Performance evaluation: Homework 4

Vaubien Jimi 226977, Zimmermann Timon 223720

February 13, 2019

1 Data exploration

Before starting to fit model on the time serie we need to have some information about it, helping us to choose the best model. First, we checked the mean and standard deviation of the time serie over time to get an idea concerning the stationarity: we chose a window of size 100. We can see on Figure 1 the moving mean and standard deviation. Fitting a simple linear model we can see that globally, these values oscillate around a horizontal straight line. We now can say that the serie is stationary.

Being more precise, we used the Augmented Dickey-Fuller test. Here are the results:

ADF Statistic	-8.254
Significance Level 1%	-3.432
Significance Level 5%	-2.432
Significance Level 10%	-2.567

Table 1: ADF test results

Running test gives us statistic value of -8.254 (see Table 1). The more negative this statistic, the more likely we have a stationary serie. We can see that our statistic value of -8.254 is less than the value of -3.449 at 1%. This suggests that the serie is stationary with a significance level of less than 1% (i.e. a low probability that the result is a statistical fluke).

An noteworthy property of the analyzed time serie is his seasonality. Indeed, we clearly see a pattern between weeks, which are at $\delta = 168$ samples (7 days * 24 hours). Consequently, we decided to do a lag plot for this delta which can be found in Figure 3, the plots have been computed for the whole serie as well as for the different significant yearly event, such as Winter holidays (the dates were fetched from the EPFL's Memento). We see that the samples show no obvious correlation when observing their value at an interval of *168*.

More generally, we plotted the autocorrelation for each of these intervals (whole serie, semester 1, Winter holidays, and so on) in Figure 4.

2 First approach: Recurrent Neural Network

The first technique we tried was to train a neural network architecture with our data. Indeed, the recurrent construction of our network, using LSTM cells, allows to feed sequences of values to predict an output. In the case of time serie forecasting, the idea is to feed $H(y(t-k), y(t-k+1), ..., y(t)) = \widetilde{y}(t+1)$, where H is our system (the neural network), the value k represents the memory of the LSTM cells and $\widetilde{y}$ the prediction. The error of our system is then simply characterized by $\epsilon = \widetilde{y}(t+1) - y(t+1)$, but in our case we used a *Mean-Squared Error (MSE)* to emphasize high errors and lower the impact of small errors: $\epsilon = (\widetilde{y}(t+1) - y(t+1))^2$.

The network is then trained using standard back propagation technique with a classic technique of gradient optimization called *RMSProp*. The architecture of the overall network, which can be found in Figure 5, is quite simple, and explained above with the only exception of the *dropout layers* which have been added to avoid over fitting the training set. We also keep track of the training through a validation set consisting of *5%* of the data.

When trying to perform the prediction with this approach, we obtained the result in Figure 6, and the corresponding numerical values:

Date	Predicted	95% CI
05-01-2015 10-11h	175.66	±109.29
05-01-2015 15-16h	182.69	±109.59
05-02-2015 15-16h	176.21	±110.74

3 Second approach: ARMA fitting

The second approach we tried was to fit an ARMA model (ARIMA was not necessary since we found that the serie was stationary, so there was no point in differentiating). To find the p and q parameters we fit different ARMA models and compare them based on their BIC score. We tried different parameters until p=9, q=9 because it was becoming quiet long to fit ARMA with bigger parameter on our laptops. Finally ARMA(9,9) turned out to be the better model (in term of BIC score), so we used this model for prediction. As you can see in Figure 7, this gives us narrower prediction interval than our first model and so we kept this one as the best.

When trying to perform the prediction with this approach, we obtained the result in Figure 7, and the corresponding numerical values:

Date	Predicted	95% CI
05-01-2015 10-11h	195.64	±85.22
05-01-2015 15-16h	208.48	±85.47
05-02-2015 15-16h	203.40	±97.10

4 Discussion about results

We clearly see that the results are better in term of confidence when we use the ARMA fitting. This might be due the incapacity to explicitly embed the seasonality of the serie in the Recurrent Neural Network. One possible extension to do so might be to use the temporal information vectorization, which is a quite new and advanced technique used to augment the possibilities of such network, but that was way out of the scope of the project.

The introduction of different model for each event of the scholar year (Winter holidays, Easter holidays and so on), might also be a good trail to increase the likelihood of the forecasting to represent the actual data and thus improve results. Another approach to boost performances might have been to find open-sourced electrical consumption time series, in order to use their correlations with our own serie and get a better forecast of future dates.

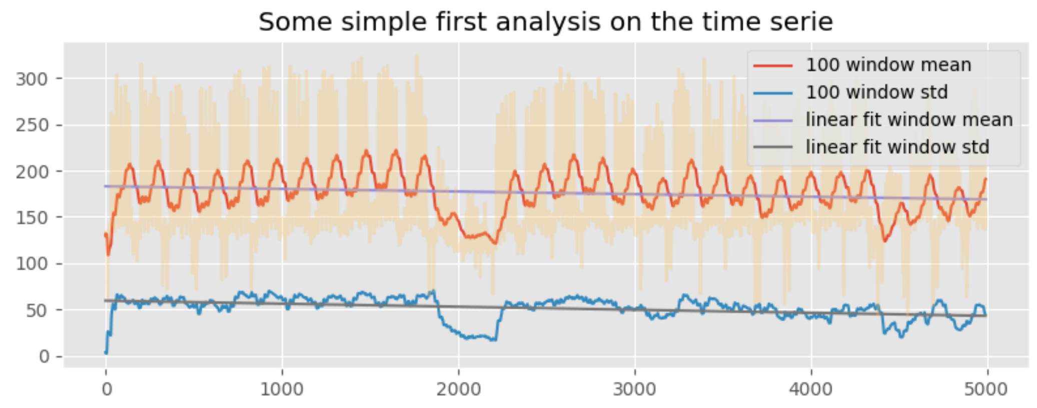

Figure 1: Full time serie property analysis

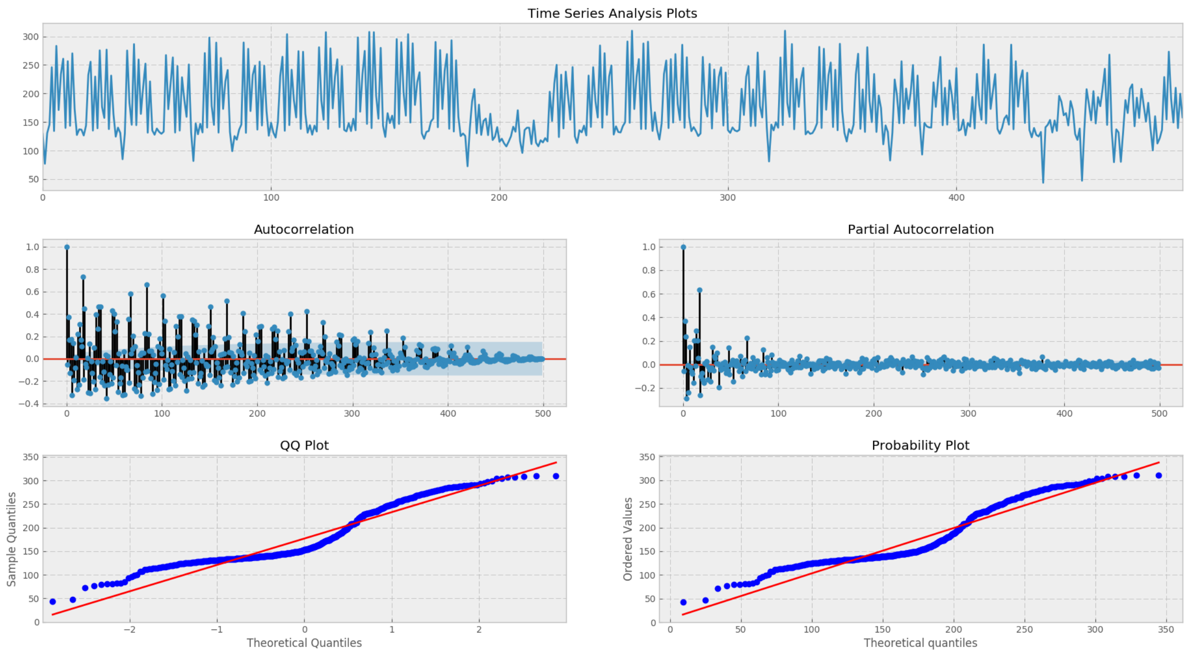

Figure 2: Full time serie property analysis part 2

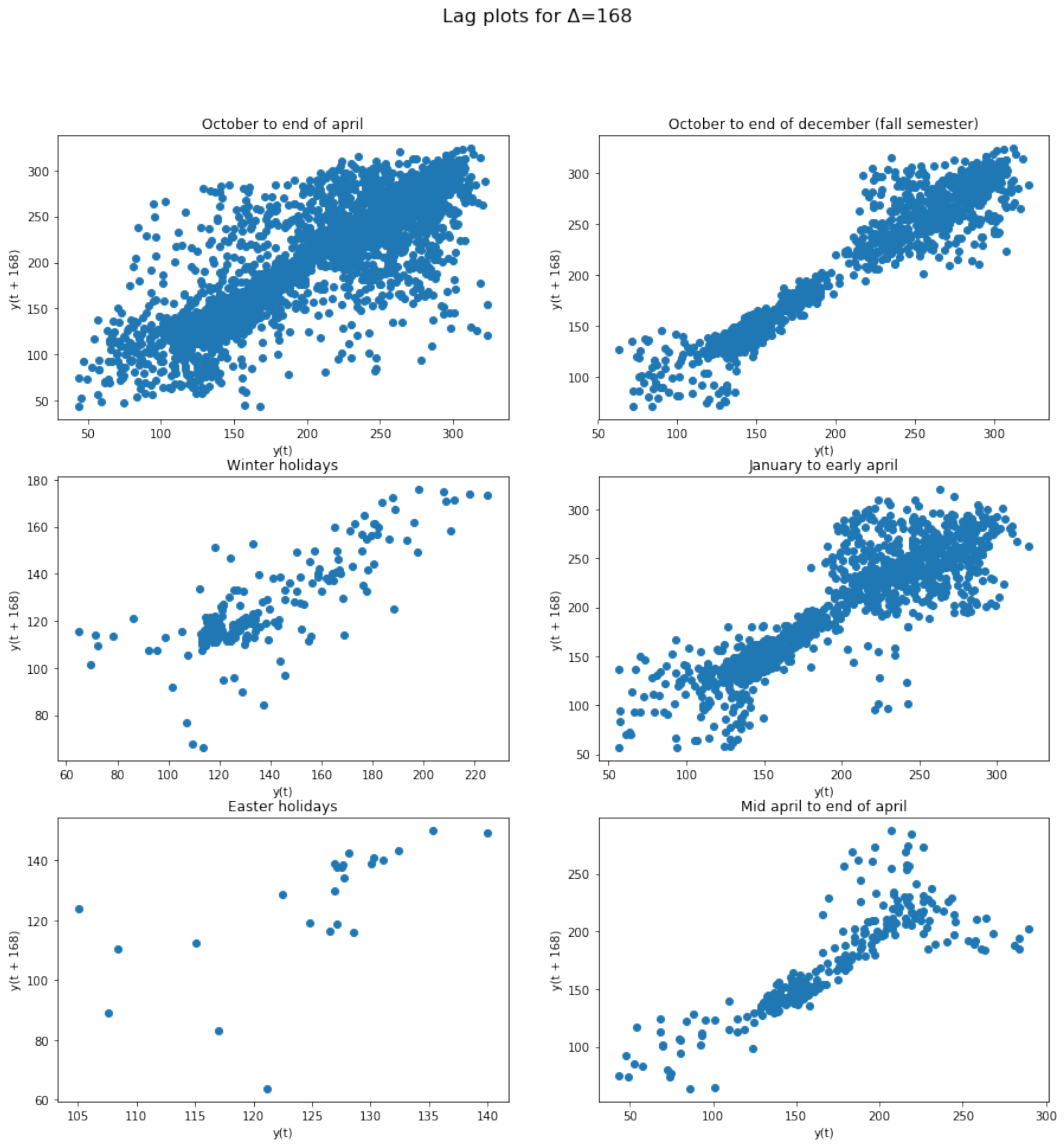

Figure 3: Lag plots for the whole serie and different significant parts of it

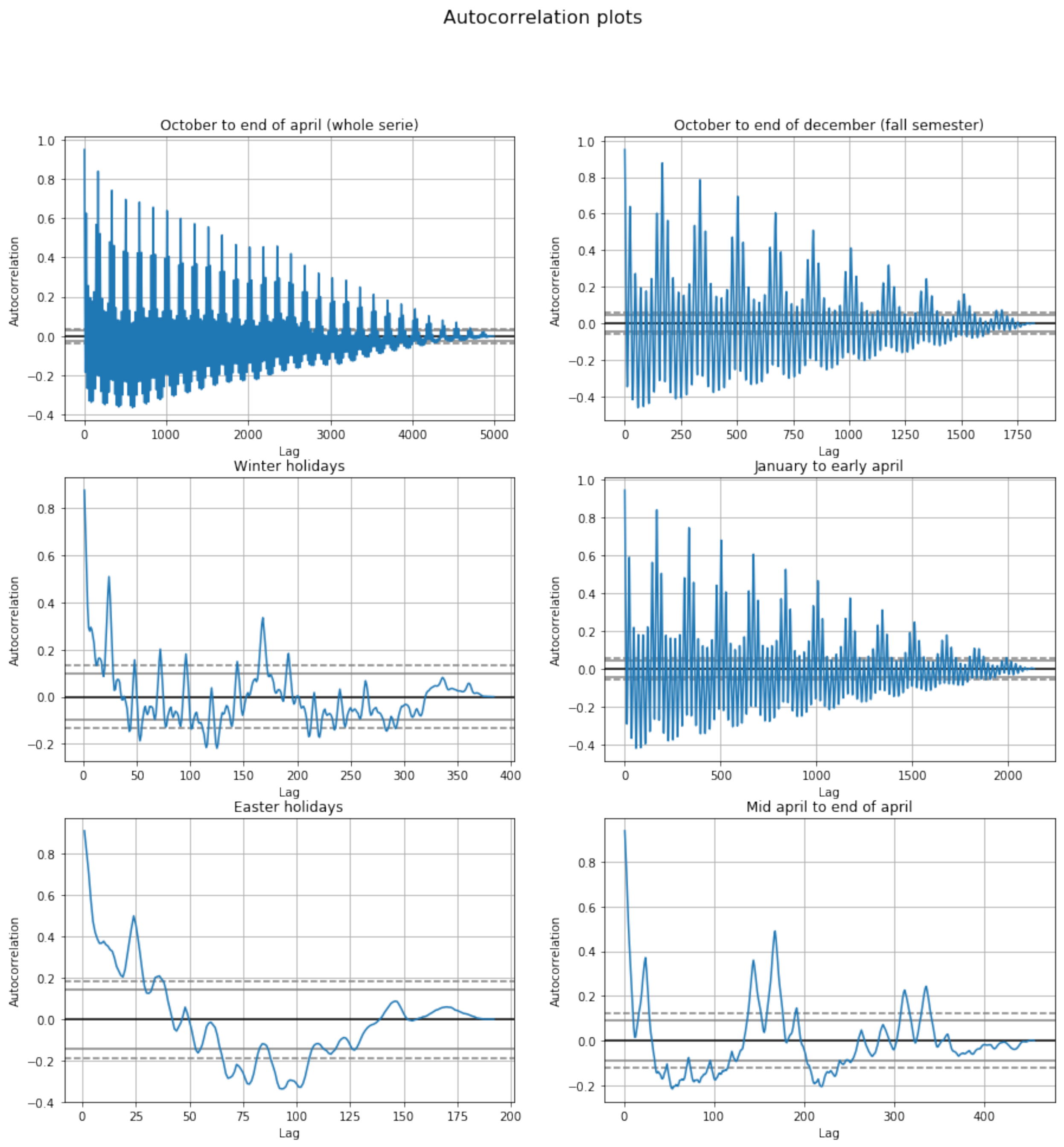

Figure 4: Autocorrelation plots for the whole serie and different significant parts of it

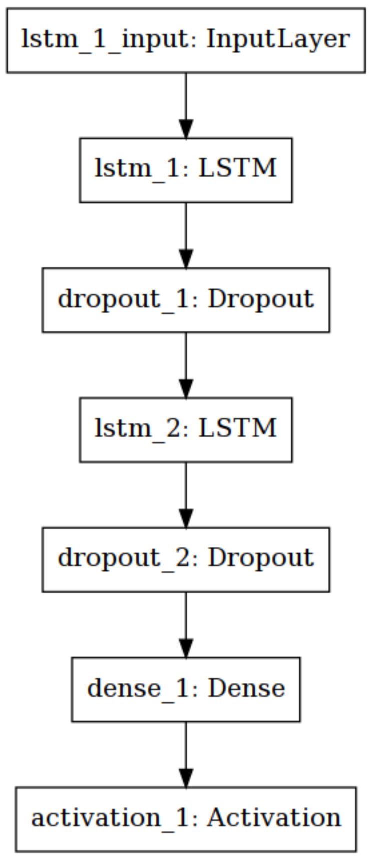

Figure 5: The architecture of our Recurrent Neural Network, made with Keras

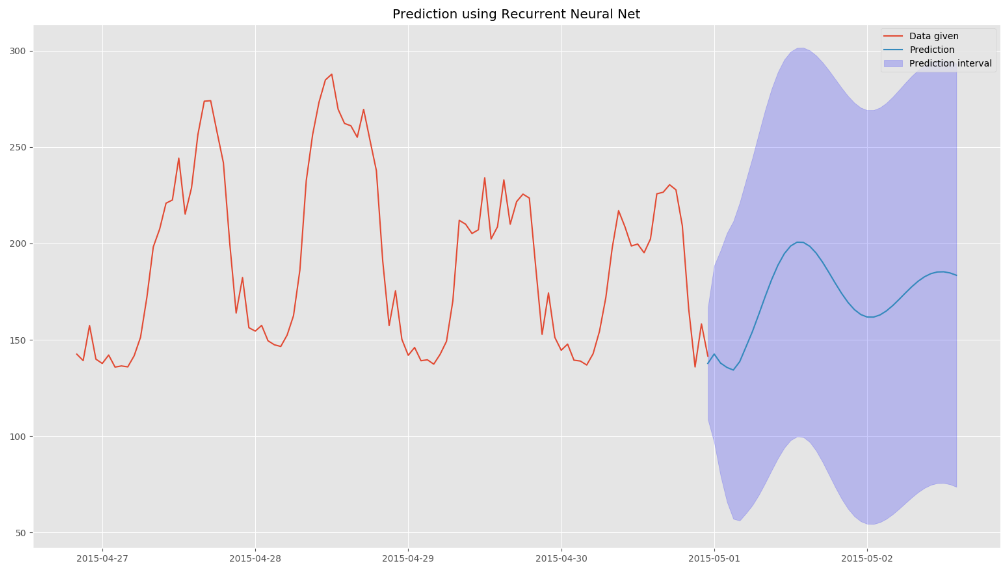

Figure 6: Predicted values for the first approach and its corresponding prediction interval

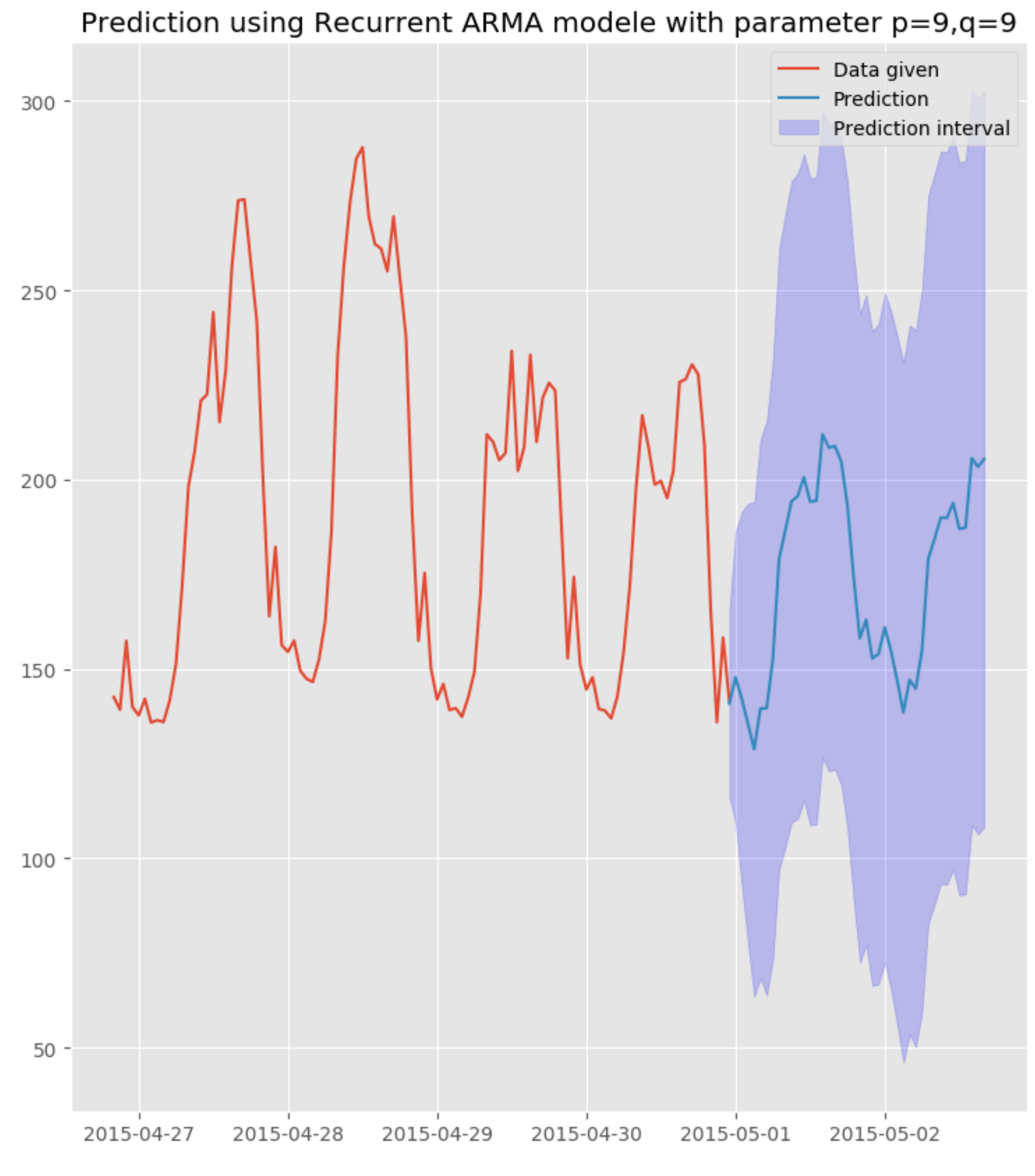

Figure 7: Predicted values for the second approach and its corresponding prediction interval