**Part A: Classification problem**

**Question 1**

* Learning Rate:
* L2 Regularization with Weight Decay parameter:
* Batch Size: 32
* Number of neurons: 10

|  |
| --- |
|  |
|  |
|  |

**Question 1a**

Use the training dataset to train the model and plot both accuracies on training and testing data against epochs.

|  |
| --- |
|  |

**Question 1b**

State the approximate number of epochs where the test error converges.

|  |
| --- |
|  |
|  |

The test error converged at about 1103 epochs.

**Question 2a**

Plot cross-validation accuracies against the number of epochs for different batch sizes. Limit search space to batch sizes {4,8,16,32,64}.

|  |
| --- |
|  |

Plot the time taken to train the network for one epoch against different batch sizes.

|  |
| --- |
|  |

**Question 2b**

Select the optimal batch size and state reasons for your selection.

The optimal batch size is 32 as shown in the figure in question 2a. At 10000 epoch where the test data converges, batch size 32 achieved the best testing accuracy as compared to the other batch size.

**Question 2c**

Plot the train and test accuracies against epochs for the optimal batch size.

|  |
| --- |
|  |

**Question 3a**

Plot cross-validation accuracies against the number of epochs for different number of hidden-layer neurons. Limit search space of the number of neurons to {5,10,15,20,25}.

|  |
| --- |
|  |

**Question 3b**

Select the optimal number of neurons for the hidden layer. State the rationale for your selection.

To avoid underfitting and overfitting, there is a need to increase in the number of neurons. As the number of hidden neurons increase, the network tends to remember the training data set with increasing number of parameters. This results in the network to minimize the training error at the expense of its generalization ability on unseen data.

Based on the figure above, the optimal number of neurons is 20 where it achieved the highest testing accuracy as compared to the rest.

**Question 3c**

Plot the train and test accuracies against epochs with the optimal number of neurons.

|  |
| --- |
|  |

Question 4a

Plot cross-validation accuracies against the number of epochs for the 3-layer network for different values of decay parameters. Limit the search space of decay

parameters to {0, , , , }.

|  |
| --- |
|  |

**Question 4b**

Select the optimal decay parameter. State the rationale for your selection.

All the decay parameter converges at about 10k epoch. The optimal decay parameter gives the best accuracy. Based on the figure above, decay parameter of 1e-9 is the optimal as it has the highest accuracy when compared to the rest.

**Question 4c**

Plot the train and test accuracies against epochs for the optimal decay parameter.

|  |
| --- |
|  |

**Question 5a**

Plot the train and test accuracy of the 4-layer network.

|  |
| --- |
|  |

**Question 5b**

Compare and comment on the performances of the optimal 3-layer and 4-layer networks.

|  |  |
| --- | --- |
|  |  |

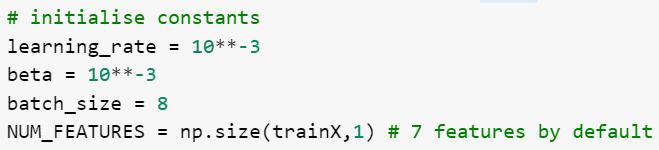
Based on the figures above, the 4 layer network has better training accuracy. However, when it comes to test accuracy, both 3 layer and 4 layer achieved similar accuracy with 4 layer network achieving higher test accuracy. Such difference is not very significant and can be ignored. Hence, the 3-layer network is still a better performer as compared to the 4-layer network in terms of computational cost.

# 

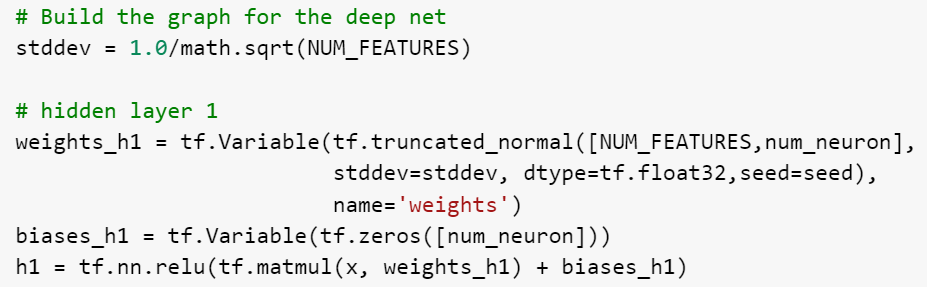
# Part B: Regression Problem

## Question 1

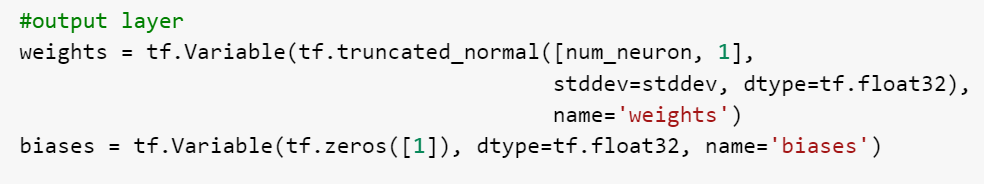
### Hyperparameters



### A hidden-layer of 10 neurons having ReLU activation functions

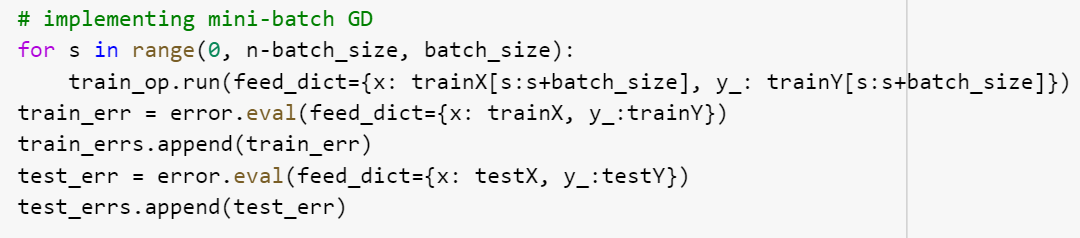


### A Linear Output Layer

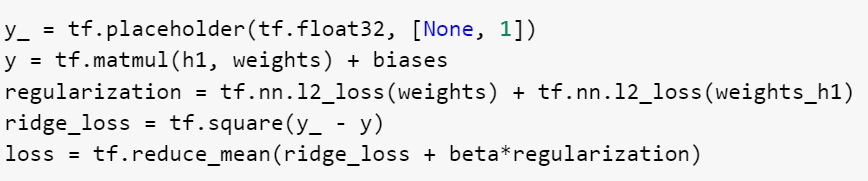
****

### 

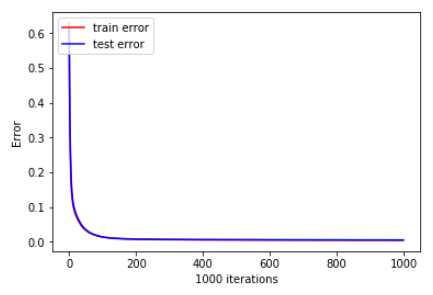
### Mini-batch gradient descent with a batch size = 8

****

### 𝐿2 regularization at weight decay parameter 𝛽 = 10−3



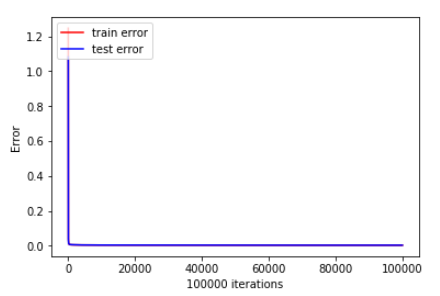
### Use the train dataset to train the model and plot both the train and test errors against epochs.



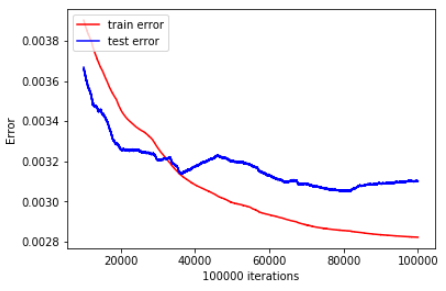
### 

### State the approximate number of epochs where the test error is minimum and use it to stop training.

To approximate number of epochs where the test error is minimum, the network was trained with 100000 epochs.

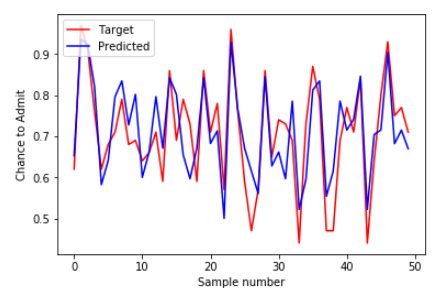
****

However, the minimum test error cannot be visually determined with this graph, because the test error looks like a straight line. Therefore, the first 10000 epochs was removed from the graph.

****

From the graph above, test error drops to about 0.0032 at approximately epochs = 30000. After that, the test error starts to fluctuate. This may be caused by overfitting. Therefore, the number of epochs where the test error is minimum is about 30000.

### Plot the predicted values and target values for any 50 test samples

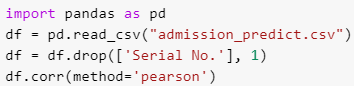


## 

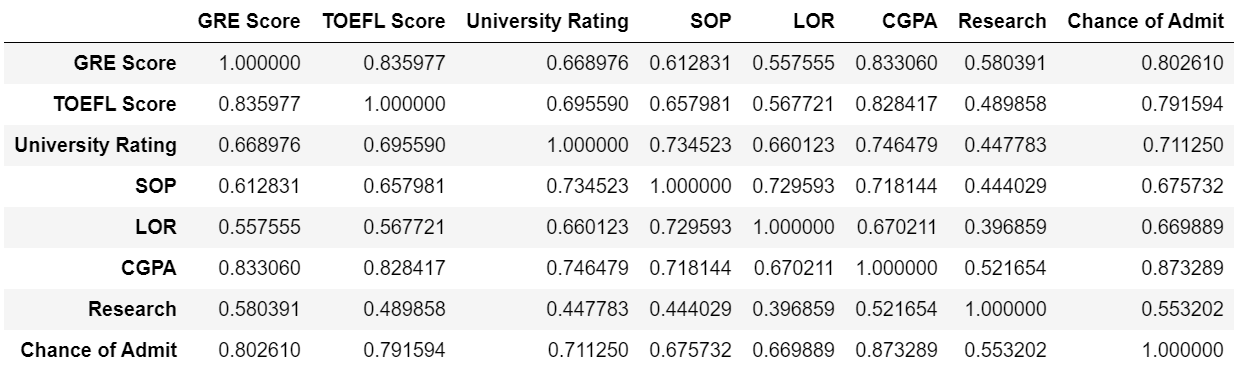
## Question 2

### Correlation Matrix

A 8X8 correlation matrix between the different feature scores and the corresponding chances of admit is plotted with pandas library. Pearson correlation coefficient is used in this question.



The result is as follows:

****

### Which features are most correlated to each other? Is it justifiable?

The result above shows that “GRE score” and “TOEFL score” has the highest correlation.

Graduate Record Examinations (GRE) is a graduate school entrance exam. Schools want to see GRE scores to make sure the student can handle graduate-level coursework. Analytical Writing, Quantitative Reasoning, and Verbal Reasoning is tested for GRE

Test of English as a Foreign Language (TOEFL) is a test of the student’s English language skills. Schools want to see TOEFL scores to make sure the student’s English skills are strong enough to do well at an English-speaking school. Reading, Listening, Speaking, and Writing is tested for TOEFL

In short, both of these features are entrance exam for a school, but the content tested are different. A student that score well in one test may not score well in the other. Therefore, the high correlation between these two features is only justifiable to a certain extent.

### What features have the highest correlations with the chances of admit?

CGPA has the highest correlations with the chances of admit.

## Question 3

### Define gradientDescentOptimizer() Function

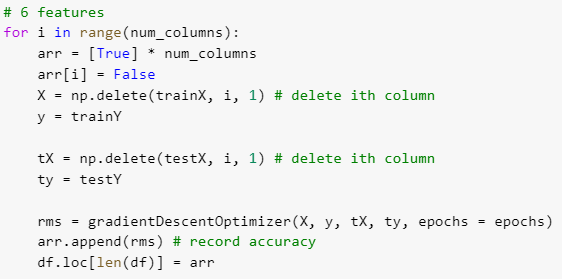
gradientDescentOptimizer() was defined to implement Recursive feature elimination. This function contains the methods for a 3-layer feedforward neural network. The implementation of this neural network and most predefined constants (like number of neurons and learning rate) are the same as Question 1.

Additionally, this function takes in trainX, trainY, testX, and testY as its parameter. Number of epochs used is set to 1000 for this question.

### 6 Input Features RFE Implementation

To measure the performance of 6 input features, this procedure is performed recursively:

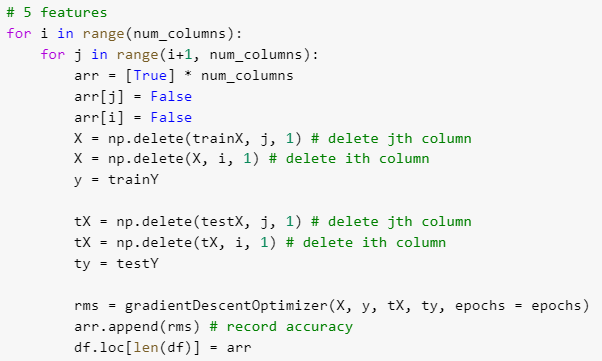
1. One input feature was removed recursively from trainX and trainY.
2. trainX, trainY, testX, and testY is passed into gradientDescentOptimizer()
3. The accuracy of the model with the input is recorded into a dataframe (with pandas library)



### 5 Input Features RFE Implementation

To measure the performance of 5 input features, this procedure is performed recursively:

1. Two input feature was removed recursively from trainX and trainY.
2. trainX, trainY, testX, and testY is passed into gradientDescentOptimizer()
3. The accuracy of the model with the input is recorded into a dataframe (with pandas library)



### Results



### Observation

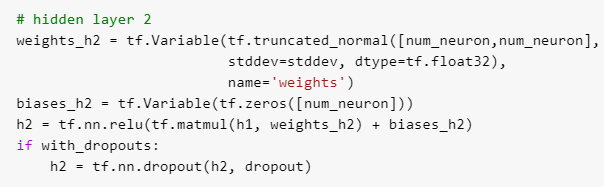
From the results, the optimal feature set is ['GRE Score', 'TOEFL Score','University Rating', 'LOR', 'CGPA', 'Research']. However, this result is achieved with a fixed random seed. If the random seed is not fixed, the performance for each input feature will be different in every run of RFE, because the weights in the hidden and output layers of the network will be initialised randomly (and differently) for each run. This shows that RFE cannot be done on neural network.

## Question 4

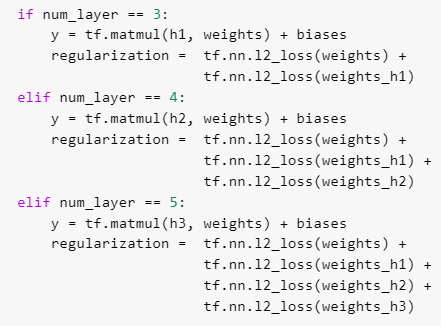
### Implementing four-layer and a five-layer neural network

gradientDescentOptimizer() from Question 3 was modified to take in “num\_layer” and “with\_dropout” as parameters.

* “with\_dropout” is used to set if dropout should be used for the hidden layer.



* “num\_layer” is used to set the number of layers used in the network

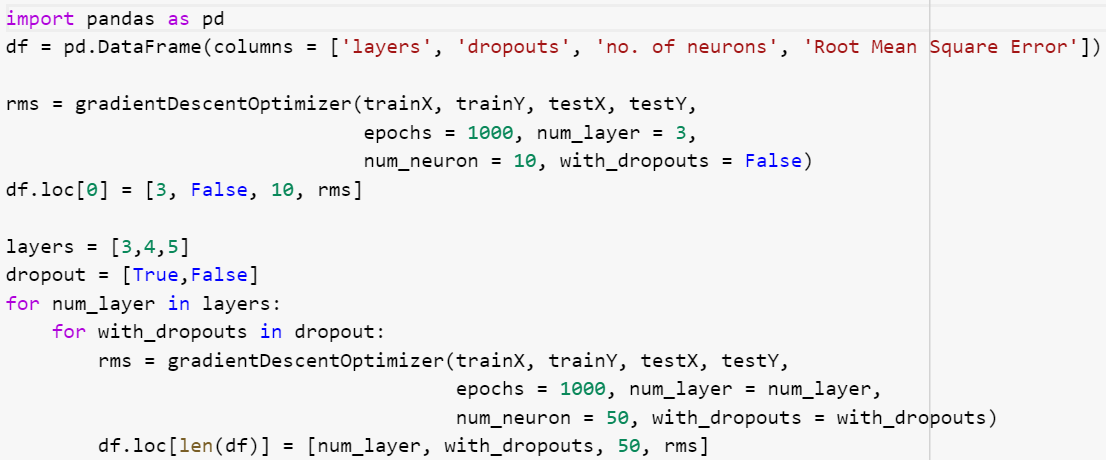


### Selecting Optimal Feature Set



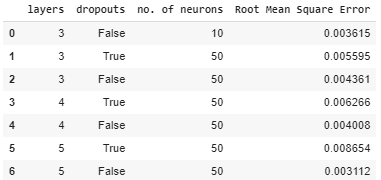
### Compare the performances of all networks (with and without dropouts)

To compare the performance, all networks were run, and the performance for each network (with and without dropouts) were recorded.



### 

### Result



From the results above, four observations can be made about this implementation and dataset

* Networks without dropout performed better when there is more layers
* Networks with dropout performed better when there is less layers
* Networks without dropouts performed better than networks with dropouts
* 3 layers with 10 neurons performed better than 3 layers with 50 neurons

This observation, however, is only true for this implementation and dataset. Changing the hyperparameter of the implementation (e.g. batch size, number of epochs, learning rate, etc) will lead to a different result and observation