Assignment 1: Power Calendar function

To calculate the number of hours, we need to first figure what is the type of input. There are four type of period input. We use regular expression of classify the input period and at the same time, extract the information from the input to obtain the start and end date. After this, we can calcuate the number of hours based on the peak input. Here is where I find a pattern. The flat hour is the sum of onpeak and offpeak hours. Meaning if we can find onpeak hour. We can easliy obtain offpeak hour as well. The flat hour is simply 24 times the number of days. For onpeak hour, we need to filter the days which is not weekend or holiday. After that mutiply the number of days by 16 will give us the result. Finally, the offpeak hour can be calculated by the flat hour minus the onpeak hour. For 2 * 16H type. We need to find days that is either weekend or holiday, then mutiply the number of days by 16. For 7 * 8. We simply mutiply total number of days by 8. Also, the small difference between Estern and Western region and be solved by adding a parameter to classify weekdays.

Assignment 2: Meter Data formatting

We start by preprocessing the data. For the new.app4 data. We need to turn the data into hourly so that the datetime could match the other dataset. For the USA_AL_Auburn-Opelika.AP.722284_TMY3_BASE data. We need to modify the datetime to have the same shape. Also, some data contain time like 24:00:00 which is not supported by python datetime structing. We have the change those to 00:00:00 for the next day. Here is the data after we preprocess.

Out[6]:

	DateTime	W_min
0	06/07 11:00	57.388943
1	06/07 12:00	27.227961
2	06/07 13:00	111.476298
3	06/07 14:00	109.021960
4	06/07 15:00	5.773963

Out[7]:

	DateTime	Electricity:Facility [kW](Hourly)	Gas:Facility [kW] (Hourly)	Heating:Electricity [kW](Hourly)	Heating:Gas [kW] (Hourly)	Cooling:Electricit [kW](Hourly
0	01/01 01:00	0.974334	4.452977	0.0	4.425010	0.
1	01/01 02:00	0.796582	4.850317	0.0	4.824566	0.
2	01/01 03:00	0.735028	5.037645	0.0	5.012193	0.
3	01/01 04:00	0.727433	5.107562	0.0	5.082468	0.
4	01/01 05:00	0.778706	5.270878	0.0	5.246732	0.

Then, I use the dfply the merge the two dataset by matching their datetime. Note the W_min column is divided by 1000 to match the unit with other columns. Here is the merged dataset.

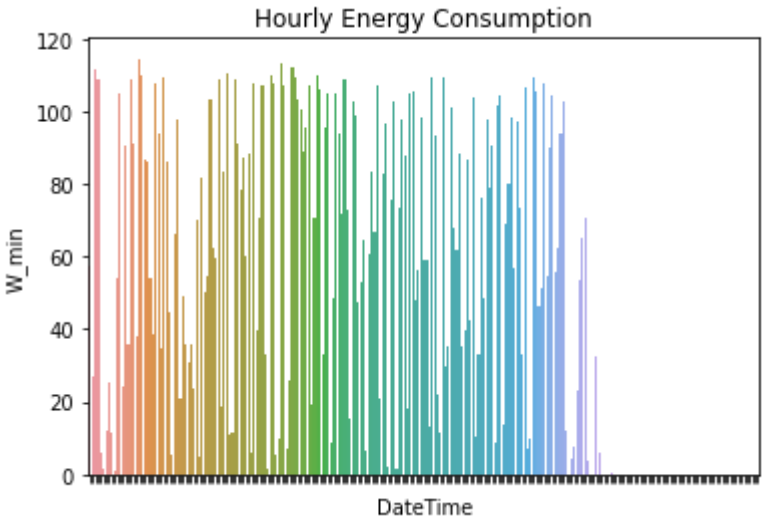
Out[8]:

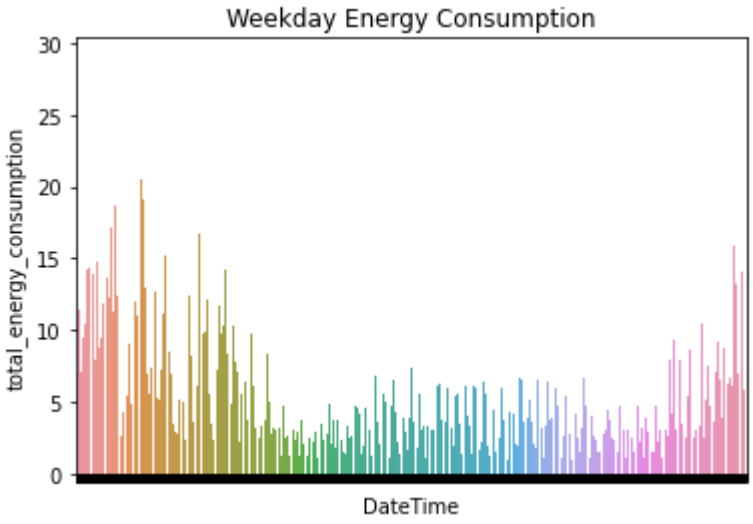
	DateTime	W_min	Electricity:Facility [kW](Hourly)	Gas:Facility [kW] (Hourly)	Heating:Electricity [kW](Hourly)	Heating:Gas [kW] (Hourly)	Coo
0	06/07 11:00	57.388943	1.479426	0.018757	0.0	0.0	
1	06/07 12:00	27.227961	1.559733	0.018441	0.0	0.0	
2	06/07 13:00	111.476298	1.702835	0.019079	0.0	0.0	
3	06/07 14:00	109.021960	1.859094	0.020153	0.0	0.0	
4	06/07 15:00	5.773963	2.100629	0.021274	0.0	0.0	

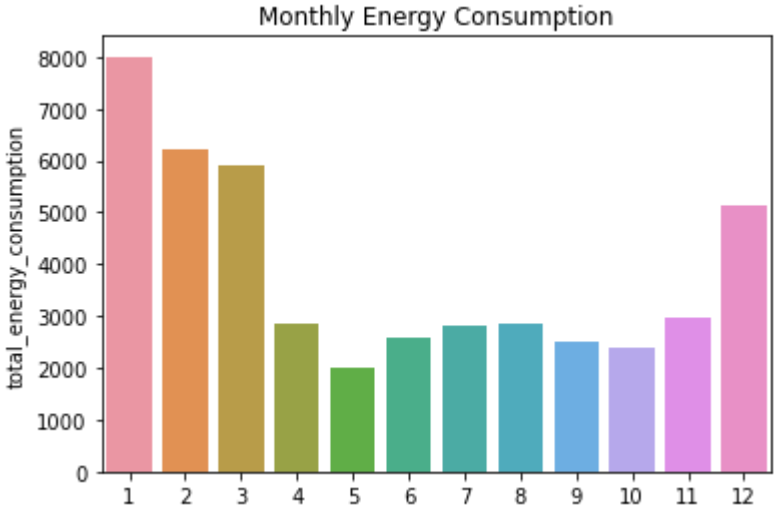
I also add another column that sum up all other columns as total energy consumption.

Out[12]:

	DateTime	W_min	Electricity:Facility [kW](Hourly)	Gas:Facility [kW] (Hourly)	Heating:Electricity [kW](Hourly)	Heating:Gas [kW] (Hourly)	Coo
0	06/07 11:00	57.388943	1.479426	0.018757	0.0	0.0	
1	06/07 12:00	27.227961	1.559733	0.018441	0.0	0.0	
2	06/07 13:00	111.476298	1.702835	0.019079	0.0	0.0	
3	06/07 14:00	109.021960	1.859094	0.020153	0.0	0.0	
4	06/07 15:00	5.773963	2.100629	0.021274	0.0	0.0	







Here are the plots. The first plot is using the new.app4. It shows the hourly energy consumption. We can see cycle in the energy use. Each cycle is one day. Between each day. There are some days where is energy comsumption is lower than others. Those are the weekends. There is also almost no energy consumption for the last part. This can be an indication that the system is shut down and no longer consume energy. The next two plots are constructed using the other dataset. The second plot is hourly energy consumption without the weekends. The energy consumption still appear to be in cycle for each day. There is rarely days that have very low energy consumption as the weekend are removed. From this, we can also see a general trend that energy use is high at the begining, then lowered to some constant in the middle and finally raise again at the end. This trend is more clearly captured by the third plot. Here we plot the energy consumption for each month. The energy use is high for the first three month and last month. This might mean that the system is operating at a higher rate from December to March.

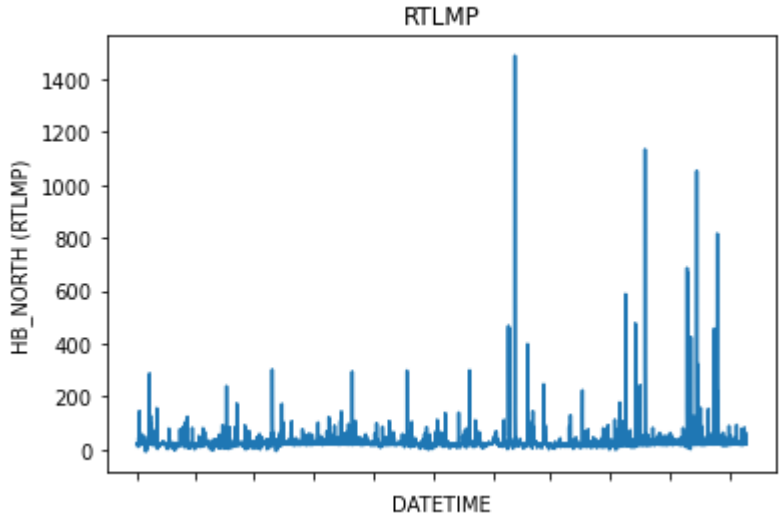
Assignment 3: EDA and forecast model

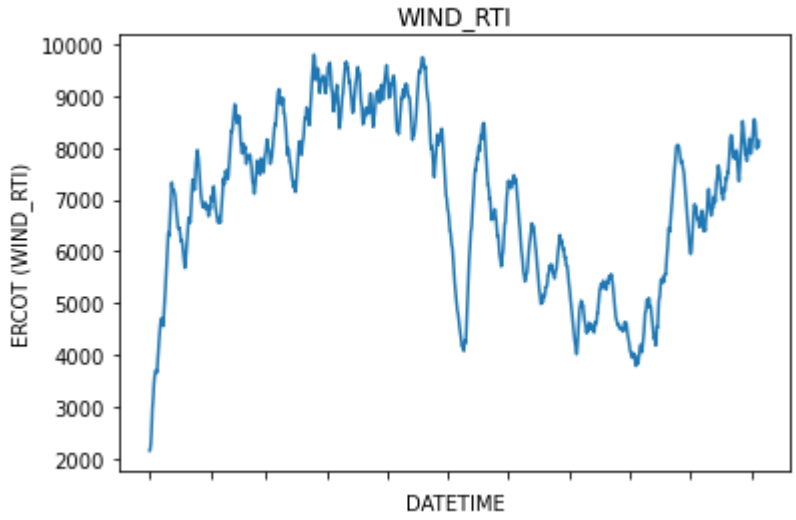
This is the date we imported. There are some missing values in the data. Those values are dropped.

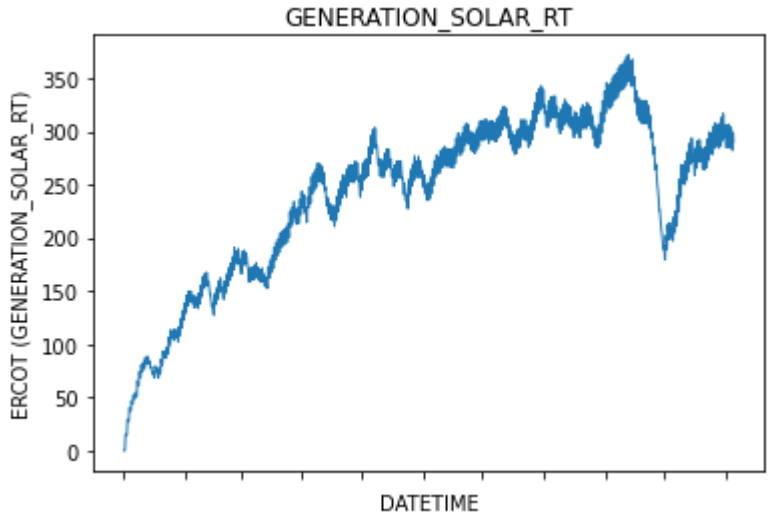
Out[19]:

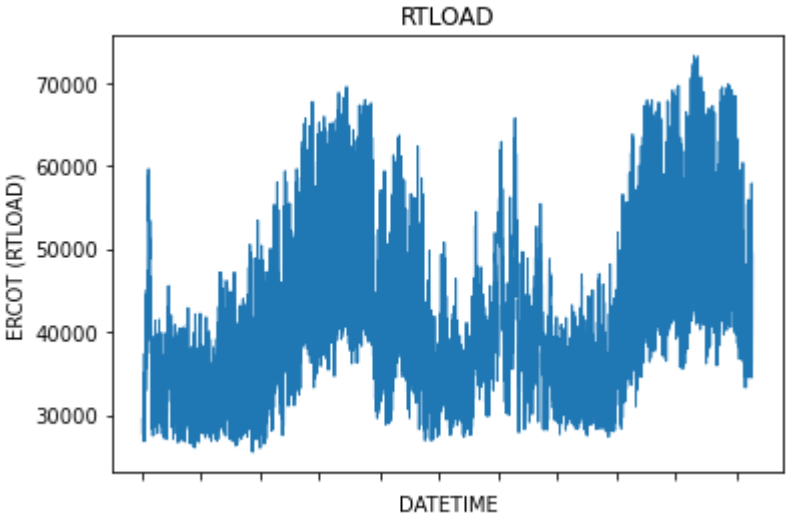
	DATETIME	HB_NORTH (RTLMP)	ERCOT (WIND_RTI)	ERCOT (GENERATION_SOLAR_RT)	ERCOT (RTLOAD)	HOURENDIN
0	2017-01-01 01:00:00	23.3575	2155.31	0.0	29485.791355	
1	2017-01-01 02:00:00	21.4650	2313.81	0.0	28911.565913	
2	2017-01-01 03:00:00	20.7350	2587.68	0.0	28238.258175	
3	2017-01-01 04:00:00	20.2700	2748.65	0.0	27821.000513	
4	2017-01-01 05:00:00	20.1200	2757.49	0.0	27646.942413	

We start by smoothing the data. The method we adopt is Exponential Smoothing. This will reduce the effect of cycles in timeserie and make the data more smooth. Below are line plots for each column in the timeserie after smoothing.

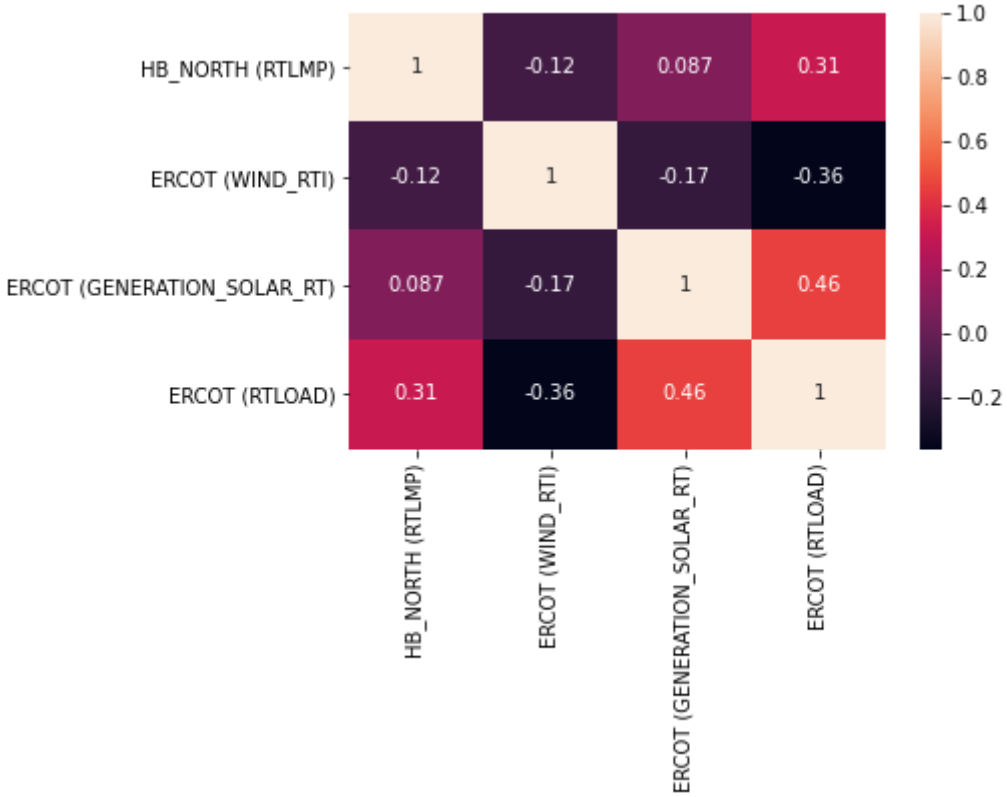








We also produce a heat map to see the correlations betweem each variable. High correlation between predictors and cause problem is contructing the forcasting model. The variable we are predicting is RTLMP, others are predictors. From the graph, we can see that none of predictors are closely correlated. Only RTLOAD and GENERATION_SOLAR_RT have a 0.46 correlation which is still at an acceptable level.



For the last part. We create a linear regression model. The choose 0.8 the data as train and 0.2 as testing. Here are the training and testing error.

Training error:

Out[37]:

195.68344917182526

Testing error:

Out[38]:

132.7135399784833