-EE / CprE / SE 492 – sdmay21-23
Grid AI
Week 5 Report

3/15/21 – 3/29/21
Client: Dr. Ravikumar Gelli
Advisor: Dr. Ravikumar Gelli

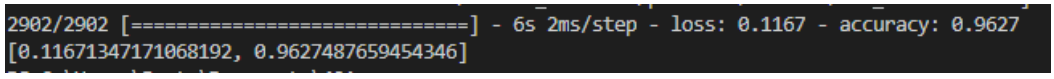
Team Members:

Justin Merkel — *ML Developer, Backend Developer*
Patrick Wenzel — *Frontend Developer*
Abhilash Tripathy — *Frontend Developer*
Karthik Prakash — *Backend Developer*
Abir Mojumder — Frontend/Backend Developer

Weekly Summary

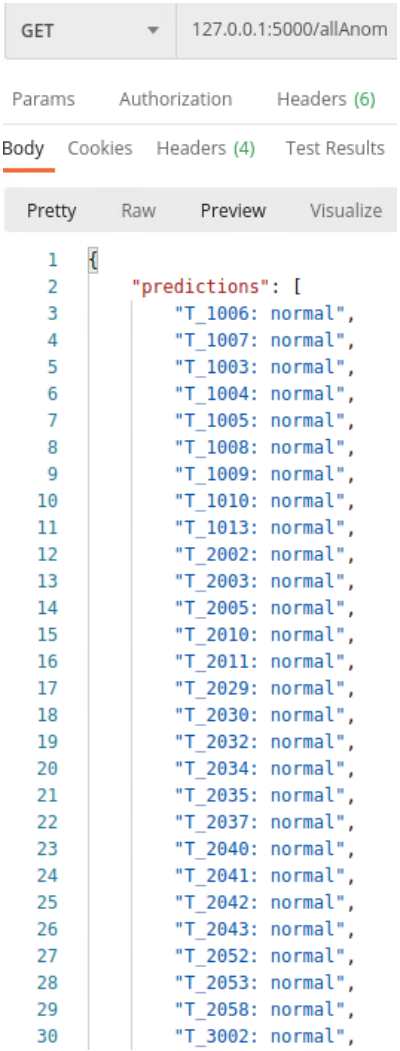
During this past working period, we have begun the process of integrating our components from the backend and frontend while also developing upon the work that we have. We have an initial visualization dashboard that uses the node data stored in the Neo4j database. The dashboard plots the different transformers in relation to each other and can show the "current value" of the transformer. As of now, the current value is a static value in the database. Work is in progress to implement functionality to dynamically update the current power output of a transformer. This will be done with a MySQL instance that holds the smart meter data. SQL query will be made to the MySQL database which will provide the data to update the nodes in the Neo4j database with a Cypher query. Our team has chosen to use the SQLAlchemy API because of its support and integration with Flask, our backend framework. Other dashboard components are also in progress to provide a more detailed representation of the power grid.

Past Week Accomplishments

- Justin-
    - Finished the other two transformer type logistic regression models
        - 

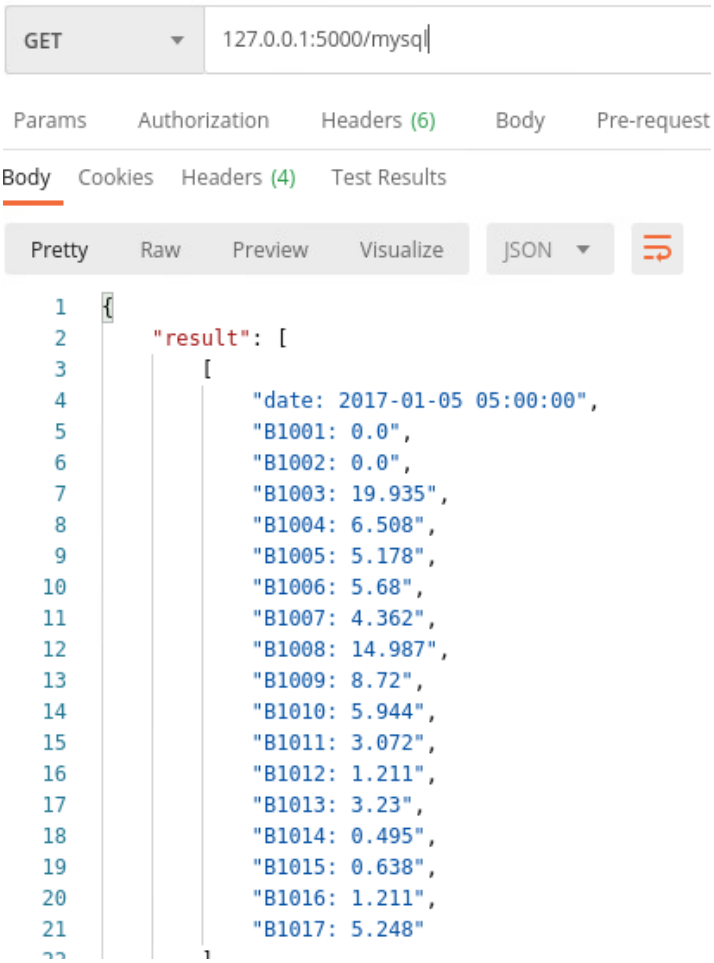
        - 96% accurate predictions on anomaly class with 0.11 sparse categorical cross entropy error which means our model is very closely mapping our testing data which the model does not get trained on.
    - Loaded the OpenDSS model

- Karthik -
    - Integrated Justin's logistic regression models for each transformer type. This allows us to determine the status of a node which can be accessed by the frontend.



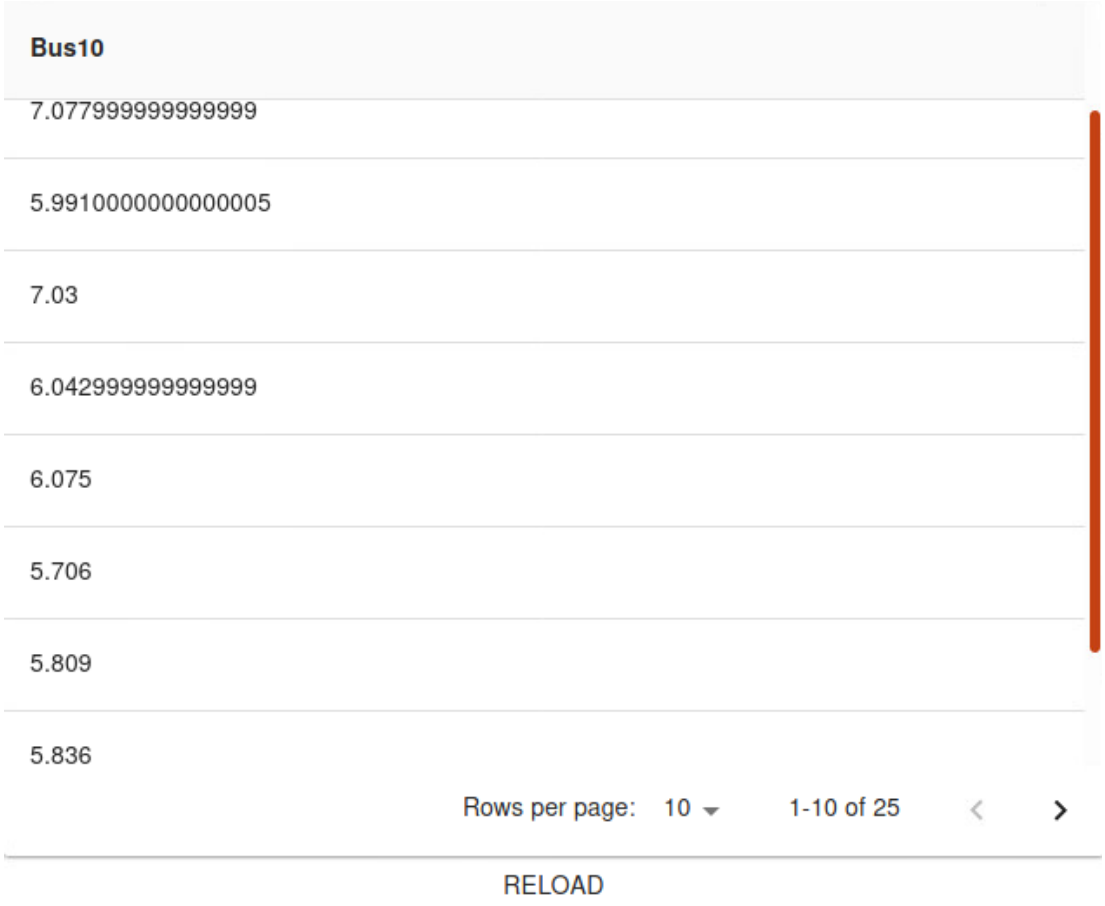
- MySQL docker instance has been initialized. Some partial time-series data has been loaded into the database to allow for testing. The goal for right now is to properly format the data that is returned from the SQL query and implement a background task that will periodically query the MySQL instance to update the data in Neo4j.

```python
@app.route('/mysql')
def get_table():
    b = time.localtime()
    query="select * from smart_meter where date='2017-01-05 5:00:00'"
    result = mysql_conn.execute(query)
    output = []
    temp = []
    for row in result:
        for column in row._fields:
            temp.append(column + ": " + str(row[column]))
        output.append(temp)
    return {'result': output}
```



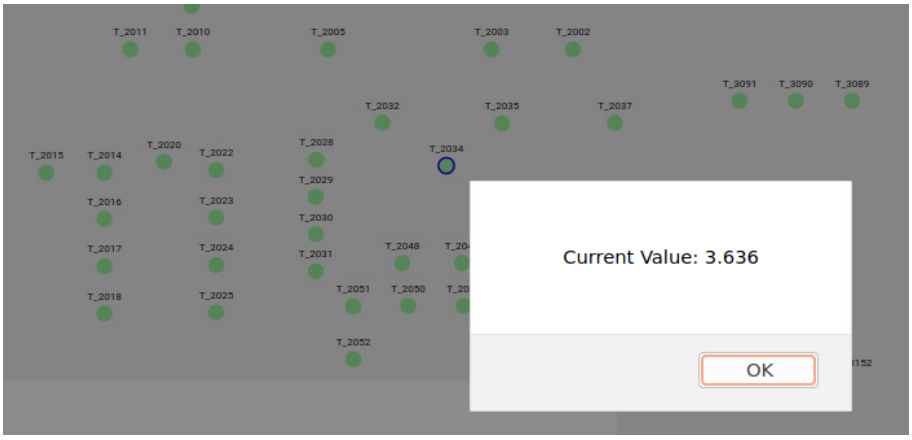
Abhilash -

- Made tables more modular (table components now accept null, undefined arrays as well as arrays with multi-property objects as elements).
- Added UI features like scrollability, pages, and row per page options to the Table component. Changed the table design that is suited better for displaying tables on constrained sections of a single page.
- Reorganized folder structure smoother development process.

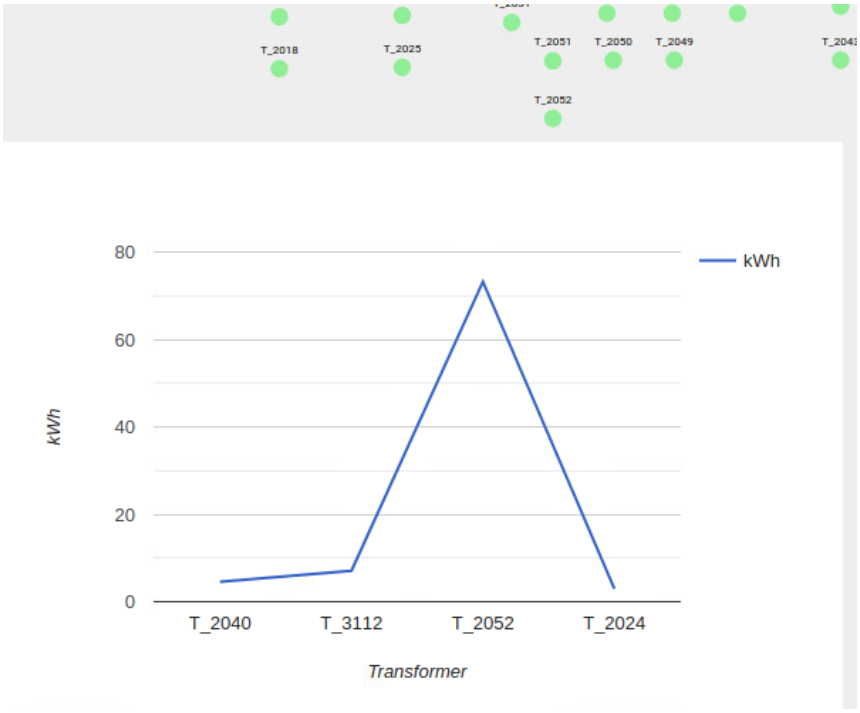


Abir - Experiment with Async/Await method calls to fetch api data.

- Current value based on node clicked can be fetched from the backend asynchronously in React.



- Implementing an output of line chart data to show time-series readings of the clicked node, and show the predicted value in the chart. Currently placing dummy data in the line chart based on previous nodes click until I figure out React state updating without crashing.



Patrick:

- Compared Flask vs. preset routing and determined that we should move forward with the preset routing of the dashboard. Starting on making these work with our tables and grid map on different pages.

Pending Issues

- Understanding where pages get their routes from to be able to set up custom pages with custom routes
- Implement real-time data updates for kWh outputs with new MySQL instance and task scheduler
- Import node relationships for Neo4j
- Allow for new data to be uploaded into databases
- Figure out forcing react render update, or updating state by correctly updating current state.

Individual Contributions

Team Member	Contribution	Weekly Hours	Total Hours
Patrick Wenzel	Determining best setup for routing	4	36
Justin Merkel	Finished the Logistic regression models for the rest of the transformer types and began openDSS simulator investigation	8	41
Abir Mojumder	Async methods and setting up data display methods with fetched data (line chart/bar chart made by Abhilash)	7	31
Karthik Prakash	Configured endpoints for transformer classification models and started MySQL instance with sample data and endpoint	7	32
Abhilash Tripathy	Converting components to be more modular.	8	32

Plans for Coming Week

- Patrick - Understanding where pages get their routes from to be able to set up custom pages with custom routes
- Justin - Create a script to generate contiguous data from openDSS so that the backend will be tested with real-time data.
- Abir - Get time-series data from api once setup and make a basic stable version running on master VM.
- Karthik - Implement real-time updates to kWh outputs stored in Neo4j, create more endpoints for getting time-series data from MySQL, and look into scripts for uploading data from files to database instances
- Abhilash - Migrate all frontend feature developments to the master. Make design decisions to make the frontend simpler while representing all types of data.