

INFO 7390

Assignment 3 Report

Processor: Sri Krishnamurthy

Group 1

Chang Yu, Songzhe Zhang, You Li

Project Statement

In this case we tried to analyze the consumption of every single property, and then predicted the consumption for the next year. We combined the weather data and holiday data with the original dataset. And create a web page for the application.

The link of the application is liyouneu.com, which is running on AWS Elastic Beanstalk. The main page is for the micro analysis for every single building, the macro analysis is not finished. The visualization tag is for the visualization we did in Tableau.

1.Pre-process

1.1 Original Data

Original Data is 73 csv files each represent the electricity consumption of one building. The category and building name information are included in the file names. In the csv files, Account, Date, Channel, Unit and time from 0:05 to 24:00 are columns and electricity consumption information are included in each row.



COB-
SCH.CLE....2014.csv



COB-
SCH.CLE....2014.csv



COB-SCH.COURT
ST.2014.csv



COB-
SCH.DEV....2014.csv

Account	Date	Channel	Units	0:05	0:10	0:15	0:20	0:25	0:30	0:35	0:40
26440621006	1/1/2014	507115413 1 kWh	kWh	3.74	3.428	3.612	3.552	3.568	3.844	3.504	3.728
26440621006	1/1/2014	507115413 1 Power Factor	Power Factor	0.704842	0.694435	0.706715	0.704722	0.709489	0.708211	0.708723	0.712055
26440621006	1/1/2014	507115413 2	kVARh	3.764	3.552	3.616	3.576	3.544	3.832	3.488	3.676
26440621006	1/2/2014	507115413 1 kWh	kWh	3.752	3.364	3.548	3.708	3.56	3.388	3.36	3.376
26440621006	1/2/2014	507115413 1 Power Factor	Power Factor	0.706353	0.697919	0.707505	0.693512	0.708698	0.701287	0.684754	0.661829
26440621006	1/2/2014	507115413 2	kVARh	3.76	3.452	3.544	3.852	3.544	3.444	3.576	3.824

1.2 Target Dataset Format

For convenient using the dataset to build prediction model, we need to transform the dataset to the format which Account, Date, Time, kWh, PowerFactor and kVARh are columns and the consumption of electricity are in each row.

Account	Date	Time	Catergroy	Building	Kwh	Power_Factor	KVRAH
---------	------	------	-----------	----------	-----	--------------	-------

1.3 Transform Dataset

1.3.1 Transpose Dataset

For mutli-apply the transformation to all the files, we decide to use R as tool to complete the transformation. The first step to transform dataset is to transpose the Units and time columns and combine them whit Account and Date. Then we will add the Category and building name which extracted from the file name to the dataset. Last but not lest, we derive the day, month and year from date column and combine them with other data. Using R-1 code, we accomplished this transformation and get the result. (R code will be attached to the end of the report)

	Time	Account	Date	kWh	Power_Factor	kVRAh	Catergroy	Building	Day	Month	Year
V5	0:05	26440621006	1/1/2014	3.740000	0.704842	3.764000	BCYF	CURLEY CMTY CTR	1	1	2014
V6	0:10	26440621006	1/1/2014	3.428000	0.694435	3.552000	BCYF	CURLEY CMTY CTR	1	1	2014
V7	0:15	26440621006	1/1/2014	3.612000	0.706715	3.616000	BCYF	CURLEY CMTY CTR	1	1	2014
V8	0:20	26440621006	1/1/2014	3.552000	0.704722	3.576000	BCYF	CURLEY CMTY CTR	1	1	2014
V9	0:25	26440621006	1/1/2014	3.568000	0.709489	3.544000	BCYF	CURLEY CMTY CTR	1	1	2014
V10	0:30	26440621006	1/1/2014	3.844000	0.708211	3.832000	BCYF	CURLEY CMTY CTR	1	1	2014
V11	0:35	26440621006	1/1/2014	3.504000	0.708723	3.488000	BCYF	CURLEY CMTY CTR	1	1	2014
V12	0:40	26440621006	1/1/2014	3.728000	0.712055	3.676000	BCYF	CURLEY CMTY CTR	1	1	2014
V13	0:45	26440621006	1/1/2014	3.544000	0.707506	3.540000	BCYF	CURLEY CMTY CTR	1	1	2014
V14	0:50	26440621006	1/1/2014	3.436000	0.708755	3.420000	BCYF	CURLEY CMTY CTR	1	1	2014
V15	0:55	26440621006	1/1/2014	3.552000	0.712703	3.496000	BCYF	CURLEY CMTY CTR	1	1	2014
V16	1:00	26440621006	1/1/2014	3.568000	0.701973	3.620000	BCYF	CURLEY CMTY CTR	1	1	2014
V17	1:05	26440621006	1/1/2014	3.708000	0.706344	3.716000	BCYF	CURLEY CMTY CTR	1	1	2014
V18	1:10	26440621006	1/1/2014	3.508000	0.707510	3.504000	BCYF	CURLEY CMTY CTR	1	1	2014
V19	1:15	26440621006	1/1/2014	3.532000	0.705507	3.548000	BCYF	CURLEY CMTY CTR	1	1	2014
V20	1:20	26440621006	1/1/2014	3.676000	0.708647	3.660000	BCYF	CURLEY CMTY CTR	1	1	2014
V21	1:25	26440621006	1/1/2014	3.440000	0.703826	3.472000	BCYF	CURLEY CMTY CTR	1	1	2014
V22	1:30	26440621006	1/1/2014	3.488000	0.707512	3.484000	BCYF	CURLEY CMTY CTR	1	1	2014
V23	1:35	26440621006	1/1/2014	3.428000	0.679234	3.704000	BCYF	CURLEY CMTY CTR	1	1	2014
V24	1:40	26440621006	1/1/2014	3.256000	0.675311	3.556000	BCYF	CURLEY CMTY CTR	1	1	2014
V25	1:45	26440621006	1/1/2014	3.308000	0.684822	3.520000	BCYF	CURLEY CMTY CTR	1	1	2014
V26	1:50	26440621006	1/1/2014	3.184000	0.675456	3.476000	BCYF	CURLEY CMTY CTR	1	1	2014

1.3.2 Merge Consumption by Date

Because the time scale is too small to use, we decided to sum the electricity consumption by date. Using code R-2, we get the daily consumption of electricity.

Account	Date	Catergroy	Building	Kwh	Power_Factor	KVRAH
26436731017	1/1/14	SCH	AGASSIZ	702.7	0.270236292	1252.03
26436731017	1/10/14	SCH	AGASSIZ	839.4	0.346441639	1856.579985
26436731017	1/11/14	SCH	AGASSIZ	656.03	0.301785562	1799.429991
26436731017	1/12/14	SCH	AGASSIZ	694.77	0.321830042	1481.42
26436731017	1/13/14	SCH	AGASSIZ	831.62	0.311717604	2100.919977
26436731017	1/14/14	SCH	AGASSIZ	737.94	0.24024841	2175.209982
26436731017	1/15/14	SCH	AGASSIZ	830.51	0.305826858	2266.959971
26436731017	1/16/14	SCH	AGASSIZ	766.03	0.273400344	2187.489971
26436731017	1/17/14	SCH	AGASSIZ	839.82	0.382614441	1727.87999
26436731017	1/18/14	SCH	AGASSIZ	564.42	0.258502583	1830.849981
26436731017	1/19/14	SCH	AGASSIZ	619.97	0.233812851	1325.71
26436731017	1/2/14	SCH	AGASSIZ	426.92	0.159191521	2047.239969
26436731017	1/20/14	SCH	AGASSIZ	699.62	0.280512254	1208.69
26436731017	1/21/14	SCH	AGASSIZ	776.01	0.288382017	1962.089979
26436731017	1/22/14	SCH	AGASSIZ	1287.659988	0.474746069	1501.959981
26436731017	1/23/14	SCH	AGASSIZ	1822.699983	0.609645444	1445.369984
26436731017	1/24/14	SCH	AGASSIZ	817.73	0.283963386	2277.76996

1.4 Combining Columns

Firstly, we import the data from: <https://www.wunderground.com/>

In the type of csv format;

EDT, 最高温度F, 平均温度F, 最低温度F, Max Dew PointF, MeanDew PointF, Min DewpointF, Max 湿度, Mean 湿度, Min 湿度, Max 海平面气压In, Mean 海平面气压In, Min 海平面气压In, Max 能见度Miles, Mean 能见度Miles, Min 能见度Miles, Max 风速MPH, Mean 风速MPH, Max 瞬间风速MPH, 降水量In, CloudCover, 活动, WindDirDegrees
 2014-3-1, 36, 26, 16, 22, 7, -10, 51, 40, 28, 30.45, 30.31, 30.10, 10, 10, 10, 13, 6, 22, 0.00, 4, , 221
 2014-3-2, 40, 35, 30, 25, 20, 14, 69, 59, 48, 30.08, 30.02, 29.98, 10, 10, 6, 16, 7, 18, T, 8, 中雪, 258
 2014-3-3, 30, 23, 16, 12, -1, -9, 50, 39, 28, 30.30, 30.14, 30.00, 10, 10, 10, 22, 12, 26, 0.00, 6, , 339
 2014-3-4, 31, 22, 12, 2, -5, -9, 42, 32, 21, 30.39, 30.31, 30.26, 10, 10, 10, 13, 8, 17, 0.00, 5, , 280
 2014-3-5, 29, 25, 20, 19, 14, 0, 81, 55, 29, 30.50, 30.33, 30.25, 10, 9, 2, 16, 10, 20, 0.00, 8, 中雪, 7
 2014-3-6, 25, 18, 11, 9, 1, -10, 67, 46, 24, 30.65, 30.58, 30.51, 10, 10, 10, 16, 7, 22, 0.00, 1, , 23
 2014-3-7, 34, 25, 16, 27, 18, 9, 78, 64, 49, 30.51, 30.27, 29.96, 10, 10, 10, 12, 5, 15, 0.00, 5, , 50
 2014-3-8, 55, 41, 27, 30, 23, 17, 81, 54, 26, 29.94, 29.80, 29.69, 10, 10, 10, 20, 8, 25, 0.00, 4, , 327
 2014-3-9, 43, 37, 30, 17, 8, 1, 51, 35, 18, 29.94, 29.90, 29.87, 10, 10, 10, 21, 11, 25, 0.00, 5, , 315
 2014-3-10, 44, 36, 28, 31, 23, 8, 85, 60, 35, 29.89, 29.79, 29.65, 10, 7, 1, 15, 8, 20, 0.03, 8, 中雨-中雪, 216
 2014-3-11, 60, 47, 34, 34, 30, 25, 92, 60, 28, 29.73, 29.59, 29.52, 10, 8, 5, 16, 9, 22, 0.00, 5, , 254
 2014-3-12, 46, 42, 37, 43, 37, 34, 89, 80, 70, 29.73, 29.40, 29.03, 10, 5, 1, 15, 5, 20, 0.03, 8, 中雨, 27

Then we transform the date data in to the same type of that in Boston Energy and combine them together in R-sql

	H	I	J	K	L	M	N
y	Date	Building	Day	Month	Year	Date00	Mean
	01/01/201	AGASSIZ	1	1	2014	01/01/201	
	01/01/201	ArtsJan	1	1	2014	01/01/201	
	01/01/201	BOSTON L	1	1	2014	01/01/201	
	01/01/201	Blackstone	1	1	2014	01/01/201	
	01/01/201	COURT ST	1	1	2014	01/01/201	
	01/01/201	DEVER	1	1	2014	01/01/201	
	01/01/201	DORCHEST	1	1	2014	01/01/201	
	01/01/201	EEC	1	1	2014	01/01/201	
	01/01/201	EECEASTB	1	1	2014	01/01/201	
	01/01/201	HARVARD	1	1	2014	01/01/201	
	01/01/201	HEADQUA	1	1	2014	01/01/201	
	01/01/201	HYDE PAR	1	1	2014	01/01/201	
	01/01/201	LILLA FRED	1	1	2014	01/01/201	
	01/01/201	MGMT	1	1	2014	01/01/201	
	01/01/201	MURPHY	1	1	2014	01/01/201	

```
second<-sqldf("select Account, sum(Kwh), avg(Power_Factor), sum(KVRAH), Categroy, Date, Building, Day,
dayData<-sqldf("select * from second join y on second.Date = y.Date00")
```

After we collect weather info from web source, we using sqldf() function combine different functions together.

Account	Kwh	avg.Power_Factor	sum.KVRAH	Categroy	Date	Building	Day	Month	Year	Date
26436731017	702.700	0.2702363	1252.030	SCH	01/01/2014	AGASSIZ	1	1	2014	01/0
26441521007	2338.313	0.8855198	1323.810	SCH	01/01/2014	ArtsJan	1	1	2014	01/0
26441591000	982.944	0.8173025	690.588	SCH	01/01/2014	BOSTON LATIN ANN	1	1	2014	01/0
26430841010	2173.590	1.0000000	0.000	SCH	01/01/2014	Blackstone Jan	1	1	2014	01/0
26432071004	3721.980	0.9163112	1632.690	SCH	01/01/2014	COURT ST	1	1	2014	01/0
26440191000	882.000	0.9123518	404.070	SCH	01/01/2014	DEVER	1	1	2014	01/0
26436491000	122.528	0.5514756	166.652	SCH	01/01/2014	DORCHESTER HIGH	1	1	2014	01/0
26435881003	562.890	0.7616985	478.875	SCH	01/01/2014	EEC	1	1	2014	01/0

1.5 Clean NA and Zero Value Data

```
nozeroData<-sqldf("select * from newdata where kwh != 0")
dayData<- "select "
is.na(dayData)
newdata<-na.omit(dayData)
```

1.6 The Final Dataset

After pre-process of the dataset, we get the final dataset which will be used in training and evaluating model.

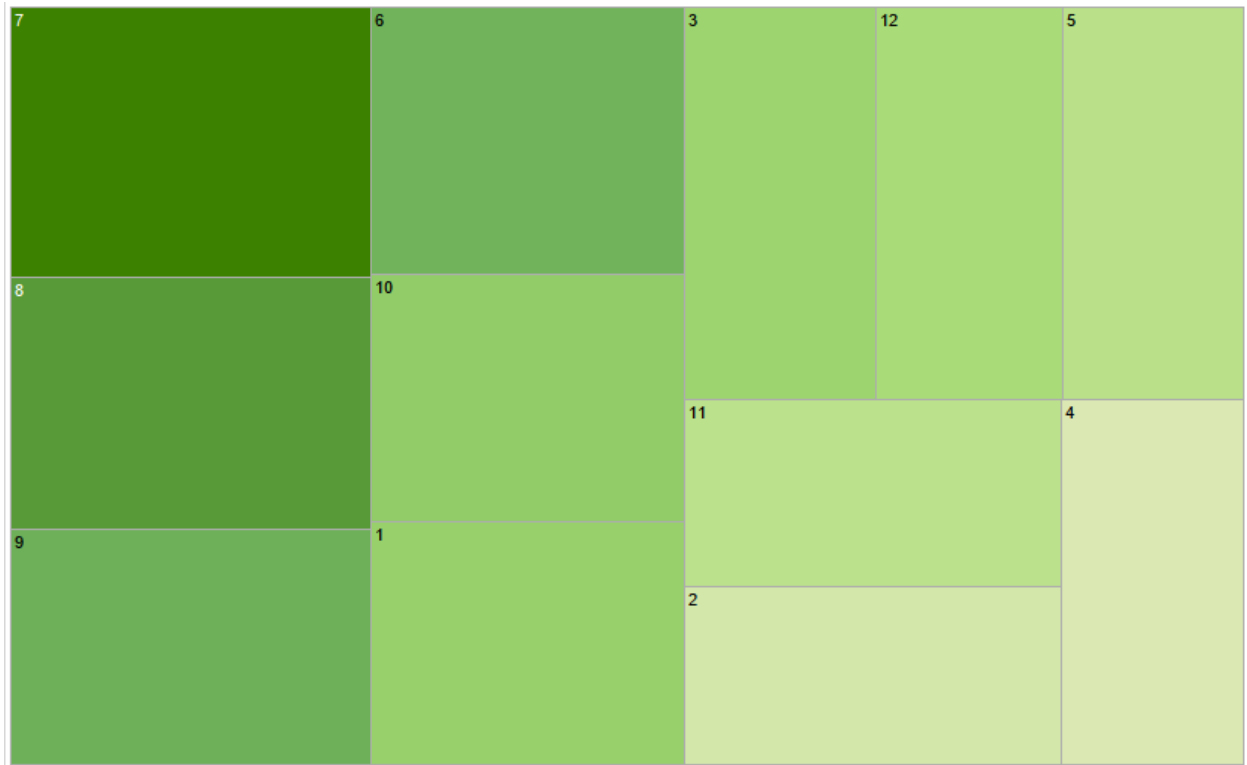
Account	Date	Catergroy	Building	TemperatureF	Humidity	Events	workingDay	Holiday	Kwh	Power_Factor	KVRAH
26436731017	1/1/14	SCH	AGASSIZ	24	43		1	1	702.7	0.270236292	1252.03
26436731017	1/10/14	SCH	AGASSIZ	28	80	Fog-Snow	1	0	839.4	0.346441639	1856.579985
26436731017	1/11/14	SCH	AGASSIZ	47	93	Fog-Rain-Thunderstorm	0	0	656.03	0.301785562	1799.429991
26436731017	1/12/14	SCH	AGASSIZ	46	67	Rain	0	0	694.77	0.321830042	1481.42
26436731017	1/13/14	SCH	AGASSIZ	41	63		1	0	831.62	0.311717604	2100.919977
26436731017	1/14/14	SCH	AGASSIZ	47	90	Fog-Rain	1	0	737.94	0.24024841	2175.209982
26436731017	1/15/14	SCH	AGASSIZ	42	76		1	0	830.51	0.305826858	2266.959971
26436731017	1/16/14	SCH	AGASSIZ	36	93	Fog	1	0	766.03	0.273400344	2187.489971
26436731017	1/17/14	SCH	AGASSIZ	40	65	Fog	1	0	839.82	0.382614441	1727.87999
26436731017	1/18/14	SCH	AGASSIZ	35	87	Fog-Rain-Snow	0	0	564.42	0.258502583	1830.849981
26436731017	1/19/14	SCH	AGASSIZ	33	73	Fog-Snow	0	0	619.97	0.233812851	1325.71
26436731017	1/2/14	SCH	AGASSIZ	14	69	Fog-Snow	1	0	426.92	0.159191521	2047.239969
26436731017	1/20/14	SCH	AGASSIZ	34	56		1	1	699.62	0.280512254	1208.69
26436731017	1/21/14	SCH	AGASSIZ	19	58	Snow	1	0	776.01	0.288382017	1962.089979
26436731017	1/22/14	SCH	AGASSIZ	13	60	Snow	1	0	1287.659988	0.474746069	1501.959981
26436731017	1/23/14	SCH	AGASSIZ	14	45		1	0	1822.699983	0.609645444	1445.369984
26436731017	1/24/14	SCH	AGASSIZ	13	43		1	0	817.73	0.283963386	2277.76996
26436731017	1/25/14	SCH	AGASSIZ	27	65	Snow	0	0	2365.959977	0.814090347	717.499999

2.Visualization Analytic

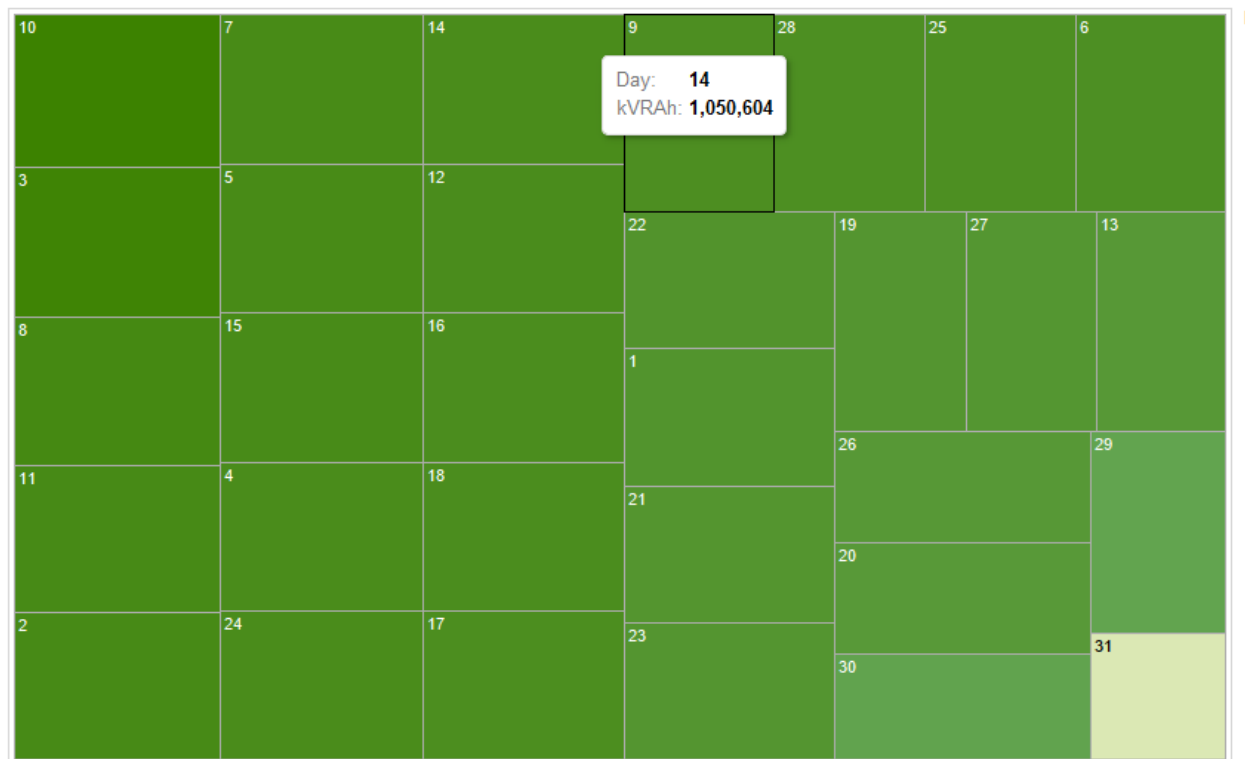
2.1 Marco Analysis

Total Energy Consumption – Month/Day

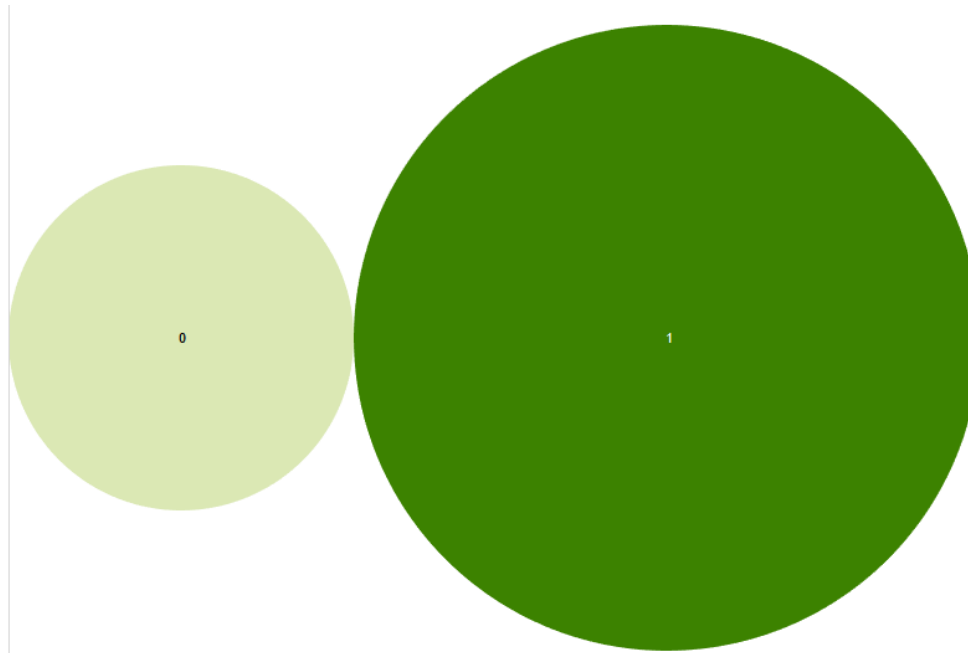
In the graph, it's very clear that the macro energy consumption changes month by month. This will help us understand the relationship between different month and their energy consumptions.



The graph below displays the range of daily usage with day's change in a month.

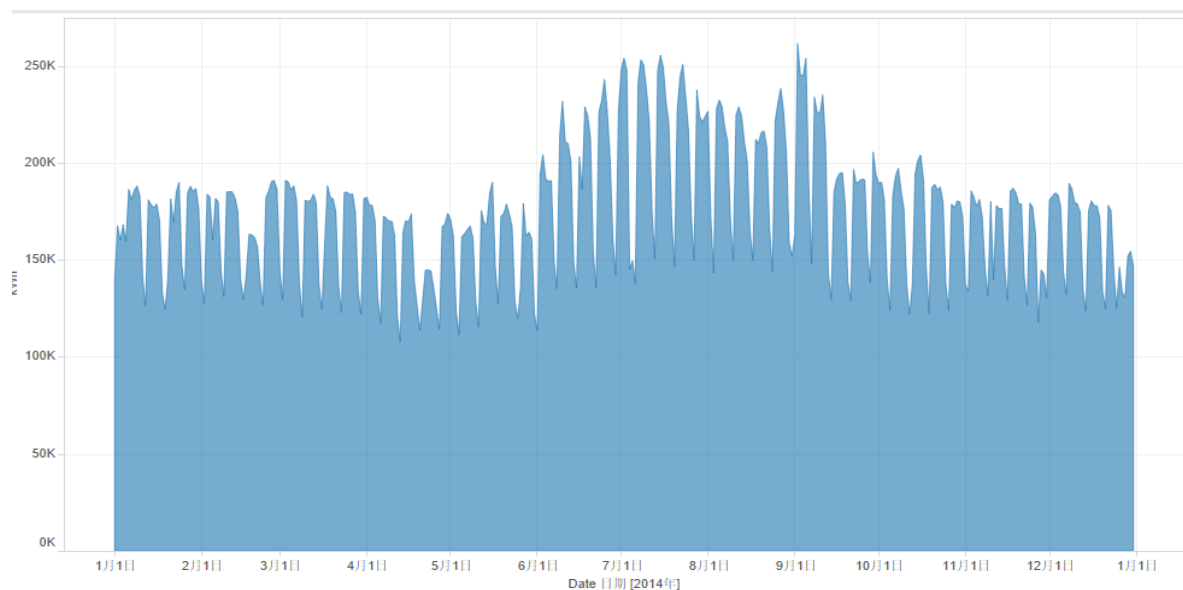


The Marco Energy Consumption with working day and no-working day

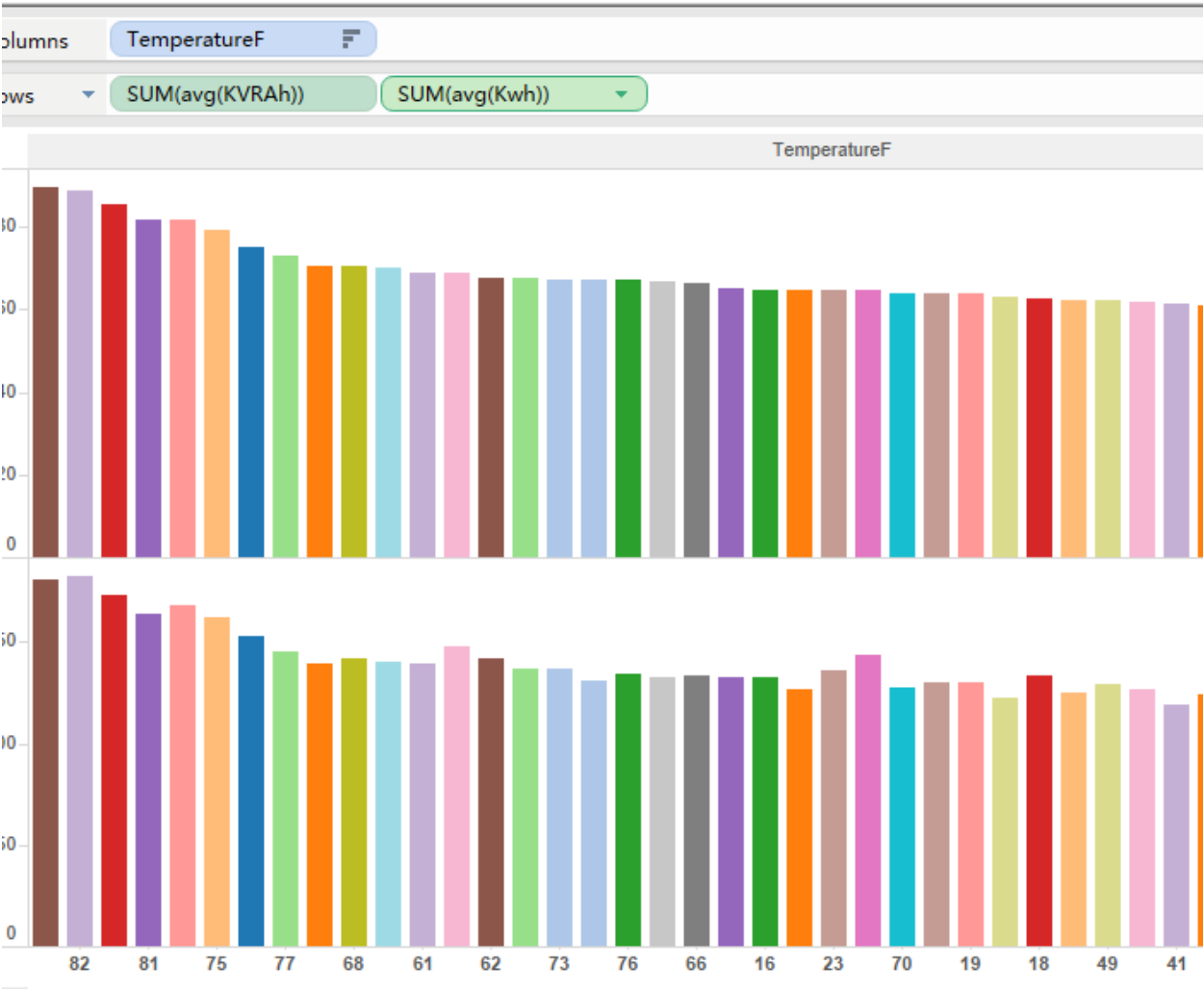


Energy Consumption – Total Value Through the Year

According to the graph we can easily distinguish the changes from month to month in one year, and changes from day by day in each month. The next step for our analysis is to get the whole trend of energy consumptions in the whole year on the unit of date.



Marco Energy Consumption – Temperature/Humidity

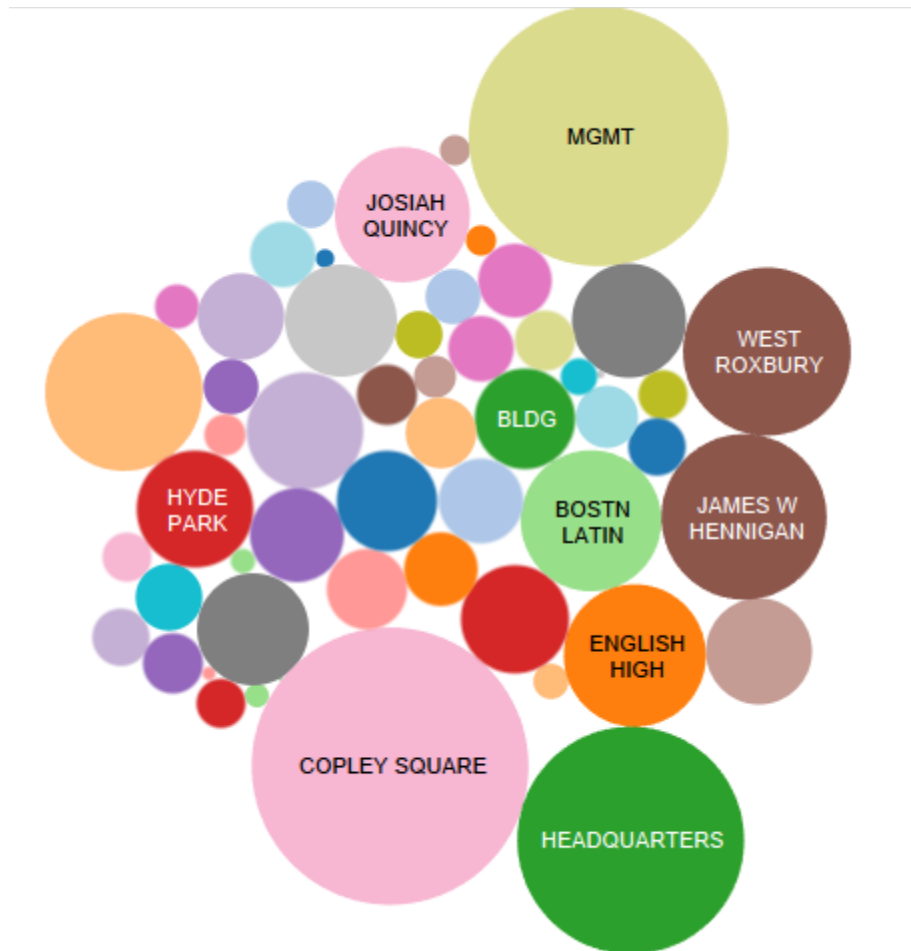




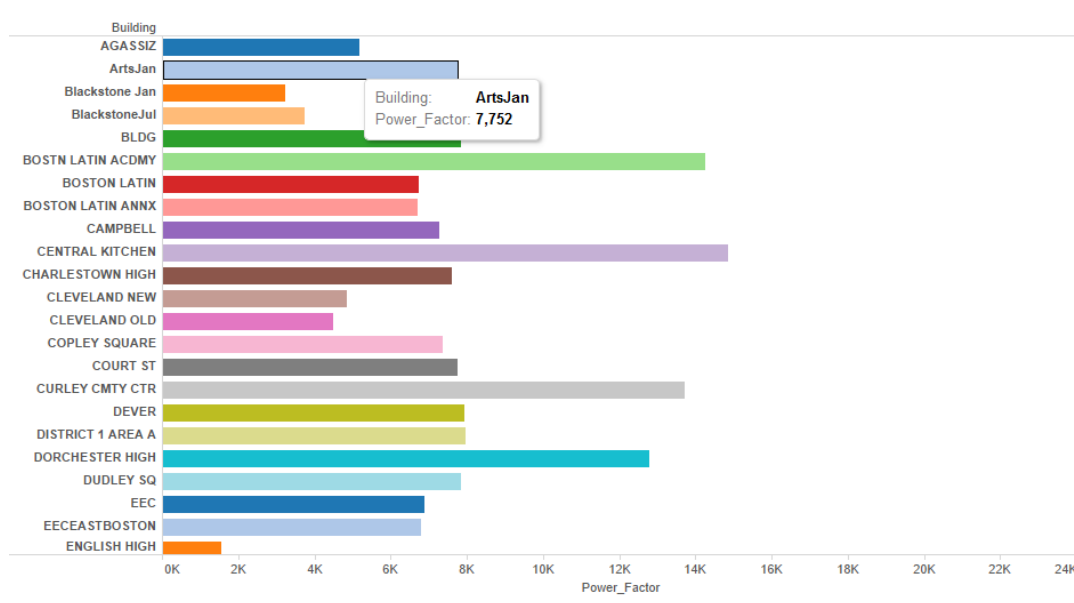
Micro-Analysis

After analysis the Marco trend of energy consumption, we now focus more on micro-analysis and their energy consumptions related to temperature, humidity and so on.

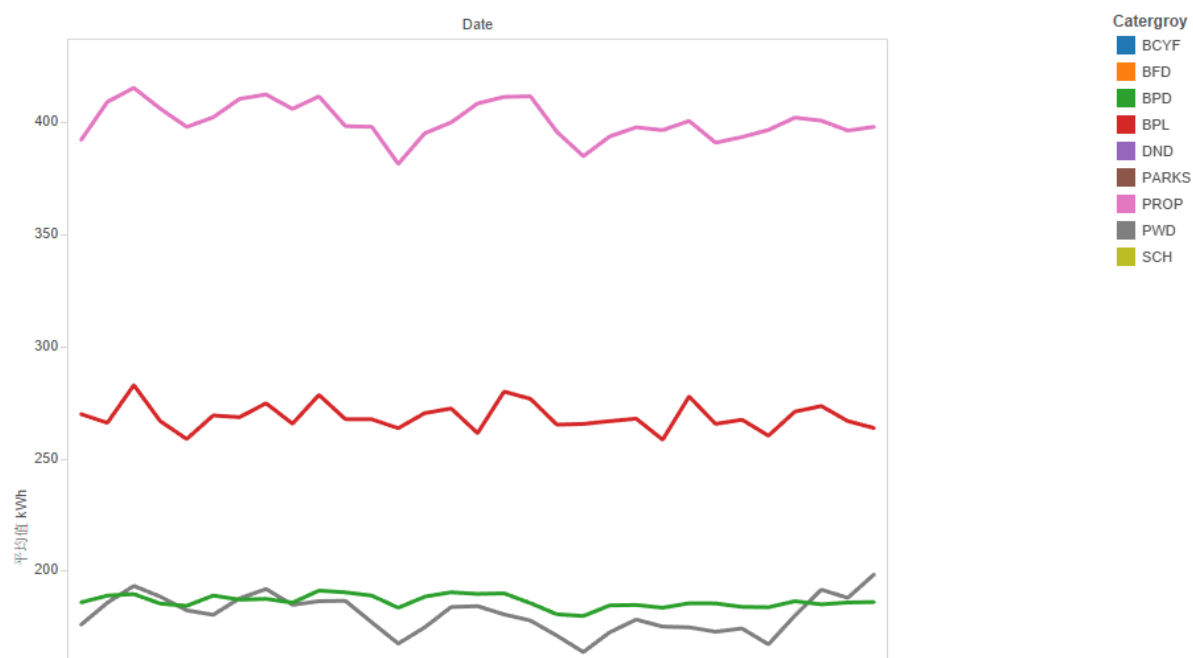
First diagram shows the energy consumption differs area by area.



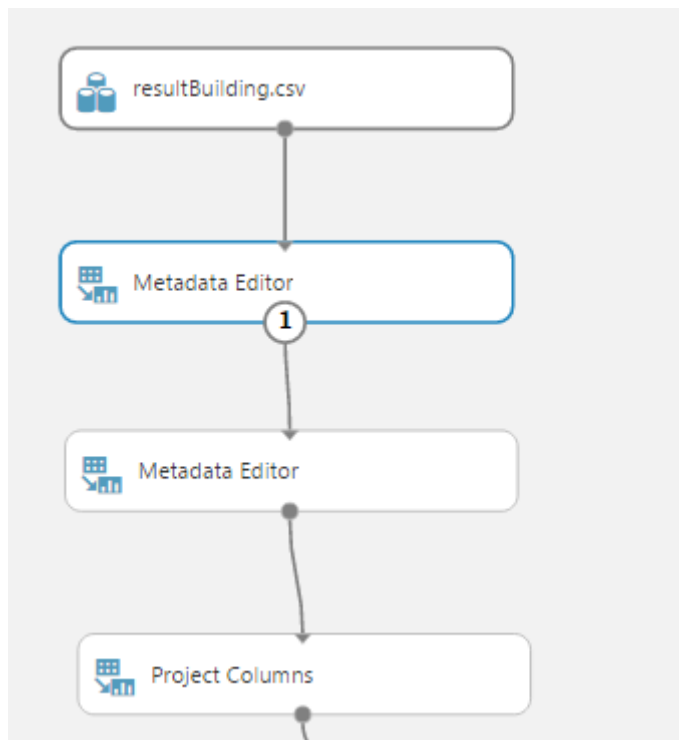
The box-graph also offer us a clear visual for building's power-factors.



There is also a diagram display the use of electricity of each day, though difference exit in each building, while compare to themselves, the graph seems to be stable.



3.Model Building



3.1 Transform and pre-clean the data for analysis

Properties Project

Metadata Editor

Column

Selected columns:
Column names:
Account

Launch column selector

Data type
String

Categorical
Unchanged

Fields
Unchanged

New column names

Properties Project

Metadata Editor

Column

Selected columns:
Column names:
Building,Catergroy,Holida

Launch column selector

Data type
Unchanged

Categorical
Make categorical

Fields
Unchanged

New column names

3.2 Define category variables

For this exercise, the column that should be cast as categorical are: Building, Category, Holiday, working Day, Events. To cast these columns, drag in the metadata editor. Specify the columns to be cast, then change the “Categorical” parameter to “Make categorical”

For no-categorical values. We define the value of Account in type of String and stay in its original type.

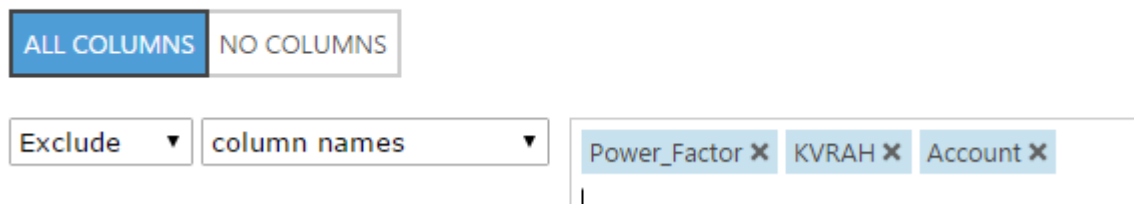
3.3 Choosing the Variables for Training

We have already define the type of different variables. Now, we should continue by identifying columns that add little-to-no value for predictive modeling. These columns will be dropped.

The first, most obvious candidate to be dropped is Account. All information provided by Account can be replaced by the Buildings. Therefore, the keys could have been completely needlessness and may add false correlations or noise to our model.

The second candidate for removal is the Power_Factor column and KVRAH. Normally, there can be generated together with Kwh, their strong relationship may have bad effect for our prediction.

Begin With



ALL COLUMNS NO COLUMNS

Exclude ▼ column names ▼

Power_Factor ✕ KVRAH ✕ Account ✕

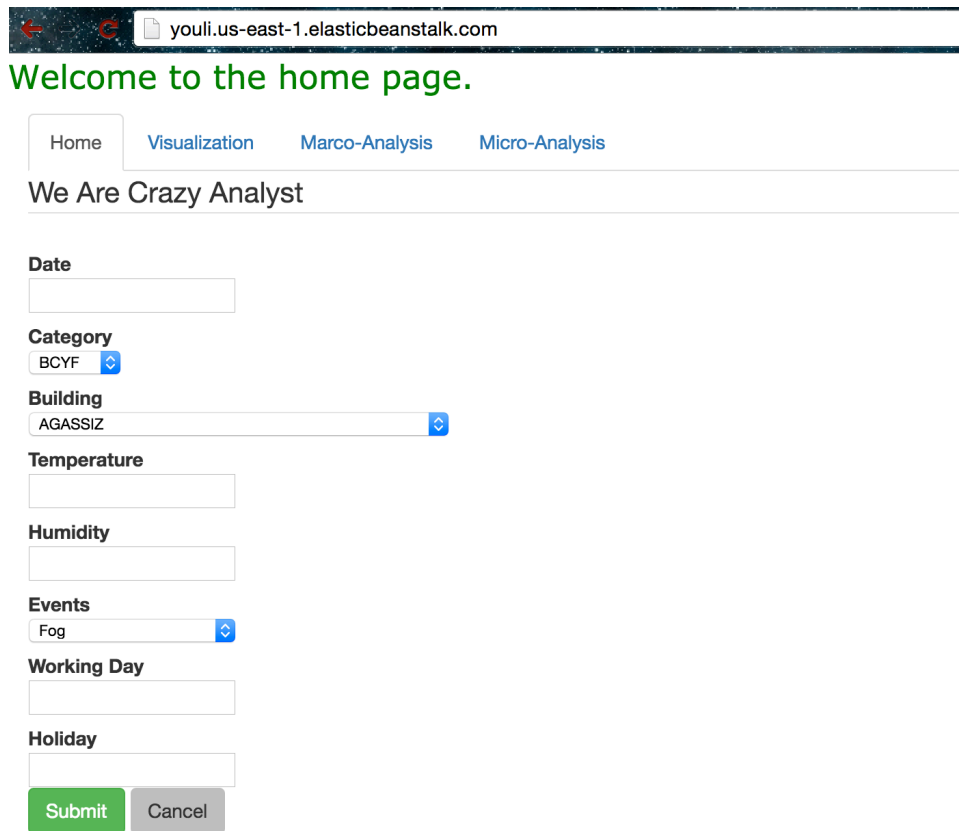
3.4 Split the Data and Training the model

4. Web Service

4.1 The Web Architecture

We built the front end web page in Bootstrap framework. The web page used a form to collect the input information from users, then the form submit the input to the backend, which is implemented by Spring MVC. Spring MVC parsed the data into JSON format and then sent the data to the RESTful service. The Spring MVC also received the response from the Microsoft Azure, parsed the data and picked the output, then sent the output to the front page to users' side.

4.2 Front End



The screenshot shows a web browser window with the address bar displaying 'youli.us-east-1.elasticbeanstalk.com'. Below the address bar, there is a green heading 'Welcome to the home page.' followed by a navigation bar with four links: 'Home', 'Visualization', 'Marco-Analysis', and 'Micro-Analysis'. The 'Home' link is highlighted. Below the navigation bar, the text 'We Are Crazy Analyst' is displayed. The main content area contains a form with several input fields and dropdown menus: 'Date' (text input), 'Category' (dropdown menu with 'BCYF' selected), 'Building' (dropdown menu with 'AGASSIZ' selected), 'Temperature' (text input), 'Humidity' (text input), 'Events' (dropdown menu with 'Fog' selected), 'Working Day' (text input), and 'Holiday' (text input). At the bottom of the form, there are two buttons: 'Submit' (green) and 'Cancel' (gray).

We use Bootstrap to create front end, the visualization tag is the link to the tableau page. And we use a simple form to collect the user's input.

4.3 Spring MVC

We use Spring MVC to parse the user's input data into JSON format. And then sent the request to the Azure model.

4.3.1 Set the header for building Connections

According to the API document, we have to add header to the request.


```

HttpPost createConnectivity(String restUrl)
{
    HttpPost post = new HttpPost(restUrl);
    String auth=new StringBuffer(username).append(":").append(password).toString();
    byte[] encodedAuth = Base64.encodeBase64(auth.getBytes(Charset.forName("US-ASCII")));
    String authHeader = "Bearer " + "e0oswmFDkUElyZ+Tb8UAuzdW2TwPdU/7VWhaHbjgNLe/4Y6GLrwg8YoQCUZ5fUg!";
    post.setHeader("AUTHORIZATION", authHeader);
    post.setHeader("Content-Type", "application/json");
    post.setHeader("Accept", "application/json");
    post.setHeader("X-Stream" , "true");
    return post;
}

```

Firstly, we set the post's header by setting parameters and connect the author Header to our API. The format of our data are setting in JSON.

```

@SuppressWarnings("finally")
String executeReq(String jsonData, HttpPost httpPost)
{
    String predict = null;
    try{
        predict = executeHttpRequest(jsonData, httpPost);
    }
    catch (UnsupportedEncodingException e){
        System.out.println("error while encoding api url : "+e);
    }
    catch (IOException e){
        System.out.println("ioException occured while sending http request : "+e);
    }
    catch(Exception e){
        System.out.println("exception occured while sending http request : "+e);
    }
    finally{
        httpPost.releaseConnection();
        return predict;
    }
}

```

```

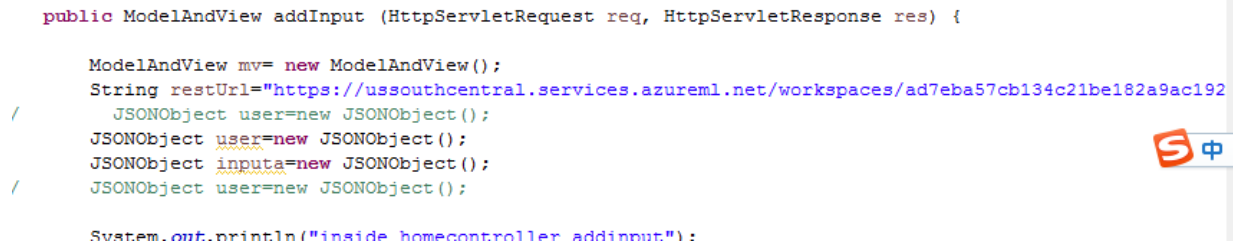
String executeHttpRequest(String jsonData, HttpPost httpPost) throws UnsupportedEncodingException, IOException
{
    HttpResponse response=null;
    String line = "";
    StringBuffer result = new StringBuffer();
    httpPost.setEntity(new StringEntity(jsonData));
    HttpClient client = HttpClientBuilder.create().build();
    response = client.execute(httpPost);
    System.out.println("Post parameters : " + jsonData );
    System.out.println("Response Code : " + response.getStatusLine().getStatusCode());
    response.getEntity().getContent()
    /
    BufferedReader reader = new BufferedReader(new InputStreamReader(response.getEntity().getContent()))
    while ((line = reader.readLine()) != null){ result.append(line); }
    System.out.println(result.toString());
}

```

4.3.2 Executing process.

In the Home Controller, we firstly direct get the data from front page and transform it into JSON format.

```
public ModelAndView addInput (HttpServletRequest req, HttpServletResponse res) {  
  
    ModelAndView mv= new ModelAndView();  
    String restUrl="https://ussouthcentral.services.azureml.net/workspaces/ad7eba57cb134c21be182a9ac192  
/  
    JSONObject user=new JSONObject();  
    JSONObject user=new JSONObject();  
    JSONObject inputa=new JSONObject();  
/  
    JSONObject user=new JSONObject();  
  
    System.out.println("inside homecontroller addinput");
```



Since we have already define the Httppost method, what we need to do next is to send the data into Azure through our API key. Then print the value of its output.

```
        System.out.println("test");  
//        String jsonData=user.toString();  
        HttpPostReq httpPostReq=new HttpPostReq();  
        HttpPost httpPost = httpPostReq.createConnectivity(restUrl);  
        String predictOutput = httpPostReq.executeReq(userInput, httpPost);  
        System.out.println("this is in the home controller -> "+predictOutput);  
//        return null;  
        mv.addObject("output", predictOutput);  
        mv.setViewName("try2");  
        return mv;  
    }
```

Attachment

R-1

```
#get catergroy and building
listTitle <- unlist(strsplit(filename, "\\."))

listTitle

Catergroy <- unlist(strsplit(listTitle[1], "-"))[2]

Catergroy

building <- listTitle[2]

building

#transpose column sucessful
rowdata <- read.csv(filename,header = FALSE)

titlerow <- rowdata[1,]

titlerow1 <- titlerow[,c(-1,-2,-3,-4)]
titlecolumn <- t(titlerow1)
dateTemp <- rowdata[1,2]
matrixTemp <- titlecolumn
firstTime <- TRUE

for (i in 2:nrow(rowdata)){
  if(rowdata[i,2] != dateTemp){
    if(!firstTime){
      result <- rbind(result,matrixTemp)
    }
    if (i != 2 && firstTime){
      result <- matrixTemp
      firstTime <- FALSE
    }

    matrixTemp <- titlecolumn
    matrixTemp <- cbind(matrixTemp, matrix(rep(rowdata[i,1],length(titlecolumn))))
    matrixTemp <- cbind(matrixTemp, matrix(rep(rowdata[i,2],length(titlecolumn))))
  }

  matrixTemp <- cbind(matrixTemp,t(rowdata[i,c(-1,-2,-3,-4)]))

  dateTemp = rowdata[i,2]

  if(i == nrow(rowdata)){
    result <- rbind(result,matrixTemp)
  }
}

#Add Catergory and Building to dataset
numRows <- nrow(result)

result <- cbind(result, matrix(rep(Catergroy,numRows)))

result <- cbind(result, matrix(rep(building,numRows)))

#Derive day month year
```

```

listDate <- result[,3]

dayFun <- function(x) {
  listTemp <- unlist(strsplit(x,"/"))
  listTemp[2]
}

monthFun <- function(x){
  listTemp <- unlist(strsplit(x,"/"))
  listTemp[1]
}

yearFun <- function(x){
  listTemp <- unlist(strsplit(x,"/"))
  listTemp[3]
}

listDay <- lapply(listDate, dayFun)

listMonth <- lapply(listDate, monthFun)

listYear<- lapply(listDate, yearFun)

result <- cbind(result,listDay)

result <- cbind(result,listMonth,listYear)

```

R-2

```

uniqueRow<- data.frame(a = 1:nrow(result))
rownames(result)<- uniqueRow[,1]

result <- data.frame(result)

result <- sqldf("select Account, Date, Catergroy, Building,sum(Kwh) as Kwh,AVG(Power_Factor) as Power_Factor, sum(KVRAH) as KVRAH from
reuslt group by Date")

```