VAN-HUNG, LE (黎文雄)

Student ID: 0860831

DEEP LEANING – REPORT HOMEWORK 1

1. Regression

In this task, we need to minimizing the sum-of-squares-error function:

$$E(w) = \sum_{n=1}^{N} (t_n - y(X_n; w))^2$$

Evaluate the performance by root-mean-square error (RMS)

$$E_{RMS}(w) = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (t_n - y(X_n; w))^2}$$

a. Network Architecture

Network Architecture	15-10-10-1
Activation function	[sigmoid, sigmoid, sigmoid]
Training $E_{RMS}(w)$	0.06099833
Testing $E_{RMS}(w)$	0.06167703
Epochs	10000
Learning rate	0.002
Batch Size	32
Training/Testing Size	75%/25%

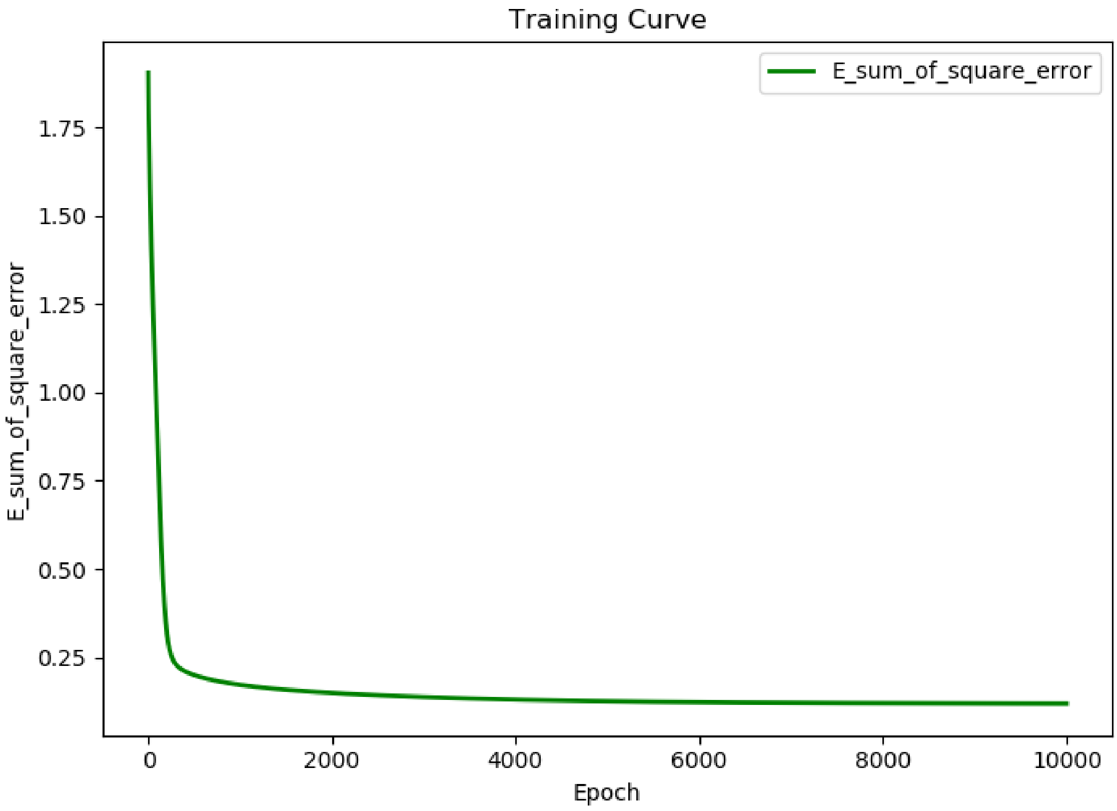
Data processing:

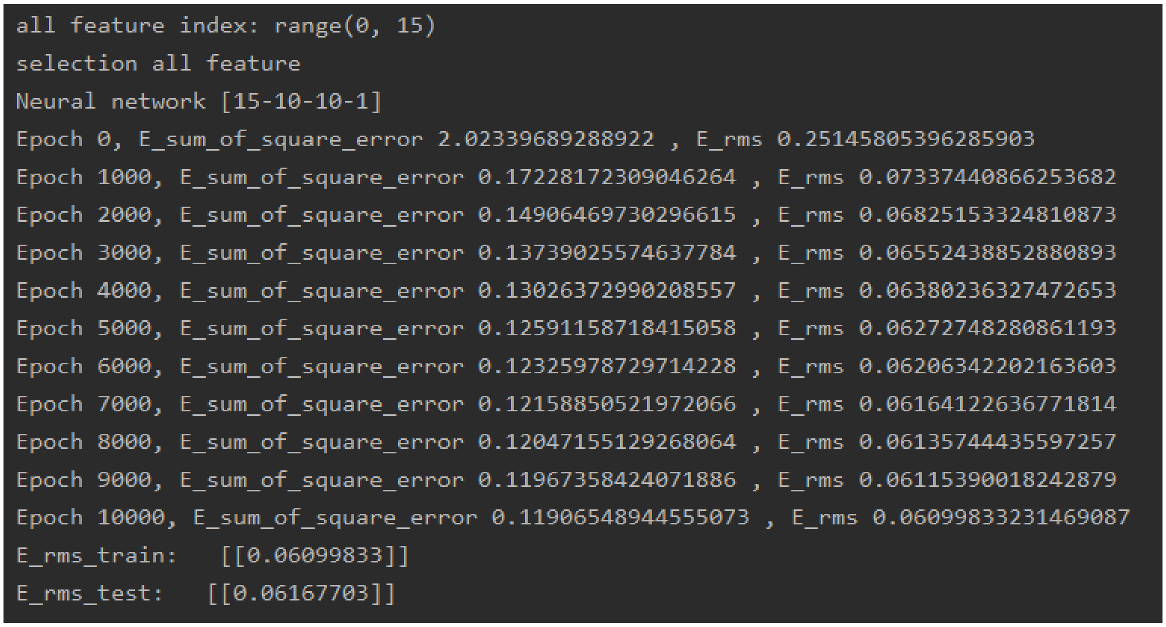
The categorical features (orientation and glazing area distribution) need to encode them into one-hot vectors.

The other features and output data are normalized to be from 0 to 1

b. Learning Curve

Plot graph $E(\mathrm{w})$ of each bath-size, it will be reduced by each epoch

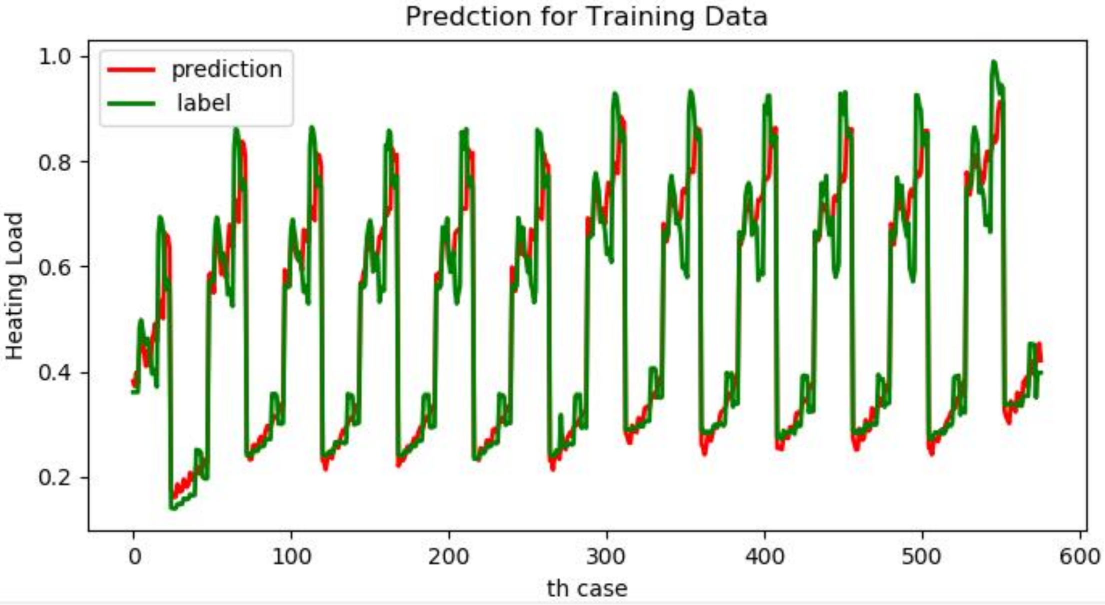




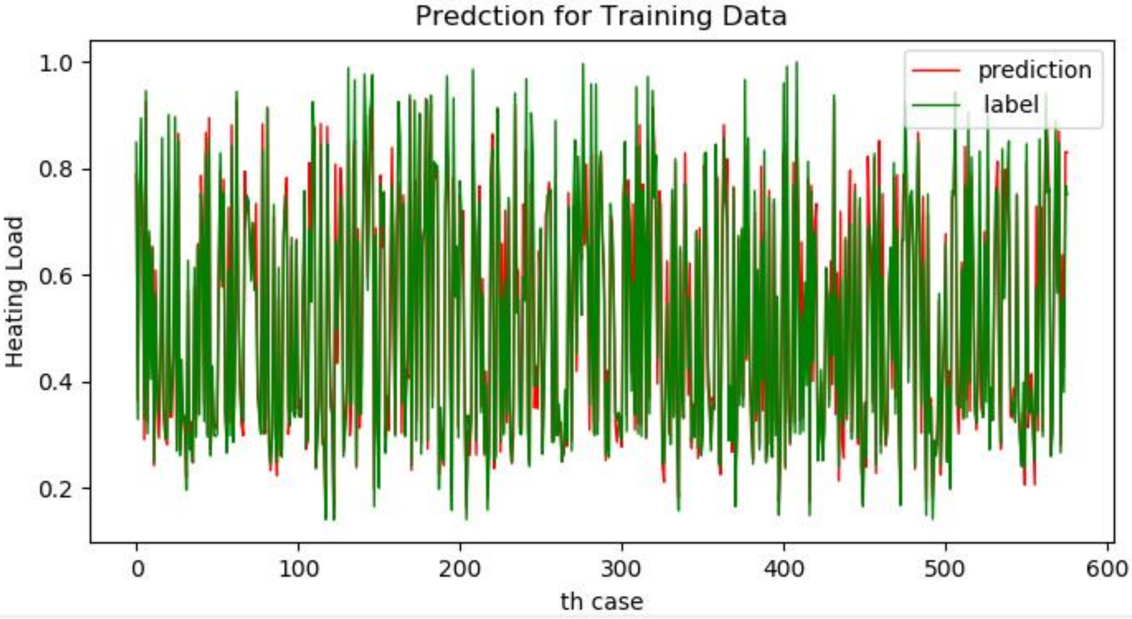
```
all feature index: range(0, 15)
selection all feature
Neural network [15-10-10-1]
Epoch 0, E_sum_of_square_error 2.02339689288922 , E_rms 0.25145805396285903
Epoch 1000, E_sum_of_square_error 0.17228172309046264 , E_rms 0.07337440866253682
Epoch 2000, E_sum_of_square_error 0.14906469730296615 , E_rms 0.06825153324810873
Epoch 3000, E_sum_of_square_error 0.13739025574637784 , E_rms 0.06552438852880893
Epoch 4000, E_sum_of_square_error 0.13026372990208557 , E_rms 0.06380236327472653
Epoch 5000, E_sum_of_square_error 0.12591158718415058 , E_rms 0.06272748280861193
Epoch 6000, E_sum_of_square_error 0.12325978729714228 , E_rms 0.06206342202163603
Epoch 7000, E_sum_of_square_error 0.12158850521972066 , E_rms 0.06164122636771814
Epoch 8000, E_sum_of_square_error 0.12047155129268064 , E_rms 0.06135744435597257
Epoch 9000, E_sum_of_square_error 0.11967358424071886 , E_rms 0.06115390018242879
Epoch 10000, E_sum_of_square_error 0.11906548944555073 , E_rms 0.06099833231469087
E_rms_train:   [[0.06099833]]
E_rms_test:   [[0.06167703]]
```

c. **Regression result with training label**

- **Shuffle = FALSE**



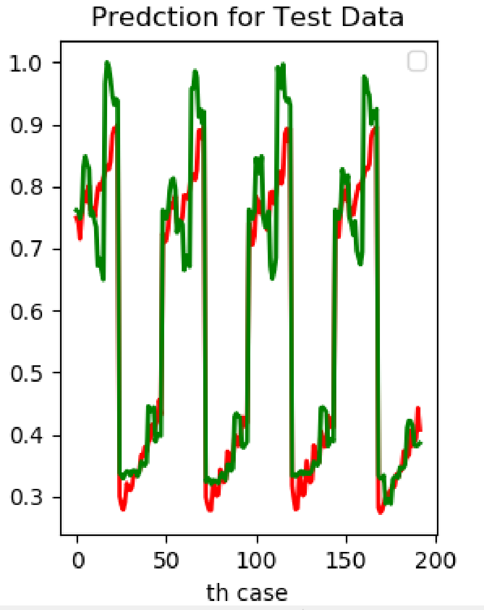
- **Shuffle = TRUE**



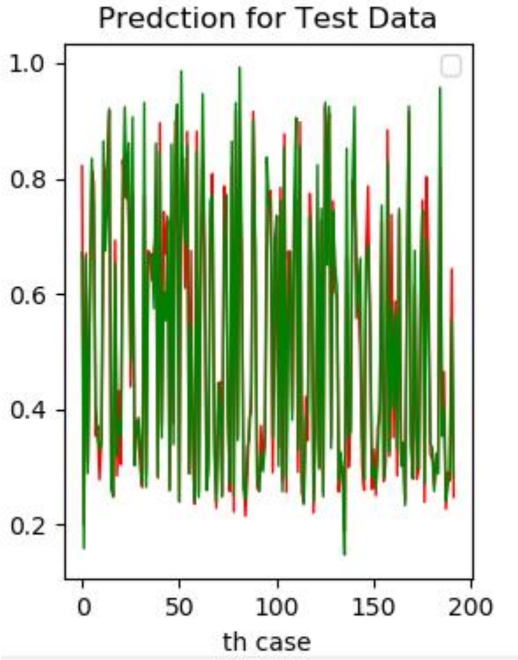
-

d. Regression result with testing label

- **Shuffle = FALSE**



- **Shuffle = TRUE**



-

e. Design a feature selection procedure

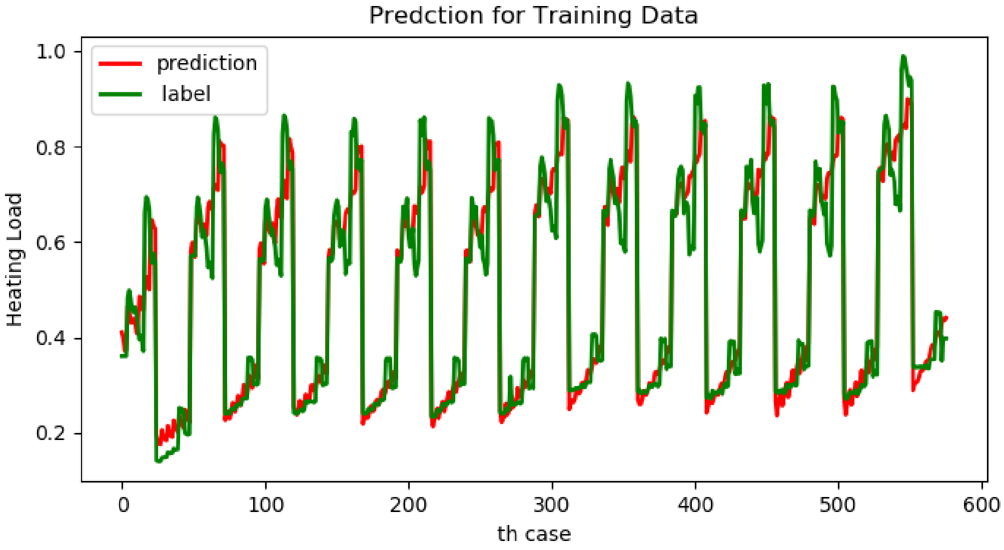
- In this task, I use **variance threshold** method to find out which input features influence the energy load significantly and which features have very little effect on energy load.

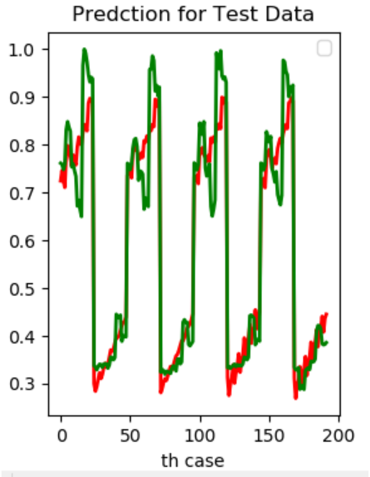
- The idea in here is when a feature doesn't vary much within itself, it generally has very little predictive power.

- Intuitively, we can see that the feature which has the value variance that smaller than a threshold (or the others) will not affect to the result. Similarly, the one with bigger value variance influence result significantly. Now I will remove features to see on it.

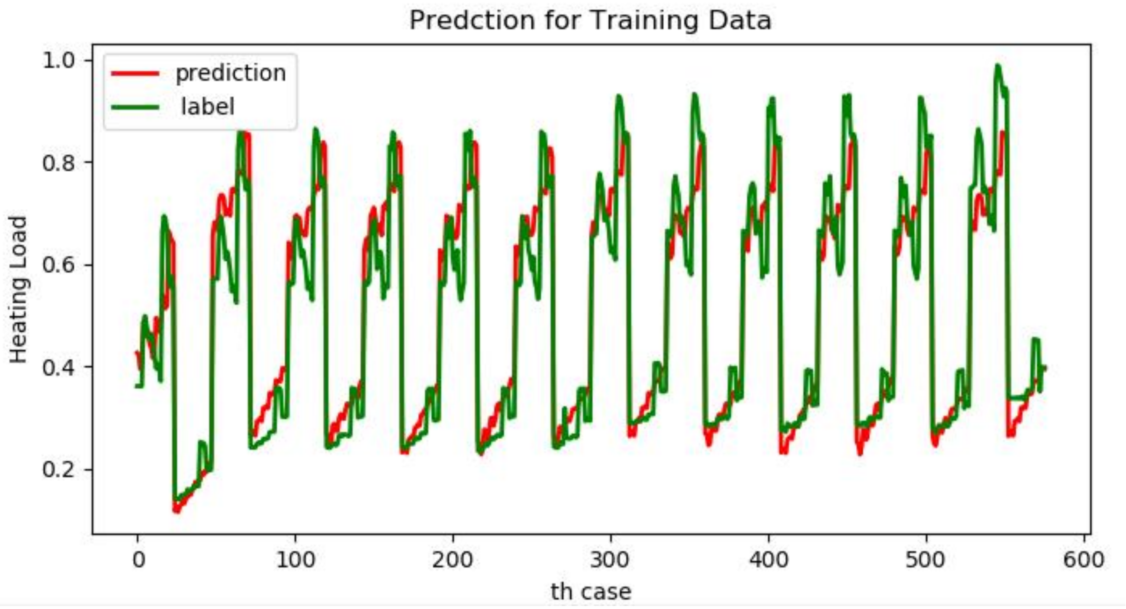
Remove 1 feature:

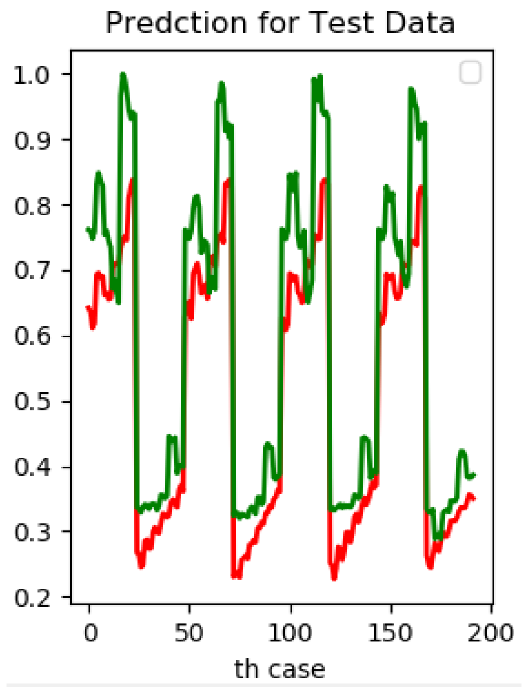
- Remove feature with minimum value variance (feature_index = 2).The result is **NO** different than when using all feature. So I can say that this feature is not important with power prediction.





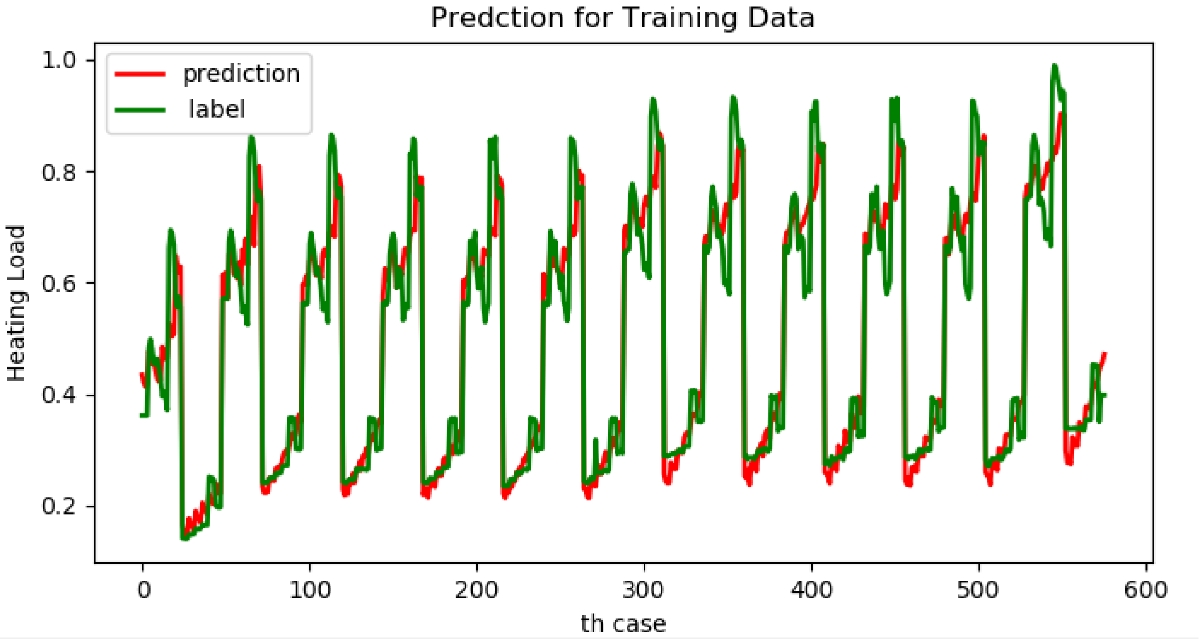
- **Remove feature with maximum value variance** (feature_index = 5). The result is different than when using all feature. So, I can say that this feature is important with power prediction.

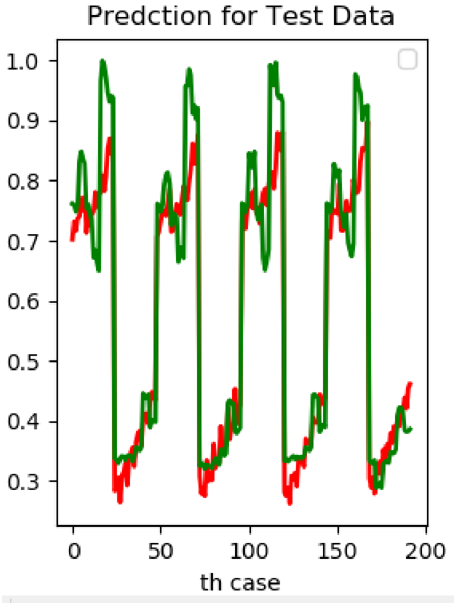




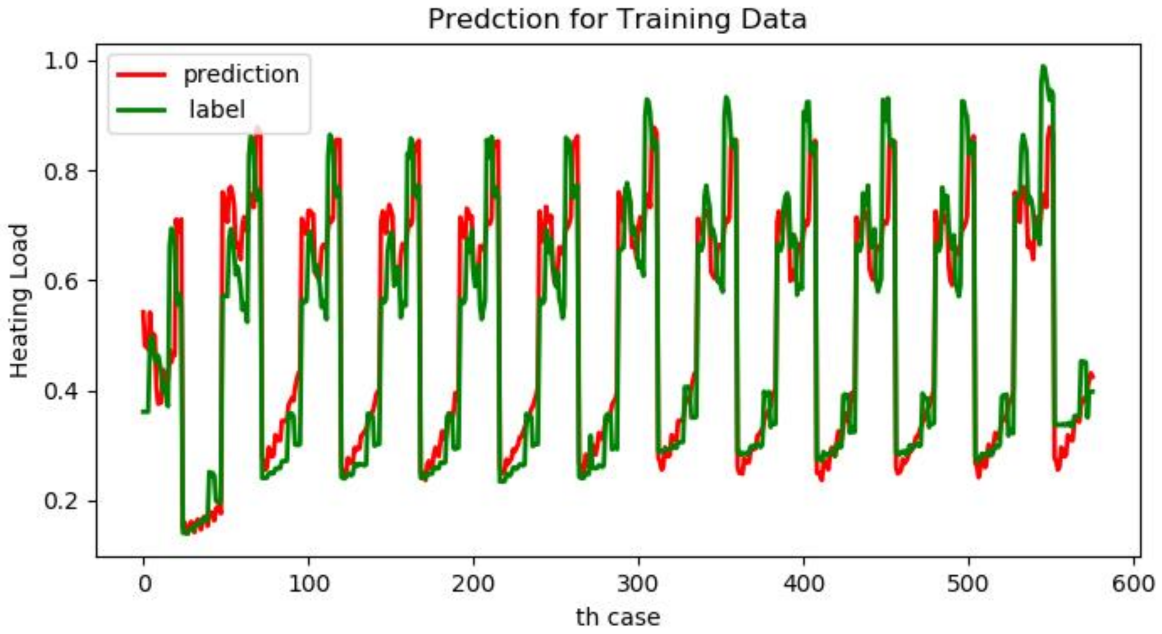
Remove 2 features:

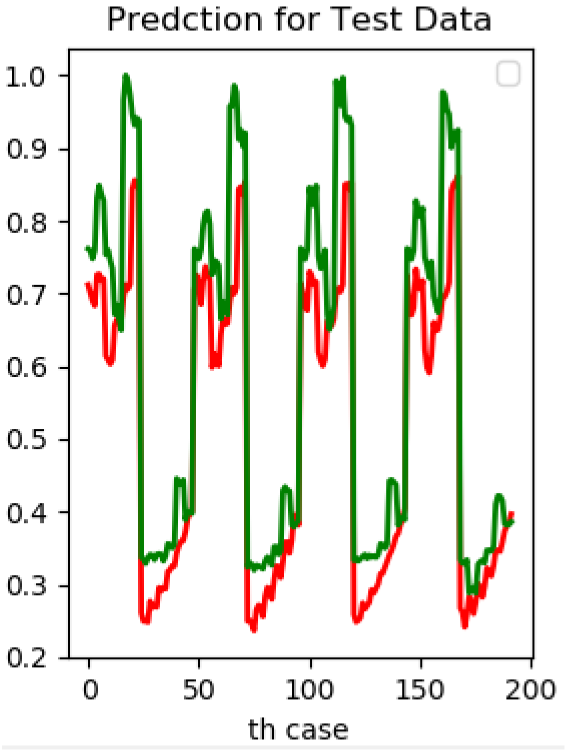
Remove 2 features with minimum value variance (feature_index = [0,2]).The result is verry little different than when using all feature.





Remove 2 features with maximum value variance (feature_index = [4,5]). The result is different than when using all feature.





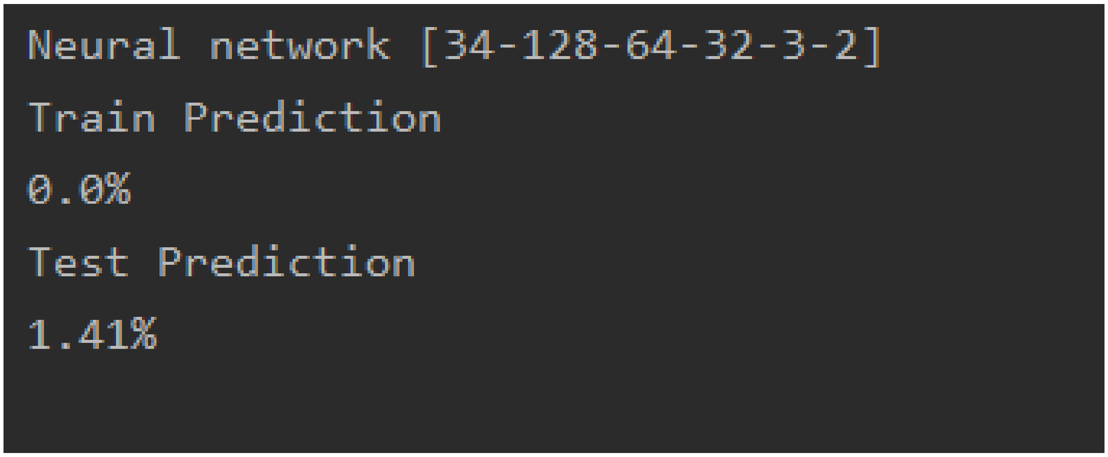
The error per data comparing between training data and testing data:

	Error Training	Error Testing
Remove 1 feature (minimum value variance)	0.06160198	0.06363681
Remove 2 feature (minimum value variance)	0.06262843	0.07005903
Remove 1 feature (maximum value variance)	0.07054658	0.10302955
Remove 2 feature (maximum value variance)	0.07904408	0.10479482

2. Classification

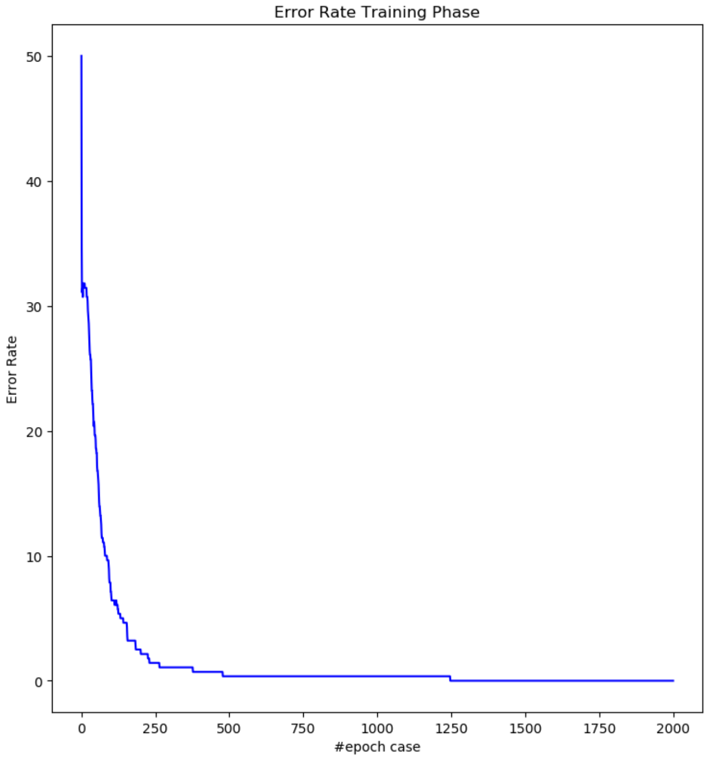
a. Network Architecture

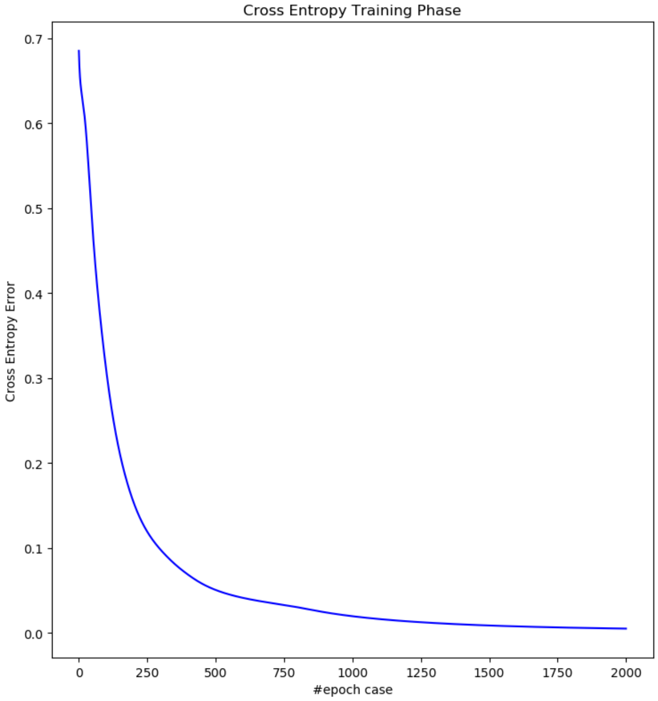
Network Architecture	34-128-64-32-3-2
Error Rate Training	0.0%
Error Rate Testing	1.41%
Epoches	2000
Learning Rate	0.0007
Batch Size	32
Training Size / Testing Size	80% / 20%



```
Neural network [34-128-64-32-3-2]
Train Prediction
0.0%
Test Prediction
1.41%
```

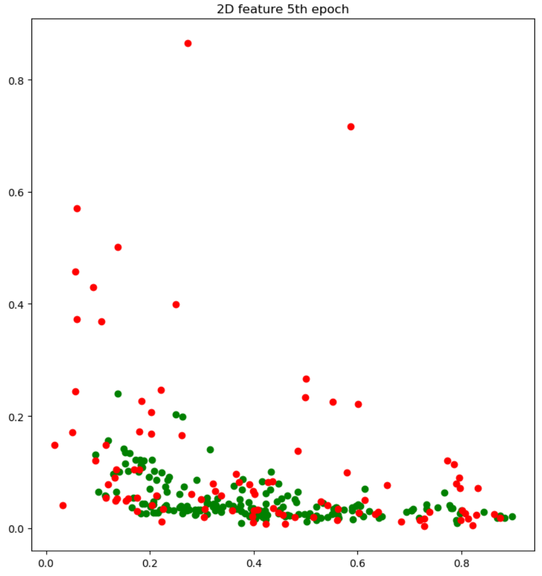
b. Learning Curve

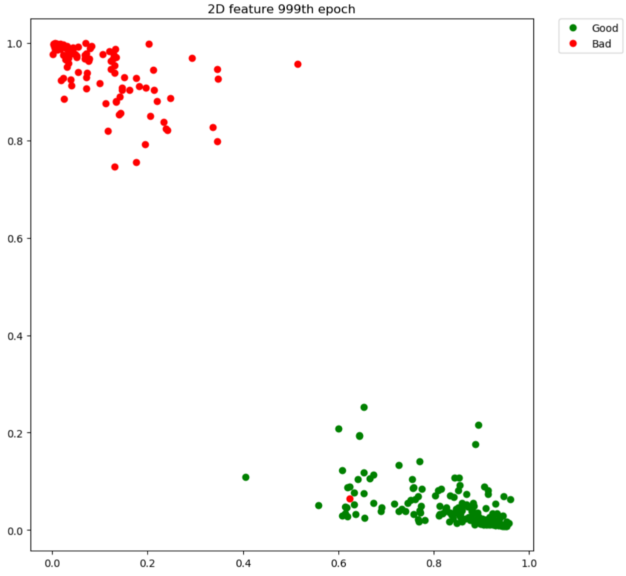


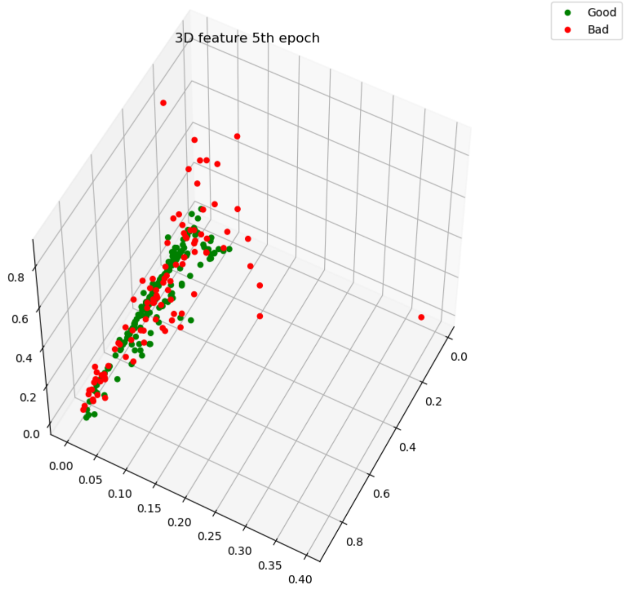


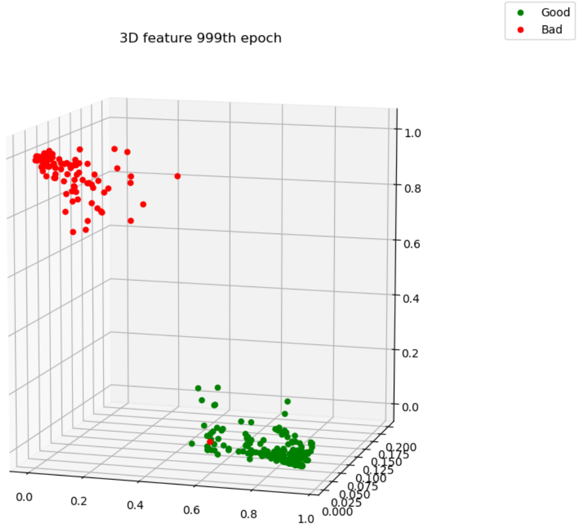
c. Plotting latent features at different training state

By changing the hidden layer before output layer, using 2 or 3 nodes, we then compare the result by visualize (2D or 3D) the distribution of this hidden layer at diffident training stage.









USAGE: - classification.py to run classification

- regression.py to run regression