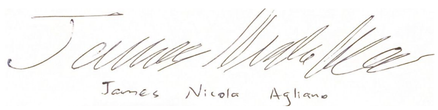
CMSC 409 Artificial Intelligence
Project 3
Due Tuesday, October 29

Student certification

Team member 1:
Print Name: Nick Agliano
Date: 10/28/2019
I have contributed by doing the following:

-

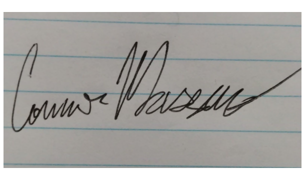
Signed:



Team member 2:
Print Name: Connor Massaro
Date: 10/28/2019
I have contributed by doing the following:

- Research, assistance with coding, assistance with creating document

Signed: Connor Massaro



1) What are the inputs and outputs for this neuron (physical meaning)?

The input to the neuron is a time of day (represented mathematically by an input value, x) and a bias, which can be thought of as a base energy consumption. The output is a kW value. The time of day is passed to a function (in this case, a summation element which can be linear, quadratic, or cubic) which outputs a prediction for the energy consumption at that time.

2) Which activation function is used in the three architectures above? Why?

A linear activation function is used which makes the architecture more powerful than the perceptron learning rule because we can find a gradient which can be minimized in order to reduce error.

3) Compare the training and testing total error obtained using the architectures on Figure 1:

Architecture	**Training A Sum-Square Error**	**Training B Sum-Square Error**	**Training C Sum-Square Error**	**Average Training Sum-Square Error**	**Testing Sum-Square Error**
Architecture A	7.693	7.800	7.883	7.792	15.777
Architecture B	5.627	5.190	5.569	5.462	12.456
Architecture C	0.315	0.420	0.259	0.331	1.234

Architecture A's average sum-square error while training was 7.792, Architecture B's average sum-square error while training was 5.462, and Architecture C's average sum-square error while training was 0.331. We see a difference of 2.33 when comparing Architecture A to Architecture B, which shows that the additional input to Architecture B improved its accuracy. This improvement is reflected during testing with Architecture B achieving 12.456 average sum-square error, and Architecture A performing worse with a 15.777 average sum-square error.

Architecture C outperforms Architecture A and B by far. Architecture C's average sum-square error while training was 0.331, which was a much more significant improvement compared to the difference between Architectures A and B. The testing sum-square error for Architecture C saw a similar improvement. This enormous jump in accuracy shows the importance of choosing an architecture which suits your dataset. Since the dataset is a cubic shape, Architecture C is able to better fit the data, much more accurately than an optimal quadratic or linear shape ever could.

Graphs for questions 3a and 3b are found at the end of this document. For Architecture C we show some of our experimentation with different learning parameters as well as graphs of our error for each iteration -- but for the sake of space only put this information for Architecture C and left it out for A and B.

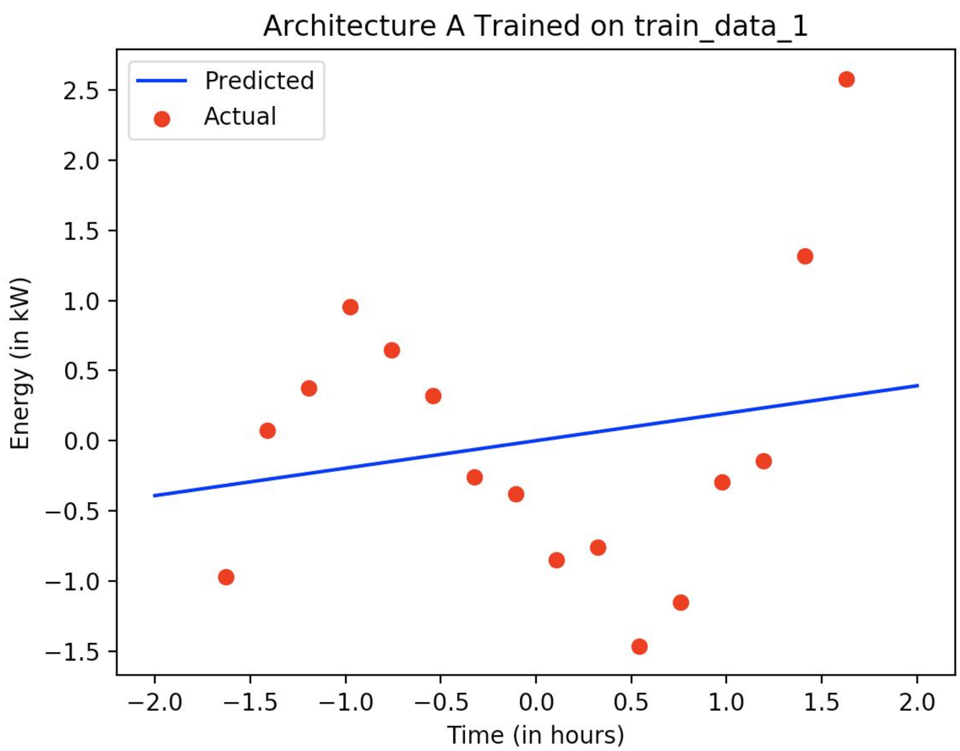
4) Report the number of iterations, the learning rate, and data pre-processing steps you may have chosen (such as normalization of input data). Clearly explain why you selected these values and steps.

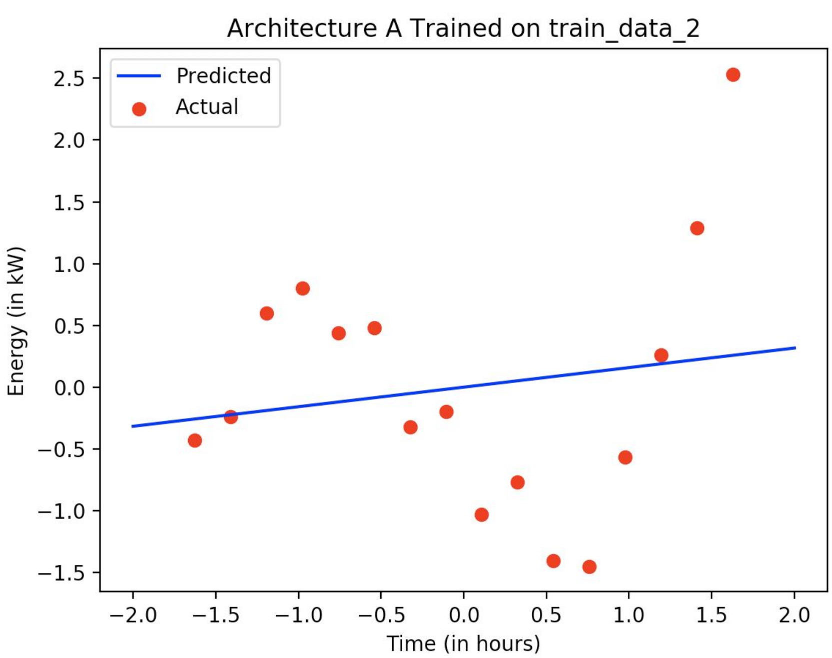
Preprocessing: We scaled our data, which made it much easier to test for the best learning rate, helped lead to faster convergence, and promoted 'stable weights'. Before scaling, we were forced to use a very small learning rate, had trouble finding an optimal value for the learning rate, and our weights would often become very large or very small in magnitude (unstable).

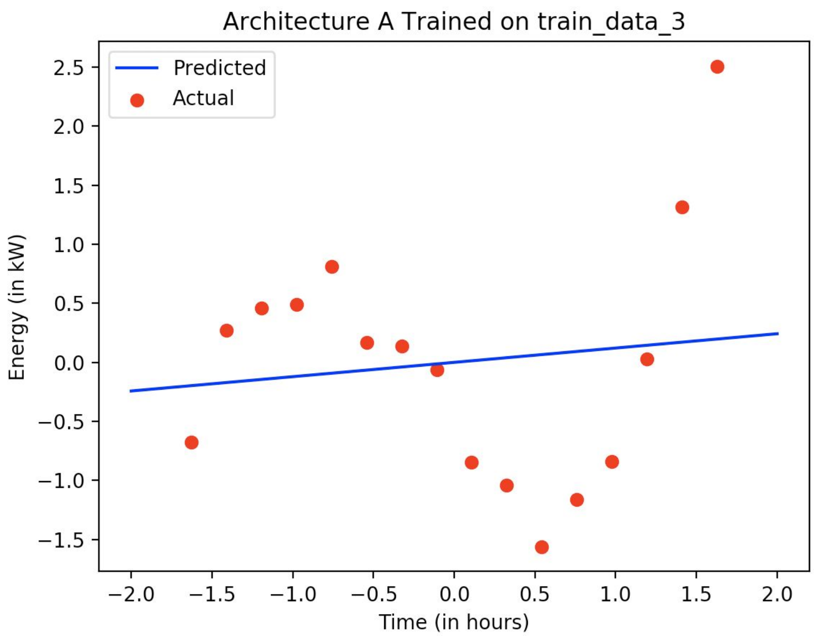
Learning Rate: Our learning rate is set to 0.001 for all three architectures. This learning rate was big enough to avoid local minima, but small enough to converge at an optimal value fairly quickly.

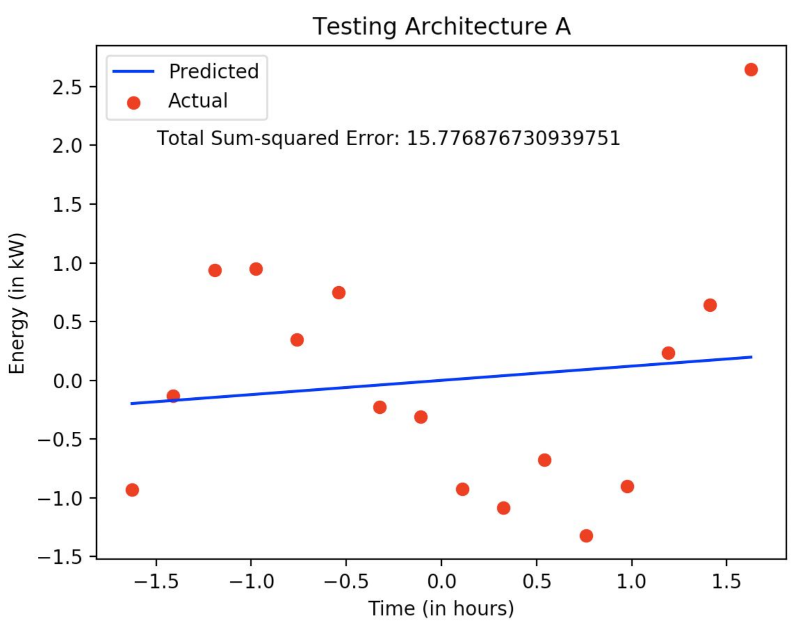
Iterations: Our iterations for all three architectures are set to 1000. At this number of iterations we are able to converge at an optimal value, and the reduction in error is no longer significant.

Architecture A (1 inputs + bias):



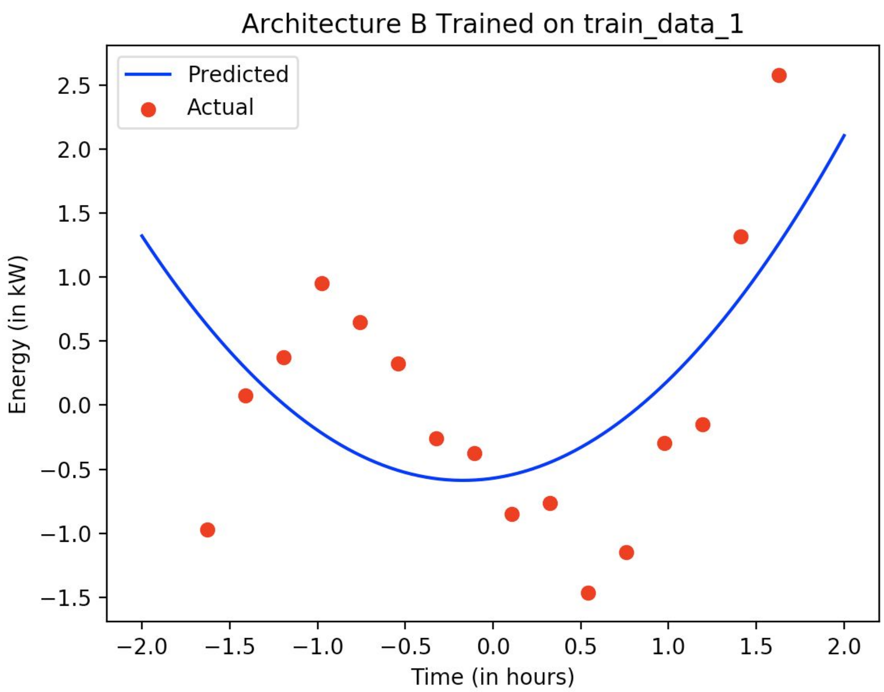


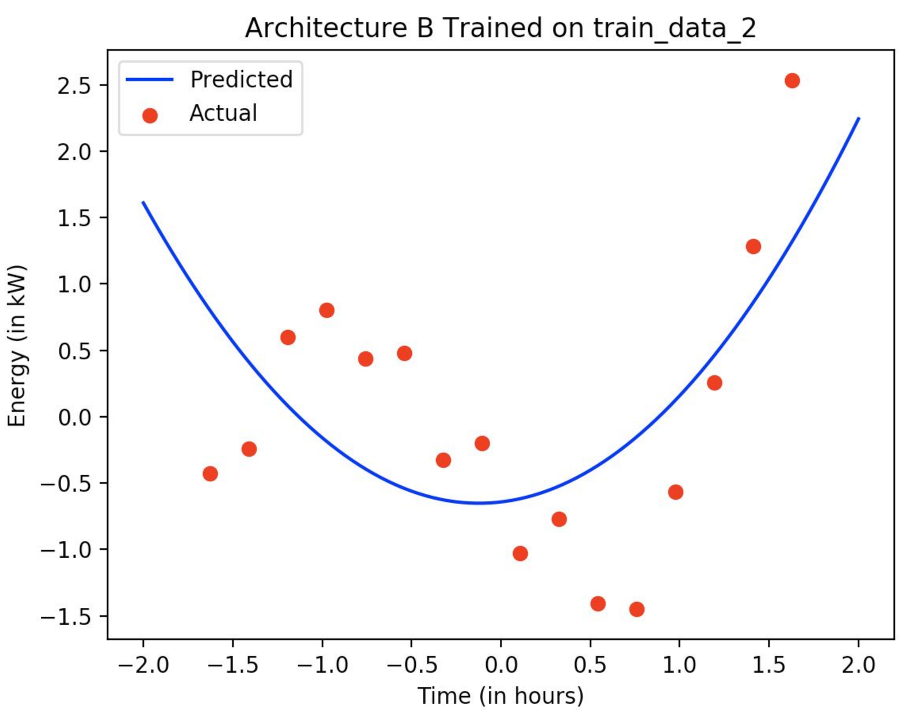


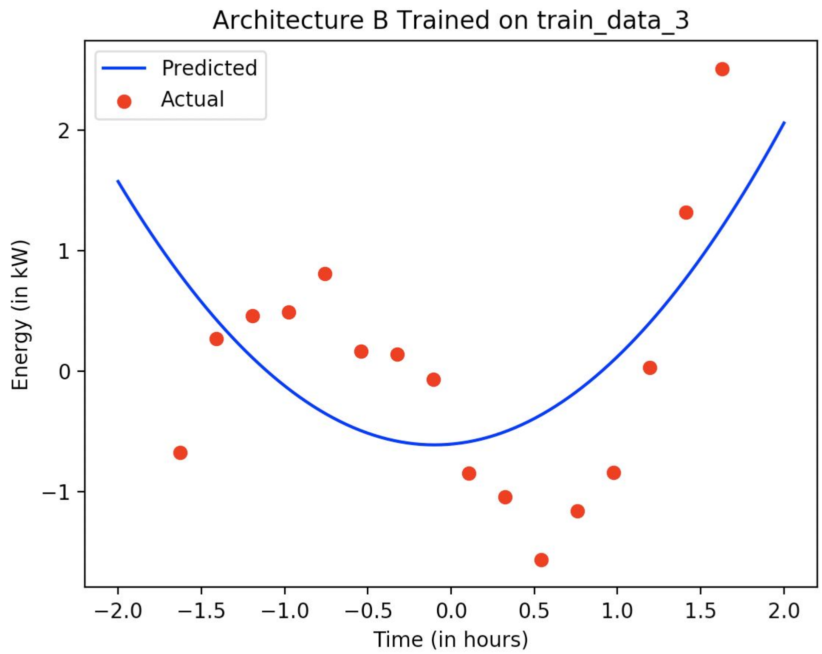


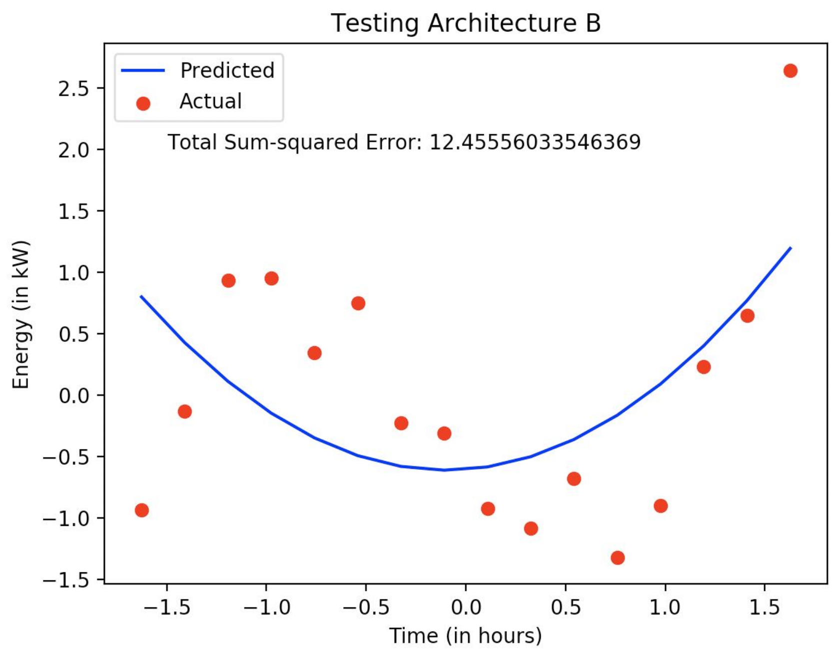
Architecture A Trained on train_data_1: 7.693254804889715
Architecture A Trained on train_data_2: 7.799422610550165
Architecture A Trained on train_data_3: 7.882546187279095

Architecture B (2 inputs + bias):



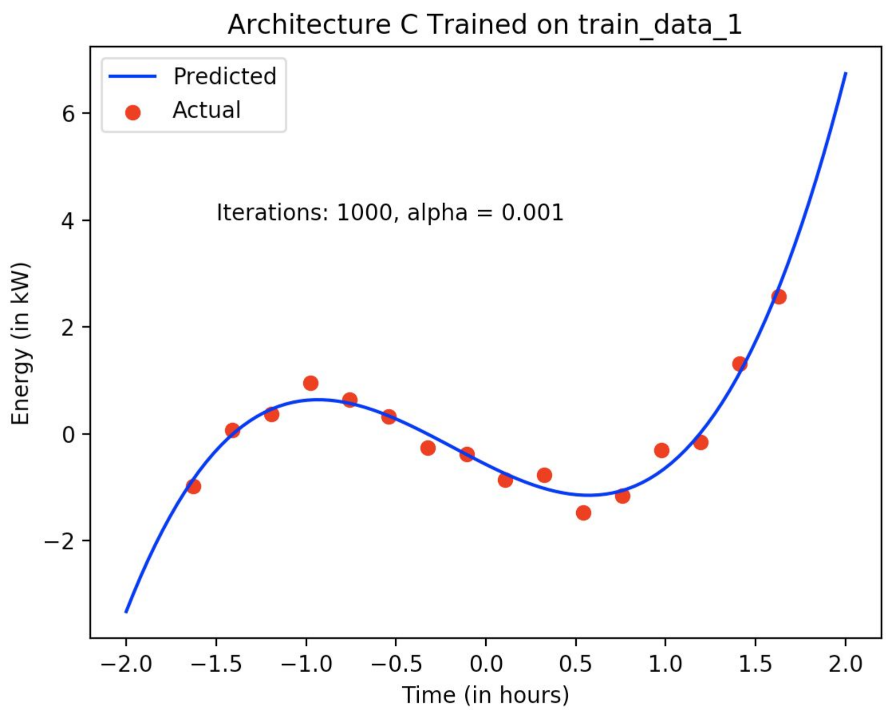


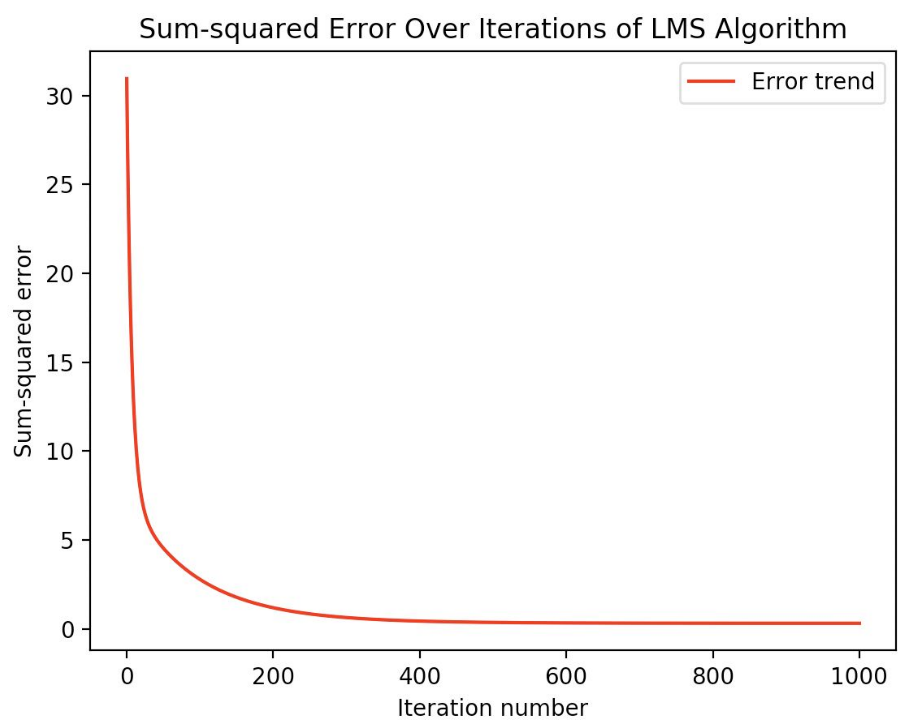


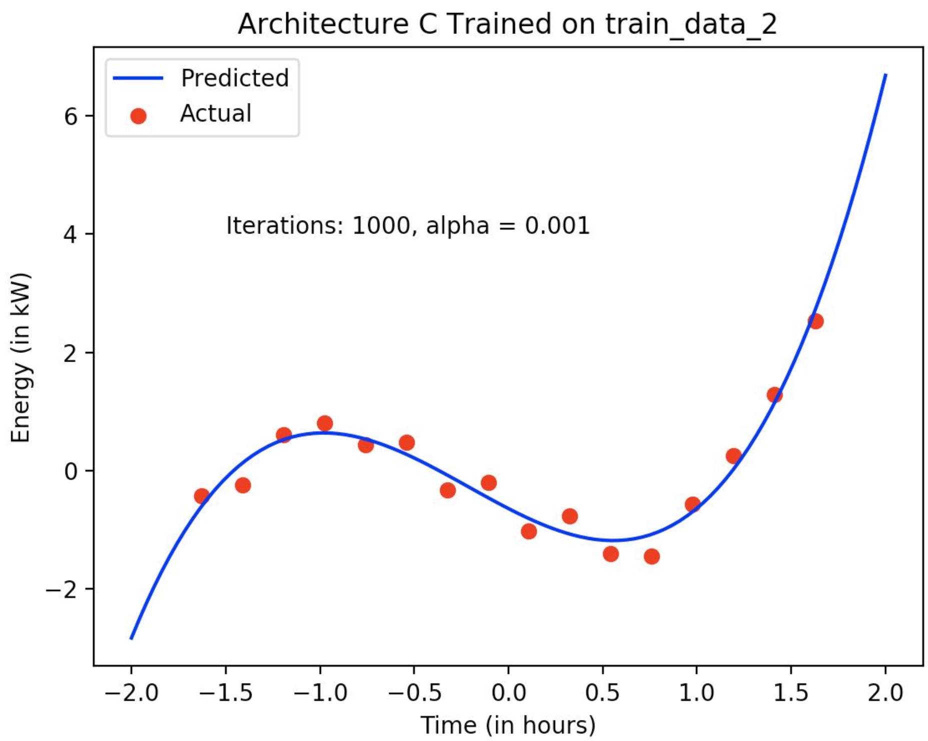


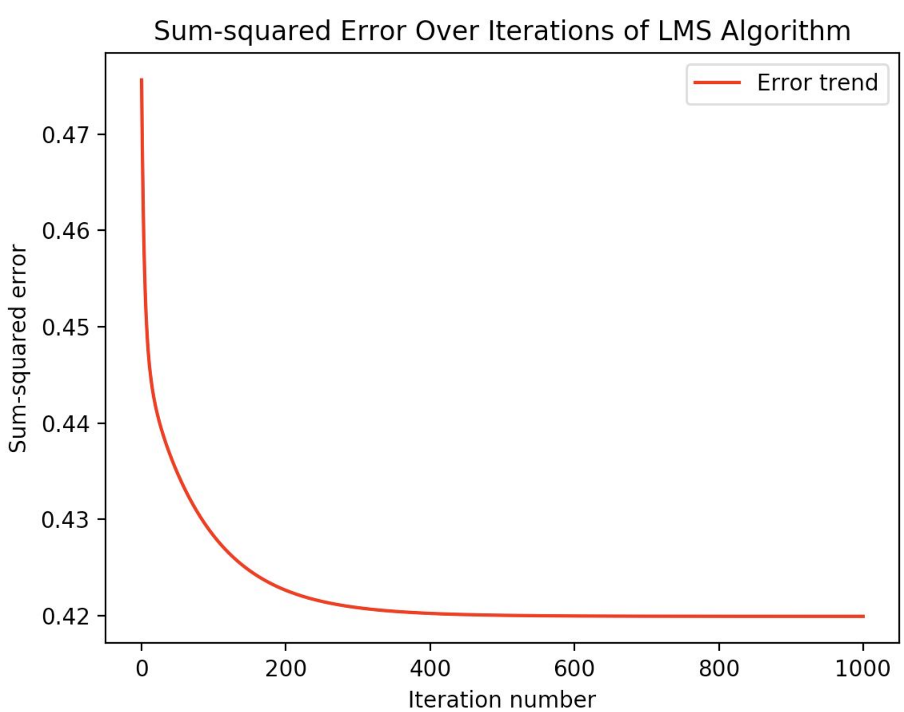
Architecture B Trained on train_data_1: 5.626560413638606
Architecture B Trained on train_data_2: 5.189790122447909
Architecture B Trained on train_data_3: 5.5694788184372905

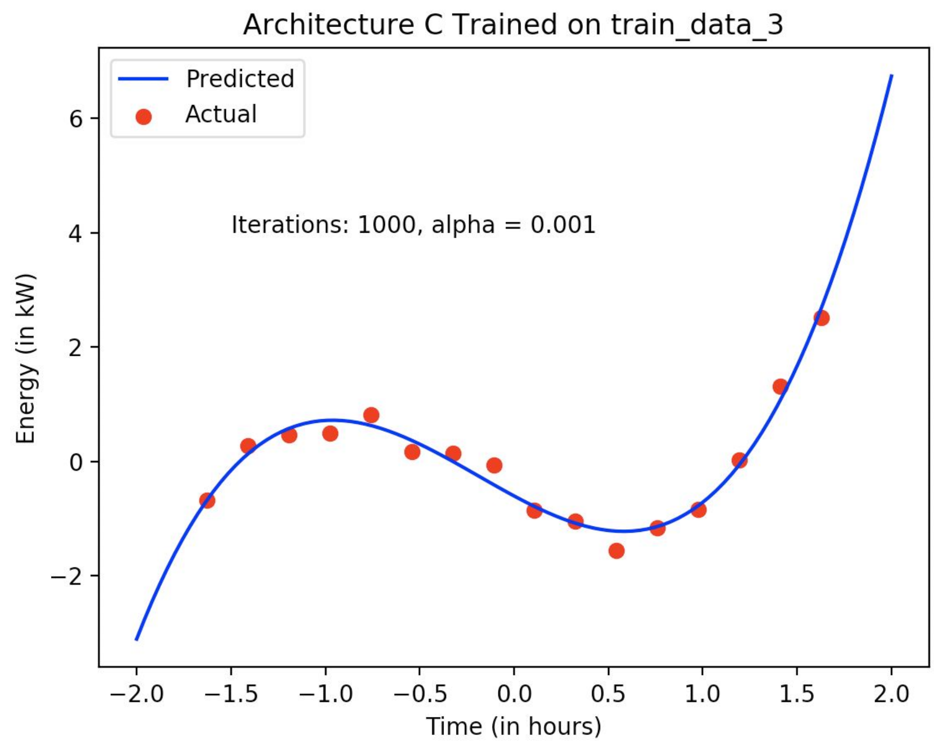
Architecture C (3 inputs + bias):

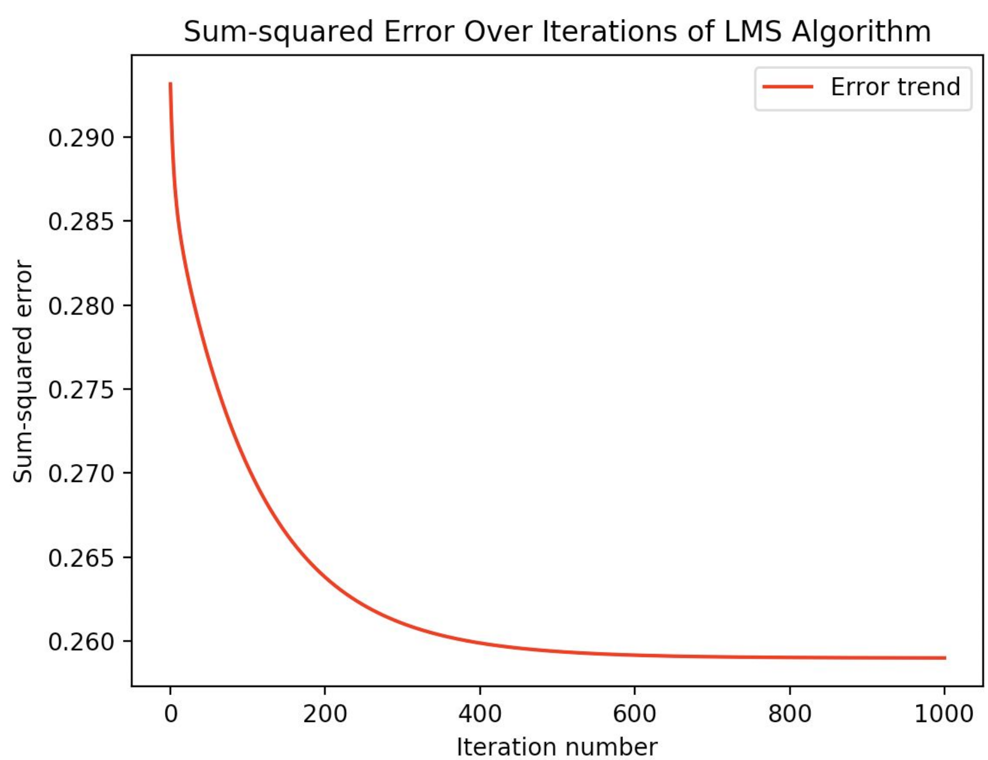


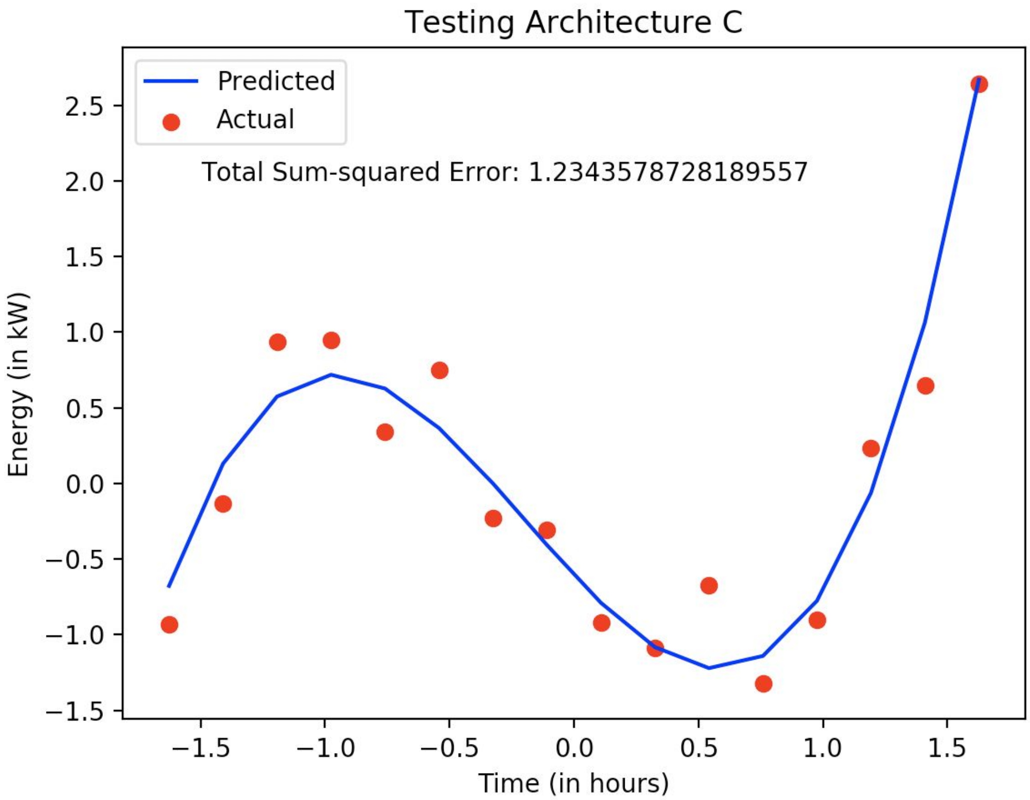



Architecture C Trained on train_data_2
Predicted
Actual
Iterations: 1000, alpha = 0.001
Energy (in kW)
Time (in hours)
6
4
2
0
−2
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0


Sum-squared Error Over Iterations of LMS Algorithm
Error trend
Sum-squared error
Iteration number
0.47
0.46
0.45
0.44
0.43
0.42
0
200
400
600
800
1000


Architecture C Trained on train_data_3
Predicted
Actual
Iterations: 1000, alpha = 0.001
Energy (in kW)
Time (in hours)
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
6
4
2
0
−2


Sum-squared Error Over Iterations of LMS Algorithm
Error trend
Sum-squared error
Iteration number
0.290
0.285
0.280
0.275
0.270
0.265
0.260
0
200
400
600
800
1000



Architecture C run with different parameters:

a = .001
Iterations = 1000
train_data_1: 0.3151432561382458
train_data_2: 0.4199510431685862
train_data_3: 0.2589978200051168
Testing dataset Sum-squared error = 1.234358000116091

a = .0001
Iterations = 1000
Sum-squared error = 2.258711797783459

a = .0001
Iterations = 5000
Sum-squared error = 1.2433636047740817

a = .001

Iterations = 5000
Sum-squared error = 1.2335562028489229
