Coolblue Assignment: Forecasting Procedure

Overview

This technical report will briefly wrap around the sale forecasting procedure at Coolblue. Here, we are given 2 time series of sales from 2013/06/21 to 2015/11/15, corresponding to 2 products. The task is to predict the sales of the 2 products in the next 15 days from 2015/11/16 to 2015/11/30.

Data Analysis

Dataset

Let's first take a look at the first few data points in the data set (see Figure 1). Data looks very simple, just includes 3 features and contains totally 1756 data points. Therefore, the 2 products namely, serie_1 and serie_2, will have 878 sale values each.
It would be hard to get any clue of the problem by looking at the raw data directly. Therefore, we will do a further analysis on the data set to get deeper inside.

Figure 1: Sample from the data set.

Quick Analysis

First step when approaching a new problem should nearly always be visualization**.** I have drawn 2 time series and 2 histograms with respect to the sales of the 2 products in Figures 2, 3, respectively (Series1 = serie_1, Series2 = serie_2). We can see that the 2 time series are relatively similar though they are different in their range of values. Series1 has a range that is about 3 times larger than Series2. This is even clearer if we observe their simple statistics in Figure 4. I therefore compute the cross-correlation between the 2 time series, which results in about 0.952945. These evidences make me strongly believe that the 2 products are mostly sold together by somehow with a ratio of 3:1.

Finally, I compute and draw the autocorrelations of the 2 time series in Figure 5. As we can see, there are clearly 7-day periodicity present in both time series. Furthermore, we can also observe some kinds of long range dependency within each time series. In both cases, the average autocorrelations will be larger than 0.5 when lags are smaller than 30 days. This dependency suggests that sales within a month are highly correlated and therefore, in order to predict the future 15 days, the past 15 days of sales will contain very good information for prediction. This would be a very important clue for developing machine learning models in the next sections.

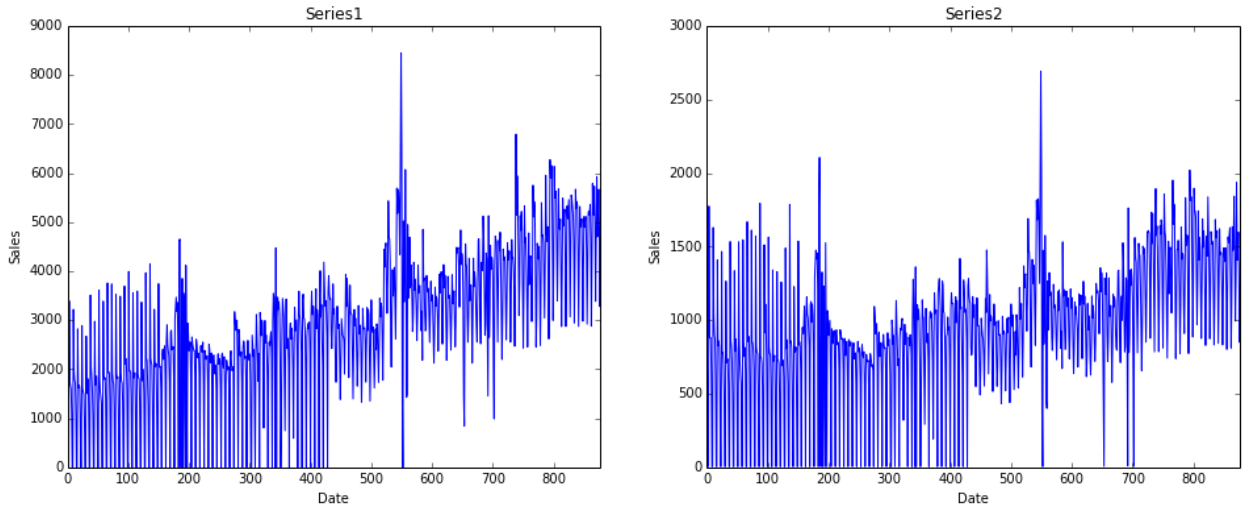


Figure 2: Time series plot of 2 products.

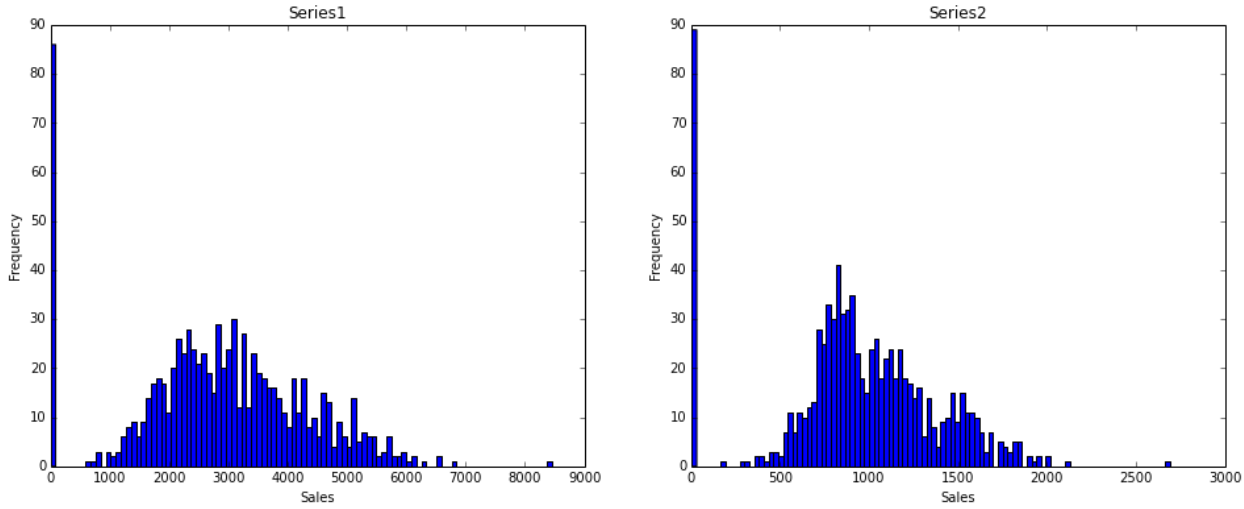


Figure 3: Histogram plot of sales of 2 products.

```
In [59]: # Summary statistics of the 2 products
         df["sales"].groupby(df["serieNames"]).describe()

Out[59]: serieNames
         serie_1     count     878.000000
                     mean     2858.643508
                     std      1462.114998
                     min         0.000000
                     25%      2050.500000
                     50%      2865.000000
                     75%      3793.000000
                     max      8448.000000
         serie_2     count     878.000000
                     mean      943.969248
                     std       448.222032
                     min         5.000000
                     25%       748.250000
                     50%       925.000000
                     75%      1206.750000
                     max      2693.000000
         dtype: float64
```

Figure 4: Simple statistics of sales of 2 products.

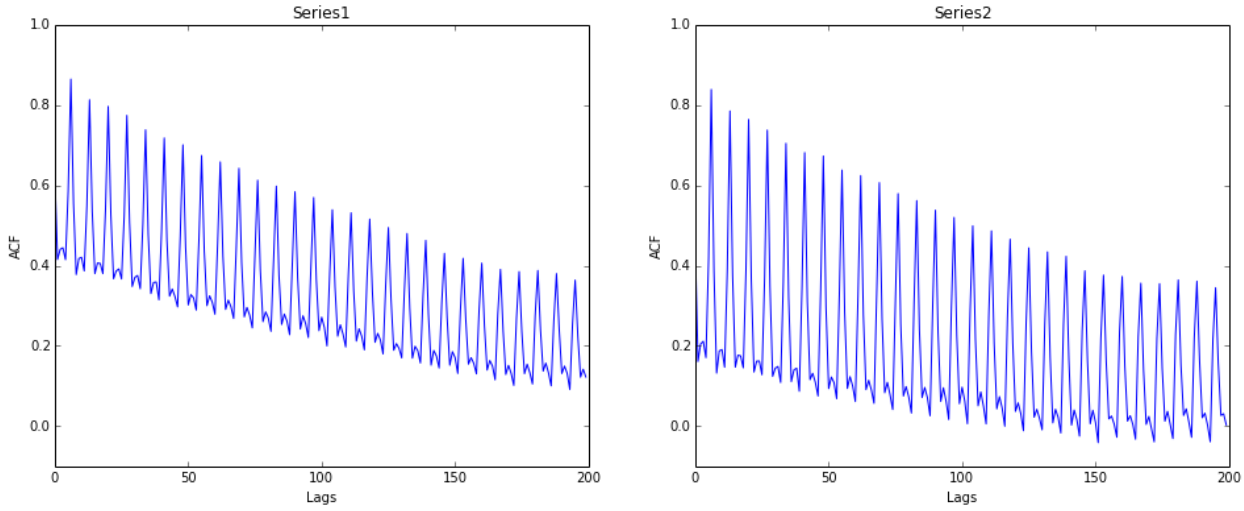


Figure 5: Autocorrelation functions of sales of 2 products.

Machine Learning Problem

The machine learning problem that I am going to solve is that, for each day, we will predict the next 15 days of each product. Therefore, we will have one example/observation per one day.
The target variables for each example will be an array of 30 values (15 days x 2 products). This array can be extended to apply for N products (15 days x N products).
For the features (or predictors), we will have 2 groups of features, each corresponds to one product. For each group of features of a product, we have a list of M lastest days of past sales of that product as M features (M is configurable), and day-of-week and day-of-month as 2 more categorical features. Once features are generated, we can do feature selection by configuring M and/or add/removing day-of-week and day-of-month.
This kind of feature generation/selection could be generalized to N products by applying same procedure for N groups of features.

ETL Work

This part will help transform the original data into the data format described in the previous section. This transformed data will be used for traditional machine learning algorithms. Furthermore, this part will also help prepare a number of stuffs like scaling the 2 time series as they have different ranges as well as do one-hot transformation for the 2 categorical variables day-of-week and day-of-month. As we do scaling only on the training data set, we will not leak data.
At the end of the ETL work, the pandas dataframe will have the following columns:
['series1', 'series2', 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 'series1_p1', 'series1_p2', 'series1_p3', 'series1_p4', 'series1_p5', 'series1_p6', 'series1_p7', 'series1_p8', 'series1_p9', 'series1_p10', 'series1_p11', 'series1_p12', 'series1_p13', 'series1_p14', 'series1_p15', 'series1_f1', 'series1_f2', 'series1_f3', 'series1_f4', 'series1_f5', 'series1_f6', 'series1_f7', 'series1_f8', 'series1_f9', 'series1_f10', 'series1_f11', 'series1_f12', 'series1_f13', 'series1_f14', 'series1_f15', 'series2_p1', 'series2_p2', 'series2_p3', 'series2_p4', 'series2_p5', 'series2_p6', 'series2_p7', 'series2_p8', 'series2_p9', 'series2_p10', 'series2_p11', 'series2_p12', 'series2_p13', 'series2_p14', 'series2_p15', 'series2_f1', 'series2_f2', 'series2_f3', 'series2_f4', 'series2_f5', 'series2_f6', 'series2_f7', 'series2_f8', 'series2_f9', 'series2_f10', 'series2_f11', 'series2_f12', 'series2_f13', 'series2_f14', 'series2_f15'].

Columns 'series**i**': sale at current day of product i. (predictors, scaled to [0, 1])
Columns 'series**i**_p**j**': sale at day j in the past of product i. (predictors, scaled to [0, 1])
Columns 'series**i**_f**j**': sale at day j in the future of product i. (target variables, scaled to [0, 1])
Columns 0 - 37: 38 dummies variables created by one-hot transforming the 2 categorical variables day-of-week and day-of-month. (predictors)

Develop and Tune Model

So now we have prepared a pandas dataframe for developing machine learning models. It's now time to develop machine learning models.

Choice of Algorithms

Since we have multiple target variables, the first algorithm we can think of is multiple linear regression. However, reminding that there exists a very large cross-correlation between sales of the 2 products (~0.95), so there is chance that collinearity will also exist between the feature columns if linear algorithms are used. To avoid this if possible, I have decided to use neural networks for model learning because neural networks can use non-linear activation functions.

Auto Tuning versus Manual Tuning

We now have decided to select neural networks for developing our machine learning models, so the next challenge will be to tune the model to search for “best” hyper parameters. There are a number of methods for searching “best” hyper parameters such as grid search, random search, manual search, SMBO, etc. In this work, I will provide the implementation of grid search and manual search. While the procedure of auto tuning with grid search is provided in the code, I prefer to use manual tuning to easier address the problem of under-/over-fitting.
For evaluation, I use the technique of **time series cross-validation** as illustrated in Figure 6. In current work, I only use 2 hyper parameters for tuning, namely:

- alpha: L2 regularization coefficient
- hid_layers: hidden layer sizes

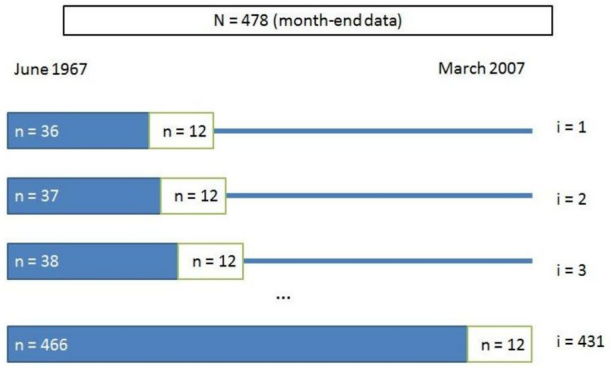


Figure 6: Time series cross-validation. (https://rpubs.com/crossxwill/time-series-cv)

Manual Tuning

Let’s start with M=30 past days for each series, 2 categorical variables day-of-week and day-of-month, hid_layers = (500,) and alpha = 0.001. Result is in Figure 7.

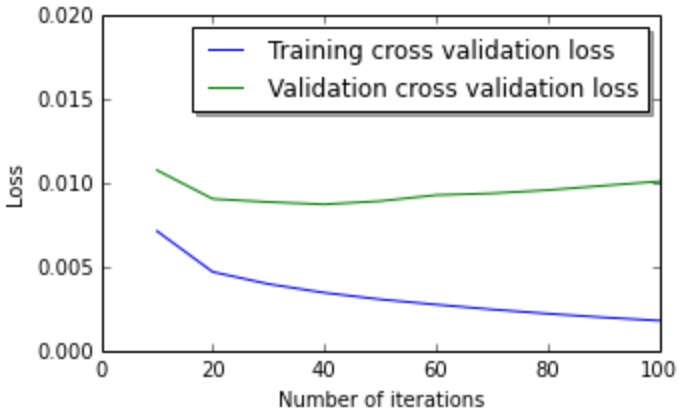


Figure 7: Initial cross validation loss.

Figure 7 clearly shows overfitting. Now, we need to do something to reduce overfitting. First thing we can do is to decrease number of features. Reminding that in the analysis at the beginning of this document, I show that we should use M=15 instead of 30 because the presence of strong autocorrelations. So now, we reduce M to 15, this means that we also should remove the day-of-month feature. Result is in Figure 8.

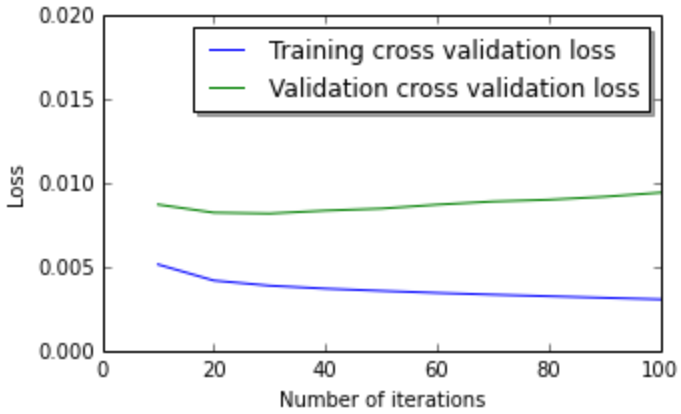


Figure 8: Improved cross validation loss (1).

Great ! Overfitting is reduced but not significantly. So, we should continue by reducing the complexity of the neural network model. We reduce the hid_layers from 500 to 100: hid_layers = (100,). Result is in Figure 9.

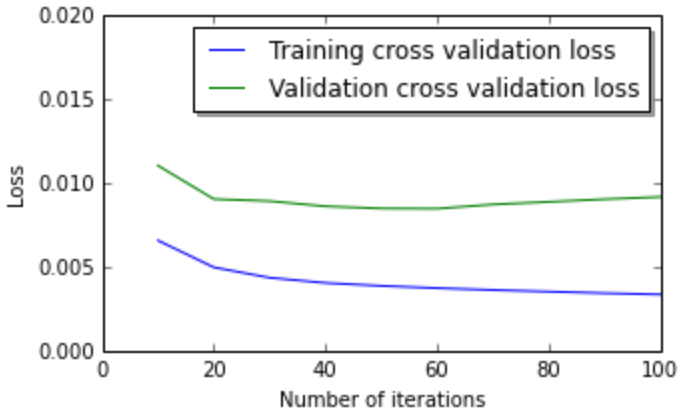


Figure 9: Improved cross validation loss (2).

Just a small decrease of overfitting, probably because the model is still complicated, so continue to reduce hid_layers from 100 to 10 and increase the L2 regularization alpha from 0.001 to 0.01. Result is in Figure 10.

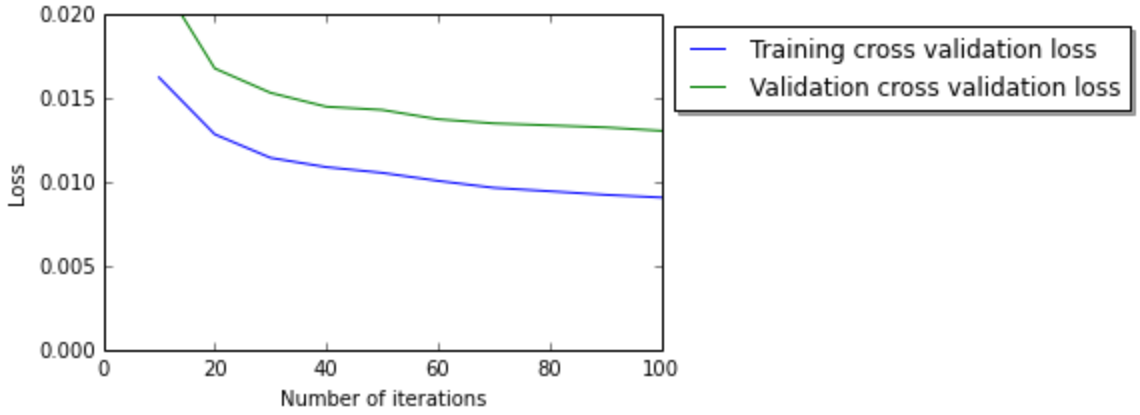


Figure 10: Improved cross validation loss (3).

So what happens now, clearly it's now underfitting. The training loss error increases a lot. Seems that we are over tuning the hyper parameters. Therefore, I will increase the hid_layers to 50. Result is in Figure 11.

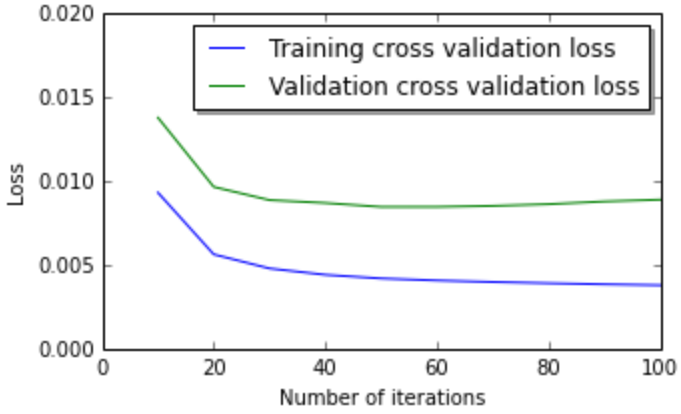


Figure 11: Improved cross validation loss (4).

Great ! Now it's not underfitting but still a little of overfitting. So, now we increase the L2 regularization coefficient alpha to 1. Result is in Figure 12.

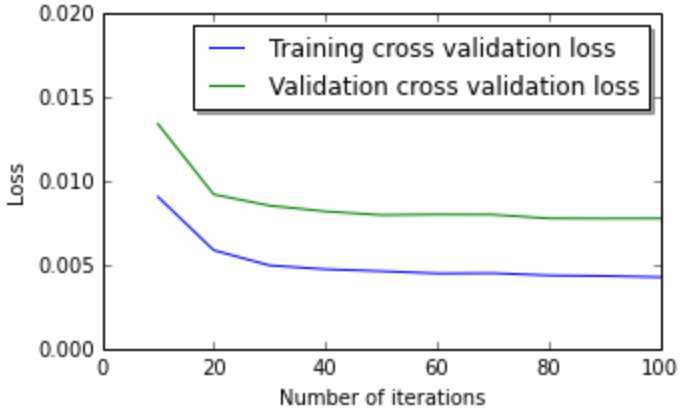


Figure 12: Improved cross validation loss (5).

Okey, so it seems not bad. I stop manually tuning here with 2 “best” hyper parameters found: hid_layers=(50,) and alpha=1. However, we can still continue to automatically do grid search and find better hyper parameters by looking the ranges around these found parameters. (The code of grid search is also provided).

Discussion

We can of course continue to tune the model to get better models. So given results in Figure 12, why we cannot narrow down the difference between training and validation loss? One of the reason is that it belongs to a kind of called irreducible errors. Irreducible errors occur because we totally lack of information in the training set that is important for prediction. Clearly, our data is relatively simple, only with 2 time series. We can only decrease the irreducible errors if we have more information related to the sales such as marketing activities, competitors activities, pricing, etc.

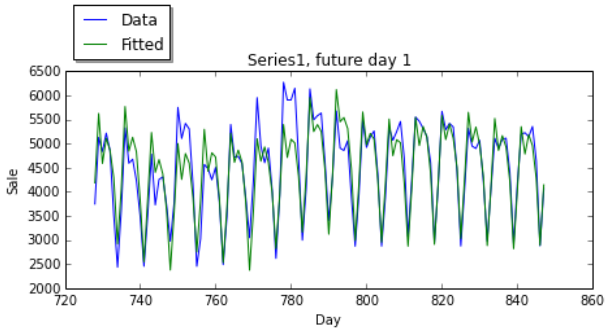
Results

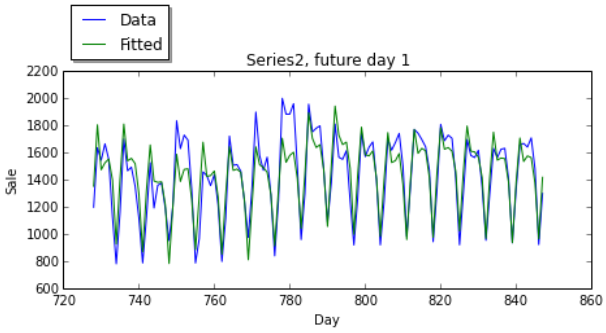
Predict for the Next 15 Days 2015/11/16 - 2015/11/30

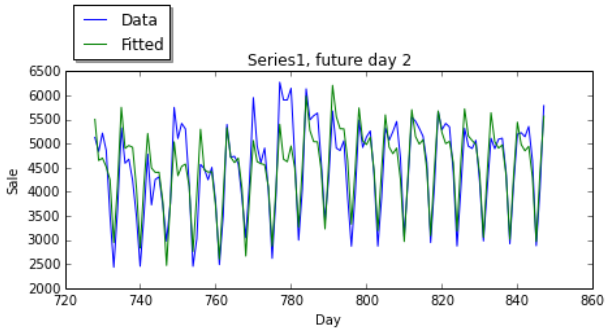
Predict of Product 1: [5754. 5328. 5103. 5405. 4506. 3205. 4506. 5773. 5412. 5161. 5325. 4426. 3166. 4410. 5770.]
Predict of Product 2: [1843. 1715. 1629. 1661. 1443. 1013. 1463. 1884. 1673. 1624. 1762. 1481. 1021. 1436. 1870.]

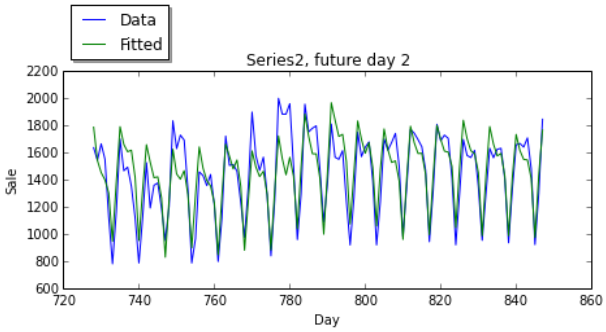
Graphical Output with the model Fitting

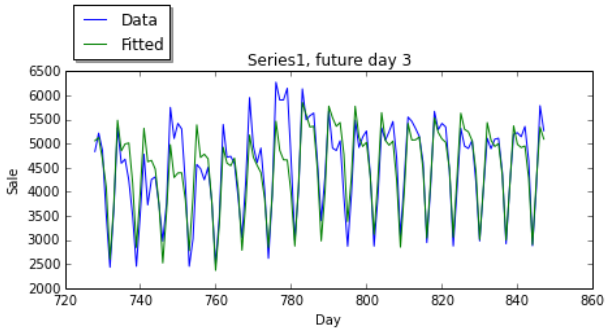
Since it’s hard to visually observe for the whole data set, I only present fitting for the last 4 months. It can be easily adjust in the code to visually observe the whole data set.

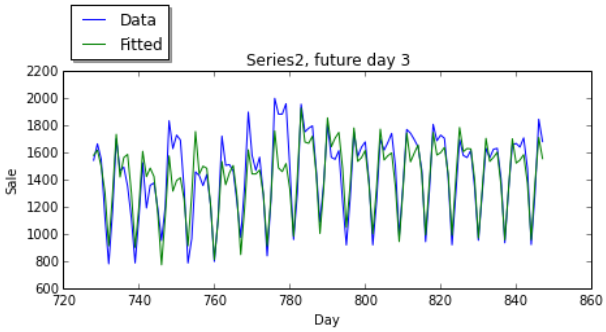


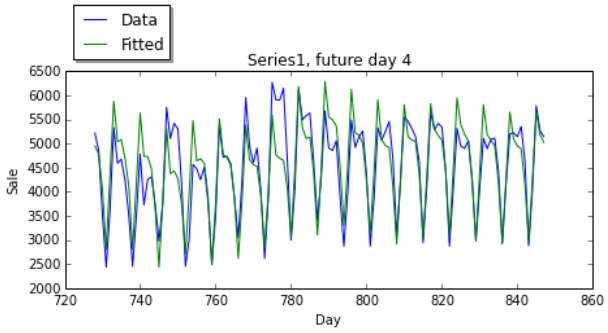


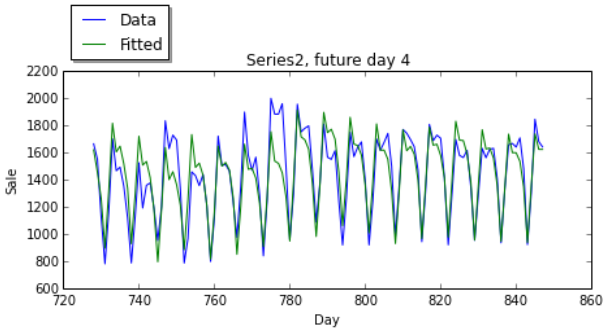

Data
Fitted
Series1, future day 2
Sale
Day

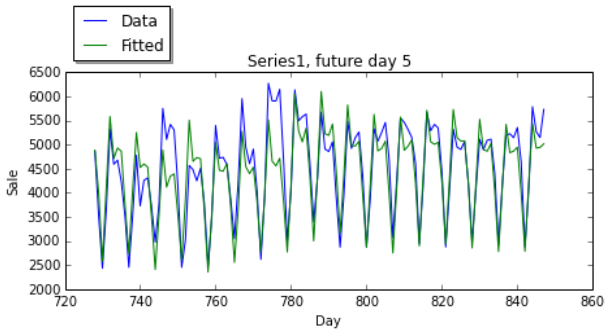

Data
Fitted
Series2, future day 2
Sale
Day

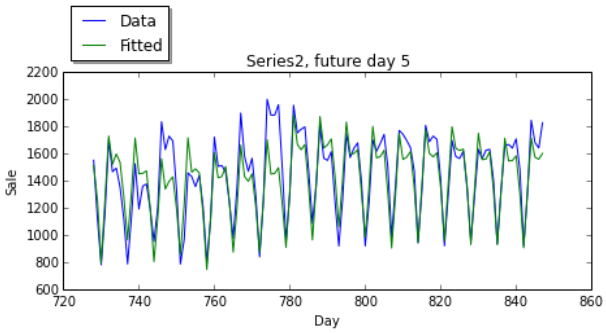

Data
Fitted
Series1, future day 3
Sale
Day

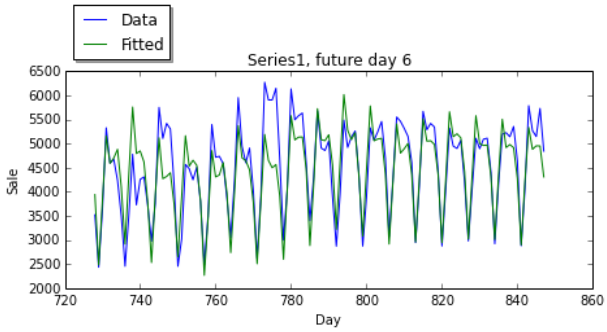

Data
Fitted
Series2, future day 3
Sale
Day

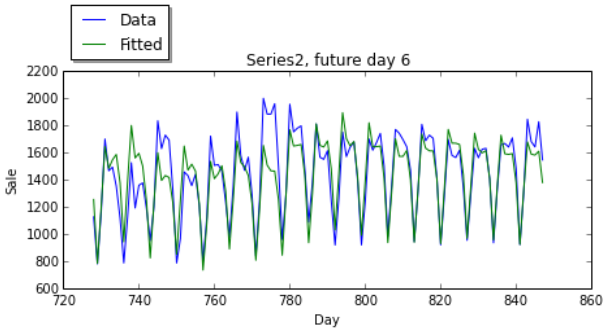

Data
Fitted
Series1, future day 4
Sale
Day

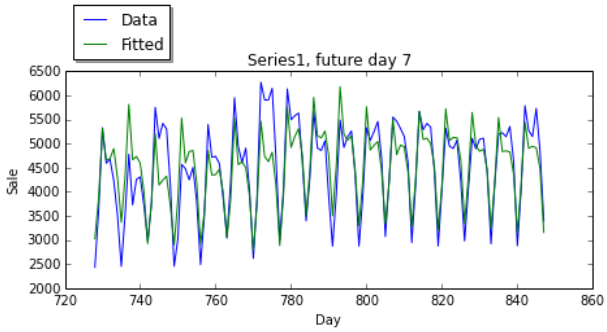

Data
Fitted
Series2, future day 4
Sale
Day

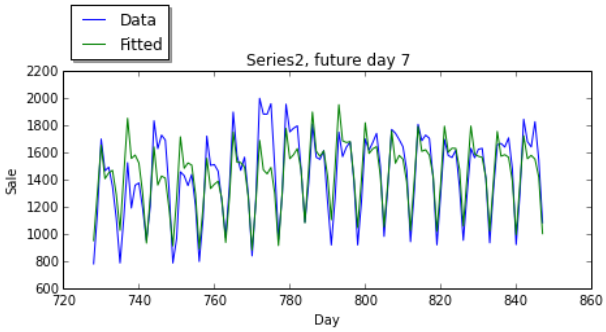

Data
Fitted
Series1, future day 5
Sale
Day

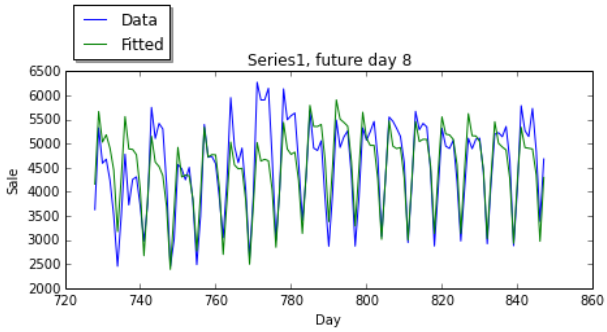

Data
Fitted
Series2, future day 5
Sale
Day

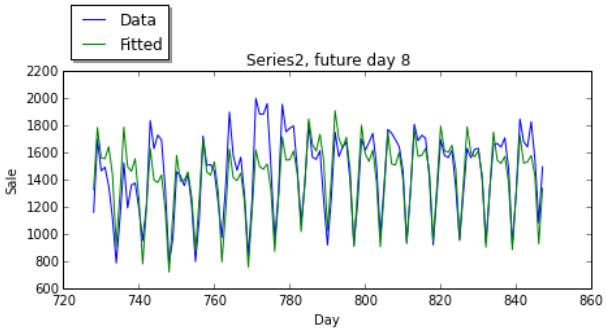

Data
Fitted
Series1, future day 6
Sale
Day

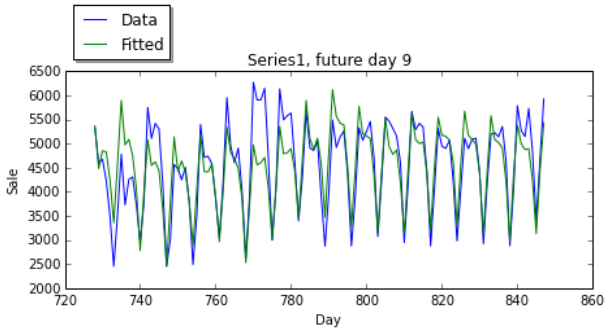

Data
Fitted
Series2, future day 6
Sale
Day

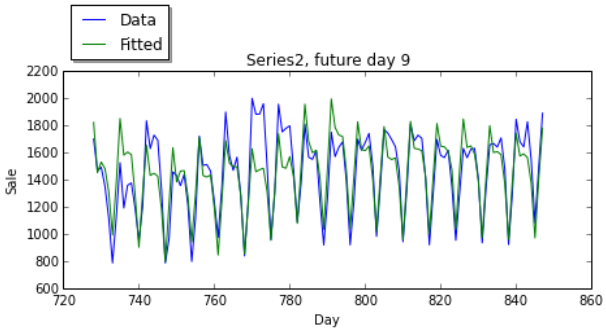

Data
Fitted
Series1, future day 7
Sale
Day

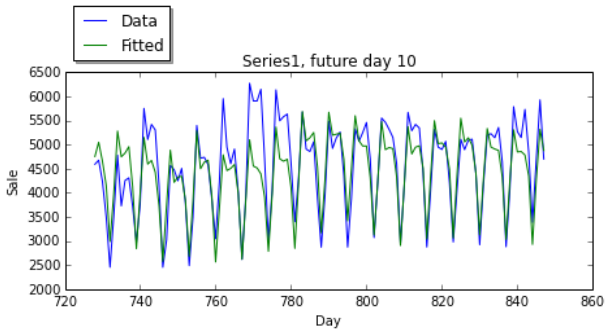

Data
Fitted
Series2, future day 7
Sale
Day

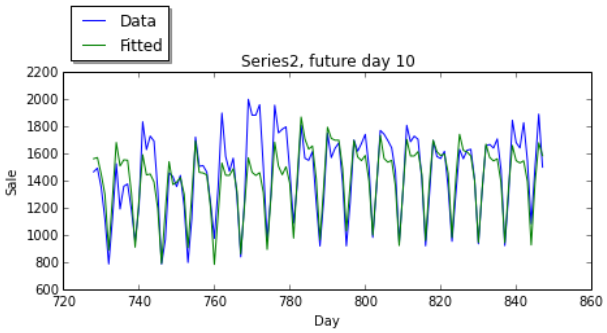

Data
Fitted
Series1, future day 8
Sale
Day

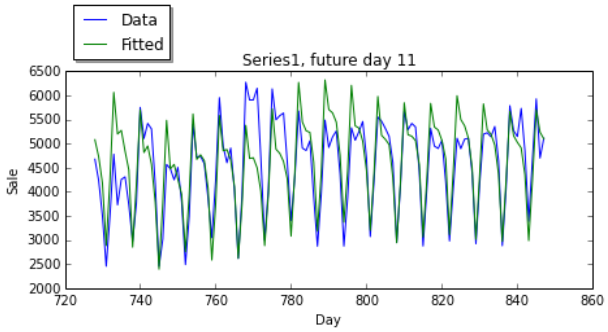

Data
Fitted
Series2, future day 8
Sale
Day

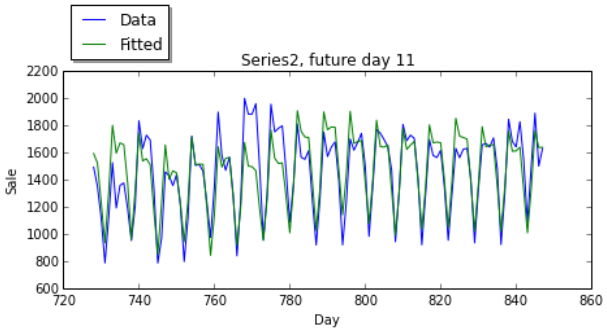

Data
Fitted
Series1, future day 9
Sale
Day

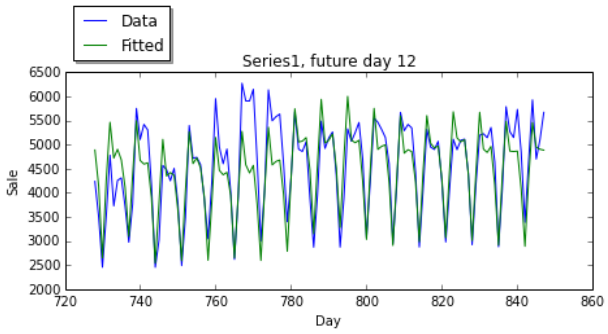

Data
Fitted
Series2, future day 9
Sale
Day

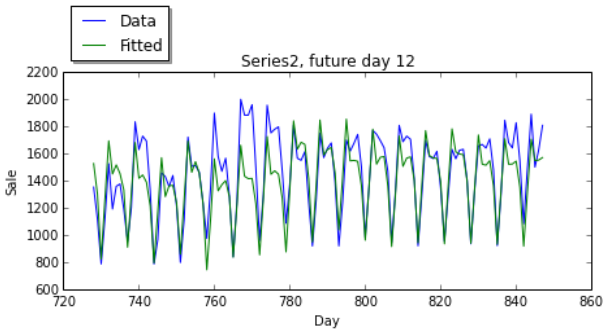

Data
Fitted
Series1, future day 10
Sale
Day

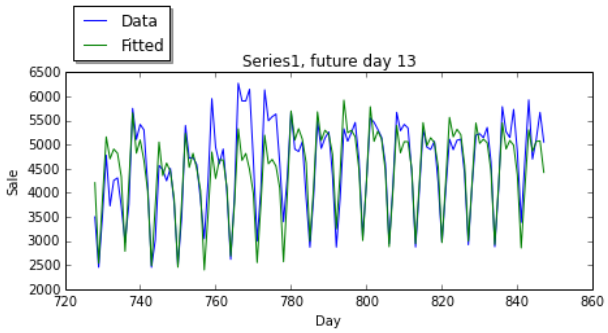

Data
Fitted
Series2, future day 10
Sale
Day

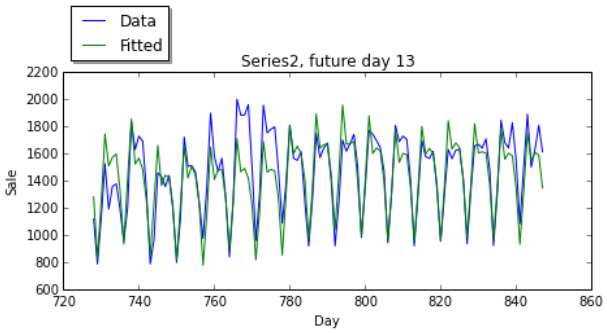

Data
Fitted
Series1, future day 11
Sale
Day

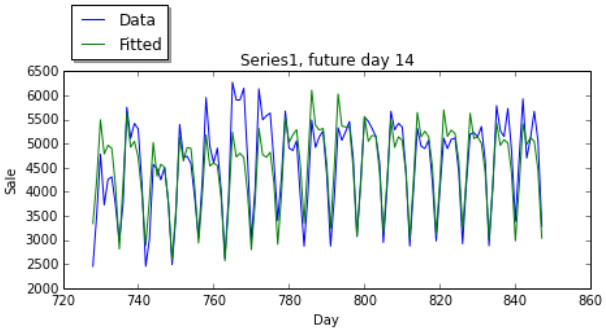

Data
Fitted
Series2, future day 11
Sale
Day

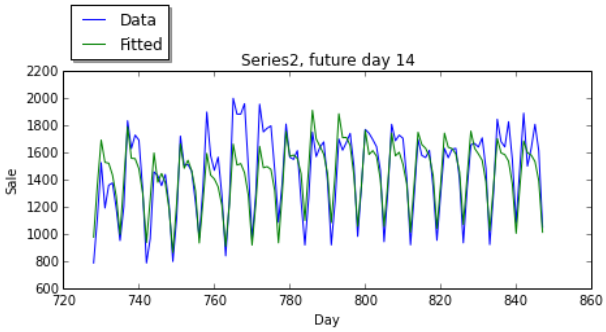

Data
Fitted
Series1, future day 12
Sale
Day

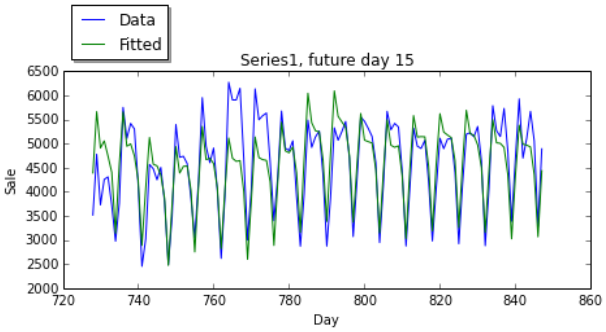

Data
Fitted
Series2, future day 12
Sale
Day

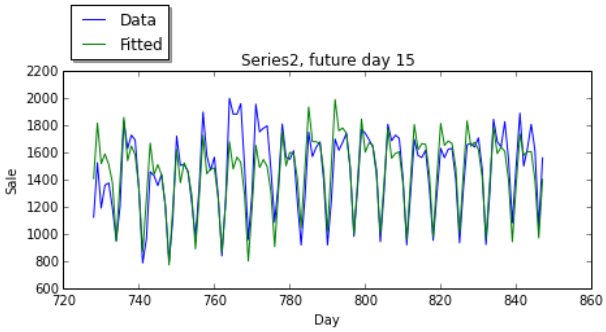

Data
Fitted
Series1, future day 13
Sale
Day


Data
Fitted
Series2, future day 13
Sale
Day


Data
Fitted
Series1, future day 14
Sale
Day
6500
6000
5500
5000
4500
4000
3500
3000
2500
2000
720
740
760
780
800
820
840
860


Data
Fitted
Series2, future day 14
Sale
Day
2200
2000
1800
1600
1400
1200
1000
800
600
720
740
760
780
800
820
840
860


Data
Fitted
Series1, future day 15
Sale
Day
6500
6000
5500
5000
4500
4000
3500
3000
2500
2000
720
740
760
780
800
820
840
860


Data
Fitted
Series2, future day 15
Sale
Day
2200
2000
1800
1600
1400
1200
1000
800
600
720
740
760
780
800
820
840
860