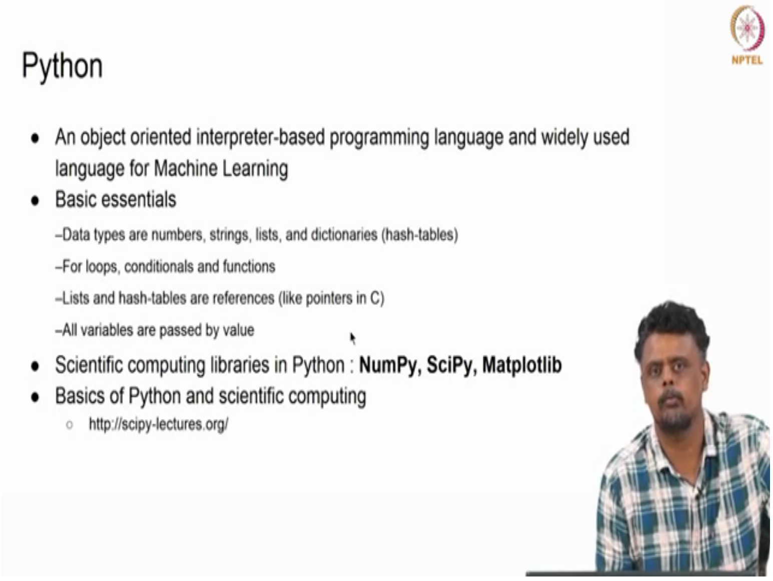
**Machine Learning for Engineering and Science Applications**
**Professor Dr. Ganapathy Krishnamurthi**
**Department of Engineering Design**
**Indian Institute of Technology, Madras**
**Introduction to Packages**

(Refer Slide Time: 0:14)

Introduction to Packages

Hello and welcome back. So we will give you a small introduction to various packages available for implementing some of the machine and deep learning algorithms that you will learn in this course.

(Refer Slide Time: 0:26)



Python

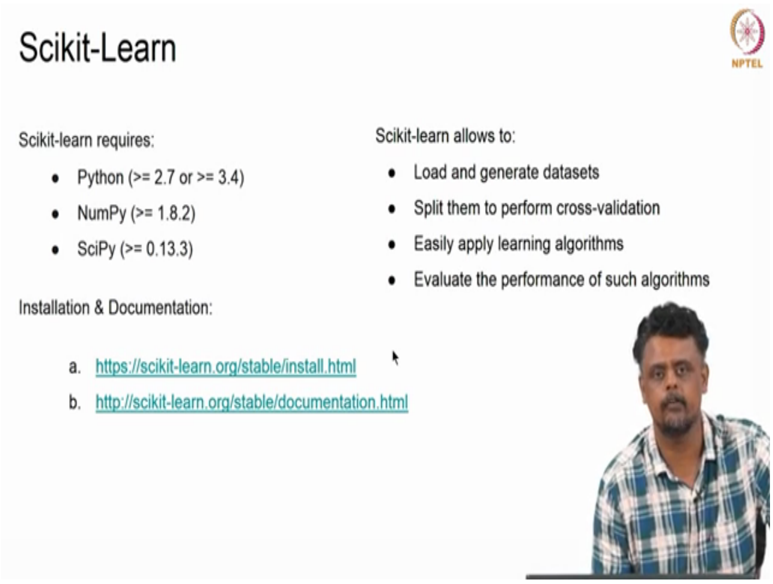
- An object oriented interpreter-based programming language and widely used language for Machine Learning
- Basic essentials
  - –Data types are numbers, strings, lists, and dictionaries (hash-tables)
  - –For loops, conditionals and functions
  - –Lists and hash-tables are references (like pointers in C)
  - –All variables are passed by value
- Scientific computing libraries in Python : **NumPy, SciPy, Matplotlib**
- Basics of Python and scientific computing
  - ○ http://scipy-lectures.org/

So to start off with of course all of you must be familiar with python, it is a interpreter based programming language that is very popular specially for prototyping and in many cases even for production level software. So it lets you to represent data types there are several data types includes numbers, strings, list in the form of strings, lists and dictionaries. It has a typical programming constructs like for loops, conditionals and functions and all variables in general are passed by value.

So it has its own scientific computational libraries like NumPy and SciPy and also plotting capabilities the Matplotlib these are modules that you can import into Python as you program it, okay so there are various resources available on the web, one of them is mentioned at the bottom of the page of this slide and you can welcome to explore these options. So there are again online courses available which leads you learn Python from scratch so that you are more comfortable programming in this language.

This is important for many of the other packages that we will see in the (())(1:35) slides. So it is good to have basic capability in Python and get it started before we get to the point where we do some programming.

(Refer Slide Time: 1:45)



Scikit-Learn

Scikit-learn requires:

- Python (>= 2.7 or >= 3.4)
- NumPy (>= 1.8.2)
- SciPy (>= 0.13.3)

Installation & Documentation:

a. https://scikit-learn.org/stable/install.html
b. http://scikit-learn.org/stable/documentation.html

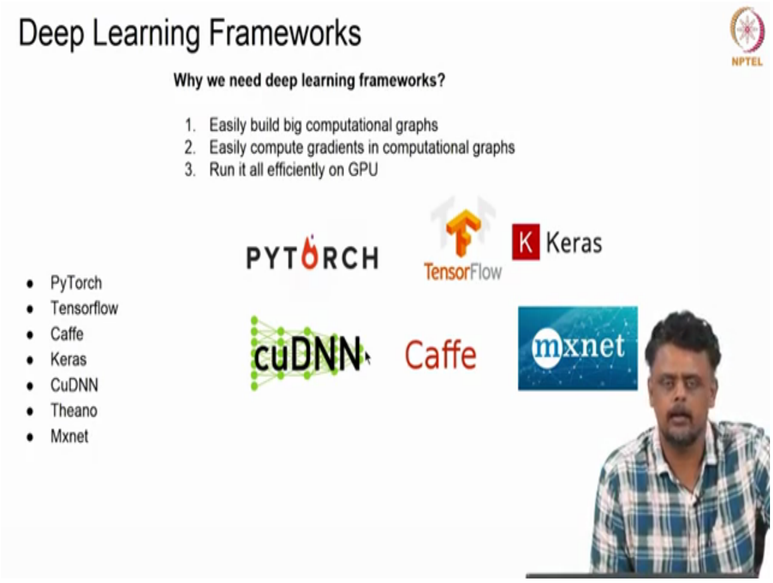
Scikit-learn allows to:

- Load and generate datasets
- Split them to perform cross-validation
- Easily apply learning algorithms
- Evaluate the performance of such algorithms

So Scikit learn is another module that comes with that can be installed in Python and it is a module that comes into Python and it requires both Python and the modules Numpy and Scipy. Scikit learn has a lot of the machine learning algorithms already implemented machine learning and computer vision algorithms available as part of it which we can just call like a function.

Once again Scikit learn comes with some excellent manuals and documentation and example code which you should try out, okay. So this is Python based so if you know programming in Python you should be able to use Scikit learn without any difficulty.

(Refer Slide Time: 2:22)



Deep Learning Frameworks

Why we need deep learning frameworks?

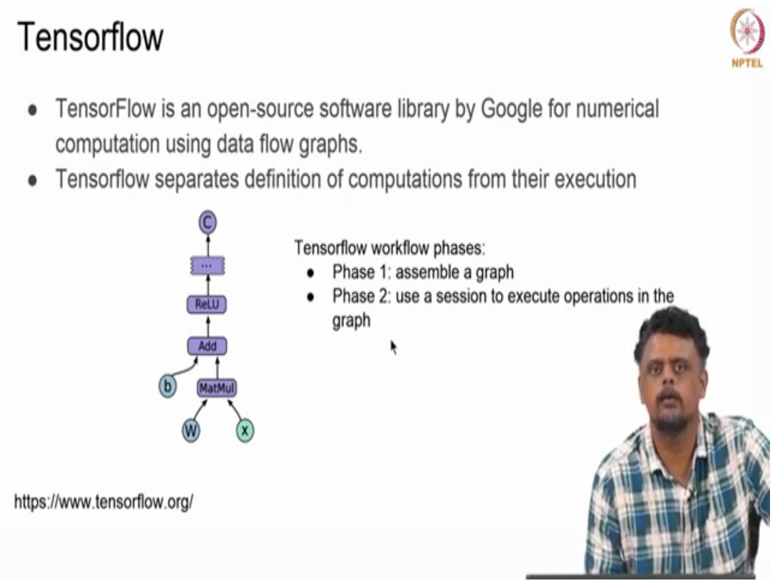
1. Easily build big computational graphs
2. Easily compute gradients in computational graphs
3. Run it all efficiently on GPU

- PyTorch
- Tensorflow
- Caffe
- Keras
- CuDNN
- Theano
- Mxnet

Okay, now we move on to some of the deep learning frameworks, so why do we need them? Probably because it helps you to easily implement in prototype deep learning algorithms, okay. So in general coding these algorithms is scratch even though it is a good exercise, it can distract you from your primary purpose, we expect that most of you are working in some engineering domain where you want to solve a particular problem and not necessarily solve a programming problem in this case implementing deep learning from scratch.

So there are multitude of solutions available, I have listed some of the more popular ones here, so PYTORCH, Tensor Flow, Keras Keras comes on top it is an API of Tensor Flow, cuDNN is offered by NVidia, Caffe and mxnet. So when some of these are offering some large companies with as freeware as open source software. So we are welcome to adapt any of them for your course, for your assignments in order to learn the concepts that we will provide you in this video.

(Refer Slide Time: 3:30)



Tensorflow

- TensorFlow is an open-source software library by Google for numerical computation using data flow graphs.
- Tensorflow separates definition of computations from their execution

Tensorflow workflow phases:

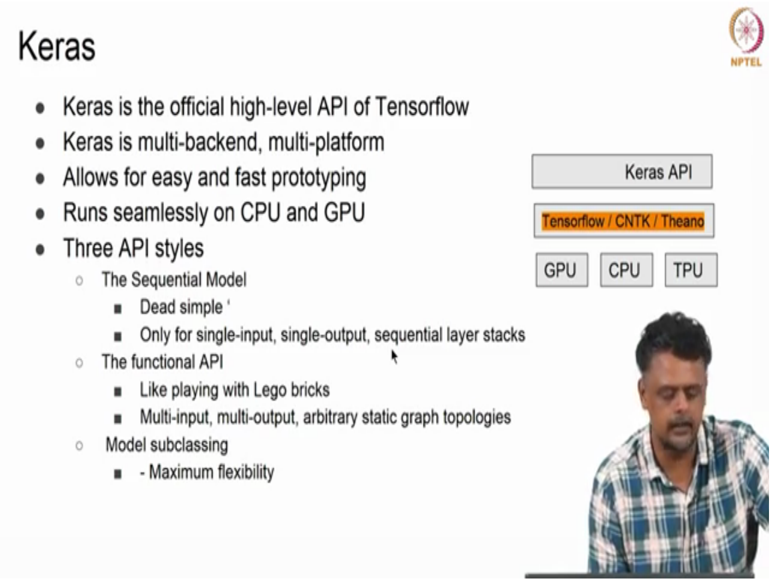
- Phase 1: assemble a graph
- Phase 2: use a session to execute operations in the graph

https://www.tensorflow.org/

So if you look at Tensorflow it is an open source library which is offered by Google, it is used for setting up data flow graphs. So for any neural network which you will see in the following weeks in deep neural networks you will see in the following weeks can be set up in the form of a graph. So Tensorflow works in the following fashion you assemble a graph, you basically define the nodes computational nodes in the graph and then you invoke a session to execute the computations in the nodes, okay.

So it is a very nifty tool for implementing deep neural networks, it is also very convenient because a lot of the more popular deep learning algorithms that you see (())(4:09) algorithms like (())(4:10) some architecture that works very well, many of the implementations or the architectures are available in through Tensorflow so it is free to download and you can learn to code just by looking at those examples.

(Refer Slide Time: 4:27)



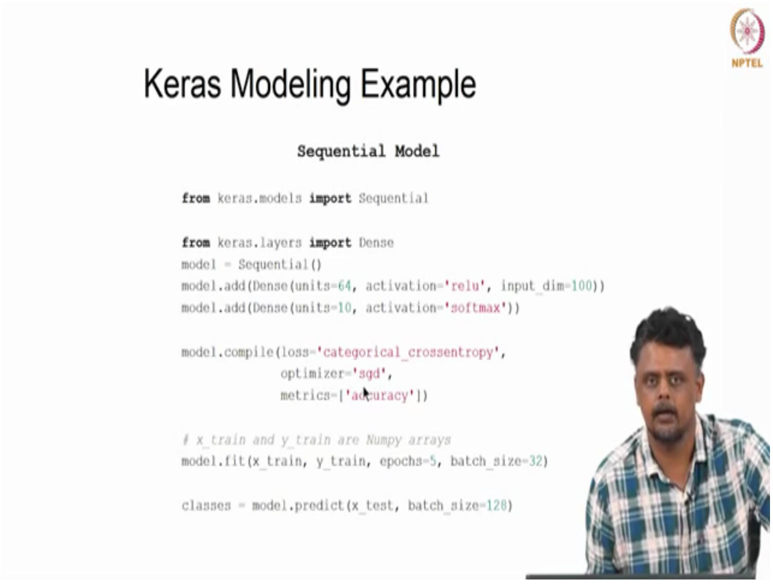
## Keras

- Keras is the official high-level API of Tensorflow
- Keras is multi-backend, multi-platform
- Allows for easy and fast prototyping
- Runs seamlessly on CPU and GPU
- Three API styles
  - The Sequential Model
    - Dead simple '
    - Only for single-input, single-output, sequential layer stacks
  - The functional API
    - Like playing with Lego bricks
    - Multi-input, multi-output, arbitrary static graph topologies
  - Model subclassing
    - - Maximum flexibility

Once again Keras is the official high level API for Tensorflow, again it greatly reduces the programming complexity involved because if you start coding even with Tensorflow even though Tensorflow makes coding easier specially for implementing some of the deep learning algorithms. Keras provide one more layer on top so that gives you a very simple way of implementing some of the more popular architectures or more conventional architectures that we see in deep learning.

So you will see support for sequential models for cuDNN like convolutional neural network like models and regular neural network models, okay and it provides you this functionally API where we can call functions at different points in the code, you can move them around like this Lego bricks.

(Refer Slide Time: 5:18)



## Keras Modeling Example

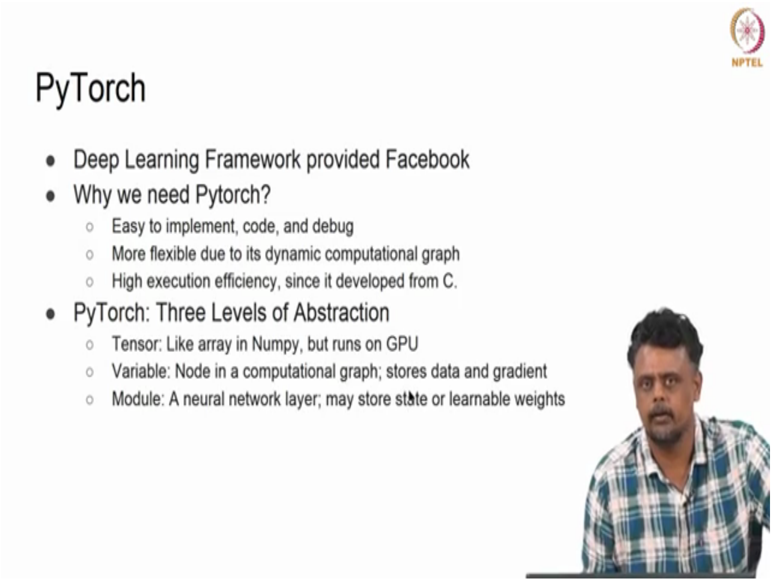
Sequential Model

```
from keras.models import Sequential

from keras.layers import Dense
model = Sequential()
model.add(Dense(units=64, activation='relu', input_dim=100))
model.add(Dense(units=10, activation='softmax'))

model.compile(loss='categorical_crossentropy',
              optimizer='sgd',
              metrics=['accuracy'])

# x_train and y_train are Numpy arrays
model.fit(x_train, y_train, epochs=5, batch_size=32)

classes = model.predict(x_test, batch_size=128)
```

So for instance here is a very simple script for implementing a typical sequential model neural network and see that it is accomplished in a very short piece of code, so some of these terminologies may not be very familiar to you but what is going on here is that it lets you define some of the what we call the computational aspects of some of the neural networks are defined very easily using this model, okay.

So even if you Keras even a very complicated neural deep learning model can be implemented in a very few lines of code it is very transparent. So that is one of the reasons why you would need some of these packages, right.

(Refer Slide Time: 6:00)



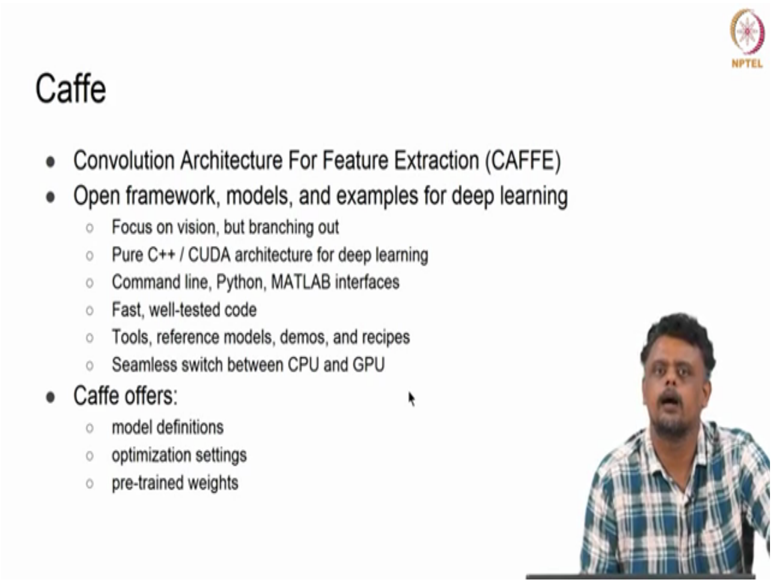
## PyTorch

- Deep Learning Framework provided Facebook
- Why we need Pytorch?
  - Easy to implement, code, and debug
  - More flexible due to its dynamic computational graph
  - High execution efficiency, since it developed from C.
- PyTorch: Three Levels of Abstraction
  - Tensor: Like array in Numpy, but runs on GPU
  - Variable: Node in a computational graph; stores data and gradient
  - Module: A neural network layer; may store state or learnable weights

We will move on to the next one which is PyTorch once again PyTorch, Tensorflow both are very popular among the deep learning community, both of them offer a lot of features that are very say very good for prototyping and both of them also in fact PyTorch also offers a lot of the more popular deep learning architectures already coded with the weights, etc we will see what this means later on, okay.

And PyTorch again gives you combine seamlessly with Python. So if we learn Python, PyTorch becomes very easy that is another advantage. In terms of capabilities again PyTorch and Tensorflow are quite similar in terms gets you core some of the deep learning algorithms that you will see in this course. Again they define variable slightly differently but rather than that again PyTorch is as good platform to start with as Tensorflow again and PyTorch is provided by Facebook as one of their open source implementations of deep learning algorithms.

(Refer Slide Time: 7:04)

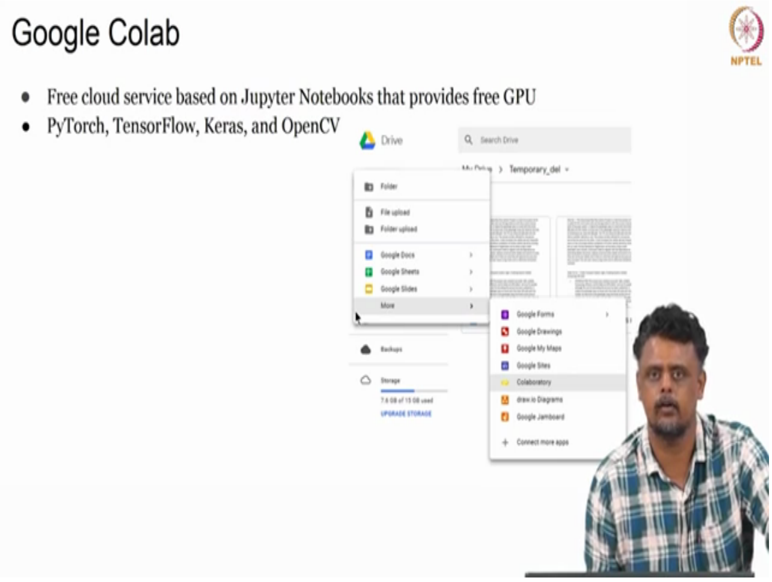


Caffe

- Convolution Architecture For Feature Extraction (CAFFE)
- Open framework, models, and examples for deep learning
  - Focus on vision, but branching out
  - Pure C++ / CUDA architecture for deep learning
  - Command line, Python, MATLAB interfaces
  - Fast, well-tested code
  - Tools, reference models, demos, and recipes
  - Seamless switch between CPU and GPU
- Caffe offers:
  - model definitions
  - optimization settings
  - pre-trained weights

So rather if you like to work with C plus plus when you like to code that way then Cafee is the package for you it is again a open source package from I think Berkeley and again it is written in C plus plus, initially done supported lot of convolutional neural network architecture but now I think it is branching out into deep learning (())(7:25) also. Once again this is again an open source version lot of support on the online community you can go online look for you know if you want to code something up how do you do it? there will be some support community there for Caffe and if you are some person who likes coding in C plus plus then probably this is the I think for you.

(Refer Slide Time: 7:44)



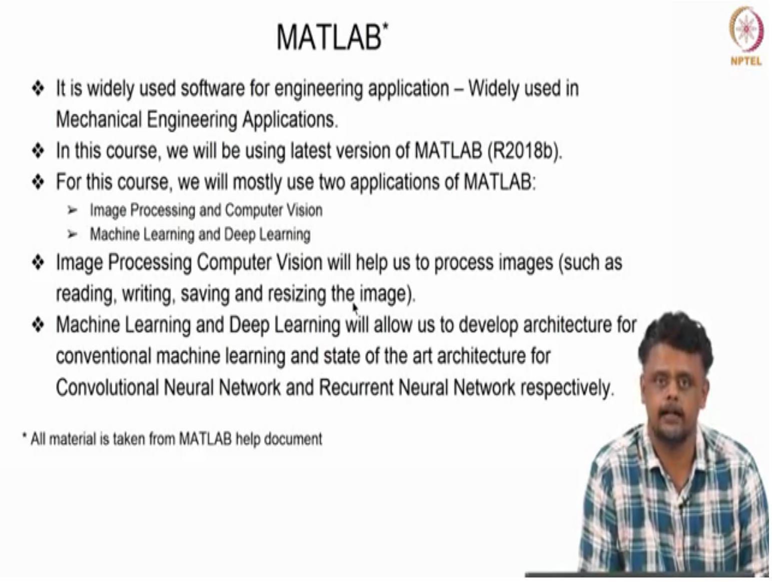
## Google Colab

- Free cloud service based on Jupyter Notebooks that provides free GPU
- PyTorch, TensorFlow, Keras, and OpenCV

Google Colab, again this is not exactly like the package that we talked about but Google Colab is, I would like for you to check it out, it provides you free cloud service in fact gives you access to free GPU's and what they call TPU's tensor processing units, it also supports PyTorch, Tensorflow, Keras and other open source software packages that you can use for implementing deep learning models.

Once again the attracted aspect of this is the availability of free computing power so from your very you know (())(8:13) laptop you can actually run it is slightly sophisticated code, so if you are programming in Python and want to try little bit more adventurous let us say you are using PyTorch, Tensorflow then this might be a good option for you, and again Colab is just the cloud computing architecture and not exactly a package.

(Refer Slide Time: 8:30)



MATLAB*

- It is widely used software for engineering application – Widely used in Mechanical Engineering Applications.
- In this course, we will be using latest version of MATLAB (R2018b).
- For this course, we will mostly use two applications of MATLAB:
  - Image Processing and Computer Vision
  - Machine Learning and Deep Learning
- Image Processing Computer Vision will help us to process images (such as reading, writing, saving and resizing the image).
- Machine Learning and Deep Learning will allow us to develop architecture for conventional machine learning and state of the art architecture for Convolutional Neural Network and Recurrent Neural Network respectively.

* All material is taken from MATLAB help document

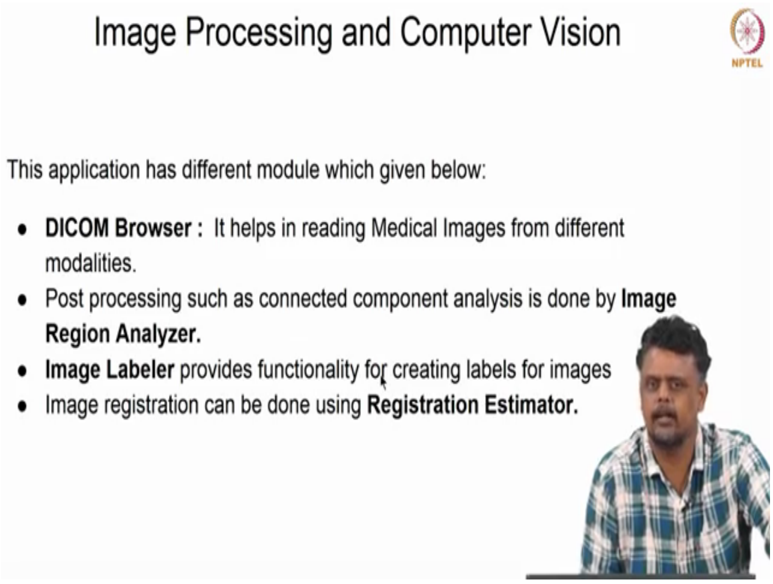
So finally we will come to MATLAB which is what we will use in this course primarily because MATLAB is providing us with most of the providing the students who are enrolled in this course the free account that we can log into so it is MATLAB online so you can it (()) (8:46) you the interface which is very simpler to the MATLAB desktop and you can try may of the deep learning and machine learning algorithms that we cover in this course, okay.

Another reason for choosing MATLAB is that we expect a lot of the students to sign up of this course to come from a variety of engineering disciplines and I am sure most of you are familiar with MATLAB which is the programming platform of choice for engineering students in general both in research and academics and as this one in the class room. So we will be using some of the tool boxes in the more recent version of MATLAB this is 2018b.

The computer vision tool box, there is a statistics and machine learning tool box as well as a deep learning tool box I think the deep learning tool box was formerly refers to as the neural network tool box. So it is being around for some time but now there is (())(9:37) interest, okay. So you are interested in machine vision problems or computer vision problems there are some handy algorithms already available that we can use again the non-deep learning machine learning algorithms are also available for you to try and test out, they also provide you some datasets which you can load in and play with those, okay.

In addition, the deep learning tool box offer support for very easy to use interface for developing convolutional neural network architectures LSTM things like that.

(Refer Slide Time: 10:14)



## Image Processing and Computer Vision

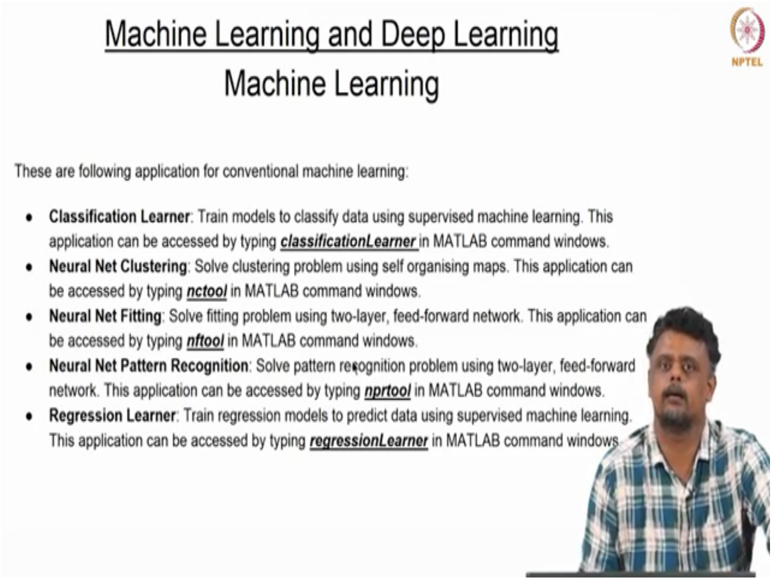
This application has different module which given below:

- **DICOM Browser :** It helps in reading Medical Images from different modalities.
- Post processing such as connected component analysis is done by **Image Region Analyzer.**
- **Image Labeler** provides functionality for creating labels for images
- Image registration can be done using **Registration Estimator.**

What appeals to me personally is that they have a very nice module for accessing medical images so many of the medical images that are generated or stored in something called a DICOM format in MATLAB allows you gives you the software routines to read those images into the memory, okay. So once again a lot of the post processing that you will do on images we will see later on in the course is also available already coded in the form of some function easy to call functions in MATLAB specially for medical image analysis.

Again image registration if someone who works in image processing then it has a very nice tool box image registration tool box or setup routine for doing image registration which again is a integral part of medical image processing.

(Refer Slide Time: 11:02)



## Machine Learning and Deep Learning
## Machine Learning

These are following application for conventional machine learning:

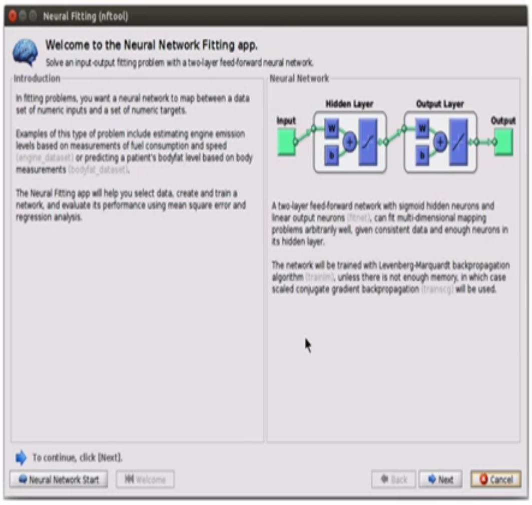
- **Classification Learner**: Train models to classify data using supervised machine learning. This application can be accessed by typing ***classificationLearner*** in MATLAB command windows.
- **Neural Net Clustering**: Solve clustering problem using self organising maps. This application can be accessed by typing ***nctool*** in MATLAB command windows.
- **Neural Net Fitting**: Solve fitting problem using two-layer, feed-forward network. This application can be accessed by typing ***nftool*** in MATLAB command windows.
- **Neural Net Pattern Recognition**: Solve pattern recognition problem using two-layer, feed-forward network. This application can be accessed by typing ***nprtool*** in MATLAB command windows.
- **Regression Learner**: Train regression models to predict data using supervised machine learning. This application can be accessed by typing ***regressionLearner*** in MATLAB command windows.

Once again there are other resources for conventional machine learning algorithms, okay they are listed here I will not go through all of them but what you are trying to say is that many of them have are command line interface, okay so you can do them from the MATLAB command line and for this course we will be given access to online MATLAB license and you can try this commands in that browser also. So that is a very convenient thing that MATLAB allows you to do.

In addition, they also have the nftool which actually provides you the GUI for creating deep learning algorithms.

(Refer Slide Time: 11:42)

Neural Net Fitting (nftool)



Welcome to the Neural Network Fitting app.

Solve an input-output fitting problem with a two-layer feed-forward neural network.

Introduction

In fitting problems, you want a neural network to map between a data set of numeric inputs and a set of numeric targets.

Examples of this type of problem include estimating engine emission levels based on measurements of fuel consumption and speed (engine_dataset) or predicting a patient's bodyfat level based on body measurements (bodyfat_dataset).

The Neural Fitting app will help you select data, create and train a network, and evaluate its performance using mean square error and regression analysis.

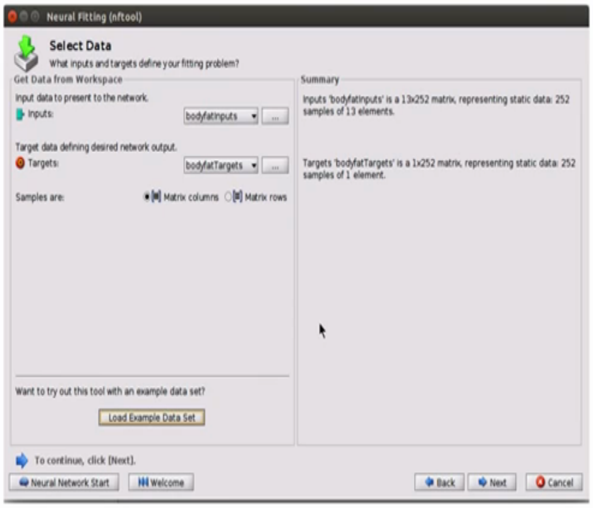
Neural Network

A two-layer feed-forward network with sigmoid hidden neurons and linear output neurons (fitnet), can fit multi-dimensional mapping problems arbitrarily well, given consistent data and enough neurons in its hidden layer.

The network will be trained with Levenberg-Marquardt backpropagation algorithm (trainlm), unless there is not enough memory, in which case scaled conjugate gradient backpropagation (trainscg) will be used.

To continue, click [Next].

Loading the example dataset



Select Data

What inputs and targets define your fitting problem?

Get Data from Workspace

Input data to present to the network.

Inputs: bodyfatInputs

Target data defining desired network output.

Targets: bodyfatTargets

Samples are: ● Matrix columns ○ Matrix rows

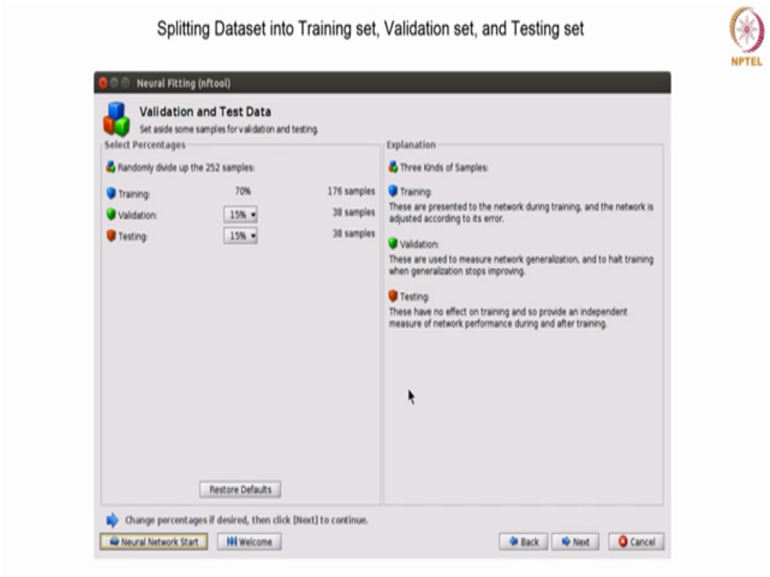
Want to try out this tool with an example data set?

Summary

Inputs 'bodyfatInputs' is a 13x252 matrix, representing static data: 252 samples of 13 elements.

Targets 'bodyfatTargets' is a 1x252 matrix, representing static data: 252 samples of 1 element.

To continue, click [Next].



Splitting Dataset into Training set, Validation set, and Testing set

Neural Fitting (nftool)

Validation and Test Data

Set aside some samples for validation and testing.

Select Percentages

Randomly divide up the 252 samples:

| | | |
|---|---|---|
| Training: | 70% | 176 samples |
| Validation: | 15% | 38 samples |
| Testing: | 15% | 38 samples |

Explanation

Three Kinds of Samples:

Training:
These are presented to the network during training, and the network is adjusted according to its error.

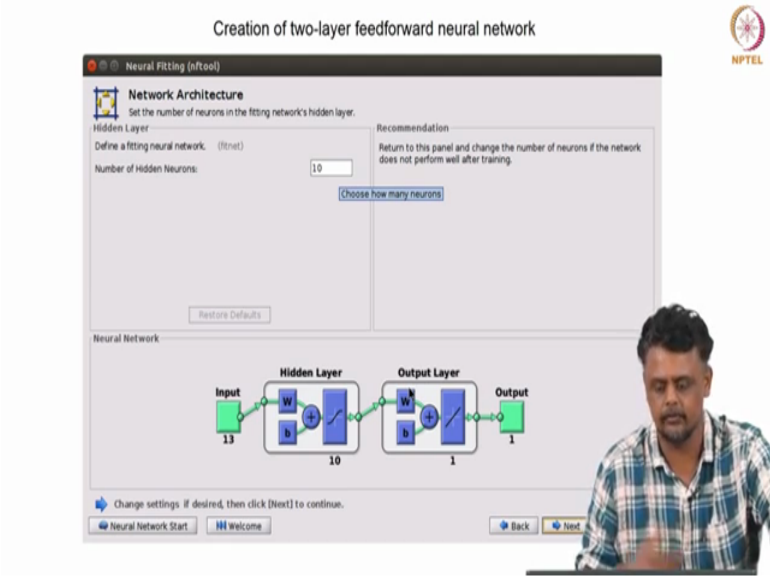
Validation:
These are used to measure network generalization, and to halt training when generalization stops improving.

Testing:
These have no effect on training and so provide an independent measure of network performance during and after training.

Change percentages if desired, then click [Next] to continue.

So we will look at some of them so for instance if you load up enough tool you will come up with a interface like this where you can define your deep neural network. So for instance you can say where the data is in this case this is the dialog that leads you load data which you have already stored in some format, okay. Once you have loaded the data set then you can divide it into training, testing and validation so we can give you it automatically tells so many samples there are and how many do you want in terms of training and testing and validation data.

(Refer Slide Time: 12:12)



Creation of two-layer feedforward neural network

Neural Fitting (nftool)

Network Architecture

Set the number of neurons in the fitting network's hidden layer.

Hidden Layer

Define a fitting neural network. (fitnet)

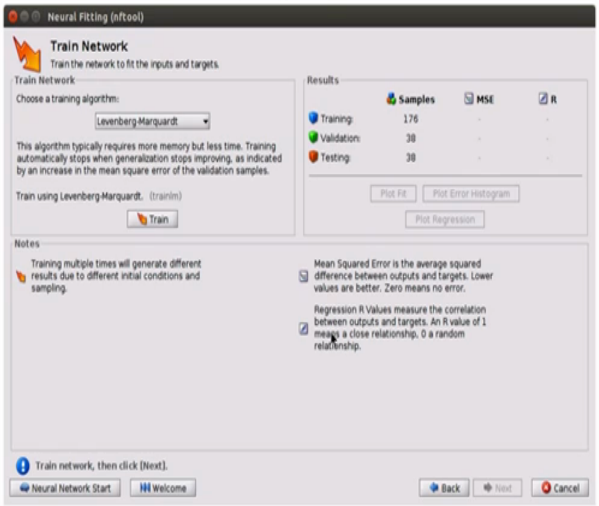
Number of Hidden Neurons: 10

Recommendation

Return to this panel and change the number of neurons if the network does not perform well after training.

Change settings if desired, then click [Next] to continue.

Training of network



Neural Fitting (nftool)

**Train Network**

Train the network to fit the inputs and targets.

Train Network

Choose a training algorithm:

Levenberg-Marquardt

This algorithm typically requires more memory but less time. Training automatically stops when generalization stops improving, as indicated by an increase in the mean square error of the validation samples.

Train using Levenberg-Marquardt. (trainlm)

Results

| | Samples | MSE | R |
|---|---|---|---|
| Training: | 176 | - | - |
| Validation: | 38 | - | - |
| Testing: | 38 | - | - |

Notes

Training multiple times will generate different results due to different initial conditions and sampling.

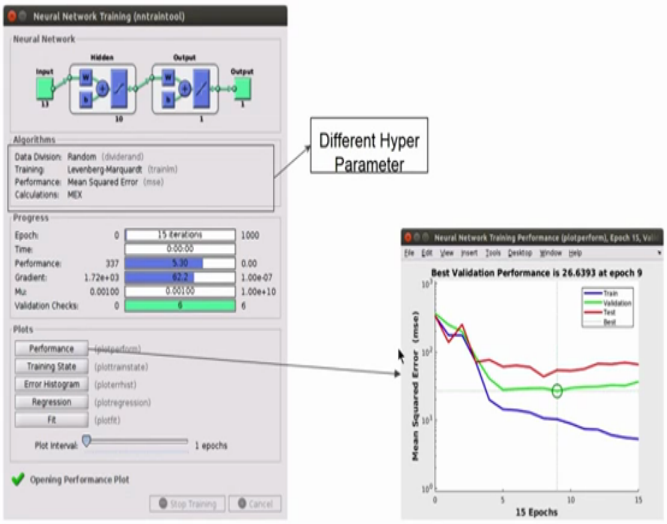
Mean Squared Error is the average squared difference between outputs and targets. Lower values are better. Zero means no error.

Regression R Values measure the correlation between outputs and targets. An R value of 1 means a close relationship, 0 a random relationship.

Train network, then click [Next].

And then of course you can also use the same dialogue box to define neural network architectures that how many intend layers etc. Again if you start with these terminologies are not familiar to you just have to wait another week or so and you will be fine. So we just go walking you through some of the more easier aspects of using MATLAB here. And again it is very simple to choose the training algorithm which is again you are familiar with one by now gradient descent is the training algorithm that you have learnt right now. So here it leads you some other optimization technique as a training algorithm.

(Refer Slide Time: 12:44)



Neural Network Training (nntraintool)

Neural Network

Algorithms

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

Progress

| | | | |
|---|---|---|---|
| Epoch: | 0 | 15 iterations | 1000 |
| Time: | | 0:00:00 | |
| Performance: | 337 | 5.30 | 0.00 |
| Gradient: | 1.72e+03 | 62.2 | 1.00e-07 |
| Mu: | 0.00100 | 0.00100 | 1.00e+10 |
| Validation Checks: | 0 | 6 | 6 |

Plots

Performance (plotperform)
Training State (plottrainstate)
Error Histogram (ploterrhist)
Regression (plotregression)
Fit (plotfit)

Plot Interval: 1 epochs

Opening Performance Plot

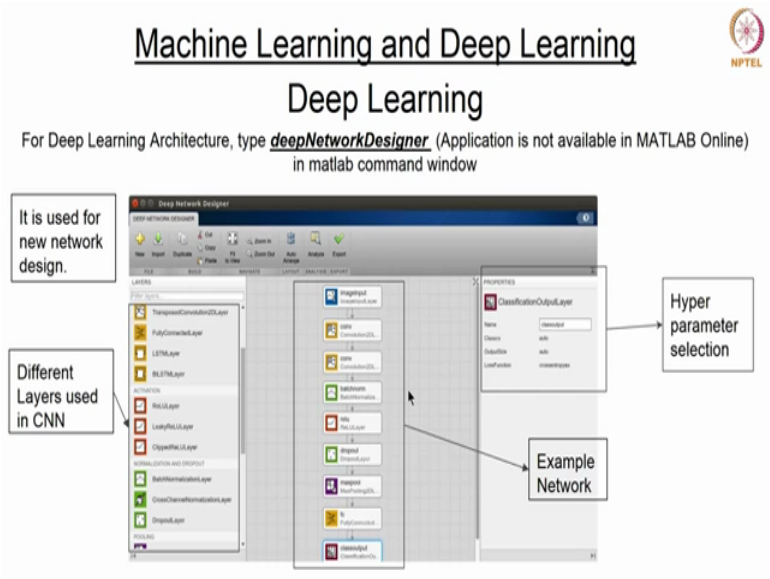
Different Hyper Parameter

Neural Network Training Performance (plotperform), Epoch 15, Vali

Best Validation Performance is 26.6393 at epoch 9

On top of that you can see the performance in the training, in the form of graphs it tells you what the training and testing validation accuracy are, define various hyper parameters, monitor them, etc using the same GUI.

(Refer Slide Time: 12:58)



Machine Learning and Deep Learning

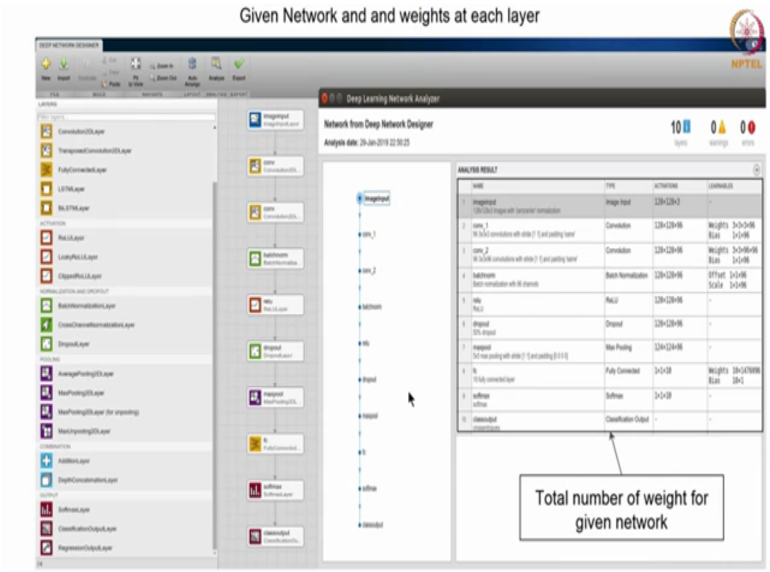
Deep Learning

For Deep Learning Architecture, type ***deepNetworkDesigner*** (Application is not available in MATLAB Online) in matlab command window

It is used for new network design.
Different Layers used in CNN
Hyper parameter selection
Example Network

Once again for deep neural networks what we saw till so far is the conventional neural network, for deep neural network again lets you access the various layers in a network in the form of this module that you can drag and drop to create your own. So once you created that it generates code automatically that you can go and edit, so that is the idea behind using this GUI for creating deep neural networks.

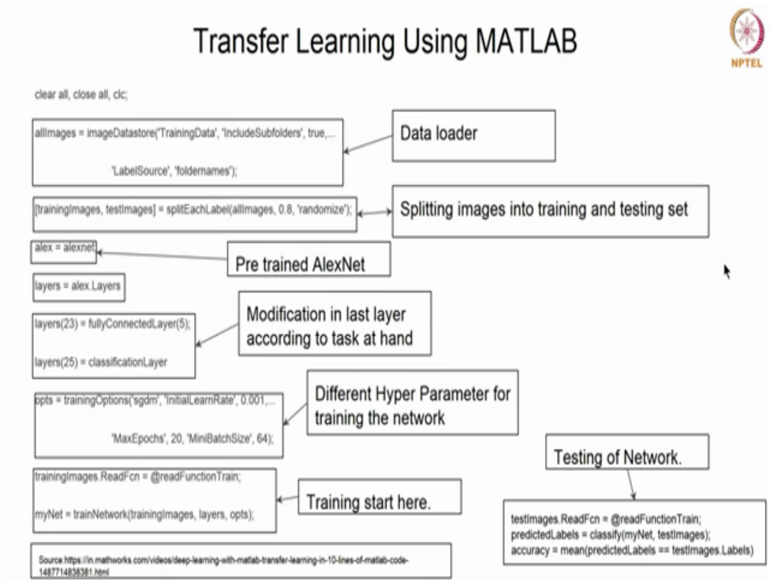
(Refer Slide Time: 13:19)



Given Network and and weights at each layer

Again I will not go through all the details but what this GUI shows you is that you can define so you can define different layers here and you can examine what each of these layers are on the right hand side tool box. So it is like drag and drop that allows you that functionality, okay.

(Refer Slide Time: 13:38)



Transfer Learning Using MATLAB

```
clear all, close all, clc;

allImages = imageDatastore('TrainingData', 'IncludeSubfolders', true,...
                'LabelSource', 'foldernames');

[trainingImages, testImages] = splitEachLabel(allImages, 0.8, 'randomize');

alex = alexnet

layers = alex.Layers

layers(23) = fullyConnectedLayer(5);

layers(25) = classificationLayer

opts = trainingOptions('sgdm', 'InitialLearnRate', 0.001,...
                'MaxEpochs', 20, 'MiniBatchSize', 64);

trainingImages.ReadFcn = @readFunctionTrain;

myNet = trainNetwork(trainingImages, layers, opts);

testImages.ReadFcn = @readFunctionTrain;
predictedLabels = classify(myNet, testImages);
accuracy = mean(predictedLabels == testImages.Labels)
```

Data loader
Splitting images into training and testing set
Pre trained AlexNet
Modification in last layer according to task at hand
Different Hyper Parameter for training the network
Training start here.
Testing of Network.
NPTEL

Source https://in.mathworks.com/videos/deep-learning-with-matlab-transfer-learning-in-10-lines-of-matlab-code-1487714838381.html

So again transfer learning thinks we are too far ahead at this point so I will not really go in detail in this particular slide but transfer learning is allowed in the sense that you have a lot of the large popular convolutional networks and deep neural networks already available as part of MATLAB which we can reuse in order for your limited data set. So in this course we will we expect you to use MATLAB primarily because MATLAB is supporting this course, it supports this course by providing all of the resistance access to online MATLAB license which provides you the interface very similar to a desktop interface.

It also provides you 250 megabytes of hard drive data so you can upload your own data, of course not very large data sets but reasonably large for most of the task that you will be doing and they also provide you support in sense that if you have trouble accessing etc they provide you support for that also. So it takes away a lot of the troubles you might face if you were to find figure out entire platform for yourself having said that, we are platform agnostic to some fat degree and we are okay in fact we do personally we have used many of the different software platforms.

So we are okay if you want to pursue any of the other packages that you have mentioned, but please note that it is difficult for us to provide support in any form if you have any trouble with the platform itself, okay. So but MATLAB in this case can provide you limited support if you have trouble with that platform, so that is the possibility. So I hope that we have given you a broad overview of the different options available for you to learn to code, we live it to you to choose the best option whatever suits you.

Please remember that there is nothing wrong in starting with MATLAB because it is easy to start that way provides you the lot and specially if you are familiar with MATLAB already it is okay to start with MATLAB learn and because many of the programming concepts are similar across the board so it is not too hard to switch to other platforms in the future if the need arises, thank you.