IERG4998R Final Year Project II

Machine Learning and Big Data Analytics for Load Profiling in Smart Power Grids

By

Chu Kai Ming

1155094746

A FINAL YEAR PROJECT REPORT
SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF BACHELOR OF INFORMATION ENGINEERING
DEPARTMENT OF INFORMATION ENGINEERING
THE CHINESE UNIVERSITY OF HONG KONG

May, 2020

Abstract

This is a further continuation project on the application of machine learning and big data analytics for load profiling in smart power grids. An algorithm program that could only be used by engineers is not able to populate the usage of machine learning. The benefits brought from technologies should beneficial to every person in the world. To encourage the general usage applications of machine learning, creating a platform that is easy to use and access is one of the best solutions.

The goal of the project is to provide a website that updating the daily wholesale price and the simulations results of the reinforcement learning algorithm to public. The website would update the information daily and automatically. It will get the wholesale price of electricity and display the graphical results of the simulation on daily basis. To achieve the goals, a full-stack website including a backend for data processing and a frontend for result displaying is required. The frontend would be built with React framework of JavaScript. The backend web API would be built with Django REST framework of Python. These two parts joint together to make machine learning and big data analysis more approachable to the public.

Table of Contents

1. Introduction

1.1 Background

In the first part of the project, the machine learning algorithm [1] had been verified and the mathematics representations of the electricity market algorithm had been analyzed. Two test simulation results were generated by the reinforcement learning program [2][3]. However, the program was only be able to be used by professionals or engineers. It is not easy to either understand or use the program as non-technical backgrounded people.

The ideal solution is to provide a data visualization website to display the results of estimated retail price directly. With the help of data visualization tools, the users could still understand and get use of the results from the website. It can benefit most users with less efforts and resources. For example, the customers of electricity retailer would like to know the pricing trend of their electricity usage, they could plan their usage with these results. Another example is for some professionals like data scientist, they could fetch the sorted and estimated results from the website and further elaborate or research on the data.

1.2 Modern Web Development Tools

There are many tools could be used in development of website. Frameworks are one of the package tools that modified and provide more pre-build modules base on a programming language. APIs are also great developer tools built by other developers. Packages of data and pre-build functions are included in the APIs for convenience of developers.

1.3 Rest API

Rest stands for Representational State Transfer, it is a stateless and cacheable application program interface that uses HTTP requests to deal with data. HTTP methods like “GET”, “POST”, “DELETE”, “CONNET”, “TRACE”, “PUT” and more. In this project, the “GET” and “POST” methods are mainly be used. The “GET” method is for the website to get the hourly wholesale price from the ComEd’s API [4][5]. The “POST” method is for the backend server to post the results after finished the simulation.

```
OPTIONS /api/fypapi/

HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept
```

Fig.1 REST API


```
{
    "id": 1667,
    "whole_sale_price": 2.0545454545454542,
    "hours": "01:00",
    "dates": "05/17/2020",
    "reward": 0.0
},
```


Fig.2 Json Format Data fetched from ComEd’s API

2. Methodology

2.1 Frontend Development

Frontend is the user interface that users see on a website. HTML, CSS, JavaScript are the three most popular and essential tools for web frontend development. React is a popular component-based framework that widely used by developers to make the view of a website. It is based on JavaScript and introduce JSX which change the way of writing codes with different mindset. A dashboard appearance with two components had been created with material.

The frontend of the website contains 2 first components. The first component would display a table of reward by time of pricing strategy from yesterday. The data would be sent by backend after the simulation completed. The second component would display the graphic simulation result of the machine learning program.

The frontend would fetch the simulation result from the backend server with a “GET” request at a certain time every day. A table with values of dates, times, wholesale prices and rewards calculated from the machine learning algorithm. The graph with estimated retail price with highest reward would be generated by the machine learning program and be fetched to the frontend as well.

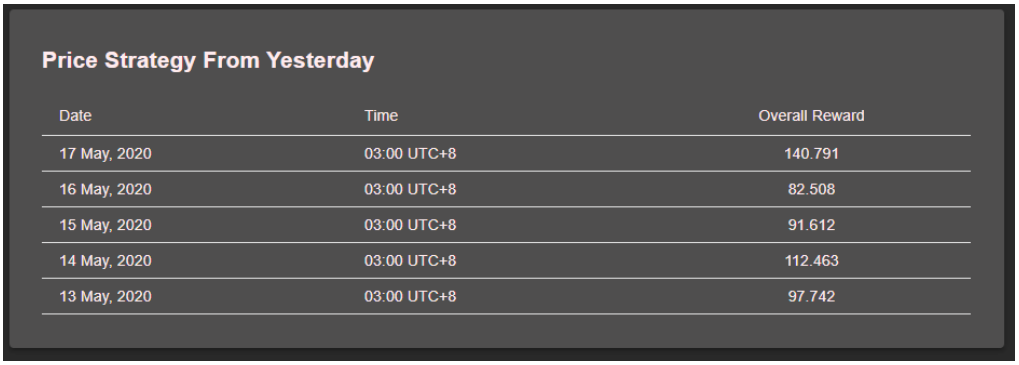

Fig.3 Frontend Interface 1

2.2 Backend Development

Backend is the source of data of the front website view. Django is the backend framework chosen in the project. A Postgres database is created for storing the data fetched from the open API showing the wholesale prices of electricity by time provided by Commonwealth Edison Company. The data would be fetched with REST API with a “GET” request. Then, the fetched data would be transformed into a table and store into the database. Then, the backend server would change the parameters in the machine learning algorithm. The program would be executed automatically in the backend server. Once the simulation results and graphic result are ready, the backend server would give a “Post” request to the frontend client and the results will be displayed on the website.

```
class Fypapi(models.Model):
    whole_sale_price = models.FloatField()
    hours = models.TextField()
    dates = models.TextField()
    reward = models.FloatField()
```

Fig.4 Database Model

```
[17/May/2020 15:35:44] "GET /favicon.ico HTTP/1.1" 404 3052
[17/May/2020 23:47:41] "GET /?format=api HTTP/1.1" 200 5350
[17/May/2020 23:47:41] "GET /static/rest_framework/css/bootstrap.min.css HTTP/1.1" 304 0
[17/May/2020 23:47:41] "GET /static/rest_framework/css/bootstrap-tweaks.css HTTP/1.1" 304 0
[17/May/2020 23:47:41] "GET /static/rest_framework/css/prettify.css HTTP/1.1" 304 0
[17/May/2020 23:47:41] "GET /static/rest_framework/css/default.css HTTP/1.1" 304 0
[17/May/2020 23:47:41] "GET /static/rest_framework/js/prettify-min.js HTTP/1.1" 200 13632
[17/May/2020 23:47:41] "GET /static/rest_framework/js/bootstrap.min.js HTTP/1.1" 200 39680
[17/May/2020 23:47:41] "GET /static/rest_framework/js/csrf.js HTTP/1.1" 200 1719
[17/May/2020 23:47:41] "GET /static/rest_framework/js/default.js HTTP/1.1" 200 1268
[17/May/2020 23:47:41] "GET /static/rest_framework/js/ajax-form.js HTTP/1.1" 200 3597
[17/May/2020 23:47:41] "GET /static/rest_framework/img/grid.png HTTP/1.1" 200 1458
[17/May/2020 23:47:41] "GET /static/rest_framework/js/jquery-3.4.1.min.js HTTP/1.1" 200 88145
[17/May/2020 23:47:44] "GET /?format=api HTTP/1.1" 200 5350
[18/May/2020 00:00:00] "POST /api/fypapi/ HTTP/1.1" 201 98
[18/May/2020 00:00:00] "POST /api/fypapi/ HTTP/1.1" 201 99
[18/May/2020 00:00:00] "POST /api/fypapi/ HTTP/1.1" 201 99
[18/May/2020 00:00:00] "POST /api/fypapi/ HTTP/1.1" 201 98
```

Fig.5 Server Log HTTP methods example

2.3 Full Stack Topologies

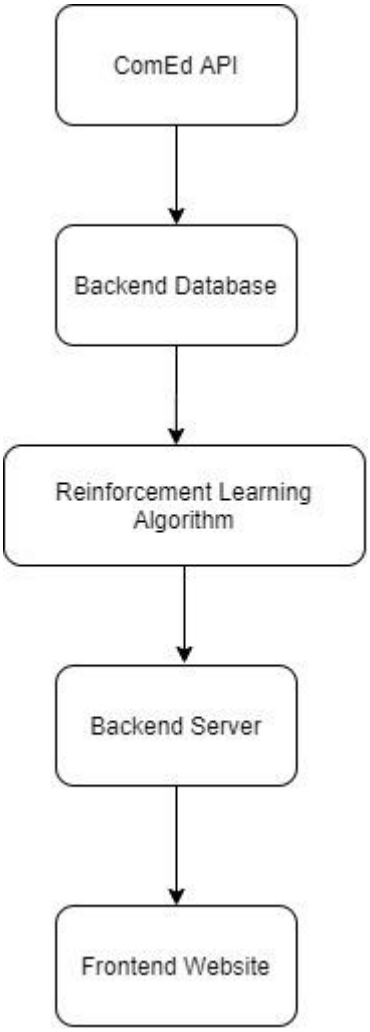

Fig.6 Workflow Diagram

The above diagram shows the workflow in a straight order. There are 5 main steps to create this daily updating full-stack website. The flow would be rerun every day at 00:00 and update the website at 03:00. (Gap time reserved for running the machine learning algorithm)

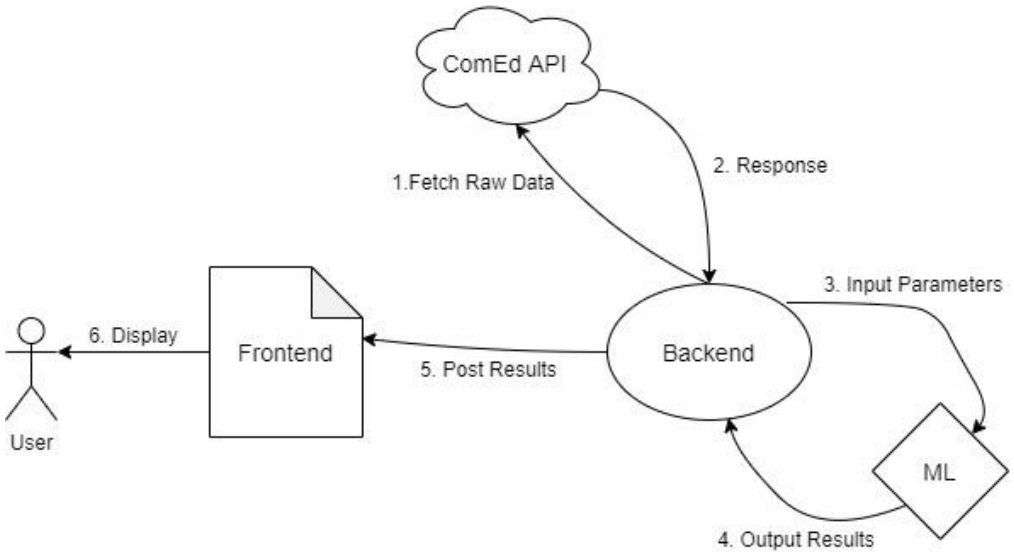

Fig.7 Relationship Diagram between interfaces

The above diagram shows the relationships by interfaces and the actions between these interfaces. Firstly, the backend server would ask for the API of ComEd the data and get the response from the API. The raw data would be processed by a function in the backend server and be stored into the database. Then, the stored data would be proceeded to change the parameters in the machine learning algorithm (ML). After the ML program finished the simulation, the results would be sent back to the backend database and the database would post results to frontend interface. The user would see the results from the frontend of the website.

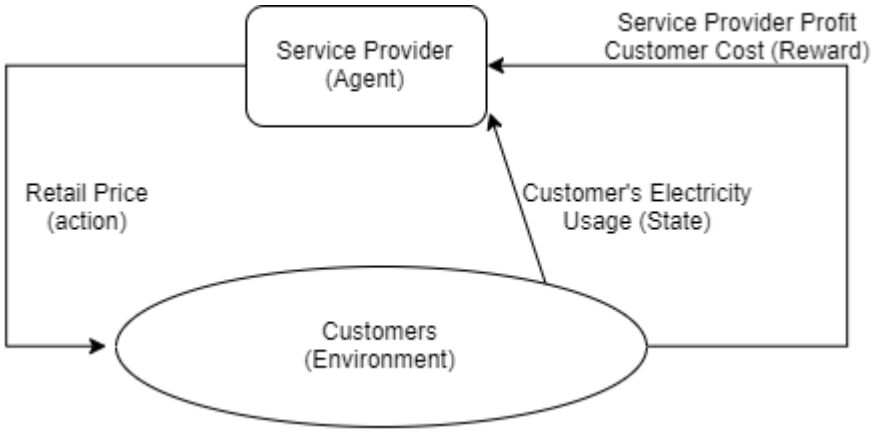

Fig.8 ML Model from previous project (Electricity Retail Market)

3. Results

Fig.9 Capture of the ComEd API (unprocessed)

```
fyp=# Select * from fypapi_fypapi;
  id  |  whole_sale_price  | reward | hours |   dates
------+--------------------+--------+-------+------------
 1665 |  1.481818181818182 |      0 | 00:00 | 05/17/2020
 1666 | 2.0545454545454542 |      0 | 02:00 | 05/17/2020
 1667 | 2.0545454545454542 |      0 | 01:00 | 05/17/2020
 1668 |  2.018181818181818 |      0 | 03:00 | 05/17/2020
 1669 | 1.9363636363636372 |      0 | 05:00 | 05/17/2020
 1670 | 1.9909090909090912 |      0 | 04:00 | 05/17/2020
 1671 | 1.9090909090909098 |      0 | 06:00 | 05/17/2020
 1672 | 1.7363636363636368 |      0 | 07:00 | 05/17/2020
 1673 | 1.7090909090909094 |      0 | 08:00 | 05/17/2020
 1674 | 1.6909090909090911 |      0 | 09:00 | 05/17/2020
 1675 |  1.654545454545454 |      0 | 11:00 | 05/17/2020
 1676 | 1.3363636363636362 |      0 | 12:00 | 05/17/2020
 1677 | 1.1454545454545455 |      0 | 13:00 | 05/17/2020
 1678 | 1.1090909090909091 |      0 | 14:00 | 05/17/2020
 1679 |  1.090909090909091 |      0 | 15:00 | 05/17/2020
 1680 |  1.072727272727273 |      0 | 16:00 | 05/17/2020
 1681 | 1.6727272727272726 |      0 | 10:00 | 05/17/2020
 1682 | 1.0454545454545459 |      0 | 17:00 | 05/17/2020
 1683 | 1.0181818181818185 |      0 | 18:00 | 05/17/2020
 1684 | 0.9909090909090913 |      0 | 19:00 | 05/17/2020
 1685 | 0.9454545454545457 |      0 | 21:00 | 05/17/2020
 1686 |  0.963636363636364 |      0 | 20:00 | 05/17/2020
 1687 | 0.9272727272727274 |      0 | 22:00 | 05/17/2020
 1688 | 0.8454545454545456 |      0 | 23:00 | 05/17/2020
(24 rows)
```

Fig.10 Capture of the Postgres Database (processed)

The above figure is the console of the database displaying the data table. Data in the table showing the wholesale price get from 05/17/2020 which is obtained at 00:00 on 05/18/2020. The data would be passed to the machine learning program and execute by the backend server with the new wholesale price parameters.

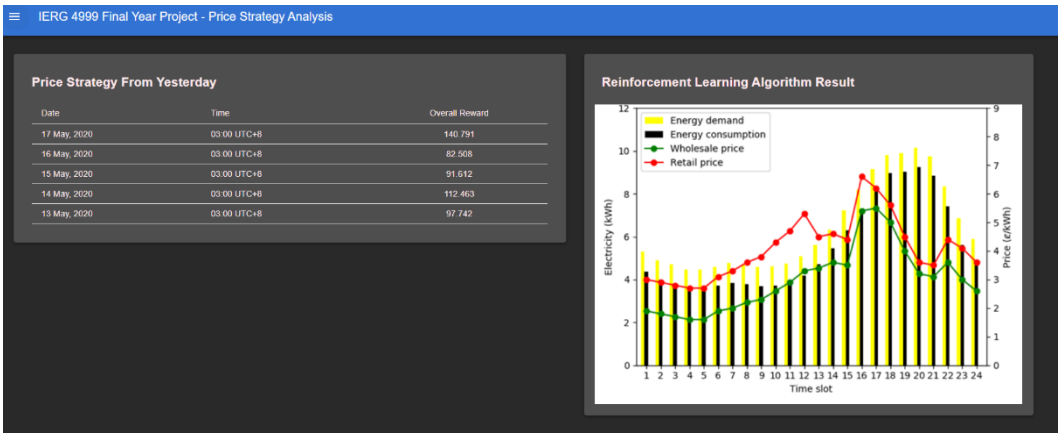

Fig.11 Capture of the website

The above figure is the screen capture of the finished website with results displaying for the users. The source code had been uploaded on GitHub. The left table would show the final rewards of the simulation results in last 5 days. The left box would display the graph. The results achieved the expected outcome of the project.

4. Conclusions

Machine learning is a new era of human society and technology community. In this project, the goal is to build a full-stack website for the machine learning program that were analyzed from the previous project. Merging the machine learning algorithm with web development is popular for data visualization and data processing.

People develop highly skilled technologies to benefit people. If these technologies are no approachable for non-technical background people, it means the technology lost a lot of value and possibilities. Creating bridges for people is the responsibility of an engineer, in both physical and technological ways. Programs are arts created by programmers. The quality of life would be improved by projects and products crafted by engineers. In this project, the efficiency and transparency of electricity market were increased. In other projects, more problems and issues would be solved.

5. Future Directions

The wholesale price of electricity is updating every hour. However, this project is analyzing the pricing strategy every 24 hours. Due to the lack of resources and technical skills, it is not able to be a real-time data analyzing and visualization tool. To achieve the above goal, the server needs to be running the machine learning algorithm fast enough and change the update routine of the whole workflow to become 24 times in a day.

Also, the data provide by the website is too less. It is possible for the website to get other related data and display on the same page to make the content more useful. For example, the comparison between wholesale price, estimated retail price and real retail price comparison are useful information as well.

Finally, the website could also provide a submit form for professional users like data scientists to customize their own parameters and test the results with the reinforcement learning program.

6. References

[1]: “A Dynamic pricing demand response algorithm for smart grid: Reinforcement learning approach”, Renzhi Lu, Seung Ho Hong , Xiongfeng Zhang, Department of Electronic Engineering, Hanyang University, Republic of Korea

[2]: “Q-Learning”, ”Machine Learning”, Christopher J.C.H.Watkins & Peter Dayan, May 1992, Kluwer Academic Publisher, 279-292

[3]: Open Source Q-Learning Algorithm for dynamic pricing of smart grids : https://github.com/hekimgil/grid_pricing

[4]: APIs from Commonwealth Edison Company https://hourlypricing.comed.com/hp-api/

[5]: Hourly price of electricity wholesale price from Commonwealth Edison Company: https://hourlypricing.comed.com/live-prices/

[6]: “Incentive-based demand response for smart grid with reinforcement learning and deep neural network”, Renzhi Lu, Seung Ho Hong, Department of Electronic Engineering, Hanyang University, Republic of Korea: https://www.sciencedirect.com/science/article/abs/pii/S0306261918318798

[7]: ReactJS Template from Material UI: https://material-ui.com/zh/getting-started/templates/

[8]: Documentations of ReactJS: https://reactjs.org/docs/getting-started.html

[9]: Documentations of Django: https://docs.djangoproject.com/en/3.0/

[10]: REST API Tutorial: https://restfulapi.net/