



Project to Predict **Grid Stability** and **Failure Prevention**

Group 5





Section 1: The objective, how it has changed over time, what was needed to interact with OpenDSS, lead into the map



Section 2: What was learned for the Map, lead into the ML



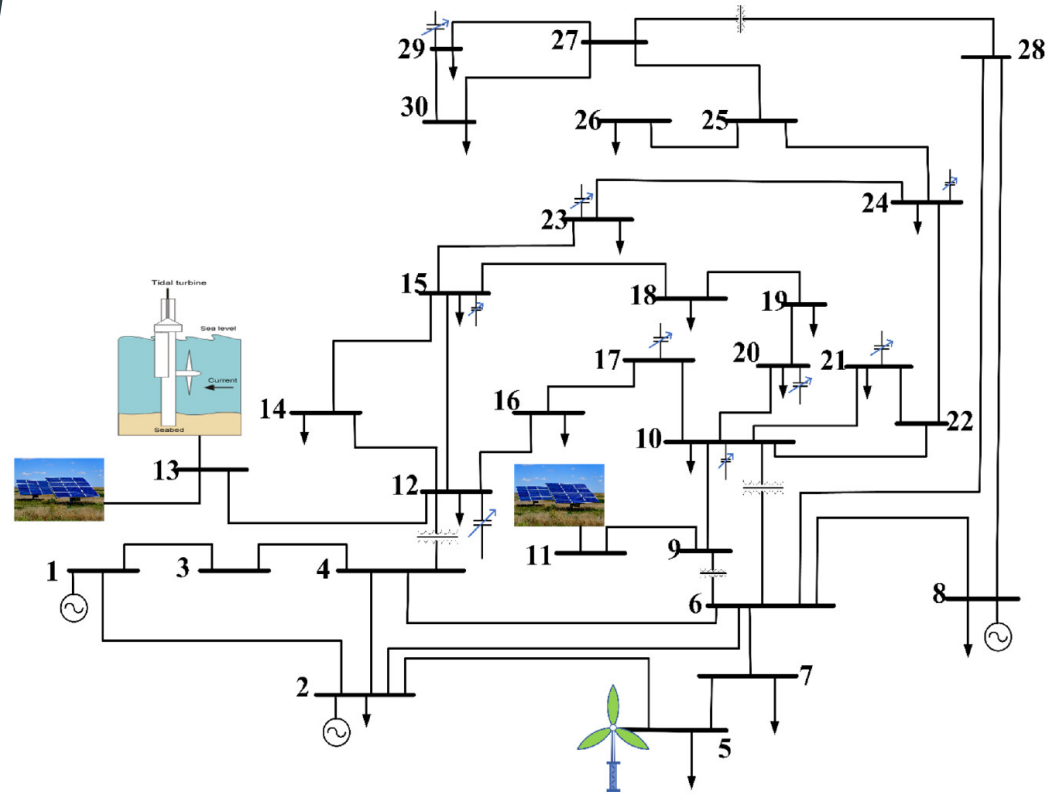
Section 3: ML algorithm ultimately chosen and why



Section 4: Results, Demo, Thoughts and experience

Background Knowledge: Bus System

- This is a test case used by IEEE to replicate an American Electrical Power System
- Bus - a node where several lines are connected, may include loads and generators
- Load - electrical device that consumes electrical energy and converts it to another form



Project inception

- When we first joined this team the project leader spoke of how he wanted to develop a model that would replicate load flow analysis.
- Load flow analysis is the numerical analysis of the flow of electrical power. This helps electrical engineers and students alike view changes within bus systems and electrical grids
- Load flow is a computation and therefore every company has a different approach/formula on how to calculate this for their own specific needs. The point of this project would be to develop a model that would allow students and professors alike to get the results of the analysis without the need of a company's specific approach

More on project

- ▶ The model will specifically be used to try to predict potential disastrous results within a power grid such the failure of transmission lines which can lead to a blackout
- ▶ To determine if there was a failure or not we need to focus on a specific value known as the Voltage Per Unit(VPU) or Per Unit Voltage.
- ▶ Most apparatus (generators, transformers, and lines) have a similar VPU so if any of them within a system have a value out of the normal range of about .95 to 1.05 it is indicative of potential damage to the grid

Specifications

- ▶ We would be using python to interact with a program known as OpenDSS. OpenDSS will be doing the load flow analysis on any bus system that is fed to the program.
- ▶ Using Python we would manipulate the value of the real and reactive power of each load within a bus system by a percentage to see the effect on the VPU at each load.
- ▶ This data will be fed to a ML algorithm to predict future changes in the VPU value
- ▶ Finally, they wanted what we made to be interactive and user friendly. To do this they exclaimed that since most components have real world coordinates they wanted a map that would display the components. From the map the user would be able to manipulate the values and see changes.

What is OpenDSS

- ▶ The Open Distribution System Simulator or OpenDSS is comprehensive electrical system tool for electric utility distribution systems
- ▶ The program supports nearly all analyses commonly performed for utility distribution systems planning and analysis.
- ▶ The program comes with a vast list of commands that we will be making use of in our project

Interacting with OpenDSS:

- To implement some of the functionality that you will later see on the map we would have to make use of imports, methods and functions that allow us to communicate with OpenDSS. Along with creating our own functions to execute what our project leaders wanted. Some of the important methods/properties:
- **dssText.Command(cmd)** - the method takes a string that represents the command to be executed in OpenDSS
- **dssSolution.Solve()** - this will perform the necessary calculations to solve load flow after we make changes to values
- **dssCircuit.Loads.kw/kvar** - the two properties that will be manipulated to see changes in the system

The Map (initially)

- ▶ When the idea of implementing the map was initially brought up to us the project leaders initially wanted the map to highly interactive. Initially they recommended ArcGIS so that we could make use of the coordinates associated with each component of the IEEE 9500 bus and plot them
- ▶ From there they wanted the ability to click on each load that was plotted and manually change their values for real(kW) and reactive(kvar) power.
- ▶ Moreover, the interactive map would give users the ability to select a bus or multiple busses, select certain areas, or select components like transformers.

Trial and Error.

- ▶ **ArcGIS & ArcPy:** We tried using ArcGIS with Python API. We faced sign-in issues, especially with the school accounts. It also lacks the interactive features we wanted if we use the browser option.
- ▶ **Python Libraries(like Geopandas, Plotly, Folium):** While Geopandas excels at static geographic plots, it lacks real-time interactive capabilities. Folium, Plotly does offer basic map interactions but falls short in dynamic data updates and direct user edits on map points. These constraints mean neither tool fully meets our goal for a dynamic interactive platform.

Solution???

► Creating a web-based GUI.

- Considering the challenges, we planned to create a full-stack project using HTML, JavaScript, and Python, integrating Folium, and Leaflet with Django/Flask for advanced interactivity.
- This will enable users of an interactive map to click on points, edit attributes of specific buses, or select a region, or filter components and make changes, and reflect changes on the map in real-time. .

► However...

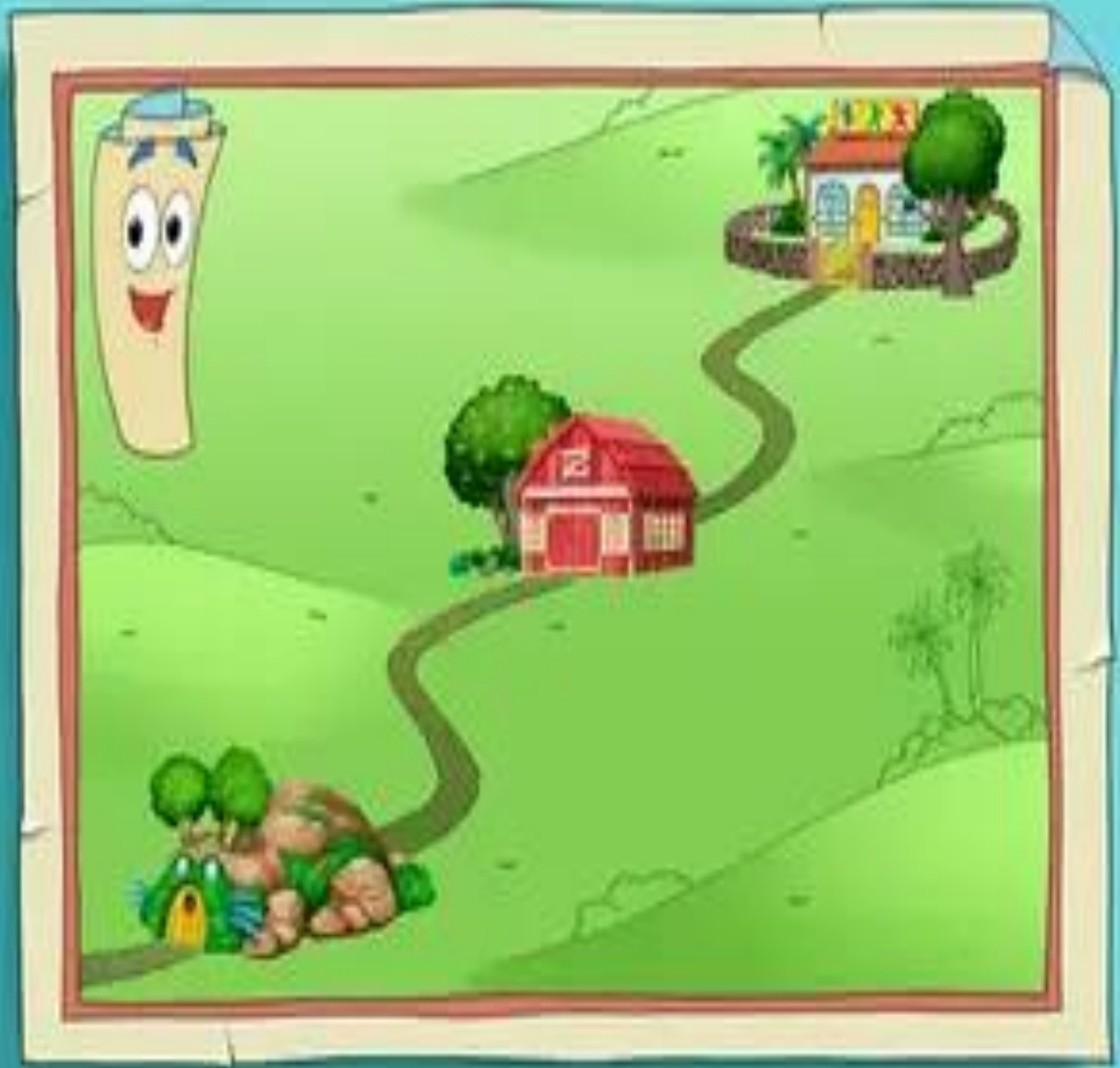
A decision was made here.



- Creating a **full stack project** given our tight timeline wasn't possible.
- We had to look for an intermediary solution.
- We decided to stick using **Python only**, therefore we researched deeper into its libraries.

The map (now)

- After a lot of simplifications, this is how the map looked like.



Python libraries to the rescue

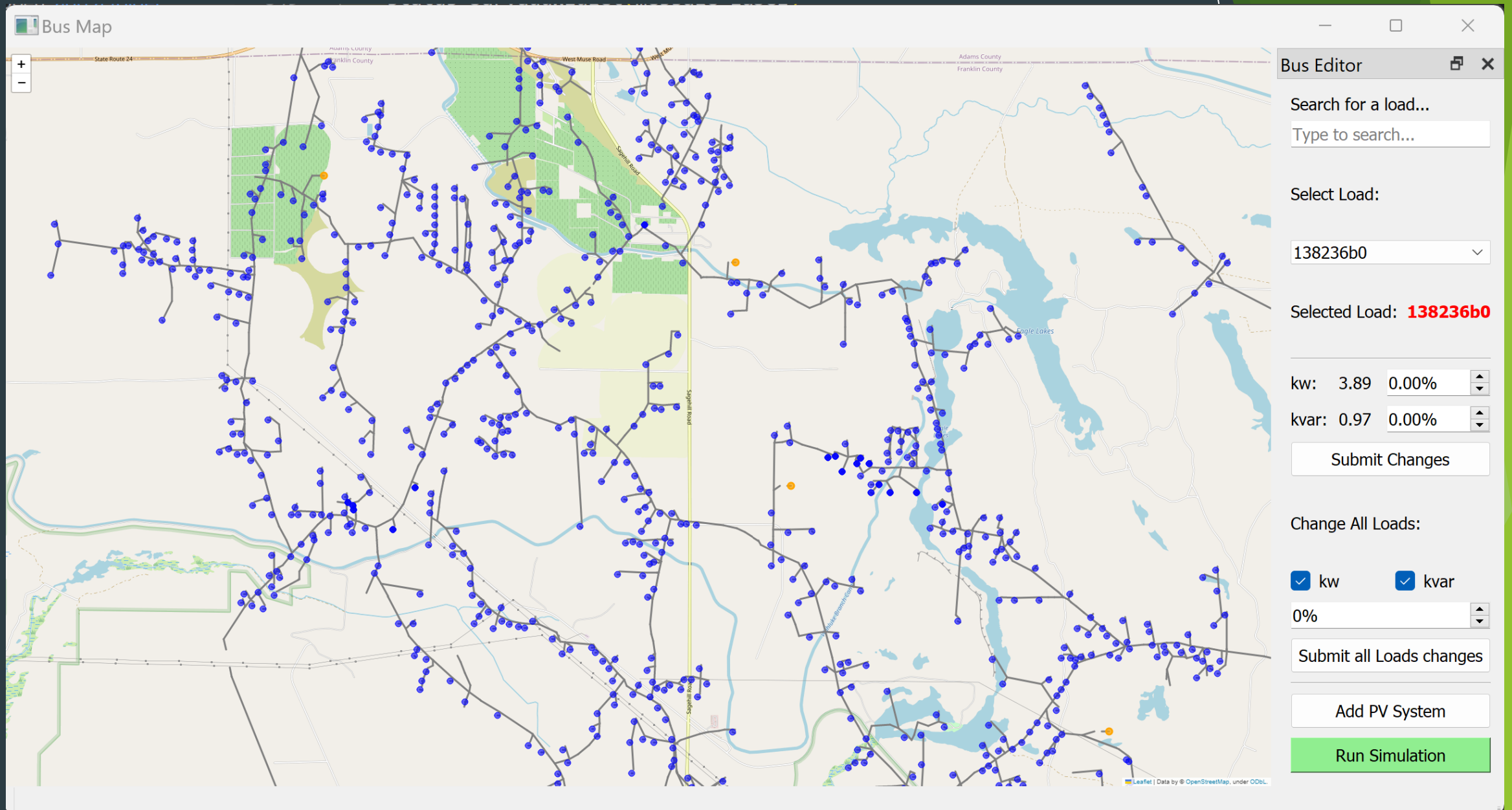
Folium library for basic map interactions.

PyQt library for GUI.

PyQt would be used to create buttons that would run functions that will update the loads based on whatever changes you wanted to them

The user will be able to select a load from a dropdown that they want to change and from there based on the buttons they can manipulate the load

The Map (Now, for real)



Features

- ▶ Search
- ▶ Select
- ▶ Change selected load
 - Kw
 - Kvar
- ▶ Change all loads
- ▶ Submit changes
- ▶ Run simulation
 - Run the load flow model on backend

The screenshot shows a 'Bus Editor' window with the following elements:

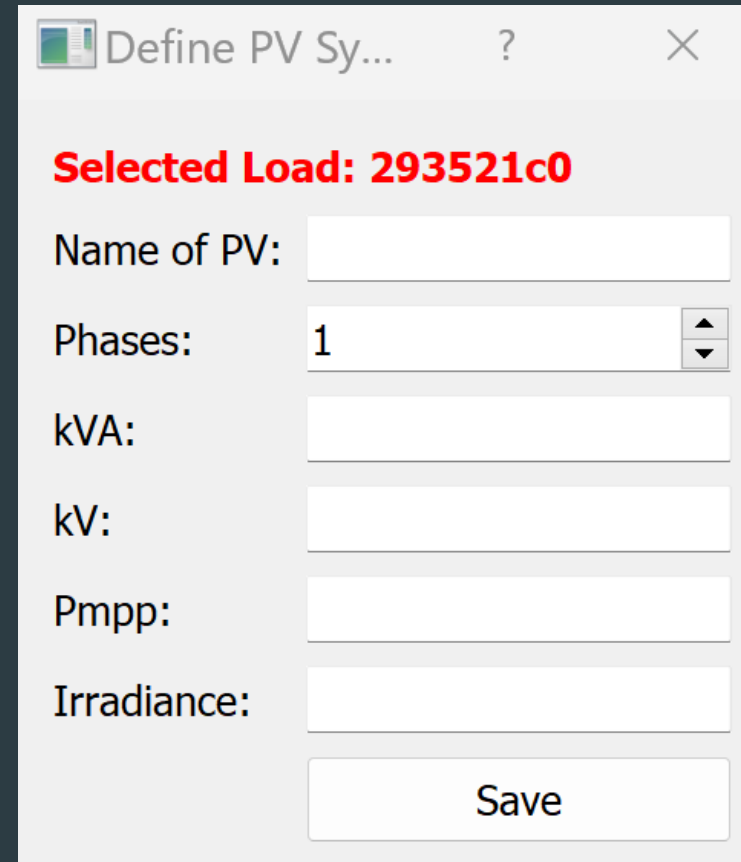
- Search for a load...**: A text input field with the placeholder 'Type to search...'.
- Select Load:**: A dropdown menu currently showing '138236b0'.
- Selected Load:**: A label displaying '138236b0' in red text.
- Configuration fields**:
 - kw:** A value of '3.89' and a percentage spinner set to '0.00%'.
 - kvar:** A value of '0.97' and a percentage spinner set to '0.00%'.
- Submit Changes**: A button to save the selected load's configuration.
- Change All Loads:** A section with two checked checkboxes, 'kw' and 'kvar', and a percentage spinner set to '0%'.
- Submit all Loads changes**: A button to apply changes to all loads.
- Add PV System**: A button to add a new photovoltaic system.
- Run Simulation**: A prominent green button to execute the simulation.

One more feature

The greatest intersection with the above ground team

Adding a PV System (solar panel).

- ▶ The user selects a load where a solar panel would be added
- ▶ Enters the specifications of the panel
- ▶ Reruns the simulation, to see the impact on the grid.



Define PV Sy... ? X

Selected Load: 293521c0

Name of PV:

Phases: ▲ ▼

kVA:

kV:

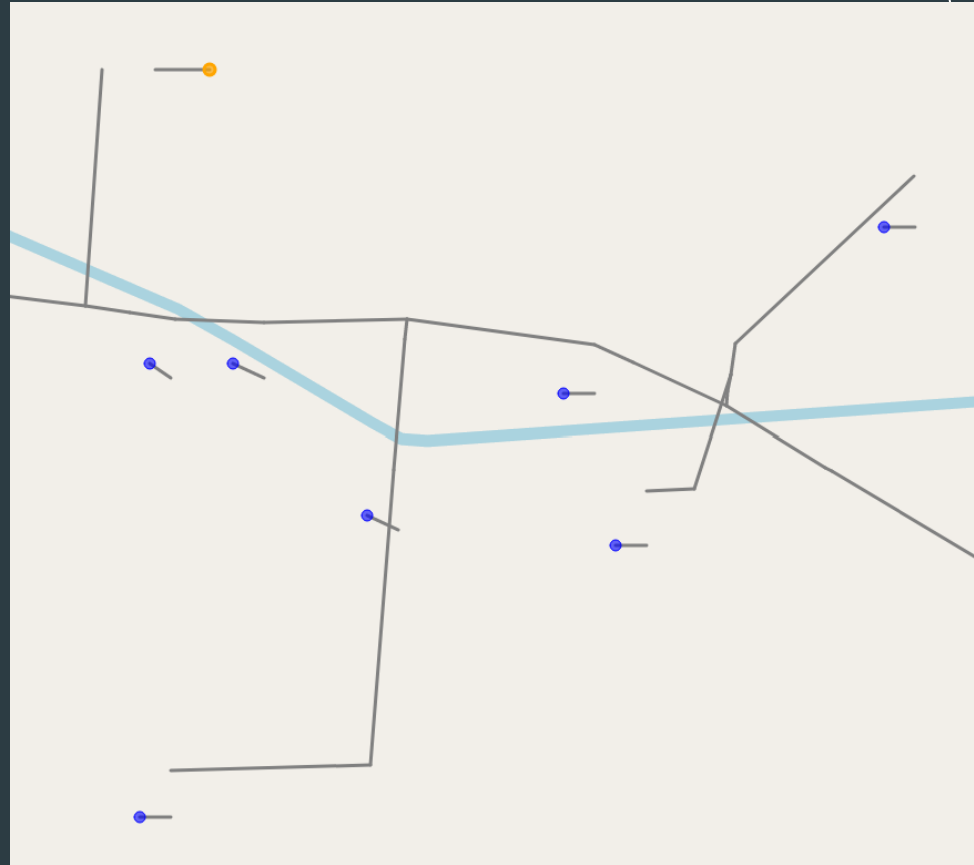
Pmpp:

Irradiance:

Save

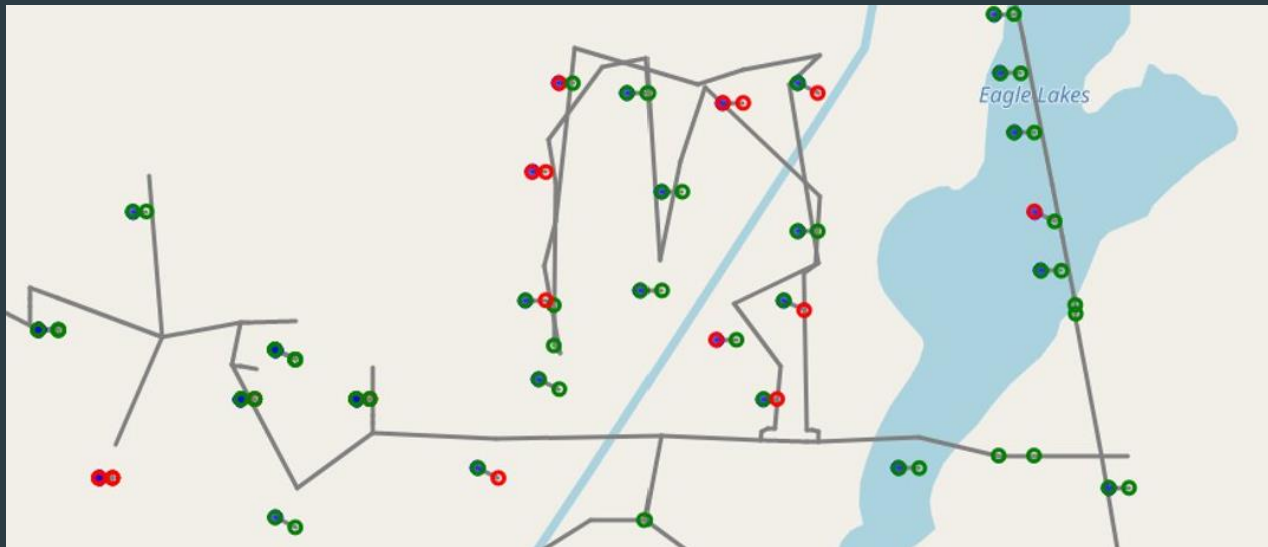
Color markers (initially)

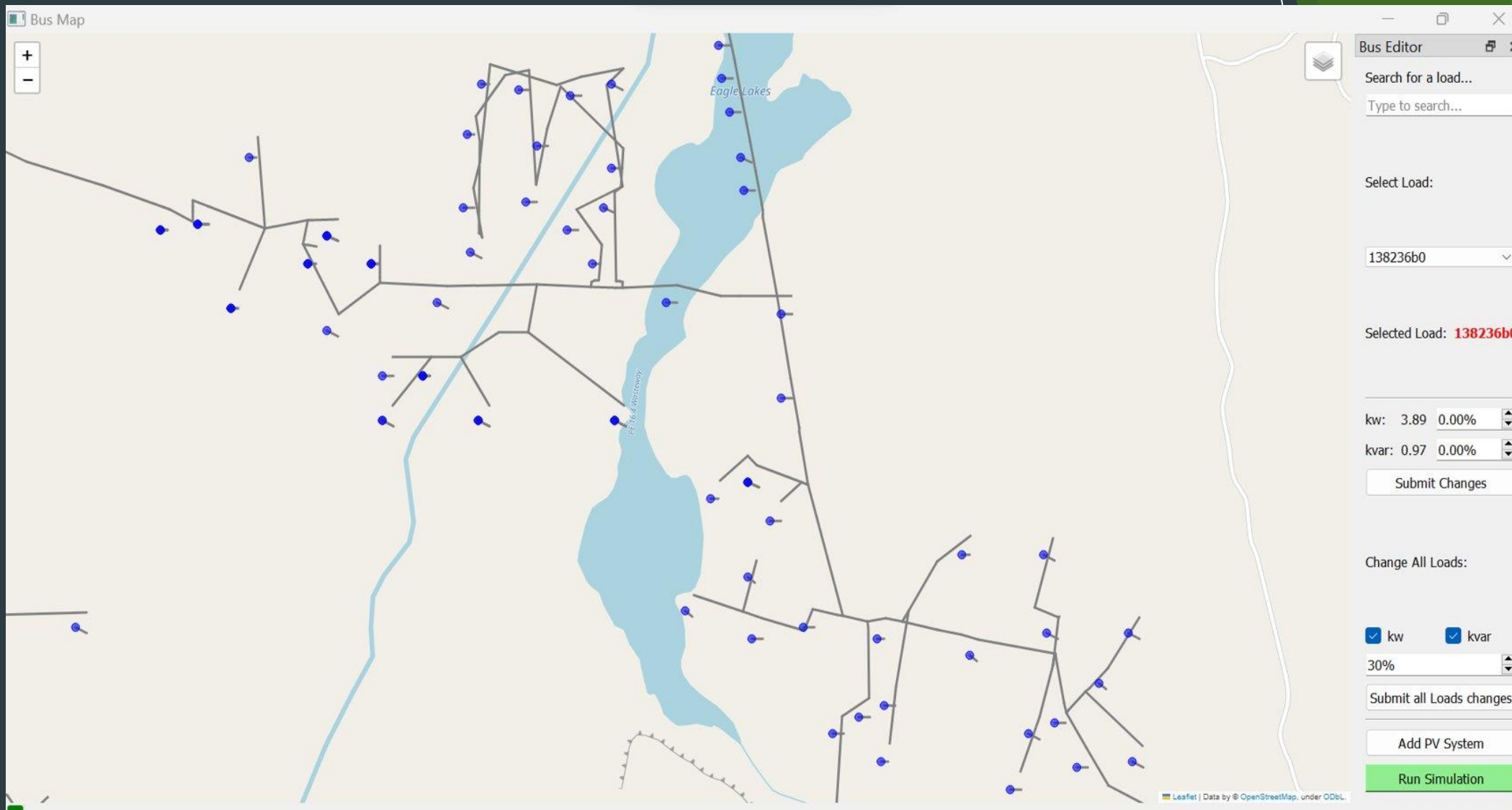
- ▶ Lines - colored in gray
- ▶ Loads - colored in blue
- ▶ Generators - colored in orange



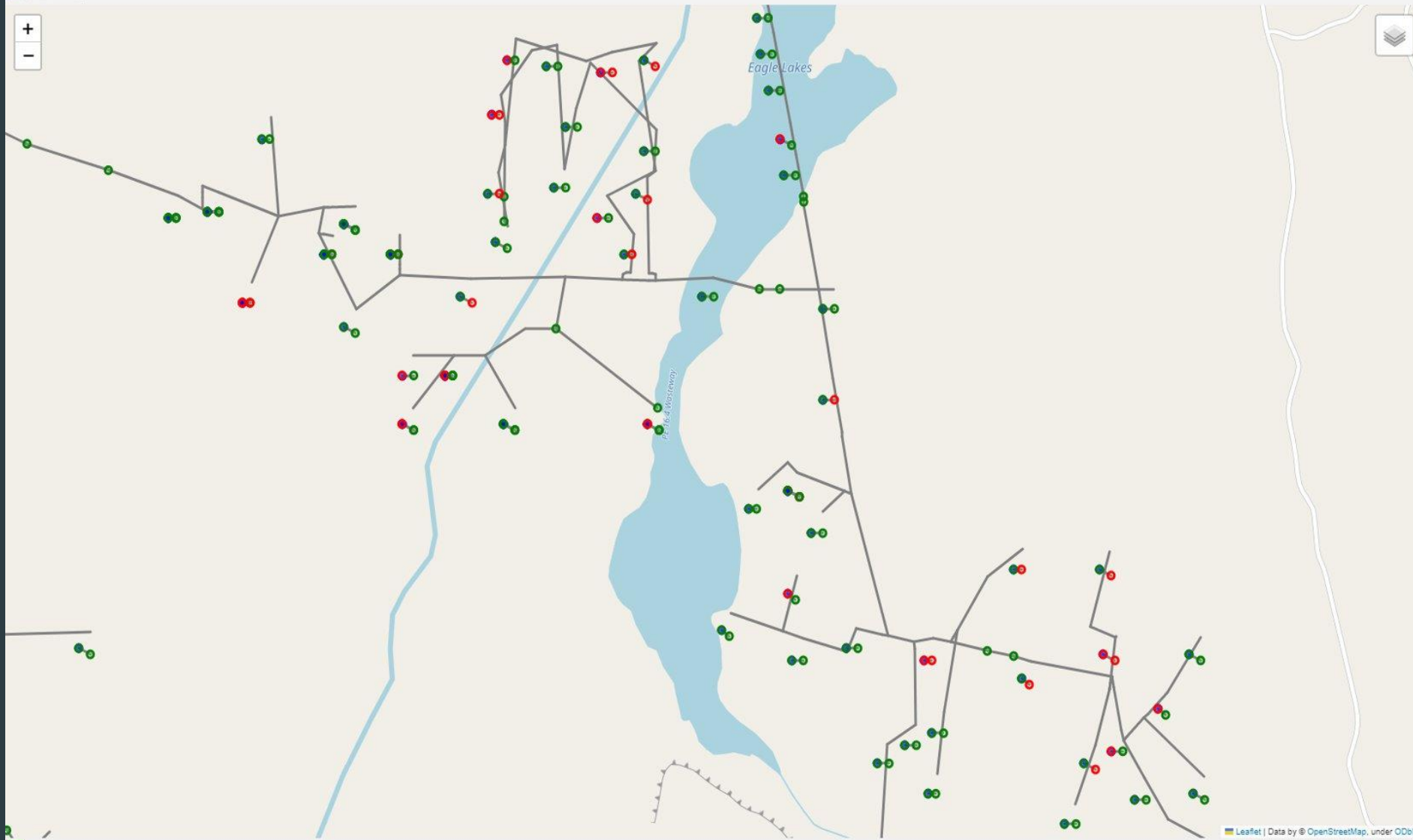
After some changes on the system

- ▶ Once the user makes changes on the side panel and presses the simulation button the map will change. Any bus that has a PU Voltage within the acceptable range will be circled in green, if not within the range it will be circled in red
- ▶ Clicking on the circle itself will allow the user to see the PU voltage, confirming the status at this bus





Bus Map



Bus Editor

Search for a load...

Type to search...

Select Load:

138236b0

Selected Load: **138236b0**

kw: 3.89 0.00%

kvar: 0.97 0.00%

Submit Changes

Change All Loads:

☒ kw ☒ kvar

30%

Submit all Loads changes

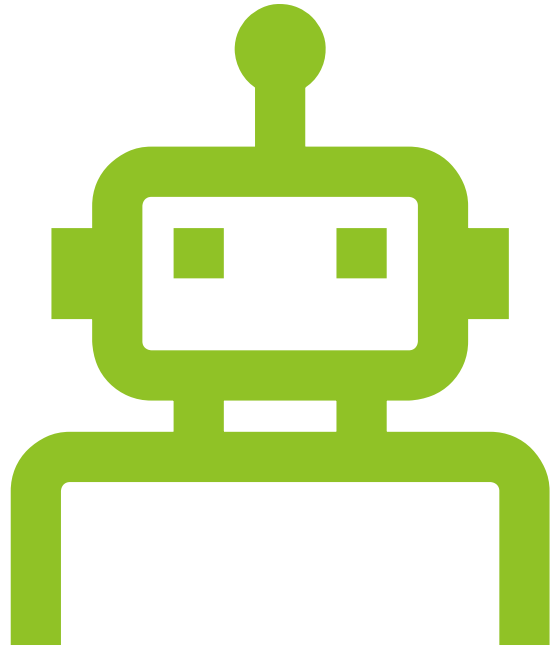
Add PV System

Run Simulation

Leaflet | Data by © OpenStreetMap, under ODbL

Changing gears

- ▶ So far, everything we showed was using OpenDSS on the backend to solve the load flow model.
- ▶ Now, moving into the machine learning part.



Introduction to Machine Learning in Electrical Grid Analysis:

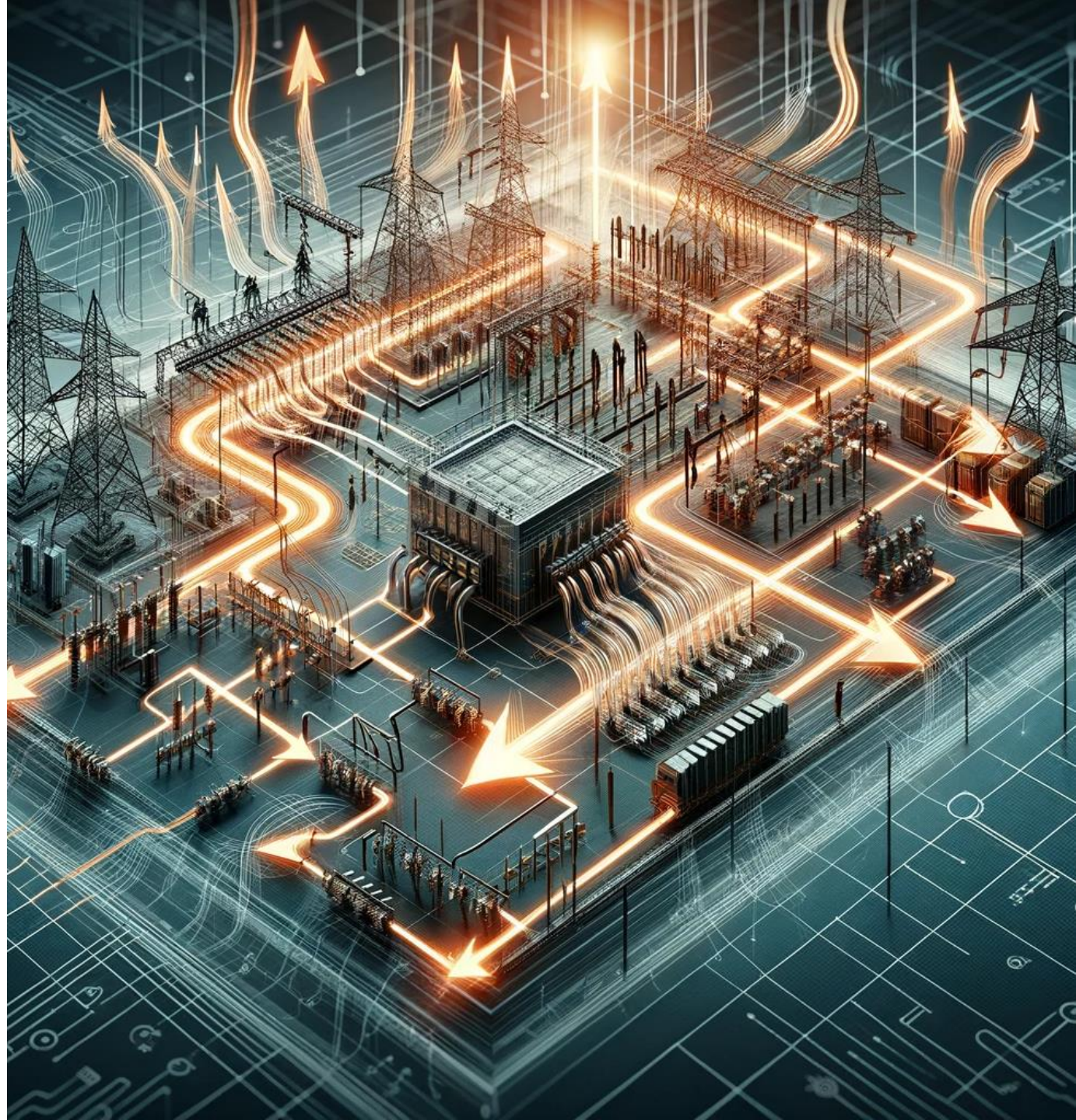


But why Machine Learning?



The Challenge of Power Flow Analysis:

- **Traditional power flow analysis** is essential for evaluating how electricity moves through the grid.
- Steady-state power flow analysis helps **determine** system variables like **voltage and power flow** on each node and line.
- DC power flow model is common but has limitations, especially for **renewable energy sources which are unpredictable and intermittent.**



Innovation through Machine Learning:



Machine Learning (ML) offers advanced predictive capabilities and **faster**, more accurate results than traditional DC power flow models.



ML algorithms can **adapt** to complex power systems, handle variable input from renewable sources, and offer robust solutions.

Data: The Foundation of Our Analysis

Input data:

- Bus names,
- KV, KVar
 - loads,
 - generators,
 - PV systems
- Label:
 - PU Voltage Values



Output data:

- PU Voltage Values

PU Values

0.95



Good



1.05

Generating Training Data:

Method:

- **Tool Used:** OpenDSS for power system simulation.
- **Simulations Run:** 2500 scenarios for varied conditions.

Data Parameters:

- **Inputs:** Bus Names, Load, Generators, PV Systems.
- **Adjustments:** Voltages, kW, kVAR (range: -50% to +50%).

Generating Training Data:

Method:

- **Tool Used:** OpenDSS for power system simulation.
- **Simulations Run:** 2500 scenarios for varied conditions.

Data Parameters:

- **Inputs:** Bus Names, Load, Generators, PV Systems.
- **Adjustments:** Voltages, kW, kVAR (range: -50% to +50%).

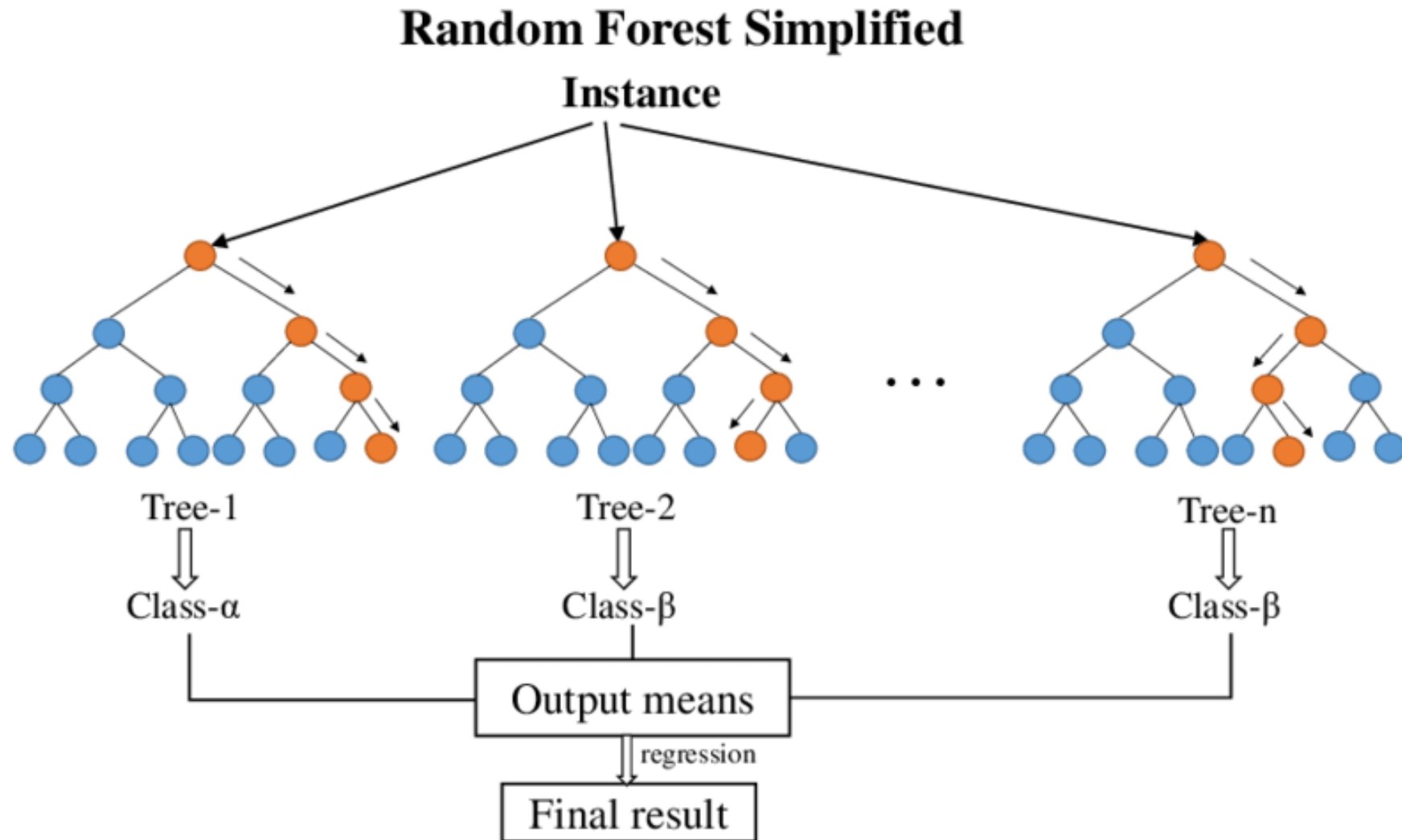
Process:

- **Python script** automates simulation and collects data.
- Data structured in Pandas DataFrame, exported as CSV.

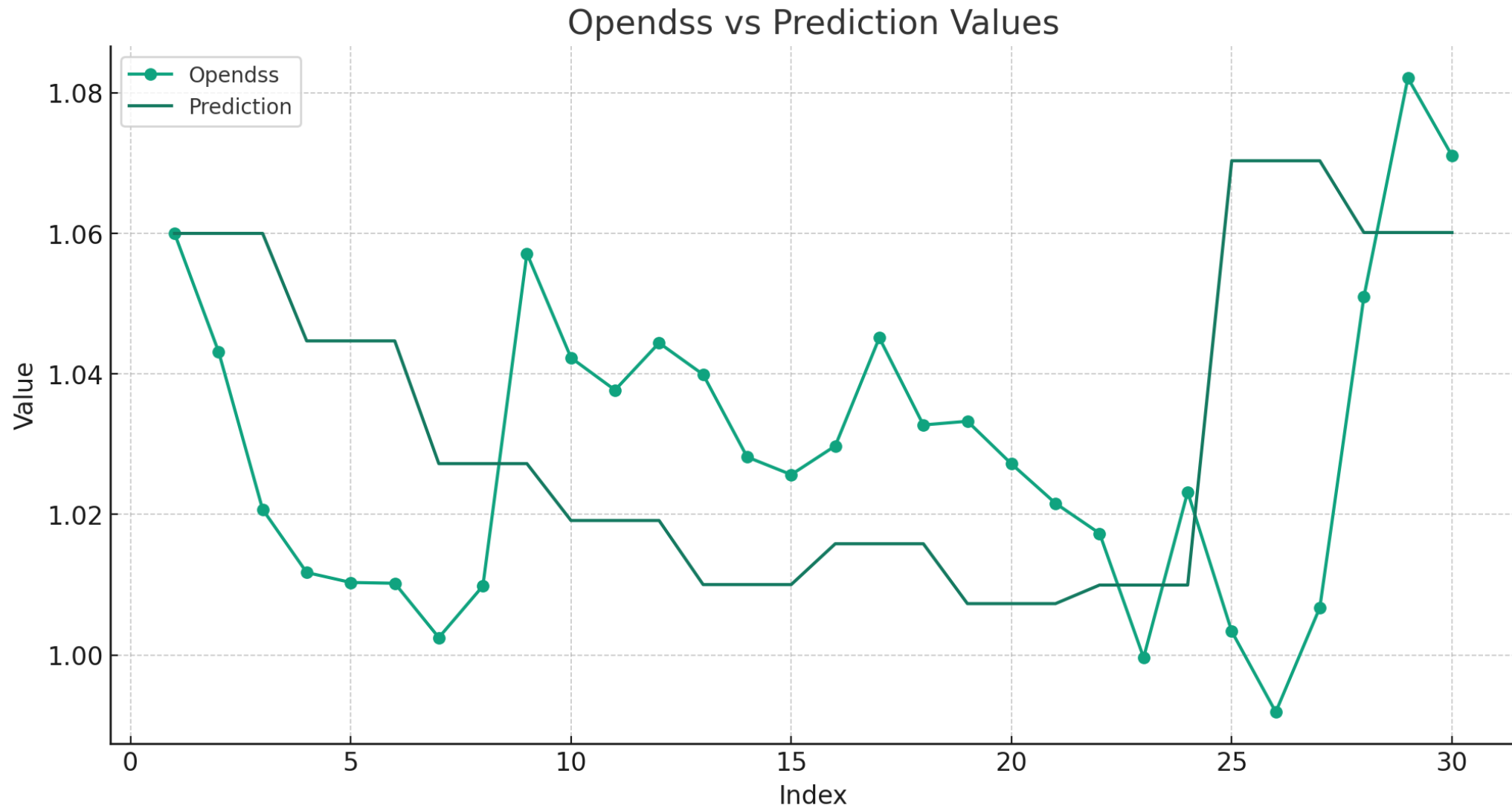
Outcome:

- Dataset capturing realistic grid behavior.
- Focus on Per Unit (PU) value changes to assess grid health.

Random Forest- First Model

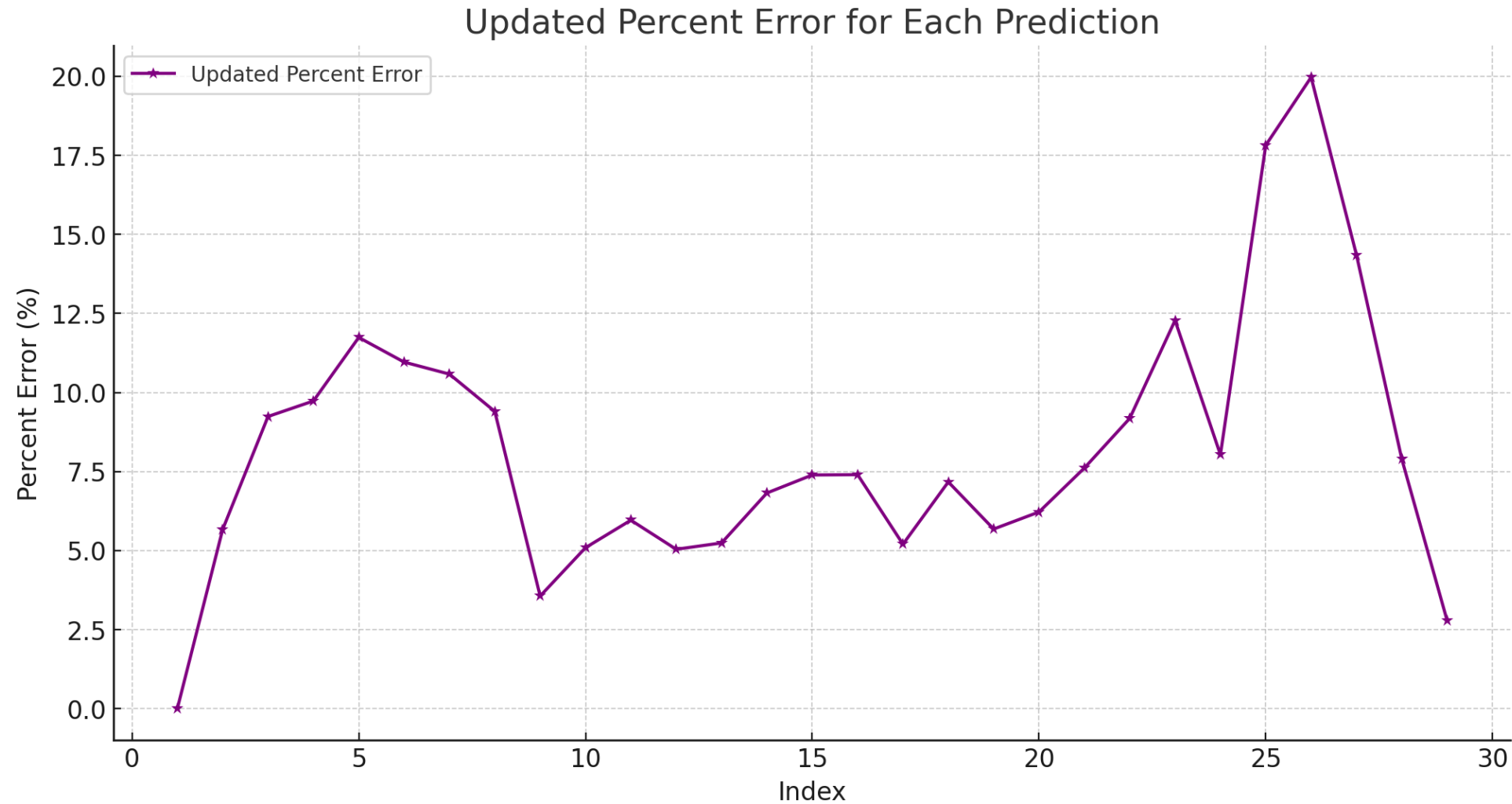


Expected PU values vs Prediction PU values(IEEE-30, Base case)

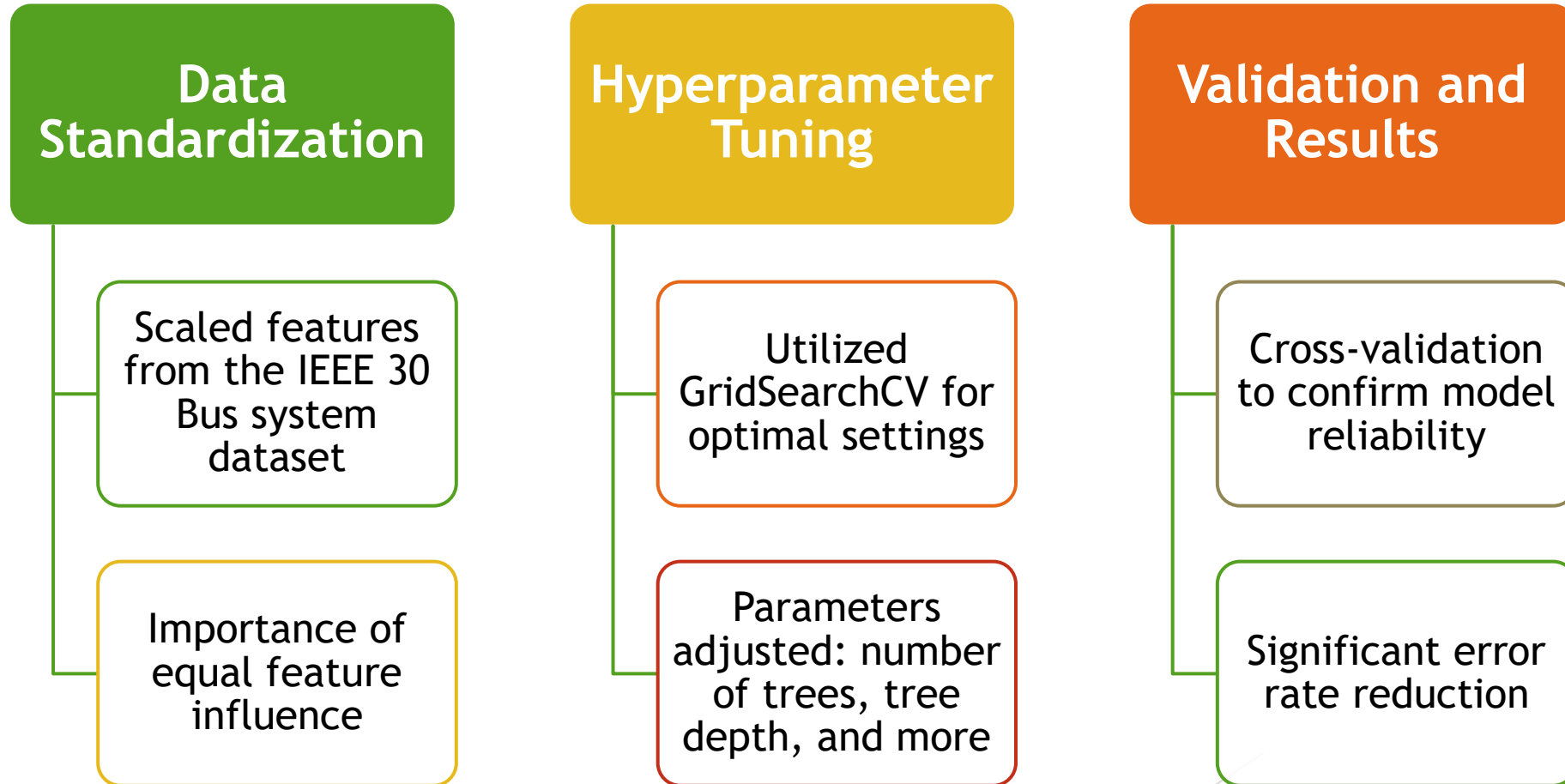


Expected PU values vs Prediction PU values(IEEE-30, Base case)

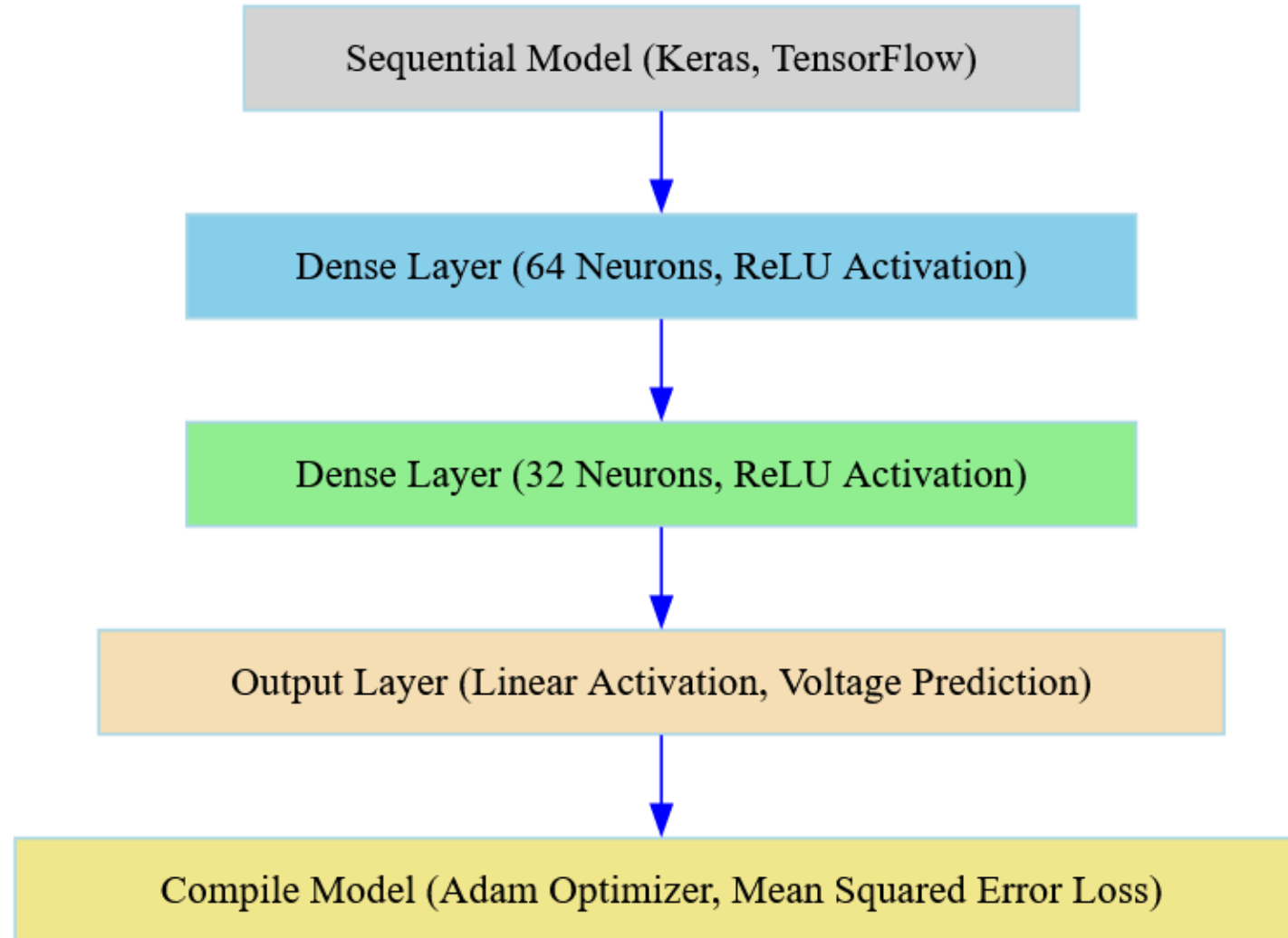
Error %

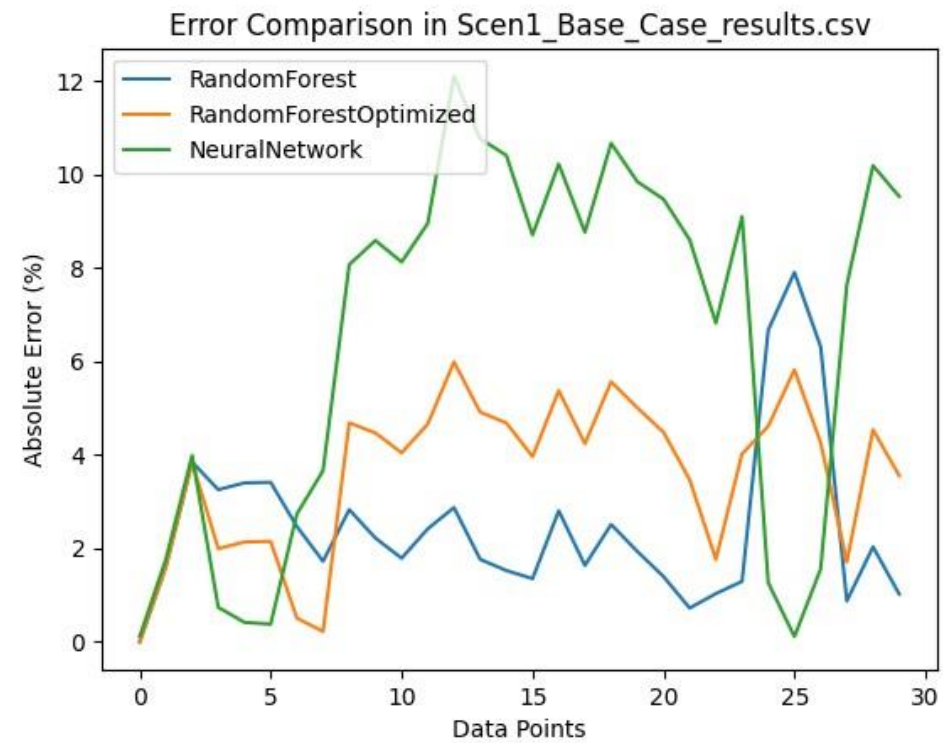


Optimization: **GRID SEARCH**

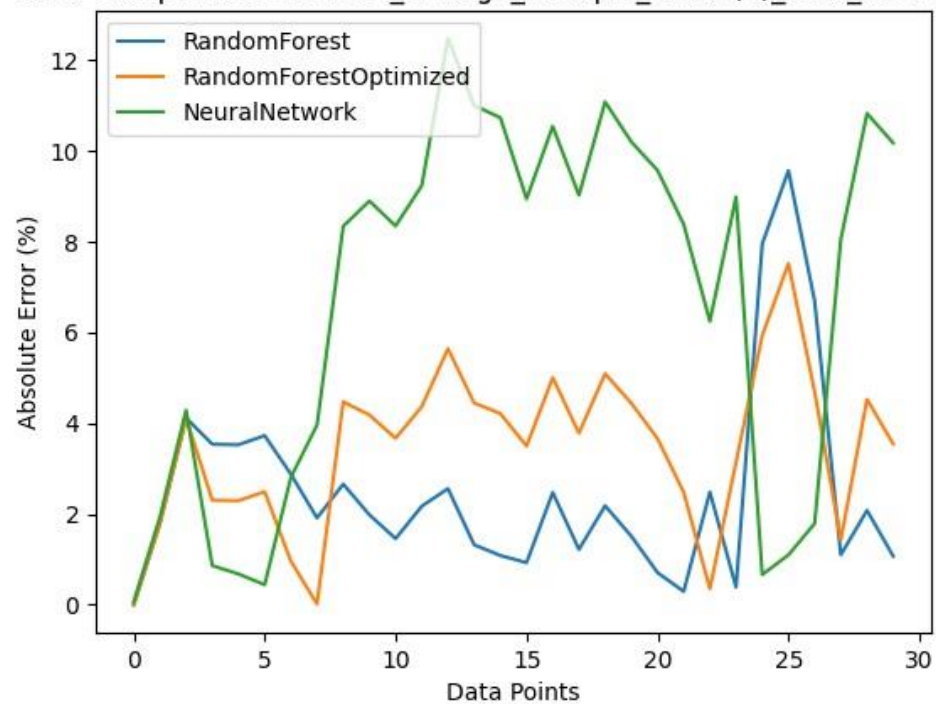


Neural Network

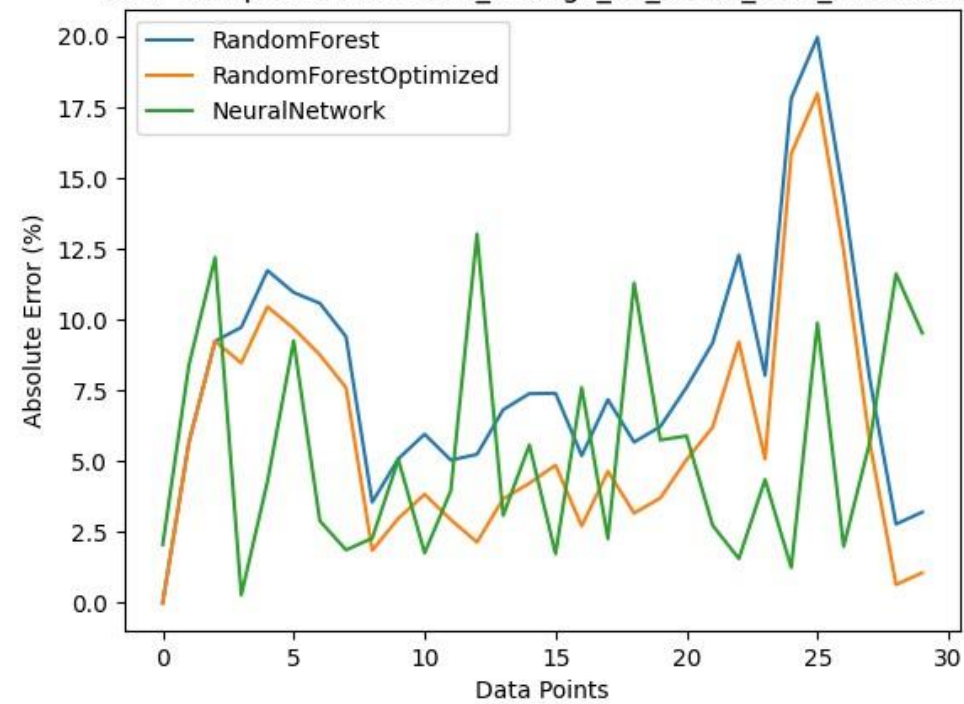




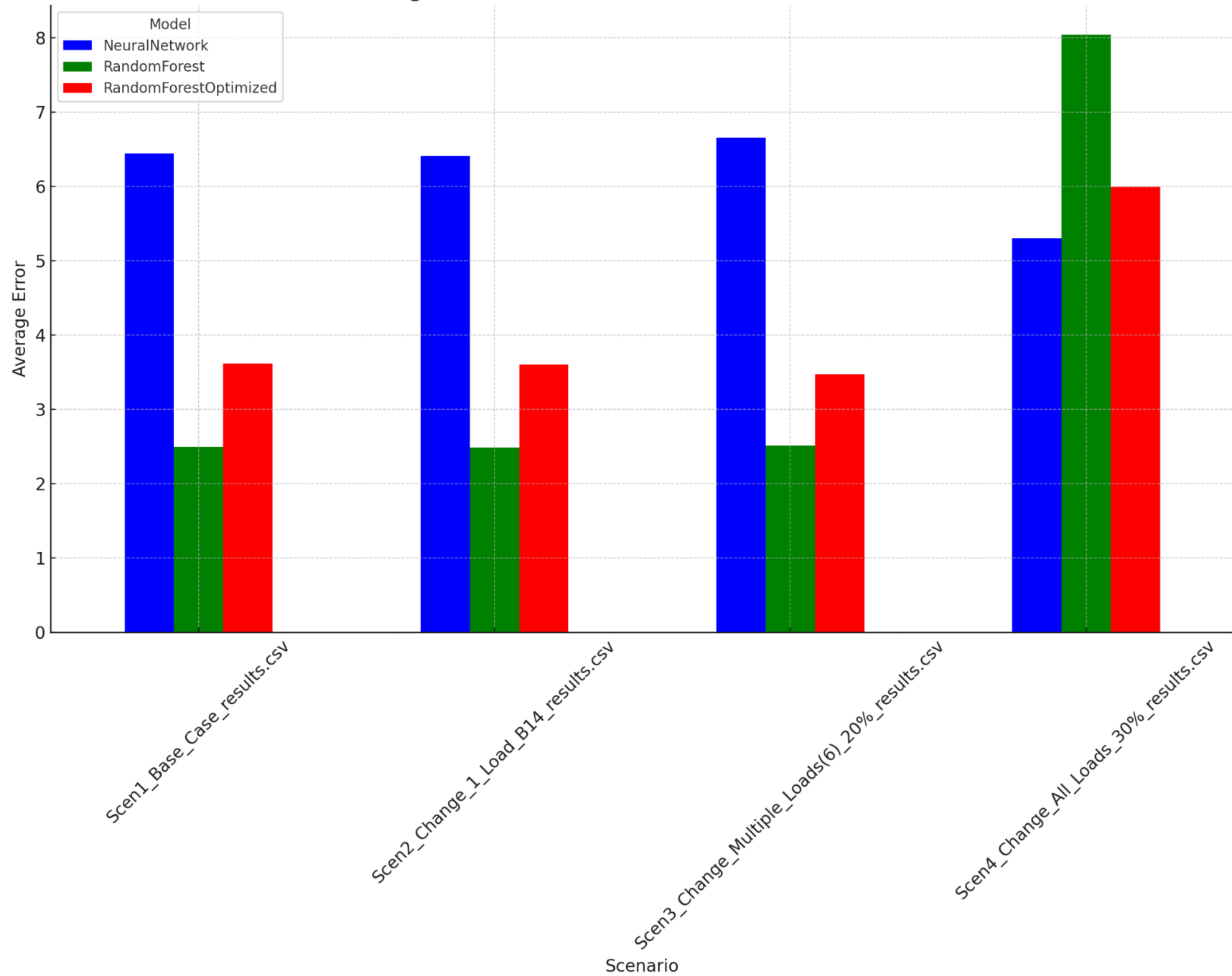
Error Comparison in Scen3_Change_Multiple_Loads(6)_20%_results.csv



Error Comparison in Scen4_Change_All_Loads_30%_results.csv



Average Error Across Different Scenarios and Models



Integrating with GUI- Final Workflow



Accessible Color Palettes:

- Use of high-contrast and colorblind-friendly visuals
- Importance for users with visual impairments

Integration of QAccessible Framework:

- Making the application navigable with screen readers

QAccessible Methods Implemented:

QAccessible.Alert:

- Immediate alerts for system messages

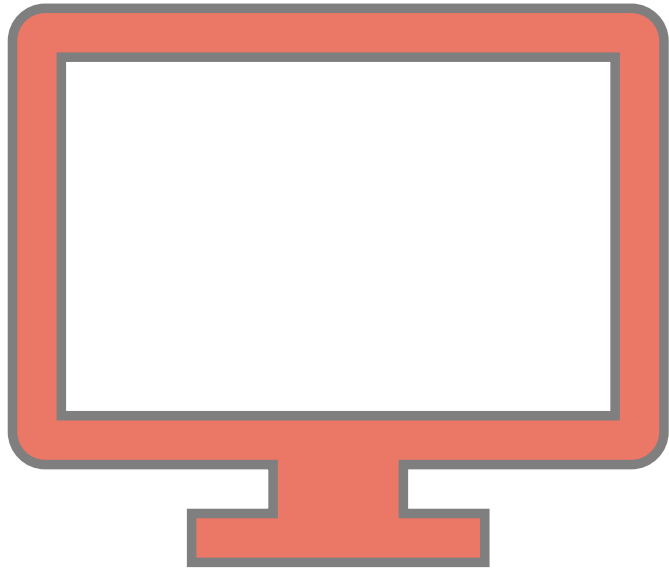
QAccessible.DialogStart/End:

- Notifications when dialogues are opened or closed

QAccessible.PopupMenuStart/End:

- Announcements for pop-up menu actions

**Commitment
to
Accessibility**



Live Demo



Project Experience

- Overall, we got to learn what it is like to work with project leaders and how to work within a deadline
- We learned there will be times where your project leader will make a request for something to be implemented just to later find out that the implementation is not possible
- In cases like this it is best to inform them of the limitation and then pivot to something as close to their original vision as possible
- Communication is of the utmost importance

Recap:

- ← OpenDDS performed flow analysis on the provided bus system
- ← Developed a model for load flow analysis replication
- ← Aimed to predict potential disastrous outcomes in a power grid
- ← Focused on identifying transmission line failures that could lead to blackouts

Next Step...

- ← Working on the report, and meet with our team leads
- ← Write a detailed documentation because the project might be continued on the future by adding more features on the map, or make it a full stack app for high interactivity
- ← Also, the ML model could be revised to make it more accurate

