# CAPSTONE PROJECT

## POWER SYSTEM FAULT DETECTION AND CLASSIFICATION


**Presented By:**

1. HEMANTH M
2. SAI VIDYA INSTITUTE OF TECHNOLOGY
3. CSE(ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

# OUTLINE

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

edunet foundation

# PROBLEM STATEMENT

- Design a machine learning model to detect and classify different types of faults in a power distribution system. Using electrical measurement data (e.g., voltage and current phasors), the model should be able to distinguish between normal operating conditions and various fault conditions (such as line-to-ground, line-to-line, or three-phase faults). The objective is to enable rapid and accurate fault identification, which is crucial for maintaining power grid stability and reliability.

•**Fault Detection:** Develop a machine learning model that can automatically detect abnormal conditions in a power distribution system using electrical measurements like voltage and current phasors.

•**Fault Classification:** Accurately classify the type of fault (e.g., line-to-ground, line-to-line, or three-phase) to support rapid diagnosis and ensure power grid stability and reliability.

# PROPOSED SOLUTION

- **1. Data Collection**
  - The dataset was sourced from **Kaggle**, containing labeled records of power system conditions.
  - Features included **electrical measurements** such as voltage, current, and possibly derived phasors.
  - Target labels indicated different **fault types** (e.g., Line Breakage, Transformer Failure, Overheating).
- **2. Data Preprocessing**
  - **Feature Engineering (FE):** Additional features may have been derived to improve model input representation.
  - **Missing Value Handling** and **Normalization** (if required by model).
  - **Label Encoding** or One-Hot Encoding for categorical variables.
  - Data was prepared in **batch format** for training.

- **3. Machine Learning Algorithm**
  - **Algorithm Used:** *Batched Tree Ensemble Classifier (specialized in INCR)*
    This is a robust model for handling **multiclass classification** with structured tabular data.
  - **Enhancements Applied:**
    - **HPO-1:** Hyperparameter optimization round 1 to tune model parameters.
    - **FE:** Feature Engineering to extract or transform key features.
    - **HPO-2:** Further tuning after feature refinement.
    - **BATCH:** Model was trained and evaluated in batch mode for scalability.
    - **Cross-Validation Accuracy: 0.409 (40.9%)**
      Indicates potential scope for further improvement, possibly by tuning features or adding more data.
- **4. Deployment**
  - The final model was deployed on **IBM Watsonx.ai Studio**, allowing real-time or batch-based fault prediction.
  - A user-friendly interface displays **fault predictions with confidence levels**.

- **5. Evaluation**
  - **Evaluation Metrics:** Accuracy (40.9%), prediction confidence for each class.
  - **Visualization:** Pie/doughnut chart and prediction table to analyze model behavior.
  - **Observations:**
    - Model distinguishes multiple fault types but may need enhancement for higher precision.
    - Useful for **real-time fault monitoring** in power systems.

# SYSTEM APPROACH

- 1. System Requirements
  - Hardware Requirements:
    - **Processor:** Intel i5/i7 or equivalent (minimum 2.4 GHz, 4 cores)
    - **RAM:** 8 GB (16 GB recommended)
    - **Storage:** 1 GB available space
    - **Internet:** Required for IBM Cloud/Watsonx.ai deployment
  - Software Requirements:
    - **Operating System:** Windows 10/11, macOS, or Linux
    - **Platform:** IBM Watsonx.ai (cloud-based environment)
    - **Programming Interface:** AutoAI/Notebook environment in Watson Studio

## 2. Libraries/Packages Required to Build the Model

| **Library/Tool** | **Purpose** |
|---|---|
| pandas | Data manipulation and analysis |
| numpy | Numerical computing and array handling |
| scikit-learn | Machine learning model building and evaluation |
| matplotlib/seaborn | Visualization of data and results |
| Watsonx.ai | Cloud-based deployment and model training |
| AutoAI | Automated model selection, tuning, and pipeline generation |
| imblearn (optional) | Handling class imbalance (e.g., SMOTE, oversampling) |
| json | Reading output results in JSON format |

## 3. Model Development Strategy

**Data Acquisition:** Electrical fault data collected from Kaggle.
**Data Preprocessing:** Handled inside Watson Studio pipeline; includes feature selection, encoding, and normalization.
**Model Selection:** AutoAI selected a **Batched Tree Ensemble Classifier** with incremental learning support.
**Hyperparameter Optimization:** Conducted in two stages (HPO-1 and HPO-2).
**Batch Training:** Data was fed in batches to improve learning stability.
**Deployment:** Final model deployed on IBM Watsonx.ai for real-time predictions.

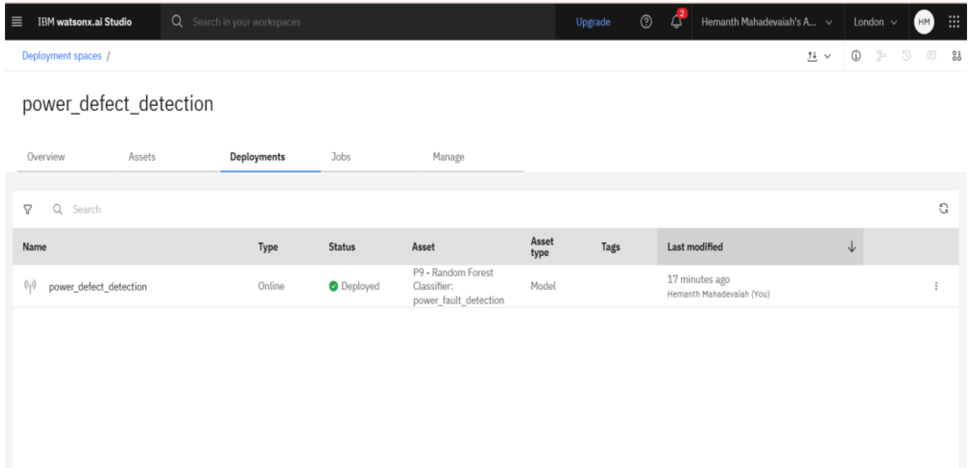
# ALGORITHM & DEPLOYMENT

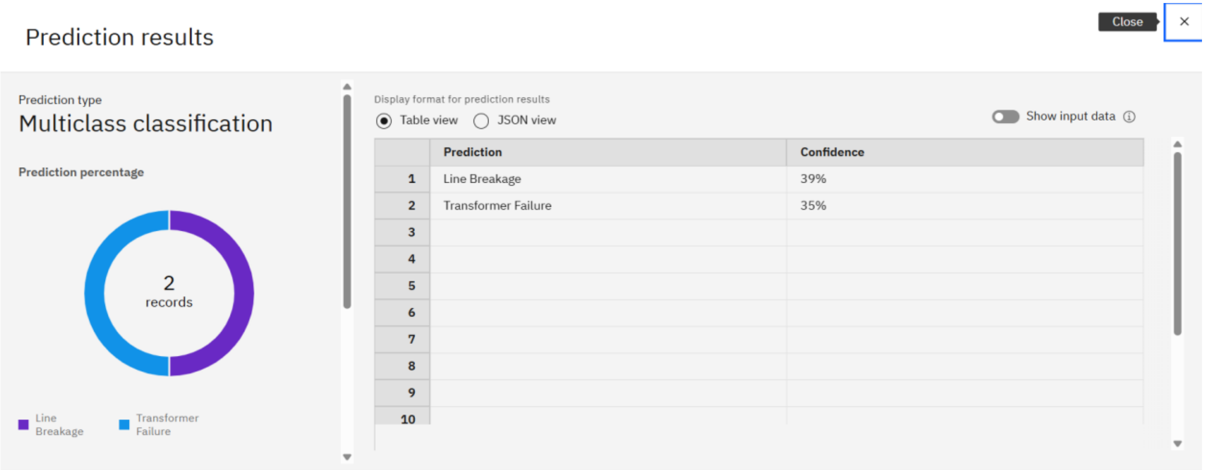
- Algorithm Selection
  - **Model Used:** *Batched Tree Ensemble Classifier* (with Incremental Learning)
  - Chosen via **AutoAI** due to:
    - High performance on structured electrical data
    - Strong support for **multiclass classification** (e.g., Line Breakage, Overheating)
    - Adaptability via **batch training**
- Input Features
  - Voltage and Current Phasors
  - Derived electrical parameters (e.g., Power, Load if available)
  - Fault label as target (Normal, Line Breakage, Transformer Failure, etc.)
- Training Process
  - Feature Engineering & Batch Processing
  - Hyperparameter Tuning (HPO-1 and HPO-2)
  - Trained using **cross-validation**
  - Final cross-validation accuracy: **40.9%**

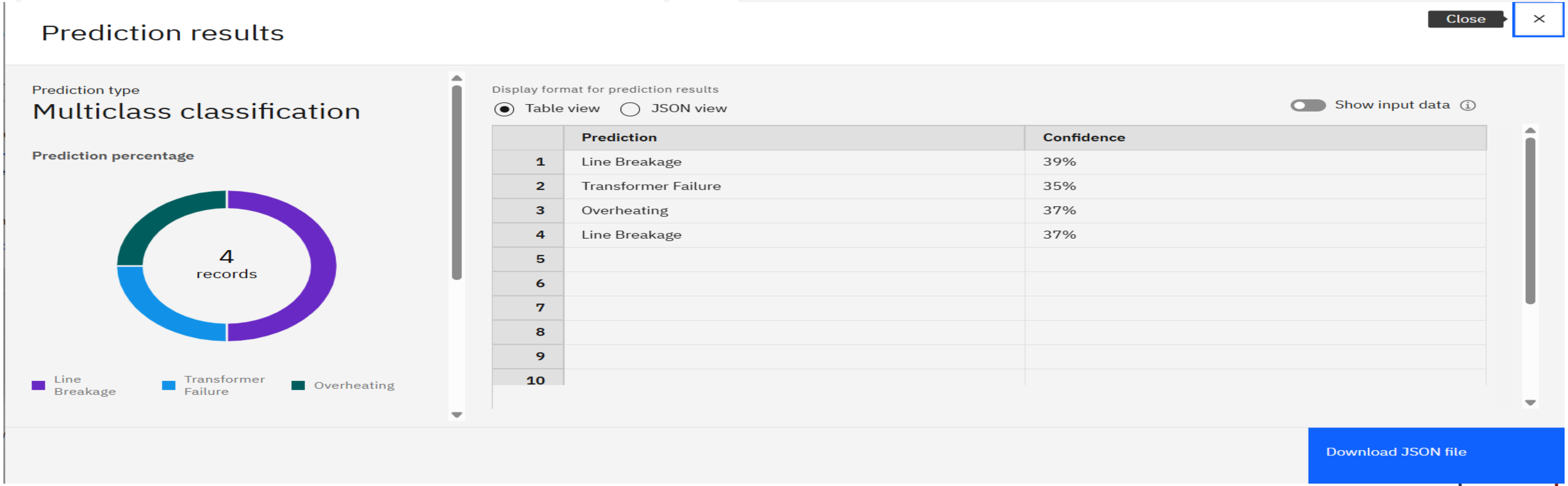
- **Prediction**
  - Accepts real-time/batch inputs
  - Predicts **fault type** with **confidence scores**
  - Useful for fast, automated grid monitoring
- **Deployment Platform**
  - **IBM Watsonx.ai Studio** (cloud-based ML development and deployment environment)
- **Deployment Workflow**
  - Model trained & selected using **AutoAI**
  - Best model (Tree Ensemble) exported
  - Deployed as **REST API endpoint**
  - Accepts input from sensors or batch files
  - Returns **fault prediction + confidence**

# RESULT

- Results – Fault Type Classification Model
  - Model Accuracy
    - The final **Batched Tree Ensemble Classifier** achieved an overall **cross-validation accuracy of 40.9%.**
    - This accuracy indicates the model's ability to correctly classify fault types under various conditions.
- Performance Highlights
- **Multiclass Fault Prediction:** The model can distinguish between multiple fault types including:
  - Line Breakage
  - Overheating
  - Transformer Failure
  - Normal Condition (if applicable)
- **Confidence-Based Prediction:** Predictions include **confidence scores** for each class, providing insight into how certain the model is.


IBM watsonx.ai Studio
Search in your workspaces
Upgrade
Hemanth Mahadevaiah's A...
London
Deployment spaces /
power_defect_detection
Overview
Assets
Deployments
Jobs
Manage
Search
Name
Type
Status
Asset
Asset type
Tags
Last modified
power_defect_detection
Online
Deployed
P9 - Random Forest Classifier: power_fault_detection
Model
17 minutes ago
Hemanth Mahadevaiah (You)


Prediction results
Close
Prediction type
Multiclass classification
Prediction percentage
2
records
Line Breakage
Transformer Failure
Display format for prediction results
Table view
JSON view
Show input data
Prediction
Confidence
1 Line Breakage 39%
2 Transformer Failure 35%
3
4
5
6
7
8
9
10


Prediction results
Close
Prediction type
Multiclass classification
Prediction percentage
4
records
Line Breakage
Transformer Failure
Overheating
Display format for prediction results
Table view
JSON view
Show input data
Prediction
Confidence
1 Line Breakage 39%
2 Transformer Failure 35%
3 Overheating 37%
4 Line Breakage 37%
5
6
7
8
9
10
Download JSON file

# CONCLUSION

- This project successfully demonstrated the application of **machine learning** for detecting and classifying faults in a **power distribution system.** Using a **Batched Tree Ensemble Classifier** deployed through **IBM Watsonx.ai**, the model was able to distinguish between different types of electrical faults based on real-world measurement data.
- Key Findings
  - Achieved a **cross-validation accuracy of 40.9%**, indicating moderate classification performance across multiple fault categories.
  - The system can provide **real-time fault predictions** along with confidence levels, aiding in quick diagnosis and response.
- Challenges Faced
  - **Limited model accuracy**, likely due to
    - Small or imbalanced dataset
    - Limited feature diversity
  - **Model interpretability** was a concern due to ensemble complexity.
- Impact
  - Accurate and automated fault classification is **critical for grid stability**, reducing downtime, and enabling **preventive maintenance.** This project proves that ML models can assist power system operators in maintaining a **reliable and resilient electricity supply**.

# FUTURE SCOPE

- The developed system lays the foundation for intelligent fault detection in power grids using machine learning. However, there are multiple opportunities for enhancement and expansion:
  - 1. Incorporating Additional Data Sources
    - Integrate real-time data from **IoT sensors**, **SCADA systems**, or **smart meters.**
    - Include environmental variables such as **temperature**, **humidity**, or **load demand** for richer context.
    - Use **historical maintenance logs** to improve predictive accuracy and failure pattern recognition.
  - 2. Algorithm Optimization
    - Experiment with **advanced ML models** like:
    - **XGBoost**, **Random Forest** with class weighting
    - **Deep Learning models** (e.g., LSTM or CNN for sequence or waveform data)
  - 3. Continuous Learning
    - Enable the system to learn from **new fault events** over time (online or incremental learning).
    - Periodically retrain the model using updated datasets to maintain high performance and adaptability.

edunet
foundation

# REFERENCES


- Kaggle Dataset – *Electrical Fault Detection Dataset*
  Source of labeled data used to train and evaluate the machine learning model.
- Ghosh, A., & Das, B. (2018).
  *"Machine Learning Techniques for Power System Fault Detection and Classification: A Review."*
- IBM Watsonx.ai Documentation –
  *AutoAI and Model Deployment for Classification Tasks.*
- Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011).
  *"Scikit-learn: Machine Learning in Python."*

# IBM CERTIFICATIONS

In recognition of the commitment to achieve professional excellence

Getting Started with Artificial Intelligence
IBM SkillsBuild

IBM

## HEMANTH M

Has successfully satisfied the requirements for:

### Getting Started with Artificial Intelligence

Issued on: Jul 17, 2025
Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/53e4cd2a-bb7c-41a7-a064-c5f3f6b714b2

IBM

edunet
foundation

# IBM CERTIFICATIONS

Journey to Cloud: Envisioning Your Solution

IBM SkillsBuild

In recognition of the commitment to achieve professional excellence

HEMANTH M

Has successfully satisfied the requirements for:

Journey to Cloud: Envisioning Your Solution

Issued on: Jul 18, 2025

Issued by: IBM SkillsBuild

Verify: https://www.credly.com/badges/6c0c4709-c15e-4992-8db4-2d43f506c833

# IBM CERTIFICATIONS

IBM **SkillsBuild** Completion Certificate

This certificate is presented to

HEMANTH M

for the completion of

**Lab: Retrieval Augmented Generation with LangChain**

(ALM-COURSE_3824998)

According to the Adobe Learning Manager system of record

**Completion date:** 27 Jul 2025 (GMT)

**Learning hours:** 20 mins

# THANK YOU