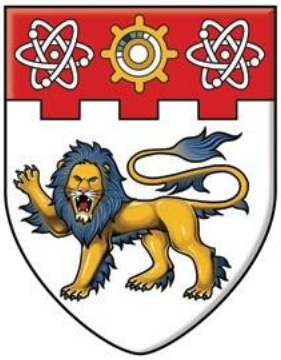

NANYANG TECHNOLOGICAL UNIVERSITY

Assignment 1 Report

CZ4042 Neural Networks

Name: He Yuhao
Matric. Number: U1722945E

Name: Li Guanlong
Matric. Number: U1722033H

Introduction

This project is to apply neural network to address real case problems. We endeavour to get optimal models by turning hyper-parameters and selecting features. For part A, Cardiotocography dataset is a classification problem, and we explore the optimal hyper-parameters including batch-size, number of hidden neurons, decay, and number of hidden layers. For part B, Graduate Admissions Predication is a regression problem, and we draw correlation map and do feature selection, and compare different model complexity to get better and simpler model with stronger generalization ability.

Methods

For both parts, the programming language used in this project is Python, the framework used is TensorFlow.

In Part A, the neuron network is designed for classification problem. The hidden layers implement ReLU activation function, and the output layer implements softmax function. The weight initialized for the training is based on a normal distribution. The standard deviation of the normal distribution is the reciprocal of the number of features (21). Since this is a classification problem, cross entropy is utilized to calculate the loss function for the training. A weight decay parameter is introduced for regularization to reduce the overfitting.

In Part B, the neuron network is designed for regression problem. The hidden layers implement ReLU activation function, followed with He Normal weight initialization. The loss function is mean square error since it is a regression problem. Regularization and dropout are used to prevent model from overfitting. Early stop is also used to get the optimal model when it is converged. Correlation map is used to detect the relationship between features and the label, and it is also a good evidence for feature selection. Recursive feature elimination is the way to remove extra features by checking if the accuracy is still high without the feature removed.

Experiments and Results

Part A: Classification Problem

1. The 3-layer feedforward neural network (FFN) is constructed with a learning rate of α = 0.01, batch size of 32, hidden layer neuron number of 10, and weight decay parameter of β = 10^{-6}:

a) The accuracies on both training and testing data against epochs are shown in the Figure 1 below:

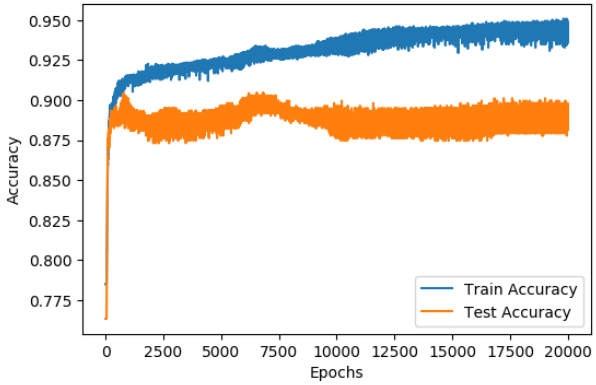

Figure 1

b) The approximate number of epochs where the test error converges is 7000 epochs. After 7000 epochs, the test accuracy starts dropping and finally reaches a steady state.

2. This part is to find the optimal batch size from 4, 8, 16, 32, 64. The batch sizes are evaluated based on their cross-validation accuracies and the time taken for training.
a) The cross-validation accuracies against epochs for five different batch sizes are shown in Figure 2 below:

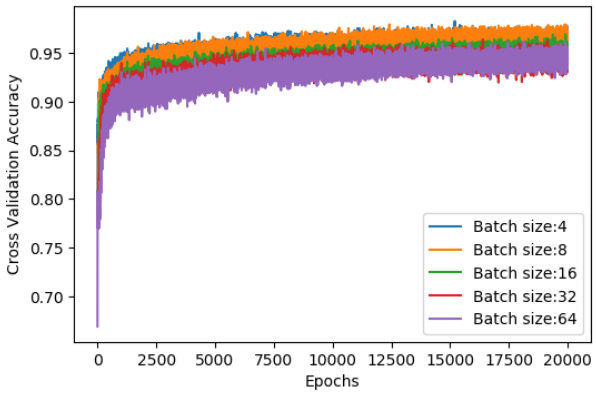

Figure 2

Besides, the time taken to train the FFN for one epoch against batch sizes are shown in Figure 3 below:

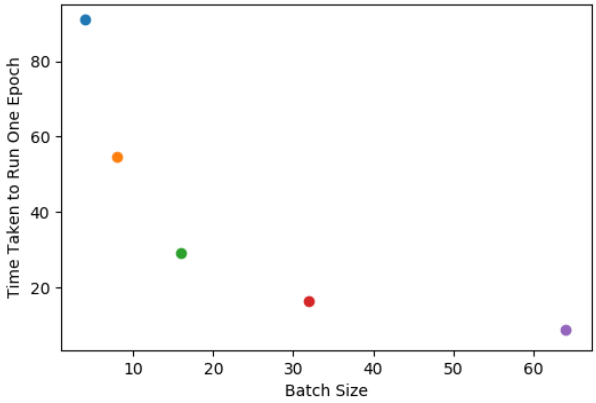

Figure 3

b) Based on results obtained in Figure 2 and Figure 3, the batch size of 8 is selected to be the optimal batch size. The reasons it is chosen include:

- A smaller batch size leads to a faster increase in the cross-validation accuracy at the beginning of the training.
- In the end, the cross-validation accuracy for batch size of 8 is similar to that for batch size of 4, and higher than the accuracies for other batch sizes.
- When batch size is 8, the time taken to train the FFN for one epoch is significantly lower than the time taken for batch size of 4.

c) When the batch size is changed to 8, the accuracies on both training and testing data against epochs are shown in the Figure 4 below:

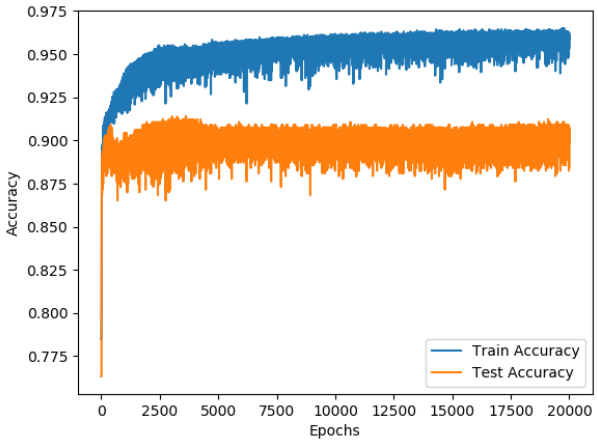

Figure 4

3. This part is to find the optimal number of neurons in the hidden-layer, the number is chosen from 5, 10, 15, 20, 25. The neuron numbers are evaluated based on their cross-validation accuracies.

a) The cross-validation accuracies against epochs for five different hidden-layer neuron numbers are shown in Figure 5 below:

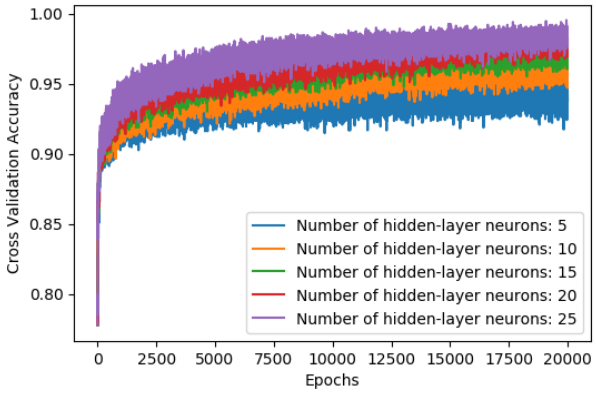

Figure 5

b) Based on Figure 5, it can be observed that the cross-validation accuracy for 25 hidden-layer neurons is above the cross-validation accuracies for other numbers of hidden neurons. Therefore, 25 is selected to be the optimal number of hidden neurons.

c) When the number of hidden neurons is changed to 25, the accuracies on both training and testing data against epochs are shown in the Figure 6 below:

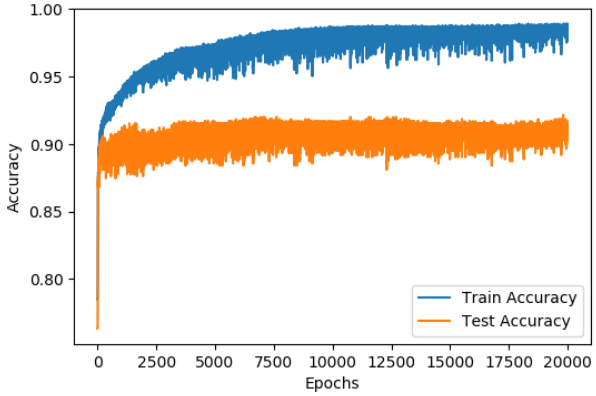

Figure 6

4. This part is to find the optimal weight decay parameter β from 0, 10^{-3}, 10^{-6}, 10^{-9}, 10^{-12}. The weight decay parameters are evaluated based on their cross-validation accuracies.
a) The cross-validation accuracies against epochs for five different weight decay parameters are shown in Figure 7 below:

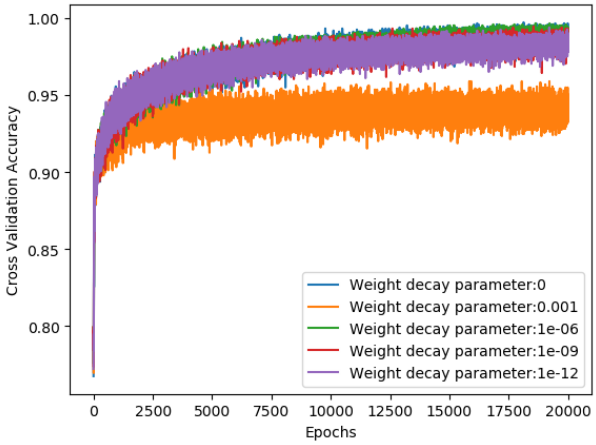

Figure 7

b) Based on Figure 7, it can be concluded that the optimal weight decay parameter is 0. The reason is that there is no significant difference between the cross-validation accuracies for weight decay parameters of 0, 1e-6, 1e-9 and 1e-12. Thus, there is no need for the additional computation using the weight decay parameter. Training the model with a weight decay parameter of 0 can reduce the usage of computational resources.
c) When the weight decay parameter is changed to 0, the accuracies on both training and testing data against epochs are shown in the Figure 8 below:

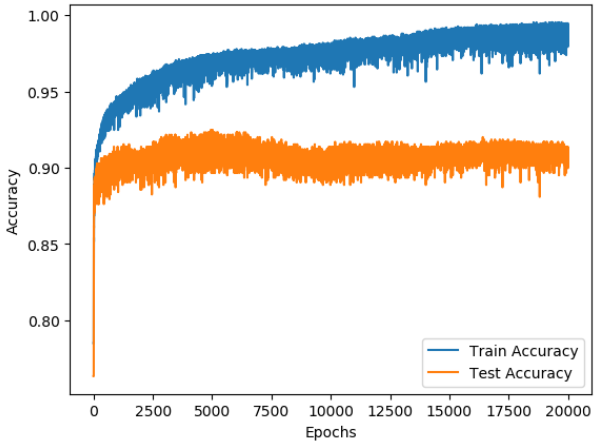

Figure 8

5. The 4-layer FFN is constructed with a learning rate of α = 0.01, batch size of 32, hidden layer neuron number of 10 for the two hidden layers, and weight decay parameter of β = 10^{-6}:
a) The accuracies on both training and testing data against epochs for the 4-layer FFN are shown in the Figure 9 below:

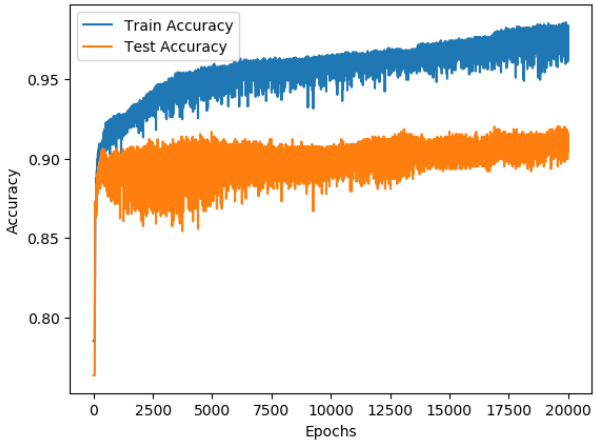

Figure 9

b) The optimal 3-layer FFN is constructed with a learning rate of α = 0.01, batch size of 8, hidden layer neuron number of 25, and weight decay parameter of β = 0, its accuracies on both training and testing data against epochs are shown in Figure 8 in part 4.

Compare Figure 9 with Figure 8, it can be observed that:

- In terms of the prediction accuracy on unseen data, 3-layer and 4-layer FFN have similar performance. For both of them, the test accuracies on unseen data converge to around 90%.
- During the training process, the train and test accuracies for the 4-layer FFN has a higher tendency to fluctuate up and down. At the beginning of the training, the test accuracy for the 4-layer FFN even fluctuates up and down in a range of around 5%, while the 3-layer FFN generally fluctuates within a range of around 2%.

Part B: Regression Problem

1. The 3-layer neural network is constructed with a learning rate of $\alpha = 10^{-3}$, batch size of 8, hidden layer neuron number of 10, and weight decay parameter of $\beta = 10^{-3}$:

a) The accuracies on both training and testing data against epochs are shown in the Figure 10 below

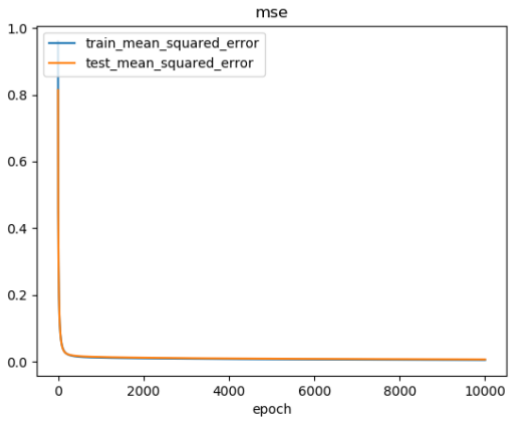

Figure 10

b) Since there is no overfitting, the early stopping technique (with monitor mse, patience 100, and threshold 1%) is used to get the optimal model, which is the one with 2900 epochs [Feature 11].

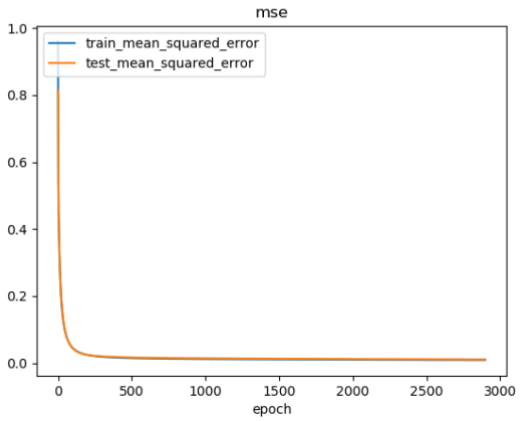

Figure 11

c) Then the optimal model is used to predict 50 test samples that randomly selected, and the mean squared error is 0.068967 [Figure 12].

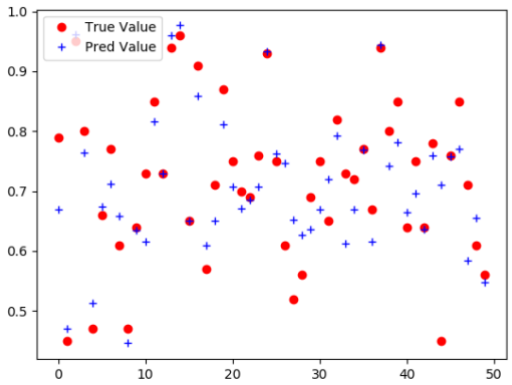

Figure 12

2.The correlation value is plotted in a heat map [Figure 13].

a) The most correlated features are 'GRE Score' and 'TOEFL Score' with correlation score 0.83598. This makes sense because they are both English tests, and people usually perform quite the same in the similar tests.

b) The feature which is most correlated to label is 'CGPA' with correlation score 0.87329.

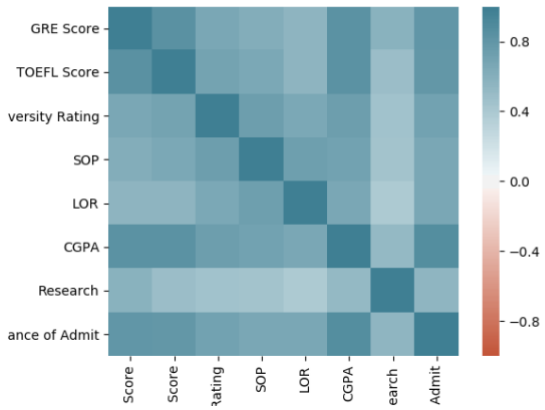

Figure 13

3.

Recursive feature elimination is used by removing the one that causes minimum drop in accuracy. In this project, we try to see by removing which feature, the mse is the lowest, which means the test error is still low when training model without the feature. We still use early stopping because we know that the dataset is not overfitting, and after removing features, the dataset gets more robust and less likely to overfit. So each model is still trained 2900 epochs.

The features removed are research and TOEFL score [Figure 14], which makes sense because research is the feature least correlated to the label with correlation score 0.553202, and TOEFL has high correlation with GRE Score (0.835977) and CGPA(0.833060). We can remove the feature that is not very related to the label since it does not affect the label much and remove some of the features that are highly correlated.


---------- Question 3(a) ----------
All features: ['GRE Score', 'TOEFL Score', 'University Rating', 'SOP', 'LOR', 'CGPA', 'Research']..
Removed GRE Score...
feature kept: ['TOEFL Score', 'University Rating', 'SOP', 'LOR', 'CGPA', 'Research']
mse: 0.011580242620160183
Removed TOEFL Score...
feature kept: ['GRE Score', 'University Rating', 'SOP', 'LOR', 'CGPA', 'Research']
mse: 0.0109443715463082
Removed University Rating...
feature kept: ['GRE Score', 'TOEFL Score', 'SOP', 'LOR', 'CGPA', 'Research']
mse: 0.012019321074088414
Removed SOP...
feature kept: ['GRE Score', 'TOEFL Score', 'University Rating', 'LOR', 'CGPA', 'Research']
mse: 0.011756140148888032
Removed LOR...
feature kept: ['GRE Score', 'TOEFL Score', 'University Rating', 'SOP', 'CGPA', 'Research']
mse: 0.013872299902141094
Removed CGPA...
feature kept: ['GRE Score', 'TOEFL Score', 'University Rating', 'SOP', 'LOR', 'Research']
mse: 0.012075148864338796
Removed Research...
feature kept: ['GRE Score', 'TOEFL Score', 'University Rating', 'SOP', 'LOR', 'CGPA']
mse: 0.010529165528714657
The first feature removed is Research
---------- Question 3(b) ----------
Removed GRE Score...
feature kept: ['TOEFL Score', 'University Rating', 'SOP', 'LOR', 'CGPA']
mse: 0.011755607649683952
Removed TOEFL Score...
feature kept: ['GRE Score', 'University Rating', 'SOP', 'LOR', 'CGPA']
mse: 0.011075917445123196
Removed University Rating...
feature kept: ['GRE Score', 'TOEFL Score', 'SOP', 'LOR', 'CGPA']
mse: 0.012531961686909199
Removed SOP...
feature kept: ['GRE Score', 'TOEFL Score', 'University Rating', 'LOR', 'CGPA']
mse: 0.012833314885695776
Removed LOR...
feature kept: ['GRE Score', 'TOEFL Score', 'University Rating', 'SOP', 'CGPA']
mse: 0.011952942827095588
Removed CGPA...
feature kept: ['GRE Score', 'TOEFL Score', 'University Rating', 'SOP', 'LOR']
mse: 0.013865043222904206
The second feature removed is TOEFL Score

Figure 14

4.

5 models are compared:

model_layer3_without_dropout: 1 hidden layer of 10 nodes, no dropout

model_layer4_without_dropout: 2 hidden layer of 50 nodes each, no dropout

model_layer4_with_dropout: 2 hidden layer of 50 nodes each, have dropout

model_layer5_without_dropout: 3 hidden layer of 50 nodes each, no dropout

model_layer5_with_dropout: 3 hidden layer of 50 nodes each, have dropout

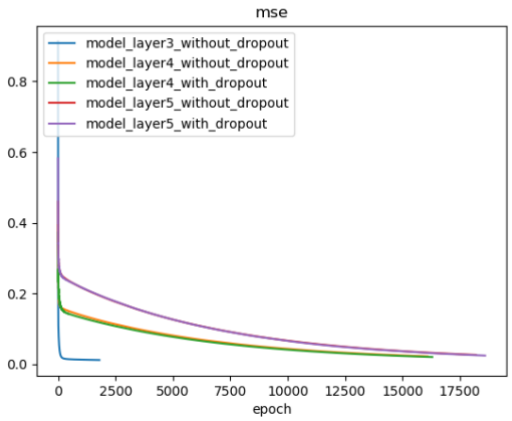

Figure 15

Early stop is used for all models since no overfitting occurs. As we can see, the complex model takes more time to converge and ends up with higher mse. Hence, the simple model with 3 layer performs best in this case. We can use more simple models when dataset is not hard, which can save time and also have better generalization.

Conclusion

In this project, we endeavour to get optimal model for each dataset. First, we can do feature selection to remove extra features. This can reduce the chance of overfitting and improve generalization. Commonly we can remove the features that 1. have low variance, 2. have low variance with the label, 3. Highly correlated to other features. After getting the clean data, we can turning the hyper-parameters one at a time by keeping others the same. Cross validation is a good way to ensure the general performance of model, and early stopping can be used to get optimal one. Also, weight decay and dropout keeps model less complex and prevent from overfitting. For simple dataset, simple model sometimes performs better than complex ones, also, by Occam's razor, simpler model (even with slightly worse performance) is a better model.