A

Minor Project Report on

Image Recognition System

Submitted in the partial fulfilment of the requirement for the

award of Degree of

**Master of Computer Application**

**(Session: 2019-2021)**

Under Supervision of**:** Submitted By:

**Dr. Ramesh Kumar Kait NAVEEN ARYA**

**Assistant professor MCA-6th Sem**

Roll No: 8100457



**DEPARTEMENT OF COMPUTER SCIENCE & APPLICATION**

**KURUKSHETRA UNIVERSITY KURUKSHETRA**

**Department of Computer Science & Applications**

**Kurukshetra University, Kurukshetra**

**DECLARATION**

I, Naveen Arya, a student of Master of Computer Applications (MCA),VI semester in the Department of Computer Science & Applications, Kurukshetra University, Kurukshetra, under Roll No.8100457, session 2019-21, hereby declare that the project entitled “**Image Recognition System***”* has been completed by me on “**Python Machine Learning” .**

The matter embodied in this project work has not been submitted earlier for award of any degree or diploma to the best of my knowledge and belief.

**Date: 30th May,2021 (NAVEEN ARYA)**

**Department of Computer Science & Applications**

**Kurukshetra University, Kurukshetra**

**Dr. Ramesh Kumar Kait**

**(Assistant Professor)**

**CERTIFICATE**

It is certified that Mr. Naveen Arya, a student of Master of Computer Applications (MCA), under **Roll No.8100457** for the session 2019-2021, has completed the project entitled “**Image Recognition System”** working on **Python Machine Learning** under my supervision. He has attended the Department of Computer Science & Applications, Kurukshetra University, Kurukshetra and **Python Machine Learning** for required number of days with classes of VI semester.

I wish him all success in his all endeavours.

**(Dr. Ramesh Kumar Kait)**

**Department of Computer Science & Applications**

**Kurukshetra University, Kurukshetra**

**Dr. Rajender Nath**

**Professor & Chairman**

**CERTIFICATE**

It is certified that Mr. Naveen Arya is a bona fide student of Master of Computer Applications (MCA), under **Roll No.8100457**. He has undertaken the project entitled **Image Recognition System** on **Python Machine Learning** under the supervision of Dr. Ramesh Kumar Kait.

I wish him all success in his all endeavours.

**(Dr. Rajender Nath)**

**Acknowledgement**

Acknowledgement is the perfect way to convey heartiest thanks to all outstanding personalities.

I consider myself fortunate to get the opportunity of doing a project, yet the opportunity could

not have been utilized without the guidance and support of many individuals.

First of all, I wish to express my gratitude to my guide Dr. Ramesh Kumar Kait Upadhyaya, Professor,

Department of computer science & application, K.U. Kurukshetra for his cooperation and tell

me the right direction in preparing this project. I was fortunate to have her as my project guide.

She was ever willing to give me all kind of support andencouragement.

I also register my gratitude to project guide Dr.Pardeep Kumar (Department of Computer Science and Application) for giving me option to create Project in Python Machine Learning.

Finally, I would be failing to my duty if I did not put on record my deep sense of obligation to my

guardian and friends, who provided me an environment conducive to hard work.

**(Naveen Arya)**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **NAME OF TOPIC** | **PAGE NO.** |
| **1.** | **INTRODUCTION TO PROJECT** | 7-8 |
| **2.** | **PYTHON OVERVIEW** | 9-12 |
| **3.** | **MACHINE LEARNING OVERVIEW** | 13-17 |
| **4.** | **HARDWARE AND SOFTWARE REQUIRMENT** | 18 |
| **5.** | **SYSTEM DESIGN** | 19-22 |
| **6.** | **WORKING OF PROJECT** | 23-45 |
| **7.** | **CODING** | 46-53 |
| **8.** | **FUTURE SCOPE** | 54-55 |
| **9.** | **BIBLIOGRAPHY** | 56 |

**CHAPTER 1: INTRODUCTION TO PROJECT**

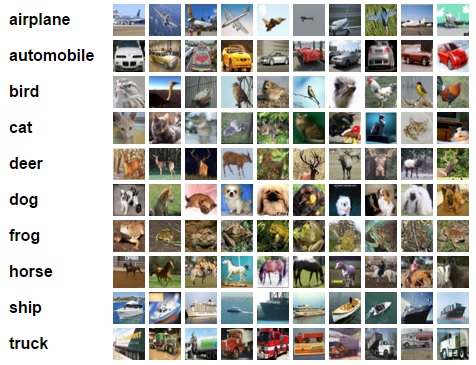
* 1. **Objectives of the Project**

“IMAGE RECOGNITION SYSTEM” is a system that provide image recognition ability to the Machine. The Image recognition is the ability of a system or software to identify objects, people, places and actions in images. It uses machine vision technologies with artificial intelligence and trained algorithms to recognize images through a camera system.

* 1. **About the Project**

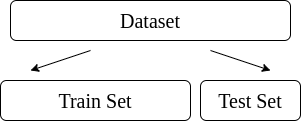
The system is basically created around image dataset with deferent classification. It is developed by using Python to create Image Recognition Model. Images will go through to make Dataset Model.

1.Image flatten data:

The project took image from a folder and store in a variable of python with respect to that class. When all folder with deferent class and images are store in a dataset variable. Each class image comes under deferent array. These are separated by another array which identify class of dataset.

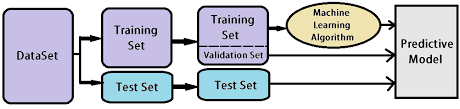
2. Split dataset:

Dataset then split into two-part train data set and test dataset. Hear train dataset participate in training model. Then test data set participate to find accuracy of data set.



3.Traning dataset:

Train dataset is converted into model after completion of training.



4. Testing Model:

Model predict output for test dataset. How many prediction are correct based on that accuracy is generated.

**1.3Platform**

This project is a website based on Python language as the back end and frontend coding.

Tools Used:

1) Python

2) Notepad

3) C0mmand Prompt

**CHAPTER 2: PYTHON OVERVIEW**

Python is a high-level, interpreted, interactive and object-oriented scripting language.

Python is designed to be highly readable. It uses English keywords frequently whereas the

other languages use punctuations. It has fewer syntactical constructions than other

languages.

**Python is Interpreted:** Python is processed at runtime by the interpreter. You do

not need to compile your program before executing it. This is similar to PERL and

PHP.

**Python is Interactive:** You can actually sit at a Python prompt and interact with

the interpreter directly to write your programs.

**Python is Object-Oriented:** Python supports Object-Oriented style or technique

of programming that encapsulates code within objects.

**Python is a Beginner's Language:** Python is a great language for the beginner level

programmers and supports the development of a wide range of applications

from simple text processing to WWW browsers to games.

**2.1 History of Python**

Python was developed by Guido van Rossum in the late eighties and early nineties at the

National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++,

Algol-68, Small Talk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the

GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although

Guido van Rossum still holds a vital role in directing its progress.

Python 1.0 was released in November 1994. In 2000, Python 2.0 was released.

Python 2.7.11 is the latest edition of Python 2.

Meanwhile, Python 3.0 was released in 2008. Python 3 is not backward compatible

with Python 2. The emphasis in Python 3 had been on the removal of duplicate

programming constructs and modules so that "There should be one -- and

preferably only one -- obvious way to do it." Python 3.5.1 is the latest version of

Python 3.

**2.2 Python Features**

Python's features include-

**Easy-to-learn:** Python has few keywords, simple structure, and a clearly defined

syntax. This allows a 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-maintained.

**A broad standard library:** Python's bulk of the library is very portable and cross platform

compatible on UNIX, Windows, and Macintosh.

**Interactive Mode:** Python has support for an interactive mode, which allows

interactive testing and debugging of snippets of code.

**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.

**Scalable:** Python provides a better structure and support for large programs than

shell scripting.

Apart from the above-mentioned features, Python has a big list of good features. A few

are listed below-

It supports functional and structured programming methods as well as OOP.

It can be used as a scripting language or can be compiled to byte-code for building

large applications.

It provides very high-level dynamic data types and supports dynamic type

checking.

It supports automatic garbage collection.

It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

**2.3 Use of Python:**

Python is a general-purpose, popular programming language and it is used in almost every technical field. The various areas of Python use are given below.

* Data Science
* Date Mining
* Desktop Applications
* Console-based Applications
* Mobile Applications
* Software Development
* Artificial Intelligence
* Web Applications
* Enterprise Applications
* 3D CAD Applications
* Machine Learning
* Computer Vision or Image Processing Applications.
* Speech Recognitions

**2.4 Basic Syntax of Python:**

There is no use of curly braces or semicolon in Python programming language. It is English-like language. But Python uses the indentation to define a block of code. Indentation is nothing but adding whitespace before the statement when it is needed. **For example**

# defining init method for class

def \_\_init\_\_(self, r, i):

self.real = r

self.img = i

# overloading the add operator using special function

def \_\_add\_\_(self, a):

r = self.real + a.real

i = self.img + a.img

return complex(r,i)

# overloading the add operator using special function

def \_\_sub\_\_(self, a):

r = self.real - a.real

i = self.img - a.img

return complex(r,i)

# overloading the add operator using special function

def \_\_mul\_\_(self, a):

r = self.real \* a.real - self.img \* a.img

i = self.real \* a.img + self.img \* a.real

return complex(r, i)

# overloading the add operator using special function

def \_\_truediv\_\_(self, a):

r = (self.real \* a.real - self.img \* (-a.img))/(a.real-a.img)

i = (self.real \* (-a.img) + self.img \* a.real)/(a.real-a.img)

return complex(r, i)

# string function to print object of Complex class

def \_\_str\_\_(self):

return '('+str(self.real)+'+'+str(self.img)+'i)'

# P2: Input

c1 = Complex(40,3)

c2 = Complex(100,5)

print('(', c1, ') + (', c2,')=', c1+c2, sep = ' ',end = '.\n')

print('(', c1, ') - (', c2,')=', c1-c2, sep = ' ',end = '.\n')

print('(', c1, ') \* (', c2,')=', c1\*c2, sep = ' ',end = '.\n')

print('(', c1, ') / (', c2,')=', c1/c2, sep = ' ',end = '.\n')

OUTPUT:

( (40+3i) ) + ( (100+5i) )= (140+8j).

( (40+3i) ) - ( (100+5i) )= (-60-2j).

( (40+3i) ) \* ( (100+5i) )= (3985+500j).

( (40+3i) ) / ( (100+5i) )= (42.26315789473684+1.0526315789473684j).

**2.5 Python File I/O:**

Files are used to store data in a computer disk. In this tutorial, we explain the built-in file object of Python. We can open a file using Python script and perform various operations such as writing, reading, and appending. There are various ways of opening a file. We are explained with the relevant example. We will also learn to perform read/write operations on binary files.

**2.6 Python Modules:**

Python modules are the program files that contain a Python code or functions. There are two types of module in the Python - User-define modules and built-in modules. A module that the user defines, or we can say that our Python code saved with **.py** extension, is treated as a user-define module.

Built-in modules are predefined modules of Python. To use the functionality of the modules, we need to import them into our current working program.

**2.7 Python Exceptions:**

An exception can be defined as an unusual condition in a program resulting in the interruption in the flow of the program.

Whenever an exception occurs, the program stops the execution, and thus the further code is not executed. Therefore, an exception is the run-time errors that are unable to handle to Python script. An exception is a Python object that represents an error.

**2.8 Python CSV:**

A **csv** stands for "comma separated values", which is defined as a simple file format that uses specific structuring to arrange tabular data. It stores tabular data such as spreadsheet or database in plain text and has a common format for data interchange. A **csv** file opens into the excel sheet, and the rows and columns data define the standard format. Visit the following tutorial to learn the CSV module in detail.

**2.9 Python Sending Mail:**

We can send or read a mail using the Python script. Python's standard library modules are useful for handling various protocols such as PoP3 and IMAP. We will learn how to send a mail with the popular email service SMTP from a Python script.

**CHAPTER 3: MACHINE LEARNING OVERVIEW**

The journey of AI began in the 1950's when the computing power was a fraction of what it is today. AI started out with the predictions made by the machine in a fashion a statistician does predictions using his calculator. Thus, the initial entire AI development was based mainly on statistical techniques. In this chapter, let us discuss in detail what these statistical techniques are.

**3.1 Statistical Techniques:**

The development of today’s AI applications started with using the age-old traditional statistical techniques. You must have used straight-line interpolation in schools to predict a future value. There are several other such statistical techniques which are successfully applied in developing so-called AI programs. We say “so-called” because the AI programs that we have today are much more complex and use techniques far beyond the statistical techniques used by the early AI programs.

Some of the examples of statistical techniques that are used for developing AI applications in those days and are still in practice are listed here:

• Regression

• Classification

• Clustering

• Probability Theories

• Decision Trees

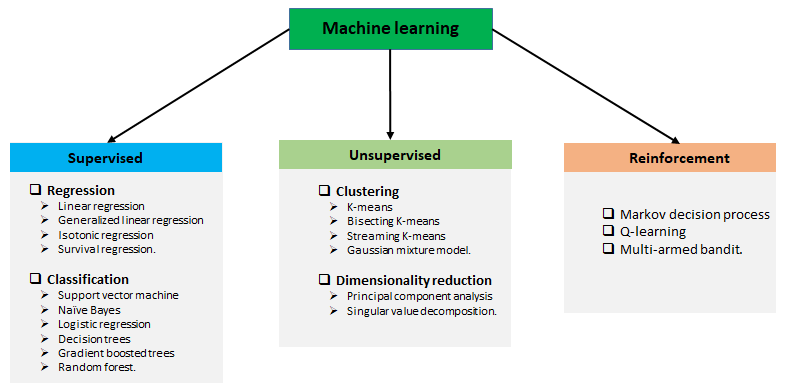
Here we have listed only some primary techniques that are enough to get you started on AI without scaring you of the vastness that AI demands. If you are developing AI applications based on limited data, you would be using these statistical techniques.

However, today the data is abundant. To analyse the kind of huge data that we possess statistical techniques are of not much help as they have some limitations of their own. More advanced methods such as deep learning are hence developed to solve many complex problems.

**3.2 Type of Machine Learning:**

There are 3 types of Machines Learning such as:

* Supervised Learning
* Unsupervised Learning
* Reinforcement



**Supervised learning:**

Supervised learning is analogous to training a child to walk. You will hold the child’s hand, show him how to take his foot forward, walk yourself for a demonstration and so on, until the child learns to walk on his own.

Regression

In the case of supervised learning, you give concrete known examples to the computer. You say that for given feature value x1 the output is y1, for x2 it is y2, for x3 it is y3, and so on. Based on this data, you let the computer figure out an empirical relationship between x and y.

Once the machine is trained in this way with a sufficient number of data points, now you would ask the machine to predict Y for a given X. Assuming that you know the real value of Y for this given X, you will be able to deduce whether the machine’s prediction is correct. Thus, you will test whether the machine has learned by using the known test data.

Once you are satisfied that the machine is able to do the predictions with a desired level of accuracy (say 80 to 90%) you can stop further training the machine. Now, you can safely use the machine to do the predictions on unknown data points, or ask the machine to predict Y for a given X for which you do not know the real value of Y. This training comes under the regression that we talked about earlier.

Classification:

You may also use machine learning techniques for classification problems. In classification problems, you classify objects of similar nature into a single group. For example, in a set of 100 students say, you may like to group them into three groups based on their heights - short, medium and long. Measuring the height of each student, you will place them in a proper group.

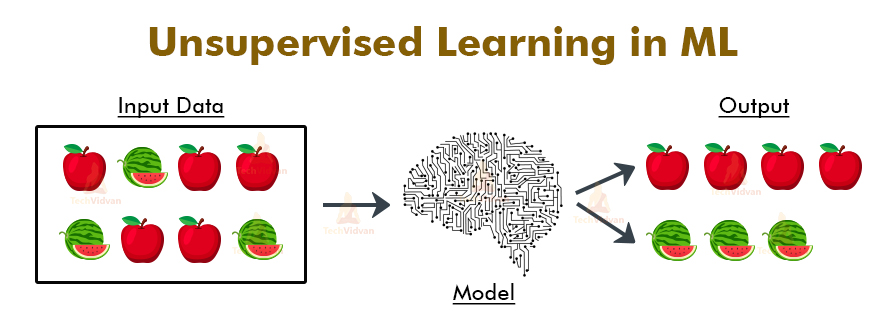
Now, when a new student comes in, you will put him in an appropriate group by measuring his height. By following the principles in regression training, you will train the machine to classify a student based on his feature – the height. When the machine learns how the groups are formed, it will be able to classify any unknown new student correctly. Once again, you would use the test data to verify that the machine has learned your technique of classification before putting the developed model in production.

Supervised Learning is where the AI really began its journey. This technique was applied successfully in several cases. You have used this model while doing the hand-written recognition on your machine. Several algorithms have been developed for supervised learning. You will learn about them in the following chapters.

**Unsupervised Learning**

In unsupervised learning, we do not specify a target variable to the machine, rather we ask machine “What can you tell me about X?”. More specifically, we may ask questions such as given a huge data set X, “What are the five best groups we can make out of X?” or “What features occur together most frequently in X?”. To arrive at the answers to such questions, you can understand that the number of data points that the machine would require to deduce a strategy would be very large. In case of supervised learning, the machine can be trained with even about few thousands of data points. However, in case of unsupervised learning, the number of data points that is reasonably accepted for learning starts in a few millions. These days, the data is generally abundantly available. The data ideally requires curating. However, the amount of data that is continuously flowing in a social area network, in most cases data curation is an impossible task.

The following figure shows the boundary between the yellow and red dots as determined by unsupervised machine learning. You can see it clearly that the machine would be able to determine the class of each of the black dots with a fairly good accuracy.

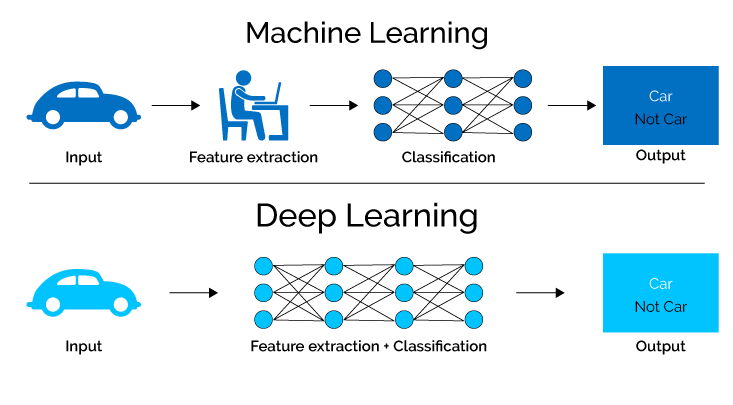


**Deep Learning**

The deep learning is a model based on Artificial Neural Networks (ANN), more specifically Convolutional Neural Networks (CNN)s. There are several architectures used in deep learning such as deep neural networks, deep belief networks, recurrent neural networks, and convolutional neural networks.

These networks have been successfully applied in solving the problems of computer vision, speech recognition, natural language processing, bioinformatics, drug design, medical image analysis, and games. There are several other fields in which deep learning is proactively applied. The deep learning requires huge processing power and humongous data, which is generally easily available these days.

We will talk about deep learning more in detail in the coming chapters.



**3.3Machine Learning – Artificial Neural Networks**

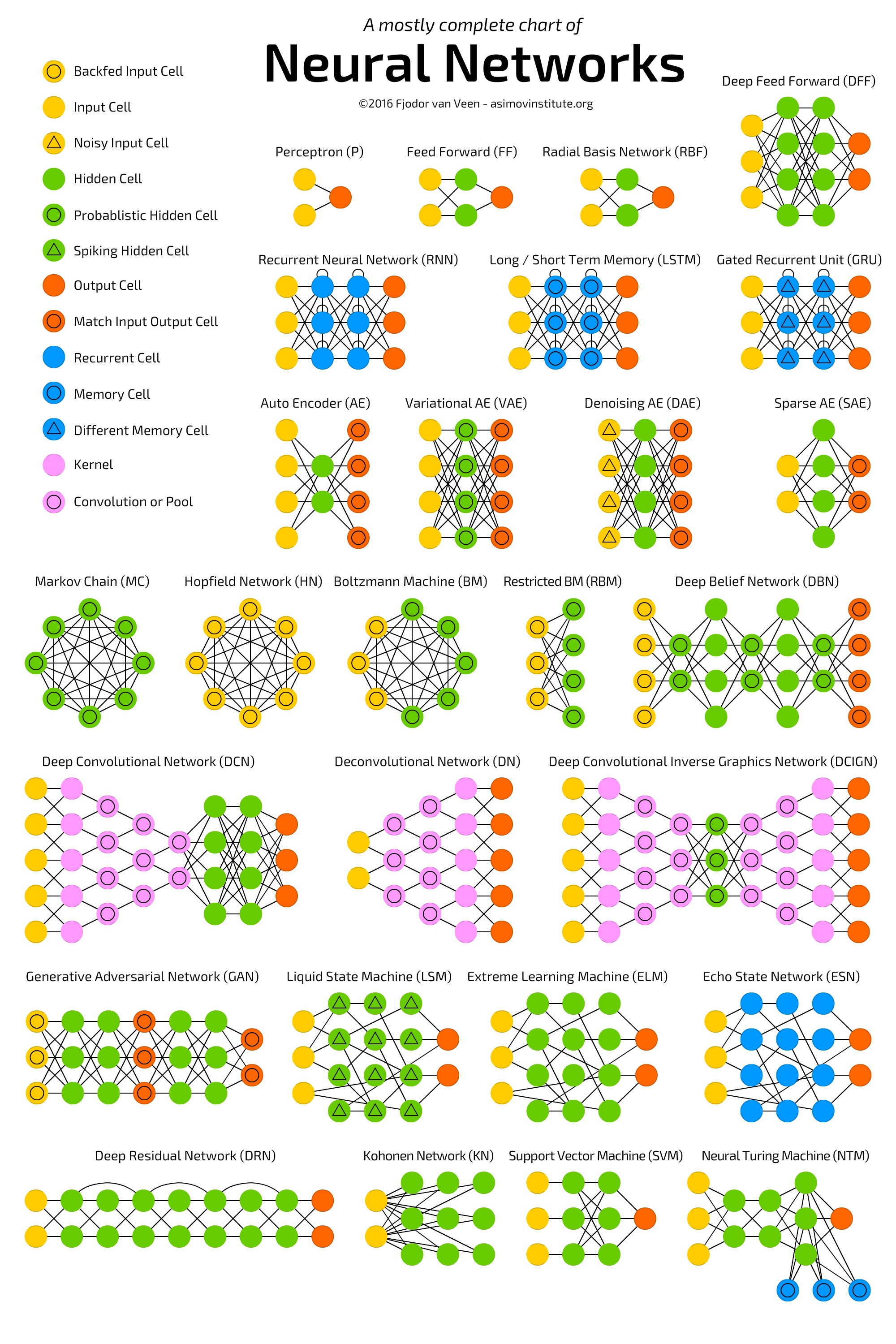
The idea of artificial neural networks was derived from the neural networks in the human brain. The human brain is really complex. Carefully studying the brain, the scientists and engineers came up with an architecture that could fit in our digital world of binary computers. One such typical architecture is shown in the diagram below:

There is an input layer which has many sensors to collect data from the outside world. On the right hand side, we have an output layer that gives us the result predicted by the network. In between these two, several layers are hidden. Each additional layer adds further complexity in training the network, but would provide better results in most of the situations. There are several types of architectures designed which we will discuss now.

**ANN**

Architectures The diagram below shows several ANN architectures developed over a period of time and are in practice today.

Each architecture is developed for a specific type of application. Thus, when you use a neural network for your machine learning application, you will have to use either one of the existing architecture or design your own. The type of application that you finally decide upon depends on your application needs. There is no single guideline that tells you to use a specific network architecture.



**CHAPTER 4: HARDWARE AND SOFTWARE REQUIRMENT**

**4.1 HARDWARE REQUIRMENT:**

Minimum hardware requirements are:

* Processor – Intel Xeon E2630 v4 – 10 core processor, 2.2 GHz with Turboboost upto 3.1 GHz. ...
* Motherboard – ASRock EPC612D8A.
* RAM – 128 GB DDR4 2133 MHz.
* 2 TB Hard Disk (7200 RPM) + 512 GB SSD.
* GPU – NVidia TitanX Pascal (12 GB VRAM)
* Intel Heatsink to keep temperature under control.

**4.2 SOFTWARE REQUIRMENT:**

* CUDA Deep Neural Network (cuDNN) 7.5 library
* NVIDIA CUDA 10.1
* NVIDIA GPU driver 418.40
* NVIDIA NCCL2 2.4
* Anaconda 2018.12
* Windows10/ Red Hat Enterprise Linux (RHEL) 7.6

**CHAPTER 5: SYSTEM DESIGN**

**5.1 What Is System Design:**

System design provides the understandings and procedural details necessary for

implementing the system recommended in the system study. Emphasis is on the

translating the performance requirements into design specifications. The design phase is a

transition from a user-oriented document (System proposal) to a document oriented to the

programmers or database personnel.

System design goes through two phases of development:

**5.2 Logical Design:**

The logical design of an information system is analogous to an engineering blue print of

an automobile. It shows the major features and how they are related to one another. The

detailed specification for the new system was drawn on the bases of user’s requirement

data. The outputs inputs and databases are designed in this phase. E-R diagrams

graphically representation of the relationship between the various entities. That is it used

to represent the relationship between two or more entity. A data flow diagram shows the

logical flow of the system. For a system it describes the input (source), output

(destination), database (data stores) and procedures (data flows) all in a format that meets

the user’s requirement. The logical design also specifies input forms and screen layouts.

Data flow diagrams of the system are given in section 4.1.1.

* 1. Flow Chart
  2. Data Flow Diagrams (DFDs)

(a) Flow Chart:

A flow chart is defined as a pictorial representation describing a process being studied or even used to plan stages of a project. Flow charts tend to provide people with a common language or reference point when dealing with a project or process.

Flow charts provide an excellent form of documentation for a process, and quite often are useful when examining how various steps in a process work together.

TABED VIEW

VIEW HOME PAGE

DELETE MODEL

USE MODEL

CREATE MODEL

FLOW CHART 1: IMAGE RECOGNITION

INPUT OUTPUT VIEW

INPUT IMAGE DATASET

INPUT MODEL NAME

SHAPE DATA SET FLATERN ARRAY

SPLLET DATA INTO TRAIN AND TEST

TRAIN TRAINY DATA SET AND CREATE MODEL

VIEW MODEL ACCURACY

SAVE MODEL AND CATAGRES

FLOW CHART 2: CREATE MODEL

INPUT OUTPUT VIEW

INPUT IMAGE

INPUT MODEL NAME

SHOW IMAGE

AND PRREDICTION

FLOW CHART 3: USE MODEL

INPUT OUTPUT VIEW

INPUT MODEL NAME

SHOW MODEL DELETE OR NOT?

FLOW CHART 4: DELETE MODEL

(b) Data Flow Diagram (DFD):-

The data flow diagram is the graphical representation that depicts information flow and

transforms that are applied as data move from input to output. DFD is a model, which

gives the insight into the information domain and functional domain at the same time.

DFD is refined into different levels. The more refined DFD is, more details of the system

are incorporated. In the process of creating a DFD, we decompose the system into

different functional subsystems. The DFD refinement results in a corresponding

refinement of data. After going through the current working process of the department,

we can create the Data Flow Diagram (DFD).

DFD Symbols:-

In the DFD, there are four symbols

1. A square defines a source(originator) or destination of system data.

2. An arrow identifies data flow. It is the pipeline through which the information flows. Data

move in a specific direction from an origin to a destination.

3. A circle or a bubble represents a process that transforms incoming data flow into outgoing

data flows.

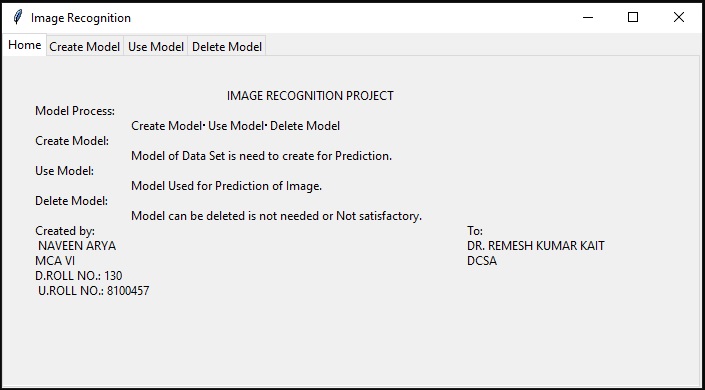
4. An open rectangle is a data store, data at rest or a temporary repository of data

We can create the Data Flow Diagram (DFD)

|  |  |
| --- | --- |
| Process that transforms data flow |  |
| Data Flow |  |
| Input data |  |
| Call sub-process |  |

**CHAPTER 6: WORKING OF PROJECT**

Home page when project Start:

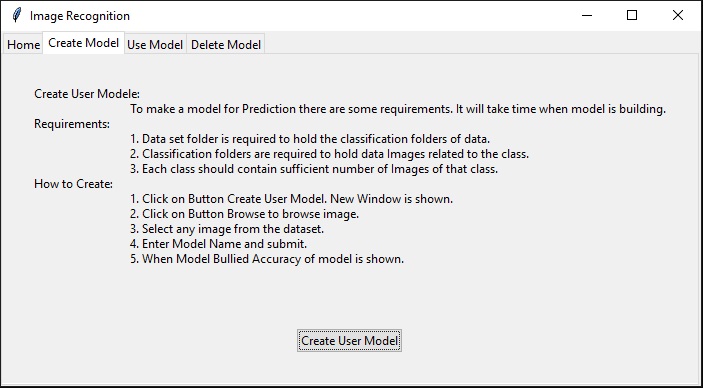


SCREENSHOT 1.

Create Model: Tab make view as in screenshot 2.

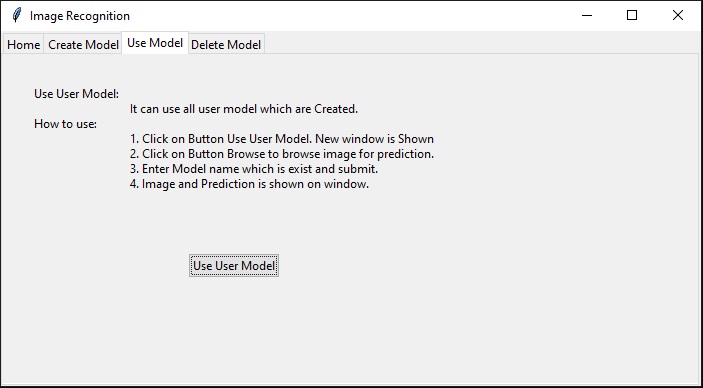
Use Model: Tab make view as in screenshot 3.

Delete Model: Tab make view as in screenshot 4.



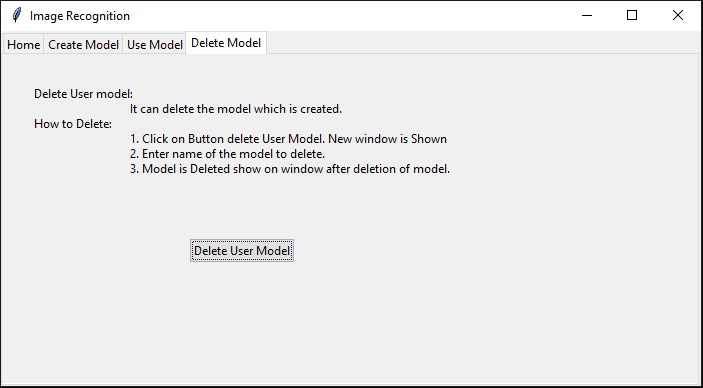
SCREENSHOT 2

Create User Model: open new window to create model screenshot 5.



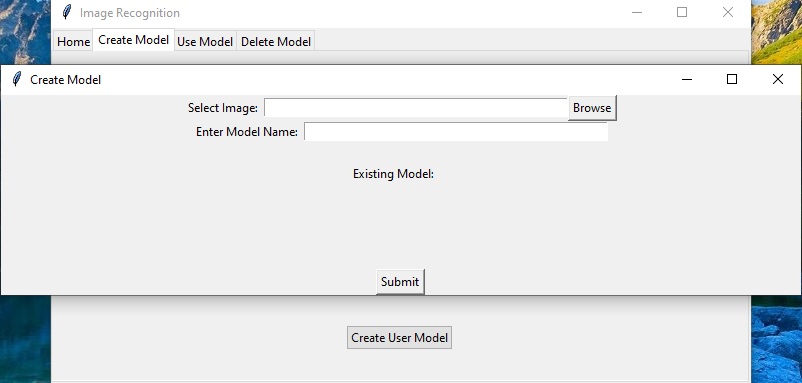
SCREENSHOT 3

Use User Model: open new window to create model screenshot 11.



SCREENSHOT 4

Delete User Model: open new window to create model screenshot 5.



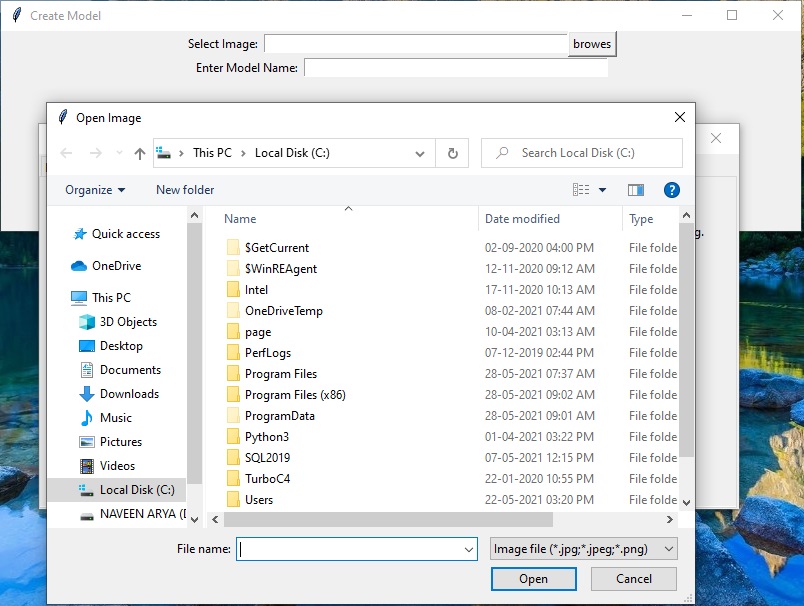
SCREENSHOT 5

Browse: open new window to input image path in dataset screenshot 6.

Model Name entered by user in input box.

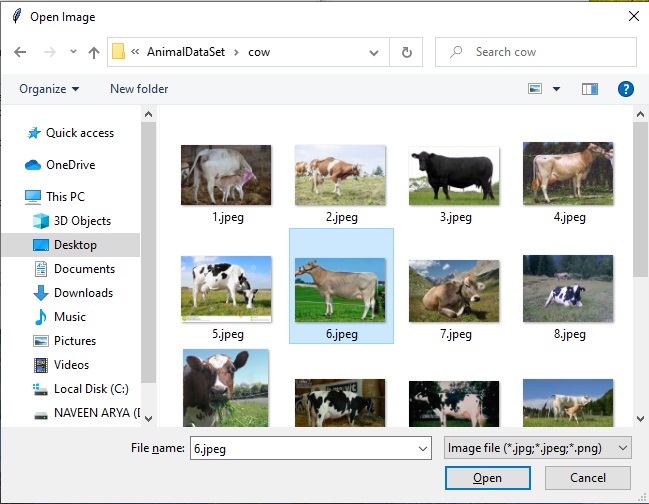
Then submit: To start process of building model.

After model building accuracy of model is shown.



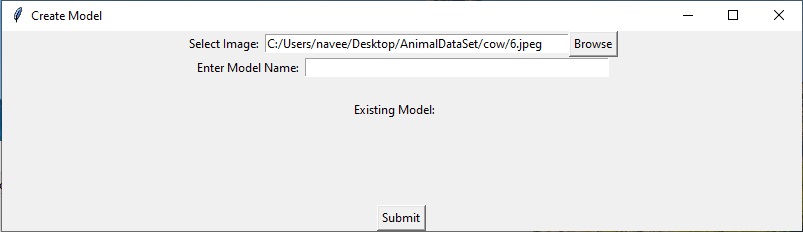
SCREENSHOT 6

Go to the location of dataset images screenshot 7.



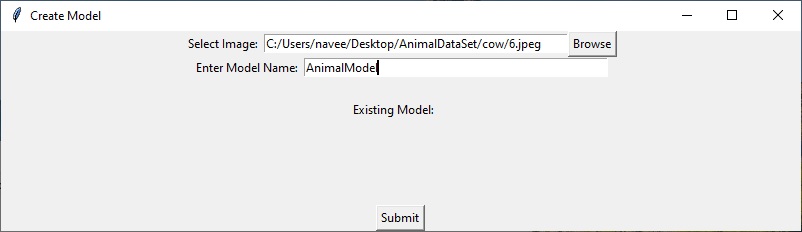
SCREENSHOT 7

Open: Select any image from dataset screenshot 8.



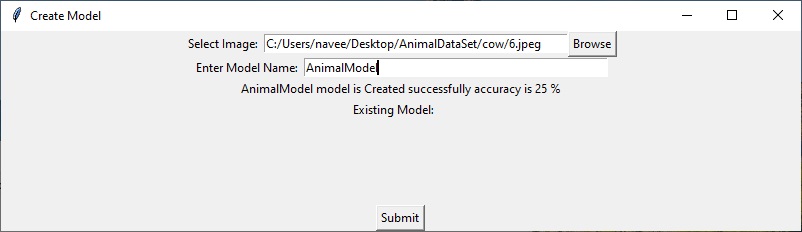
SCREENSHOT 8

Enter Model name and submit screenshot 9.



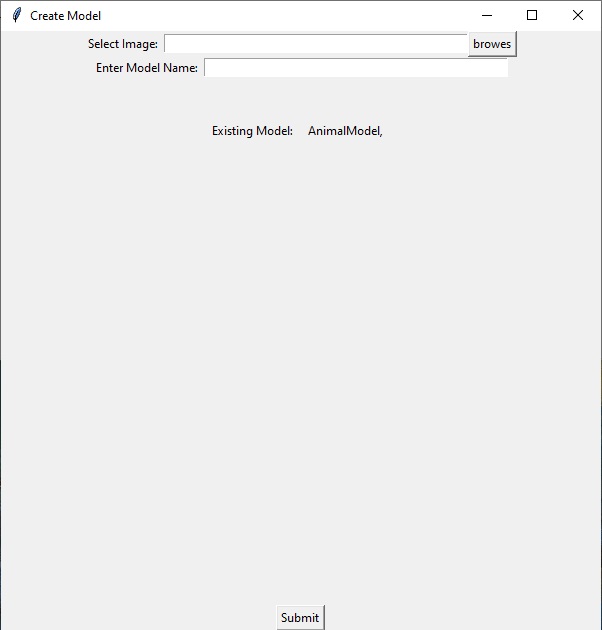
SCREENSHOT 9

It will take some time and go to screenshot 10.



SCREENSHOT 10

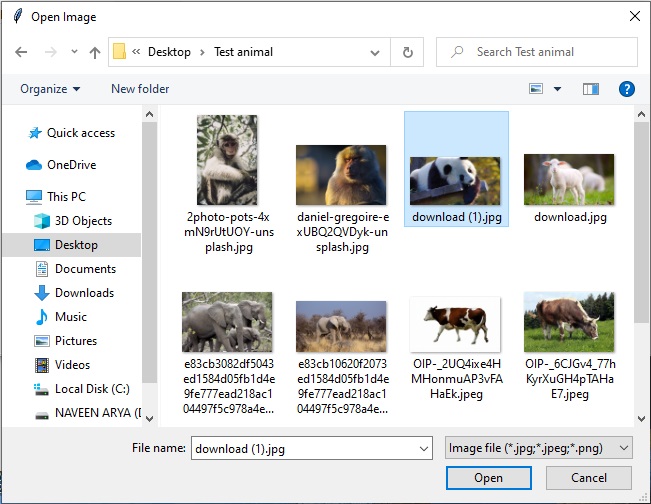
Model is created successfully with accuracy 25%.



SCREENSHOT 11

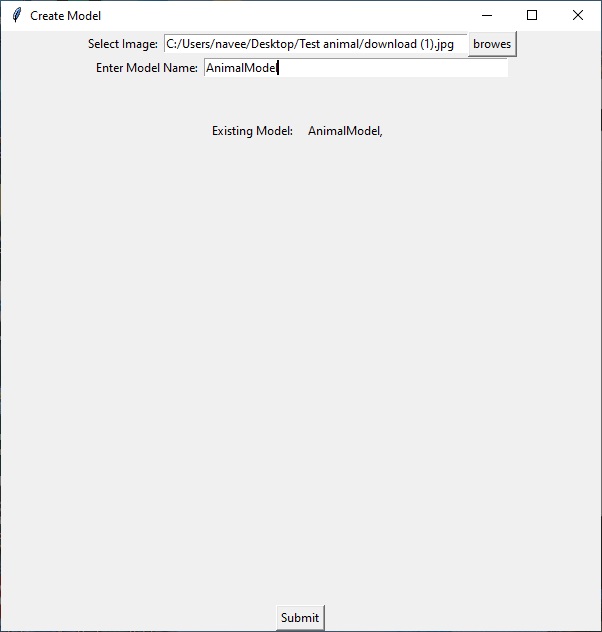
Brewes: It will give option to select the image for prediction as screenshot 12.

Enter Model Name: Model name as give Existing model as screenshot 13.



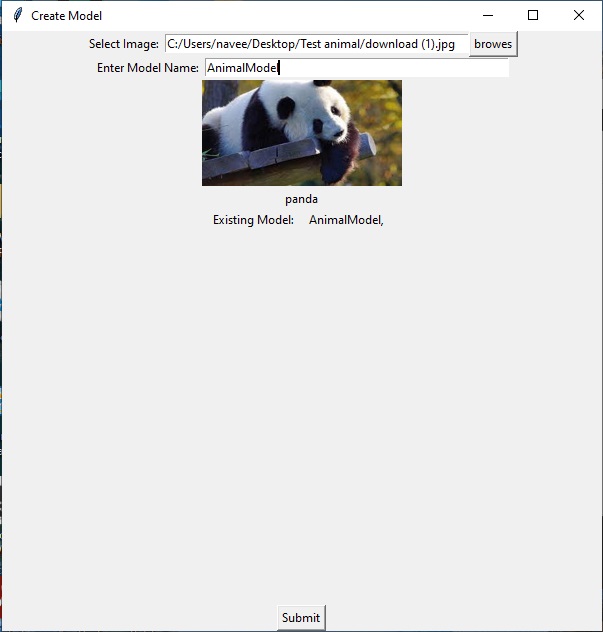
SCREENSHOT 12

Open: Image path is shown in select image as screenshot 13.



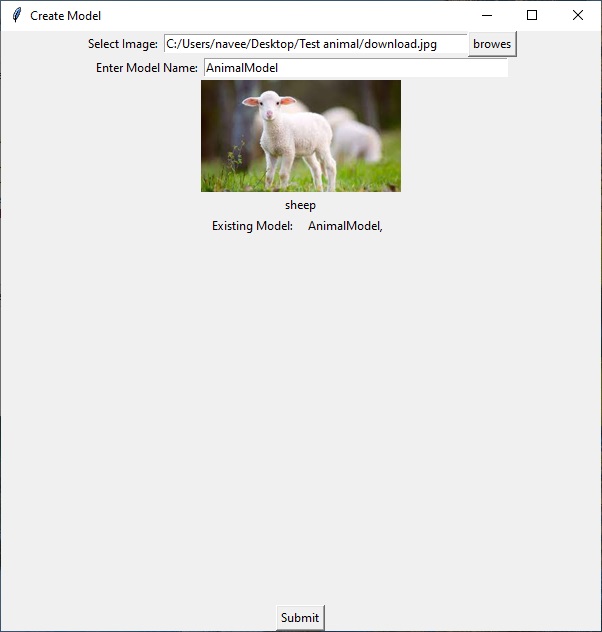
SCREENSHOT 13

Submit: After summation image and prediction is shone as in screenshot 14-18.



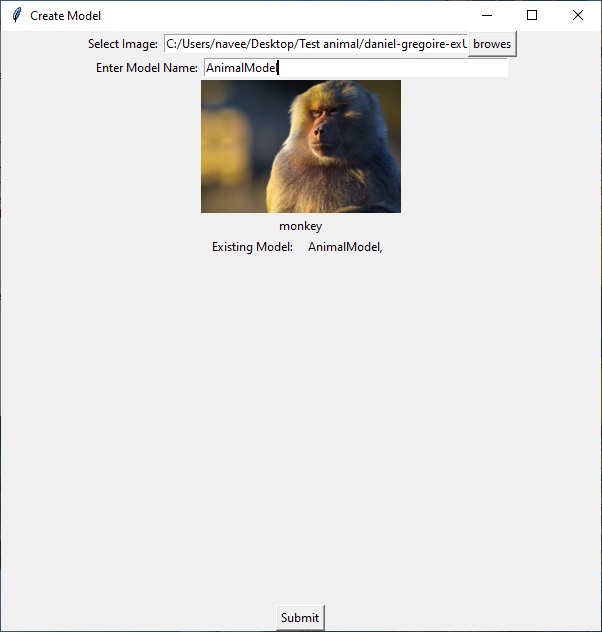
SCREENSHOT 14

Image and Prediction is shown.



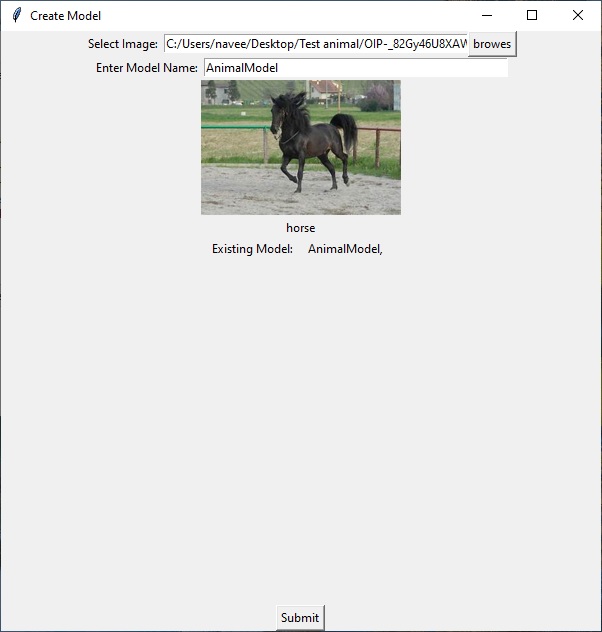
SCREENSHOT 15

Image and Prediction is shown.



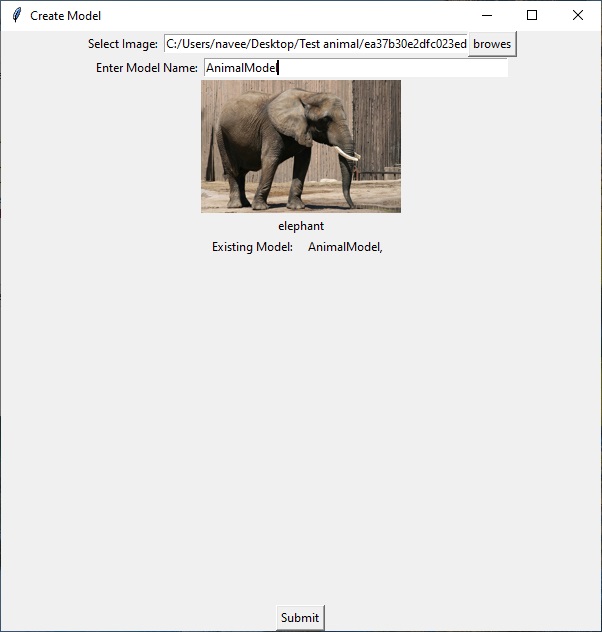
SCREENSHOT 16

Image and Prediction is shown.



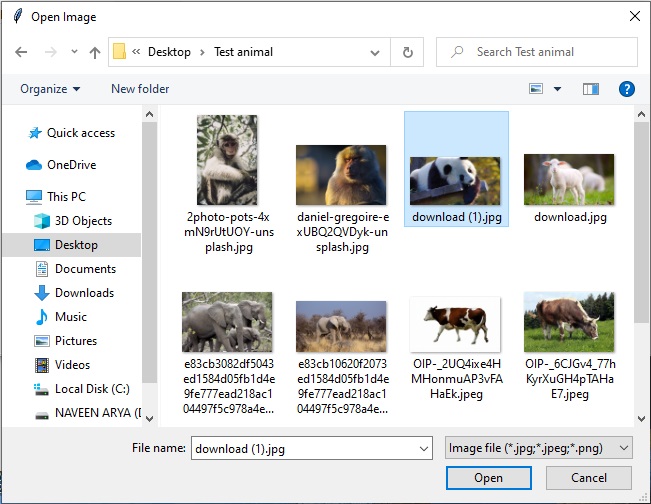
SCREENSHOT 17

Image and Prediction is shown.



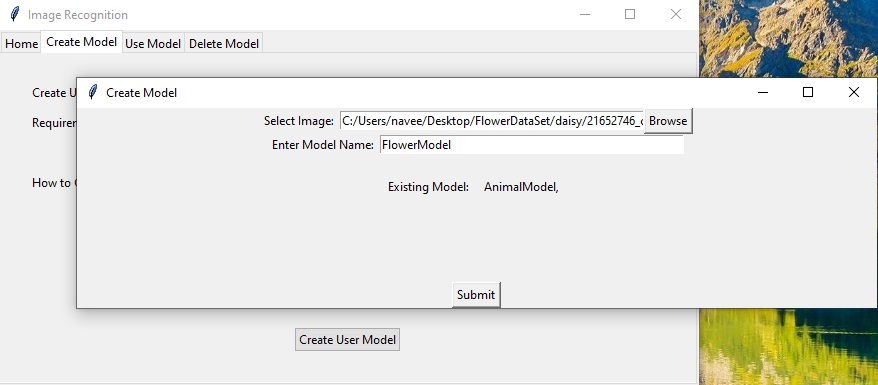
SCREENSHOT 18

Image and Prediction is shown.



SCREENSHOT 19

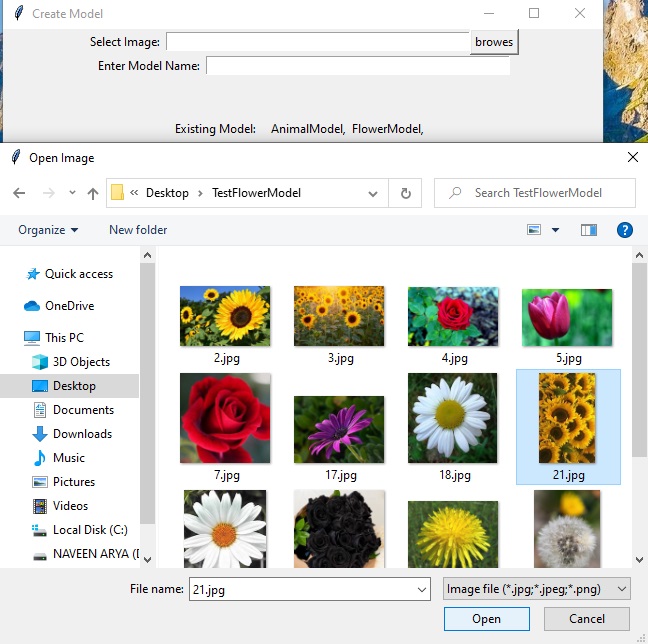
Loading dataset for Making FlowerModel.



SCREENSHOT 20

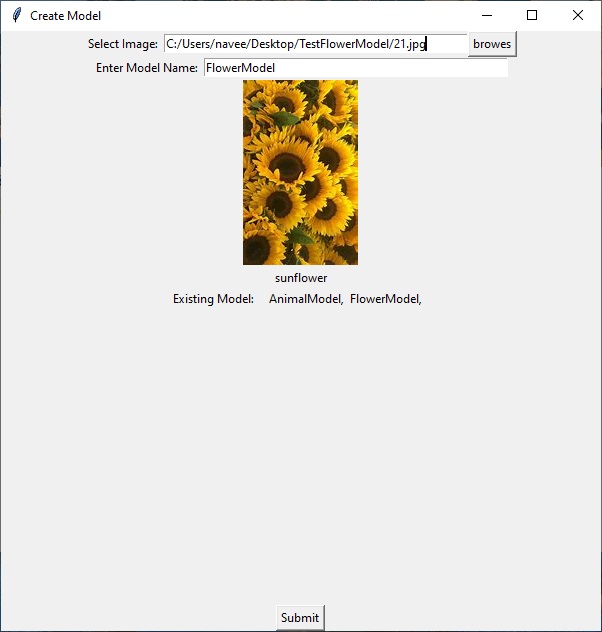
Model name FlowerModel while AnimalModel is existing.

Hear Model name should not be same as existing model.

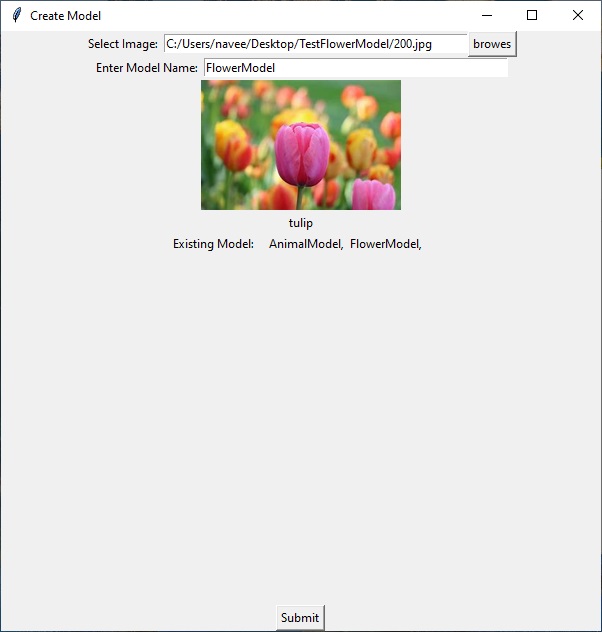


SCREENSHOT 21

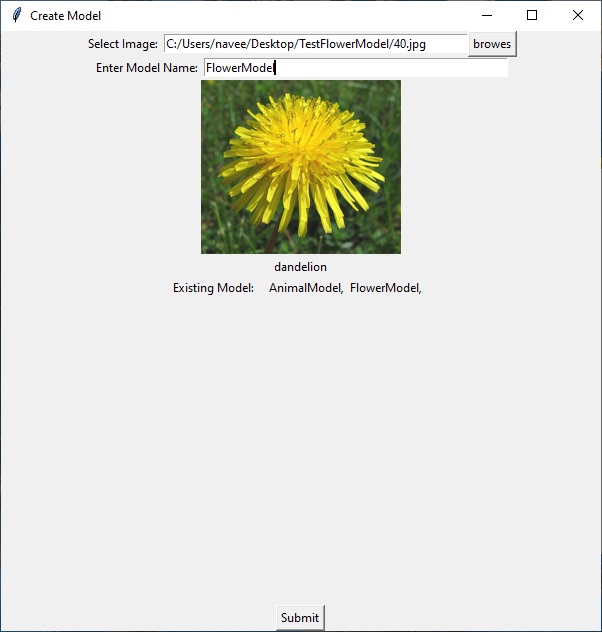
Testing Flower Model having accuracy 44%



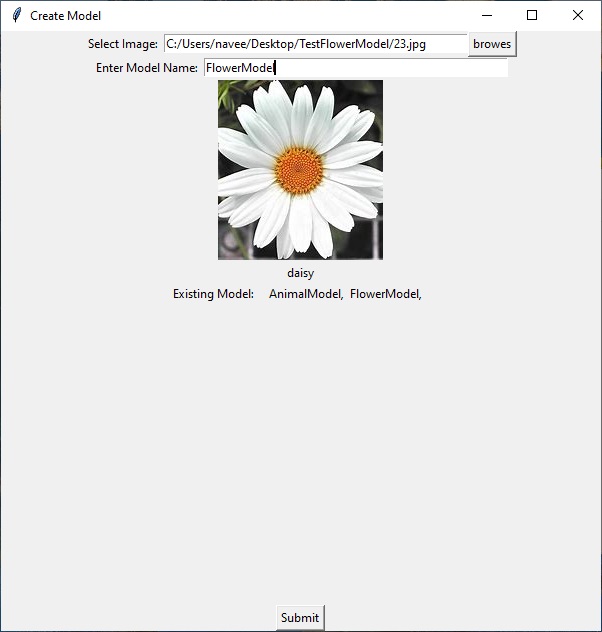
SCREENSHOT 22



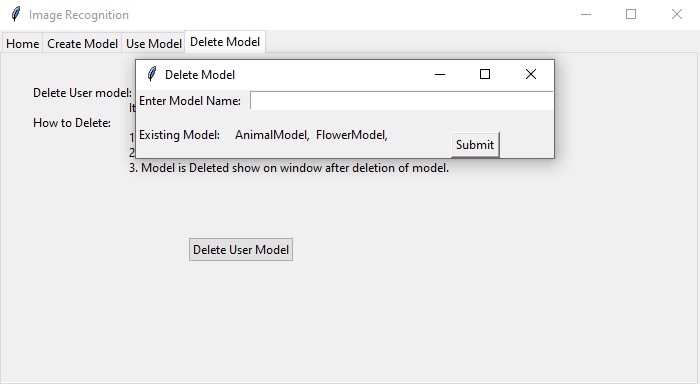
SCREENSHOT 23



SCREENSHOT 24



SCREENSHOT 25

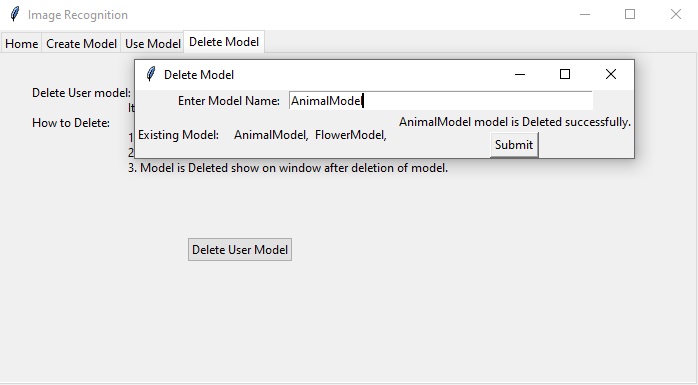


SCREENSHOT 26

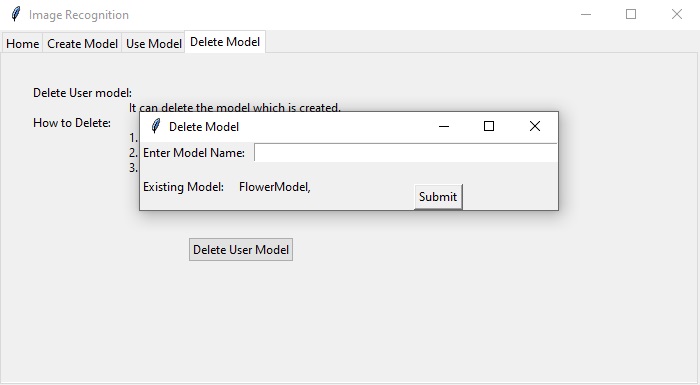
As accuracy of AnimalModel is lass. So to delete this model.

Enter Model Name: Enter model which is need to delete.

Submit: Then model delete or not shown as in Screenshot 27.



SCREENSHOT 26



SCREENSHOT 28

When reopen Delete Model.

**CHAPTER 7: CODING**

Hear 4 file are in the project 1 is to combine all files and other can be run indivigually.

**7.1 ImageRecognition.py:**

Code:

import tkinter as tk

from tkinter import ttk

from CreateModel import CreateModel

from UseModel import UseModel

from DeleteModel import DeleteModel

class ImageRecognition:

def \_\_init\_\_(self):

self.modelname="MyModelUsed"

def windowBox(self):

root = tk.Tk()

root.title("Image Recognition")

tabControl = ttk.Notebook(root)

tab1 = ttk.Frame(tabControl)

tab2 = ttk.Frame(tabControl)

tab3 = ttk.Frame(tabControl)

tab4 = ttk.Frame(tabControl)

tabControl.add(tab1, text ='Home')

tabControl.add(tab2, text ='Create Model')

tabControl.add(tab3, text ='Use Model')

tabControl.add(tab4, text ='Delete Model')

tabControl.pack(expand = 1, fill ="both")

ttk.Label(tab1, text ="\t\t\t\tIMAGE RECOGNITION PROJECT\nModel Process:\n\t\tCreate ModelUse ModelDelete Model\nCreate Model:\n\t\tModel of Data Set is need to create for Prediction.\nUse Model:\n\t\tModel Used for Prediction of Image.\nDelete Model:\n\t\tModel can be deleted is not needed or Not satisfactory.\nCreated by:\t\t\t\t\t\t\t\tTo:\n NAVEEN ARYA\t\t\t\t\t\t\t\tDR. REMESH KUMAR KAIT\nMCA VI\t\t\t\t\t\t\t\t\tDCSA\nD.ROLL NO.: 130\n U.ROLL NO.: 8100457").grid(column = 0, row = 0, padx = 30, pady = 30)

ttk.Label(tab2, text ="Create User Modele:\n\t\tTo make a model for Prediction there are some requirements. It will take time when model is building.\nRequirements:\n\t\t1. Data set folder is required to hold the classification folders of data.\n\t\t2. Classification folders are required to hold data Images related to the class.\n\t\t3. Each class should contain sufficient number of Images of that class.\nHow to Create:\n\t\t1. Click on Button Create User Model. New Window is shown.\n\t\t2. Click on Button Browse to browse image.\n\t\t3. Select any image from the dataset.\n\t\t4. Enter Model Name and submit.\n\t\t5. When Model Bullied Accuracy of model is shown.").grid(column = 0, row = 0, padx = 30, pady = 30)

ttk.Label(tab3, text ="Use User Model:\n\t\tIt can use all user model which are Created.\nHow to use:\n\t\t1. Click on Button Use User Model. New window is Shown\n\t\t2. Click on Button Browse to browse image for prediction.\n\t\t3. Enter Model name which is exist and submit.\n\t\t4. Image and Prediction is shown on window.").grid(column = 0, row = 0, padx = 30, pady = 30)

ttk.Label(tab4, text ="Delete User model:\n\t\tIt can delete the model which is created.\nHow to Delete:\n\t\t1. Click on Button delete User Model. New window is Shown\n\t\t2. Enter name of the model to delete.\n\t\t3. Model is Deleted show on window after deletion of model.").grid(column = 0, row = 0, padx = 30, pady = 30)

ttk.Button(tab2, text = "Create User Model", command=self.CreateUserModel).grid(column = 0, row = 1, padx = 30, pady = 30)

ttk.Button(tab3, text = "Use User Model", command=self.UseUserModel).grid(column = 0, row = 1, padx = 30, pady = 30)

ttk.Button(tab4, text = "Delete User Model", command=self.DeleteUserModel).grid(column = 0, row = 1, padx = 30, pady = 30)

root.mainloop()

def CreateUserModel(self):

self.modelname = "MyModel"

cmod=CreateModel()

cmod.windowBox()

def UseUserModel(self):

self.modelname = "MyModel"

umod=UseModel()

umod.windowBox()

def DeleteUserModel(self):

self.modelname = "MyModel"

dmod=DeleteModel()

dmod.getpath()

dmod.windowBox()

if \_\_name\_\_=='\_\_main\_\_':

obj=ImageRecognition()

obj.windowBox()

**7.2 CreateModel.py:**

Code:

# preprocessing

import os

import pickle

import numpy as np

from pathlib import Path

from sklearn.metrics import accuracy\_score,confusion\_matrix

from skimage.io import imread

from skimage.transform import resize

from sklearn.model\_selection import train\_test\_split

from sklearn.model\_selection import GridSearchCV

from sklearn import svm

from tkinter import \*

from tkinter import filedialog

class CreateModel:

def \_\_init\_\_(self):

self.target = []

self.images = []

self.flat\_data = []

def getPath(self):

self.path=self.entry1.get()

self.name=self.entry2.get()

f=True

for a in os.listdir(self.paths):

if a==self.name:

f=False

if f:

self.trainSet()

def getLable(self):

self.paths=os.path.join(os.getcwd(),'UserAccounts')

str1="Existing Model:\t"

for a in os.listdir(self.paths):

str1+=a+', '

return str1

def openFile(self):

filepath=filedialog.askopenfilename(initialdir="C:\\",

title="Open Image",

filetypes=(("Image file",".jpg"),

("Image file",".jpeg"),

("Image file",".png")))

file=open(filepath,'r')

self.entry1.insert(0,filepath)

file.close

def windowBox(self):

str1=self.getLable()

Cwindow =Tk()

frm1 = Frame(Cwindow)

frm1.pack(side=TOP)

name = Label(frm1, text = "Select Image: ").pack(side=LEFT)

self.entry1=Entry(frm1,width = 50)

self.entry1.pack(side=LEFT)

frm2 = Frame(Cwindow)

frm2.pack(side=TOP)

name1 = Label(frm2, text = "Enter Model Name: ").pack(side=LEFT)

self.entry2=Entry(frm2,width = 50)

self.entry2.pack(side=LEFT)

b1 = Button(frm1,text="Browse",command=self.openFile).pack(side=RIGHT)

self.lb = Label(Cwindow)

self.lb.pack(side=TOP)

modelsname = Label(Cwindow,text = str1).pack(side=TOP)

b2 = Button(Cwindow,text="Submit",command=self.getPath).pack(side=BOTTOM)

Cwindow.title("Create Model")

Cwindow.geometry("800x200")

Cwindow.mainloop()

def trainSet(self):

paths=Path(r""+self.path)

paths=paths.parent.parent

print(paths)

DATADIR=paths

CATEGORIES=os.listdir(DATADIR)

self.cat=CATEGORIES

for category in CATEGORIES:

class\_num = CATEGORIES.index(category)# label Encoding the values

path = os.path.join(DATADIR,category)#Create path to use all the images

for img in os.listdir(path):

img\_array=imread(os.path.join(path,img))

img\_resized=resize(img\_array,(250,250,10))

self.flat\_data.append(img\_resized.flatten())

self.images.append(img\_resized)

self.target.append(class\_num)

flat\_data = np.array(self.flat\_data)

images = np.array(self.images)

target = np.array(self.target)

#Split Data into Traning and Testing

x\_train,x\_test,y\_train,y\_test = train\_test\_split(flat\_data,target,test\_size=0.1,random\_state=109)

self.trainModel(x\_train,x\_test,y\_train,y\_test)

def trainModel(self,x\_train,x\_test,y\_train,y\_test):

param\_grid = [

{'C':[1,10,100,1000],'kernel':['linear']},

{'C':[1,10,100,1000],'gamma':[0.001,0.0001],'kernel':['rbf']}

]

svc=svm.SVC(probability=True)

clf=GridSearchCV(svc,param\_grid)

clf.fit(x\_train,y\_train)

y\_pred = clf.predict(x\_test)

accu=round(accuracy\_score(y\_pred,y\_test)\*100)

self.lb.configure(text=self.name+' model is Created successfully accuracy is '+str(accu)+' %')

self.lb.text=self.name+' model is Created successfully accuracy is '+str(accu)+' %'

paths=self.findPath()

pickle.dump(self.cat,open(os.path.join(paths,'cat.txt'),'wb'))

pickle.dump(clf,open(os.path.join(paths,'Model.p'),'wb'))

def findPath(self):

paths=os.path.join(os.getcwd(),'UserAccounts')

os.mkdir(paths+'\\'+self.name)

return os.path.join(paths,self.name)

if \_\_name\_\_=='\_\_main\_\_':

obj=CreateModel()

obj.windowBox()

**7.3 UseModel.py:**

Code:

import pickle

import tkinter as tk

import os

import numpy as np

from skimage.io import imread

from skimage.transform import resize

from tkinter import \*

from tkinter import filedialog

from PIL import Image, ImageOps, ImageTk

class UseModel:

def \_\_init\_\_(self):

self.path=None

def getPath(self):

self.path=self.entry1.get()

self.name=self.entry2.get()

f=False

for a in os.listdir(self.paths):

if a==self.name:

f=True

if f:

self.testModel()

else:

self.lb.configure(text='Model Not Exist.')

self.lb.text='Model Not Exist.'

def testModel(self):

paths=self.findPath()

model=pickle.load(open(paths+'\\model.p','rb'))

cat=pickle.load(open(paths+'\\cat.txt','rb'))

flat\_data =[]

img=imread(self.path)

imgl=Image.open(self.path)

imgl.thumbnail((200,200))

im = ImageTk.PhotoImage(imgl)

self.lbl.configure(image=im)

self.lbl.image=im

img\_resized= resize(img,(250,250,10))

flat\_data.append(img\_resized.flatten())

flat\_data = np.array(flat\_data)

y=model.predict(flat\_data)

self.lb.configure(text=cat[y[0]])

self.lb.text=cat[y[0]]

def getLable(self):

self.paths=os.path.join(os.getcwd(),'UserAccounts')

str1="Existing Model:\t"

for a in os.listdir(self.paths):

str1+=a+', '

return str1

def openFile(self):

filepath=filedialog.askopenfilename(initialdir="C:\\",

title="Open Image",

filetypes=(("Image file",".jpg"),

("Image file",".jpeg"),

("Image file",".png")))

file=open(filepath,'r')

self.entry1.insert(0,filepath)

file.close

def windowBox(self):

str1=self.getLable()

Uwindow =tk.Toplevel()

frm1 = Frame(Uwindow)

frm1.pack(side=TOP)

name = Label(frm1, text = "Select Image: ").pack(side=LEFT)

self.entry1=Entry(frm1,width = 50)

self.entry1.pack(side=LEFT)

frm2 = Frame(Uwindow)

frm2.pack(side=TOP)

name1 = Label(frm2, text = "Enter Model Name: ").pack(side=LEFT)

self.entry2=Entry(frm2,width = 50)

self.entry2.pack(side=LEFT)

button1 = Button(frm1,text="browes",command=self.openFile).pack(side=RIGHT)

self.lbl = Label(Uwindow)

self.lbl.pack(side=TOP)

self.lb = Label(Uwindow)

self.lb.pack(side=TOP)

modelsname = Label(Uwindow,text = str1).pack(side=TOP)

button2 = Button(Uwindow,text="Submit",command=self.getPath).pack(side=BOTTOM)

Uwindow.title("Create Model")

Uwindow.geometry("600x600")

Uwindow.mainloop()

def findPath(self):

paths=os.path.join(os.getcwd(),'UserAccounts')

return os.path.join(paths,self.name)

if \_\_name\_\_=='\_\_main\_\_':

obj=UseModel()

obj.windowBox()

**7.4 DeleteModel.py:**

Code:

# preprocessing

import os

import pickle

import tkinter as tk

from pathlib import Path

from tkinter import \*

class DeleteModel:

def \_\_init\_\_(self):

self.paths=None

def getModel(self):

name=self.entry1.get()

f=False

for a in os.listdir(self.paths):

if a==name:

f=True

if f:

self.deleteModel(name=name)

def getLable(self):

str1="Existing Model:\t"

for a in os.listdir(self.paths):

str1+=a+', '

return str1

def windowBox(self):

str1=self.getLable()

Dwindow =tk.Toplevel()

listModel=os.listdir(self.paths)

frm1 = Frame(Dwindow)

frm1.pack(side=TOP)

name = Label(Dwindow,text = str1).pack(side=LEFT)

name1 = Label(frm1,text = "Enter Model Name: ").pack(side=LEFT)

self.entry1=Entry(frm1,width = 50)

self.entry1.pack(side=LEFT)

self.lb = Label(Dwindow)

self.lb.pack(side=TOP)

b1 = Button(Dwindow,text="Submit",command=self.getModel).pack(side=BOTTOM)

Dwindow.title("Delete Model")

Dwindow.mainloop()

def deleteModel(self,name):

p=os.path.join(self.paths,name)

for a in os.listdir(p):

os.remove(os.path.join(p,a))

os.rmdir(os.path.join(self.paths,name))

self.lb.configure(text=name+' model is Deleted successfully.')

self.lb.text=name+' model is Deleted successfully.'

def getpath(self):

self.paths = os.path.join(os.getcwd(),'UserAccounts')

if \_\_name\_\_=='\_\_main\_\_':

obj=DeleteModel()

obj.getPath()

obj.windowBox()

**CHAPTER 8: FUTURE SCOPE**

Image recognition technology has revolutionized online visualization with its applications in facial-recognition, driverless cars, medical disease identification, and even in the areas of education. The future of image recognition applications is extensive and we discussed 7 of them.

**8.1 Improving Augmented Reality Gaming and Applications:**

Fairly new and exciting, but rapidly progressing – Augmented reality has given a whole new perspective to ‘daydreaming with your eye open’. The gaming arena has started using image recognition technology coupled with augmented reality to their advantage, as it helps to provide gamers with a realistic experience. Of course, developers benefit greatly as they can utilize image recognition in creating realistic gaming environments and characters.

Various non-gaming augmented reality applications are boosted by image recognition, like Crowd Optics, and Blip par – crowd behaviour monitoring and augmented reality advertising applications, respectively.

**8.2 Assisting in the Educational System:**

What good is technology if it is not solving the various problems faced by humans. The Image recognition system is adding great value in the educational sector by enabling students with learning disabilities to register knowledge in a way that is easier for them. For example, applications that rely on computer vision, allow text-to-speech options –this greatly assists visually impaired or dyslexic students to read the content.

This isn’t the limitation of image recognition technology’s contribution to the student bodies. It is also helping in breaking free from the traditional teaching boundaries and equipping educators with high-tech learning tools.

**8.3 Optimizing Medical Imagery:**

Considering this is the rise of photo-centric age, images, pictures, and video is of higher preference – Medicine field isn’t far behind with their medical data consisting of 90% of medical images, making this their largest data source in healthcare.

So, connecting one dot with another, these medical images will be trained by the smart image recognition technology to revolutionize the art of diagnosis – making detection of severe diseases, including cancer easier

Senior Manager for intelligent information systems at IBM Research, John Smith informed that detection of melanoma (a type of cancer), is one of the promising automated image processing applications.

**8.4 Boosting Driverless Car Technology:**

The technology for a near-perfect smooth journey in a self-driving car is still in the works, as several incidents of autonomous vehicles crashing emerged. However, this definitely is the future of the automotive industry.

Computer vision is one of the key components fuelling self-driving technology, including the enhancement of safety features. In addition, AI is implemented to identify objects on the roads; which includes other vehicles, sharp turns, people, pathways, and moving objects in general.

AI, ML, computer vision, coupled with Image recognition is actively being incorporated by many companies aiming to introduce driverless cars to their market.

Image recognition technology, in particular, is enabling speed prediction, the behaviour of other moving objects, and location. Since image recognition backed by AI and ML, in a nutshell, is facilitating inanimate technology with a pair of eyes – Researchers are close to creating AI that would allow cars to see during the dark. Pretty impressive.

**8.5 Giving Machines a Vision:**

As discussed above, image recognition applications help identify objects, visualize them further, analyse them, and then enable machines to make decisions accordingly with respect to the visual inputs received.

Apart from the discussed uses of IR, it has been embedded in vital manufacturing and industrial processes, which opens possibilities to an impeccable production process world-wide. What more is that this technology to take visual inputs intelligently allows machines to detect defects in the manufacturing structure, without depending on human assistance. Of course, AI follows closely, operating to fetch necessary information by learning the behaviour on its own.

**8.6 Iris Recognition Improvement:**

Iris recognition has been improved considerably with the help of image recognition technology that recognizes the unique patterns in the iris. One of the most important and essential applications of iris recognition is biometric identification.

However, an innovative application of this would be its integration in high-end flagship smartphones; Samsung Galaxy Note 7, Galaxy S8, and Windows Lumia 950, to name a few. Unfortunately, smartphones’ security wasn’t guaranteed through iris recognition. Nonetheless, this technology in the works still has given way to the visualization of many new upcoming technologies that can rely on iris recognition

**CHAPTER 9: BIBLIOGRAPHY**

**9.1 References:**

I have done this project under the guidance from DCSA.

* <https://www.javatpoint.com/python-tutorial> code Understanding
* <https://github.com> dataset folders .zip
* https://www.edureka.co code Understanding