# MACHINE LEARNING

(​Predicting of Hand Gesture Recognition) Summer Internship Report Submitted in partial fulfilment of the requirement for undergraduate degree of

**Bachelor of Technology**​

​ In​

**Computer Science Engineering**​

​ By​

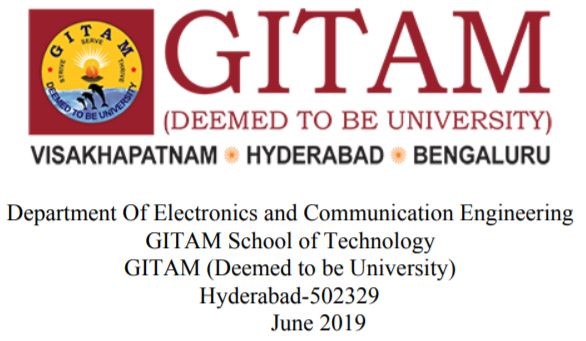
**POLU NANDINI REDDY**

**221710315036**

​​Under the Guidance of

**Mr…………………..**

Assistant Professor​



i​

# DECLARATION

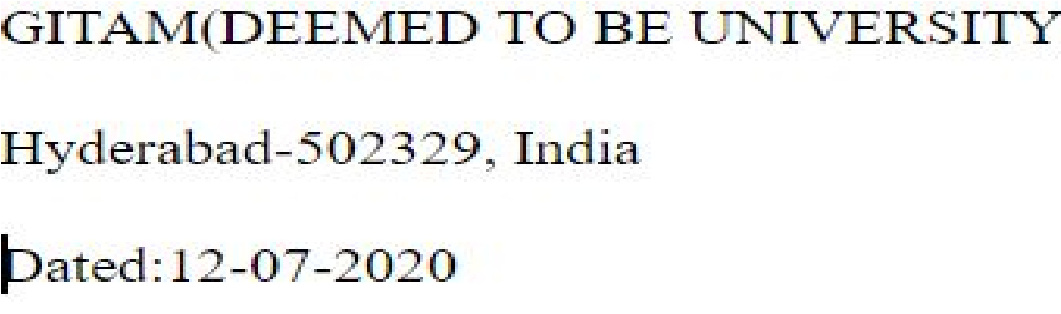
I submit this industrial training work entitled ​**“HAND GESTURE RECOGNITION​”**​to GITAM (Deemed To Be University) , Hyderabad in partial fulfilment of the requirements for the award of the degree of ​**“Bachelor of Technology ”**​ in ​**“ Computer Science Engineering ”**​. I declare that it was carried out independently by me under the guidance of ​**Mr……………..**​, Asst. Professor, GITAM (Deemed To Be University), Hyderabad, India.

The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Place: HYDERABAD POLU NANDINI REDDY

Date:22-07-2020 221710315036

ii​



## ​CERTIFICATE

This is to certify that the Industrial Training Report entitled **“HAND GESTURE RECOGNITION ”**is being submitted by POLU NANDINI REDDY(221710315036)​ in partial fulfilment of the requirement for the award of **Bachelor of Technology in Computer**​ **Science Engineering** at GITAM (Deemed To Be University), Hyderabad during the academic​year 2020-21 It is faithful record work carried out by her at the **Computer Science** **Engineering Department,** GITAM University Hyderabad Campus under my guidance and​ supervision.

### Mr……………………. ​ Dr.S.Phani Kumar​

Assistant Professor Professor and HOD

Department of CSE Department of CSE

**ACKNOWLEDGEMENT**

Apart from my effort, the success of this internship largely depends on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have helped me in the successful competition of this internship.

I would like to thank respected **Dr. N. Siva Prasad**​ , Pro Vice Chancellor, GITAM​ Hyderabad and**Dr. CH. Sanjay**​, Principal, GITAM Hyderabad​.

I would like to thank respected **Dr.S.Phani Kumar**​ , Head of the Department of​ Computer Science Engineering for giving me such a wonderful opportunity to expand my knowledge for my own branch and giving me guidelines to present an internship report. It helped me a lot to realize what we study for.

I would like to thank the respected faculties **Mr. ………………..**​ who helped me to​ make this internship a successful accomplishment.

I would also like to thank my friends who helped me to make my work more organized and well-stacked till the end.

POLU NANDINI REDDY

221710315036

# ABSTRACT

Machine learning algorithms are used to predict the values from the​ data set by splitting the data set in to train and test and building Machine learning algorithms models of higher accuracy to predict the values is the primary task to be performed on the Train and Test data set. My perception of understanding the given data sets has been in the view of undertaking different activities performed by 30 subjects and prediction of the activity.

To get a better understanding and work on a strategic approach for solution of the client, I have adapted to look at the activities performed by plotting few graphs to understand it in a better way,for further deep understanding of the problem, I have taken the describe function to know more about my data set , and my primary objective of this case study was to predict the activity performed with at most accuracy,so i have used several algorithms to know which one yields the highest accuracy to predict my outcome.

**Table of Contents:**

**LIST OF FIGURES**

## CHAPTER 1:MACHINE LEARNING

1.1 INTRODUCTION…………………………………………………………10

1.2 IMPORTANCE OF MACHINE LEARNING…………………………….10

1.3 USES OF MACHINE LEARNING……………………………………….12

1.4 TYPES OF LEARNING ALGORITHMS………………………………...12

1.4.1 Supervised Learning……………………………………………. 12

1.4.2 Unsupervised Learning………………………………………….13

1.4.3 Semi Supervised Learning………………………………………14

1.5 RELATION BETWEEN DATA MINING,MACHINE LEARNING AND

DEEP LEARNING………………………………………………………………………….…15

## CHAPTER 2:PYTHON……...……………………………………………….………………16

2.1 INTRODUCTOIN TO PYTHON…………………………………….……16

2.2 HISTORY OF PYTHON……………………………………………….….16

2.3 FEATURES OF PYTHON………………………………………………...16

2.4 HOW TO SETUP PYTHON……………………………………………....17

2.4.1 Installation(using python IDLE)………………………………...17

2.4.2 Installation(using Anaconda)……………………………………18

2.5 PYTHON VARIABLE TYPES…………………………………………….20

2.5.1 Python Numbers………………………………………………. 21

2.5.2 Python Strings………………………………………………….21

2.5.3 Python Lists…………………………………………………….22

2.5.4 Python Tuples………………………………………………….22

2.5.5 Python Dictionary……………………………………………...23

2.6 PYTHON FUNCTION……………………………….…….......…………...24

2.6.1 Defining a Function…………………………………………….24

2.6.2 Calling a Function………………………………………………24

2.7 PYTHON USING OOP’s CONCEPTS………….………………………….25

2.7.1 Class…………………………………………………………….25

2.7.2 \_init\_ method in class………………………………………...26

## CHAPTER 3:CASESTUDY..………..…………………………………………………...27

3.1​ PROBLEM STATEMENT………………………………………………….27

3.2 DATA SET………………………………………………………..………...27 3.3 OBJECTIVE OF THE CASE STUDY……..…………………………................…….27

## CHAPTER 4:MODEL BUILDING…..…………………………………………….…..28

4.1​ PREPROCESSING OF THE DATA……………….......................….………………….28

4.1.1 UPLOAD FILE ………………………..........................…………………….. 28

4.1.2 IMPORTING THE DATA SET FROM KAGGLE …………………………….29

4.1.3 IMPORTING THE LIBRARIES …………………………………………..30

4.1.4 CREATING FEATURES & TARGET VARIABLES …………………………32

4.2 DISPLAYING IMAGES WITH LABES ………………………................................….33

4.2.1 CONVERT FEATURES LIST INTO A NUMPY ARRAY .............................. 34

4.2.2 CREATING TRAIN, VALIDATION AND TEST SET ........................…….. 34

4.3 DEFINING MODEL ARCHITECTURE ……………………………………….34

4.3.1 INSTANTIATING CALLBACKS……….....…….……………..........35

4.3.2 DATA AUGMENTATION……………………………………………..36

4.3.3 COMPILING THE MODEL .............................................……………36

4.3.4 TRAININGTHE MODEL .............................................……........……37

4.3.5 MODEL PERFORMANCE .............................................……………39

4.3.6 PREDICTIONS ON TEST SET.............................................……………40

4.3.7CONFUSION MATRIX & ACCURACY SCORE ...................……………41

4.4 CONCLUSION……………………………………........................................41

**LIST OF FIGURES:**

Figure 1 : The Process Flow……………………..……………………………….……...12

Figure 2 : Unsupervised Learning………………………..………………………………15

Figure 3 : Semi Supervised Learning……………………………………..……………...16

Figure 4 : Python download…………………….…....…………………………………...17

Figure 5 : Anaconda download…………....…….………………………………………..21

Figure 6 :Jupyter notebook……………………….……………………………………...21

Figure 7 : Defining a Class………………………..……………………...……………....28

Figure 8 : Installing kaggle……………………..………………………....…………..28

Figure 9 : importing files from googlecolab ………..…………………………………..28

Figure 10 : creating directory ...........………………...……………………..…………..29

Figure 11 : choosing file from kaggle ..………...………...…………………………….. 29

Figure 12 : download the dataset …..…………....………………………………………...29

Figure 13 : extracting zip file ……….……………………..…..…..………………...30

Figure 14 : importing required libraries .……………………………………………..32

Figure 15 : creating feature and target variables ............………………………………32

Figure 16 : displaying images with labels ………………………………………...…. 33

Figure 17 : convert features list into a numpy array …………..………………….34

Figure 18 : creating train, validation and test sets …...…………....………………………….34

Figure 19 : defining model architecture ……………………………..……………………….35

Figure 20 : instantiating callbacks …………………...…………………………….35

Figure 21 : data augmentation………...……………………………………………………36

Figure 22 : compiling the model ...................…………………...…………...………..36

Figure 23 : model summary ………….……………………………………..……….37

Figure 24 : model layers …………………..…………..………………...37

Figure 25 : printing epoches …………………………………………….38

Figure 26 : ploting epochs accuracy graphs …….…………………………………......39

Figure 27 : ploting epochs loss graphs ……….………………………………….……..40

Figure 28 : prediction on test ...........................………………….…………….………40

Figure 29 : confusion matrix & accuracy score ………………….…………….………41

**CHAPTER 1**

**MACHINE LEARNING**

# 1.1​INTRODUCTION​ :​

Machine Learning(ML) is the scientific study of algorithms and statistical models that computer systems use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of Artificial Intelligence(AI).

# 1.2​IMPORTANCE OF MACHINE LEARNING:​

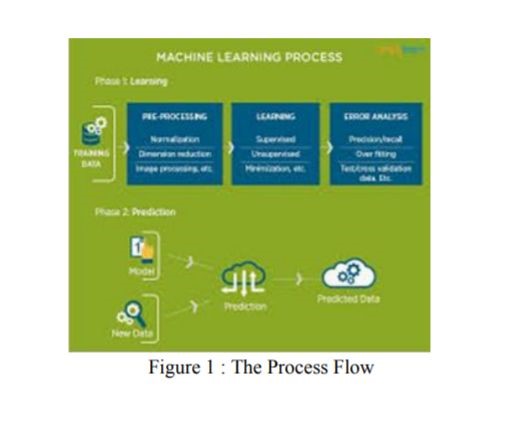
​Consider some of the instances where machine learning is applied: the self-driving Google car, cyber fraud detection, online recommendation engines—like friend suggestions on Facebook, Netflix showcasing the movies and shows you might like, and “more items to consider” and “get yourself a little something” on Amazon—are all examples of applied machine learning. All these examples echo the vital role machine learning has begun to take in today’s data-rich world.

Machines can aid in filtering useful pieces of information that help in major advancements, and we are already seeing how this technology is being implemented in a wide variety of industries.

With the constant evolution of the field, there has been a subsequent rise in the uses, demands, and importance of machine learning. Big data has become quite a buzzword in the last few years; that’s in part due to increased sophistication of machine learning, which helps analyze those big chunks of big data.

Machine learning has also changed the way data extraction, and interpretation is done by involving automatic sets of generic methods that have replaced traditional statistical techniques.

The process flow depicted here represents how machine learning works



# 1.3​ USES OF MACHINE LEARNING:​

Earlier in this article, we mentioned some applications of machine learning. To understand the concept of machine learning better, let’s consider some more examples: web search results, real-time ads on web pages and mobile devices, email spam filtering, network intrusion detection, and pattern and image recognition. All these are by-products of applying machine learning to analyze huge volumes of data

Traditionally, data analysis was always being characterized by trial and error, an approach that becomes impossible when data sets are large and heterogeneous. Machine learning comes as the solution to all this chaos by proposing clever alternatives to analyzing huge volumes of data.

By developing fast and efficient algorithms and data-driven models for real-time processing of data, machine learning can produce accurate results and analysis

# 1.4​TYPES OF LEARNING ALGORITHMS:​

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

**1.4.1**​**Supervised Learning .**​

When an algorithm learns from example data and associated target

responses that can consist of numeric values or string labels, such as classes or tags, in

order to later predict the correct response when posed with new examples comes under the category of supervised learning.

Supervised machine learning algorithms uncover insights, patterns, and

relationships from a labelled training dataset – that is, a dataset that already contains a known value for the target variable for each record. Because you provide the machine learning algorithm with the correct answers for a problem during training, it is able to “learn” how the rest of the features relate to the target, enabling you to uncover insights and make predictions about future outcomes based on historical data.

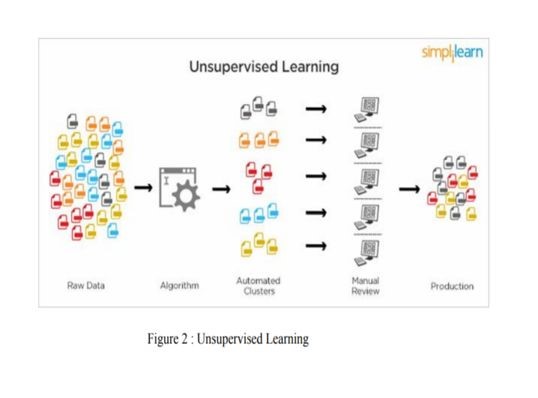
Examples of Supervised Machine Learning Techniques are Regression, in which the algorithm returns a numerical target for each example, such as how much revenue will be generated from a new marketing campaign.

Classification, in which the algorithm attempts to label each example by choosing between two or more different classes. Choosing between two classes is called binary classification, such as determining whether or not someone will default on a loan. Choosing between more than two classes is referred to as multiclass classification.

**1.4.2**​**Unsupervised Learning:**​

When an algorithm learns from plain examples without any associated response, leaving to the algorithm to determine the data patterns on its own.

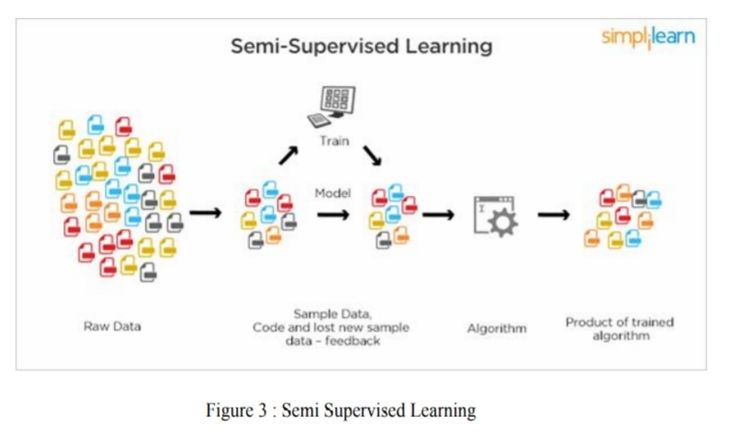
This type of algorithm tends to restructure the data into something else, such as new features that may represent a class or a new series of uncorrelated values. They are quite useful in providing humans with insights into the meaning of data and new useful inputs to supervised machine learning algorithms



Popular techniques where unsupervised learning is used also include self-​ organizing maps, nearest neighbor mapping, singular value decomposition, and k-means clustering. Basically, online recommendations, identification of data outliers, and segment text topics are all examples of unsupervised learning.

**1.4.3**​**Semi Supervised Learning:**​

As the name suggests, semi-supervised learning is a bit of both supervised and unsupervised learning and uses both labelled and un labelled data for training. In a typical scenario, the algorithm would use a small amount of labelled data with a large amount of un labelled data.



**1.5**​**RELATION BETWEEN DATA MINING,MACHINE**​  **LEARNING AND DEEP LEARNING:**

Machine learning and data mining use the same algorithms and techniques as data mining, except the kinds of predictions vary. While data mining discovers previously unknown patterns and knowledge, machine learning reproduces known patterns and knowledge—and further automatically applies that information to data, decision-making, and actions.

Deep learning, on the other hand, uses advanced computing power and special 5 types of neural networks and applies them to large amounts of data to learn, understand, and identify complicated patterns. Automatic language translation and medical diagnoses are examples of deep learning.

**CHAPTER 2**

# PYTHON

Basic programming language used for machine learning is : PYTHON

**2.1 INTRODUCTION TO PYHTON:**

* Python is a high-level, interpreted, interactive and object-oriented scripting language.
* Python is a general purpose programming language that is often applied in scripting roles
* Python is Interpreted: Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is like PERL and PHP.
* Python is Interactive: You can sit at a Python prompt and interact with the interpreter directly to write your programs.
* Python is Object-Oriented: Python supports the Object-Oriented style or technique of programming that encapsulates code within objects.

**2.2 HISTORY OF PYTHON:**

* Python was developed by GUIDO VAN ROSSUM in early 1990’s
* Its latest version is 3.7 , it is generally called as python3

**2.3**​  **FEATURES OF PYTHON:**

* Easy-to-learn: Python has few keywords, simple structure, and a clearly defined syntax, This allows the student to pick up the language quickly.
* Easy-to-read: Python code is more clearly defined and visible to the eyes.
* Easy-to-maintain: Python's source code is fairly easy-to-maintaining.
* A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* Databases: Python provides interfaces to all major commercial databases.
* GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

**2.4 HOW TO SETUP PYTHON:**

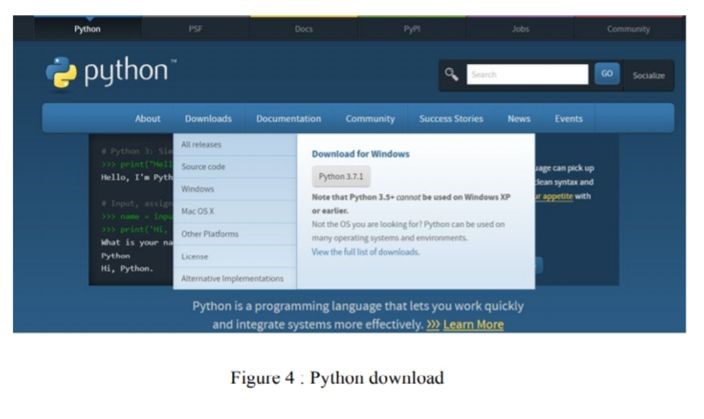
* Python is available on a wide variety of platforms including Linux and Mac OS X.

Let's understand how to set up our Python environment.

* The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python.

**2.4.1 Installation(using python IDLE):**

* Installing python is generally easy, and nowadays many Linux and Mac OS distributions include a recent python.
* Download python from www.python.org
* When the download is completed, double click the file and follow the instructions to install it.
* When python is installed, a program called IDLE is also installed along with it. It provides a graphical user interface to work with python.



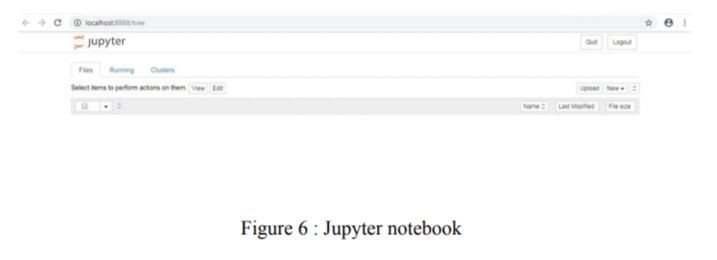
**2.4.2 Installation(using Anaconda):**

* Python programs are also executed using Anaconda.​
* Anaconda is a free open source distribution of python for large scale data processing, predictive analytics and scientific computing.
* Conda is a package manager quickly installs and manages packages.
* In WINDOWS:
* In window:
* Step 1: Open Anaconda.com/downloads in web browser.
* Step 2: Download python 3.4 version for (32-bitgraphic installer/64 -bit graphic installer)
* Step 3: select installation type( all users)
* Step 4: Select path(i.e. add anaconda to path & register anaconda as

default python 3.4) next click install and next click finish

* Step 5: Open jupyter notebook ( it opens in default browser)





**2.5 PYTHON VARIABLE TYPES:**

* Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

* Variables are nothing but reserved memory locations to store values.
* Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory.

* Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable.

* Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.
* Python has five standard data types –

* + - * + Number. o Strings o Lists o Tuples
        + Dictionary

**2.5.1**​  **Python Numbers:**

* Number data types store numeric values. Number objects are created when you assign a value to them.
* Python supports four different numerical types − int (signed integers) long (long integers, they can also be represented in octal and hexadecimal) float (floating point real values) complex (complex numbers).

**2.5.2 Python Strings:**

* Strings in Python are identified as a contiguous set of characters represented in the quotation marks.
* Python allows for either pairs of single or double quotes.
* Subsets of strings can be taken using the slice operator ([ ] and [:] ) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.
* The plus (+) sign is the string concatenation operator and the asterisk (\*) is the repetition operator.

**2.5.3**​  **Python Lists:**

* + - Lists are the most versatile of Python's compound data types.

* + - A list contains items separated by commas and enclosed within square brackets

([]).

* + - To some extent, lists are similar to arrays in C. One difference between them is

that all the items belonging to a list can be of different data type.

* + - The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1.
    - The plus (+) sign is the list concatenation operator, and the asterisk (\*) is the repetition operator**.**​

**2.5.4 Python Tuples:**

* A tuple is another sequence data type that is similar to the list.
* A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.

* The main differences between lists and tuples are: Lists are enclosed in brackets ( [ ] ) and their elements and size can be changed, while tuples are enclosed in parentheses ( ( ) ) and cannot be updated.
* Tuples can be thought of as read-only lists.
* For example − Tuples are fixed size in nature whereas lists are dynamic. In other words, a tuple is immutable whereas a list is mutable. You can't add elements to a tuple. Tuples have no append or extend method. You can't remove elements from a tuple. Tuples have no remove or pop method.

**2.5.5**​  **Python Dictionary:**

* Python's dictionaries are kind of hash table type. They work like associative arrays 12 or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object.
* Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).
* You can use numbers to "index" into a list, meaning you can use numbers to find out what's in lists. You should know this about lists by now, but make sure you understand that you can only use numbers to get items out of a list.
* What a dict does is let you use anything, not just numbers. Yes, a dict associates one thing to another, no matter what it is.

**2.6**​  **PYTHON FUNCTION:**

**2.6.1**​  **Defining a Function:**

You can define functions to provide the required functionality. Here are simple rules to define a function in Python. Function blocks begin with the keyword def followed by the function name and parentheses (i.e.()).

Any input parameters or arguments should be placed within these parentheses.

You can also define parameters inside these parentheses.

The code block within every function starts with a colon (:) and is indented. The statement returns [expression] exits a function, optionally passing back an expression to the caller. A return statement with no arguments is the same as return None.

**2.6.2 Calling a Function:**

Defining a function only gives it a name, specifies the parameters that are to be included in the function and structures the blocks of code. Once the basic structure of a function is finalized, you can execute it by calling it from another function or directly from the Python prompt.

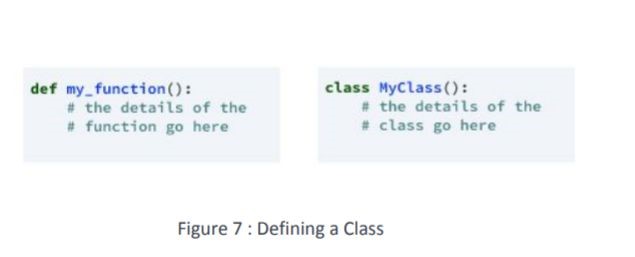
**2.7**​  **PYTHON USING OOP’s CONCEPTS:**

**2.7.1 Class:**

* Class: A user-defined prototype for an object that defines a set of attributes that characterize any object of the class. The attributes are data members (class variables and instance variables) and methods, accessed via dot notation.
* Class variable: A variable that is shared by all instances of a class. Class variables are defined within a class but outside any of the class's methods. Class variables are not used as frequently as instance variables are.
* Data member: A class variable or instance variable that holds data associated with a class and its objects.
* Instance variable: A variable that is defined inside a method and belongs only to the current instance of a class.

## ● Defining a Class:​

* We define a class in a very similar way how we define a function.
* Just like a function ,we use parentheses and a colon after the class name(i.e. ():) when we define a class. Similarly, the body of our class is 14 indented like a functionns body is.



**2.7.2 \_\_init\_\_ method in Class:**

* + The init method — also called a constructor — is a special method that runs when an instance is created so we can perform any tasks to set up the instance.
  + The init method has a special name that starts and ends with two underscores:\_\_init\_\_().

**CHAPTER 3**

# CASE STUDY

**3.1 PROBLEM STATEMENT:**

HAND GESTURE RECOGNITION

To predict the accuracy of the activity recognized by different algorithms and choose the best one.

* 1. **DATA SET:**

We have my data that will be available in a different link so if we look in our data we can see we have 10 different folders starting from 00 to 09. So in each folder we have another 10 different folders each named by the gesture of hand as we open these folders we can see the images of specified gestures in different angles. In each folder of labelled gesture contains 200 images. Therefore in my dataset I have overall 20,000 images.

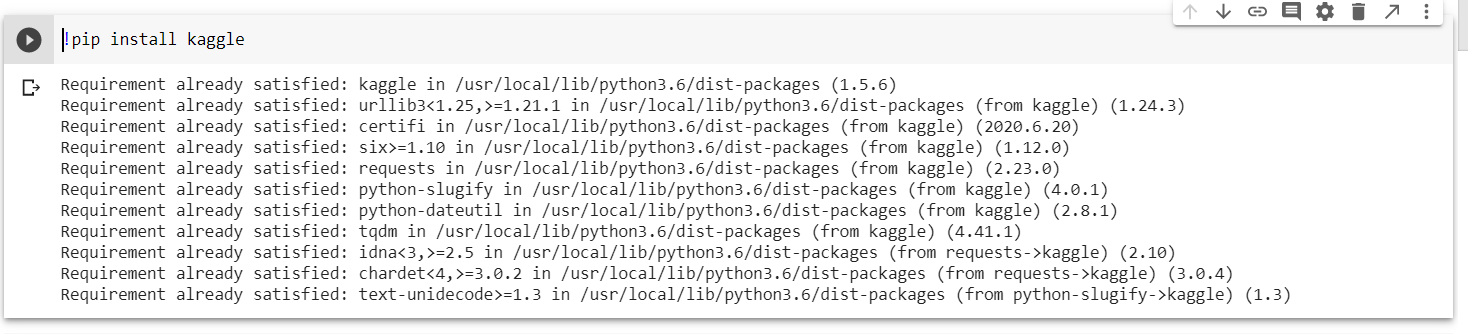
**3.3 OBJECTIVE OF THE CASE STUDY:**

Hand Gesture Recognition is a full fledged gesture recognition system, which will substitute mouse and keyboards completely. The goal of hand gesture recognition is to classify the given hand gesture data represented by some features into some predefined finite number of gesture classes.

# CHAPTER 4​

## 4.1 PREPROCESSING OF THE DATA:

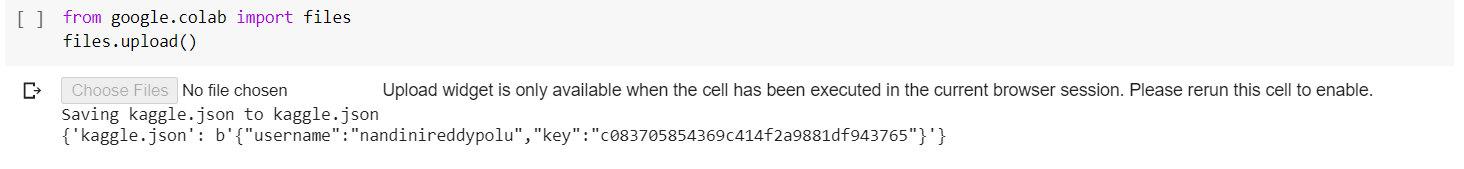
To procure the data we have many ways , but as my data is available in KAGGLE.com I have installed kaggle into my colab notebook.



**4.1.1 Upload File:**

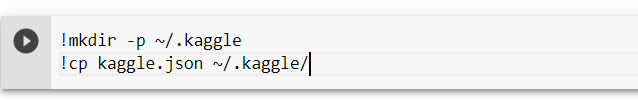
After creating an account in kaggle we have to create an API token in kaggle .while creating a new API , kaggle.json file will be downloaded in to the system .

Now we have to open that file in colab notebook .



## A choose option will appear. Choose the kaggle.json file for system download.

Now change the directory



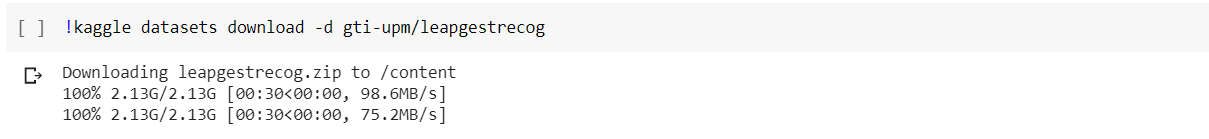
We need save this file into a directory named kaggle. Run the following commands to accomplish this task



Now we will import data set. So go to kaggle.com. Sign in and choose any dataset you like. Now accept terms and conditions if there any and copy download API command.

Now go back to google colab and paste the command and add "!" exclamation mark as a prefix and run it. Wait for a moment and the data set will be downloaded in a zip file.

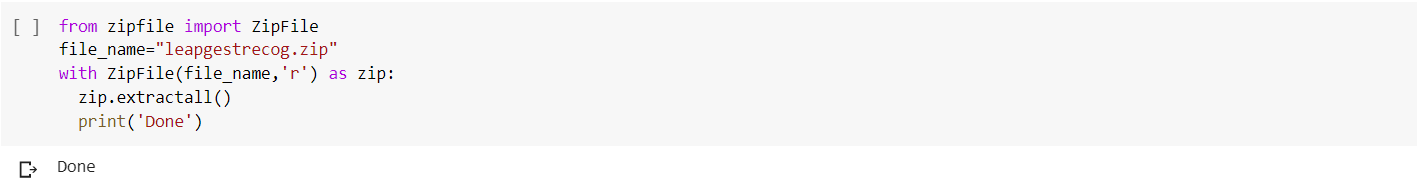
**4.1.2 IMPORTING THE DATA-SET FROM KAGGLE:**



Since the dataset is in zip file, we need to extract it. To do that first we need zip file pacakages.



Now extract the files from the zip file by using the following code



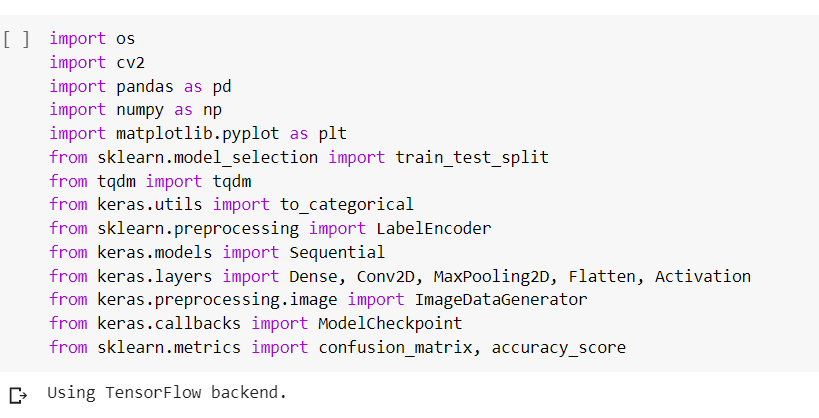
* importing required modules.
* specifying the zip file name.
* opening the zip file in READ mode.
* extracting all the files.

**4.1.3 IMPORTING THE LIBRARIES:**

We have to import the libraries as per the requirement of the algorithm.

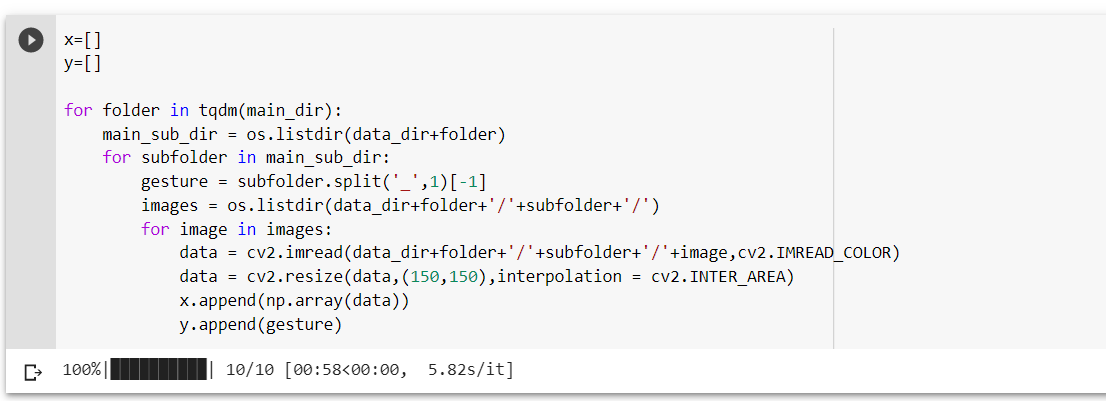
* **OS** module in python provides functions for interacting with the operating system.
* **OpenCV**-Python is a library of Python bindings designed to solve computer vision problems. .It makes use of Numpy, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays
* **numpy** package can be used to perform mathematical operations like 'mean'.​
* **pandas** package can be used to process dataframes.​
* **sklearn** is the main package which is used for machine learning.​
* **Matplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
* **train\_test\_split:**  module is used to split the data into training and testing sets.​
* TQDM: works on any platform (Linux, Windows, Mac, FreeBSD, NetBSD, Solaris/SunOS), in any console or in a GUI, and is also friendly with IPython/Jupyter notebooks.
* **Keras** is an [open-source](https://en.wikipedia.org/wiki/Open-source_software) [neural-network](https://en.wikipedia.org/wiki/Artificial_neural_network) library written in [Python](https://en.wikipedia.org/wiki/Python_(programming_language)). It is capable of running on top of [TensorFlow](https://en.wikipedia.org/wiki/TensorFlow) . Designed to enable fast experimentation with [deep neural networks](https://en.wikipedia.org/wiki/Deep_learning), it focuses on being user-friendly, modular, and extensible.
* **to\_categorical:** Converts a class vector (integers) to binary class matrix.
* **LabelEndoder:** Encode target labels with value between 0 and n\_classes-1.This transformer should be used to encode target values, i.e. y, and not the input X.
* **Sequential** **model**: It is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.
* Dense implements the operation: output = activation(dot(input, kernel) + bias) where activation is the element-wise activation function passed as the activation argument, kernel is a weights matrix created by the layer, and bias is a bias vector created by the layer (only applicable if use\_bias is True).
* Conv2D:This layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs. If use\_bias is True, a bias vector is created and added to the outputs. Finally, if activation is not None, it is applied to the outputs as well.
* A **flatten** operation on a tensor reshapes the tensor to have the shape that is equal to the number of elements contained in tensor non including the batch dimension
* **Activation** can either be used through an Activation layer, or through the activation argument supported by all forward layer.
* **ImageDataGenerator:** Generate batches of tensor image data with real-time data augmentation.
* **ModelCheckpoint:** callback is used in conjunction with training using model.fit() to save a model or weights (in a checkpoint file) at some interval, so the model or weights can be loaded later to continue the training from the state saved.
* **confusion\_matrix**: Compute confusion matrix to evaluate the accuracy of a classification.
* **Accuracy\_score:** The set of labels predicted for a sample must exactly match the corresponding set of labels

All the classes has been trained using deep learning.Libraries

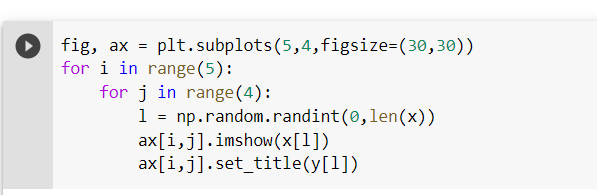




**4.1.4 Creating Features & Target Variables:**

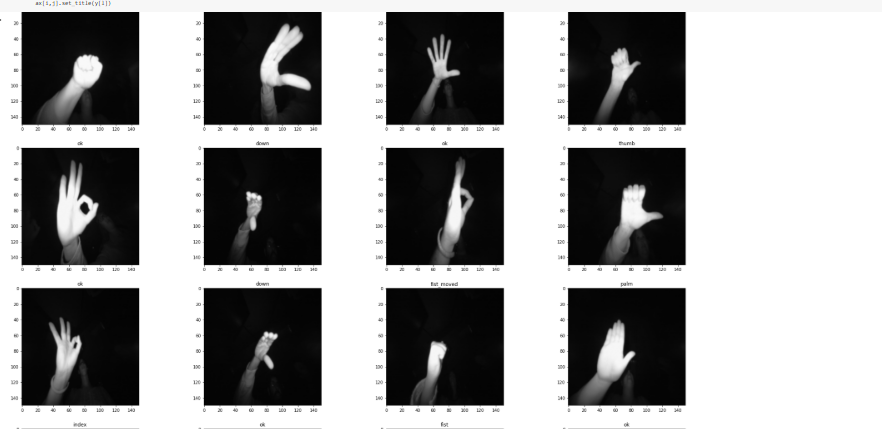


**4.2 Displaying images with labels:**



Displaying images by using subplots:

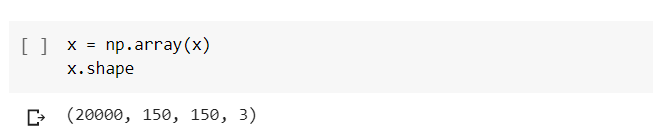
Here we can observe some of the images related to hand gesture

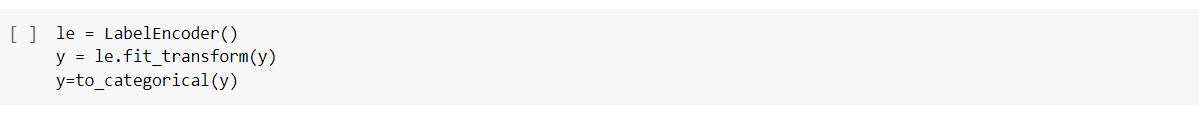


Here we can observe some of the gestures like ok, fist, down, palm .

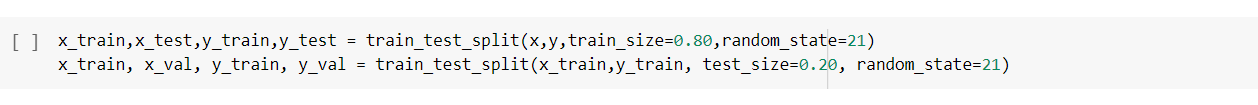
# 4.2.1 Convert Features List into a numpy array:

By using numpy.array():





# Creating Train, validation and test sets:



# 4.3 Defining Model Architecture:

It consists of three parts –

* Convolution
* Polling
* Flattening

The primary purpose of Convolution is to extract features from the input image. Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data.

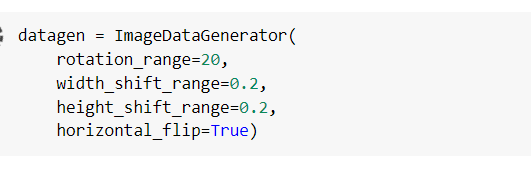


* **filters**: Integer, the dimensionality of the output space (i.e. the number of output filters in the convolution).
* **kernel\_size**: An integer or tuple/list of 2 integers, specifying the height and width of the 2D convolution window. Can be a single integer to specify the same value for all spatial dimensions.
* **Pooling** (also called subsampling or downsampling) reduces the dimensionality of each feature map but retains the most important information. In case of **Max Pooling**, we define a spatial neighbourhood (for example, a 2×2 window) and take the largest element from the rectified feature map within that window. Instead of taking the largest element we could also take the average (Average Pooling) or sum of all elements in that window. In practice, Max Pooling has been shown to work better.
* After pooling comes flattening. Here the matrix is converted into a linear array so that to input it into the nodes of our neural network.

# 4.3.1 Instantiating Callbacks:

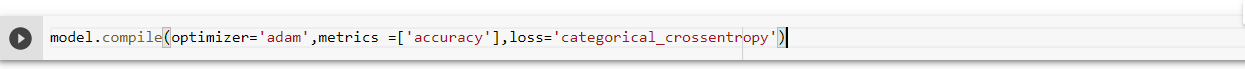


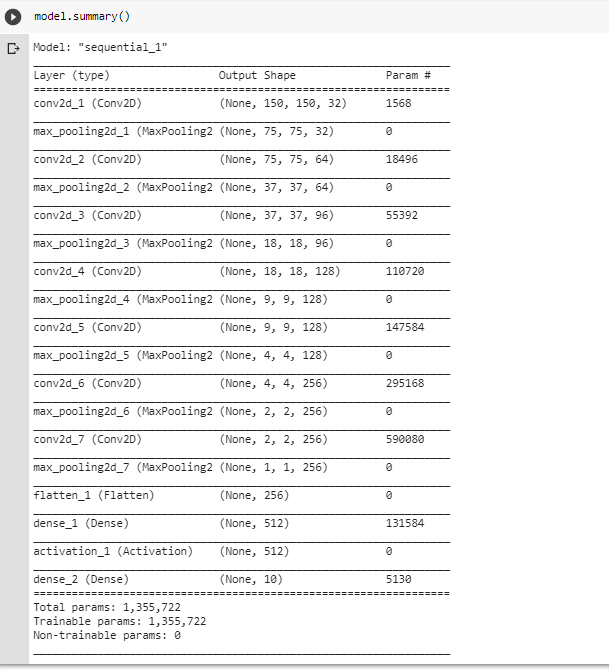
# 4.3.2 Data Augmentation:



Data augmentation is a way we can reduce over fitting on models, where we increase the amount of trainingdata using information only in our training data. The field of data augmentation is not new, and in fact, various data augmentation techniques have been applied to specific problems.

# 4.3.3 Compiling the Model:

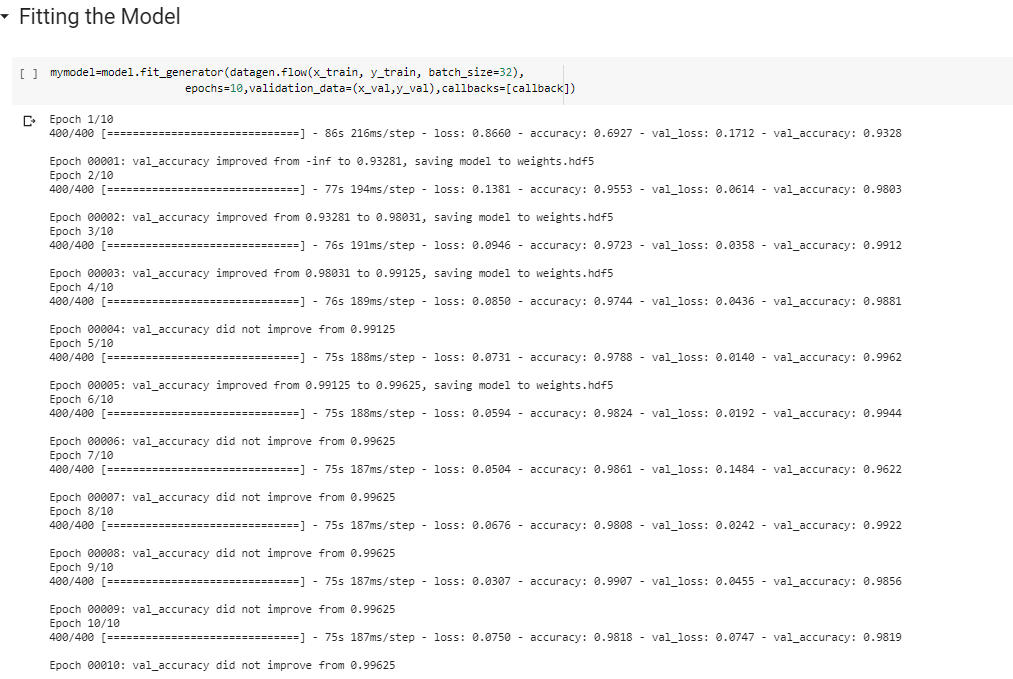




4.3.4 Training the model:

## Now the model is defined and the data is ready. To start the training of our model we use the model.fit() function which takes the training set, validation set, batch size and no of epochs.

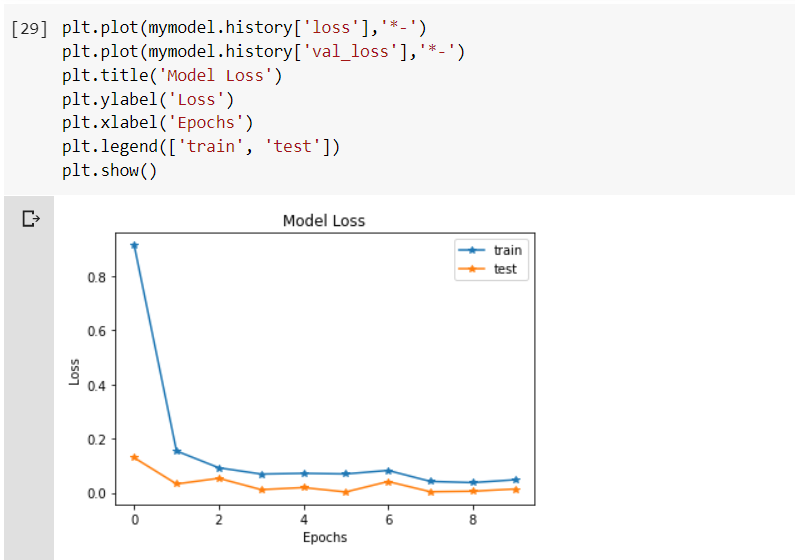
After training the model for 10 epochs we will save the model.



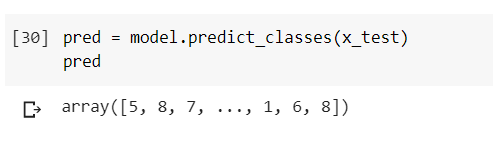
# 4.3.5 Model Performance:

With the help of matplotlib functions, we will plot the graph of training and validation accuracy.

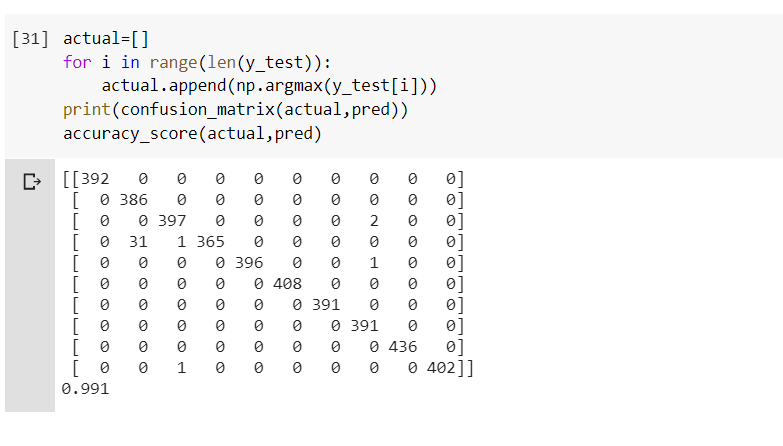




# 4.3.6 Predictions on Test Set:

****

# 4.3.7 Confusion Matrix & Accuracy Score:



By using confusion matrix and accuracy score I found that the accuracy of my dataset is 98%.

4.4 CONCLUSION:

The experimentation is tested on various dataset which justifies that the proposed solution outperforms the existing methods by being robust to scale variance and does not require any predefined templates for recognition.