

# Report: Active Learning for Power Grid Security Classification

---

**Author:** Gašper Leskovec

**Date:** May 14, 2025

## 1. Introduction

- Brief description of the problem (e.g., power grid security assessment)
  - Objective of the task: using active learning to improve binary classification efficiency
- 

## 2. Dataset Description

- **Source:** Dataset created using `prepare_dataset.py` by combining:
    - `distributed_generators.csv`
    - `distributed_loads_uniform.csv`
  - **Size:** 8769 rows × 273 columns
  - **Features:**
    - `timestamp`: simulation timestamp (not a real time series)
    - `status`: secure/insecure label
    - `max_line_loading_percent_*`: highest line loading
    - `min/max_bus_voltage_pu_*`: voltage limits
    - `load_*`, `gen_*`, `sgen_*`: active power (in MW) for loads, generators, and static generators
- 

## 3. Simulation Context

- Each row represents an **independent N-1 contingency simulation**
  - Simulations are **short-term synthetic events** (seconds to minutes)
  - **No real-world chronological time** — data is **not a time series**
- 

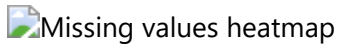
## 4. Exploratory Data Analysis (EDA)

### Class Distribution

- `secure`: 4497 samples
- `insecure`: 4272 samples

### Missing Values

- Only 3 features contain missing values:
  - `max_line_loading_percent_contingency`
  - `min_bus_voltage_pu_contingency`
  - `max_bus_voltage_pu_contingency`



For more detailed analysis and visualizations related to the dataset, refer to the full EDA report:

[EDA Report](#)

**Path:** `smart-energy-ml-analysis-jsi/reports/eda_classifier_report.md`

---

## 5. Data Preparation

Two data splitting approaches were used in the experiments:

### Manual Temporal Split (more frequent)

- First 90% of the dataset used as **training pool** (`X_pool`, `y_pool`)
- Last 10% used as the **validation set** (`X_val`, `y_val`)
- While the dataset has a `timestamp` column, it **does not reflect real-world temporal dependence**, so a manual chronological split was used for simulation consistency.

### Random Train/Test Split

- Used `train_test_split()` from scikit-learn with a 90/10 split ratio
  - Enabled **random shuffling**, suitable because data samples are independent simulation runs
- 

### Additional Notes

- **Label Encoding:**
    - Column `status` was converted to `status_binary`:  
`"secure" → 1`, `"insecure" → 0`
  - **Feature Selection:**
    - The following columns were removed:  
`timestamp`, `status`, `status_binary`, and contingency/basecase voltage/loading features.
  - **Normalization:**
    - Not applied, since **Random Forests** are insensitive to feature scales.
- 

## 6. Active Learning Setup

### Strategies Compared

- `random`
- `uncertainty sampling`
- `entropy-based`
- `margin sampling`

### Parameters

- `initial_size`: e.g., 100
  - `batch_size`: e.g., 50
  - `test_size`: 10%
  - Up to 100 **iterations** used
  - **Metrics tracked**:
    - Final accuracy
    - Accuracy mean and standard deviation
    - Execution time
- 

## 7. Results

### 7.1 Overview Table

The full table with all active learning runs and evaluation metrics is available here:

[active\\_learning\\_runs\\_v1.csv](#)

**Path:** `smart-energy-ml-analysis-jsi/tables/active_learning_runs_v1.csv`

---

### 7.2 Visualizations

Two folders contain the generated figures, depending on the data splitting strategy used:

#### Figures from Train/Test Split

**Path:** `smart-energy-ml-analysis-jsi/figures/al_train_test_split/`

Contains plots generated using random `train_test_split`.

#### Figures from Manual Temporal Split

**Path:** `smart-energy-ml-analysis-jsi/figures/al_temporal_split/`


Contains similar visualizations, but generated using **manual temporal splitting**, where earlier data is used for training and later for validation.

---

### 7.2 Visualizations

#### Accuracy vs. Number of Labeled Samples

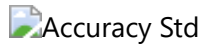
Shows how accuracy evolves as more samples are labeled.

 Accuracy vs Samples

---

#### Accuracy Standard Deviation per Strategy

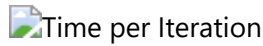
Boxplot showing the variability of accuracy across different runs for each strategy.



---

## Time per Iteration by Strategy

Shows average time per iteration for each strategy, highlighting computational differences.



---

## Final Accuracy Comparison

Distribution of final accuracy scores per strategy.



## 7.3 Key Observations

### Which strategy was most accurate and stable?

- **Most accurate:**
  - Under both splitting strategies, **Margin sampling** and **Uncertainty sampling** consistently reached the highest final accuracies.
    - Example (Manual Temporal Split): **margin** reached up to **0.9190**
    - Example (Train/Test Split): **uncertainty** reached **0.9475**
- **Most stable** (lowest **accuracy\_std**):
  - **Margin** and **Entropy** achieved the lowest standard deviation, indicating consistent results across different runs.

---

### Which strategy was fastest?

- **Random** and **Uncertainty** sampling were **consistently faster** per iteration.
- **Entropy** and **Margin** strategies required more computation time, likely due to additional scoring mechanisms.
  - See: **time\_per\_iteration.png**

---

### Did active learning outperform random?

- **Yes**, especially with more iterations.
  - Under **Train/Test Split** at 100 iterations:
    - **uncertainty: 0.9374**
    - **random: 0.9130**
  - Under **Manual Temporal Split** at 100 iterations:
    - **uncertainty: 0.9164**
    - **random: 0.9106**
- In early iterations (1–5), differences were small or negligible.

- Over time, active learning strategie (especially **uncertainty**) showed more consistent improvement.
- 

### Best Overall Result - Train/Test Split


- The **highest recorded accuracy** across all experiments was:

**Final Accuracy:** 0.9475  
**Strategy:** Uncertainty Sampling  
**Data Splitting:** Train/Test Split  
**Parameters:**

- `initial_size`: 100
- `batch_size`: 50
- `iterations`: 40
- `total_labeled_samples`: 2100

- This suggests that:
  - **Uncertainty sampling** with a **moderate batch size** (50) and **sufficient iterations** (40) is highly effective.
  - The **Train/Test Split** setup led to a slightly better top score than Manual Temporal Split, possibly due to less strict data separation.

### Supporting Visualization:

 Uncertainty vs Random – Best Accuracy

You can find this result in the table under timestamp 20250513\_183415.

### Best Result – Manual Temporal Split

- **Highest final accuracy** using the **Manual Temporal Split**:

**Final Accuracy:** 0.9259  
**Strategy:** Random Sampling  
**Data Splitting:** Manual Temporal Split  
**Parameters:**

- `initial_size`: 100
- `batch_size`: 100
- `iterations`: 39
- `total_labeled_samples`: 4000
- `timestamp`: 20250513\_210537

- Although uncertainty-based methods performed well, in this case, **random sampling** achieved the top accuracy with the manual time split.

### Supporting Visualization:

 Best Accuracy - Manual Split

You can find this result in the table under timestamp [20250513\\_210537](#).