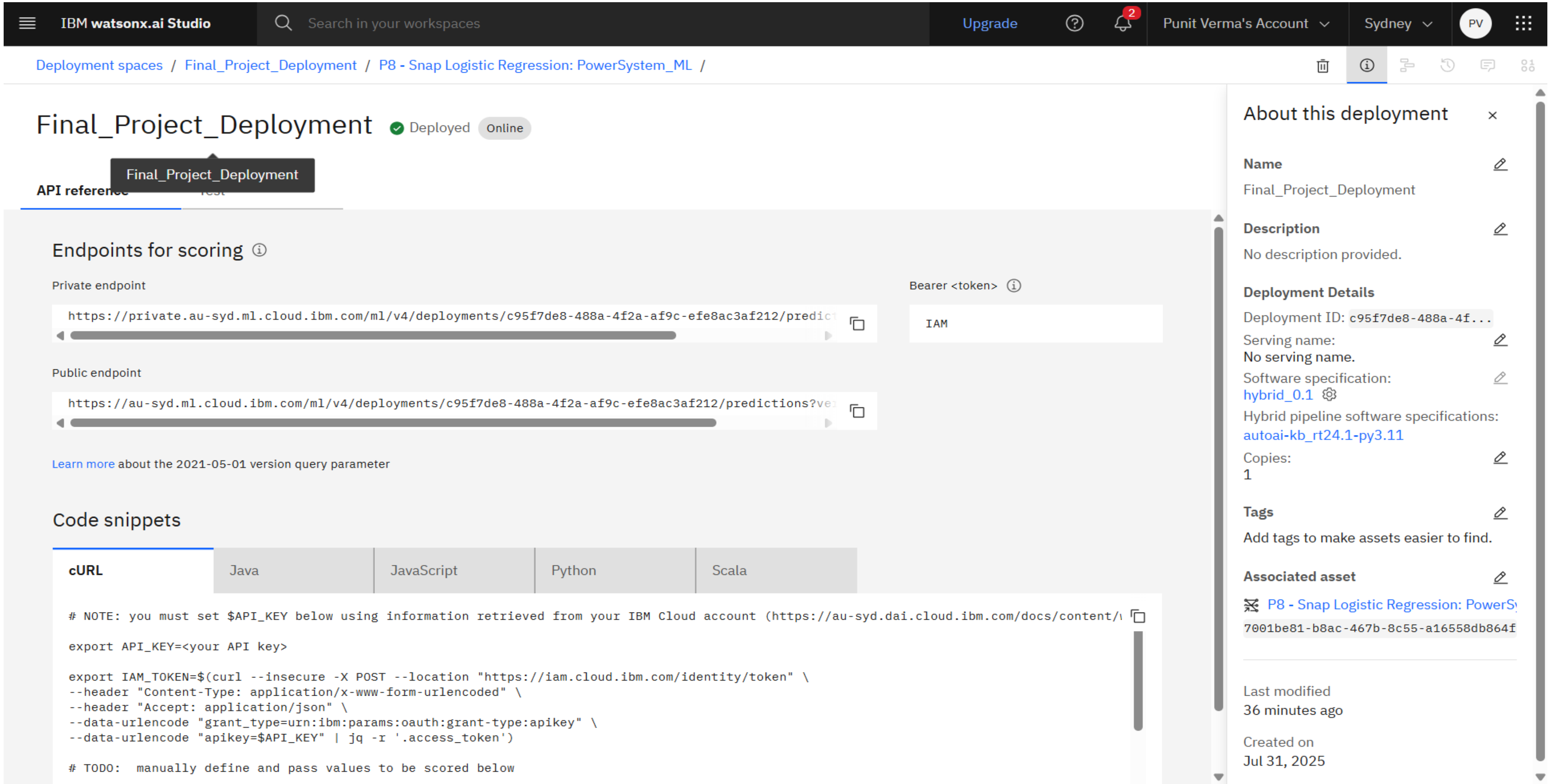
- **Prediction Process**:
  - Once trained, the model takes new phasor readings as input and classifies them into one of the fault categories or normal operation.
  - The model is deployed on IBM Cloud and can be integrated into a real-time system for live fault monitoring and prediction.

# RESULT




edunet foundation

# RESULT


IBM watsonx.ai Studio
Search in your workspaces
Upgrade
Punit Verma's Account
Sydney
PV
Deployment spaces / Final_Project_Deployment / P8 - Snap Logistic Regression: PowerSystem_ML /
Final_Project_Deployment
Deployed
Online
Final_Project_Deployment
API reference
Endpoints for scoring
Private endpoint
https://private.au-syd.ml.cloud.ibm.com/ml/v4/deployments/c95f7de8-488a-4f2a-af9c-efe8ac3af212/predic
Bearer <token>
IAM
Public endpoint
https://au-syd.ml.cloud.ibm.com/ml/v4/deployments/c95f7de8-488a-4f2a-af9c-efe8ac3af212/predictions?ve
Learn more about the 2021-05-01 version query parameter
Code snippets
cURL
Java
JavaScript
Python
Scala
# NOTE: you must set $API_KEY below using information retrieved from your IBM Cloud account (https://au-syd.dai.cloud.ibm.com/docs/content/
export API_KEY=<your API key>
export IAM_TOKEN=$(curl --insecure -X POST --location "https://iam.cloud.ibm.com/identity/token" \
--header "Content-Type: application/x-www-form-urlencoded" \
--header "Accept: application/json" \
--data-urlencode "grant_type=urn:ibm:params:oauth:grant-type:apikey" \
--data-urlencode "apikey=$API_KEY" | jq -r '.access_token')
# TODO: manually define and pass values to be scored below
About this deployment
Name
Final_Project_Deployment
Description
No description provided.
Deployment Details
Deployment ID: c95f7de8-488a-4f...
Serving name:
No serving name.
Software specification:
hybrid_0.1
Hybrid pipeline software specifications:
autoai-kb_rt24.1-py3.11
Copies:
1
Tags
Add tags to make assets easier to find.
Associated asset
P8 - Snap Logistic Regression: PowerSy
7001be81-b8ac-467b-8c55-a16558db864f
Last modified
36 minutes ago
Created on
Jul 31, 2025

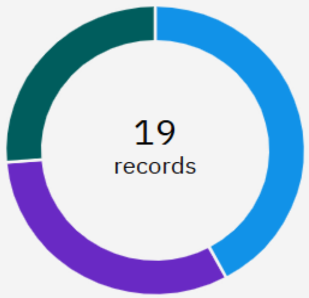
# RESULT

## Prediction results

Prediction type

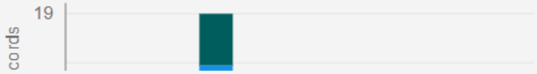
**Multiclass classification**

**Prediction percentage**




Transformer Failure | Line Breakage | Overheating

**Confidence level distribution**




Display format for prediction results

(•) Table view ( ) JSON view

Show input data

| | Prediction | Confidence |
|---|---|---|
| 1 | Transformer Failure | 36% |
| 2 | Line Breakage | 36% |
| 3 | Line Breakage | 36% |
| 4 | Transformer Failure | 34% |
| 5 | Line Breakage | 38% |
| 6 | Overheating | 37% |
| 7 | Line Breakage | 38% |
| 8 | Overheating | 38% |
| 9 | Transformer Failure | 36% |
| 10 | Line Breakage | 36% |
| 11 | Overheating | 39% |
| 12 | Transformer Failure | 35% |
| 13 | Line Breakage | 36% |
| 14 | Overheating | 37% |

Download JSON file

edunet foundation

# CONCLUSION

The proposed machine learning-based solution successfully demonstrates the ability to detect and classify faults in a power distribution system using electrical measurement data. By employing Snap Logistic Regression, the model was able to classify various fault types based on voltage and current phasor inputs. The entire model was developed and deployed on IBM Cloud, leveraging a computing environment with 8 CPUs and 32 GB RAM.

- **Effectiveness of the Solution:**
  - The model achieved an accuracy of **41%**, which reflects moderate performance due to the **limited dataset size** (only 506 records).
  - It provides a proof of concept that fault classification can be automated using machine learning with appropriate data and system setup.

- **Challenges Encountered:**
  - **Insufficient data volume** made it difficult for the model to generalize well across fault types.
  - Lack of feature variety or real-time input sources may have limited the learning potential of the algorithm.
  - Removing the Fault ID to avoid leakage also reduced the number of directly interpretable labels.

- **Importance of Accurate Predictions:**

  Accurate fault detection is essential for minimizing downtime, preventing equipment damage, and ensuring power grid reliability. Even a small improvement in fault classification accuracy can lead to faster response times and improved operational efficiency in real-world power systems.

# FUTURE SCOPE

**Potential Improvements:**

- **Increasing dataset size** with more real-time fault scenarios would significantly enhance prediction accuracy.
- Exploring **advanced models** like Random Forest, SVM, or Neural Networks could yield better performance.
- Incorporating additional features such as phase angles, frequency deviations, and time-based indicators could improve model sensitivity.

# REFERENCES


1. **Kaggle Dataset**
   *"Power System Fault Detection Dataset."*
   Retrieved from: https://www.kaggle.com/
   *(Used as the main dataset for training and evaluating the model.)*
2. **IBM Cloud Platform**
   *Practical training received during internship on building and deploying machine learning models using IBM Cloud's AutoAI and cloud-based services.*
   *(Used for model creation, resource allocation, and deployment.)*

# IBM CERTIFICATIONS

In recognition of the commitment to achieve professional excellence


## Punit Verma

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/840400a7-38eb-4c6c-b940-23d9a32cb187

IBM

# IBM CERTIFICATIONS

Journey to Cloud: Envisioning Your Solution
IBM SkillsBuild

In recognition of the commitment to achieve professional excellence

## Punit Verma

Has successfully satisfied the requirements for:

### Journey to Cloud: Envisioning Your Solution

Issued on: Jul 17, 2025

Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/d8008135-69f2-4ad4-a746-ce57a3324c07

IBM

# IBM CERTIFICATIONS

IBM **SkillsBuild** Completion Certificate

This certificate is presented to

Punit Verma

for the completion of

**Lab: Retrieval Augmented Generation with LangChain**

(ALM-COURSE_3824998)

According to the Adobe Learning Manager system of record

**Completion date:** 23 Jul 2025 (GMT)

**Learning hours:** 20 mins

# THANK YOU

GITHUB LINK :- HTTPS://GITHUB.COM/PUNITVERMA0009/IBM-CLOUD-PROJECT