Electric Load Prediction

Haiyue Yang, Yun Shen, Xin Peng

The prediction of electric load is a task that can allow generators to decide how much production to ramp up, consumers to estimate electricity prices, and the grid operator to make sure enough transmission capacity is available.

We are using 4-year electrical consumption, generation, pricing, and weather data for Spain from https://www.kaggle.com/nicholasjhana/energy-consumption-generation-prices-and-weather (https://www.kaggle.com/nicholasjhana/energy-consumption-generation-prices-and-weather) to build up time series models to predict future electric load using the previously observed values and features like weather.

And we have explored the data, brainstormed what factors could help improve the electric load prediction, proposed several models, and implemented the variants below.

Add-on Value of the Data

- Help with planning future infrastructure. If the electricity is increasing gradually in a regeion, new plants might need to be constructed.
- Prepare extra energy supply if extreme weather happens. For example, the recent Texas outage resulted from the unexpected shutdown of generating plants. Extra energy supply and regular inspections could help prevent such outage from happening.
- Propose energy pricing strategies. The company could decide different prices for different times of the day due to the amount of electricity used and needed.
- Determine if new energy source is needed. When the demand exceeds the supply for a continuous time, more efficient energy source should be considered.

Key Metrics to Measure Success

- Avoid energy breakdown in extreme weather. Public could track the number of outages happened in a year to see if the company has improved unexpected energy breakdown.
- Save of energy. Moreover, saving energy means saving the environment. With global warming, the company could utilize the prediction to generate the right amount of the energy to avoid any extra generation of energy.
- For our models, we used MSE as the metrics to determine the performance of the models. We also looked at whether the trend of predicted values fits the actual values on the plots to see if the model precisely predicts the electricity demand.

Data Exploration

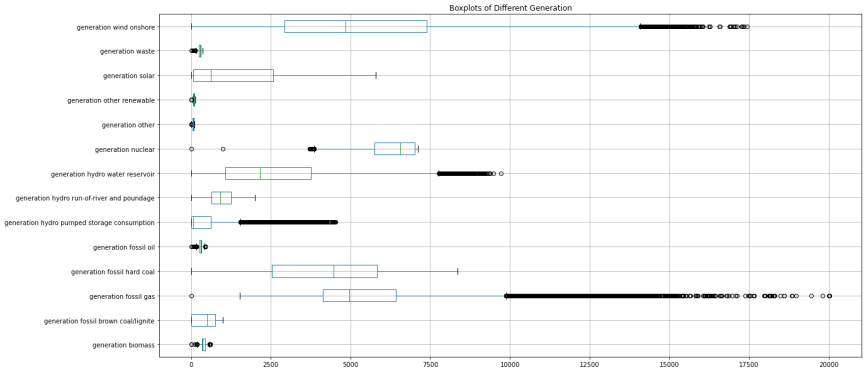
Problems in the data

	time
date	
2015-02-01	13
2015-01-05	17
2016-04-25	22
2015-10-02	22
2017-11-14	22

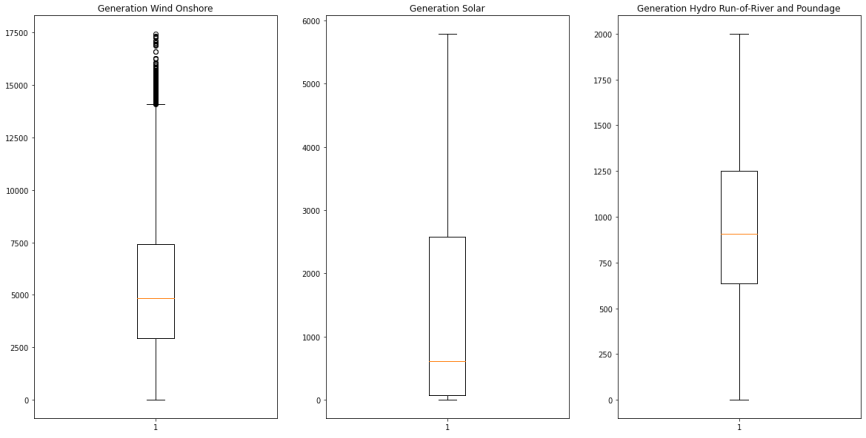
In the energy dataset, it is expected that the data of all 24 hours in a given day should be included. However, we discovered that for some given days, data in some hours is missing.

	date	time	generation biomass	generation fossil brown coal/lignite	generation fossil gas	generati fossil ha co
744	2015-02-01	00:00:00	433.0	323.0	4826.0	488
745	2015-02-01	01:00:00	445.0	316.0	4891.0	467
746	2015-02-01	02:00:00	441.0	320.0	4652.0	469
747	2015-02-01	03:00:00	440.0	323.0	4603.0	458
748	2015-02-01	04:00:00	432.0	310.0	4212.0	459
749	2015-02-01	05:00:00	443.0	237.0	4215.0	471
750	2015-02-01	06:00:00	435.0	282.0	4641.0	493
754	2015-02-01	10:00:00	460.0	304.0	5393.0	595
755	2015-02-01	11:00:00	450.0	317.0	4967.0	599
764	2015-02-01	20:00:00	479.0	326.0	8036.0	602
765	2015-02-01	21:00:00	484.0	326.0	7546.0	602
766	2015-02-01	22:00:00	486.0	326.0	7109.0	602
767	2015-02-01	23:00:00	489.0	326.0	5930.0	602

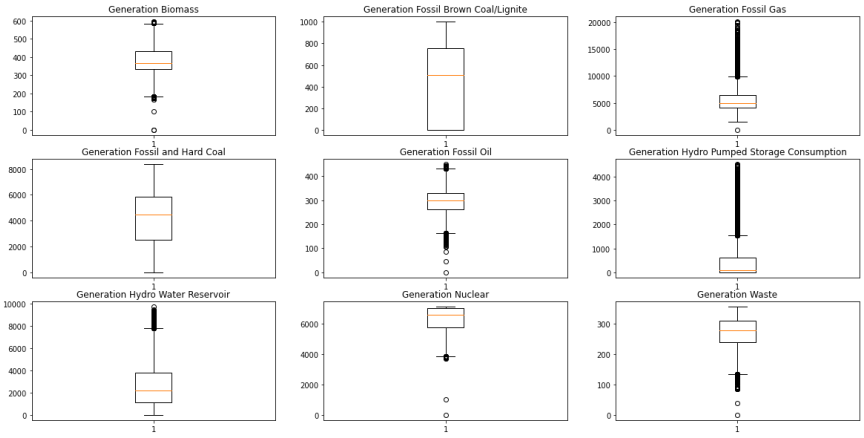
For instance, in 2015-02-01, the data from 7am-9am and the data from 12pm - 19am are not included in the dataset.



We draw the boxplots of different ways of generations. We discovered that the generation of different ways varies a lot, wind onshore, nuclear, fossil hard coal and gossil gas tend to have more generations, with waste, fossil oil and biomass have less generations.

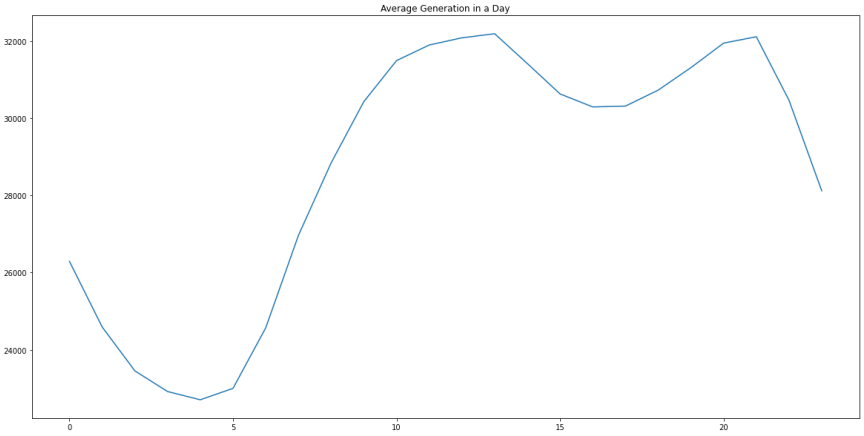


We take a deeper insight into the variable generations: wind onshore, solar, and hydro run of river and poundage. We discovered that among these three ways of generations, wind onshore has the largest variance, probably because the sun and run-of-river are comparatively stable with time, but wind is less predictable.



Then, we also look as fixed generations. Although these generation methods are considered to be fixed, some of them even have larger variance than the variable ones. I guess probably the variable generations are all clean energies and are used with a higher priority, and the fixed generations can be used as a complement to meet different demands. Especially the fixed generations that are relatively cheaper, like fossil gas, hydro pumped storage consumption, and hydro water reservoir, can be used to generate more when the demand is very high, so these three generations have outliers in the larger side.

Present trends and periodicity



We investigated the periodic pattern of demand in different hours of a day. We discover that, on average, the demand is much higher during the daytime and is lower during the night. Moreover, the demand slightly drops in the afternoon, but increases again in the evening.

Brainstorm a comprehensive list of factors that could affect the recorded data, or the trends your models will capture.

Before building up a model, it is important to notice that there are many factors that could conceivably influence the recorded electric load in real life. Here are some of these factors:

- Holidays and special events.
- Unusual times, like COVID-19 lockdown, there is dramatic reduction in electric use in services and industry, however, demand is higher for residential use.
- Technology improvement that decreases electricity use in industry.
- Regulatory changes in using electricity, for example, time-of-use electricity rate, Tiered billing rate.
- Policy change, for example, industrial transformation.
- Price, for example, if the electricity price is high, consumers tend to use less electricity.
- Day light savings.

Complementary Sources of Data

Since there are many other factors that can affect the electric load. So to better make a prediction, we also need some complementary sources of data.

- Holiday date.
- Special event record.
- Day light savings date of each year.
- Record of regulatory change in using electricity.
- Reocrd of policy change that would affect electricity use like industrial transformation.

Model Staircase

Here are some models we proposed.

- Predict same demand as 24h ago (Baseline Model).
- Autoregressive Model, and use previous 48 hours to predict next 24 hours.
- Linear regression (Variant 1).
- LSTM, and use previous 48 hours to predict next 24 hours (Variant 3 subvariant 1).
- GRU, and use previous 48 hours to predict next 24 hours (Variant 3 subvariant 2).
- Convert time series forcasting into a supervised learning problem and add weather feature to predict. (Variant 4)
- Ensemble previous models. (Variant 5)

Baseline Model

In the baseline model, we predict same demand as 24h ago. For example, we predict the electricity demand in 2018-12-31 23:00:00 to be the same as the electricity demand in 2018-12-30 23:00:00.

```
12746358.793856103
```

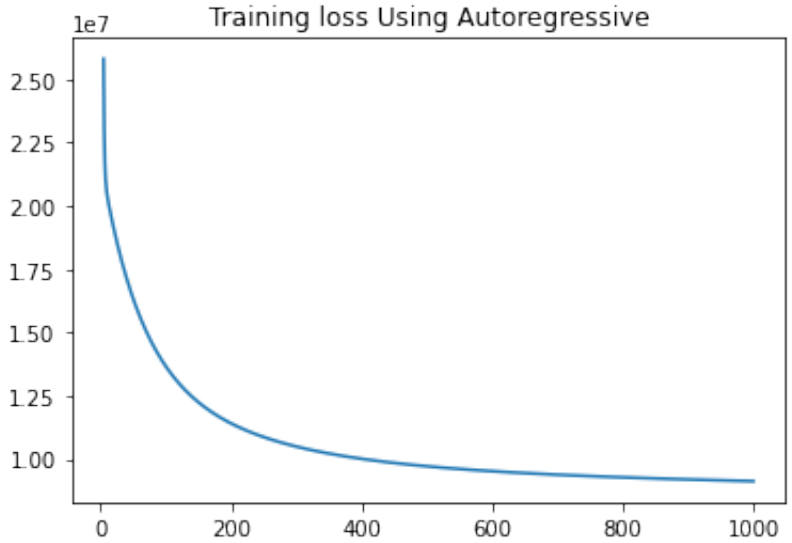
By using the baseline model, we can see that MSE is 12746359, which is relatively large. And in the variants below, we will compare the testing MSE of the variants with the baseline MSE.

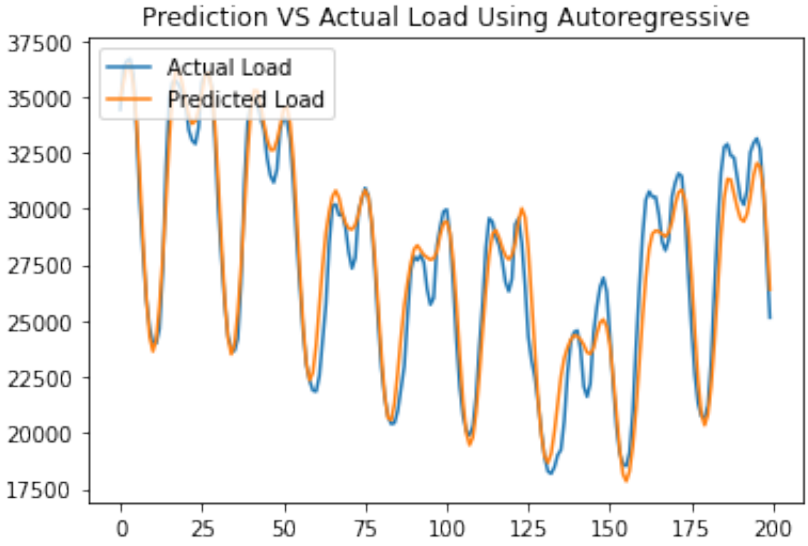
Autoregressive Model

Autoregressive model is provided for us to predict electric load. Autoregressive model predicts future values based on past values, and in this autoregressive model, electric load of the previous 48 hours is used to predict the electic load of the next 24 hours. However, this model has assumed the linearity between previous observations and future observations. It also has some other limitations like not considering other features like weather, ignoring the covariance, etc. Also from the plot of "Prediction VS Actual Load Using Autoregressive", we can find that there still exists some problems in prediction as the prediction does not fit the actual value very well.

So in the next few variants, we will try some other models to do this time series prediction.

```
0 368025730.0 380860670.0
100 13668292.0 13495825.0
200 11404723.0 11133426.0
300 10488203.0 10231205.0
400 10003367.0 9768374.0
500 9708450.0 9489148.0
600 9513156.0 9303581.0
700 9374784.0 9170876.0
800 9270716.0 9069912.0
900 9188280.0 8989001.0
```





In the above plot, we show the result of using previous 48 hours to predict a single next hour.

```
2351017.8
```

The testing MSE of using autoregressive model is 2351017.8.

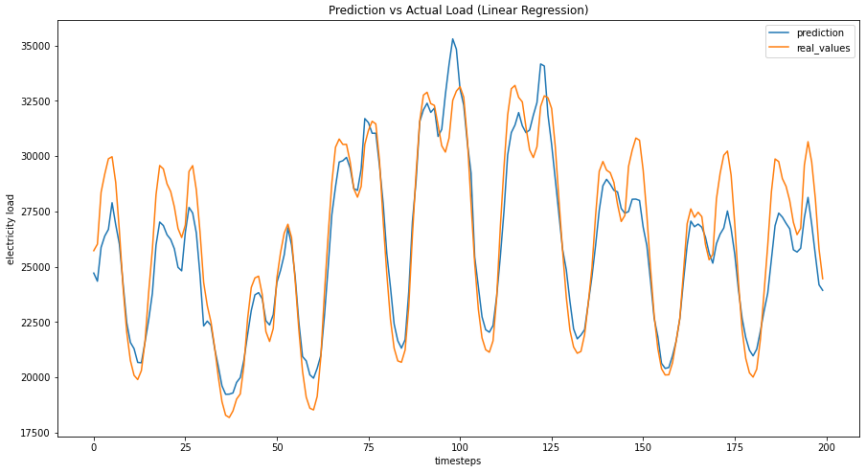
Variant 1: Linear Regression

First, we would like to see if a simple linear regression model can satisfactorily predict the electric load. Our baseline model is an autogressive model, which is actually a linear regression model but using previous behaviours to predict future behaviours. However, we only used the "total load actual" data for the autoregressive model. In this linear regression model, we used the amount of electricity generatd by each methods to predict the total load.

We extracted all the variables regarding the amount of electricity generated from different methods and saved them in X. We saved the actual load as y. We then used a standard scaler to transform the data. We also split the data into train and test sets, and used the train set to fit a linear regression model.

```
LinearRegression(copy_X=True, fit_intercept=True
, n_jobs=None, normalize=False)
```

```
Text(0.5, 1.0, 'Prediction vs Actual Load (Linea
r Regression)')
```



From the plot, we saw that the performance of the linear regression model was not quite satisfactory. It preserved the ability to predict the peak and off-peak trend over the time, but the predicted values were far away from the actual values.

```
Test Mean Squared Error: 2157031.8485672977
Test Mean Absolute Error: 1160.3325599411407
```

The test MSE is around 2157031 and the test MAE is around 1160. Compared to the basedline model, which has a test MSE of 12746358, the test MSE decreased by around 83%. However, its test MSE was roughly the same as the autogressive model. This makes sense because the autoregressive model is basically a linear regression model using previous values to predict future values.

Limitation

However, after we carefully looked over this approach again, we found that the total load is roughly the same as the sum of all the electricity generated by different methods. For the sake of this dataset, we want to predict future electricity demand so that the right amount of energy could be generated to avoid any waste of energy. Thus, this linear regression approach might not be very useful because it lacks the power of predict future demand from previous demand. We still included this method because we wanted it to serve as a new attempt to solve the problem with multi-variate data. This model also inspired us to merge the weatehr data.

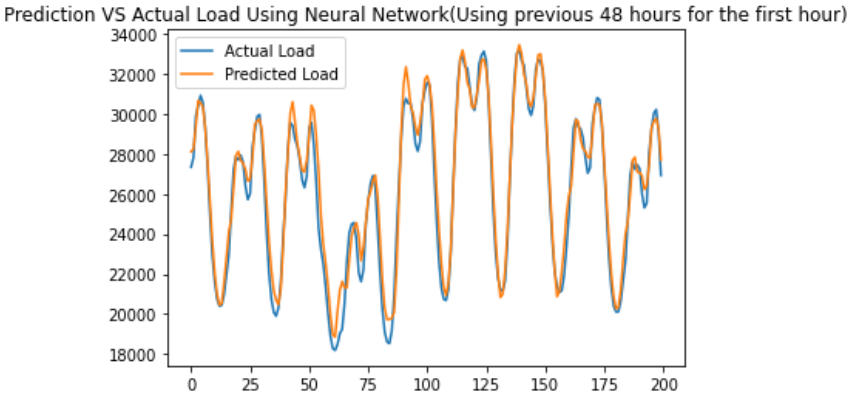
Variant 2: Neural Network

In this variant, we used a MLP Neural Network to predict the next 24 hours eletric load based on previous 48 hours electric load. Also we used the first 30000 data as training data, and the rest data as testing data.

Before training the model, we did some data preprocessing in this variant. We used data normalization using MinMaxScaler, so that variables lie in the range of 0 and 1. And we used the preproccessed data to train the model, and also used MSE as the loss function. Since the data is normalized and in the range of [0, 1], the training loss would be far smaller than the training loss in the autoregressive model, so we would not compare the training loss between Variant 2 and Autoregressive model here.

And we still used MSE as the loss function and Adam as the optimizer. And in the MLP, we used a single hidden layer with hidden size = 10, and ReLU on it.

```
Epoch: 0, loss: 0.34788
Epoch: 100, loss: 0.02924
Epoch: 200, loss: 0.02595
Epoch: 300, loss: 0.02077
Epoch: 400, loss: 0.01863
Epoch: 500, loss: 0.01835
Epoch: 600, loss: 0.01824
Epoch: 700, loss: 0.01817
Epoch: 800, loss: 0.01812
Epoch: 900, loss: 0.01811
```



```
919912.7
```

The performance of the Neural Network is quite good, and from the plot, we can see the first hour it predicts using the previous 48 hours is quite close to the actual demand, and it has greatly captured the trend and seasonality of this time series data.

Also using Neural Network can achieve testing MSE = 919913, which is only 7.2% of the MSE using baseline model, and even compared to using autoregressive model, neural network decreases testing MSE by 60.9%.

And although compared to the baseline model, using Neural Network is computationally expensive, but compared to RNN, the training time of this variant is fast enough.

Variant 3: Recurrent Neural Network(LSTM and GRU)

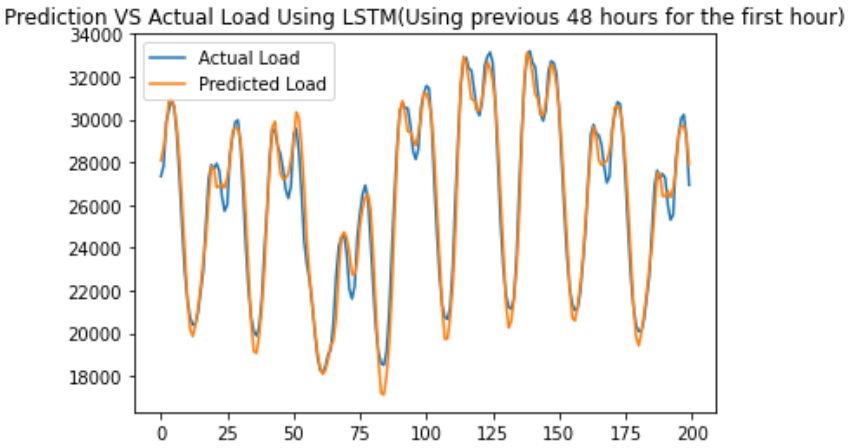
RNN is a good way to deal with time series problems since it can retain state from one iteration to the next. However, it has some problems like gradient vanishing, not being able to capture long-term dependencies in a sequence. So in this variant, we used two RNN architectures, Long Short Term Memory and Gated Recurrent Units. GRU has two gates: reset gate and update gate and LSTM has two more gates: forget gate and output gate.

In both LSTM and GRU, we still used the previous 48 hours to predict the next 24 hours. Also we used the first 30000 data as training data, and the rest data as testing data. Again, we did data normalization using MinMaxScaler.

Subvariant 1: LSTM

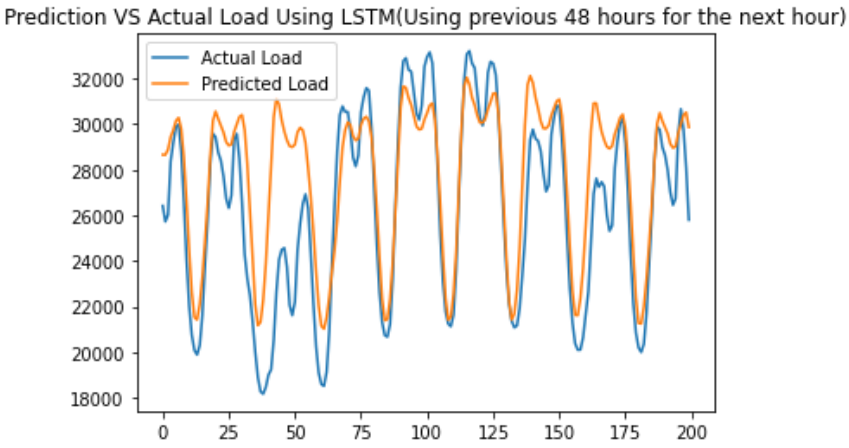
We used LSTM here, and used Adam optimizer, number of layers = 1, hidden size = 7.

```
Epoch: 0, loss: 0.26526
Epoch: 100, loss: 0.03666
Epoch: 200, loss: 0.03303
Epoch: 300, loss: 0.01849
Epoch: 400, loss: 0.01577
Epoch: 500, loss: 0.01434
Epoch: 600, loss: 0.01226
Epoch: 700, loss: 0.01159
Epoch: 800, loss: 0.01129
Epoch: 900, loss: 0.01111
```



Plot 1: The predicted first hour

```
839657.06
```



Plot 2: The predict 24th hour

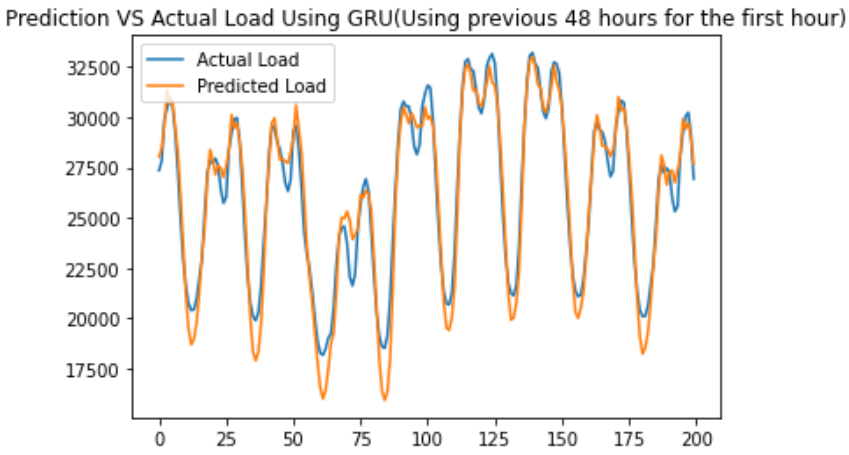
In variant 3, we still used previous 48 hours to predict next 24 hours, for example, we used t1, t2, t3, ..., t48 to predict t49, t50, ... t72. And from the above two plots, we can see that the predicted first hour (i.e., t49) is already quite close to the actual load (plot 1), however, the predicted 24th hour (i.e., t72) does not perform well (plot 2), and there is much difference in the predicted load and actual load. This is not surprising, since the first hour (i.e., t49) just used its previous 48 hours, while the 24th hour (i.e., t72) used data 24-hour before it, which cannot capture the trend very well.

And the testing MSE of using LSTM is 839657.06.

Subvariant 2: GRU

We used GRU here, and used Adam optimizer, number of layers = 1, hidden size = .

```
Epoch: 0, loss: 0.26963
Epoch: 100, loss: 0.03463
Epoch: 200, loss: 0.02257
Epoch: 300, loss: 0.01726
Epoch: 400, loss: 0.01398
Epoch: 500, loss: 0.01290
Epoch: 600, loss: 0.01249
Epoch: 700, loss: 0.01225
Epoch: 800, loss: 0.01201
Epoch: 900, loss: 0.01182
```




```
878779.7
```

The performance of using GRU is good, but compared to using LSTM, it fails to capture some trend, especially during off-peak hours, it always predicts demand lower than actual demand. And the testing MSE of using GRU is 878779.7, which is also a little higher than using LSTM.

Variant 3 Summary

	$TestingMSE$
$BaselineModel$	12746359
$AutoregressiveModel$	2351018
$LSTM$	839657
GRU	878780

Using LSTM and GRU has all greatly improved the performance of predicting electricity demand, and lowered the testing MSE. Compared to GRU, LSTM performs slightly better, and has a slightly lower testing MSE. And using LSTM has decreased the testing MSE by 93.4% compared to the baseline model, and decreased the testing MSE by 64.2% compared to using autoregressive model.

Variant 3 Limitation

However, there still exists some limitations for using variant 3. First, it does not take other features like weather or special event into consideration. So if the weather is extreme in some day, variant 3 will still use the previous 48 hours to predict as usual, and fail to consider the extra electric load in the extreme weather.

Also, both GRU and LSTM did not capture the trend very well during off-peak hours, and they would predict demand lower than actual demand during off-peak hours.

And for a more general case of time series prediction, if the data is non-stationary and prone to change, for example, financial data, RNN is not a good way to predict data in this senario.

Variant 4: Weather Feature

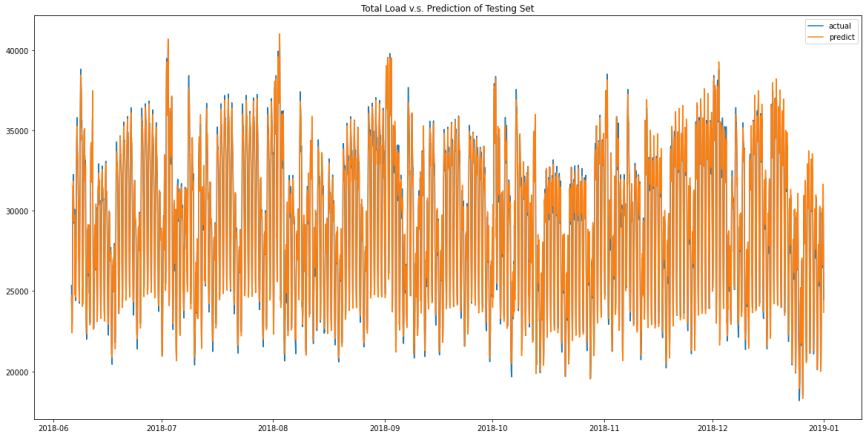
In this variant, we transform the time series forecasting problem into a supervised learning problem. That is, we use the sliding window of the demands in the previous 48 hours as features to predict current demand.

In addition to the previous 48 hours demand, we also add the weather features: average temperature, pressure, humidity, and wind_speed to the model and conduct a linear regression.

```
/usr/local/lib/python3.7/dist-packages/ipykernel
_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice
from a DataFrame

See the caveats in the documentation: https://pa
ndas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy
  This is separate from the ipykernel package so
we can avoid doing imports until


/usr/local/lib/python3.7/dist-packages/ipykernel
_launcher.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice
from a DataFrame

See the caveats in the documentation: https://pa
ndas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy
  import sys
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel
_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice
from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value
instead

See the caveats in the documentation: https://pa
ndas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel
.
/usr/local/lib/python3.7/dist-packages/ipykernel
_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice
from a DataFrame

See the caveats in the documentation: https://pa
ndas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy
  This is separate from the ipykernel package so
we can avoid doing imports until
/usr/local/lib/python3.7/dist-packages/pandas/co
re/series.py:1060: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice
from a DataFrame

See the caveats in the documentation: https://pa
ndas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy
  self._set_labels(key, value)
/usr/local/lib/python3.7/dist-packages/IPython/c
ore/interactiveshell.py:2882: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice
from a DataFrame

See the caveats in the documentation: https://pa
ndas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy
  exec(code_obj, self.user_global_ns, self.user_
ns)



LinearRegression(copy_X=True, fit_intercept=True
, n_jobs=None, normalize=True)
```

```
0.9763935736037983
```



From the graph and data above, the result of this variant is satisifying, with a score of 0.9763935736037983 on the testing set, which means the model has captured most variation in the demand of energy.

Variant 5: Ensemble

In this variant, we use bagging to ensemble the autoregressive model and the neural network model. For each of the two model, we resampled 30000 observations with replacement from the originl training set to train the model and predict. Then, we take the average of the two predictions as the final prediction of the ensembled model.

Model 1: Neural Network

```
Epoch: 0, loss: 893804864.00000
Epoch: 100, loss: 20391010.00000
Epoch: 200, loss: 19574238.00000
Epoch: 300, loss: 18942746.00000
Epoch: 400, loss: 18077172.00000
Epoch: 500, loss: 17077756.00000
Epoch: 600, loss: 16280303.00000
Epoch: 700, loss: 15738405.00000
Epoch: 800, loss: 15408188.00000
Epoch: 900, loss: 15227831.00000
Epoch: 1000, loss: 15136685.00000
Epoch: 1100, loss: 15091161.00000
```

Model 2: Autoregressive

```
0 368025730.0 380860600.0
100 13668292.0 13495824.0
200 11404721.0 11133426.0
300 10488205.0 10231205.0
400 10003368.0 9768374.0
500 9708450.0 9489148.0
600 9513155.0 9303582.0
700 9374783.0 9170877.0
800 9270715.0 9069912.0
900 9188280.0 8989001.0
```

Ensemble

```
array([[30923., 28894., 28642., ..., 35882., 351
70., 35640.],
       [28894., 28642., 26657., ..., 35170., 356
40., 34910.],
       [28642., 26657., 25303., ..., 35640., 349
10., 32677.],
       ...,
       [30753., 27882., 25147., ..., 32728., 326
42., 32155.],
       [27882., 25147., 23068., ..., 32642., 321
55., 30428.],
       [25147., 23068., 21779., ..., 32155., 304
28., 28015.]],
      dtype=float32)
```

```
4351404.0
```

In the ensembled model, the testing MSE is 4351404, which is lower than the MSE of the autoregressive model but higher than the MSE of the neural network.

Probably because the performance of the autoregressive model is much worse than the nerual network, ensembling this two model is a large improvement to the autoregressive model, but cannot improve the performance of the nerual netwokr.