Electricity Load Forecasting and Its Impact on Energy Cost Minimization

A Masters Project Report

Submitted in partial fulfillment of requirements for the degree of

Master of Technology

by

Salke Surendra Subhash
Roll No : 113050003

under the guidance of

Prof. Krithi Ramamritham

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay

June, 2013

Abstract

Existing rate of resource consumption and environmental impact is not sustainable. The rising global demand for energy is best addressed by adopting sustainable methods of power consumption and energy conservation. Load forecasting can be very useful in energy conservation. Load forecasting is usually made by constructing models on relative information, such as climate and previous load demand data. We use online resources that are available in typical office building as an attribute to train Support Vector Regression (SVR) model that can predict building energy use for per 15 minute time intervals. Cost minimization techniques uses forecasted energy demands to plan for possible savings. We study the impact of mispredictions on battery based cost minimization algorithm. We propose an extension to cost minimization algorithm.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Existing rate of resource consumption and environmental impact is not sustainable. Burning fossil fuel increases the emission of greenhouse gas. India's electricity-related emissions are high, as we rely primarily (56% of India's installed electricity capacity)[1] on coal for electricity generation and coal is the most greenhouse-intensive fuel. The rising global demand for energy is best addressed by adopting sustainable methods of power consumption and energy conservation.

1.1 Peak Power Demand

Electricity demand varies constantly. At times of low demand, only the utility's lowest marginal cost plants operate, while at peak times, almost all of the utility's available power plants must run to meet the demand and prevent system outages.

Peak period is a time in which electrical power is expected to be provided at a significantly higher rate than average supply.
The reason behind peak demand is electricity cannot be stored cost-effectively and thus must be supplied at the same time that it is being used. Many power plants are usually over provisioned for such peak power demand. Over provisioning of resources is cost intensive. Peak power plants are used for very small time duration hence maintenance cost of peak power plant is very high. The lowest marginal cost plants are often the most fuel efficient. It is common practice among the power production companies to select most inefficient power generation plant to meet the peak power demand. This cause even more CO_2 emission during the peak.

1.1.1 Effect of peak on the grid

Peak demand make power grid unstable. Recently one such incident was reported in India. On July-30-2012, parts of north India came to a standstill, crippled with power failure that affected the many states . This power failure was attributed to overdrawing of power by some states. After this incident, northern grid was successful in partially restoring power in many parts. After 15 hours, the northern grid collapsed again on the next day, July 31.This time, a cascading effect saw the collapse of the eastern and north eastern power grids as well.

To minimize the effect of peak demand power companies sets a new pricing model called as *peak pricing* model. Peak pricing model is slab based pricing scheme with three slabs as *off peak, mid peak and on peak.* Cost of electricity increases if users electricity demand goes from *off peak* slab to *mid peak* slab, or from *mid peak* slab to *on peak* slab. Elevation in the cost of electricity in *on peak* slab can be as much as 80% to 90% as compared with *off peak* slab.

[1] http://en.wikipedia.org/wiki/Electricity_sector_in_India

1.2 Techniques to flatten the electricity demand

Fattening electricity demand means reducing the difference between elevated energy consumption and average energy consumption. Peak shaving using batteries and appliance scheduling are the two techniques to flat the peak power demand. We will discuss theses techniques in this section.

1.2.1 Battery based peak shaving technique

Use of battery as a backup power source is very common in many houses. Battery can be used as a buffer for storing power when price of electricity is low. When demand for the electricity goes beyond the threshold, charged batteries can be used to reduce the power drawn from grid. This novel approach can help in saving energy cost significantly. The charging and discharging decision of the batteries decides the saving opportunities.

Predicting the future electricity demand will assist the battery based peak shaving technique to make intelligent charging and discharging decision.

1.2.2 Appliance scheduling

Electrical appliances can be categorized in to two classes as *interactive* and *non-interactive* appliances. Lighting and entertainment electrical appliances fall in the *interactive* category. Air-conditioners, dish-washers, dryers, refrigerators, dehumidifiers belong to class of *non-interactive or background* appliances. *Non-interactive* appliances can be scheduled such that peak energy demand can be avoided or reduced.

Predicting the future electricity demand and predicting the possible appliances that contributes to power consumption will assist in scheduling the background load.

1.3 Energy usage forecasting for flattening the peak demand.

We have seen in the section 1.2 how battery based peak shaving technique and appliance scheduling can help in peak power reduction. Battery based peak shaving technique need to make intelligent decision about whether to draw power from battery or grid. Battery based peak shaving technique will require knowledge of future power consumption demand to take charging or discharging decisions.

Forecasting electricity usage will also help in appliance scheduling. If forecast shows high energy consumption for certain period, then the background load scheduled in that interval can be shifted to *off peak* interval.

1.4 Problem Definition

Since last two decades the use of computer and Internet is increasing. Many corporate, academic and government agencies started using this computing infrastructure to adopt paperless office procedures. Use of IT in office buildings generates large amount of digital data. When users of IT use this infrastructure for their every days activity, they leave behind digital footprint of usage.

We propose to use this digital footprint to forecast electricity usage in office buildings. We considered KRESIT building for our experiment. We propose to use following online data for electricity usage prediction. *Weather information, departmental time table, room scheduler, departmental events, CSE web calendar, Professor's google calendar, student notices sent on institute mailbox, news forum on the moodle* etc. Many of these attributes can directly affect the usage of electricity.

In section 1.2.1 we have seen how energy storage technology can help in reducing the energy cost by shaving the peak electricity demand. Cost minimization technique uses day ahead forecast of energy demand to decide upon the saving window. We propose to study the impact of mispredictions on the cost minimization techniques.

To answer these question in chapter 2 we see the current techniques to deal with energy forecasting. In chapter 3 we see the prediction framework for energy usage forecast. Data Collection techniques and datasets used for prediction are in chapter 4. In chapter 5 we see the effect of weather attributes on energy usage forecast. Attribute subset selection method is covered in chapter 6. Energy cost minimization techniques are covered in chapter 7. In chapter 8 we study the effect of misprediction on the energy cost minimization algorithm. We introduce extensions to earlier energy minimization techniques in chapter9. In Chapter 10 we evaluate energy cost minimization technique with real dataset. We end our discussion with conclusion and future-work in chapter 11.

Chapter 2

Related Work

Electricity usage and price forecasting is not a new area. Several researchers have studied the problem of the modeling and predicting building energy usage consumption. Energy prediction model is based on several parameters that are estimated using existing data that typically include energy consumption and temperature measurements recorded in the past.

Prediction models proposed in literature are Regression models, Artificial Neural Network models (ANN) by Mandal et al. (2007), Time-Series model by Nogales et al. (2002), Wavelet Transform and ARIMA model by Contreras et al. (2003). Each of these modeling approaches has its advantages and disadvantages, and the choice of a model is often application dependent. Regression models are useful for predicting average consumption over longer periods such as days or months Yu et al. (2010). These models can be developed quickly as they require calculation of only a few parameters. Models based on Artificial Neural Networks (ANN) have also been effective for building energy predictions as demonstrated by Olofsson and Andersson (2001) to predict building energy use for both short and long term periods and for hourly energy use. Chen et al. (2004) have used Support Vector Machines for city level maximum load forecasting.

In our work, we used Support Vector Regression (SVR) for fine grained and coarse grained load forecasting in office buildings. SVR is resistant to outliers.

Chapter 3

Model Selection

We used technique called support vector machines (SVM) to estimate office buildings energy consumption. SVM, developed by Vapnik and his coworkers in 1995, has been widely applied in classification, forecasting and regression.

3.1 Support Vector Machines for Regression

Support vector machines (SVM) are based on the structural risk minimization (SRM) inductive principle, which seeks to minimize an upper bound of the generalization error consisting of the sum of the training error and a confidence level. This is different from commonly used empirical risk minimization (ERM) principle, which only minimizes the training error. Based on such induction principle, SVM usually achieves higher generalization performance than the traditional neural networks that implement the ERM principle in solving many machine learning problems. Another key characteristic of SVM is that training SVM is equivalent to solving a linearly constrained quadratic programming problem so that the solution of SVM is always unique and globally optimal, unlike other network's training which requires non-linear optimization with the danger of getting stuck into local minima Dong et al. (2005). In SVM, the solution to the problem is only dependent on a subset of training data points which are referred to as support vectors. Using only support vectors, the same solution can be obtained as using all the training data points. One disadvantage of SVM is that the training time scales somewhere between quadratic and cubic with respect to the number of training samples. So a large amount of computation time will be involved when SVM is applied for solving large-size problems Cao and Tay (2003). However, in this study, a small dataset is considered.

3.1.1 Theory of SVM for Regression

Given training data $(x_1, y_1), \ldots, (x_n, y_n)$ where x_i are input vectors and y_i are the associated output value of x_i.

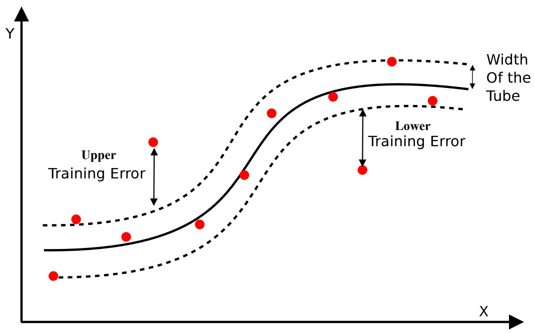


Figure 3.1: Support Vector Regression

SVM approximates the function using the following form.

$$f(x) = w.\phi(x) + b \tag{3.1}$$

where $\phi(x)$ represents the high-dimensional feature space which is nonlinearly mapped from the input space x. The support vector regression technique estimates w and b by solving an optimization problem:

$$\begin{aligned}
&\underset{w,b,\xi,\xi^*}{\text{minimize}} && \frac{1}{2}w^T w + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\
&\text{subject to} && y_i - (w^T\phi(x_i) + b) \leq \epsilon + \xi_i, \\
& && (w^T\phi(x_i) + b) - y_i \leq \epsilon + \xi_i^*, \\
& && \xi_i, \xi_i^* \geq 0, i = 1, \ldots, n.
\end{aligned}$$

where

$(x_1, y_1), \ldots, (x_n, y_n) = TrainingData.$ $\quad \xi_i = UpperTrainingError.$

$\xi_i^* = LowerTrainingError.$ $\quad C = CostOfError.$

$\epsilon = WidthOfTheTube.$ $\quad \phi = MappingFunction$

$W = WeightVector.$ $\quad b = Bias$

The first term $\frac{1}{2}w^T w$ is called the regularized term. Minimizing $\frac{1}{2}w^T w$ will make a function as flat as possible, thus playing role of controlling the function capacity. The second term $\sum_{i=1}^{n}(\xi_i + \xi_i^*)$ is the empirical error measured by the ϵ-insensitive loss function. This defines a ϵ tube 3.1 so that if the predicted value is within the tube the loss is zero, while if the predicted point is outside the tube, the loss is magnitude of the difference between the predicted value and the radius ϵ of the tube.

SVR avoids under-fitting and over-fitting the training data by minimizing the training error $\sum_{i=1}^{n}(\xi_i + \xi_i^*)$ as well as the regularization term $\frac{1}{2}w^T w$. For traditional least-square regression ϵ is always zero and data are not mapped into higher dimensional spaces. Hence SVR is a more general and flexible treatment on regression problems

3.2 LIBSVM Tool

For our experiments, we used a software LIBSVM by Chang and Lin (2011). LIBSVM is an open library for support vector machines. LIBSVM provides support library for classification and regression in variety of programming languages including C, python, java, MATLAB etc. For our experiments we have used LIBSVM support library in C languages.

3.3 Experimental Settings

We have used a variant of SVR called *epsilon-SVR* and *radial basis function* as a kernel type for our experiment.

- $KernelFunction = e^{(\gamma * ||u-v||^2)}$
- $\gamma = \frac{1}{NumberOfFeatures}$
- C=Cost Of Error=1
- $\epsilon = WidthOfTheTube = 0.1$

u and v are parameters of kernel function. One of these will be the support vector and the other will be the testing data point.

3.4 Accuracy Measure

We use MAPE (Mean Absolute Percentage Error) as a measure of accuracy. MAPE expresses accuracy as a percentage. MAPE is defined by the formula.

$$MAPE = 100\frac{\sum_{i=1}^{n}|\frac{A_i-P_i}{A_i}|}{n} \tag{3.2}$$

Where
$A_i = ActualValue$
$P_i = PredictedValue$
$n = NumberofDataPoints$

3.5 Summary

In this chapter we described SVR prediction framework. We introduced LIBSVM tool which implements SVR algorithm for prediction. Experimental settings used with LIBSM tool are described. MAPE is introduced as an accuracy measure.

Chapter 4

Data Collection

To build prediction model, we collect weather data, room occupancy data and energy usage data. This chapter provides the information about the data sources that we used for forecasting.

4.1 KRESIT Energy Usage Dataset

We use L&T 3 Phase CT Operated Trivector Meter to record the energy usage of KRESIT building. We metered the electricity consumption at every 15 minutes. The multiplication factor of this metering facility is 400. This meter is not provisioned for automatic meter reading. Hence we use *Tina-Time lapse* android application to capture the snapshot of the meter after interval of every 15 minutes. The file descriptor of the image is used to get the time and day at which the image was captured. This is also the time and day at which electricity consumption was recorded. We manually convert the meter reading snapshots into the electricity usage values.

Manual conversion of metered data adds human error in the actual energy consumption values. We use spike analysis technique to detect the human error. If any aperiodic sudden increase in the consumption of electricity is followed by a sudden decrease in the electricity consumption, then that event can be identified as a spike. Generating such spikes is very unlikely for electrical appliances, and hence can only be generated as a result of human error.

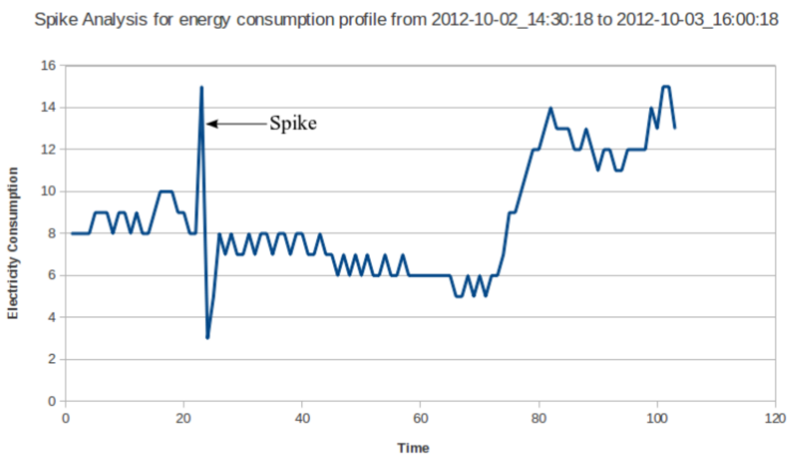


Figure 4.1: Spike Analysis

After detecting the spikes, we verified the conflicting consumption values against the picture snapshots. We made appropriate corrections in the dataset to make data error free.

4.2 Weather Dataset

We considered Temperature, Humidity and Dew-Point as attributes to model the weather condition. We obtain weather data from Weather Underground website [1]. This website provides a web form which takes the input as a date and returns a CSV file containing weather attributes for the given date ranges. The weather data is available at a variable granularity of 30 minutes to 1 hour. Some time weather information is not available for as long as 3 hours. For our fine-grained model, we interpolate or extrapolate the missing values as necessary to 15-min durations.

4.3 Occupancy Dataset

Occupancy information in the academic and corporate settings can be inferred from the timetable, room scheduler and event calendar. We use publicly available Room Scheduler of CSE department-IIT Bombay [2] to infer the occupancy information. Room scheduler gives information about the time interval during which certain hall is occupied and maximum capacity of the hall. Online room scheduler gives occupancy information of following halls.

Conference Room	Meeting Room	FC Kohli Auditorium
Meeting Room SIA221	Lecture Hall SIB201	Medium Classroom SIC201
Medium Classroom SIC205	Meeting Room SIA321	Medium Classroom SIC301
Small Classroom SIC304	Small Classroom SIC305	

Table 4.1: KRESIT Occupancy Dataset

4.4 Summary

In this chapter we have introduced a data collection process and the datasets used to perform the experiments. We have describe the occupancy dataset and weather dataset in this chapter.

[1] http://www.wunderground.com
[2] http://www.cse.iitb.ac.in/room-scheduler/

Chapter 5

Effect of Weather and Occupancy on Energy Usage Forecast

Energy demand for a building depends upon the external weather conditions and occupancy. In this chapter we study the effect of weather and occupancy on the energy usage forecast.

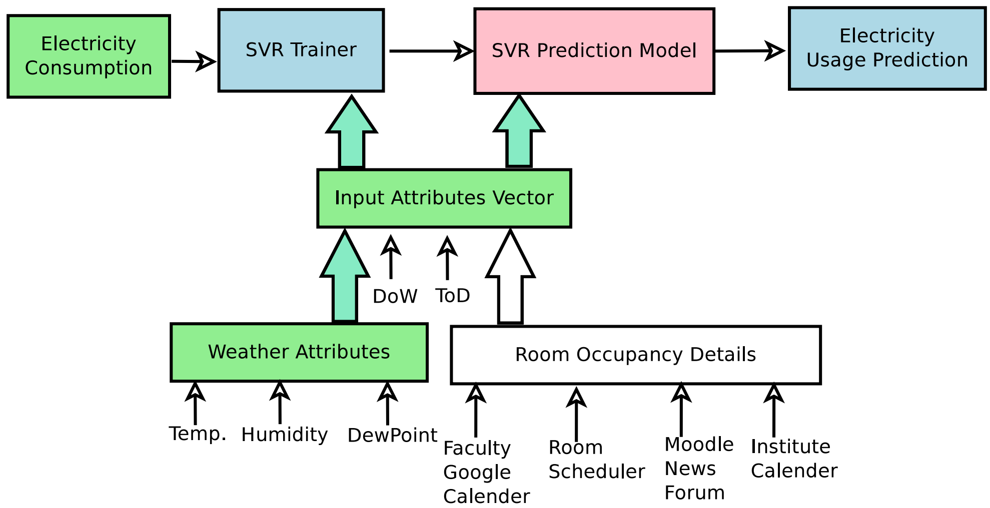


Figure 5.1: Prediction Framework

Figure 5.1 shows the block diagram of the prediction framework using SVR model. We consider a input feature vector. It comprises of weather attributes and occupancy attributes. We use these attributes to train a SVR model. Trained model is used to predict per 15 minute energy consumption.

5.1 Role Of Weather

The influence of weather on energy consumption, particularly electricity demand, has been widely reported in the past. Though energy consumption of certain constant loads, such as lighting, fan and plug loads do not change with weather variables. But there is a big family of electrical appliances whose usage is heavily depends on the weather condition. On some dry-hot day increase in the electricity consumption can be attributed to the use of air-conditioners. While on a freezing-cold day many people uses a room heater to bring temperature to the comfort zone, hence energy consumed by room-heaters in the winter season can be attributed to decrease in the temperature. Many coastline residents use a dehumidifier to control the level of humidity. Hence electricity consumption by the dehumidifier devices also depends on weather condition.

We study the effect of three weather attributes on the energy usage forecast namely Temperature, Dew-point, Moisture. We calculate the correlation between energy consumption and weather attributes.
"Correlation is a measure of the strength and direction of the linear relationship between two variables."

$$correlation(x, y) = \frac{covariance(x, y)}{StandardDeviation(x).StandardDeviation(y)} \tag{5.1}$$

We use following heuristic to decide which weather attributes should be part of prediction model.

5.1.1 Heuristic

Weather attributes having very high positive or negative correlation with energy consumption, will assist in forecast.

We validate this heuristic in subsequent sections.

5.1.2 Effect of Temperature on Energy Usage Forecast

Previous studies have shown that temperature is usually the most significant weather variable influencing electricity consumption Chen et al. (2004). But as our problem focuses on forecasting energy usage of academic and office buildings where most of the activities are periodic. We answer the following question in this section.
How much temperature affects the electricity consumption in office buildings ?

We have collected per 15 minute energy usage data of KRESIT building, from 25-August-2012

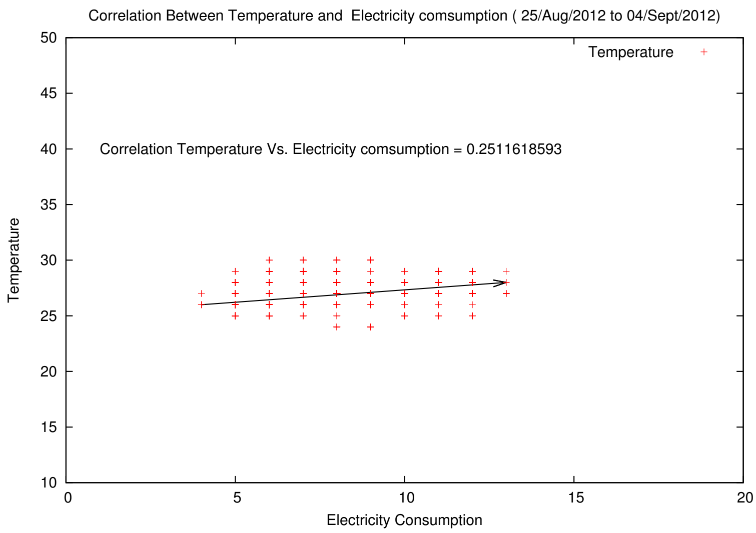


Figure 5.2: Correlation between Temperature and Electricity Consumption

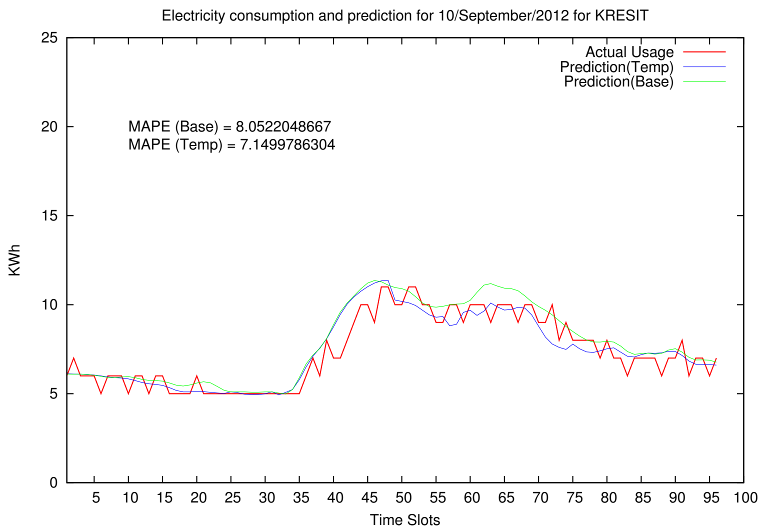


Figure 5.3: Effect of Temperature on Prediction Accuracy

to 04-September-2012 and per 15 minute temperature readings for the same days for Mumbai city. We study the correlation between these two attributes, which turns out to be *0.2511618593*. Figure 5.2 shows the plot of temperature versus electricity consumption.

We train SVR Model for the dataset with the input attribute as *time of day*, *day of a week* and associated output label is *energy usage*. *Time of a day* and *day of a Week* is very critical for forecasting. The prediction which are based on only those two attributes, we call them *base prediction*. We used above trained model to forecast the energy consumption on 10-September-2012.

Figure 5.3 shows the plot of actual energy consumption versus predicted energy consumption. Experimental results shows MAPE for *Base prediction* framework is *8.0522048667*.

We consider *temperature* as an attribute in our prediction framework along with *Base Attributes*. Experimental results show MAPE with temperature is *7.1499786304*, which is lower than *base prediction*. Results shows that adding temperature improves the prediction accuracy by 0.902226236.

5.1.3 Effect of Dew-Point on Energy Usage Forecast

"The dew point is the temperature below which the water vapour in a volume of humid air at a constant barometric pressure will condense into liquid water".
We have collected per 15 minute energy usage data of KRESIT building, from 25-August-2012 to 04-September-2012 and per 15 minute Dew-Point temperature readings for the same days for Mumbai city. We study the correlation between these two attributes, which turns out to be *0.141472147*. Figure 5.4 shows the plot of Dew-Point versus electricity consumption.

We consider *Dew-Point* as an attribute in our prediction framework along with *base attributes*. Figure 5.5 shows the plot of actual energy consumption versus predicted energy consumption. Experimental results show MAPE with temperature and Dew-Point is *7.4296838098*, which is lower than *Base Prediction*.

Results show that adding temperature and Dew-Point improves the prediction accuracy over *Base Prediction* by 0.622521057. Result shows, considering only temperature for prediction scores more than considering temperature along with Dew-Point.

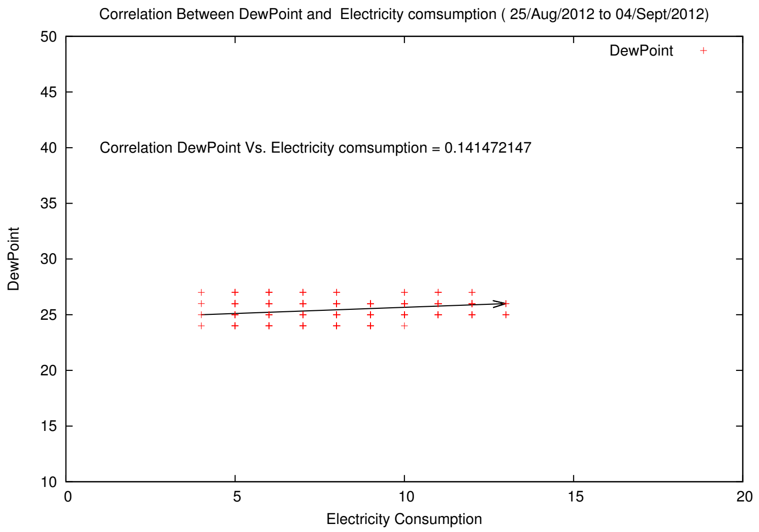


Figure 5.4: Correlation between Dew-Point and Electricity Consumption

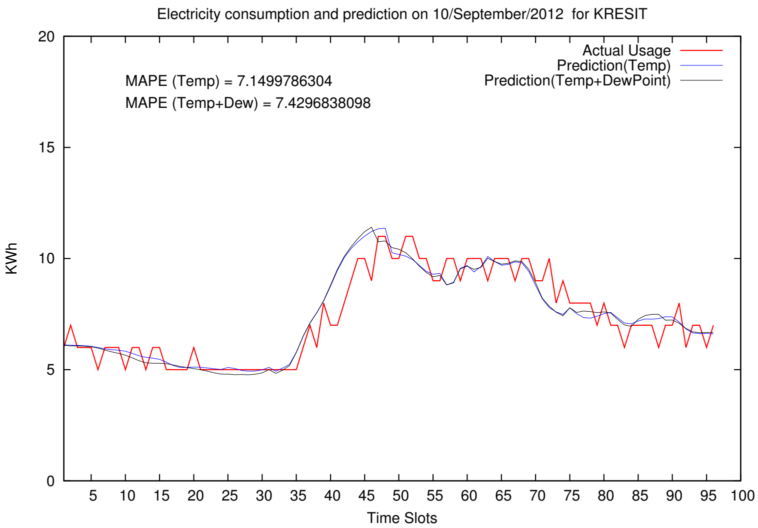


Figure 5.5: Effect of Dew-Point on Prediction Accuracy

5.1.4 Effect of Humidity on Energy Usage Forecast

"Humidity is a measure of the amount of water vapour in the air."
We have collected per 15 minute energy usage data of KRESIT building, from 25-August-2012 to 04-September-2012 and per 15 minute humidity readings for the same day. We study the correlation between these two attributes, which turns out to be *-0.1498616324*. Humidity and electricity consumption are negatively correlated, which means as humidity increases electricity consumption decreases. Figure 5.6 shows the plot of Humidity versus electricity consumption.

We consider *Humidity* as an attribute in our prediction framework along with *base attributes*. Figure 5.7 shows the plot of actual energy consumption versus predicted energy consumption.

Experimental results show MAPE for temperature and Humidity is *9.0437927797*, which is higher than *Base Prediction*.

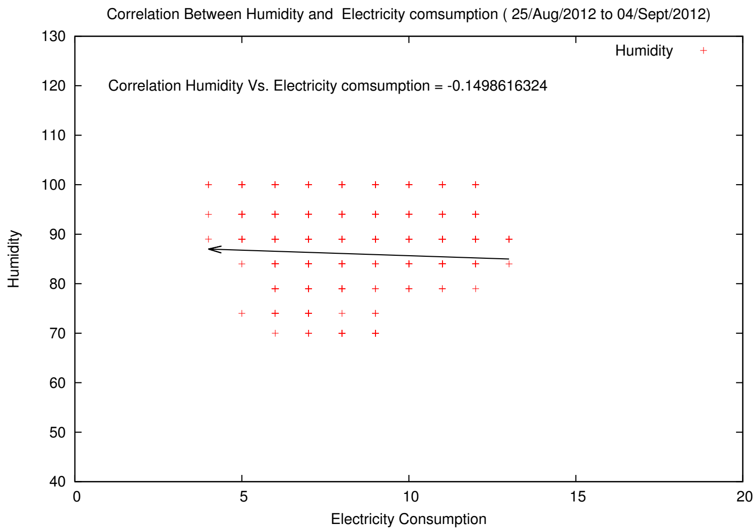


Figure 5.6: Correlation between Humidity and Electricity Consumption

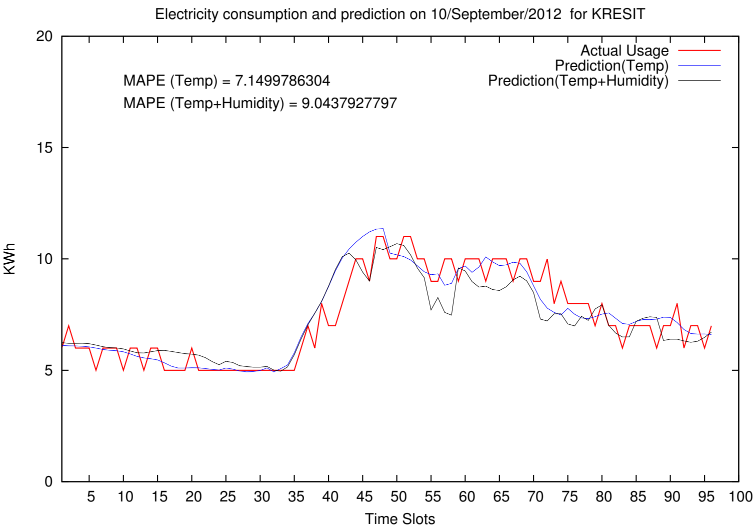


Figure 5.7: Effect of Humidity on Prediction Accuracy

Results shows that adding temperature and Humidity degrades the prediction accuracy over *Base Prediction* by 0.991587913. Result shows, considering only temperature for prediction score more than considering temperature along with Humidity.

5.1.5 Experimental Results

Table 5.1 shows performance comparison of weather attributes on the basis of MAPE.

ToD	DoW	Temp.	Dew-Point	Humidity	MAPE
✓					7.78
✓	✓				8.05
✓		✓			8.61
✓	✓	✓			7.15
✓	✓	✓	✓		7.43
✓	✓	✓		✓	9.04
✓	✓	✓	✓	✓	8.94

Table 5.1: Prediction Error for weather attributes

Experimental result shows that considering *Day of a Week* and *Temperature* independently with *Time of a Day* degrades results in comparison with considering only *Time of a Day* as an attribute. But if we consider both *Day of a Week* and *Temperature* simultaneously along with *Time of a Day*, then prediction accuracy increases in comparison with only *Time of a Day*.

Attribute set *Day of a Week*, *Temperature* and *Time of a Day* gives best accuracy, while *Day of a Week*, *Temperature*, *Time of a Day* and *Humidity* gives degraded accuracy.

5.2 Effect of Occupancy on Electricity Forecast

Electricity usage for a building is depends upon the occupancy. Energy requirement of a big auditorium is different from energy requirement of a classroom. In this chapter we study the effect of occupancy on the energy usage forecast accuracy.

5.2.1 Role of occupancy

Occupancy gives information about how many people occupied certain hall for how much time? We have collected occupancy details of KRESIT building using publicly available room scheduler. We have considered each hall in the KRESIT building as an attribute for the SVR based prediction framework. Figure 5.8 shows the plot of actual energy consumption versus predicted energy consumption with occupancy as one of the attribute.

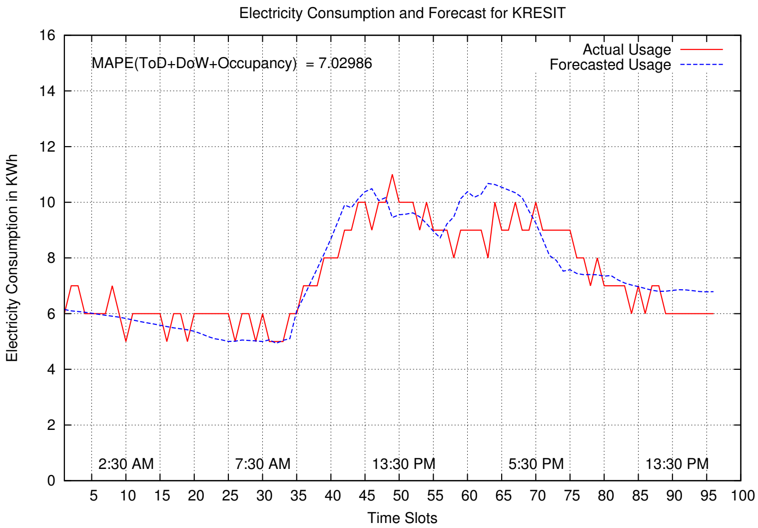


Figure 5.8: Effect of Occupancy on Prediction Accuracy

Experimental results show MAPE with occupancy is *7.02986*, which is less than *base prediction*. Also considering the occupancy in the prediction framework gives best forecasting accuracy.

5.3 Summary

In this chapter we studied the effect of weather parameters such as temperature, humidity and dew-points on the energy usage prediction. We also looked into effect of occupancy on energy usage forecast.

Chapter 6

Attribute Subset Selection

The experiments conducted so far are on the dataset, with all the attributes for weather and occupancy. The 25 attributes which we are using to train forecasting model may have some redundant attributes. Redundant attributes provides no more information than currently selected attributes. Irrelevant attributes do not provide any useful information.

The attribute set works best for certain day, may not work well for other days. For predicting the energy usage we have to choose the best subset of attributes from 25 available attributes. The problem of selecting a subset of relevant features for building robust learning model is called *Feature Selection*. Feature selection removes most irrelevant and redundant features from the data and helps improve the performance of learning models.

6.1 Subset Selection

In statistics, the most popular form of feature selection is stepwise regression. It is a greedy algorithm that adds the best feature or deletes the worst feature at each round. We have limited dataset, hence we did an exhaustive search for the best possible attribute set for a given testing data.

We have considered data for each day in a week as an independent dataset and done an exhaustive search for best subset of attributes in each dataset. We did this experiment for seven days from 'Friday 07/September/2012' to 'Thursday 13/September/2012' and select seven best possible subsets of attribute for each day of a week.
Figure 6.1 and Figure 6.2 shows the best possible prediction for Friday and Saturday with attribute set 1 and attribute set 2 respectively.

Attribute Set1	Time Slot	Day Of a Week	Medium Classroom SIC301
	F.C. Kohli Auditorium	Medium Classroom SIC 201	
Attribute Set2	Time Slot	Day Of a Week	Small Classroom SIC304
	Conference room	Meeting Room SIA 113	F.C. Kohli Auditorium
	Meeting Room SIA221	Medium Classroom SIC201	Medium Classroom SIC205
	Meeting Room SIA321	Medium Classroom SIC301	

Table 6.1: Attribute set for Friday and Saturday

Figure 6.3 and Figure 6.4 shows the best possible prediction for Sunday and Monday with attribute set 3 and attribute set 4 respectively.

Figure 6.5 and Figure 6.6 shows the best possible prediction for Tuesday and Wednesday with attribute set 5 and attribute set 6 respectively.

Figure 6.7 shows the best possible prediction for Thursday with attribute set 7.

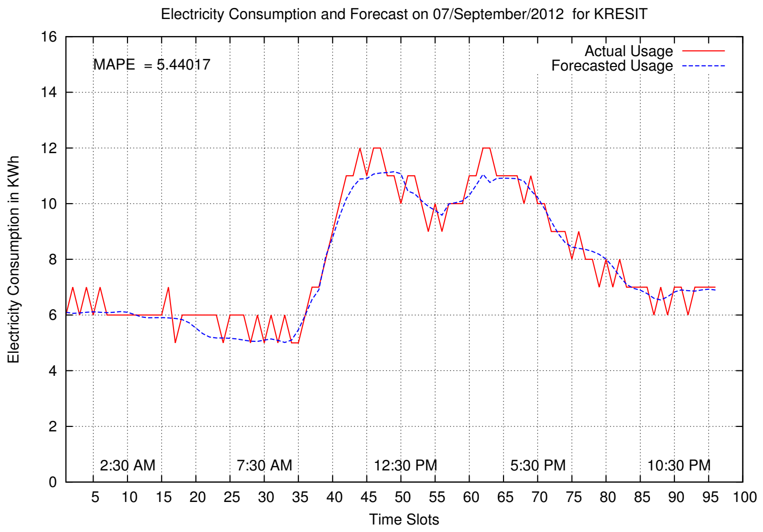


Figure 6.1: Electricity Consumption and Forecast on Friday 07/September/2012

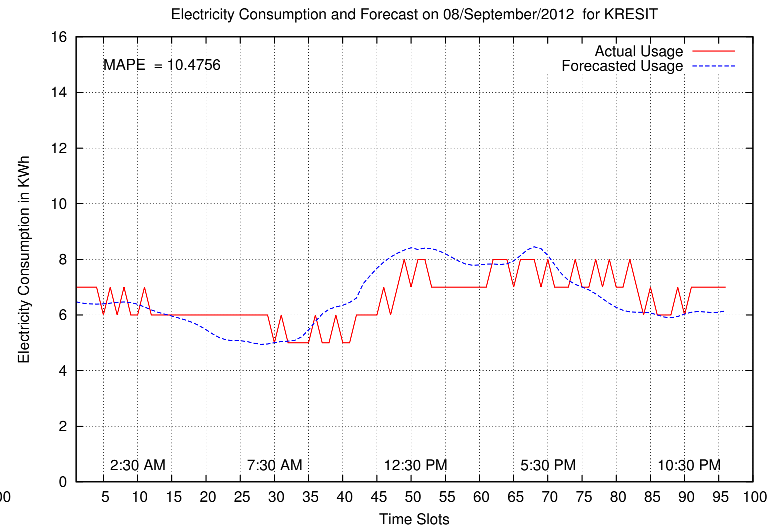


Figure 6.2: Electricity Consumption and Forecast on Saturday 08/September/2012

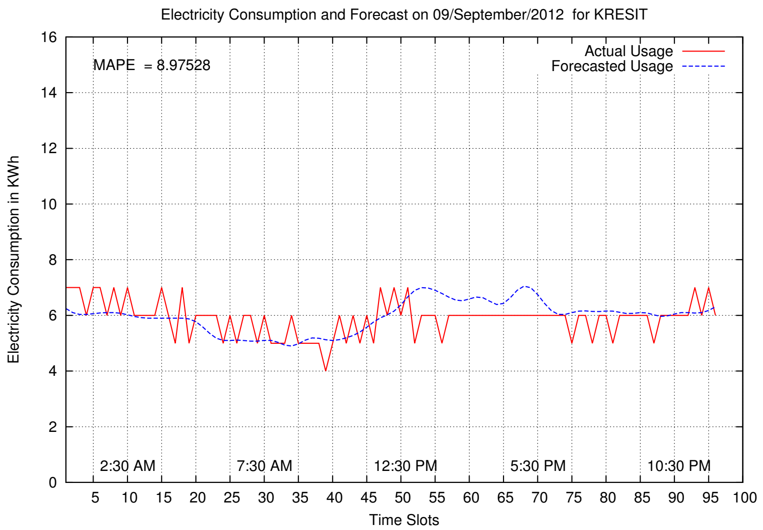


Figure 6.3: Electricity Consumption and Forecast on Sunday 09/September/2012

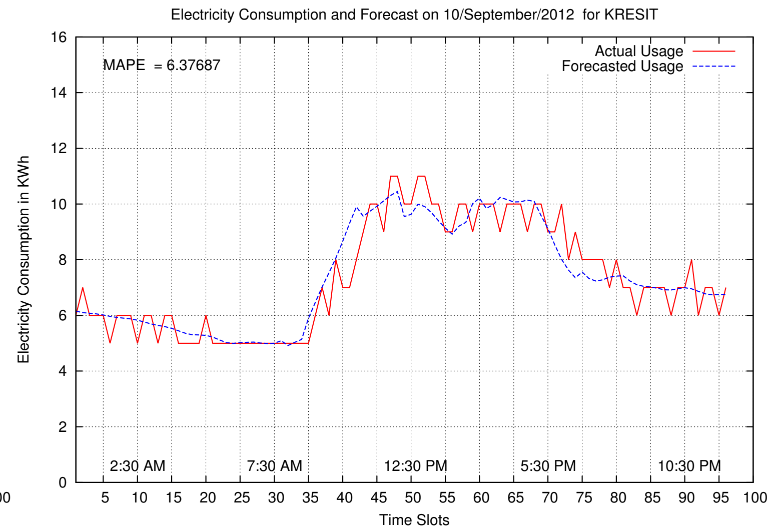


Figure 6.4: Electricity Consumption and Forecast on Monday 10/September/2012

Attribute Set3	Time Slot	Day Of a Week	Medium Classroom SIC205
	Conference room	Meeting Room SIA 113	Meeting Room SIA221
Attribute Set4	Time Slot	Day Of a Week	Small Classroom SIC304
	Temperature	Meeting Room SIA 113	F.C. Kohli Auditorium
	Meeting Room SIA221	Medium Classroom SIC201	Meeting Room SIA321

Table 6.2: Attribute set for Sunday and Monday

Attribute Set5	Time Slot	Day Of a Week	DewPoint
	Conference room	Meeting Room SIA 113	Medium Classroom SIC205
Attribute Set6	Time Slot	Day Of a Week	Conference room

Table 6.3: Attribute set for Tuesday and Wednesday

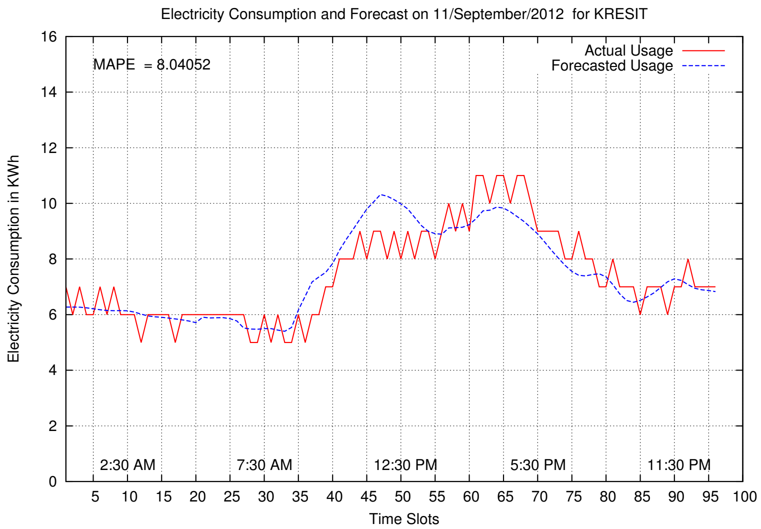


Figure 6.5: Electricity Consumption and Forecast on Tuesday 11/September/2012

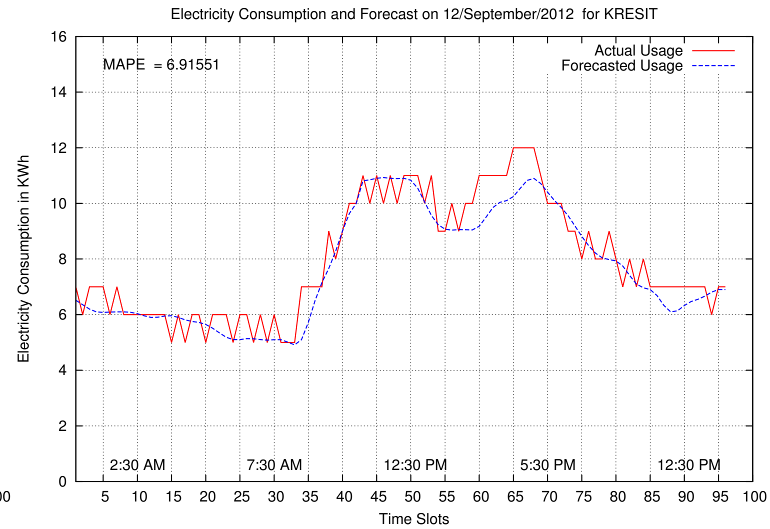


Figure 6.6: Electricity Consumption and Forecast on Wednesday 12/September/2012

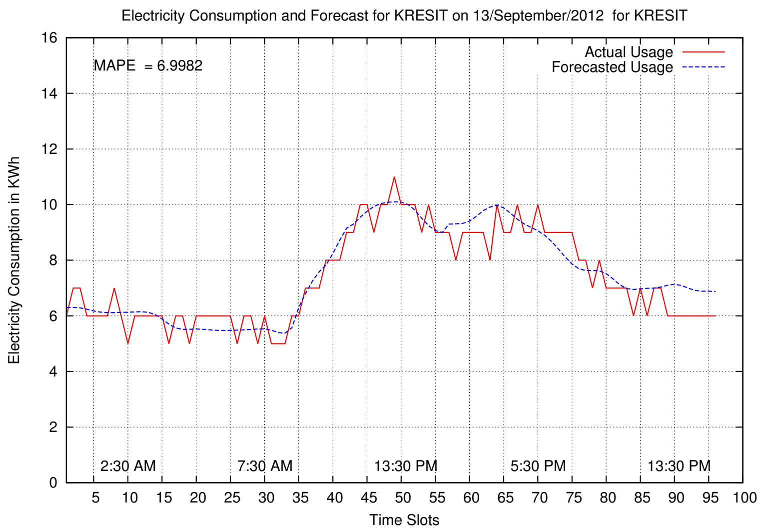


Figure 6.7: Electricity Consumption and Forecast on Thursday 13/September/2012

Attribute Set7	Time Slot	Day Of a Week	Lecture Hall SIB201
	Conference room	Meeting Room SIA221	Medium Classroom SIC201
	Small Classroom SIC304	Meeting Room SIA321	

Table 6.4: Attribute set for Thursday

6.2 Experimental Results

Table 6.5 shows the experimental results with MAPE for different attribute settings, and for seven days of a week.

Attribute	Friday 07/09/2012	Saturday 08/09/2012	Sunday 09/09/2012	Monday 10/09/2012	Tuesday 11/09/2012	Wendnesday 12/09/2012	Thursday 13/09/2012	Average
Set1	5.44017	13.6084	14.7368	7.76467	9.74204	7.74964	10.925	9.99524
Set2	6.35938	10.4756	9.32921	8.89207	8.77289	8.79183	7.43932	8.58004
Set3	6.16691	10.4805	8.97528	8.88956	8.54714	9.24163	7.23855	8.50565
Set4	6.58728	12.9209	19.0713	6.37687	10.638	9.27213	9.54465	10.63016
Set5	6.4263	11.3768	10.6523	9.4614	8.04052	10.0994	7.57595	9.09038
Set6	6.12934	15.787	18.8766	7.8033	9.59482	6.91551	10.8501	10.85095
Set7	6.34025	11.5726	9.76701	9.13732	8.49868	9.23602	6.9982	8.79286
Average	6.20709	12.3174	13.05835	8.33217	9.11915	8.75802	8.65311	×

Table 6.5: MAPE of Forecasting for Different Attributes Sets

Each Column of Table 6.5 shows MAPE for a certain day in a week for attribute sets which are shown in the rows. Attribute set 1 is chosen as best possible attribute subset for Friday using exhaustive search in the given attribute space. Hence prediction done for Friday using attribute set 1, shows least MAPE in comparison with other attribute sets.

Similarly, Attribute set 2 is chosen as a best possible attribute subset for Saturday using exhaustive search in the given attribute space. Hence prediction done for Saturday using attribute set 2, shows least MAPE in comparison with other attribute sets, and so on.

The last row in the Table 6.5 shows MAPE for each day in a week averaged over all the attribute sets. While the last column in the Table 6.5 shows MAPE for each attribute set averaged over all days in a week.

6.3 Summary

In this chapter we described attribute subset selection method to eliminate redundant and irrelevant attribute. Attribute subset selection improved the performance of prediction.

Chapter 7

Energy Cost Minimization Using Battery

In this chapter, we define the Non-Linear pricing scheme. We then study and analyze the SmartStore algorithm.

7.1 Pricing Policy

To reduce peak usage, utilities discourage peak demands by charging them with a higher price. Utilities also incentivize low power demands by charging them with lower price. Utilities use various electricity pricing schemes to implement such a policy. Peak pricing, real-time pricing and demand-based non-linear pricing are some examples of electricity pricing schemes.

For further discussions, we have used a Demand-Based Non-Linear pricing scheme. If demands of a customer is given as a sequence $(d_1, d_2, ..., d_n)$, then the total energy cost E is of the form,

$$E = \sum_{t=1}^{n} {d_t}^x \tag{7.1}$$

where, x is a tuning parameter and

$$x > 1$$

In demand-based non-linear pricing scheme, a significant fraction of energy cost is contributed by peak demand over a given billing interval. Electricity bill minimization in demand-based non-linear pricing therefore involves minimizing the maximum demand over a given billing interval.

Under the Non-Linear pricing scheme, energy cost can be significantly reduced by using batteries. Whenever demand and hence the price of electricity is low, purchase more energy from the grid and store in the battery for future use. When demand for energy reaches peak, use the stored energy from battery to satisfy peak demand. This will lower the peak demand and in turn reduce the energy cost.

7.2 SmartStore Algorithm

7.2.1 Introduction

The SmartStore algorithm by Rushiraj Chavan (2013) is an extension of the offline threshold-based algorithm (Bar-Noy et al., 2009). SmartStore uses batteries to minimize the maximum peak. For the given energy consumption forecast, it gives the minimum peak and significantly reduce the cost of the electricity. In this discussion we focus only on unbounded battery setting. With unbounded battery setting, the battery starts empty. We assume that there is no restriction on charge rate and discharge rate of the battery. In a slot, battery can either be charged or discharged, but both charging and discharging cannot happen in the same slot.

7.2.2 Algorithm

Following notations are used in the algorithm. We denote demand in slot t as d_t. Interval of billing is $[1, n]$. Number of slots in the billing interval is denoted by n. Recharge and discharge of battery in slot t is denoted R_t and D_t respectively. We denote power consumed from the grid (request to grid) in slot t as g_t. We denote battery maximum energy level as B_{max}. Battery minimum energy level is denoted by B_{min}, usually zero. Battery energy level at the start of slot t is denoted by B_t.

Let $\mu(j,k) = \frac{1}{k-j+1}\sum_{t=j}^{k} d_t$ be the mean forecasted-demand of the region $[j,k]$. The first maximum mean among all the prefix regions up to n is given as $\hat{\mu}(1,n) = max_{1\leq j\leq n}\mu(1,j)$. Assume that $\hat{\mu}(1,n)$ occurs at slot t. For the subinterval $[1,t]$, use $T_1 = \hat{\mu}(1,n)$ as threshold.

Now let $\hat{\mu}(t+1,n) = max_{t+1\leq j\leq n}\mu(t+1,j)$ be the maximum mean among all the prefix regions in the subinterval $[t+1,n]$ and assume that it occurs at slot k. For the subinterval $[t+1,k]$, use $T_2 = \hat{\mu}(t+1,n)$ as threshold.

Repeat the process of finding maximum mean and assigning it to the respective subinterval until the n^{th} slot is reached.

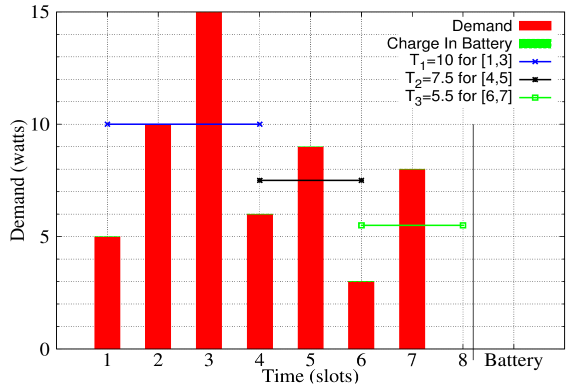
At the end, we get I number of subintervals with a_i as the starting slot, b_i as the ending slot and Ti as the threshold of the i^{th} subinterval.

After calculating the subintervals and the corresponding thresholds, algorithm execute the following steps.

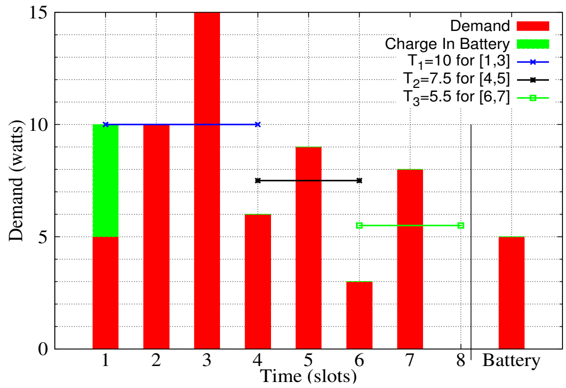
for $i = 1..I$ **do**
    for $t = a_i..bi$ **do**
        if $(d_t > T_i)$ **then**
            Discharge $D_t = \min(d_t - T_i, B_t - B_{min})$ from battery and request T_i from grid
            $B_{t+1} = B_t - D_t$
        else
            Charge $R_t = \min(B_{max} - B_t, T_i - d_t)$ to battery and request T_i from grid
            $B_{t+1} = B_t + R_t$
        end if
    end for
end for

7.2.3 Analysis

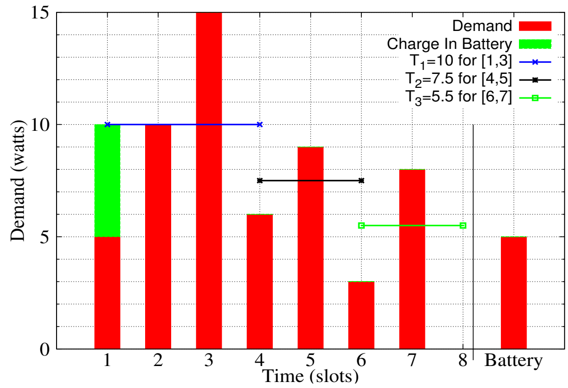
Battery energy level at the end of the slot at which the maximum mean occurs will become zero. This is true for unbounded battery setting where battery starts empty. In figure 7.1,



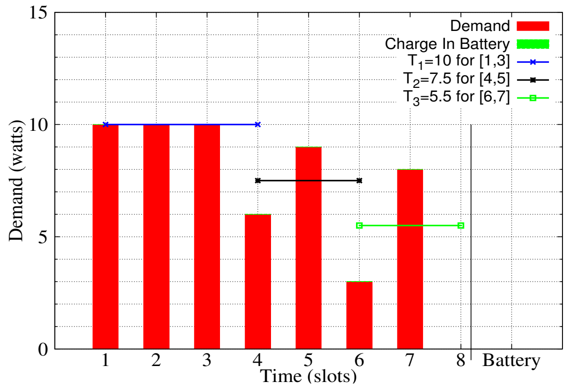
(a) Demands and Thresholds for billing period of 7 slots



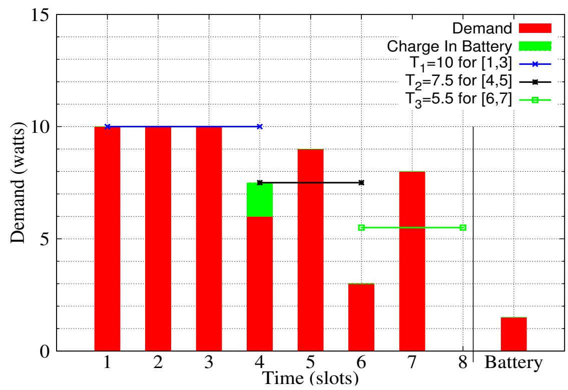
(b) Slot 1 : $R_1 = 5$



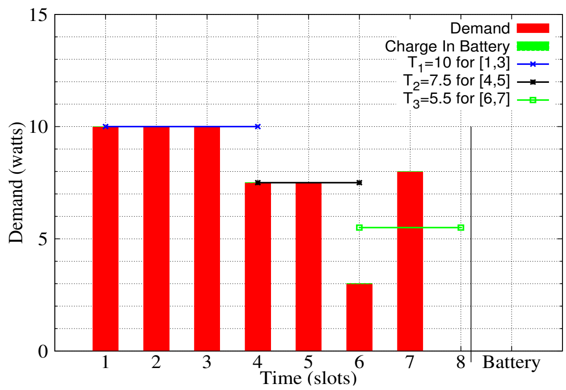
(c) Slot 2 : $R_2 = D_2 = 0$



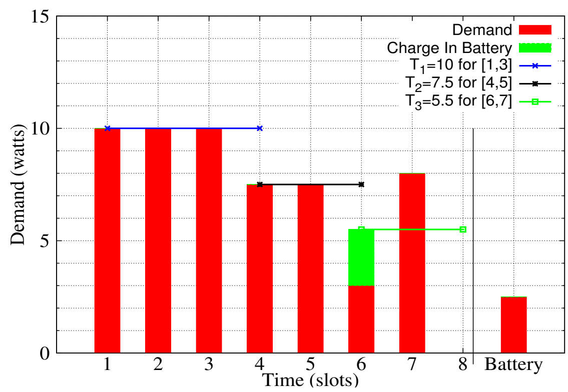
(d) Slot 3 : $D_3 = 5$



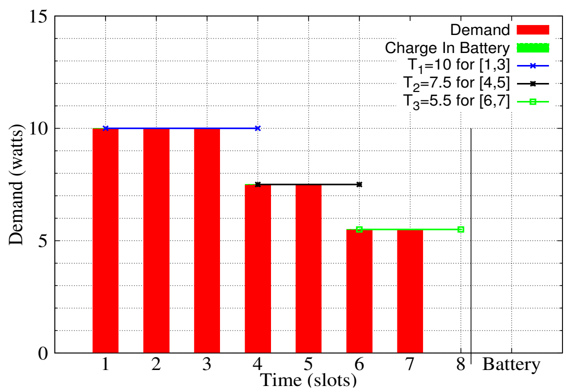
(e) Slot 4 : $R_4 = 1.5$



(f) Slot 5 : $D_5 = 1.5$



(g) Slot 6 : $R_6 = 2.5$



(h) Slot 7 : $D_7 = 2.5$

Figure 7.1: A SmartStore Algorithm

from start to the end of slot 3 battery energy will be completely used to minimize the peak and battery will become empty.

SmartStore divides the billing interval in a set of subintervals in such a way that at the end of each subinterval the battery will become empty. This ensures that the battery will become empty at the end of billing interval and there is no unnecessary accumulation of energy in battery. In addition to this, SmartStore uses offline threshold-based algorithm for peak shaving within a subinterval which minimizes the maximum demand within the subinterval. The maximum mean in a subinterval will always occur at the end slot of the subinterval and hence battery becomes empty at the end of the subinterval.

The time complexity of SmartStore algorithm is $O(n^3)$. SmartStore is an offline algorithm and value of n is usually smaller. Hence, complexity of $O(n^3)$ is acceptable.

Figure 7.1 shows the working of SmartStore algorithm. Figure 7.1a shows the demands and thresholds for the corresponding subintervals. In SmartStore, thresholds are decreasing step function as against the offline threshold-based algorithm where a constant threshold is used for all the slots. SmartStore algorithm works exactly same as the offline threshold-based algorithm up to the first subinterval i.e., $[1, 3]$ because the thresholds are same. In the second subinterval $[4, 5]$ at slot 4, threshold decreases and therefore the amount of recharge to the battery in slot 4 decreases as shown in figure 7.1e. As SmartStore uses the offline threshold based algorithm within a subinterval, the next slot 5 (end slot of subinterval 2) has a discharge equal to the recharge in slot 4 i.e., in a subinterval, whatever charge is put in the battery will be completely used before the starting slot of next subinterval as shown in figure 7.1f. This ensures that no unnecessary accumulation of energy is done i.e., take the exact amount of energy needed to minimize the maximum demand within a subinterval. This also ensures that the battery will not be recharged in peak slots within a subinterval instead these peaks are further minimized as shown in figure 7.1e and 7.1f. Similarly, battery operations in the last subinterval $[6, 7]$ are shown in figure 7.1g and 7.1h.

Heterogeneity in demands is a big problem for utilities. Utilities spend a lot of costly resources such as diesel for the diesel generators to satisfy the peak demands. Using SmartStore, peaks within a subinterval are lowered as well as demands in a subinterval are smoothed which reduces the heterogeneity in demands and in turn saves the costly resources of utilities.

7.3 Summary

In this chapter we studied the SmartStore algorithm to reduce the energy cost with demand based non-linear pricing scheme.

Chapter 8

SmartStore With Mispredicted Energy Demands

Accurate forecasting of the electricity demand is very critical for the success of the offline SmartStore algorithm. Actual electricity usage deviates from predicted electricity usage as no forecasting technique can be absolutely accurate. In this Chapter we will look into the effect of mispredicted energy demands on the performance of SmartStore algorithm.

8.1 Effect Of Mispredicted Energy Demands On SmartStore

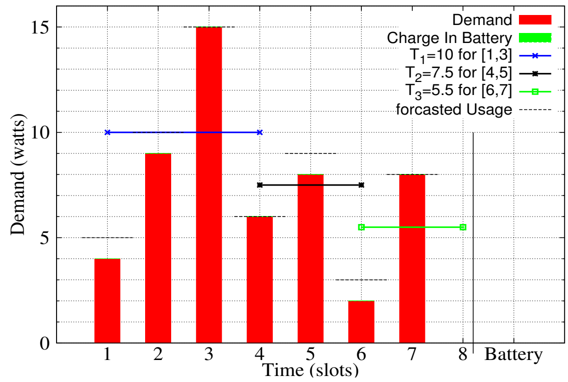
We will understand the effect of mispredictions on SmartStore using a following example.

- Example
  - Number of Slots =7
  - Forecasted Demands = 5,10,15,6,9,3,8
  - Actual Demands = 4,9,15,6,8,2,8
  - Battery Settings =Unbounded

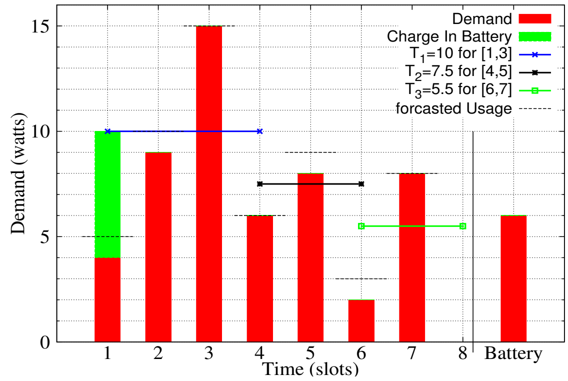
Figure 8.1 shows the working of the SmartStore algorithm with above toy example. Figure 8.1a shows the actual demands, forecasted usage/demands and thresholds calculated using the forecasted demands for the corresponding subintervals. In figure 8.1c the first subinterval i.e., $[1, 3]$ shows, at slot 1, and slot 2 actual demand is one less than the forecasted usage. As SmartStore assumes accurate forecast hence, this mispredictions went unnoticed. This can be seen in figure 8.1d, as at the end of subinterval $[1, 3]$, the charge left in the battery is 2 Watts. Ideally the amount of charge left in the battery at the end of any interval should be zero.

Again, in the second subinterval $[4, 5]$ at slot 5, actual demand is one less than the forecasted usage. Mispredictions causes unnecessary accumulation of extra charge in the battery or drain out battery quickly and leave charge for the planned usage by the subsequent slots. Charge accumulated in battery because of misprediction in subinterval $[4, 5]$ is shown in figure 8.1f. Similarly 8.1g shows that, extra charge stored in battery during slot 6 because of misprediction, also left unused by SmartStore algorithm. Figure 8.1h shows the state of the battery at the end of last slot.

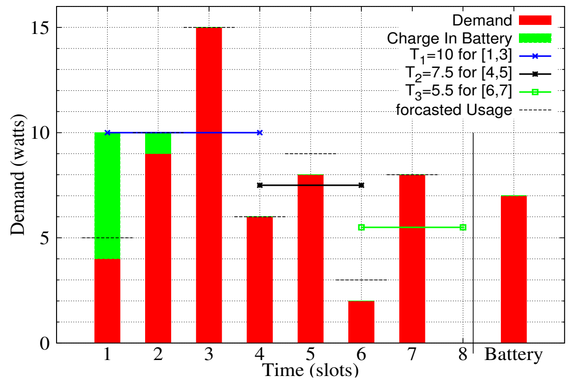
Accumulation of charge in the battery is the result of an error in the precise forecasting of future electricity demand and inability of SmartStore algorithm to use extra charge available in the battery instead of taking power from the grid.



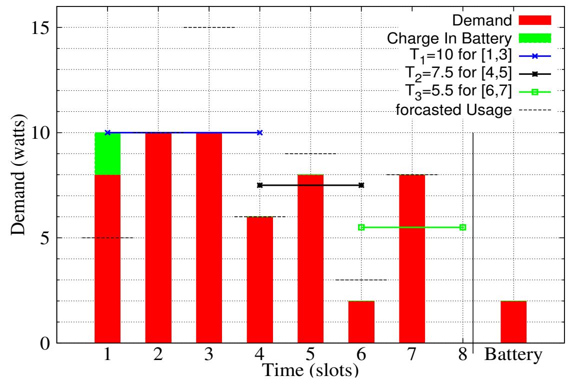
(a) Actual Demands and Precomputed Thresholds using predicted demand for billing period of 7 slots



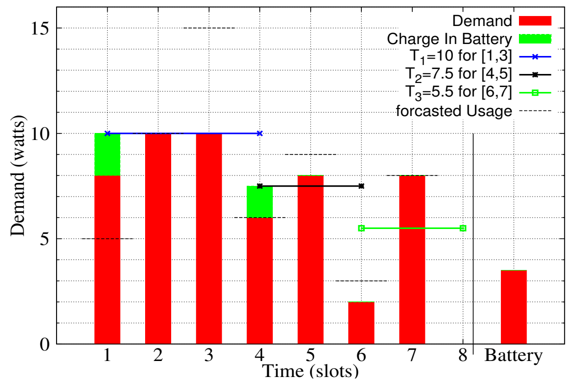
(b) Slot 1 : $R_1 = 6$



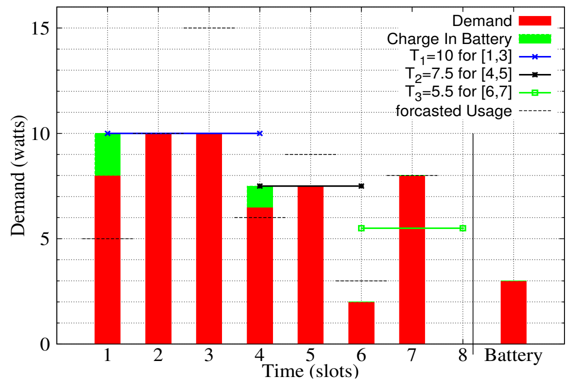
(c) Slot 2 : $R_2 = 1$



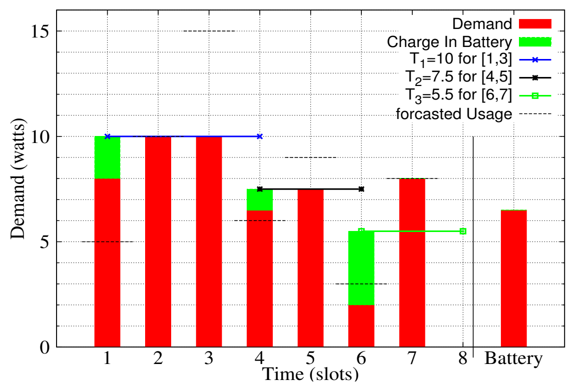
(d) Slot 3 : $D_3 = 5$



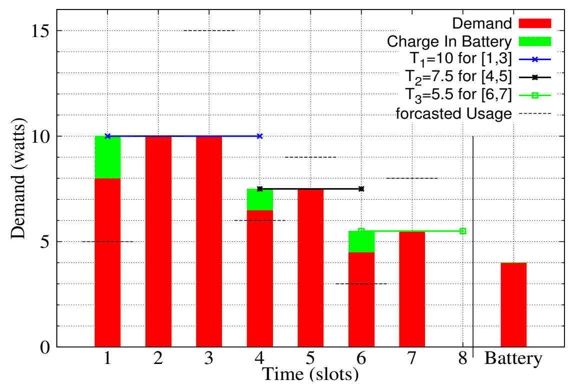
(e) Slot 4 : $R_4 = 1.5$



(f) Slot 5 : $D_5 = 0.5$



(g) Slot 6 : $R_6 = 3.5$



(h) Slot 7 : $D_7 = 2.5$, Charge left in the battery is 4 Watts

Figure 8.1: A SmartStore Algorithm, with inaccurate forecast

8.2 SmartStore With Different Error Values and Error Locations

In this section we will study the behavior of SmartStore with different error values and error location.

We have used $MeanAbsolutePercentageError(MAPE)$ as an accuracy measure to check the accuracy of forecasted electricity demand. MAPE calculated using forecasted data and actual data answers the question 'How much is the error?'. MAPE tells us the quantity of error, but it does not tell about 'At which location error occured?' The demand-based nonlinear pricing model defined in 7.1, penalize peak demands more severely than non-peak demands. One unit electricity can cost differently, during peak hours and and non-peak hours. Hence it's not only important to know 'How much is the error?', but it is also important to know 'Where the error occurred?'.

We have prepared a synthetic dataset to check the performance of SmartStore algorithm. Our dataset contains actual and forecasted data with 2%, 3%, 4% and 5% MAPE. This variation in the dataset will answer the 'How Much?' part of the question. To test the performance of SmartStore with the same amount of error, but with different error locations our datasets have following variants for each MAPE value,

- Overestimation at peak demand region.
- Overestimation at non-peak demand region.
- Uniform overestimation.
- Uniform misprediction (both overestimation and underestimation).
- Uniform underestimation.
- Underestimation at peak region.
- Underestimation at non-Peak region.

Following settings are used while evaluating the performance of SmartStore on synthetic dataset.

- Number of slots =96
- Battery settings =Unbounded
- $EnergyPrice = PowerfromGrid^{1.4}$

We have compared the performance of the SmartStore with different MAPE's to the ideal case of SmartStore where MAPE=0 i.e., predicted demands are 100% accurate. Figure 8.2 depicts such an ideal behavior of SmartStore where actual demand follows the forecasted demand.

8.2.1 Precise Forecast

In figure 8.2 curve 'Power in-to/out-off Battery' shows the amount of power that goes into the battery or out of the battery at any slot. Curve 'cumulative Energy In Battery' is drawn against $X1 - Y2$ axes and shows the total energy available in a battery. According to the SmartStore algorithm power taken from grid in any slot is equal to the threshold calculated for subinterval in which the that slot lies. The power taken from the grid and hence the threshold, is shown by curve 'power From Grid'.

In figure 8.2, first maximum mean occurs at Slot 76. Power taken from grid in subinterval $[1, 76]$ is greater than actual demand till slot 38, hence excess power will be stored in the battery. From slot 38 to slot 76, actual demand is greater than the threshold, hence excess power accumulated in the battery into slot 38 is used to satisfy the demand. The amount of power accumulated in the battery during this subinterval $[1, 76]$ is used in the same subinterval. This ideal synthetic example shows cumulative power in the battery after completion of each subinterval is zero.

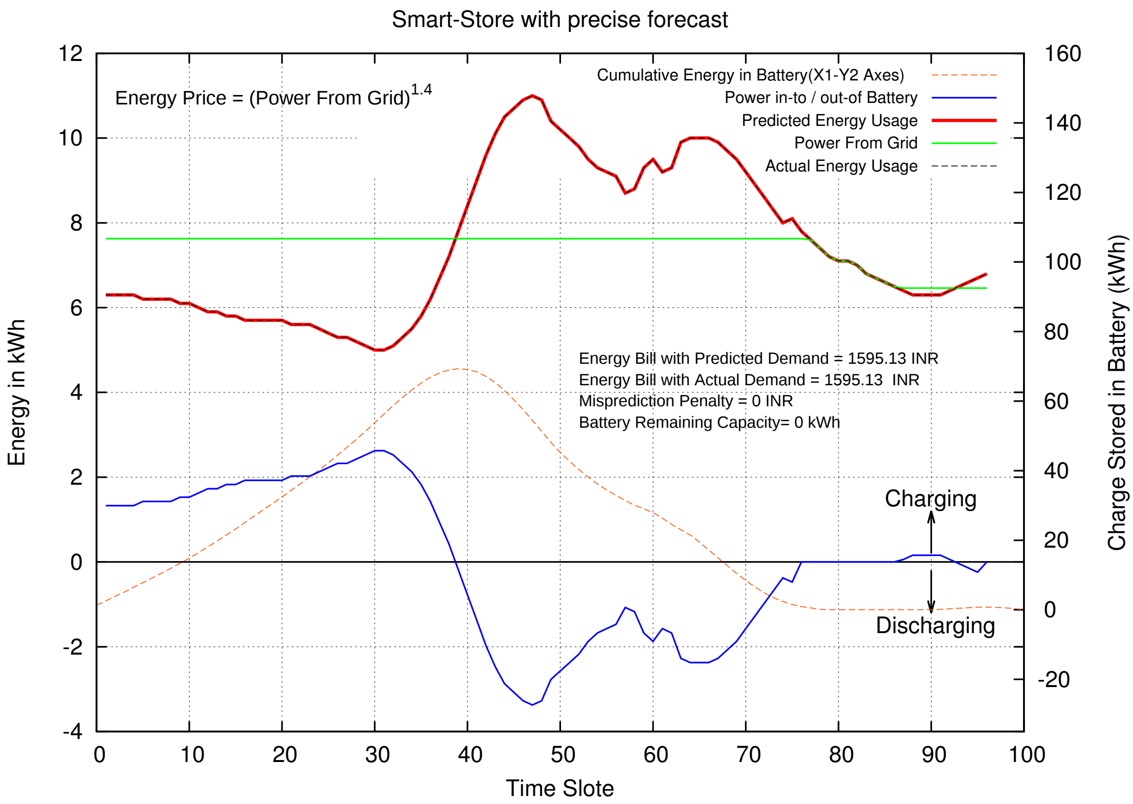


Figure 8.2: SmartStore with precise forecast

Figure 8.2 shows for precise forecast, the amount of charge left in the battery and 'Misprediction Penalty' after last slot is zero. 'Misprediction Penalty' is calculated using following formula.

$$\begin{aligned} MispredictionPenalty = CostOfElectricityWithPreciseForecast - \\ CostOfElectricityWithImpreciseForecast \end{aligned} \tag{8.1}$$

8.2.2 Overprediction of Energy Demands at Peak

Figure 8.3 shows the behavior of the SmartStore algorithm with a forecasted energy demand having overestimation at peak. Overestimation forces SmartStore-offline algorithm to calculated the elevated threshold than required. The amount of power taken from the grid depends on the threshold, hence more than required power is taken from the grid and stored in to the battery.

In figure 8.3 'Cumulative Charge in the battery' curve shows non-zero charge in the battery at the end of last slot. Overprediction of the energy demands results into increase in the cost of electricity and non-zero misprediction penalty.

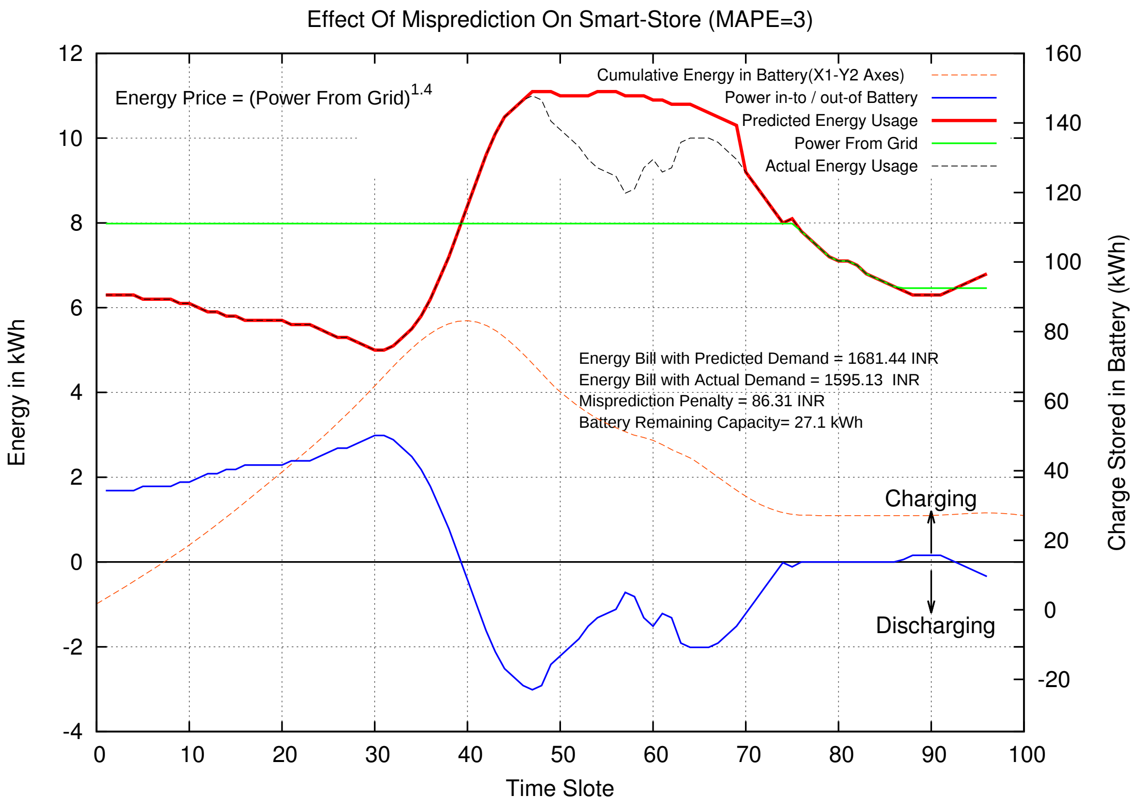


Figure 8.3: Effect of overestimation at the peak on SmartStore

8.2.3 Overprediction of Energy Demands at Non-Peak

Figure 8.4 also shows the effect of overprediction of the energy demands on SmartStore where overprediction occurred at non-peak region. From figure it's clear that effect of overprediction of the energy demands at non-peak is less penalizing as compared to the effect of overprediction of the energy demands at the peak region. Both 'Cumulative Charge in the battery' and 'Misprediction Penalty' is less in case of overprediction of the energy demands at non-peak, as compared to overestimation of the energy demands at peak.

8.2.4 Uniform Overprediction of Energy Demands

Figure 8.5 also shows the effect of overprediction of the energy demands on SmartStore where overprediction occurred uniformly over an entire region as opposed to overestimation at specific region. From figure 8.3,8.4 and 8.5 it's clear that overprediction of the energy demands is not desirable and reduces the saving opportunity.

8.2.5 Underprediction of Energy Demands at Peak

Figure 8.7 shows the behavior of the SmartStore algorithm with a forecasted energy demand having underestimation at peak. Underestimation forces SmartStore-offline algorithm to calculate the lower threshold than required. The amount of power taken from the grid depends on the threshold, hence less than required power is taken from the grid.

With Underestimated demands we can't satisfy the actual demand and battery will completely drain out before reaching the end of the subinterval. Once battery reaches to zero, then power taken from grid follows the actual demand. Hence the saving opportunity created by computing thresholds is lost.

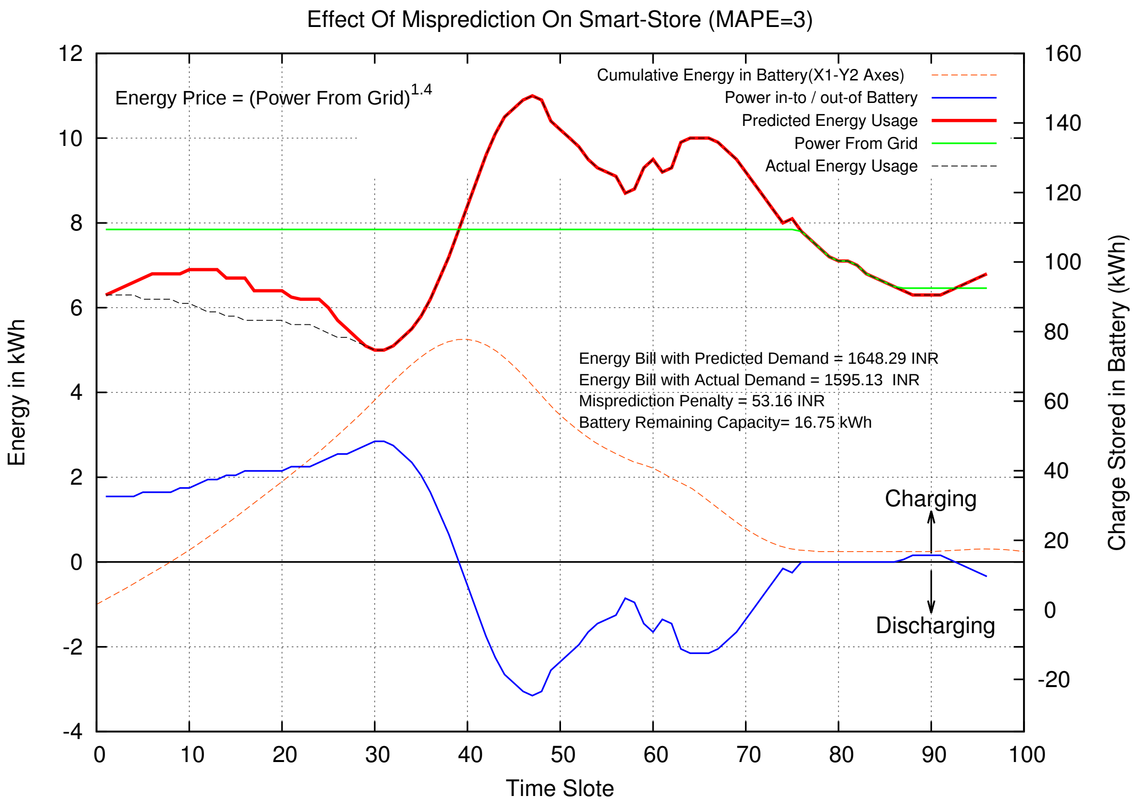


Figure 8.4: Effect of overestimation at the non-peak on SmartStore

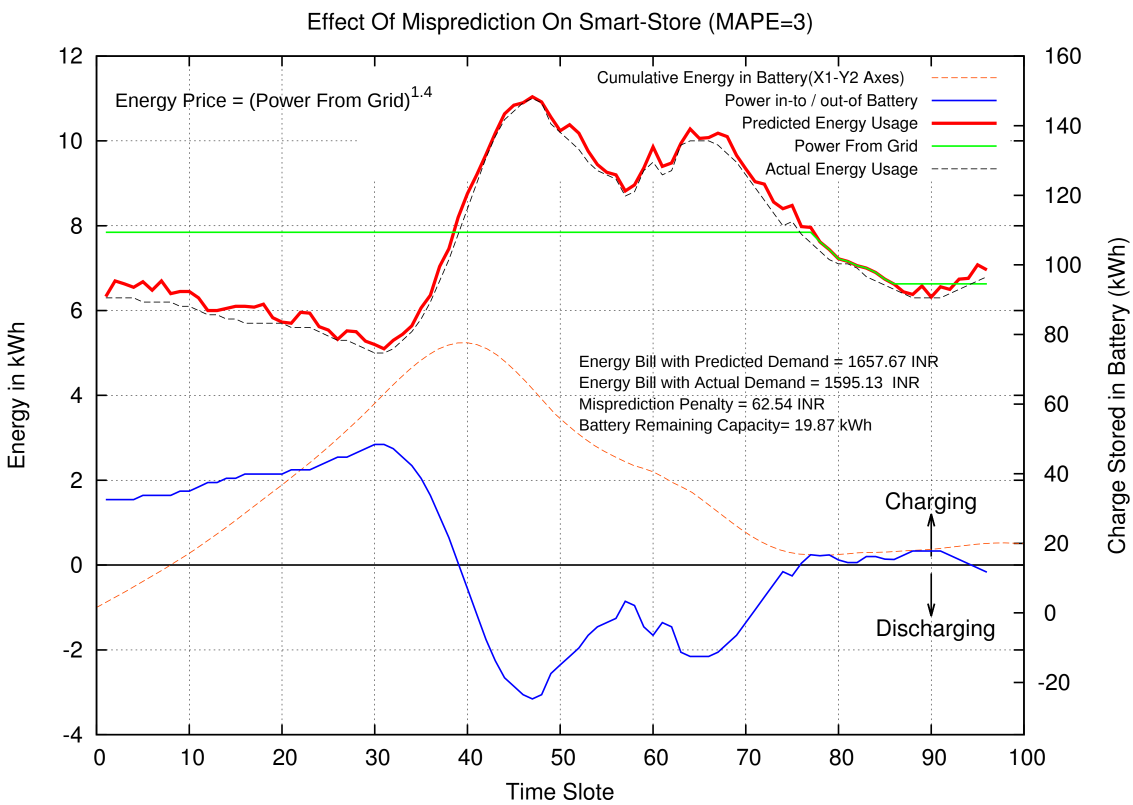


Figure 8.5: Effect of uniform overestimation on SmartStore

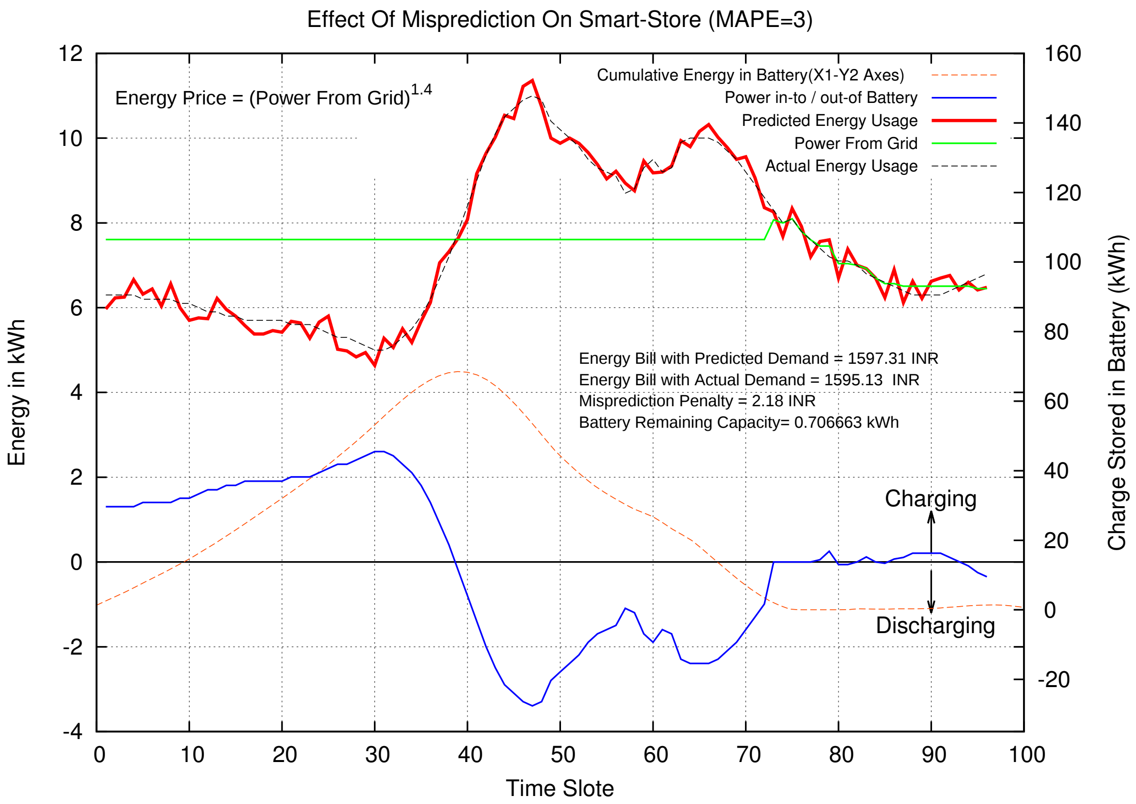


Figure 8.6: Effect of 3 % MAPE on SmartStore

In figure 8.7 'Cumulative Charge in the battery' curve hits the zero before the end of subinterval. Underpredicted of demand results into non-zero misprediction penalty and hence increase in the cost of electricity.

8.2.6 Underprediction of Energy Demands at Non-Peak

Figure 8.8 also shows the effect of undeprediction of the energy demands on SmartStore where undeprediction occurred at the non-peak region. From figure it's clear that the effect of undeprediction of the energy demands at non-peak is less penalizing as compared to the effect of undeprediction of the energy demands at the peak region. 'Misprediction Penalty' is less in case of undeprediction of the energy demands at non-peak, as compared to undeprediction of the energy demands at peak.

8.2.7 Uniform Underprediction of Energy Demands

Figure 8.9 also shows the effect of undeprediction of demands on SmartStore where undeprediction occurred uniformly over an entire region as opposed to undeprediction of the energy demands at specific region. From figure 8.7,8.8 and 8.9 it's clear that undeprediction of the energy demands is not desirable and reduces the saving opportunity.

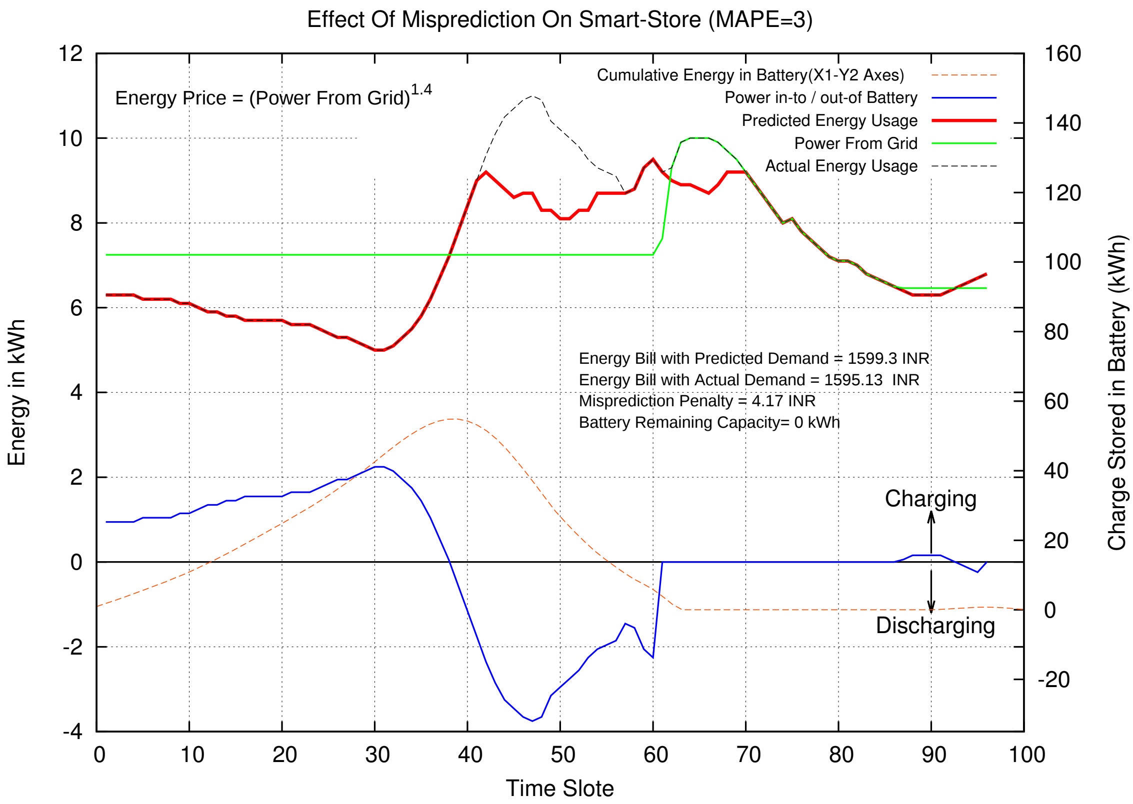


Figure 8.7: Effect of underestimation at Peak on SmartStore

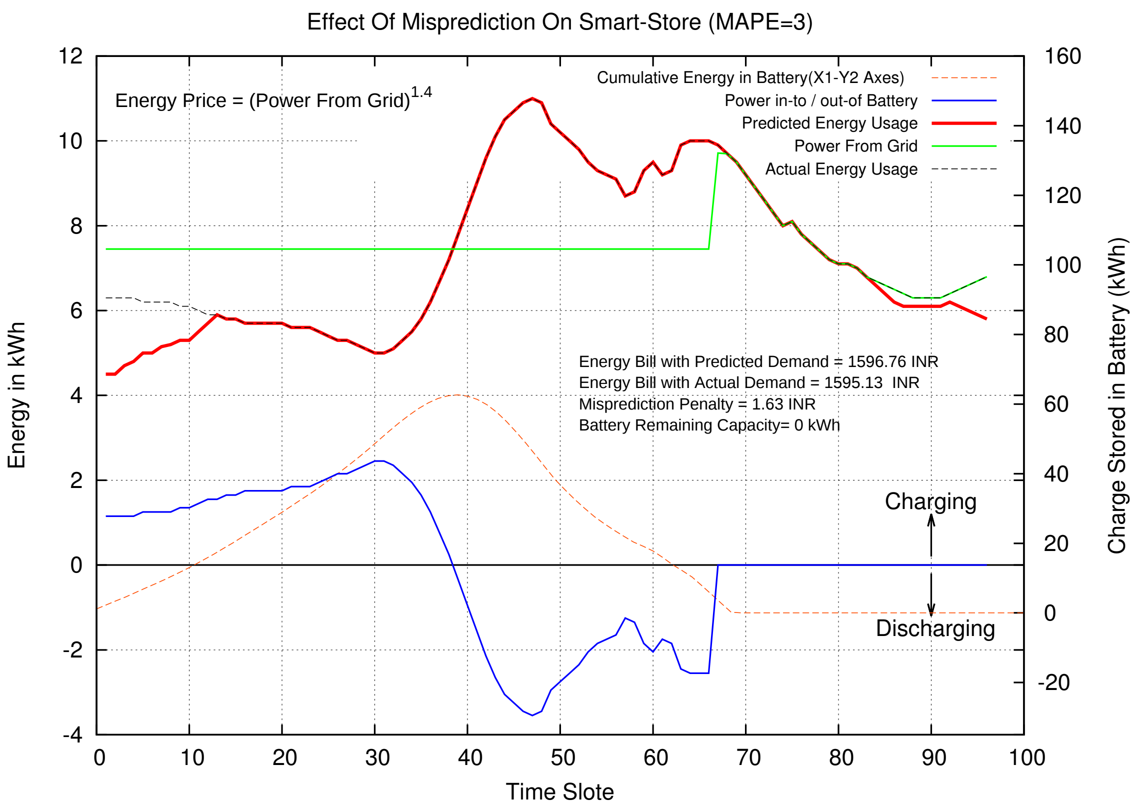


Figure 8.8: Effect of underestimation at non-peak on SmartStore

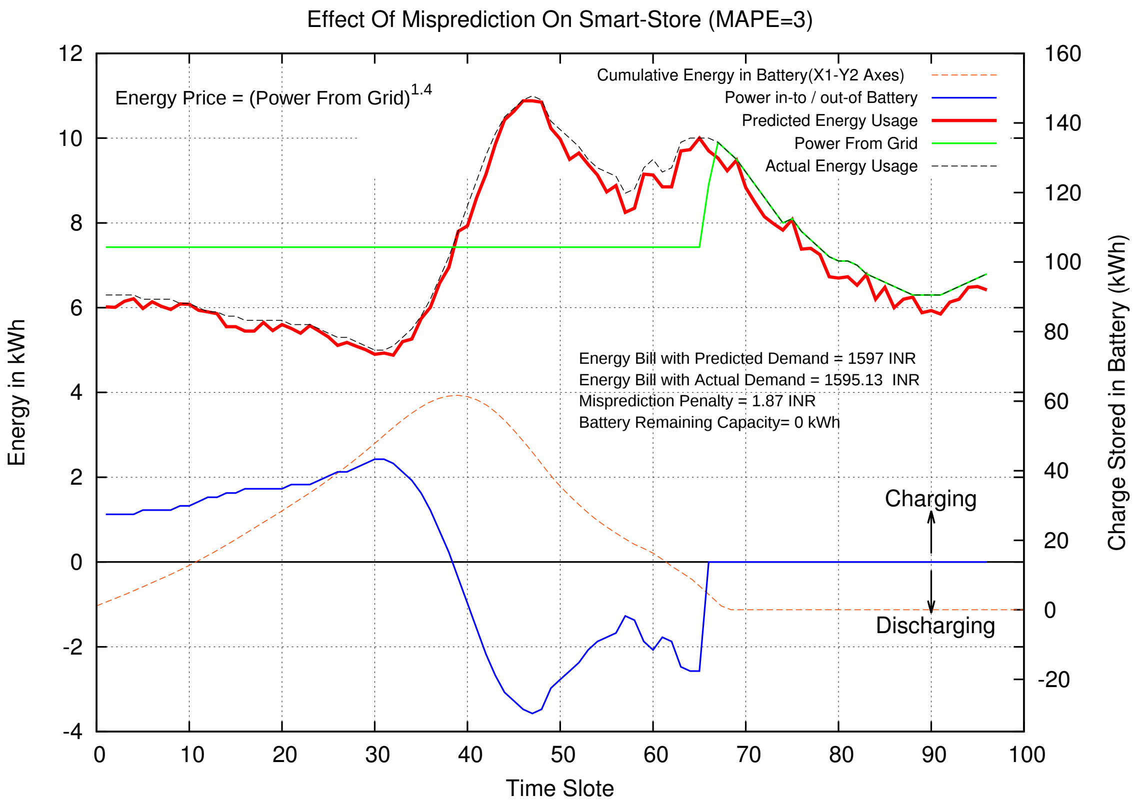


Figure 8.9: Effect of uniform underestimation on SmartStore

8.3 Simulation Results With Different MAPE

The table 8.1 summarizes the performance of SmartStore with different MAPEs. Table shows misprediction penalty increases with increase in MAPE. For constant MAPE, SmartStore shows changes in behavior depending upon the error location. Prediction error at peak fetches higher penalty, while uniform misprediction along curve get least penalty. Underprediction of energy demand gets lighter penalties, because battery does not holds extra charge for underpredicted demands.

	Misprediction Penalty (Rupees)				**Battery Remaining Capacity (kWh)**			
MAPE	**2%**	**3%**	**4%**	**5%**	**2%**	**3%**	**4%**	**5%**
Overestimation At Peak	56.83	86.31	117.26	147.44	17.9	27.1	36.7	46.00
Overestimation At Non-Peak	34.84	53.16	71.71	88.88	11	16.75	22.55	27.9
Uniform Overestimation	42.65	62.54	82.96	105.37	13.61	19.87	26.37	33.40
Uniform Misprediction	0.06	2.18	2.34	13.93	0	0.71	0.67	4.27
Uniform Underestimation	1.23	1.87	3.05	3.85	0	0	0	0
Underestimation at Peak	2.72	4.17	5.35	6.37	0	0	0	0
Underestimation at Non-Peak	1.27	1.63	2.32	3.13	0	0	0	0

Table 8.1: Effect of MAPE on SmartStore

8.4 Summary

In this chapter we studied the effect of different error values and error locations on the SmartStore algorithm.

Chapter 9

Extensions to SmartStore Algorithm

SmartStore is an offline energy cost minimization algorithm which uses battery to take the charging and discharging decisions by considering the future electricity demand. Precisely forecasting future electricity demand is very critical for the performance of SmartStore. In reality, actual electricity usage deviates from predicted electricity usage as no forecasting technique can be absolutely accurate. Chapter 8 shows, overprediction of the energy demands accumulates the extra charge in the battery.

In this chapter we will look into extensions to the SmartStore algorithm to use accumulated extra charge in the battery efficiently.

9.1 SmartStore Extension to Use Charge Accumulated in a Subinterval

In this section we will introduce an extension to the SmartStore algorithm, SmartStore-E1. SmartStore-E1 is an online algorithm. This algorithm will use the extra charge accumulated in a subinterval to recompute the thresholds, such that energy cost should be minimized.

9.1.1 Algorithm

For SmartStore-E1 algorithm, we are using same notations which we have introduced for the SmartStore algorithm in section 7.2.2. Procedure for calculating the subintervals and initial thresholds, is same as the SmartStore algorithm.

After calculating the subintervals and the corresponding thresholds, algorithm execute the following steps.

for $i = 1..I$ **do**
    if $(i \neq 1)$ **then**
        $extraCharge = B_{b_{i-1}} - B_1$
        if $((extraCharge - B_{min}) > 0)$ **then**
            $T_i = T_i - (extraCharge - B_{min})/(b_i - a_i)$
        end if
    end if
    for $t = a_i..b_i$ **do**
        if $(d_t > T_i)$ **then**
            Discharge $D_t = \min(d_t - T_i, B_t - B_{min})$ from battery and request T_i from grid
            $B_{t+1} = B_t - D_t$
        else
            Charge $R_t = \min(B_{max} - B_t, T_i - d_t)$ to battery and request T_i from grid
            $B_{t+1} = B_t + R_t$

```
        end if
    end for
end for
```

9.1.2 Analysis

SmartStore-E1, is an online algorithm. Time complexity of SmartStore-E1 algorithm is $O(n^3)$. Value of n is usually small. Hence, complexity of $O(n^3)$ is acceptable.

Figure 9.1 shows the working of the SmartStore-E1 algorithm with the help of same toy example used in chapter 8. Figure 9.1a shows the actual demands, forecasted usage/demands and thresholds calculated using the forecasted demands for the corresponding subintervals. In figure 9.1c the first subinterval i.e., $[1, 3]$ shows, at slot 1, and slot 2 actual demand is one less than the forecasted usage. Figure 9.1d, the charge left in the battery at the end of subinterval$[1, 3]$, is 2 Watts. Ideally the amount of charge left in the battery at the end of any interval should be zero.

As the SmartStore-E1 algorithm does not assume the accurate forecast of energy demand. It considers the extra charge available from previous subintervals and use that charge to readjust thresholds of the current subinterval. This readjustment of the interval can be seen in figure 9.1e. Precomputed Threshold for a second subinterval $[4, 5]$ was $T_2Old = 7.5$ and extra charge left in the first subinterval $[1, 3]$ was 2 Watts. This 2 Watts power is evenly distributed over a second subinterval $[4, 5]$ to compute a new threshold as $T_2New = 6.5$. Readjustment of threshold ensures that extra charge accumulated in any subinterval, get used in next subinterval. Now with new thresholds SmartStore-E1 algorithm will work exactly same as the SmartStore algorithm, till the end of current subinterval. Charging and discharging decisions made by SmartStore-E1 in a second subinterval $[4, 5]$ are shown in figure 9.1e and figure 9.1f.

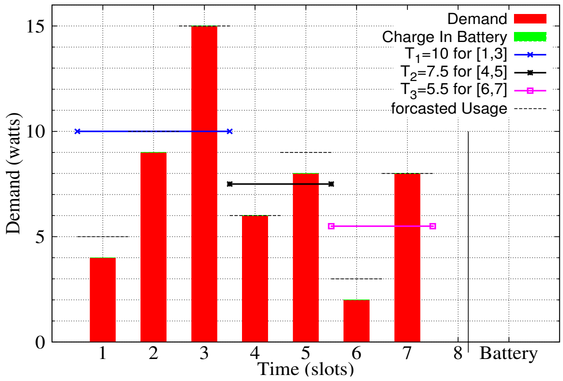
At the end of a second subinterval $[4, 5]$, 1 watt charge is left in the battery. Figure 9.1g shows how that charge is used to readjust the threshold of subinterval $[6, 7]$ from $T_3Old = 5.5$ to $T_3New = 5$. Figure 9.1h shows the states of the battery at the end of last slot.

Comparison of final results of SmartStore-E1 in figure 9.1h with final results of the SmartStore algorithm in figure 8.1h shows that, the amount of charge left in the battery after applying SmartStore-E1 is less as compared to SmartStore. Hence SmartStore-E1 outperforms SmartStore algoritm.

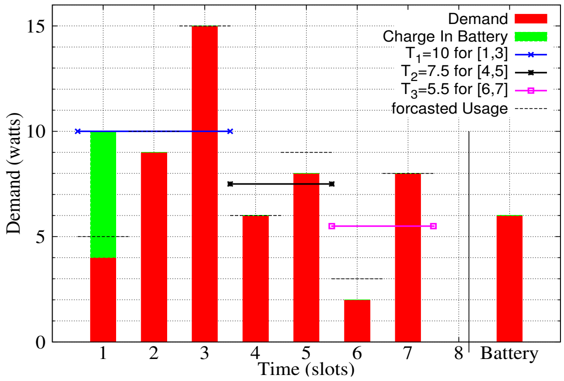
9.1.3 SmartStore-E1 With Synthetic Dataset

Figure 9.2 shows the behavior of the SmartStore-E1 algorithm with uniform overprediction of energy demand over an entire region. SmartStore-E1 works exactly same as the SmartStore-offline algorithm during first subinterval, hence thresholds computed in SmartStore-E1 are same as SmartStore algorithm for this interval. The dataset we have considered have a uniformly over-predicted energy demand data, hence the thresholds computed at step-1 are elevated thresholds. Elevated thresholds make algorithm to put an extra charge into the battery than required.

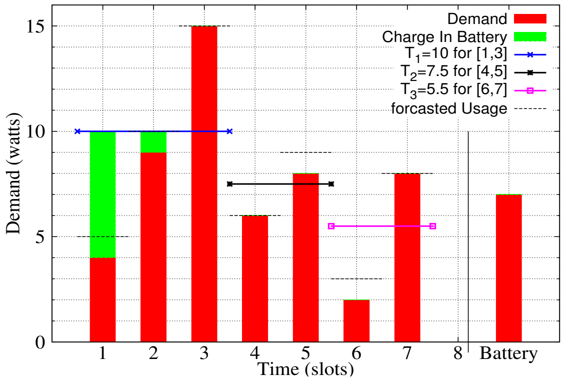
Figure 9.2 shows the charge accumulated in first subinterval $[1, 77]$, is used to readjust the thresholds of subsequent subintervals. In this case charge accumulated in first subinterval is higher than total demand of succeeding subinterval, hence readjustment of thresholds change the threshold for second subinterval to zero. The zero threshold for any slot ensures no power from the grid and entire electricity demand is satisfied from the battery alone. In figure 'Power From Grid' curve shows sudden trough during second subinterval which starts from slot 77, as major portion of energy demand is satisfied from the battery.



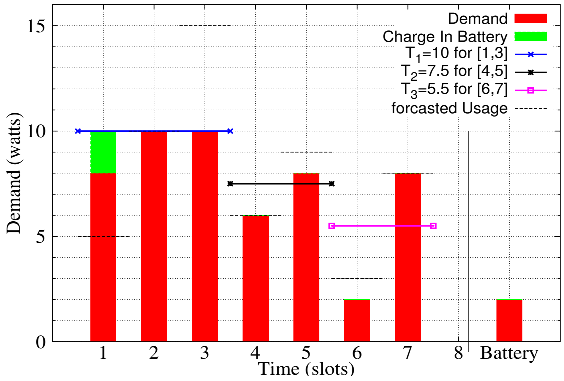
(a) Actual Demands and Precomputed Thresholds using predicted demand for billing period of 7 slots



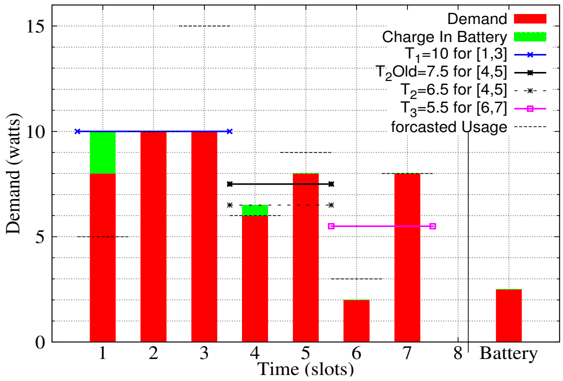
(b) Slot 1 : $R_1 = 6$



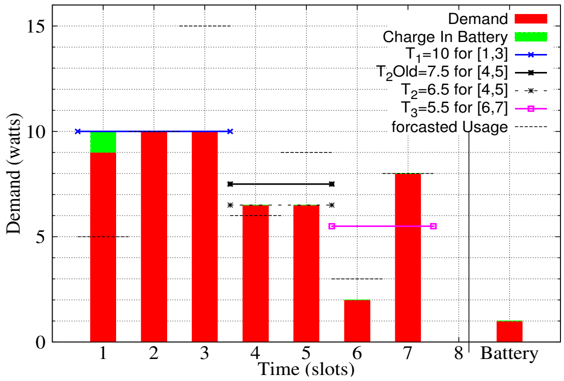
(c) Slot 2 : $R_2 = 1$



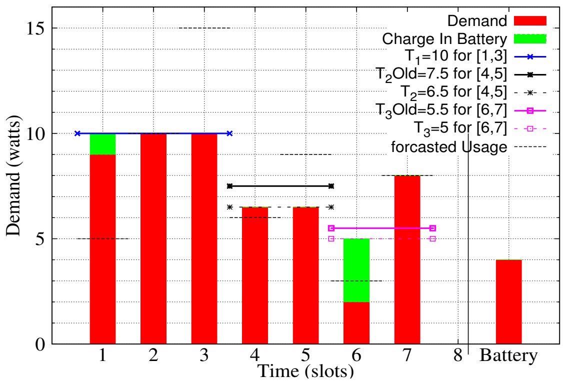
(d) Slot 3 : $D_3 = 5$



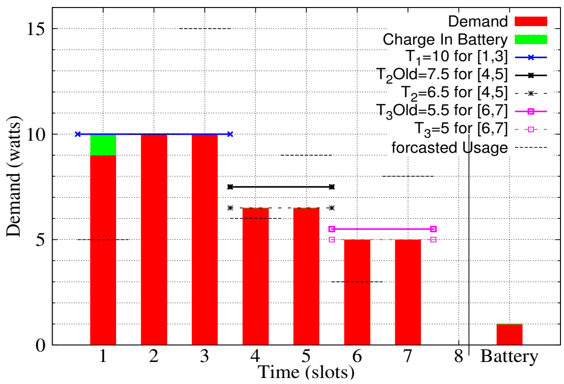
(e) Slot 4 : $R_4 = 0.5$



(f) Slot 5 : $D_5 = 1.5$



(g) Slot 6 : $R_6 = 3$



(h) Slot 7 : $D_7 = 3$, Charge left in the battery is 1 Watts

Figure 9.1: A SmartStore Algorithm, Extension-1

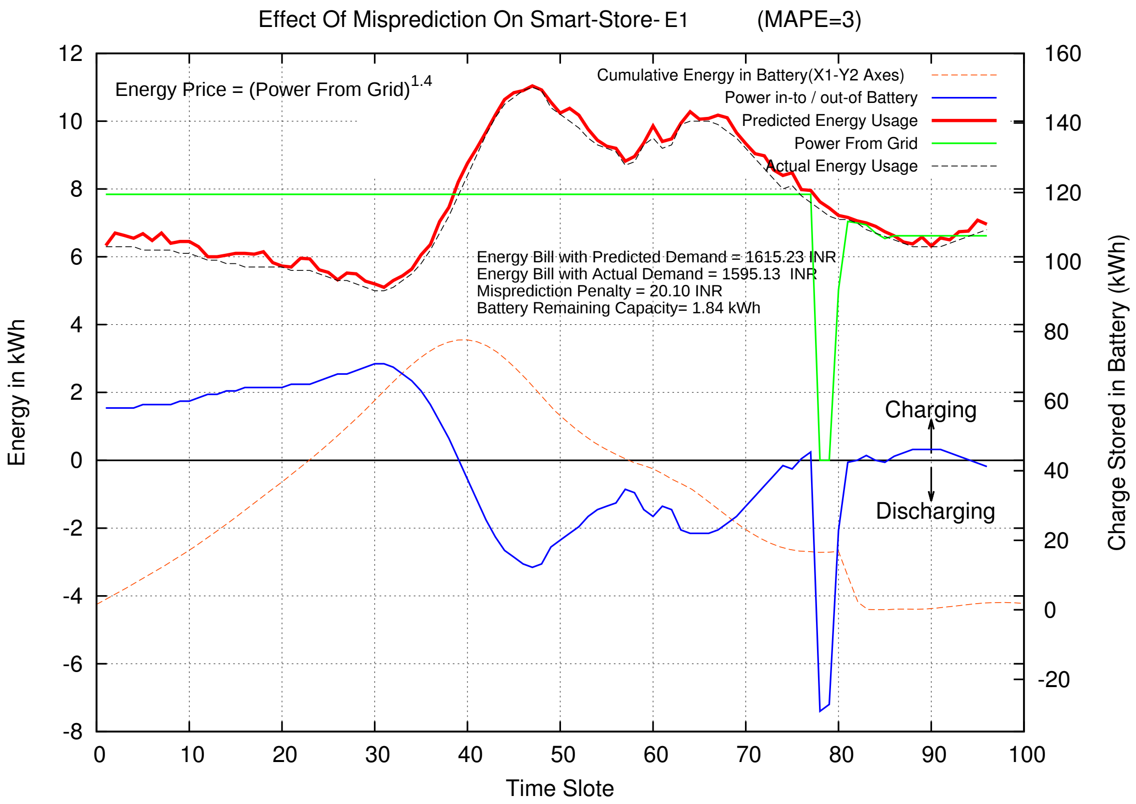


Figure 9.2: Effect of uniform overestimation on SmartStore-E1

9.1.4 Observations from SmartStore-E1

- SmartStore-E1 does better than the SmartStore algorithm in terms of energy cost minimization by using the extra charge in the battery accumulated in previous subinterval.
- SmartStore-E1 does threshold adjustment after every subinterval.
- Entire charge accumulated in current subinterval is used in next subinterval, hence power from the grid in the next subinterval suddenly drops by the amount equal to charge accumulated in previous subinterval.
- Charge accumulated in the last subinterval stays inside the battery as SmartStore-E1 does not have any provisions to utilize this charge.
- Unused charge from the last subinterval increases the total energy cost.
- Sudden change in the power drawn from the grid is not good for grid-health.
  - Consider a scenario where a grid is suppling a power to some households, all equipped with SmartStore-E1 technology to reduce energy cost. If threshold re-computation events of the majority of households happens in-phase, then total demand from the grid will suddenly drop.
  - This will increase the grid frequency and in worst case may cause some of the generators to trip-off from the power network.

9.2 SmartStore Extension to Use Extra Charge Accumulated in a Slot

In this section we will introduce an extension to the SmartStore algorithm, SmartStore-E2. SmartStore-E2 is an online algorithm. This algorithm will use the extra charge accumulated in a slot to recompute the thresholds, such that energy cost should be minimum.

9.2.1 Algorithm

For SmartStore-E2 algorithm, we are using same notations which we have introduced for the SmartStore algorithm in section 7.2.2. The procedure for calculating the subintervals and initial thresholds, is same as the SmartStore algorithm.

After calculating the subintervals and the corresponding thresholds, algorithm execute the following steps.

for $i = 1..I$ **do**
    for $t = a_i..bi$ **do**
        if $(t \neq 1)$ **then**
            $extraCharge = d_{t-1} - d^a_{t-1}$
            if $(extraCharge > 0)$ **then**
                $T_i = T_i - extraCharge/(b_i - t)$
            end if
        end if
        if $(d_t > T_i)$ **then**
            Discharge $D_t = \min(d_t - T_i, B_t - B_{min})$ from battery and request T_i from grid
            $B_{t+1} = B_t - D_t$
        else
            Charge $R_t = \min(B_{max} - B_t, T_i - d_t)$ to battery and request T_i from grid
            $B_{t+1} = B_t + R_t$
        end if
    end for
end for

9.2.2 Analysis

SmartStore-E2, is an online algorithm. The time complexity of a SmartStore-E2 algorithm is $O(n^3)$. Value of n is usually smaller. Hence, complexity of $O(n^3)$ is acceptable.

Figure 9.3 shows the working of the SmartStore-E2 algorithm with the help of same toy example used in chapter 8. Figure 9.3a shows the actual demands, forecasted usage/demands and thresholds calculated using the forecasted demands for the corresponding subintervals. In figure 9.3b the first subinterval i.e., $[1, 3]$ shows, at slot 1 actual demand is one less than the forecasted usage. Hence after slot 1, extra charge left in the battery is 1 Watt. This extra charge is used to recompute the threshold for slot 2, and slot 3 in a first subinterval $[1, 3]$. Initial threshold computed for first subinterval was $T_1(1) = 10$. After using 1 watt extra power from slot 1, to recompute the threshold, a new threshold for first subinterval will be $T_1(2) = 9.5$. This updated threshold for first subinterval, is shown in figure 9.3c. Figure 9.3c also shows that actual demand in slot 2, is one less than the forecasted usage. The difference between forecasted and actual energy demand is again used to recompute the new threshold for first subinterval $[1, 3]$ using SmartStore-E2. Figure 9.3d shows the change in the threshold from $T_1(2) = 9.5$ to $T_1(2) = 8.5$. At the end of first subinterval $[1, 3]$, SmartStore-E2 have effectively used extra charge stored during slot 1 and slot 2, hence battery is empty after slot 3.

At slot 3, actual energy usage follows the forecasted energy usage. Hence no extra charge is accumulated in the battery after slot 3. This keeps initial thresholds computed from subinterval $[4, 5]$ unchanged. Figure 9.3e, shows the unchanged thresholds $T_2 = 7.5$. Again, at slot 4, actual demand follows the forecasted demand, hence this will keep the threshold for a second subinterval $[4, 5]$ intact. Figure 9.3f shows, at slot 5 actual demand is one less than the forecasted demand. This difference between forecasted and actual energy demand is again used to recompute the new threshold for last subinterval $[6, 7]$ using SmartStore-E2. Figure 9.3g shows the change in the threshold from $T_3(1) = 5.5$ to $T_3(2) = 5$.

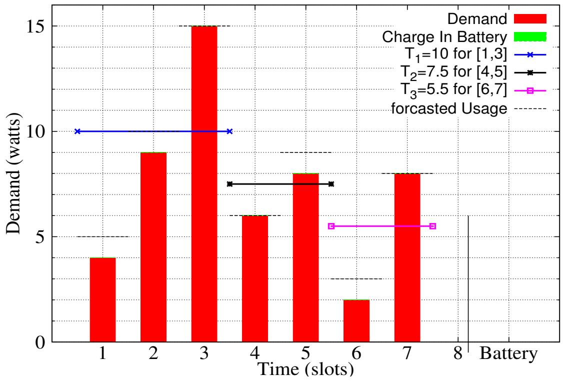
Similarly, 1 watt difference between actual and forecasted energy demand at slot 6, is again used to recompute the thresholds for third subinterval. Figure 9.3h shows this change in the threshold from $T_3(2) = 5$ to $T_3(3) = 4.5$. In last slot 7, actual demand follows the forecasted demand, hence battery is empty at the end of last subinterval.

Comparison of final results of SmartStore-E2 in figure 9.3h with final results of the SmartStore-E1 algorithm in figure 9.1h shows that, no charge left in the battery after applying SmartStore-E2 as compared to SmartStore-E1 where 1 watt charge was left in the battery. Hence SmartStore-E2 outperforms SmartStore-E1 algoritm.

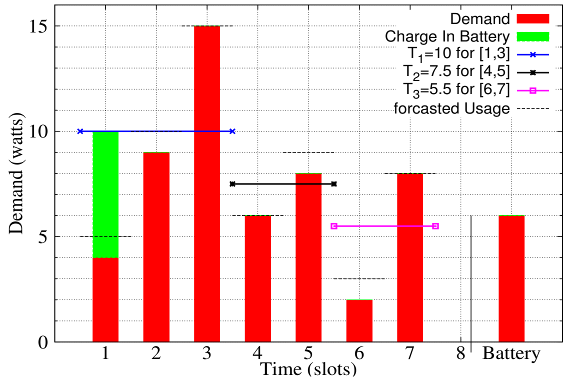
9.2.3 SmartStore-E2 With Synthetic Dataset

Figure 9.4 shows behavior of the SmartStore-E2 algorithm with uniform overprediction of energy demand over an entire region. SmartStore-E2 works same as SmartStore-E1 algorithm only during first slot. Here we have considered the example of uniform overprediction of the energy demands, hence at each slot difference between the forecasted energy usage and actual energy usage will always be a non-zero number. SmartStore-E2 will consider the this extra charge accumulated in previous slot to adjust the current threshold. In the case of uniform overprediction, SmartStore-E2 will always decrement the threshold value, to accommodate previous slot's overprediction. For all variants of SmartStore power taken from grid at any slot is equal to threshold at that slot.

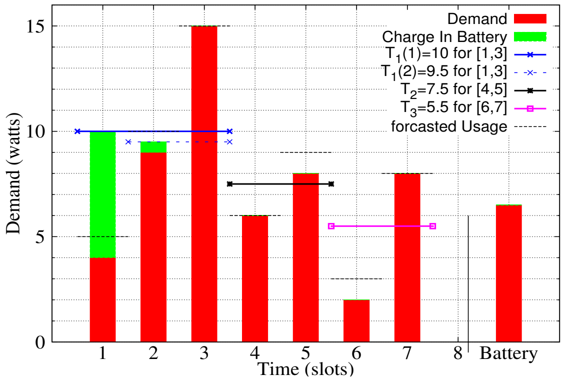
In figure 9.4 'Power From Grid' curve show the decreasing threshold in first subinterval $[1, 77]$.



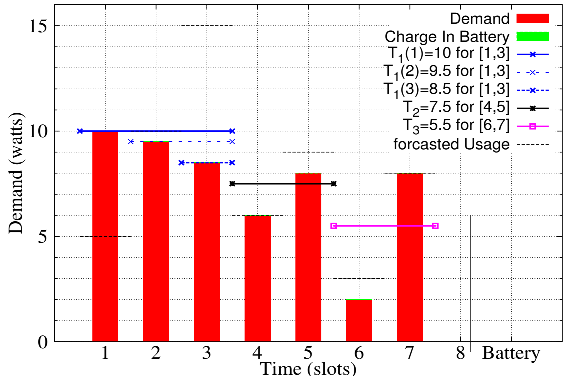
(a) Actual Demands and Precomputed Thresholds using predicted demand for billing period of 7 slots



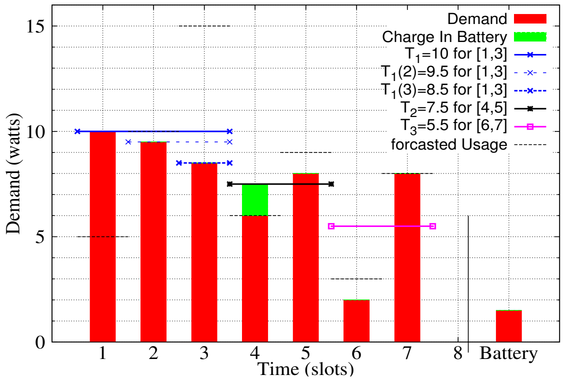
(b) Slot 1 : $R_1 = 6$



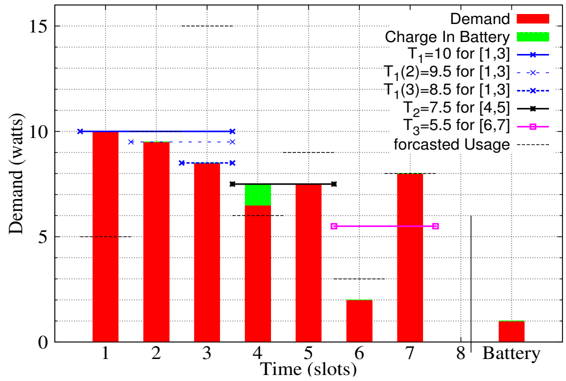
(c) Slot 2 : $R_2 = 0.5$



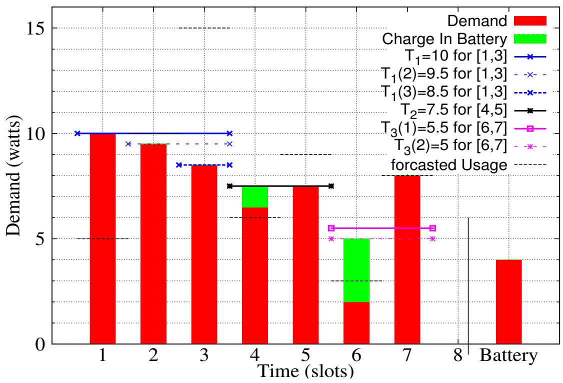
(d) Slot 3 : $D_3 = 6.5$



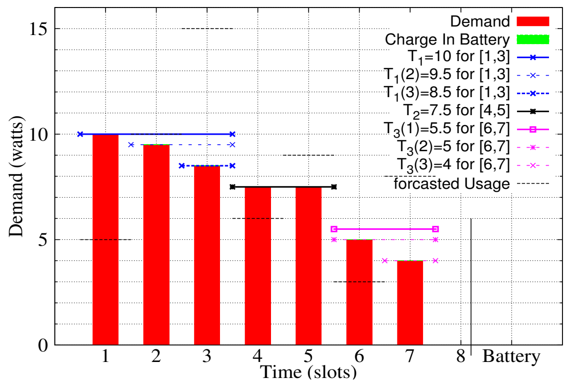
(e) Slot 4 : $R_4 = 1.5$



(f) Slot 5 : $D_5 = 0.5$



(g) Slot 6 : $R_6 = 3$



(h) Slot 7 : $D_7 = 4$, Charge left in the battery is 0 Watts

Figure 9.3: A SmartStore Algorithm, Extension-2

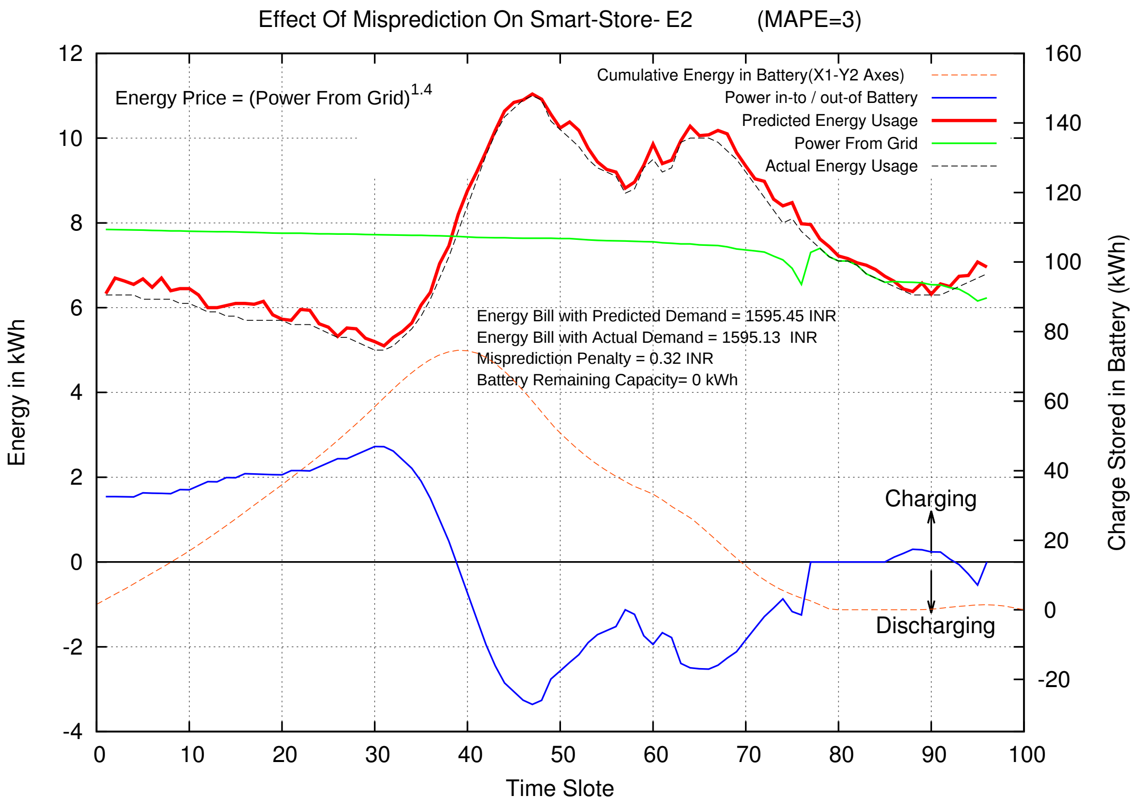


Figure 9.4: Effect of uniform overestimation on SmartStore-E2

9.2.4 Observations from SmartStore-E2

- SmartStore-E2 does better than the SmartStore algorithm in terms of energy cost minimization by using the extra charge in the battery accumulated in previous slot.
- SmartStore-E2 does threshold adjustment after every slot.
- Charge accumulated in current slot is used to adjust the thresholds of remaining slots in the current subinterval.
- Threshold adjustment happens in every slot.
- Charge accumulated in last slot stays inside the battery as SmartStore-E2 does not have any provisions to utilize this charge.
- Unused charge from last slot slightly increases the total energy cost.
- Sudden change in the power drawn from the grid is less severe than SmartStore-E1.

9.3 Performance Comparison

Table 9.1 shows the behavior of SmartStore and its variants with the synthetic data prepared. Observations from table 9.1 are as follows,

- 'Misprediction Penalty' and 'Battery Remaining Capacity' increases with MAPE.
- For dataset related to the overprediction, 'Misprediction Penalty' and 'Battery Remaining Capacity' is highest for the SmartStore algorithm and decreases for SmartStore-E1 and SmartStore-E2, in that order.

- For dataset related to the Underestimation, 'Misprediction Penalty' and 'Battery Remaining Capacity' is same for SmartStore and its variants.

		Misprediction Penalty(INR)				**Battery Capacity Left(KWh)**			
MAPE		**2%**	**3%**	**4%**	**5%**	**2%**	**3%**	**4%**	**5%**
Overestimation At Peak	**SS**	56.83	86.31	117.26	147.44	17.9	27.1	36.7	46.00
	SS-E1	14.76	23.57	31.87	43.22	0	0	0	0
	SS-E2	1.03	2.66	4.61	8.54	0	0	0	0
Overestimation At Non-Peak	**SS**	34.84	53.16	71.71	88.88	11	16.75	22.55	27.9
	SS-E1	8.22	14.26	20.53	24.52	0	0	0	0
	SS-E2	0.03	0.07	0.1	0.18	0	0	0	0
Uniform Overestimation	**SS**	42.65	62.54	82.96	105.37	13.61	19.87	26.37	33.40
	SS-E1	12.71	20.1	25.86	29.14	1.64	1.84	2.76	1.6
	SS-E2	0.16	0.32	0.52	0.76	0	0	0	0
Uniform Misprediction	**SS**	0.06	2.18	2.34	13.93	0	0.71	0.67	4.27
	SS-E1	0.08	0.1	0.25	5.28	0	0	0	0
	SS-E2	0.36	0.57	0.74	0.48	0	0	0	0
Uniform Underestimation	**SS**	1.23	1.87	3.05	3.85	0	0	0	0
	SS-E1	1.23	1.87	3.05	3.85	0	0	0	0
	SS-E2	1.23	1.87	3.05	3.85	0	0	0	0
Underestimation at Peak	**SS**	2.72	4.17	5.35	6.37	0	0	0	0
	SS-E1	2.72	4.17	5.35	6.37	0	0	0	0
	SS-E2	2.72	4.17	5.35	6.37	0	0	0	0
Underestimation at Non-Peak	**SS**	1.27	1.63	2.32	3.13	0	0	0	0
	SS-E1	1.27	1.63	2.32	3.13	0	0	0	0
	SS-E2	1.27	1.63	2.32	3.13	0	0	0	0

Table 9.1: Performance Comparison of SmartStore and Its Extensions

9.4 Summary

In this chapter we proposed two extension to SmartStore algorithm namely SmartStore-E1 and SmartStore-E2. We compared the performance of these algorithms on the synthetic datasets.

Chapter 10

Performance of SmartStore and Its Extensions with Real-Data

In this chapter we analyze SmartStore and its extensions with real datasets using simulations. We analyze the effect of misprediction s of the energy demands. We also study the effect of battery size on the energy cost in the context of SmartStore.

10.1 Dataset

We used the following datasets to analyze SmartStore.

10.1.1 KRESIT Dataset

KRESIT is an academic building in IIT-Bombay campus. KRESIT dataset contains energy consumption data for 30 Days, starting from 21/Nov/2012 up to 20/Dec/2012. 'Support Vector Regression' is used to forecast energy demand for the same duration. Average 'Mean Absolute Percentage Error'(MAPE) calculated on 30 days forecasted data is *9.93*.

In the dataset, observed MAPE on weekends is more than average MAPE over 30 days. Average MAPE on weekends is 13.48. The Dataset contains actual energy consumption and predicted energy usage at the granularity of 15 minutes. Maximum energy consumption reported in dataset for any 15 minute slot is 15 KWh, while minimum energy consumption is 3 KWh.

10.1.2 TATA POWER Dataset

TATA POWER is a power distribution company. It supplies power to Mumbai city. The dataset contains energy consumption data for 30 Days, starting from 1/Jan/2012 up to 30/Jan/2012. 'Artificial Neural Network' is used to forecast energy demand for the same duration. Average 'Mean Absolute Percentage Error'(MAPE) calculated on 30 days forecasted data is *3.07*.

Dataset contains actual energy consumption and predicted energy usage at the granularity of 15 minutes. Maximum energy consumption reported in dataset for any 15 minute slot is 432 MWh, while minimum energy consumption is 202 MWh [1].

10.2 Simulations with KRESIT Dataset

Table 10.1 shows the result of simulation experiments on KERSIT dataset. Values of all the attributes shown in the table are averaged over 30 days. Table 10.1 shows average values of electricity cost is less for SmartStore-E1 as compared to SmartStore, and further less for SmartStore-E2. Percentage electricity cost reduction from SmartStore to SmartStore-E1 is

[1] TATA POWER dataset used is downscaled from MWh to KWh for simulation analysis

1.61%, while for SmartStore-E2 it is *4.17*%.

Average battery remaining capacity is least in the case of SmartStore-E2 in comparison with other two. 'Average Misprediction Penalty' is highest for SmartStore, while it is least for SmartStore-E2. Misprediction penalty of *0.52*% is less for SmartStore-E2 and ensures least energy cost in the presence of mispredictions in the forecasted energy demand in comparison with other two.

×	SmartStore	SmartStore-E1	SmartStore-E2
Average Electricity Cost (In Rupees)	1504.14	1479.52	1441.33
Average Battery left (In KWh)	22.97	12.45	0.24
Average Misprediction Penalty (In Rupees)	70.31	45.7	7.51
Percentage Misprediction Penalty	4.67	3.09	0.52

Table 10.1: SmartStore and Its Extensions With KRESIT Dataset

10.3 Simulations with TATA POWER Dataset

Table 10.2 shows the result of simulation experiments on TATA POWER dataset. Values of all the attributes shown in the table are averaged over 30 days. Table 10.2 shows average values of electricity cost is less for SmartStore-E1 as compared to SmartStore, and lesser for SmartStore-E2. Percentage electricity cost reduction from SmartStore to SmartStore-E1 is *0.49*%, while for SmartStore-E2 it is *0.76*%.

Average battery remaining capacity is least in the case of SmartStore-E2 in comparison with other two. 'Average Misprediction Penalty' is highest for SmartStore, while it is least for SmartStore-E2. Misprediction penalty of *0.08*% is less for SmartStore-E2 and ensures least energy cost in the presence of mispredictions in the forecasted energy demand in comparison with other two.

×	SmartStore	SmartStore-E1	SmartStore-E2
Average Electricity Cost(In Rupees)	320287.17	318716.83	317835.67
Average Battery left (In KWh)	186.67	38.55	5.26
Average Misprediction Penalty (In Rupees)	2704.4	1134.1	252.9
Percentage Misprediction Penalty	0.84	0.36	0.08

Table 10.2: SmartStore and Its Extensions With TATA POWER Dataset

10.4 Effect of Battery Size on SmartStore and Its Extensions

SmartStore and its extensions uses batteries to minimize the energy cost. The savings offered by these price minimization techniques critically depends upon the size of the battery. In this section we will study the impact of maximum battery capacity on the SmartStore and its extensions.

10.4.1 SmartStore with Varying Battery Size

Figure 10.1 shows the plot for average energy cost for 30 days versus battery size for the KRESIT dataset. Figure shows for SmartStore and SmartStore-E1 energy cost goes up, as battery size increases. For SmartStore-E2 follows the ideal behavior, where energy cost goes down with increase in battery size. Figure 10.1 also shows after some point increase in battery size does not have an equivalent energy cost reduction.

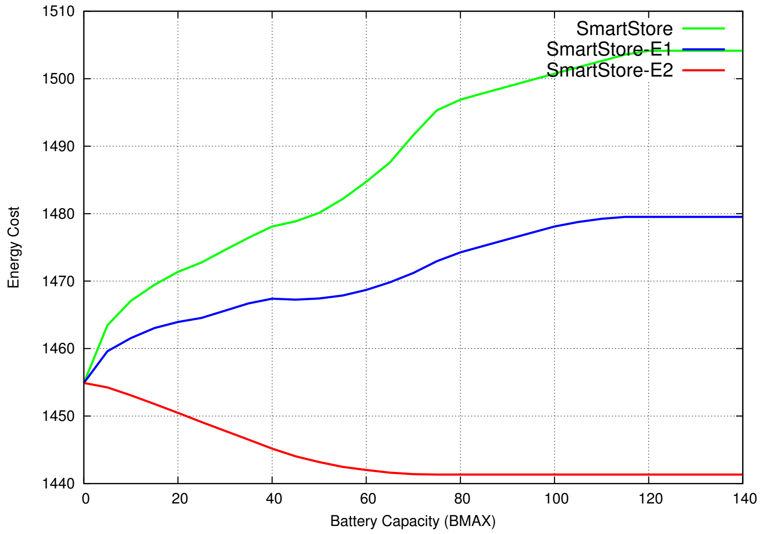


Figure 10.1: Average Energy Cost For 30 Days versus Battery Size

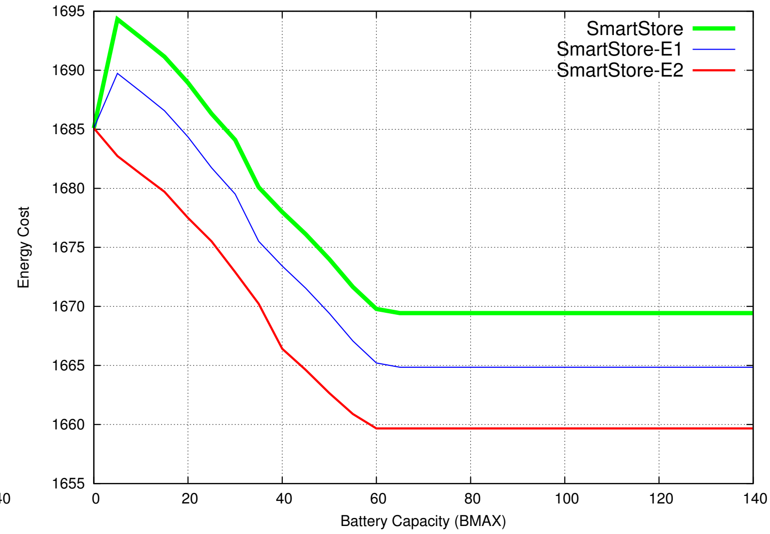


Figure 10.2: Energy Cost versus Battery Size, On Weekdays

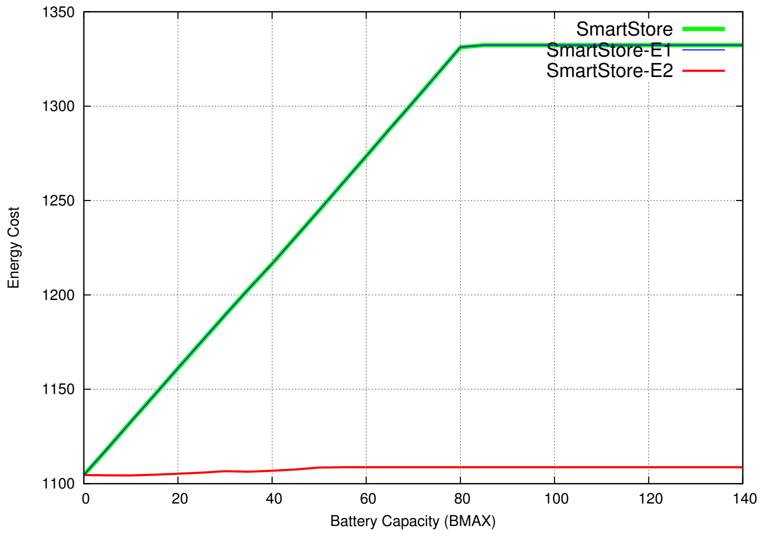


Figure 10.3: Energy Cost versus Battery Size, On Weekends

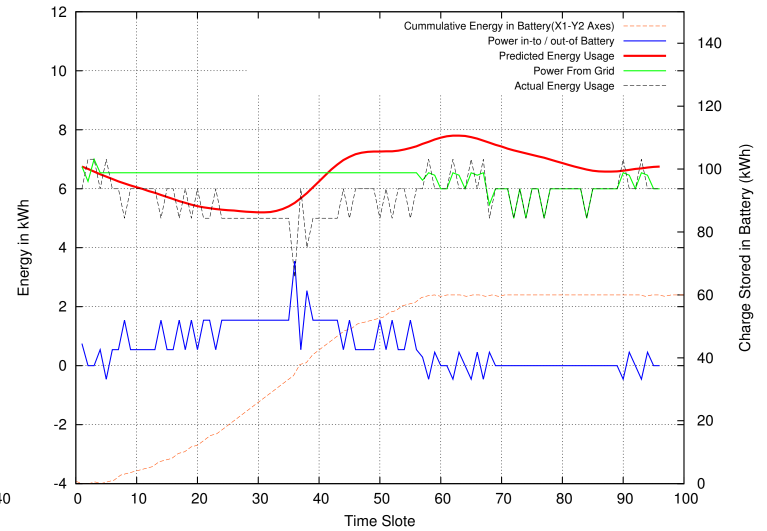


Figure 10.4: Energy Consumption On Typical Weekends With SmartStore-E1, $B_{max} = 60KWh$

10.4.2 Anomalous Behavior of SmartStore

Ideally, savings opportunities should increase as the battery size increases. SmartStore and SmartStore-E1 shows anomalous behavior as energy cost also goes up with increase in battery size. In this section we analyze the anomaly in SmartStore.

Figure 10.1 shows the plot for average energy cost for 30 days versus battery size. Averaging of energy cost over 30 days gives the overall picture, but it suppresses the effect of energy cost of individual day. Also, the KRESIT dataset we are using has a higher MAPE on weekends than weekdays. We study the impact of weekdays and weekends on energy cost separately.

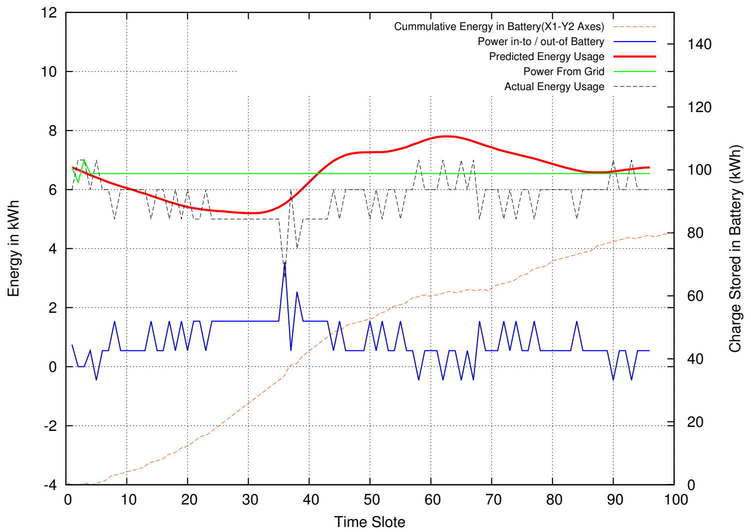


Figure 10.5: Energy Consumption On Typical Weekends With SmartStore-E1, $B_{max} = 100KWh$

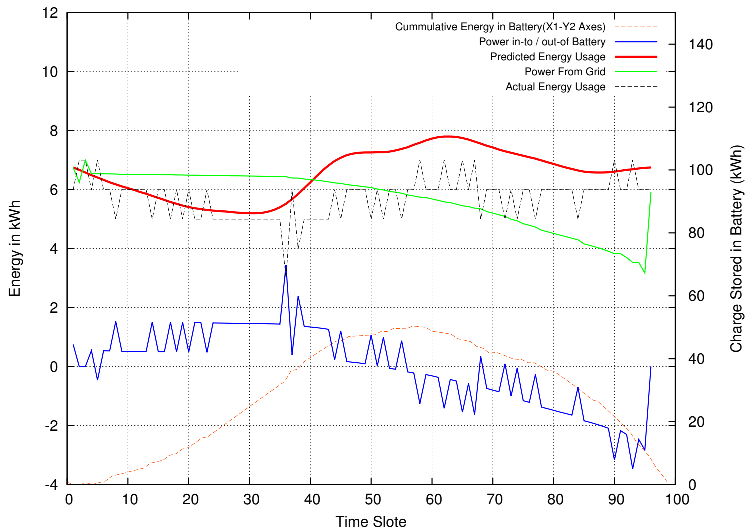


Figure 10.6: Energy Consumption On Typical Weekends With SmartStore-E2, $B_{max} = 60KWh$

Figure 10.2 shows the plot for energy cost versus battery size for a typical Weekday. Figure follows the ideal behavior as energy cost drops with increase in battery size. In Figure 10.2 curve plotted for SmartStore shows that absolute energy cost reduction after using a 60 Watts battery is 15.31 INR (1669.79 - 1685.1), than not using battery at all.

'SmartStore-E1' is a special case of original 'SmartStore' algorithm where only single threshold exist.

Figure 10.3 shows the plot for energy cost versus battery size for a typical Weekend. In figure 10.3 curves for 'SmartStore' and 'SmartStore-E1' coincides. Figure contradicts with the ideal behavior between energy cost and battery size. For typical weekend energy cost increases with battery size. The curve plotted for SmartStore shows that absolute energy cost increase after using a 60 Watts battery is 168.91 INR (1273.48-1104.57), than not using battery at all. From figure 10.2 and figure 10.3 it's clear that sum of energy cost savings achieved during 5 weekdays in a week, is less than the extra cost incurred on Saturday and Sunday.

To analyze, why energy cost goes up on weekends, we choose two points from curve 'SmartStore-E1' of figure 10.3. We analyze the curve SmartStore-E1at $B_{max} = 60KWh$ and $B_{max} = 100KWh$. Figure 10.4 shows the behavior of SmartStore-E1 with $B_{max} = 60KWh$. On typical weekdays energy consumption curve is relatively flat, hence energy cost saving opportunities with demand based non-linear pricing policy are insignificant. The KRESIT dataset have higher MAPE on weekends, than weekdays. SmartStore computes the thresholds based on the forecasted value of energy demand. In figure 10.4, for given mispredicted energy demand SmartStore computes elevated thresholds. Higher values of threshold unnecessarily accumulates extra charge in the battery. Simulation experiment shown in figure 10.4 has a maximum battery capacity limited to 60 KWh, which restricts the SmartStore-E1 from accumulating extra charge. While, simulation experiment shown in figure 10.5 has a maximum battery capacity limited to 100 KWh. Extra charge stored into the battery add up into the energy cost, and affects the overall energy cost reduction opportunities.

Figure 10.6 shows the behavior of SmartStore-E2 with $B_{max} = 60KWh$. SmartStore-E2 adjusted the threshold value after considering the deviation between actual and forecasted energy demands. Readjustment reduces the extra accumulation of charge in the battery, which in turn helps reducing energy cost.

10.5 Observations from SmartStore

- Reliable energy demand forecast is critical for SmartStore.
- With unreliable energy demand forecast, SmartStore with zero battery size performs better than SmartStore with given non-zero battery size.
- With reliable energy forecast, the savings offered by the SmartStore algorithm are proportional to the difference between peak-demand and non-peak demand, if non-peak region is placed before peak region.
- With given battery size, savings offered by SmartStore-E2 are more than or equal to SmartStore-E1 and SmartStore.
- Percentage energy cost reduction offered by SmartStore and its extensions is little, and energy storage cost may defeat the possible savings.

10.6 Summary

In this chapter we studied the behavior of SmartStore and its extensions with real datasets from KRESIT and TATA POWER. We found an anomalous behavior of SmartStore algorithm with varying battery sizes.

Chapter 11

Conclusion and Future Work

11.1 Conclusion

In this report, we studied the energy usage forecast for office buildings. We used *Support Vector Regression* statistical model for energy usage prediction. We used online data such as room-scheduler and event calender to forecast energy usage, and found out online resources assist in forecasting. We studied the impact of weather parameters on energy usage forecast. We performed attribute subset selection and removed irrelevant and redundant attributes, to improve the prediction accuracy.

We analyzed SmartStore algorithm under demand based non-linear pricing policy. We studied the problem of unnecessary energy accumulation in the battery in presence of inaccurate energy demand forecast. We proposed an extensions to SmartStore algorithm to overcome the problems encountered and analyzed it. We compared the SmartStore extensions with original SmartStore algorithm and proved that the misprediction penalty for SmartStore extensions will always be less than or equal to the misprediction penalty for original SmartStore algorithm. We studied the behavior of SmartStore and its extensions with real datasets. We found an anomalous behavior of SmartStore algorithm with varying battery sizes.

11.2 Future Work

Immediate Future Work

- In the Event of underprediction, extend the SmartStore to readjust the thresholds so as to minimize misprediction penalty.

Midterm Future Work

- In the event of underprediction, if cumulative battery capacity reaches to zero, then scrap the precomputed thresholds and restart algorithm. Compare the performance of this technique with SmartStore-E2.

Longterm Future Work

- Change the loss function in the SVR prediction framework, to create costume need based prediction models.
- Build prediction framework to do the incremental updated to earlier forcasted values.
- Given a N days forecasted energy demands, come up with an algorithm to tell from which slot should start to apply cost saving algorithm so that overall gain is maximum.

11.3 Acknowledgement

I would like to thank Prof. Krithi Ramamrithm and Prof. Sunita Sarawagi for their constant support and guidance.

I would also like to thank Rushiraj Chavan and Pratik Patre for their help.

Bibliography

A. Bar-Noy, M. Johnson, and O. Liu. Peak shaving through resource buffering. *Approximation and Online Algorithms*, pages 147–159, 2009.

L.J. Cao and F.E.H. Tay. Support vector machine with adaptive parameters in financial time series forecasting. *Neural Networks, IEEE Transactions on*, 14(6):1506–1518, 2003.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

B.J. Chen, M.W. Chang, et al. Load forecasting using support vector machines: A study on eunite competition 2001. *Power Systems, IEEE Transactions on*, 19(4):1821–1830, 2004.

J. Contreras, R. Espinola, F.J. Nogales, and A.J. Conejo. Arima models to predict next-day electricity prices. *Power Systems, IEEE Transactions on*, 18(3):1014–1020, 2003.

B. Dong, C. Cao, and S.E. Lee. Applying support vector machines to predict building energy consumption in tropical region. *Energy and Buildings*, 37(5):545–553, 2005.

P. Mandal, T. Senjyu, N. Urasaki, T. Funabashi, and A.K. Srivastava. A novel approach to forecast electricity price for pjm using neural network and similar days method. *Power Systems, IEEE Transactions on*, 22(4):2058–2065, 2007.

F.J. Nogales, J. Contreras, A.J. Conejo, and R. Espínola. Forecasting next-day electricity prices by time series models. *Power Systems, IEEE Transactions on*, 17(2):342–348, 2002.

T. Olofsson and S. Andersson. Long-term energy demand predictions based on short-term measured data. *Energy and buildings*, 33(2):85–91, 2001.

Rajeev Gupta Rushiraj Chavan, Krithi Ramamritham. Smart buildings : Electricity cost minimization using energy storage devices. Technical Report IITB-CSE-2013, Department of Computer Science, IIT-Bombay University, Powai, Mumbai, June 2013.

Z. Yu, F. Haghighat, B. Fung, and H. Yoshino. A decision tree method for building energy demand modeling. *Energy and Buildings*, 42(10):1637–1646, 2010.