MIT-WPU

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Final Year B. Tech (EE)

Trimester: X **Subject: AIML**

Name: Shreerang Mhatre **Class: Ty**

Roll No: 52 **Batch: A3**

Experiment No: 02

Name of the Experiment: Create and view custom neural networks

Performed on: 24/08/2023

Submitted on: 31/08/2023

Marks	Teacher's Signature with date

Aim: To create and view custom neural networks.

Prerequisite: Knowledge of NN tool in MATLAB, MLP, Activation function.

Objective:

To create and study the Neural Network by varying parameters.

1. Define Input and Output Variable
2. Define and custom Neural Network
3. Define Transfer Function
4. Configure the network
5. Train the network to find output

Components and Equipment required:

MATLAB with NNTool Box

Theory:

Neural networks are a set of algorithms, modelled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labelling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated. Neural networks help us cluster and classify.

In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. The concept of neural networks, which has its roots in artificial intelligence, is swiftly gaining popularity in the development of trading systems.

A neural network contains layers of interconnected nodes. Each node is a perceptron and is similar to a multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear.

In a multi-layered perceptron (MLP), perceptrons are arranged in interconnected layers. The input layer collects input patterns. The output layer has classifications or output signals to which input patterns may map.

Hidden layers fine-tune the input weightings until the neural network’s margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs.

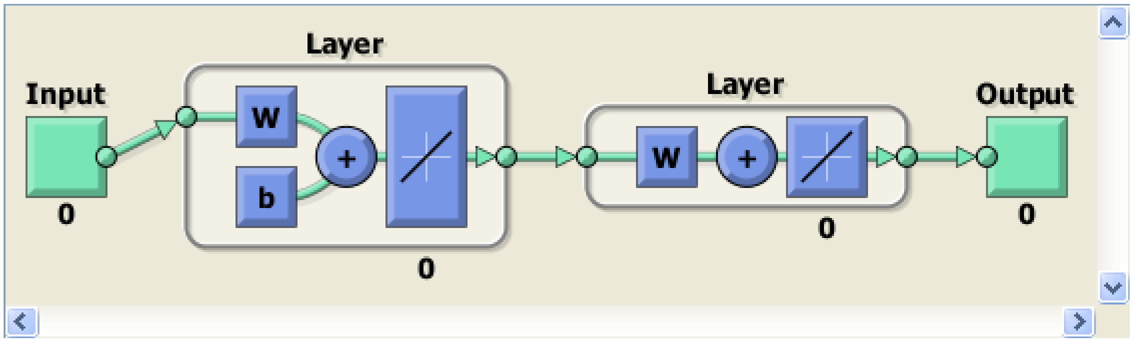
Procedure

Step-1

Define and custom network

Define input and output/ Target variable

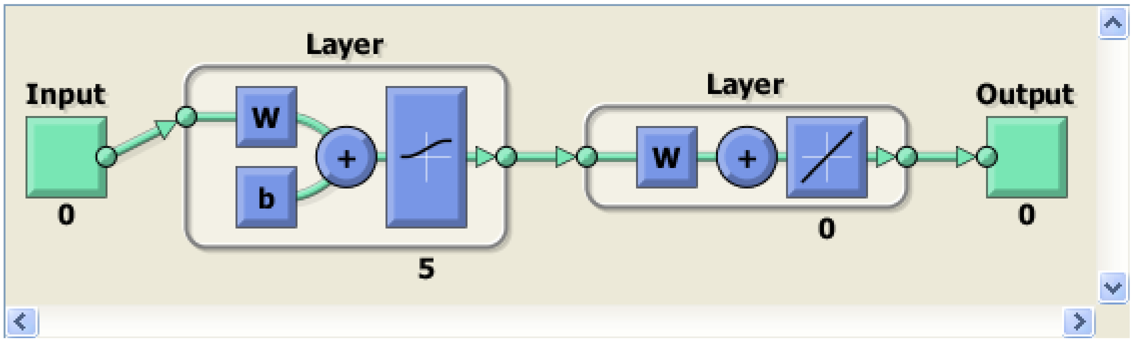
Assign input and target data



Step-2

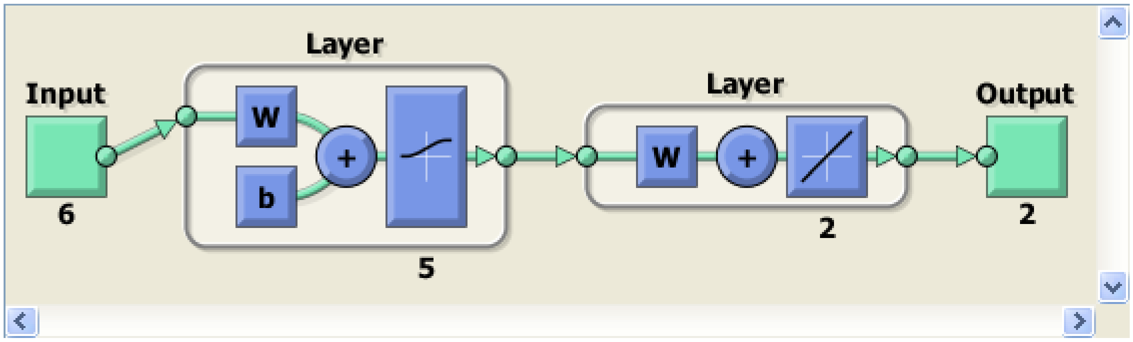
Define topology and transfer function

No. Of hidden layers and their neurons



Step-3

Configure and view network



Step-4

Train net and calculate neuron output

Observations:

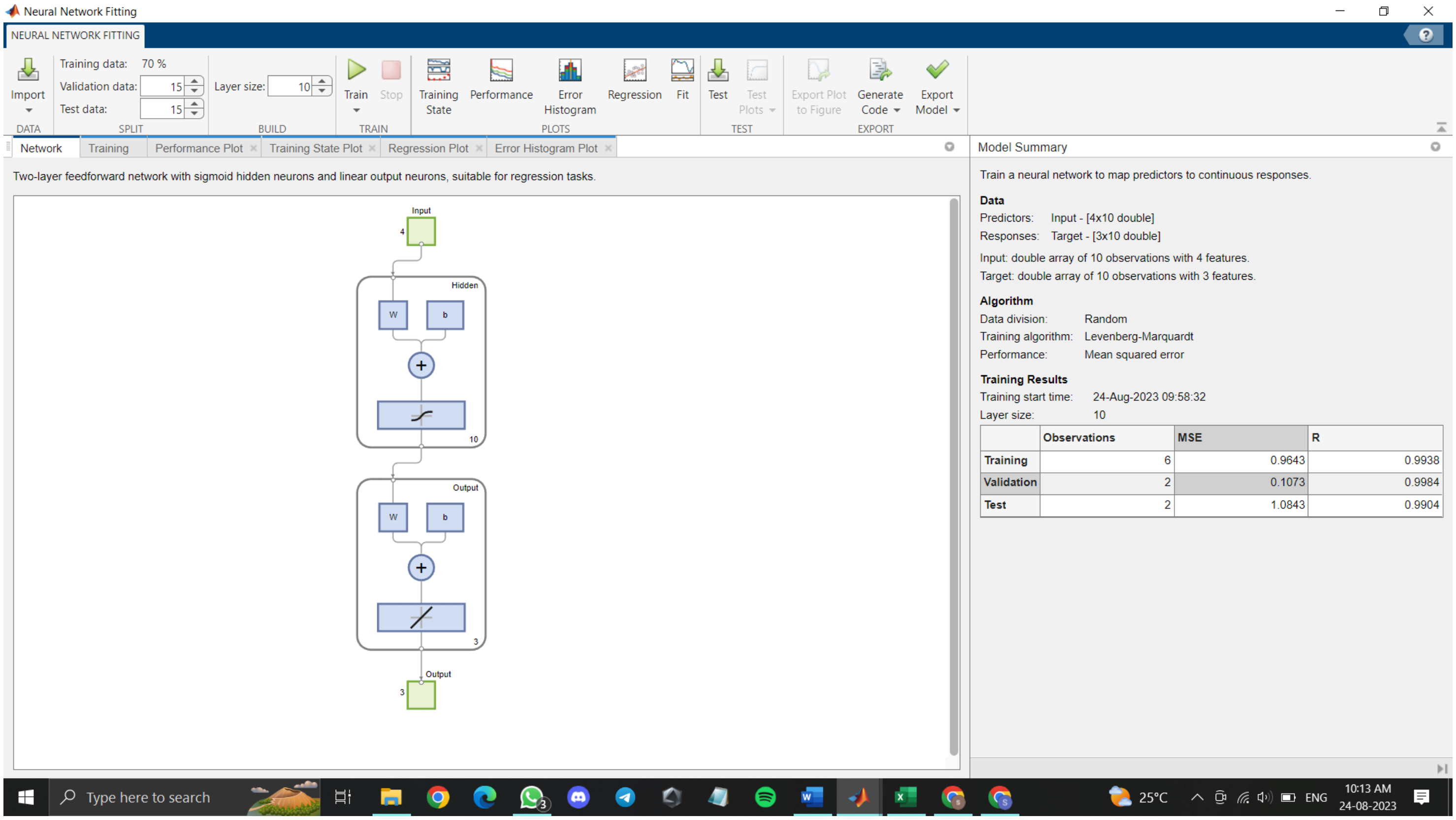
Change the training parameters and view the performance.

Performance Plot

Training Graph

Regression Curves

MATLAB Output:



Training data: 70 %
Validation data: 15
Test data: 15
Layer size: 10

Two-layer feedforward network with sigmoid hidden neurons and linear output neurons, suitable for regression tasks.

Model Summary

Train a neural network to map predictors to continuous responses.

Data

Predictors: Input - [4x10 double]
Responses: Target - [3x10 double]

Input: double array of 10 observations with 4 features.
Target: double array of 10 observations with 3 features.

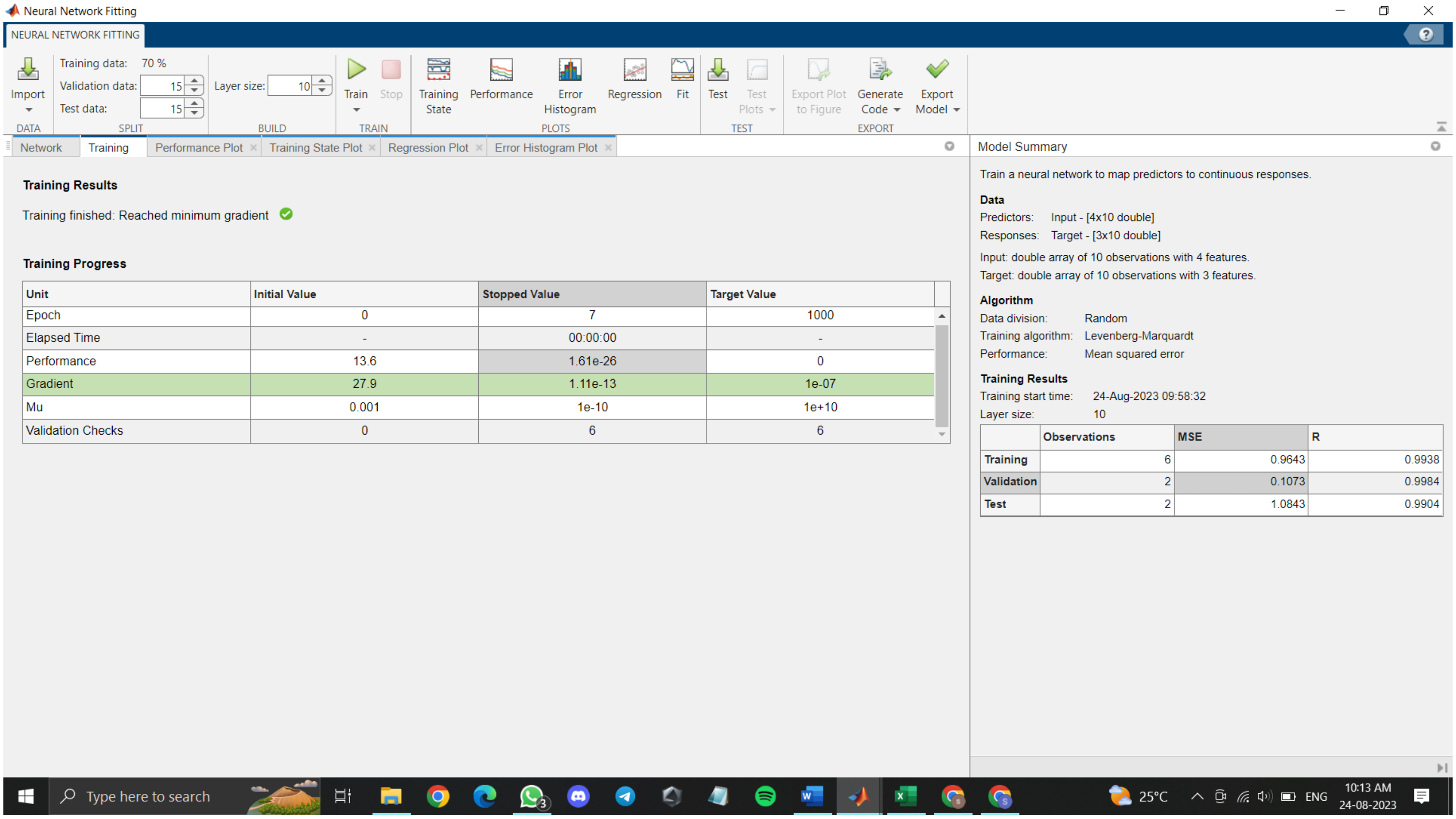
Algorithm

Data division: Random
Training algorithm: Levenberg-Marquardt
Performance: Mean squared error

Training Results

Training start time: 24-Aug-2023 09:58:32
Layer size: 10

	Observations	MSE	R
Training	6	0.9643	0.9938
Validation	2	0.1073	0.9984
Test	2	1.0843	0.9904



Training data: 70 %
Validation data: 15
Test data: 15

Training Results

Training finished: Reached minimum gradient ✓

Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	7	1000
Elapsed Time	-	00:00:00	-
Performance	13.6	1.61e-26	0
Gradient	27.9	1.11e-13	1e-07
Mu	0.001	1e-10	1e+10
Validation Checks	0	6	6

Model Summary

Train a neural network to map predictors to continuous responses.

Data

Predictors: Input - [4x10 double]
Responses: Target - [3x10 double]

Input: double array of 10 observations with 4 features.
Target: double array of 10 observations with 3 features.

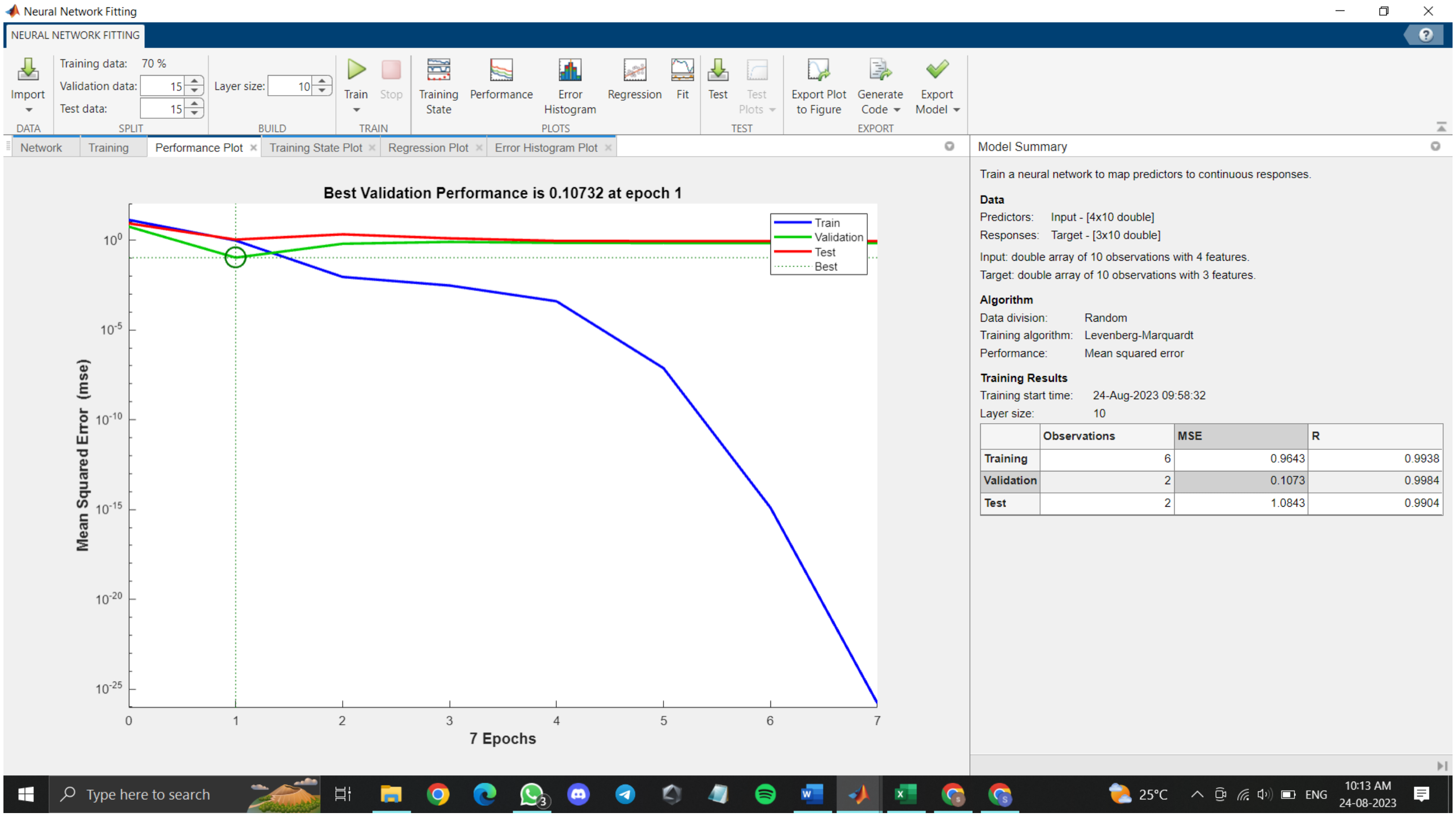
Algorithm

Data division: Random
Training algorithm: Levenberg-Marquardt
Performance: Mean squared error

Training Results

Training start time: 24-Aug-2023 09:58:32
Layer size: 10

	Observations	MSE	R
Training	6	0.9643	0.9938
Validation	2	0.1073	0.9984
Test	2	1.0843	0.9904



Training data: 70 %
Validation data: 15
Test data: 15

Model Summary

Train a neural network to map predictors to continuous responses.

Data

Predictors: Input - [4x10 double]
Responses: Target - [3x10 double]

Input: double array of 10 observations with 4 features.
Target: double array of 10 observations with 3 features.

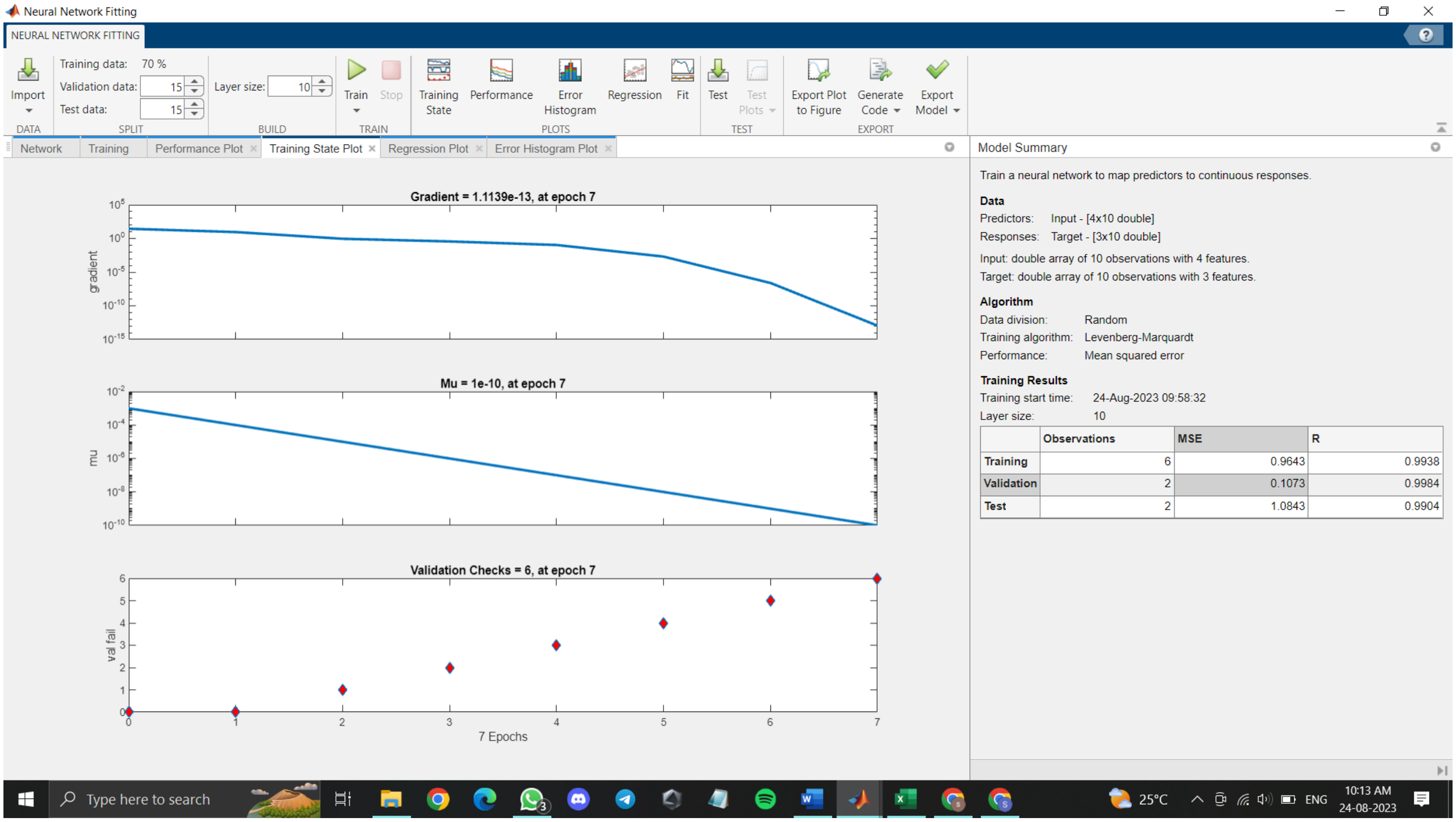
Algorithm

Data division: Random
Training algorithm: Levenberg-Marquardt
Performance: Mean squared error

Training Results

Training start time: 24-Aug-2023 09:58:32
Layer size: 10

	Observations	MSE	R
Training	6	0.9643	0.9938
Validation	2	0.1073	0.9984
Test	2	1.0843	0.9904



Model Summary

Train a neural network to map predictors to continuous responses.

Data

Predictors: Input - [4x10 double]
Responses: Target - [3x10 double]

Input: double array of 10 observations with 4 features.
Target: double array of 10 observations with 3 features.

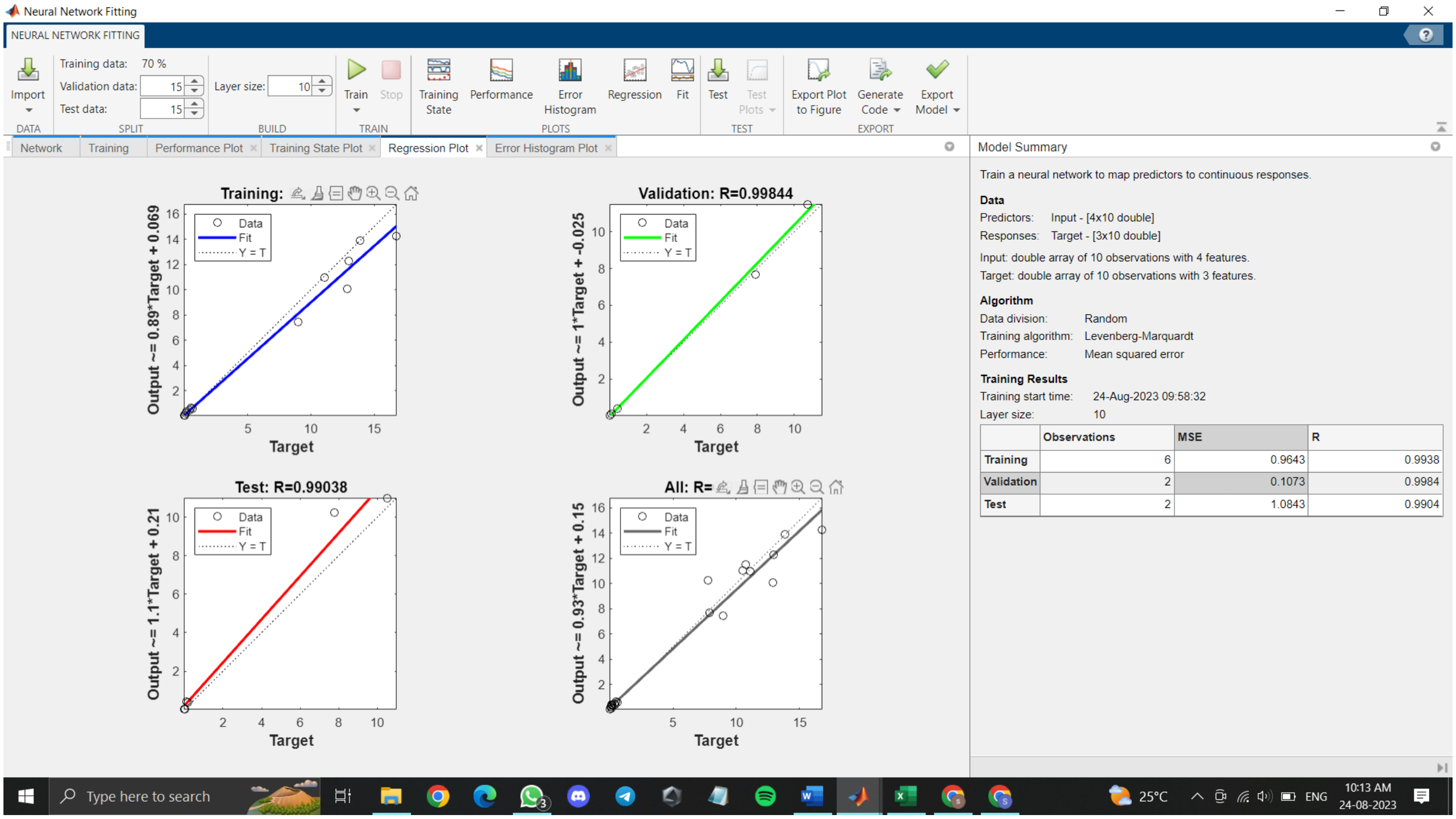
Algorithm

Data division: Random
Training algorithm: Levenberg-Marquardt
Performance: Mean squared error

Training Results

Training start time: 24-Aug-2023 09:58:32
Layer size: 10

	Observations	MSE	R
Training	6	0.9643	0.9938
Validation	2	0.1073	0.9984
Test	2	1.0843	0.9904



Model Summary

Train a neural network to map predictors to continuous responses.

Data

Predictors: Input - [4x10 double]
Responses: Target - [3x10 double]

Input: double array of 10 observations with 4 features.
Target: double array of 10 observations with 3 features.

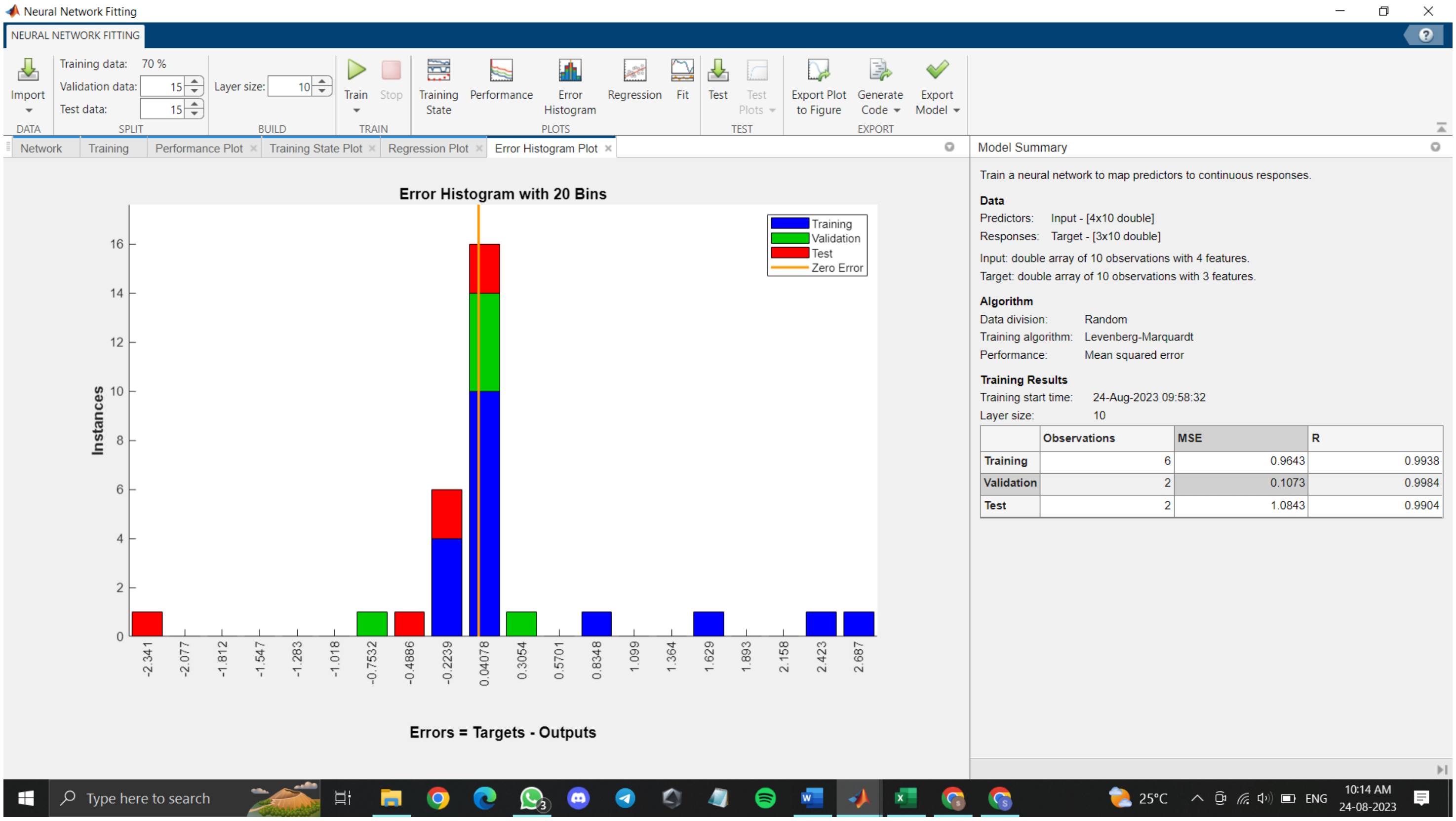
Algorithm

Data division: Random
Training algorithm: Levenberg-Marquardt
Performance: Mean squared error

Training Results

Training start time: 24-Aug-2023 09:58:32
Layer size: 10

	Observations	MSE	R
Training	6	0.9643	0.9938
Validation	2	0.1073	0.9984
Test	2	1.0843	0.9904



Train a neural network to map predictors to continuous responses.

Data

Predictors: Input - [4x10 double]
Responses: Target - [3x10 double]

Input: double array of 10 observations with 4 features.
Target: double array of 10 observations with 3 features.

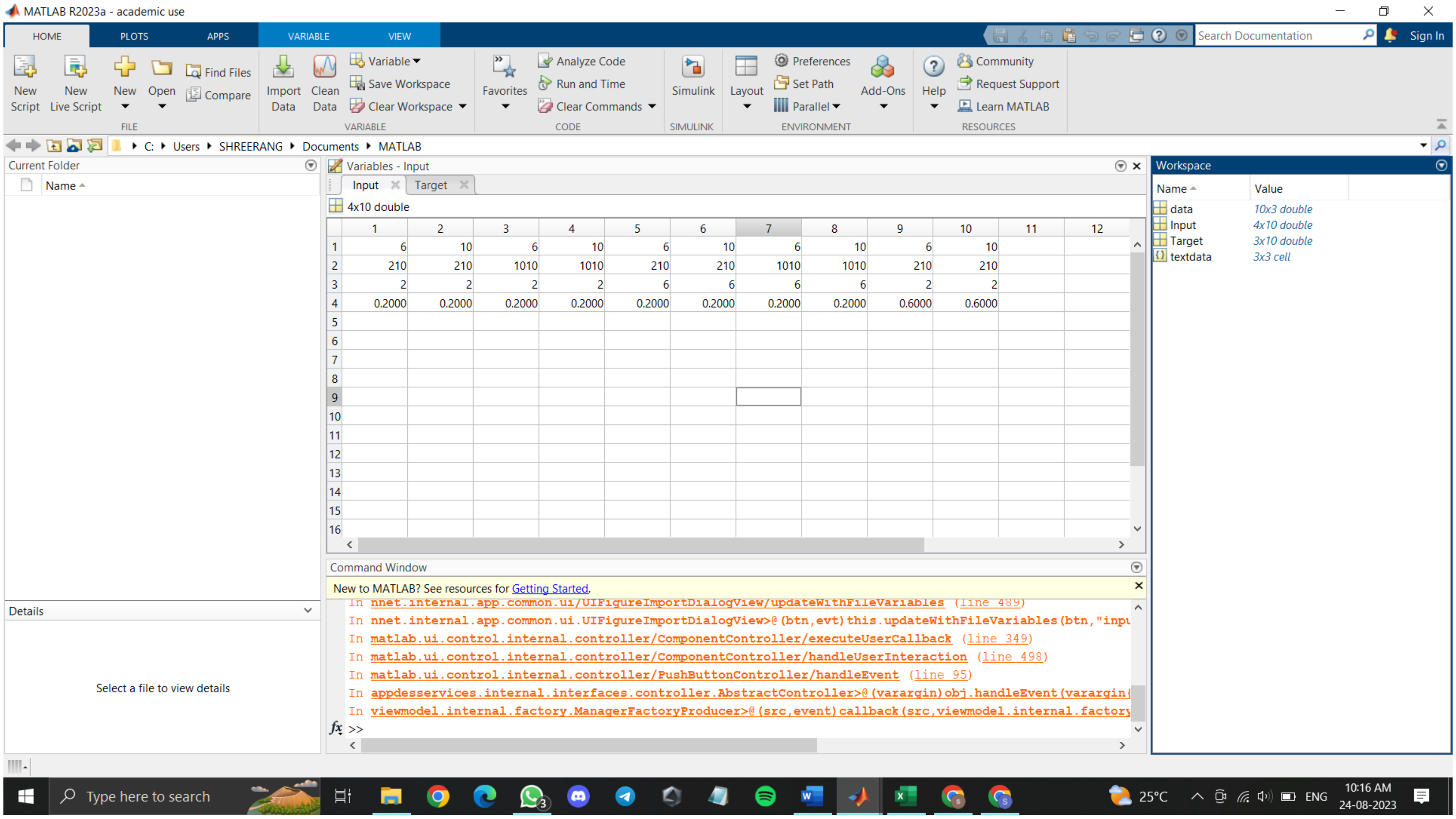
Algorithm

Data division: Random
Training algorithm: Levenberg-Marquardt
Performance: Mean squared error

Training Results

Training start time: 24-Aug-2023 09:58:32
Layer size: 10

	Observations	MSE	R
Training	6	0.9643	0.9938
Validation	2	0.1073	0.9984
Test	2	1.0843	0.9904



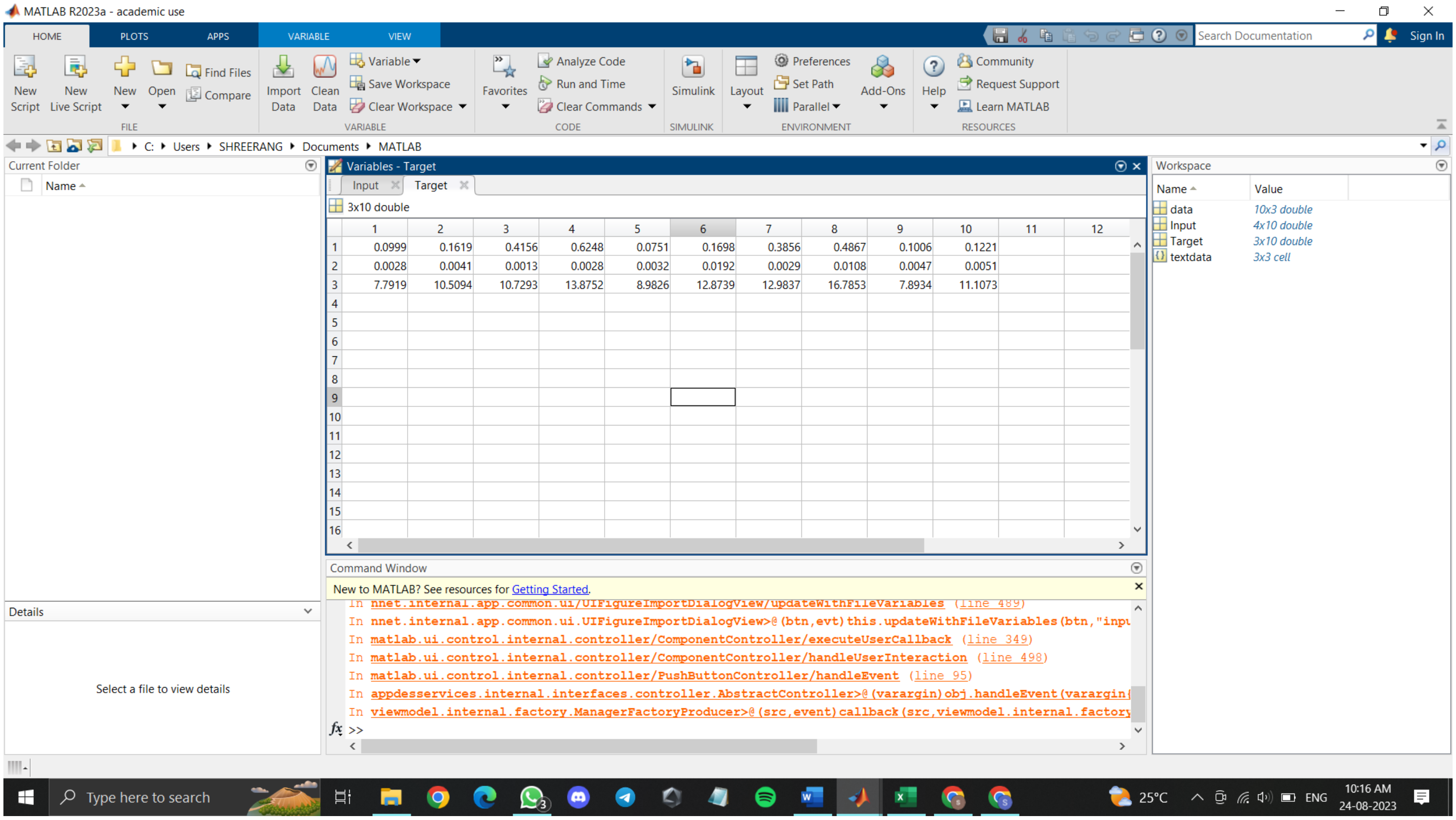
4x10 double

	1	2	3	4	5	6	7	8	9	10	11	12
1	6	10	6	10	6	10	6	10	6	10		
2	210	210	1010	1010	210	210	1010	1010	210	210		
3	2	2	2	2	6	6	6	6	2	2		
4	0.2000	0.2000	0.2000	0.2000	0.2000	0.2000	0.2000	0.2000	0.6000	0.6000		
5												
6												
7												
8												
9												
10												
11												
12												
13												
14												
15												
16												

New to MATLAB? See resources for Getting Started.

```
In nnet.internal.app.common.ui/UIFigureImportDialogView/updateWithFileVariables (line 489)
In nnet.internal.app.common.ui.UIFigureImportDialogView>@(btn,evt)this.updateWithFileVariables(btn,"inpu
In matlab.ui.control.internal.controller/ComponentController/executeUserCallback (line 349)
In matlab.ui.control.internal.controller/ComponentController/handleUserInteraction (line 498)
In matlab.ui.control.internal.controller/PushButtonController/handleEvent (line 95)
In appdesservices.internal.interfaces.controller.AbstractController>@(varargin)obj.handleEvent(varargin{
In viewmodel.internal.factory.ManagerFactoryProducer>@(src,event)callback(src,viewmodel.internal.factory
fx >>
```

Name	Value
data	10x3 double
Input	4x10 double
Target	3x10 double
textdata	3x3 cell



3x10 double

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.0999	0.1619	0.4156	0.6248	0.0751	0.1698	0.3856	0.4867	0.1006	0.1221		
2	0.0028	0.0041	0.0013	0.0028	0.0032	0.0192	0.0029	0.0108	0.0047	0.0051		
3	7.7919	10.5094	10.7293	13.8752	8.9826	12.8739	12.9837	16.7853	7.8934	11.1073		
4												
5												
6												
7												
8												
9												
10												
11												
12												
13												
14												
15												
16												

Name	Value
data	10x3 double
Input	4x10 double
Target	3x10 double
textdata	3x3 cell

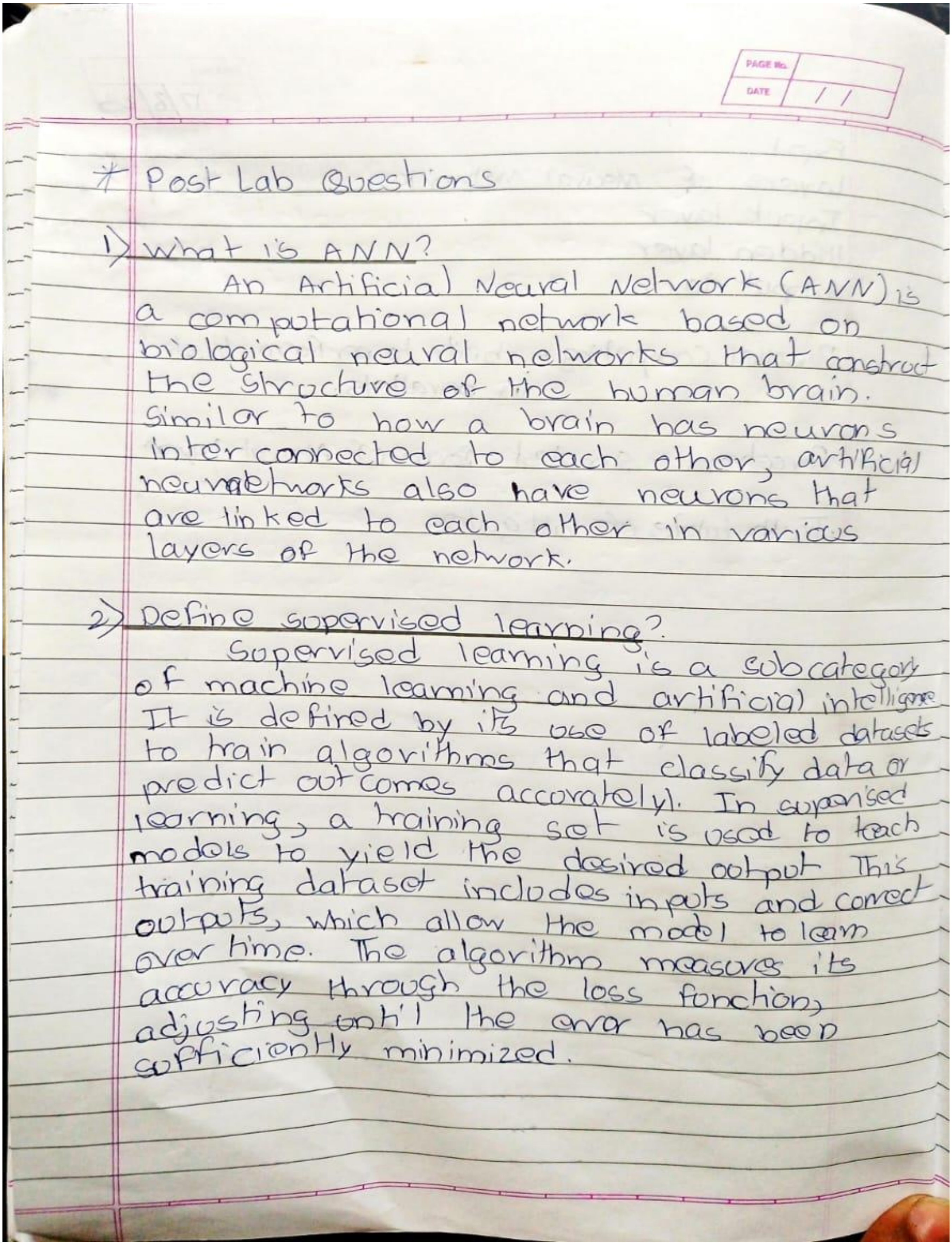
Command Window

New to MATLAB? See resources for Getting Started.

```
In nnet.internal.app.common.ui/UIFigureImportDialogView/updateWithFileVariables (line 489)
In nnet.internal.app.common.ui.UIFigureImportDialogView>@(btn,evt)this.updateWithFileVariables(btn,"inpu
In matlab.ui.control.internal.controller/ComponentController/executeUserCallback (line 349)
In matlab.ui.control.internal.controller/ComponentController/handleUserInteraction (line 498)
In matlab.ui.control.internal.controller/PushButtonController/handleEvent (line 95)
In appdesservices.internal.interfaces.controller.AbstractController>@(varargin)obj.handleEvent(varargin{
In viewmodel.internal.factory.ManagerFactoryProducer>@(src,event)callback(src,viewmodel.internal.factory
>>
```

Conclusions:

Post Lab Questions:



PAGE No.
DATE / /

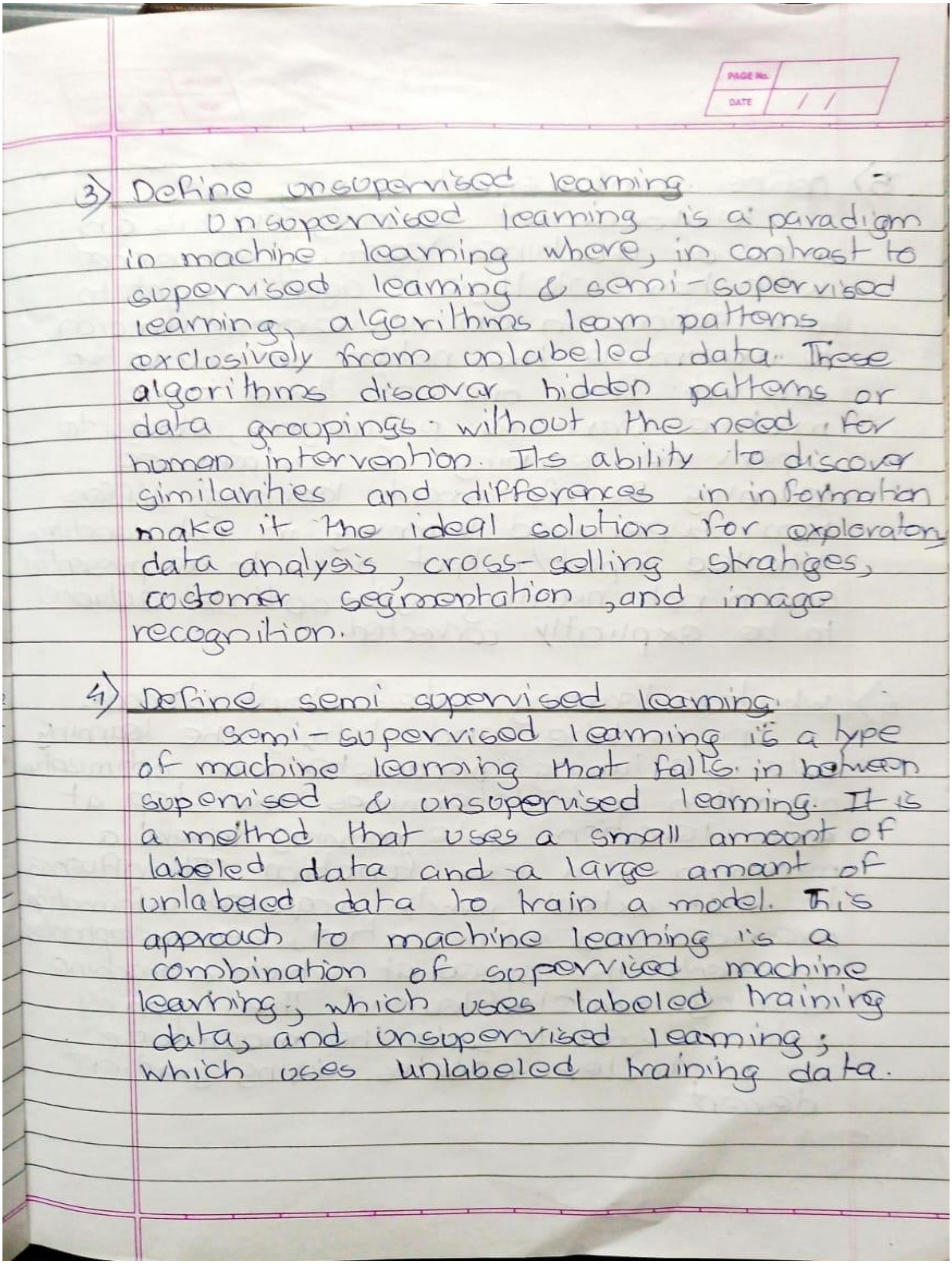
* Post Lab Questions

1) What is ANN?

An Artificial Neural Network (ANN) is a computational network based on biological neural networks that construct the structure of the human brain. Similar to how a brain has neurons interconnected to each other, artificial neuralnetworks also have neurons that are linked to each other in various layers of the network.

2) Define supervised learning?

Supervised learning is a subcategory of machine learning and artificial intelligence. It is defined by its use of labeled datasets to train algorithms that classify data or predict outcomes accuratelyl. In supervised learning, a training set is used to teach models to yield the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized.



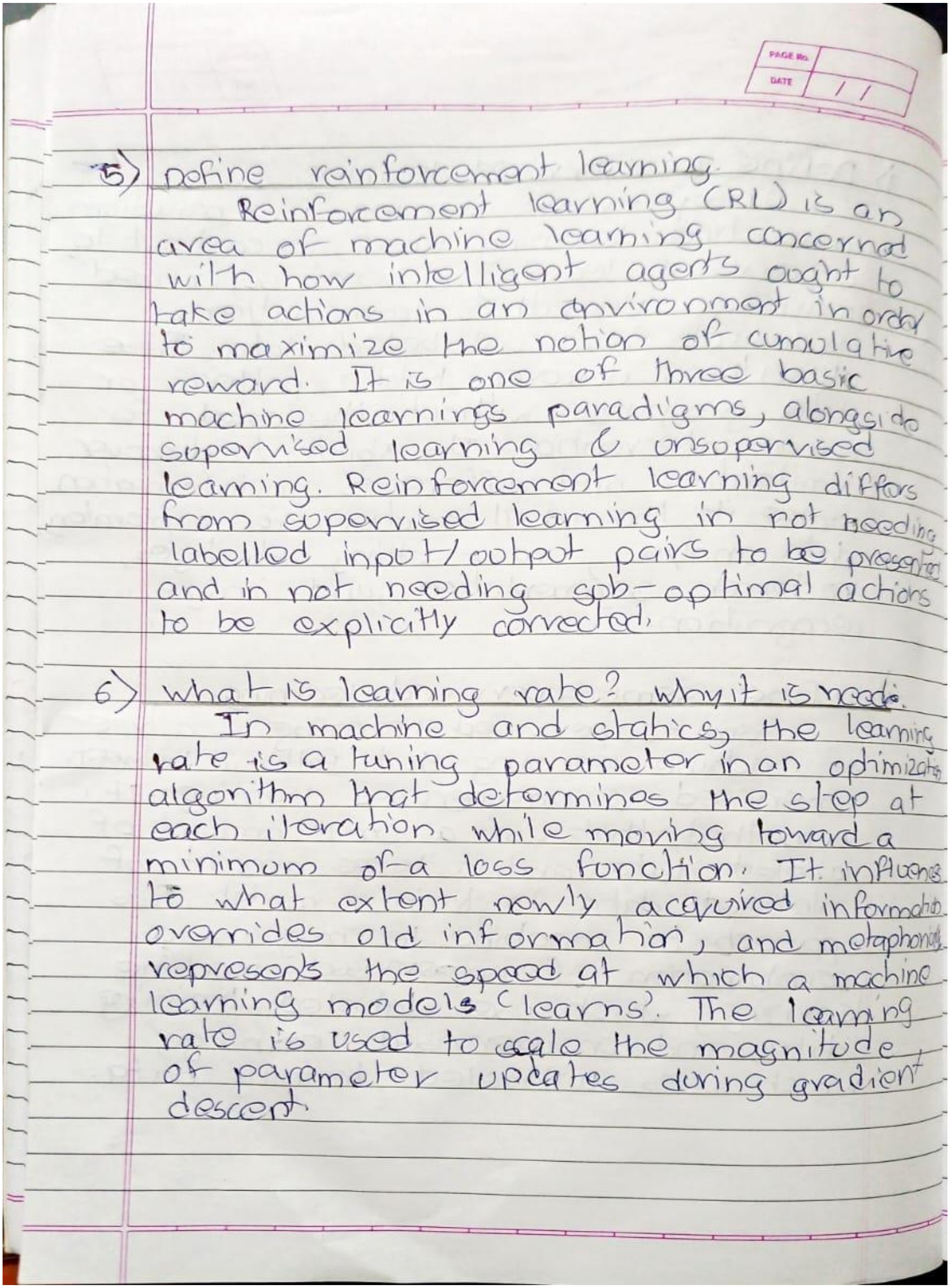
PAGE No.
DATE / /

3) Define unsupervised learning

Unsupervised learning is a paradigm in machine learning where, in contrast to supervised learning & semi-supervised learning, algorithms learn patterns exclusively from unlabeled data. These algorithms discover hidden patterns or data groupings without the need for human intervention. Its ability to discover similarities and differences in information make it the ideal solution for exploratory data analysis, cross-selling strateges, customer segmentation, and image recognition.

4) Define semi supervised learning

Semi-supervised learning is a type of machine learning that falls in between supervised & unsupervised learning. It is a method that uses a small amount of labeled data and a large amount of unlabeled data to train a model. This approach to machine learning is a combination of supervised machine learning, which uses labeled training data, and unsupervised learning; which uses unlabeled training data.



PAGE No.
DATE / /

5) Define reinforcement learning.

Reinforcement learning (RL) is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward. It is one of three basic machine learnings paradigms, alongside supervised learning & unsupervised learning. Reinforcement learning differs from supervised learning in not needing labelled input/output pairs to be presented and in not needing sub-optimal actions to be explicitly corrected.

6) What is learning rate? Why it is neede.

In machine and statics, the learning rate is a tuning parameter in an optimization algorithm that determines the step at each iteration while moving toward a minimum of a loss function. It influences to what extent newly acquired information overrides old information, and metaphorically represents the speed at which a machine learning models 'learns'. The learning rate is used to scale the magnitude of parameter updates during gradient descent.