Assignment 1: Power Calendar function

Implementation

My get_hours function is mainly designed via the following thoughts:
1. enumerate all days in the date range as **set** U 2. enumerate all the weekdays in the date range as **set** U_1 3. enumerate all the holidays in the date range as **set** U_2

Thus, we could derive peak type from set operations.

For example, 'onpeak' days are $U_1 - U_2$. Since each peak day has 16 peak hours, the number of onpeak hour is simply multiple 16 by the size of $U_1 - U_2$. In addition, '2x16H' is HE7 to HE22 for the weekend and the NERC holiday, which is obtained by 16 multiplies the size of $(U - U_1) \cup U_2$.

There are some interesting details that worth being discussed.

Holiday adjustment:

Under some scenarios, the holiday is celebrated on Sunday. In such case, this holiday will be observed on next Monday. We need to carefully adjust for every possible holiday date.

Daylight-saving adjustment:

Whenever there is a start date of daylight-saving between the time rage, we need to minus one hour to our total hours, and vice versa. It doesn't matter if there are multiple start / end of daylight-saving days because they will automatically cancelled out. Note that this adjustment only applies to non MISO and peak type of 'flat', 'offpeak' and '7x8' because this will be the only cases that start / end hour of daylight-saving (2am) cover.

Conclusion

The 'get_houts' functions takes a list of iso, peak.type and period as input, calculates and returns the number of corresponding hours. The jupyter notebook 'Assignment 1' shows an example using this function. It is proved that our results are the same with that on the reference website.

Assignment 2: Meter Data formatting

Merge files

First, I resampled the hourly-basis data in **new.app4.cvs** file and transfered it to be one appliance's hourly electricity consumptions. Next, I converted the 'time' column of **new.app4.csv** and 'Date/Time' column of **USA_AL_Auburn-Opelika.AP.722284_TMY3_BASE.csv** to be datetime series. Note that since the datetime package embedded functions only take hours within the range

$0 \sim 23$, so I implemented a function to convert 24:00:00 to be 00:00:00 at the next day.

After the above operations, both files can be merged easily on their time columns. The 'Appl:Electricity [kW](Hourly)' represents one appliance's hourly electricity consumption in kW unit, and 'Total:Electricity [kW](Hourly)' is the sum of all columns.

Abnormal Points

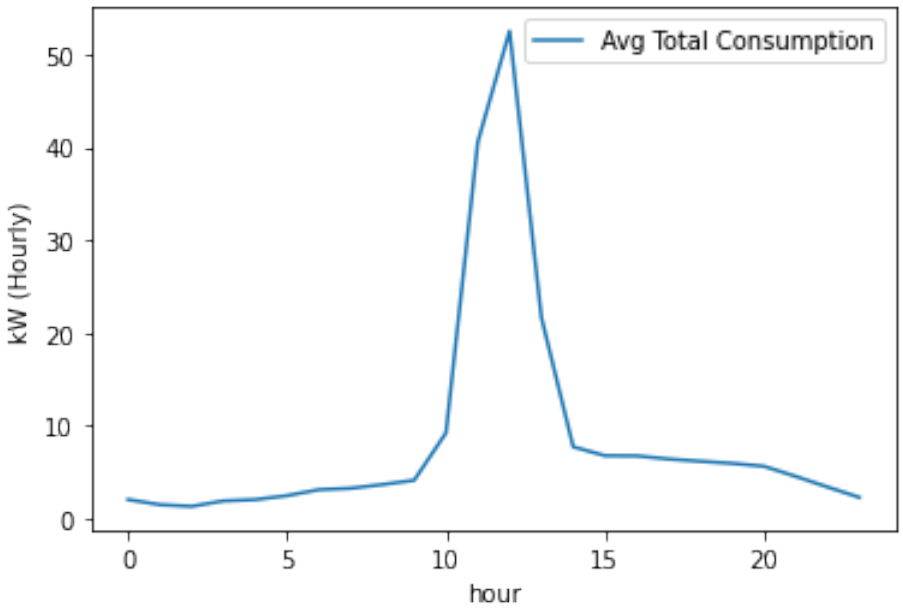
According to the distribution plots for each of the column, I observed that the 'Heating:Electricity [kW](Hourly)' and 'Heating:Gas [kW](Hourly)' always have zero values. And the appliance's electricity consumptions and total consumptions have some outliers. In order to check the characteristics of these abnormal points, I filtered them out by the $3 - \sigma$ limits, found that there is 3.835% points larger than $mean + 3\dot{s}td$ of the total consumptions. Counting the number of points by hour, I got the following table:

	hour	**count**
0	9	1
1	10	4
2	11	31
3	12	49
4	13	9
5	14	1
6	15	1

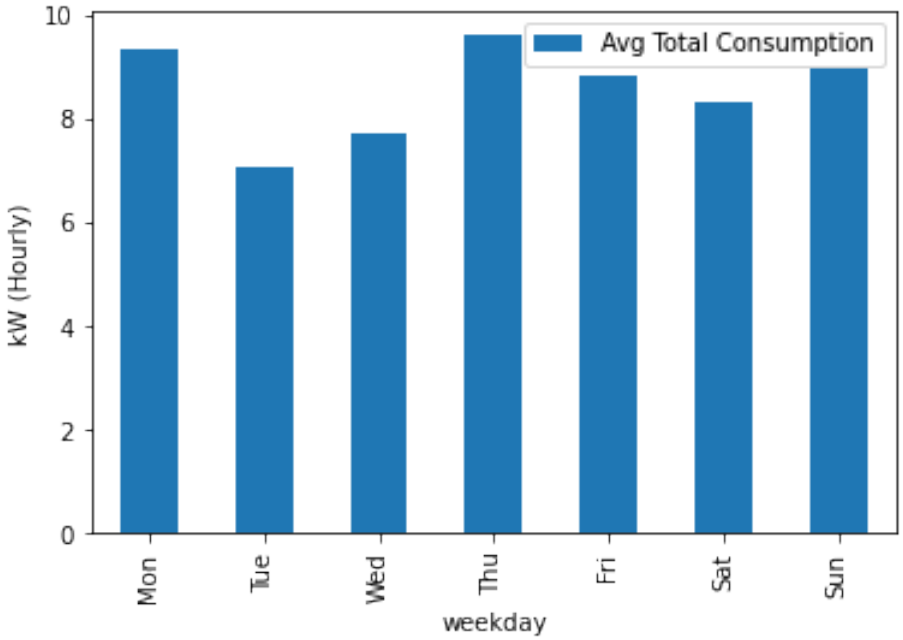
We can see that most of the time, extremely large consumptions happened during 11:00 - 13:00. I also checked the distribution of abnormal points on weekdays, but did not see obvious variation rules.

Patterns by Hour/Weekday/Month

Separately, I grouped the total electricity consumptions by hour, weekday and month.

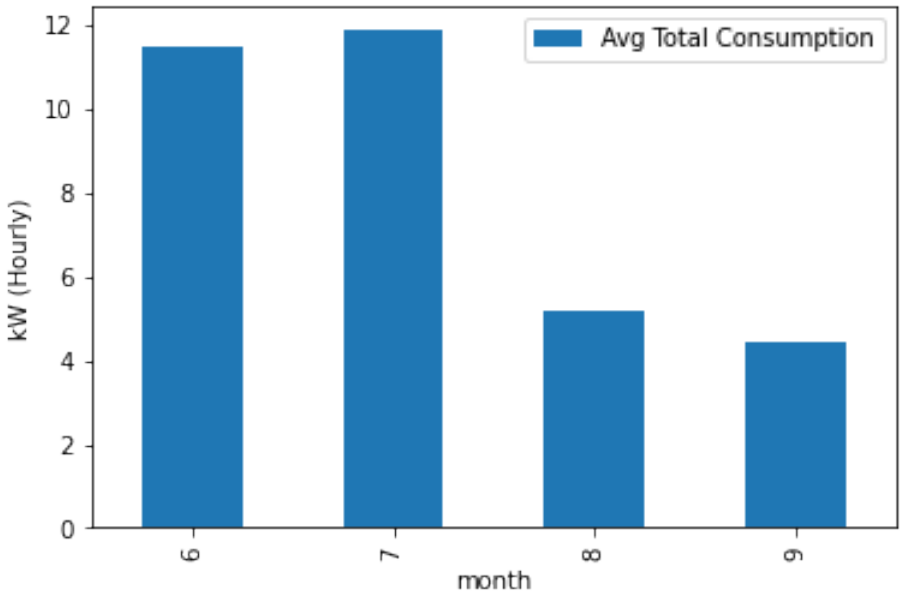


From the above plot, we can see the average hourly electricity consumptions are on peak between 10:00 - 15:00.



From the distribution of total consumptions on weekday, we could see the Tuesday

and Wednesday have relatively low consumptions, whereas Monday, Thursday and Aunday have high consumption level.



According to this figure, the average total electricity consumptions drop by more than 50% on August.

Conclusion

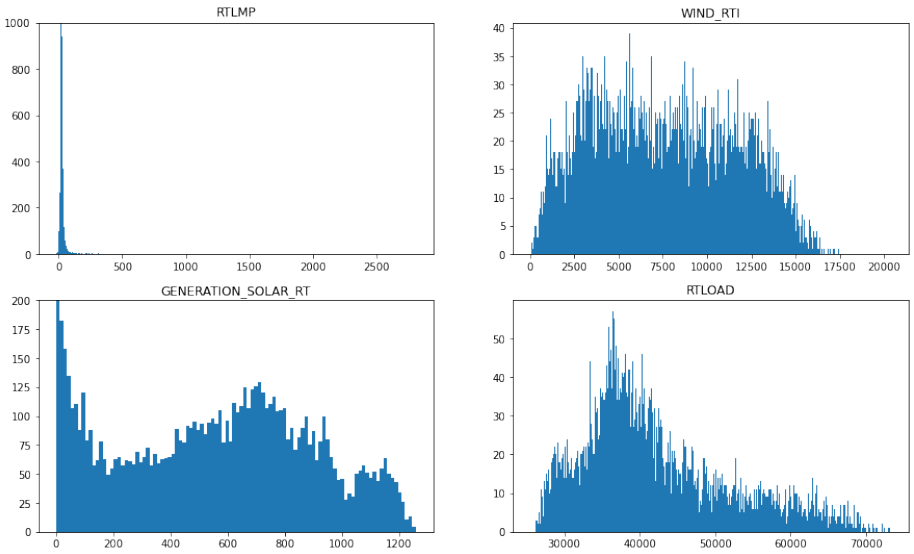
- The heating electricity and gas are always zero from June to August according to the provided data file.
- The extremely large electricity consumption usually happens during noon, from 11:00 to 13:00.
- The peak hours of electrcity consumption is 10:00 - 15:00. From the perspective of months, the power consumption on June & July is twice as much as that on August & Septemper.

Assignment 3: EDA and forecast model

Data Pre-processing

I loaded the data file for Assignment 3 and printed the basic statistics values and distribution plots for each time series.

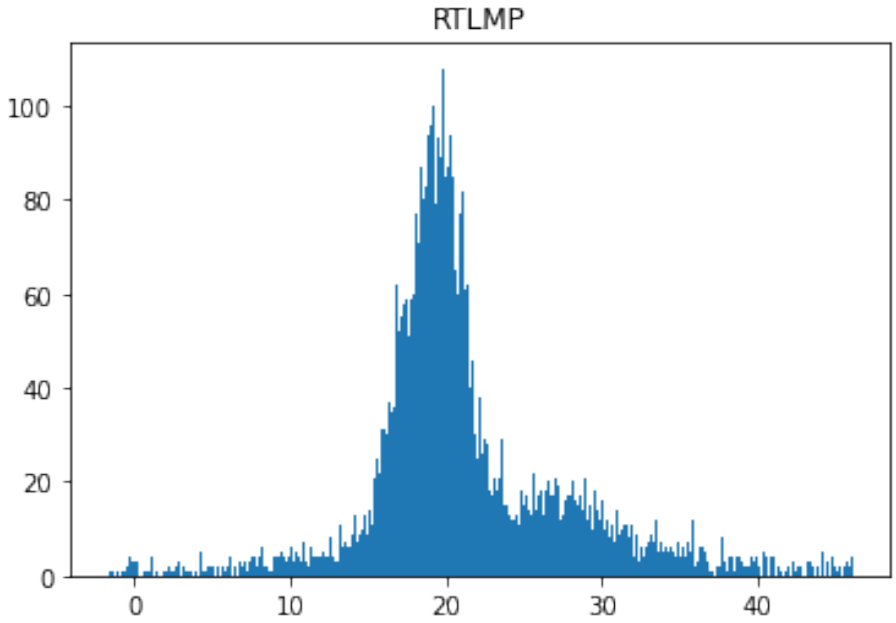
	HB_NORTH (RTLMP)	ERCOT (WIND_RTI)	ERCOT (GENERATION_SOLAR_RT)	ERCOT (RTLOAD)
count	14981.000000	14981.000000	14981.000000	14981.000000
mean	25.768281	7532.053149	291.997647	42372.917557
std	46.370881	3992.742693	370.929008	9874.696215
min	-17.860000	54.440000	0.000000	25566.511248
25%	18.042500	4134.730000	0.000000	35432.588663
50%	20.057500	7281.370000	22.150000	39935.131628
75%	25.030000	10851.640000	608.660000	47871.380668
max	2809.357500	20350.400000	1257.540000	73264.662123



From the summary table and plots, I have the following insights for each variable:

- RTLMP: The value has a high variance and many outliers, we need to remove these outliers before fitting.
- WIND_RTI: This value is symmetric, which suggest we can use the normal distribution to approimate the data.
- GENERATION_SOLAR_RT: Over 1/4 of the value is zero, and the data has a distribution of two peaks. And it has much smaller value than WIND_RTI, which suggests we can **binerize the data** by some threshold.
- RTLOAD: The value is larger than zero and has a long tail. We can use gamma distribution to approximate the data.

Since the RTLMP is highly skewed, I filtered out the values that is away from $mean + 3\sigma$. The resulting plot is shown below:



RTLMP Prediction

I used the linear regression model to predict the RTLMP. The predicting variables are:

- RTLOAD
- WIND_RTI
- PEAKTYPE (category)
- GENERATION_SOLAR_RT (discritized)
- MONTH (category)
- HOUR (category)

Note that after discretion, the 'GENERATION_SOLAR_RT' turns to be a binary variable, where 0 represents it has solar power smaller than 200 and 1 otherwaise.

The training and testing data is split by 4:1. The training data is fed to fit the model, and the testing data is used for testing the model performance.

Conclusion

According to the summary table of OLS regresssion, the variables of solar generation and peak types bear relatively large coefficient. The category of month also shows statistical significance for predicting RTLMP. Moreover, the R^2 for our regression model is 0.706. A complete description of coefficients for each variable is listed on the jupyter notebook 'Assignment 3'.

OLS Regression Results

Dep. Variable:	HB_NORTH (RTLMP)	R-squared:	0.706
Model:	OLS	Adj. R-squared:	0.705
Method:	Least Squares	F-statistic:	719.7
Date:	Sun, 22 May 2022	Prob (F-statistic):	0.00
Time:	23:26:00	Log-Likelihood:	-30936.
No. Observations:	11439	AIC:	6.195e+04
Df Residuals:	11400	BIC:	6.224e+04
Df Model:	38		
Covariance Type:	nonrobust		

As for out-sample testing, the R^2 is 0.699, mean square error is 3.616, and mean absolute error is 2.526.