

# Predicting power load

Shajahan Abdul Rehman Basith , Habeebulla Shaik , Chandrakanth seth

## Introduction:

To predict the power load for Puget Sound Power & Light Co. 24 hours in advance. We have used multiple models for this task like Linear ridge regression model in which we used cross-validation technique to select the best features to improve our accuracy. Other models that we used were Nearest neighbor, MLP.

The Data: For this we have the observation for the period January 1985- October 1990. Some of the data was withheld and we were only given the input data our job is to predict the output for those months. We have been provided 15 features for this task.

## Methodology:

We have used MATLAB as the tool to solve the problem. To get acquainted with the data we performed linear regression using the 'Fitlm' in MATLAB. We obtained a plot, which is later shown in the next section.

To calculate the covariance, first we find out the mean value of the inputs and the outputs. Then we find out the mean for the difference between the inputs and the mean of inputs and the mean for the difference between the outputs and the mean of outputs. Now, the product of these two values is the covariance.

The generalization performance has been estimated using knn and it is also pruned by removing some of the features for decreasing dimensionality and for a simple model.

In this part we create, two Multi-Layer Perceptions. To construct an MLP, we have used the Deep Learning Toolbox to create the neural network. For this we considered only a set of variables from the total variables in the training data. We construct two models by considering two set of variables. While training the neural network we have used the Bayesian Regularization Algorithm. The reason for choosing this algorithm is that, though it takes more time to train the data, but it can result in good generalization for difficult data sets. Later after constructing these two models, we also construct another neural network by considering only the important variables from the stepwise regression. Again, Bayesian regularization is used, and the neural network is constructed. All the neural networks are constructed by the 'nntool' function. The corresponding errors of each of these is shown in the next section.

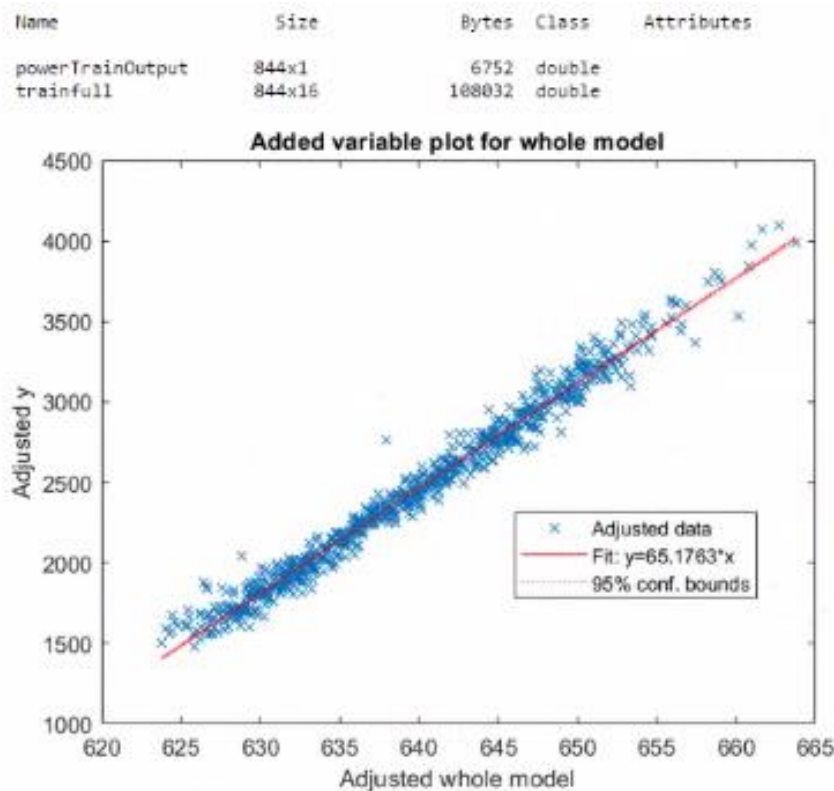
## Data and Results:

The data supplied has four sets of data.

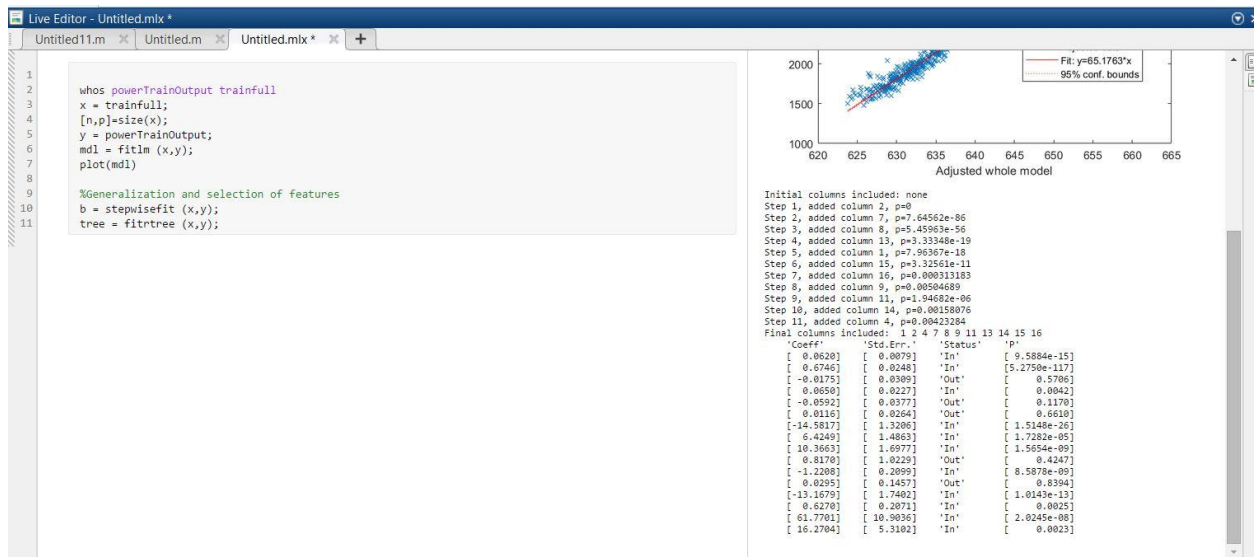
- i) powerTrainInput – This the training data which has 15 features based on which we can train our model. It contains all the variables to the process.

- ii) powerTrainOutput – This data set contains the responses which we use to train the model.
- iii) powerTestInput – This data set contains the unknown data, which the variables of the process apart from the training data.
- iv) powerTrainDate- This data set contains the date for the training samples

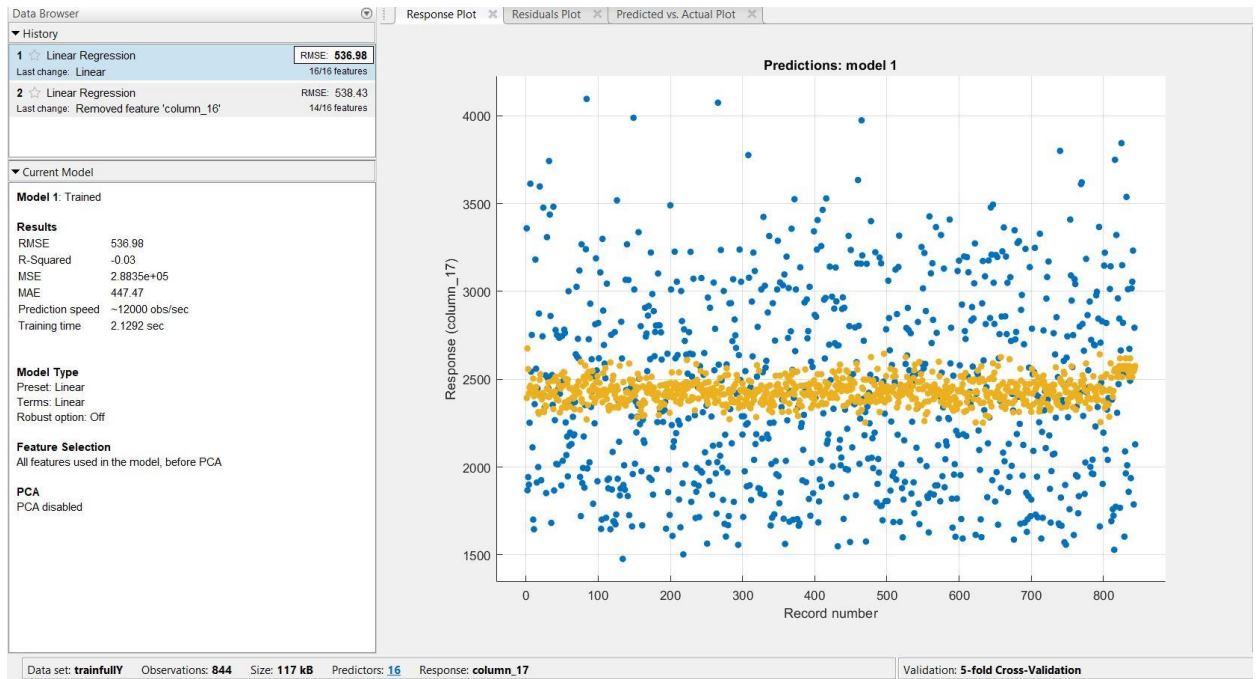
## 1. Relationships between input and output (Scatter Plot)



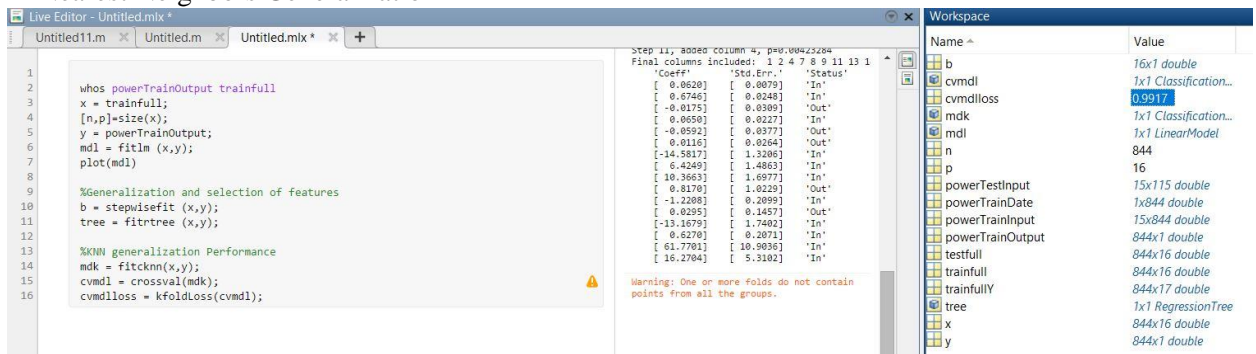
## 2. Generalization and selection of features



### 3. Linear regression



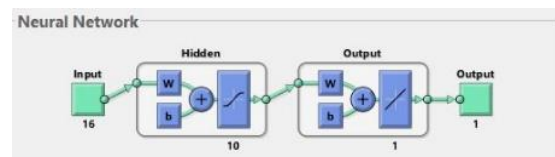
### 4. K-Nearest Neighbors Generalization



### 5. MLP Generalization Error







Inputs: trainfull  
Targets: powerTrainOutput

Training: 590 Samples(70%)  
Validation: 127 Samples(15%)  
Testing: 127 Samples(15%)



Number of Hidden Neurons: 10 (default)

Trained Neural Network using algorithm: Bayesian Regularization (for more accuracy)

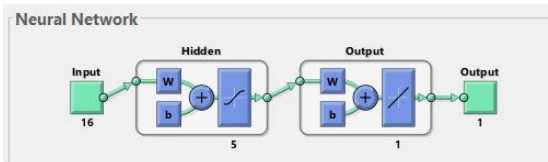
Results			
	 Samples	 MSE	 R
 Training:	590	1700.01343e-0	9.96963e-1
 Validation:	127	0.00000e-0	0.00000e-0
 Testing:	127	3024.93911e-0	9.94558e-1

## 6. Different Multilayer perceptron's and estimating the generalization performance







- NNtool = 5 networks created
- NNtool = 5 networks created

### ➤ MLP using NNtool

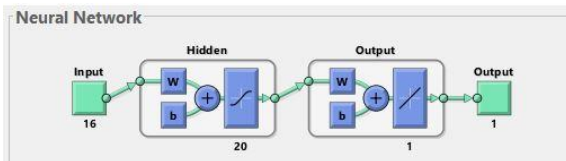
- ✓ Number of Hidden Neurons: 5









### Generalization Error

Results			
	 Samples	 MSE	 R
 Training:	590	2130.63893e-0	9.96194e-1
 Validation:	127	0.00000e-0	0.00000e-0
 Testing:	127	3851.40290e-0	9.93074e-1

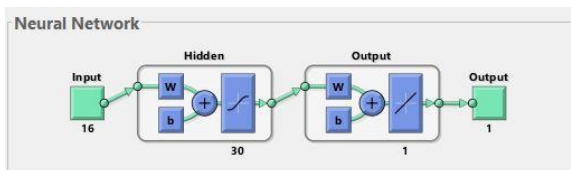
- ✓ Number of Hidden Neurons: 20









### Generalization Error

Results			
	 Samples	 MSE	 R
 Training:	590	1633.26252e-0	9.97020e-1
 Validation:	127	0.00000e-0	0.00000e-0
 Testing:	127	5319.82936e-0	9.91620e-1

- ✓ Number of Hidden Neurons: 30

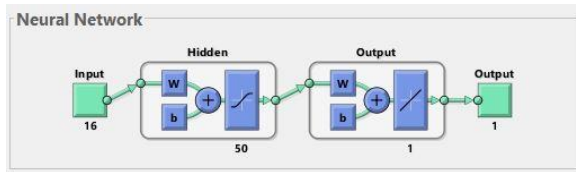


### Generalization Error

Results			
	 Samples	 MSE	 R
 Training:	590	197.49587e-0	9.99644e-1
 Validation:	127	0.00000e-0	0.00000e-0
 Testing:	127	10449.01009e-0	9.82290e-1

- ✓ Number of Hidden Neurons: 50

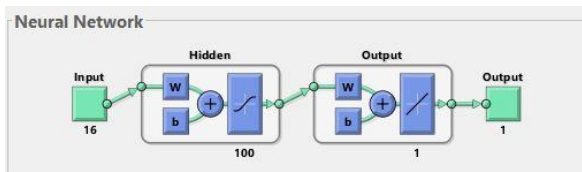
### Generalization Error



Results

	Samples	MSE	R
Training:	590	137.39077e-0	9.99757e-1
Validation:	127	0.00000e-0	0.00000e-0
Testing:	127	11309.29692e-0	9.81279e-1

✓ Number of Hidden Neurons: 100



Results

	Samples	MSE	R
Training:	590	752.19210e-0	9.98654e-1
Validation:	127	0.00000e-0	0.00000e-0
Testing:	127	5189.03067e-0	9.91153e-1

➤ MLP Using nntraintool

Neural Network Training (nntraintool)

Neural Network

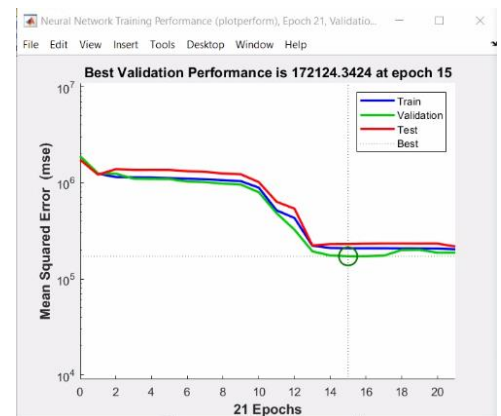
The diagram shows a neural network architecture with an input layer of 15 nodes, two hidden layers each with 10 nodes, and an output layer of 1 node. Each layer is represented by a box containing 'W' (weights) and 'b' (biases), followed by a summation node (+) and an activation function node (represented by a blue box with a curve). The layers are connected sequentially.

Algorithms

Data Division: Random (dividerand)  
 Training: Levenberg-Marquardt (trainlm)  
 Performance: Mean Squared Error (mse)  
 Calculations: MEX

Progress

Epoch: 0 21 iterations 1000  
 Time: 0:00:00  
 Performance: 1.90e+06 2.04e+05 0.00  
 Gradient: 3.44e+06 9.76e+03 1.00e-07  
 Mu: 0.00100 10.0 1.00e+10  
 Validation Checks: 0 6 6



Neural Network Training (nntraintool)

Neural Network

The diagram shows a neural network architecture with an input layer of 15 nodes, two hidden layers (the first with 20 nodes and the second with 15 nodes), and an output layer of 1 node. Each layer is represented by a box containing 'W' (weights) and 'b' (biases), followed by a summation node (+) and an activation function node (represented by a blue box with a curve). The layers are connected sequentially.

Algorithms

Data Division: Random (dividerand)  
 Training: Levenberg-Marquardt (trainlm)  
 Performance: Mean Squared Error (mse)  
 Calculations: MEX

Progress

Epoch: 0 14 iterations 1000  
 Time: 0:00:00  
 Performance: 5.59e+05 627 0.00

Neural Network Training (nntraintool)

Neural Network

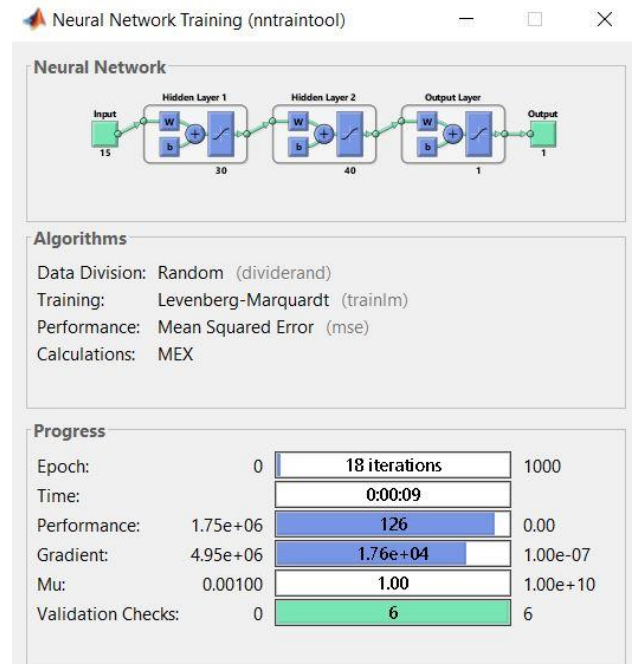
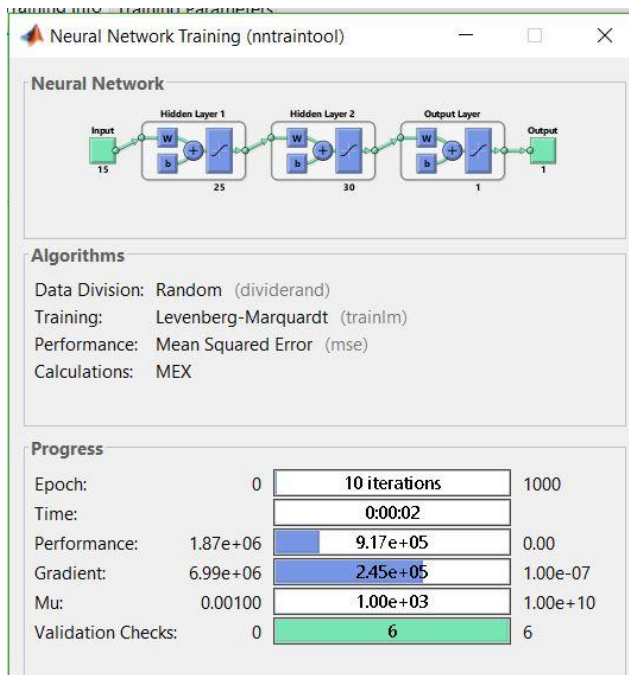
The diagram shows a neural network architecture with an input layer of 15 nodes, two hidden layers (the first with 15 nodes and the second with 25 nodes), and an output layer of 1 node. Each layer is represented by a box containing 'W' (weights) and 'b' (biases), followed by a summation node (+) and an activation function node (represented by a blue box with a curve). The layers are connected sequentially.

Algorithms

Data Division: Random (dividerand)  
 Training: Levenberg-Marquardt (trainlm)  
 Performance: Mean Squared Error (mse)  
 Calculations: MEX

Progress

Epoch: 0 14 iterations 1000  
 Time: 0:00:00  
 Performance: 5.22e+05 686 0.00



**Output:** The best models from the MLP and Linear ridge regression have been trained and tested with the test data and the output have been attached.



linear testdata  
outputs.ods



mlp testdata  
output.ods

**Code:**



regression code.m



2-LINEAR  
REGRESSION CODE.



MLP code.m