One Step Ahead Forecast of Household Electricity Consumption using SVR and ELM

Nabarun Pal
Indian Institute of Technology, Roorkee
pal.nabarun95@gmail.com

ABSTRACT

Electricity Load Forecasting remains one of the most important forecasting problems in the power generation industry. This paper aims to create a one-hour ahead forecasting model which utilizes 20 previous time lags of data using Support Vector Regression (SVR) and Extreme Learning Machines (ELM). ELM was found to have better performance for this dataset.

1. Main Objectives

- ✓ Forecasting household electricity consumption
- ✓ Comparison of various models like SVR, ELM, ARIMA, RNN LSTM

2. Status and other details

- ➢ Status – Complete for some models
- ➢ Total Time Spent
  - o Literature Review – 2 weeks
  - o Dataset Selection and Pre-processing – 1 week
  - o Codes – 3 days
  - o Parameter Selection – 2 days
  - o Report Writing – 2 days
- ➢ Further Work – Incorporation of other features

3. Major Stumbling Blocks

- ✓ Pre-processing the data
- ✓ Resampling of the data
- ✓ Imputation of data
- ✓ Python implementation of the models chosen

4. Introduction

Electricity demand prediction is very important for the power sector because how much electricity at a certain time is to be produced depends upon this. There are many applications such as switching of loads, energy purchase, generation and efficient distribution [1]. There are various forecasting models developed for this problem, but still the search for better methods is in continuance. Accurate predicitive models have always been very important part of planning and operation of an electricity ccompany. There are three categories into which load forecasts can be categorised: short-term forecasts, medium term forecasts and long-term forecasts.

In this report, we will focus on short-term forecasts in the order of hours. The report is further divided into different sections concerning different aspects. Section 5 describes the dataset, its source and its attributes. Section 6 focuses on explaining the different models used in this report. Section 7 discusses how the dataset was processed for model fitting. Section 8 describes some of the performance metrics used in the analysis. Section 9 discusses about the results obtained. Section 10 concludes the report mentioning the highlights of the experiment. The codes used in this report is open sourced at Github. [2]

5. Dataset

The dataset consists of measurements of electric power consumption in one household with one-minute sampling rate over a period of 4 years.[3] The data contained various attributes such as the active power used by the household, reactive power consumed, the time averaged voltage over that sampling window, the time-averaged current. The household in concern also had three sub-meters corresponding to different sections of the house. The first sub-meter reading gives the active power consumed by the kitchen which contains appliances such as dishwasher, oven, microwaves etc. The mentioned appliances being the most power hungry. The second sub-meter reading represents the laundry room, where washing-machine, tumble-drier and a light are the appliances. The third sub-meter calculates power consumption for an electric water-heater and an air-conditioner.

There are 2075259 data samples from December 2006 until November 2010. The samples have a one-minute sampling window. Nearly 1.25% of the rows are empty and those data points are not considered in the final training which is explained in a detailed manner in the Pre-processing section.

6. Review of Models

1. *Support Vector Regression (SVR)*

Support Vector Regression (SVR) is a subpart of a learning technique called Support Vector Machines (SVM). They are a type of supervised learning algorithm which implements structural risk minimization (SRM) inductive principle to obtain a better generalization on sparse data [4]. In addition to linear classification tasks, they can also do non-linear classifications using kernel trick which maps features to higher dimensional spaces [5]. The mathematical of SVR can be done by adding slack variables ξ_i, ξ^*_i, to the SVM convex optimization problem. (Eq.1)

Minimize

$$\frac{1}{2}\|w\|^2 + C\sum(\xi_i + \xi_i^*)$$

such that

$$\begin{cases} y_i - \langle \omega, x_i\rangle - b \le \varepsilon + \xi_i \\ \langle \omega, x_i\rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases} \quad ...1$$

Where,

||ω|| is the weight function,

X_i the inputs,

y_i the corresponding outputs with a ε precision boundary and b as the bias.

The constant C determines the trade-off or compromise between the flatness of Euclidean norm and the toleration for error greater than the precision boundary. We can formulate the langrange's dual problem using Langrangian multipliers and solve the optimization.

Maximize

$$-\frac{1}{2}\sum(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle x_i, x_j\rangle - \varepsilon\sum(\alpha_i + \alpha_i^*) + \sum y_i(\alpha_i - \alpha_i^*)$$

such that

$$\sum(\alpha_i - \alpha_i^*) = 0 \; and \; \alpha_i, \alpha \in [0, C] \quad ...2$$

Where, α_i, α^*_i >0 are the Lagrangian Multipliers following Karush-Kuhn Tucker condition:

$$f(x) = \sum(\alpha_i - \alpha_i^*)\langle x_i, x\rangle + b \qquad ...3$$

The above formula gives us the regression prediction for any particular input. This is what creates a linear separation boundary or a linear regression formula. Non-linear mappings can be created using a Kernel Function which maps the features into a higher dimensional space.

$$k(x_i, x) = \emptyset(x_i)\emptyset(x) \qquad ...4$$

The prediction can therefore now be written as:

$$f(x) = \sum(\alpha_i - \alpha_i^*)k\langle x_i, x\rangle + b \qquad ...5$$

There are various Kernel Functions like linear kernel, Gaussian kernel, RBF kernel which have their own merits and demerits.

2. *Extreme Learning Machines (ELM)*

ELM's were devised by G. B. Huang, L. Chen and C. K. Siew in 2006 [6]. There were many open ended questions like was it really necessary to have so many types of activation functions and architectures to universally approximate functions. They devised ELM to be a Single Hidden Layer Network with Radial Basis Function (RBF) neurons where the hidden the layer weights were generated randomly and the output layer weights were calculated using Moore-Penrose generalized inverse matrix (Eq.8). Due to random generation of hidden layer weights and simplified training architecture, this network was very fast and required very minimal human intervention. The data is first mapped to the RBF feature space using Eq.6

$$f_l(x) = \sum \beta_i h_i(x) = h(x)\beta \qquad ...6$$

H in Eq.7 represents the hidden layers activations and β represents the output layer weights with T being the target outputs.

$$H\beta = T \qquad ...7$$

Now, using the Moore-Penrose generalized inverse of H, β can be written as

7. Pre-processing and Tools Used

The dataset had large number of instances which were resampled to hourly intervals by summing over The period as they were energy consumption estimates. This was done so as to reduce the number of samples as well as make the data robust as minute by minute forecast is hardly needed. Then the data was normalized by subtracting the mean over the whole dataset. This was done to increase the training efficiency of the models. By analysing the correlation factors of the data with previous values, it was concluded that 20 time lags was optimum for the forecast.

8. Results – Charts and Tables

Models		MSE	RMSE	MAE
ELM	1000	1413.948	37.60	26.70
	100	1590.515	39.67	28.83
SVR	(10, 0.1)	2629.022	51.27	33.67
	(10, 0.001)	2629.270	51.28	33.66
	(100, 0.1)	204.015	14.28	2.8
	(100, 0.001)	204.121	14.29	2.71

Table 1. Training Errors

Models		MSE	RMSE	MAE
ELM	1000	1162.617	34.09	24.57
	100	1202.257	34.67	25.57
SVR	(10, 0.1)	2139.682	46.26	35.86
	(10, 0.001)	2139.849	46.26	35.86
	(100, 0.1)	2008.703	44.82	37.7
	(100, 0.001)	2008.545	44.81	37.69

Table 2. Testing Errors

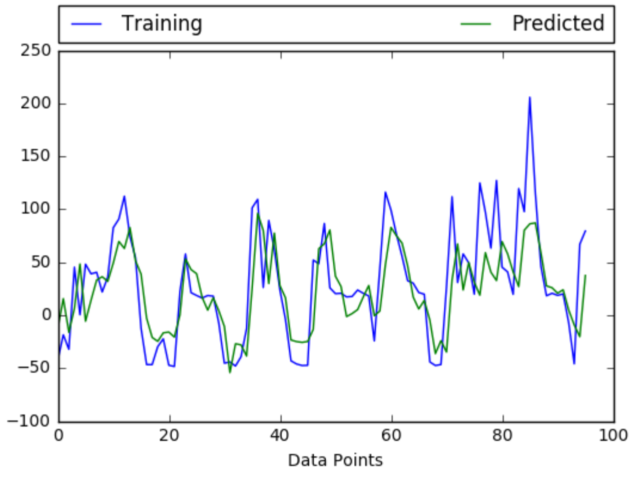
The tables show the metrics for various models prepared. Two networks with 100 and 1000 hidden RBF neurons were made for ELM models. Four SVR models were trained with two different Trade-off parameters and two different precision boundary parameters. In the table, the first value inside the brackets represent Cost and the second value represents Precision Boundary.

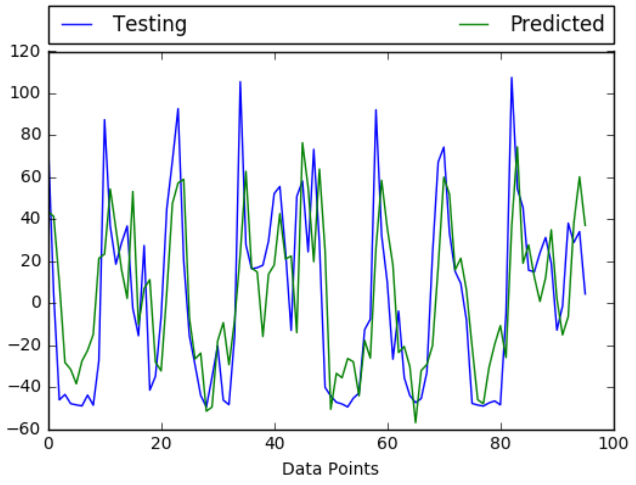
$$\beta = H^{\dagger}T \qquad \ldots 8$$

Generalization can also be improved by adding a regularization term.

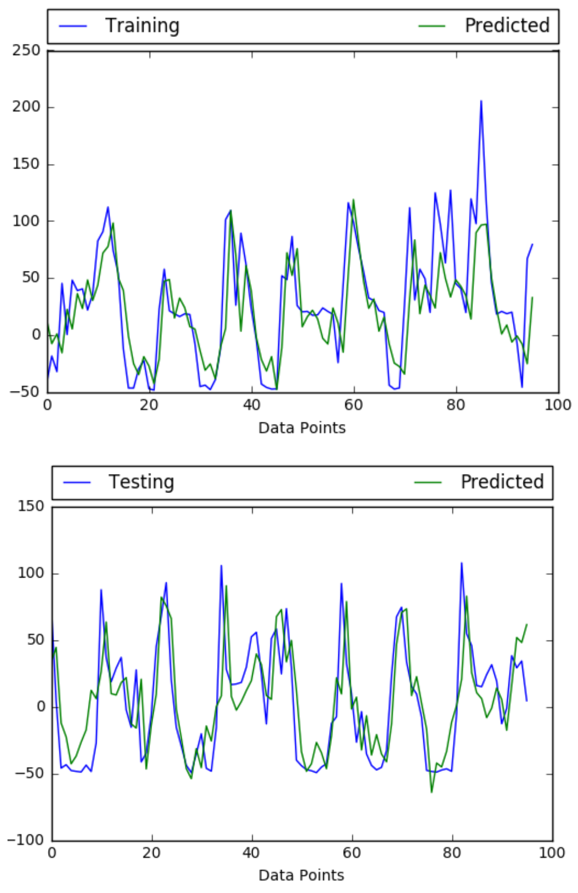
The dataset was then divided into Training and Testing sets in the ratio 80:20. The SVR modelling was done using LibSVM [7] and Scikit-learn SVR [8] [9] model which uses LibSVM as its backend. The ELM model was made using a library called hpelm [10] [11].

The performance of the models were checked using Mean Squared Error (MSE) [12], Root Mean Squared Error (RMSE) [13] and Mean Absolute Error (MAE) [14]. These are some of the most common performance indicators used in regression problems.

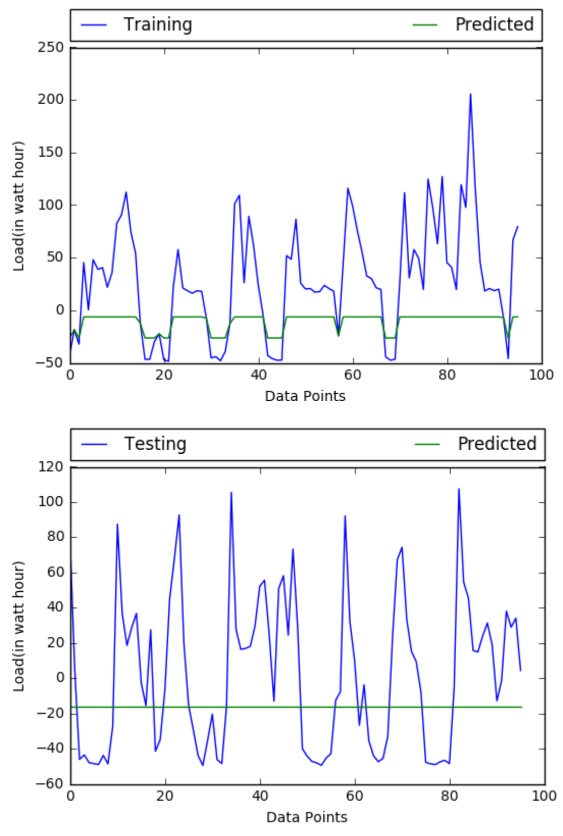




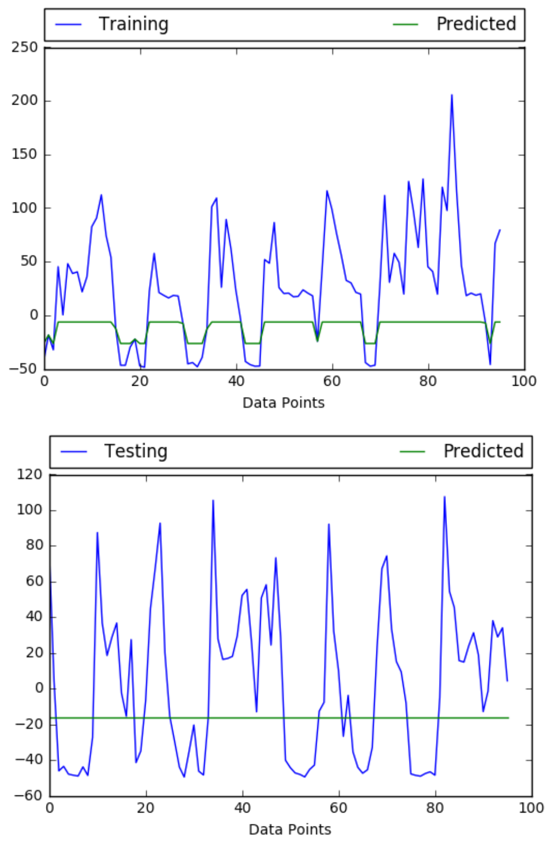
Plot 1a (Training) and 1b(Testing). ELM with 100 hidden neurons



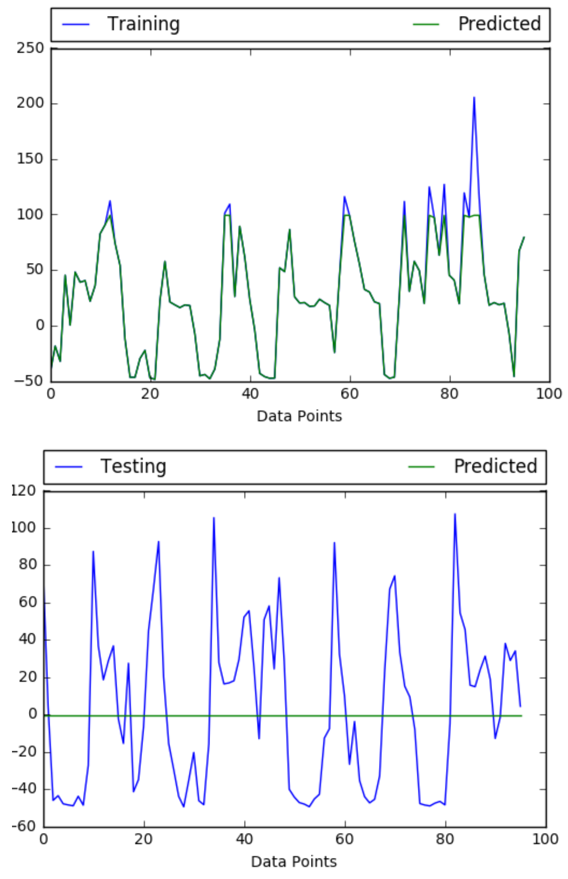
Plot 2a (Training) and 2b (Testing). ELM with 1000 hidden neurons



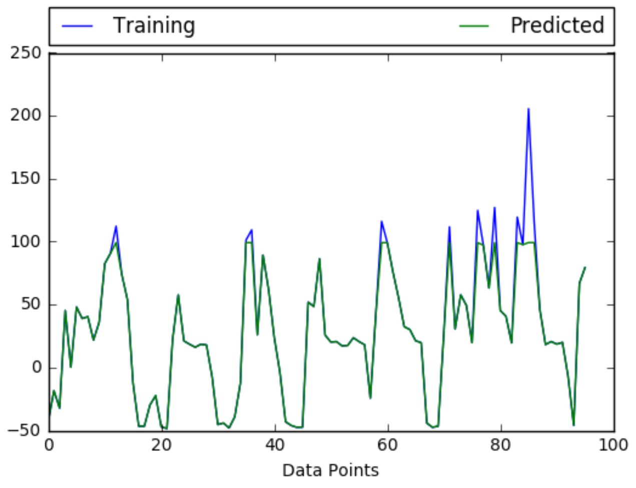
Plot 3a (Training) and 3b (Testing). SVR with C=10 and e=0.001

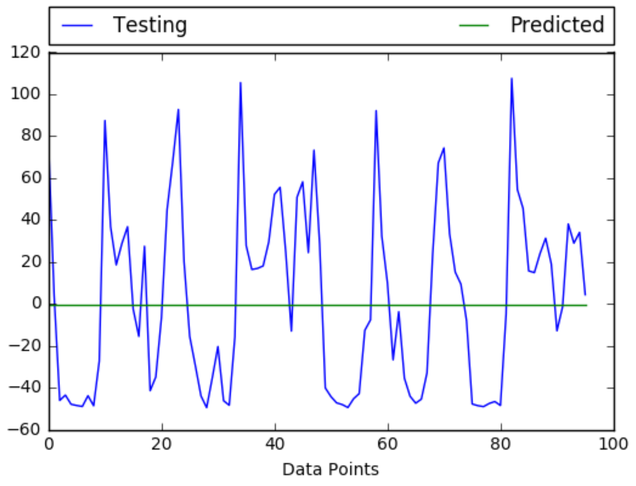


Plot 4a (Training) and 4b (Testing). SVR with C=10 and e=0.1



Plot 5a (Training) and 5b (Testing). SVR with C=100 and e=0.001





Plot 6a (Training) and 6b (Testing). SVR with C=100 and e=0.1

Note: All plots have Normalized Active Power as their y axis

9. Discussion

The results indicate that during training, the SVR models with C=100 perform the best. But, that is not the true indicator to a model's performance. The model should be judged on how it performs on unknown data. The results prove that ELM performs better on this specific dataset. In terms of both Mean Absolute Error and Root Mean Square Error, the ELM models perform better. ELM's accuracy is seen to be increasing with increasing number of hidden layer neurons, which also goes along with the theory that networks with more hidden neurons can fit more complex functions.

If only SVR is considered, it performs better with increasing C. But, the models simulated have high over-fitting. This makes them unsuitable for real life applications. This may also be attributed to the low precision boundary parameter. As a lower ε results in the model to have less room for tolerance.

10. Conclusion

It can be concluded that electric load forecasting is indeed possible with conventional data driven models with careful pre-processing of data and choice of model parameters. Support Vector Regression performs well in training, but that is due to over-fitting. It performs poor in testing. Extreme Learning Machines are better in this regard. They offered better generalisation. Also it is faster than Least Squares-SVM. Further on, Deep Neural Network Architectures like Recurrent Neural Networks and Long Short Term Memory Units can be used to model the complex patterns in the data.

11. References

[1] Steven Mill, "*Electric Load Forecasting: Advantages and Challenges*", http://engineering.electrical-equipment.org/electrical-distribution/electric-load-forecasting-advantages-challenges.html
[2] Nabarun Pal, "*Household Electricity Load Forecasting*", https://github.com/paliitr/Electricity-Load-Forecasting
[3] Georges Habrail, Alice Bacrard, Telecom Paris Tech, Clamart, France, "*Individual household electric power consumption Data Set*", https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption
[4] GEP Box, GM Jenkins, "*Time Series Analysis and Control*", 1976
[5] V. Vapnik, S. Golwich, A. Smola, "*Support Vector Method for Function Approximation, Regression, Estimation and Signal Processing*", in M. Mozer, M. Jordan, and T. Petsche (eds.), Neural Information Processing Systems, Vol. 9. MIT Press, Cambridge, MA, 1997
[6] G.-B. Huang, L. Chen and C.-K. Siew, "*Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes*", IEEE Transactions on Neural Networks, vol. 17, no. 4, pp. 879-892, 2006.
[7] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011.
[8] Scikit-learn SVR Documentation, http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html
[9] Scikit-learn SVM Documentation, http://scikit-learn.org/stable/modules/svm.html
[10] hpelm 1.0.4 PyPi, https://pypi.python.org/pypi/hpelm/1.0.4
[11] HP-ELM Documentation, http://hpelm.readthedocs.io/en/latest/
[12] Mean Squared Error, Wikipedia, https://en.wikipedia.org/wiki/Mean_squared_error
[13] Root Mean Squared Error, Wikipedia, https://en.wikipedia.org/wiki/Root-mean-square_deviation
[14] Mean Absolute Error, Wikipedia, https://en.wikipedia.org/wiki/Mean_absolute_error