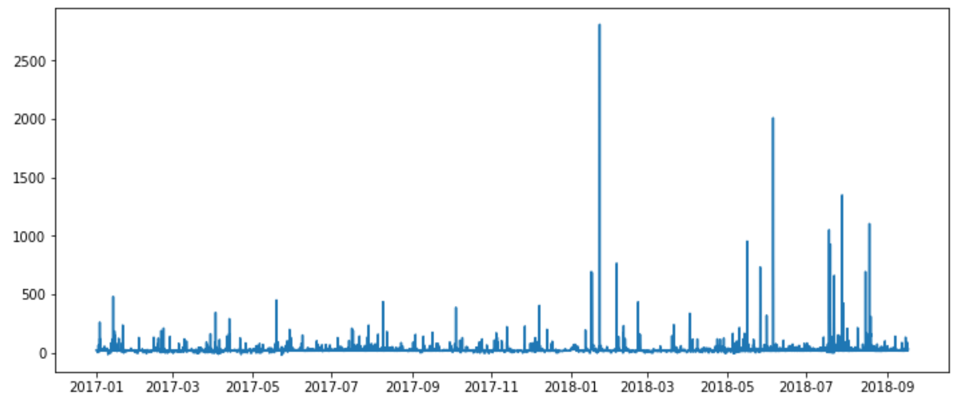
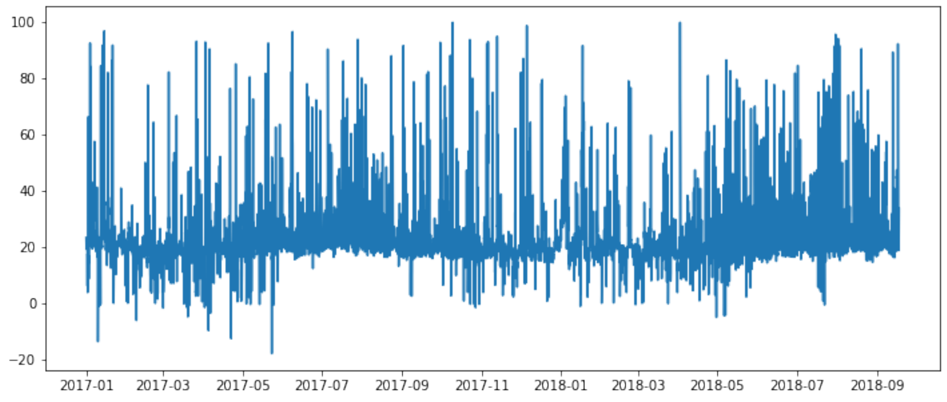
Assignment 3

I first attempt to graph the price data against time to explore possible seasonality in the data. The graph is shown below.



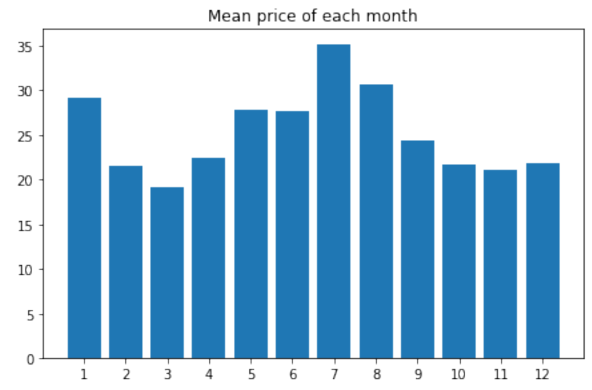
There are some extremely high prices, so I try to remove those outliers and graph the rest. I only graph prices below $100, which make up 98,7% of the dataset. Also 0.5% of the data points actually have negative prices. Due to the size of the dataset, the graph is not very informative, so I decide to explore trends in hour ending, month, year and peak type.

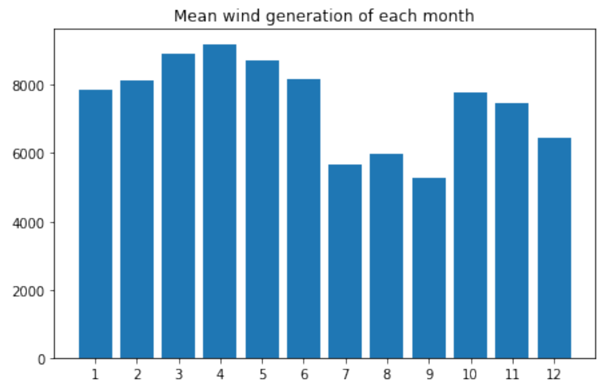


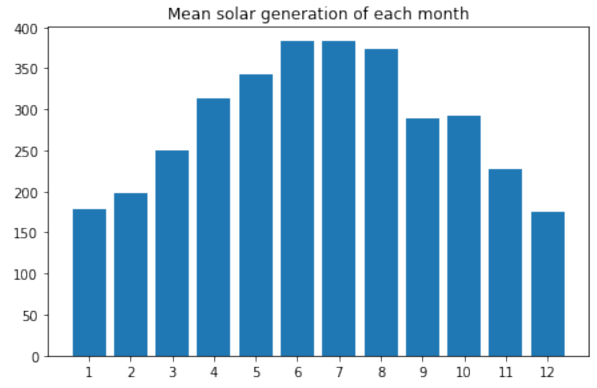
The table below shows the average price, wind generation, solar generation, and hourly load in 2017 and 2018. All the data increase in 2018.

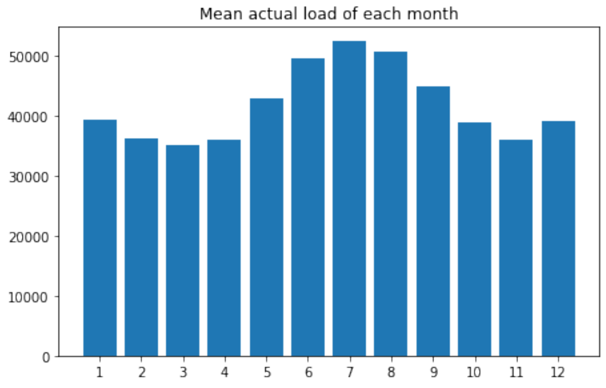
	YEAR	PRICE_MEAN	WIND_MEAN	SOLAR_MEAN	LOAD_MEAN
0	2017	23.255672	7108.951557	240.999904	40822.411716
1	2018	29.298474	8128.335901	363.707955	44551.139738

Next I graph the mean price, wind generation, solar generation and hourly load by hours in a day. As expected, electricity price is generally higher in summer and winter, and lower in spring and fall. Wind generation is low in summer, due to the fact that summer is less windy than other seasons. Conversely, solar generation is highest in summer due to the longer day time and stronger sunlight. The actual load is higher in summer as well.

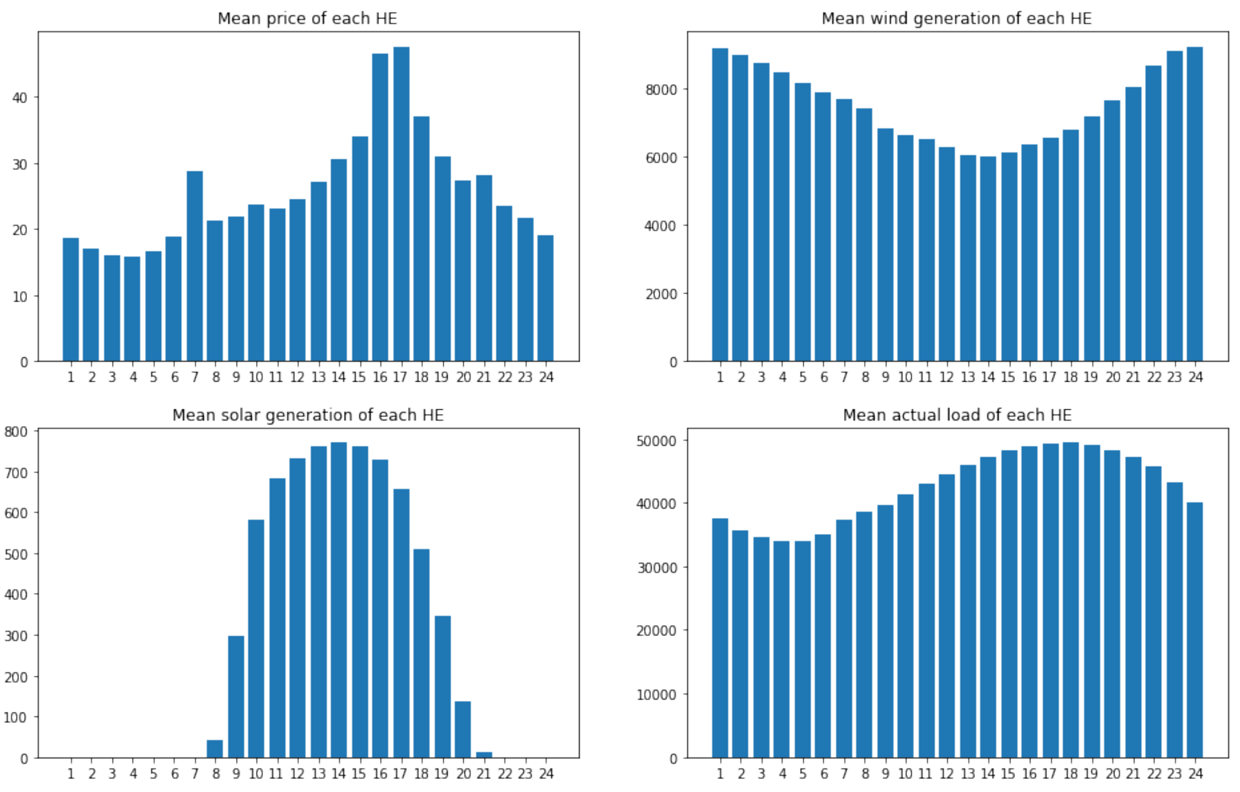




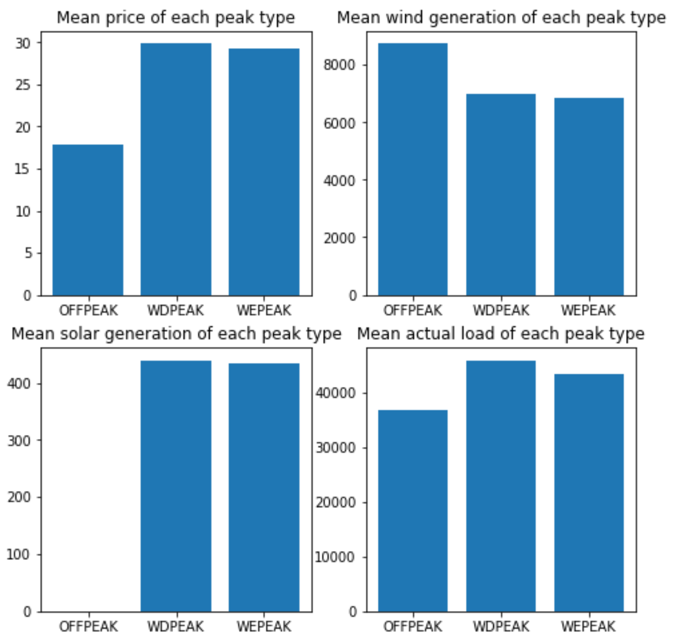




Next set of graphs is based on HE of each day. Prices are higher in the morning when everyone is waking up and in late afternoon when everyone starts to get home from work. As expected, there is more wind generation when the sun is down, and there is no solar generation without sunlight. The actual load is cyclical; it starts increasing in the morning and peaks around 7pm, and then starts decreasing and reaches lowest point at 5am.



The next set is categorized by peak types. As expected, the mean price of off-peak hours is lower than that of peak hours. Since off-peak hours are usually nighttime, there is more wind generation in those periods, and zero solar generation. The results coincide with the previous HE trends. The actual load has similar distribution as the mean price.



To forecast the real-time price, I first try traditional linear models. The dependent variable is the RTLMP real-time price, and the independent variables are WIND_RTI, GENERATION_SOLAR_LOAD, RTLoad of the previous hour. For instance, if we want to predict RTLMP at 5pm, we will get the information on the independent variables from 4pm to 5pm. One might argue that it is impossible to make prediction with short notice, but linear models are generally simpler and have fast computation speed. One can get the data at 4:59pm and still make predictions for 5pm since the calculation will be instantaneous. Since only 5 out of almost 12,000 rows have NA values, I just discard those data points. Then I split the data into training and test sets and normalize them. The test MSE of random guessing is 1097.03, with R^2 almost 0. With OLS, all the coefficients are statistically significant. The table below shows the OLS result. The test MSE is 987.60 and R^2 is 0.010, which is a slight improvement but far from ideal. I then use Ridge and LASSO with 10-fold cross-validation to find the penalty term. Both models give slightly better test MSE and R^2, but the improvement is negligible.

```
OLS Regression Results
==============================================================================
Dep. Variable:       HB_NORTH (RTLMP)   R-squared:                       0.067
Model:                            OLS   Adj. R-squared:                  0.067
Method:                 Least Squares   F-statistic:                     286.3
Date:                Tue, 17 Nov 2020   Prob (F-statistic):         1.63e-179
Time:                        12:23:07   Log-Likelihood:                -63260.
No. Observations:               11984   AIC:                         1.265e+05
Df Residuals:                   11980   BIC:                         1.266e+05
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         25.8488      0.434     59.619      0.000      24.999      26.699
x1            -5.2313      0.448    -11.680      0.000      -6.109      -4.353
x2             1.3758      0.501      2.748      0.006       0.394       2.357
x3             9.8807      0.494     20.013      0.000       8.913      10.849
==============================================================================
Omnibus:                    31437.614   Durbin-Watson:                   2.008
Prob(Omnibus):                  0.000   Jarque-Bera (JB):        939095348.695
Skew:                          30.672   Prob(JB):                         0.00
Kurtosis:                    1373.013   Cond. No.                         1.76
==============================================================================
```

I then try to include quadratic terms. Similarly, all coefficients, including linear terms and quadratic terms, are all statistically significant, while the improvements of OLS, Ridge and LASSO are also all very bad. See table below for reference.

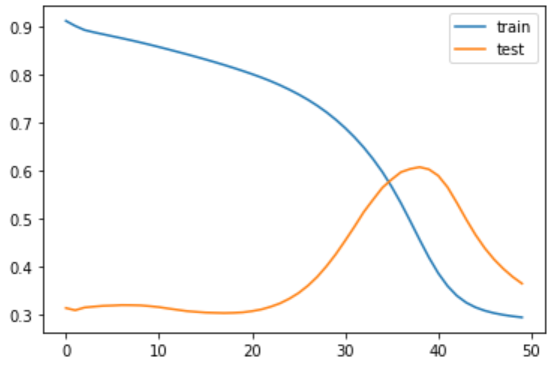
```
OLS Regression Results
==============================================================================
Dep. Variable:       HB_NORTH (RTLMP)   R-squared:                       0.087
Model:                            OLS   Adj. R-squared:                  0.086
Method:                 Least Squares   F-statistic:                     189.8
Date:                Tue, 17 Nov 2020   Prob (F-statistic):         7.10e-232
Time:                        12:23:09   Log-Likelihood:                -63131.
No. Observations:               11984   AIC:                         1.263e+05
Df Residuals:                   11977   BIC:                         1.263e+05
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         14.7516      0.902     16.360      0.000      12.984      16.519
x1            -6.0209      0.449    -13.400      0.000      -6.902      -5.140
x2            -2.5606      0.710     -3.608      0.000      -3.952      -1.169
x3             6.0150      0.573     10.496      0.000       4.892       7.138
x4             2.2897      0.449      5.096      0.000       1.409       3.170
x5             3.9804      0.582      6.840      0.000       2.840       5.121
x6             4.8270      0.388     12.437      0.000       4.066       5.588
==============================================================================
Omnibus:                    31505.106   Durbin-Watson:                   2.006
Prob(Omnibus):                  0.000   Jarque-Bera (JB):        987297894.219
Skew:                          30.820   Prob(JB):                         0.00
Kurtosis:                    1407.789   Cond. No.                         6.16
==============================================================================
```

I then try to incorporate time series terms into my model. My initial thought was to use ARIMA models, but hourly data has seasonality in hours, weekday, and month. Finding the correct terms for ARIMA with Auto_ARIMA or likewise method in Python can be very time-consuming. So, I decided to use RNN, or LSTM in particular, to predict time series data.

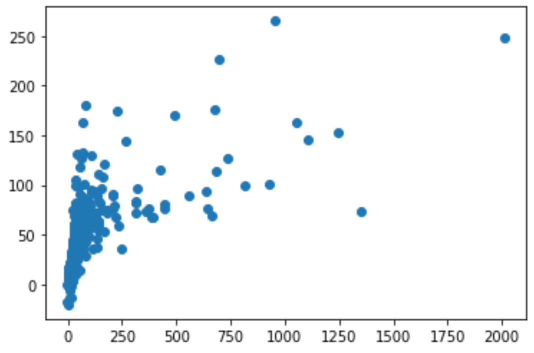
Since we desire time series data to be contiguous, for NA values, I impute them with the average of the last four available data in the same time period. For instance, if we want to impute the amount of wind generation on March 5th HE3, I will get the average of wind generation of HE3 from March 1st to March 4th. I encode categorical data month and peak type. To preserve contiguity, I have to keep in the price outliers, and make the train-test split not random. Every data point before May/6/2018 0:00 goes into the training set and the rest into the test set. I add the data of the previous hour of all four time series variables as independent variables, and I also remove the latest data on wind generation, solar generation and actual load, unlike what I did with linear regression models. I believe that neural networks are slightly more complicated, and we might not get prediction in time if we always rush with the latest data. I then normalize the data to get ready for training. See summary below for the LSTM model.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 1, 100)            43600
_________________________________________________________________
lstm_2 (LSTM)                (None, 50)                30200
_________________________________________________________________
dense_1 (Dense)              (None, 20)                1020
_________________________________________________________________
dense_2 (Dense)              (None, 10)                210
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 11
=================================================================
Total params: 75,041
Trainable params: 75,041
Non-trainable params: 0
_________________________________________________________________
```

The last dense layer has a linear activation and both LSTM layers have tanh activation. The evaluation metric is MSE and I use Adam optimizer. Epoch is 50, batch size is 72, and I use 10% of the training data for validation and change some hyperparameters. The train and validation error curves go as followed.

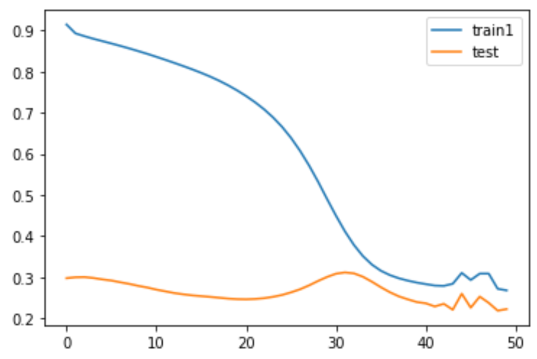


Arguably there are some signs of overfitting, but I think the difference between train and validation accuracies is acceptable. The test MSE is 4456.73 and R^2 is 0.2375. At first this result baffled me, since the MSE is much higher than other predictions while the R^2 value is much better. Then I plotted the predicted y against the actual y, and figured that very few outliers caused the high MSE. See graph below.



I then remove the data points with prices higher than $250, keeping 99.03% of the test set, and get MSE 190.84 and R^2 0.47. This model produces much better results than the others, and it means that except some rare occasions when the price goes crazy and skyrockets, on average the prediction of my model is about $13.8 away from the actual price. This number is far from optimal; some possible future works include increasing lag periods and better ways to deal with outliers.

When I was working on LSTM, I found out that the prediction was almost instantaneous. Considering that the model already seems to be slightly overfitting, I try to build the neural network with real-time wind, solar and load data, like I did with the linear regression models. The data from the previous hour is also included, so compared to the last model, there are three more independent variables. For instance if I want to predict the price at March 5th 3pm, I will use the price at 2pm, the wind, solar and load data in HE2 and HE3. In the previous LSTM model I only use data in HE2. The training and validation error curve shows as followed.



While there seems to be slight underfitting, I tried increasing the epoch but the validation error escalated quickly. The MSE with outliers is 4455.57 and the R^2 is 23.77, and the MSE without outliers is 234.13 with R^2 0.35. The result is actually worse than the previous neural network, but still significantly better than linear regression models.

See table below for a complete list of results.

Model	MSE	R^2
Random guess	1097.03	-0.00
OLS w/o quadratic term	987.60	0.10
Ridge w/o quadratic term	985.57	0.10
LASSO w/o quadratic term	987.55	0.10
OLS w/ quadratic term	974.84	0.11
Ridge w/ quadratic term	972.04	0.11
LASSO w/ quadratic term	974.71	0.11
LSTM w/ 1-lag only	4456.73	0.24
LSTM w/ 1-lag only (w/o outliers)	190.84	0.47
LSTM w/ latest data	4455.57	0.24
LSTM w/ latest data (w/o outliers)	234.13	0.35