BRAIN TUMOR PREDICTION USING MACHINE LEARNING AND DEEP LEARNING ALGORITHMS

BY

SIVAGAMI. R

20PCS007

Project Report Submitted

In partial fulfillment of the requirements for the Award of

Master of Science in Computer Science

DEPARTMENT OF COMPUTER SCIENCE

AVINASHILINGAM INSTITUTE FOR HOME SCIENCE AND

HIGHER EDUCATION FOR WOMEN

COIMBATORE – 641043

MAY – 2022

BRAIN TUMOR PREDICTION USING MACHINE LEARNING AND DEEP LEARNING ALGORITHMS

BY

SIVAGAMI. R

20PCS007

Project Report Submitted

In partial fulfillment of the requirements for the Award of

Master of Science in Computer Science

DEPARTMENT OF COMPUTER SCIENCE

AVINASHILINGAM INSTITUTE FOR HOME SCIENCE AND HIGHER EDUCATION FOR WOMEN

COIMBATORE – 641043

MAY – 2022

Signature of the Head of the Department **Signature of Supervisor**

Viva-voce Examination Held on _____

Signature of Examiners

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

I would like to express my sincere thanks to **God Almighty,** for his constant love and grace that he has shown upon me, which kept me in good health, and sound mind without which my project would not have reached a successful end.

I would like to express my deep sense of reverential gratitude and sincere thanks to **Prof. S. P. Thyagarajan, Chancellor,** Avinashilingam Institute of Home Science and Higher Education for Women, Coimbatore, for the opportunity given to me for undertaking this study and for providing all the needed facilities during the course of my study.

I owe my great deal of gratitude to **Dr. V. Bharathi Harishankar, Vice- Chancellor,** Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for extending all resources that facilitated the conduct of the present study.

I express my gratitude to **Dr. S. Kowsalya, Registrar,** Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for providing all facilities necessary for the study.

I would express my boundless thanks to **Dr. G. Padmavathi, Dean, School of Physical Sciences & Computational Sciences,** Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for granting the facility required.

I wish to place on record my deep sense of gratitude to **Dr. Vasantha Kalyani David, Professor and Head, Department of Computer Science** for support and encouragement to complete the project.

I express my heart full gratitude to my esteemed mentor **Dr. V. Radha, Professor, Department of Computer Science,** for imparting the tremendous assistance and well-timed support for triumph of our project with guidance and constant supervision as well as for providing necessary resources for the project and also for her support in completing the project.

I am grateful to the project coordinator **Dr. B. Kalpana, Professor, Department of Computer Science,** who was instrumental in granting me the facilities required for doing project.

Finally, yet importantly, I would like to thank my **parents, family members and friends** for their kind inspiration, support, encouragement, blessings and prayers, which were instrumental in the successful completion of the project.

I have great pleasure in expressing my deep sense of gratitude to all other teaching and non-teaching staff members of the Department of Computer Science, who stood behind the screen for the completion of the project.

I would extend my hearty thanks to one and all that helped me directly or indirectly for the successful completion of my project.

ABSTRACT

ABSTRACT

The formation of abnormal cells in the brain, some of which may lead to cancer, is known as a tumor. Brain tumor can be classified into two types: benign and malignant. Timely and prompt disease detection and treatment plan leads to improved quality of life and increased life expectancy in these patients. The location of the tumor, shape, and size place an important roll significant for tumor identification.

The Machine Learning and Deep Learning algorithms are used to detect brain tumors. When these algorithms are used on MRI scans, it is possible to predict brain tumors rapidly and with more accurately, which aids in the delivery of treatment to patients at an early stage.

Magnetic Resonance Imaging (MRI) scans are the most common tool for detecting tumors. Information on abnormal tissue growth in the brain is identified using MRI images. Machine Learning and Deep Learning algorithms are used by many researchers to detect tumors. When these algorithms are applied to MRI images, it is possible to predict a tumor quickly and with more accuracy, which aids in the treatment of patients. The doctor can use these predictions to make timely decisions. A self-defined Logistics Regression, Support Vector Machine (SVM), and Convolutional Neural Network (CNN) are used to detect the presence of a brain tumor in the suggested study, and their performance is evaluated based on the input images of brain tumor.

CONTENTS

TABLE OF CONTENTS

INTRODUCTION

1. INTRODUCTION

A brain tumor is a disease that develops when a tumor in the brain grows abnormally. New cells are typically created in our bodies to supplant old and harmed cells in a controlled way. Cancer cells, then again, keep on duplicating wildly in case of a brain tumor. As per the National Brain Tumor Society, around 70,000 people in the United States have an essential brain tumor. In India, the second most normal cancer is a brain tumor.

Magnetic Resonance Imaging [MRI] checking recognizes the presence of a tumor. The MRI checking ought to be analyzed by a doctor, and medicines ought to then be started relying upon the result, and this interaction might require some investment. To address this issue, proposed research proposes an automated technique for determining whether a patient has a brain tumor. This approach can aid the doctor in making early decisions, allowing them for more effective treatment. The proposed technique for preparing the model for this twofold issue incorporates strategic relapse, support vector machine (SVM), and convolutional brain organization (CNN). The expression "accuracy" is utilized to portray the method involved with assessing it.

In the created model/approach, we upgraded the dataset (MRI brain images), performed information preprocessing steps to change over the crude information, and examined two machine learning models, Logistic Regression and Support Vector Machine (SVM), as well as a profound learning model, Convolutional Neural Network (CNN). The similar examination is introduced in the outcomes segment. Every one of the previously mentioned calculations are utilized in this work contingent upon the calculation intricacy, figuring time, and different outcomes. This programmed discovery framework can assist the specialist with pursuing early choices and, thus, start treatment sooner.

1.1 ABOUT BRAIN TUMOR

A mind growth is an abnormal movement of cells in the frontal cortex. The brain life structures is very confounded, with a few areas answerable for different sensory system capacities. Brain tumors can show up anyplace on the mind or in the skull, including the defensive coating, the underside of the mind (skull base), the brainstem, the sinuses and nasal cavity, and numerous different spots. Contingent upon which tissue they rise out of, there are more than 120 distinct types of growths that can fill in the brain.

In the United States, roughly 30 persons out of 100,000 are affected with brain and nerve system tumors. Since brain tumors can come down on solid segments of the brain or spread into those areas, they are hurtful. Some brain tumors are dangerous or can possibly become tumorous. Assuming they discourage the progression of liquid encompassing the mind, bringing about an expansion in tension inside the skull, they can cause intricacies. Through the spinal liquid, a few tumors can spread to different pieces of the brain or spine.

A brain tumor is a brain injury. An injury is a little area of harmed tissue. All sores are tumors, however not all tumors are injuries. Brain sores can be brought about by strokes, wounds, encephalitis, or arteriovenous abnormality. Not all tumorous brain tumors are tumorous. Brain tumors that are benign are not tumorous.

Benign tumors develop gradually, have unmistakable lines, and seldom spread. Indeed, even benign tumors can cause intricacies. By harming and compacting region of the brain, they can cause serious brokenness. Benign tumors in a basic region of the brain can be deadly. Just a little level of benign tumors progress to tumor. Meningioma, vestibular schwannoma, and pituitary adenoma are instances of tumorous tumors.

Deep learning (a sort of machine learning) has detonated in ubiquity lately, building up momentum in practically every industry that includes independent direction, including financial matters, medical care, showcasing, and deals. In medical care, machine learning and deep learning have shown guarantee in an assortment of regions, including illness determination by means of clinical imaging, careful robots, and further developing emergency clinic execution.

Brain tumor discovery has become ordinary in the present clinical world. A brain tumor is a curved mass of tissue wherein the cells expand rapidly and fiercely, for instance the cells' advancement is uncontrolled. To extricate unusual cancer locales from the mind, picture division is utilized. The capacity to distinguish the presence of growth in the Brain requires exact division of brain tumor tissue in a MRI (attractive reverberation picture). There is an abundance of secret data in the medical care industry. With the appropriate use of solid information mining order calculations, early infection forecast can be accomplished. AI (ML) and information mining methods are basic in the clinical business. By far most of it is carried out really.

A tumor is brought about by the uncontrolled and fast expansion of cells in the brain. In the event that not treated sufficiently early, it can prompt demise. Notwithstanding various huge endeavors and promising outcomes, precise division and characterization stay a test.

A brain tumor is a sickness that happens when unusual synapses outgrow control. The two most normal sorts of brain tumors are benign (no tumorous) and malignant (tumorous). The endurance pace of a growth inclined patient is hard to anticipate on the grounds that brain tumors are extraordinary and arrived in an assortment of shapes and sizes. As indicated by UK growth research, around 15 out of each and every 100 individuals determined to have cerebrum cancer will live for quite some time or longer. The kind of tumor, the anomaly of the cells, and the area of the growth in the cerebrum all impact therapy choices for brain tumors.

Malignant brain tumors are tumorous tumors. They normally spread rapidly and invade solid pieces of the brain. Brain tumor can possibly be deadly because of the progressions it causes to the brains fundamental designs. 1malignant cancers that emerge in or influence the nose incorporate olfactory neuroblastoma, chondrosarcoma, and medulloblastoma.

Brain tumors that begin in the brain are known as primary brain tumors. Meningioma and glioma are two malignancies that most commonly arise in the brain. These tumors can break free and spread to different region of the brain and spinal line once in a long while. Tumors that have spread to the mind from different districts of the body are more normal. Primary brain tumors are nearly four times as prevalent as metastatic brain tumors. They have the ability to quickly proliferate, crowding or penetrating neighboring brain tissue.

Brain tumors can appear anywhere in the brain, but they are more common in the following areas:

1. Meningioma develop in the meninges, the brain's protective coating.
2. Tumors of the pituitary gland are known as pituitary tumors.
3. Medulloblastomas are tumors of the cerebellum or brainstem.
4. Skull base tumors create on the underside of the brain, otherwise called the skull base.
5. Other kinds of brain tumors are grouped by the cells that make them up.

1.2 Children's Brain Tumors

Every year, around 5,000 kids in the United States are determined to have brain tumors, which are the most predominant strong tumor in youngsters and youths. Astrocytomas (e.g., glioblastoma multiforme), gliomas, ependymomas, and medulloblastomas are types of brain growths that can influence youngsters.

1.3 Brain Tumor Symptoms

Since various region of the brain oversee various exercises, the side effects of a brain tumor growth will vary contingent upon where the tumor is found. A tumor growth in the cerebellum toward the rear of the head, for instance, could bring on some issues with development, strolling, equilibrium, and coordination. Vision modifications might create assuming the growth influences the optic pathway, which is answerable for vision.

The following are some of the most prevalent signs of a brain tumor:

- Dazedness.
- Seizures or spasms.
- Changes in character or conduct.
- Loss of equilibrium, dazedness, or insecurity in one piece or side of the body.
- Hearing misfortune.
- Shortcoming, deadness, or loss of motion in one section or side of the body.
- Ones perspective on things changes.
- Disarray and confusion.
- Cognitive decline.

Brain Tumor with no symptoms

Side effects are not generally present with brain tumors. The most widely recognized brain tumors in grown-ups, meningioma, develops so sluggishly that it frequently slips through the cracks. Tumors may not cause side effects until they have developed huge enough to slow down sound brain tumors tissue.

Causes and Risk Factors for Brain Tumors

Doctors are baffled as to why some cells transform into tumor cells. It could be caused by a person's genes, their surroundings, or both.

Some conceivable brain tumor causes and take a chance with factors include:

- Genetic abnormalities that predispose a person to excessive cell production. Tumor that has spread to other parts of the body
- Some types of radiation exposure.

1.4 Finding of a Brain Tumor

- To analyze a brain tumor, a neurological test, brain imaging, and, if conceivable, a biopsy are usually utilized.
- A neurological test might incorporate an assortment of tests to evaluate neurological capacities like equilibrium, hearing, vision, and reflexes.
- Imaging methodology like CT (or CAT) outputs, MRIs, and, in interesting cases, angiography or X-beams can be utilized to analyze the growth, find its area, or potentially measure the capacity of brain.
- In the event that a biopsy (tissue test assortment and investigation) is beyond the realm of possibilities, specialists will utilize different tests to recognize the brain tumor and plan treatment. Assuming a biopsy is potential, specialists can utilize it to decide the growth grade (how forceful it is) and search for biomarkers in the tumor tissue that can assist them with sorting out what's up.

Contingent upon side effects, specialist might arrange these tests to assist with affirming the analysis and preclude different ailments:

- Lumbar cut to get an example of cerebrospinal liquid and check for tumor growth cells.
- Evoked possibilities examinations or potentially electroencephalography (EEG) review are utilized the reason for this test is to recognize electrical action in the nerves as well as cerebrum.
- A neurocognitive evaluation is utilized to survey neurocognitive capacities.

A brain tumor of grade I is the most serious type.

- Beneficial (no tumorous)
- Slow-growing
- Under a microscope, the cells appear to be almost normal.
- It's usually linked to long-term survival.
- In adults, it's extremely rare.

A brain tumor of grade II:

- Slower-than-average growth
- It might spread to surrounding normal tissue and then return (recurs)
- Under a microscope, cells appear slightly odd.
- It can sometimes reappear as a higher-grade tumor.

Brain tumor of grade III:

- Nefarious (tumorous)
- Reproduces aberrant cells on a regular basis.
- The tumor spreads to other areas of the brain that are normally healthy.
- Under a microscope, cells appear odd.
- It has a proclivity for repeating, typically as a higher-grade tumor.
- Malignant grade IV mind cancer
- The most confrontational
- It develops quickly.
- Spreads quickly into normal areas of the brain.
- Reproduces aberrant cells on a regular basis.
- Under a microscope, cells appear to be exceedingly aberrant.
- To sustain high blood flow, the tumor produces new blood vessels.

Brain tumors are identified utilizing Machine Learning and Deep Learning calculations. At the point when these calculations are applied to MRI filters, it is feasible to foresee tumor growths all the more rapidly and precisely, which assists with treatment conveyance. A tumor is the development of strange cells in the cerebrum, some of which can prompt growth. There are two sorts of tumor growths: benign and malignant. In these patients, early disease detection

and treatment leads to improved quality of life and longer life expectancy. The tumors location, shape, and size all play an important role in tumors identification. The most well-known strategy for distinguishing tumors is to utilize attractive reverberation imaging (MRI) checks. Magnetic resonance imaging (MRI) images are utilized to recognize data about strange tissue development in the brain.

To be successfully treated, brain tumors must be detected accurately and early. Not only does early detection help in the development of new drugs, but it can also save a life if caught early enough. Neuro-oncologists have benefited from the introduction of computer-assisted diagnosis and biomedical informatics in a variety of ways. Machine learning algorithms have recently been used to evaluate medical imaging and data, in contrast to manual tumor identification, which is a time-consuming process prone to human error. Computer-assisted mechanisms outperform manual traditional diagnostic procedures.

Machine Learning (ML) and Deep Learning (DL) applications to computerize medical services conveyance have been extremely intriguing and energizing lately. In detecting, predicting, diagnosing, and even recommending therapies for any disease, machine learning approaches are outperforming humans. Medical Imaging Technology has been transformed by machine learning and deep learning. Artificial Intelligence has excelled in several areas of healthcare, including the diagnosis of brain tumors. A tumor is a swollen area or section of the body created by twisted tissues in which the cells begin to multiply uncontrollably. The procedure used to recognize the tumor is attractive reverberation imaging (MRI). The attractive reverberation imaging (MRI) method is broadly used in clinical science to distinguish irregularities in any piece of the body. Tumors in the brain are significantly more dangerous and difficult to cure than tumors in other parts of the body, making early detection and monitoring of brain tumors critical.

One of the most troublesome difficulties in clinical image handling is recognizing mind growths. The test is trying to finish since the photos have a great deal of assortment, as mind tumors exist in an assortment of shapes and surfaces. Brain growths are comprised of a few sorts of cells, and the cells can uncover data about the tumor's tendency, seriousness, and extraordinariness. Tumors can show up in an assortment of regions, and the area of a growth can uncover data about the kind of cells that are making it, which can assist with additional analysis. The method involved with identifying brain growths can be made more troublesome by issues that can be found in basically all advanced images, like lighting issues. The image

powers of tumor and non-growth images can cover, making it trying for any model to make exact expectations from crude images.

1.5 MOTIVATION

The primary inspiration driving brain tumor identification isn't just to identify growths, yet in addition to group the sorts of tumors. Hence, it will in general be significant in cases, for instance, expecting they should be sure the tumor is positive or negative, it can perceive a development from image and return the result as development is positive or not. This work oversees such a system, which uses PC based strategies to recognize tumor obstructs and organize the kind of development including Convolution Neural Network Algorithm for MRI images of different patients. As needs be, it will in everyday be significant in cases, for example, tolerating they ought to be certain tumor is positive or negative, it can see an improvement from image and return the outcome as advancement is positive or not. This work directs such a construction, which utilizes PC based procedures to perceive tumor upsets and arrange the sort of advancement including Convolution Neural Network Algorithm for MRI images of various patients.

1.6 PROBLEM STATEMENT

To use CNN as a Deep Learning asset to detect and classify brain tumors, as well as to deploy a Flask System.

1.7 OBJECTIVES

- Need timely consultation.
- Provide good software for doctors to use in identifying tumors and their causes.
- Patients' time will be saved.
- Provide a timely solution.

LITERATURE REVIEW

2. LITERATURE REVIEW

Ayadi et al. [2021] proposed a CNN-based computer-assisted diagnosis (CAD) system [2]. The 18-weighted layered CNN model achieved 94.74 percent classification accuracy for brain tumor-type classification and 90.35 percent classification accuracy for tumor grading in experiments conducted on three different datasets.

Bada and Barjaktarovi [2020] created a 22-layered CNN architecture for brain tumor type classification using 3064 T1-weighted contrast-enhanced MRI images [3]. Their proposed model correctly classified meningioma, glioma, and pituitary tumors with 96.56 percent accuracy.

Cinar and Yildirim [2020] used a modified version of the pre-trained ResNet-50 CNN model for brain tumour detection, replacing the last 5 layers with 8 new layers. With this modified CNN model [4], they were able to achieve 97.2 percent accuracy using MRI images.

Khan et al. [2020] used 253 real brain MRIs with data augmentation to propose a deep learning method for classifying brain tumours as cancerous or non-cancerous [9]. They used edge detection to locate the region of interest in an MRI image before using a simple CNN model to extract the features. They had a classification accuracy of 89 percent.

Mzoughi et al. [2020] demonstrated a deep multi-scale 3D CNN model for grading brain tumors from volumetric 3D MRI images [12]. The proposed method correctly classified low-grade glioma and high-grade glioma brain tumor images with 96.49 percent accuracy.

Mehrotra et al. [2020] used a deep learning-based transfer learning technique to classify brain tumour images as malignant or benign [10]. For the classification study, the most popular CNN models such as ResNet-101, ResNet-50, GoogleNet, AlexNet, and SqueezeNet were used and compared. With the help of transfer learning and a pre-trained AlexNet CNN model, they were able to achieve the highest accuracy of 99.04 percent.

Rehman et al. [2020] proposed classifying brain tumours into glioma, meningioma, and pituitary tumours using three pre-trained CNN models known as AlexNet, GoogleNet, and VGG16 [14]. During this transfer learning approach, the VGG-16 achieved the best classification accuracy of 98.69 percent. They used 3064 MRI images of the brain from 233 patients.

Abiwinanda et al. [2019] used the simplest possible CNN architecture to recognise thr ee of the most common types of brain tumours: glioma, meningioma, and pituitary tumours, with a validation accuracy of 84.19 percent at best [1].

Deepak and Ameer [2019] used a GoogleNet CNN model that was pre-trained to distinguish between glioma, meningioma, and pituitary brain tumours [5]. In this 3-class classification problem using MRI images, the average classification accuracy was 98 percent.

Kabir Anaraki et al. [2019] proposed using MRI images and a CNN and genetic algorithm (GA)-based method to noninvasively classify different grades of glioma. They classified three glioma grades with 90.9 percent accuracy and glioma, meningioma, and pituitary tumour types with 94.2 percent accuracy [7].

Talo et al. [2019] proposed using the pre-trained ResNet-34 CNN model to detect brain tumours from MRI images [15]. Although they achieved a detection accuracy of 100 percent, the number of images used for the deep learning model was 613, which is not considered a large number for machine learning studies.

Mohsen et al. [2018] classified brain MRI images into four categories: normal brain, glioblastoma, sarcoma, and metastatic bronchogenic carcinoma tumours using a deep neural network (DNN) classifier combined with discrete wavelet transform (DWT) and principal component analysis (PCA) [11].

Pereira et al. [2018] used CNN to predict tumor grade directly from imaging data by eliminating the need for expert annotations of regions of interest. They looked at two methods for making predictions: from the entire brain and from an automatically defined tumor region [13]. They had an accuracy of 89.5 percent when predicting grade from the whole brain and 92.98 percent when predicting grade from the tumor ROI.

Yang et al. [2018] investigated the effect of CNN trained with transfer learning and fine-tuning on conventional MRI images to noninvasively classify low-grade glioma (LGG) and high-grade glioma (HGG). They achieved 86.6 percent accuracy with pre-trained GoogleNet and 87.4 percent accuracy with pre-trained AlexNet [16].

Khawaldeh et al. [2017] proposed a modified version of AlexNet CNN model to classify brain MRI images into healthy [8], low-grade glioma, and high-grade glioma. 4069 brain MRI images were used to achieve an overall accuracy of 91.16 percent.

Ertosun and Rubin [2015] developed a deep learning pipeline with an ensemble of CNN [6]. In the absence of data, which is a common problem in the domain of deep learning approaches, their method was deemed quite successful. They were 96 percent accurate in the HGG vs. LGG classification task and 71% accurate in the LGG Grade I vs. LGG Grade II classification task.

SYSTEM REQUIREMENTS & DATASET DESCRIPTION

3. SYSTEM REQUIREMENTS & DATASET DESCRIPTION

3.1 EXISTING SYSTEM

Differentiating between tumors types can be difficult. Tumor misclassification can result in the wrong treatment and medications being prescribed, which can exacerbate the problem.

Disadvantages

- It necessitates a large amount of training data, an appropriate model, and it is also time-consuming.
- It's also a time-consuming and exhausting procedure.
- Convolutional networks have been around for quite a while, however their prosperity has been restricted because of the size of the organization viable.

3.2 PROPOSED SYSTEM

Therefore, the objective of this undertaking is to make a framework that can characterize a MRI into one of the accompanying tumor types:

- Glioma Tumor
- Meningioma Tumor
- No Tumor
- Pituitary Tumor

Construct a various model specifically Logistic Regression, Support Vector Machine (SVM), and Convolution Neural Network (CNN) models and analyze the exactness, and loss of each model to confirm the best model.

Advantages

- Due to its high precision and low image preprocessing prerequisites, it is viewed as the best profound learning method for image grouping.
- It is liked over feed forward brain networks since it very well may be prepared all the more successfully to accomplish higher correctness’s on account of intricate images.

- It decreases images to a development that is less intricate to process without losing highlights that are essential for good guess by utilizing fitting filtration and reusability of weights.
- It can figure out how to play out any undertaking naturally by examining preparing information; no earlier information is required.
- Profoundly prepared hand-made image highlights, like those found in SVM, are not needed. Think about strategic relapse.

3.3 HARDWARE REQUIREMENTS

PROCESSOR **:** Intel(R) Core(TM) i5-2320 CPU @ 3.00GHz 3.30 GHz

RAM **:** 7.89 GB

SYSTEM TYPE **:** 4-bit operating system, x64-based processor

3.4 SOFTWARE REQUIREMENTS & DESCRIPTION

OPERATING SYSTEM: Windows 10 Pro

SOFTWARE: Python (Jupyter Notebook, Flask)

VISUAL STUDIO CODE

- Visual Studio Code (VS Code) is a Microsoft source-code manager for Windows, Linux, and macOS. Support for researching, semantic development featuring, savvy code flawlessness, bits, code refactoring, and implanted Git are among the parts. Clients can steer the conversation in a different direction, console reinforcement approaches, and propensities, as well as present augmentations for additional handiness.
- Visual Studio Code was named the most remarkable expert climate mechanical get together in the Stack Overflow 2021 Developer Survey, with 70% of 82,000 respondents saying they use it.

ANACONDA

- Boa constrictor is a Python and R programming language dispersion pointed toward making bundle the board and organization more straightforward in logical figuring (information science, AI applications, huge scope information handling, prescient

investigation, etc). For Windows, Linux, and macOS, the conveyance incorporates information science bundles. Boa constrictor, Inc., established by Peter Wang and Travis Oliphant in 2012, is liable for its turn of events and upkeep. Boa constrictor Distribution or Anaconda Individual Edition are two Anaconda, Inc. items, the two of which are not free.

- In Anaconda, the bundle the executives framework conda is responsible for bundle variants. This pack manager was turned out like case open-source wrap since it turned out to be useful for things other than Python. Miniconda is a little, bootstrapped type of Anaconda that simply keeps down conda, Python, their circumstances, and two or three distinct groups.

JUPYTER

- Jupyter Notebook is a free open-source web application that licenses us to make and present records to live code, conditions, depictions, and story message. Information cleaning and change, mathematical diversion, obvious appearance, information depiction, AI, and different applications are a few the normal outcomes.
- Python is one of the in excess of 40 programming dialects upheld by Jupyter. The establishment of the Jupyter Notebook itself requires Python (rendition 3.3 or higher, or Python 2.7).

Install Jupyter using Anaconda

- Download and present the Anaconda Distribution, which consolidates Python, the Jupyter Notebook, and other conventionally used legitimate enlisting and data science packs.

Python

- Flask is a Python-based miniature web system. Since it requires no particular devices or libraries, it is delegated a miniature structure.
- It misses the mark on information base reflection layer, structure approval, or some other parts for which outsider libraries as of now exist.
- Flask grants us to add application features like they were integrated into the real framework. Object-social mappers, structure endorsement, and move managing

developments, as well as various open affirmation progressions and framework related contraptions, are available.

- Flask is a Python-based micro web system. Since it requires no particular devices or libraries, it is delegated a miniature structure. It misses the mark on information base reflection layer, structure approval, or some other parts for which outsider libraries as of now exist.
- Expansions, on the other hand, can be used to add application features like they were integrated into Flask itself. Object-social mappers, structure endorsement, and move dealing with expansions, as well as various open check advances and framework related mechanical assemblies, are available.

HTML

- HTML (Hyper Text Markup Language) is a markup language that awards us to make site pages. An augmentation language is utilized to make site pages. HTML addresses Hypertext Markup Language and it is a blend of the two.
- The connection between website pages is characterized by hypertext. The text report inside the name that describes the development of pages is portrayed using a markup language.
- This language is used to make sense of text so a machine can get a handle on it and control it fittingly.
- Understandable increment lingos consolidate HTML. Names are used in the language to decide the kind of text control required.

CSS

- Cascading Style Sheets (CSS) is a fundamental arrangement language expected to make the most well-known approach to making site pages decent less difficult.
- The look and feel of a page is overseen by CSS. It can utilize CSS to change the shade of the text, the text based style, the parting between passages, the size and setup of regions, the foundation pictures or tones utilized, plan plans, combinations in show for various gadgets and screen sizes, and an assortment of different impacts.
- CSS is easy to learn and appreciate, however it gives a colossal heap of control over how a HTML record looks. CSS is often utilized associated with the markup tongues HTML or XHTML.

JS

- JavaScript, otherwise called JS, is a programming language that, alongside HTML and CSS, is one of the main advances on the Internet. On the client side, more than 97% of sites use JavaScript for website page conduct, with outsider libraries every now and again consolidated. To execute the code on the clients' gadgets, all significant internet browsers have a committed JavaScript motor.

3.5 LIBRARIES

A library is a gathering of pre-joined codes that can be utilized iteratively to diminish the time expected to code. They are especially helpful for getting to the pre-made occasionally utilized codes, rather than keeping in touch with them with next to no arranging each and every time. Like the certifiable libraries, these are a gathering of reusable assets, and that recommends each library has a root source. This is the establishment behind the different open-source libraries accessible in Python.

PYTHON LIBRARY

Python library is a collection of modules that contain functions and classes that can be used by other programs to perform various tasks. Some of the main libraries are explained related to this research work namely:

- OS
- Keras
- PIL
- Matplotlib
- Sklearn
- Numpy
- Pandas

OS

- In Python, the OS module contains capacities for connecting with the working framework. The standard utility modules in Python incorporate OS. This module permits you to utilize working framework explicit usefulness in a hurry. Many

capacities to interface with the record framework are accessible in the *os* and *os.path* modules.

Keras

- Keras is an open-source programming library for fake cerebrum networks that gives a Python interface. The TensorFlow library is gotten to through Keras.
- Keras kept a blend of backends up until structure 2.3, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML. Essentially TensorFlow is kept up with as of variety 2.4.
- It is straightforward, estimated, still up in the air to allow fast experimentation with significant cerebrum associations. It was made as a piece of the ONEIROS (Open-completed Neuro-Electronic Intelligent Robot Operating System) research project, and François Chollet, a Google engineer, is the basic creator and maintainer. Chollet is moreover the maker of the critical frontal cortex network model XCeption.

PI

- The Python Imaging Library is a free and open-source increment library for Python that adds support for opening, controlling, and saving a mix of picture record plans. It chips away at Windows, Mac OS X, and Linux. PIL 1.1.7, which was conveyed in September 2009 and keeps up with Python 1.5.2-2.7, is the latest design.
- In 2011, improvement of the primary endeavor, known as PIL, was halted. Following that, the Pillow project forked the PIL vault and added Python 3.x help. In Linux circulations, for example, Debian and Ubuntu, this fork has been taken on as a swap for the first PIL (since 13.04).

Matplotlib

- Matplotlib is an information insight and graphical plotting library for Python and mathematical advancement NumPy works across stages. As such, it gives an open source decision instead of MATLAB. Matplotlib's APIs (Application Programming Interfaces) can comparably be utilized to insert plots in GUI applications.
- A Python matplotlib script is composed so exceptionally that, when in doubt, a few lines of code should make a visual information plot. Two APIs are overlaid by the matplotlib organizing layer.

- The pyplot API is a tree of Python code objects, with matplotlib at the top.
- An OO (Object-Oriented) API game plan of articles that is more adaptable than pyplot. This API licenses you to get to Matplotlib's backend layers straightforwardly.

Sklearn

- Scikit-learn (Sklearn) is Python's most amazing and trustworthy AI library.
- It goes with a Python consistency interface that integrates portrayal, backslide, gathering, and dimensionality decline instruments for AI and quantifiable illustrating.
- NumPy, SciPy, and Matplotlib are totally utilized in this library, which is for the most part written in Python.

Numpy

- NumPy is a Python bundle for cluster handling. It incorporates an elite exhibition multi-faceted cluster object as well as devices for controlling them.
- It is the fundamental Python pack for coherent figuring. It also has different components, including the going with huge ones:
- Sophisticated (broadcasting) functions.
- A strong N-layered array object
- Significant straight factor based math, Fourier change, and sporadic number capacities.
- Gadgets for integrating C/C++ and Fortran code
- NumPy can be used as a mind boggling compartment of regular data despite its obvious sensible applications.
- Numpy permits erratic information types to be characterized, permitting NumPy to coordinate with a wide scope of data sets consistently and rapidly.

Panda

- Pandas is a Python bundle for information science, information examination, and machine learning.
- It depends on Numpy, a multi-faceted exhibit support bundle.

3.6 DATASET DESCRIPTION

In computer vision, a dataset is a carefully selected set of digital images that are used to test, train, and evaluate their algorithms' performance.

3.6.1 About Dataset

The dataset under consideration contains MRI scanned images of 2870 patients, 2475 of whom are tumorous and 395 of whom are not. The goal of the work presented here is to create a detection model that can detect a tumor in a patient's MRI scanned image. Test images are displayed in Fig 3.1 including different growths specifically giloma tumor, meningioma growth, no growth, and pituitary growth.

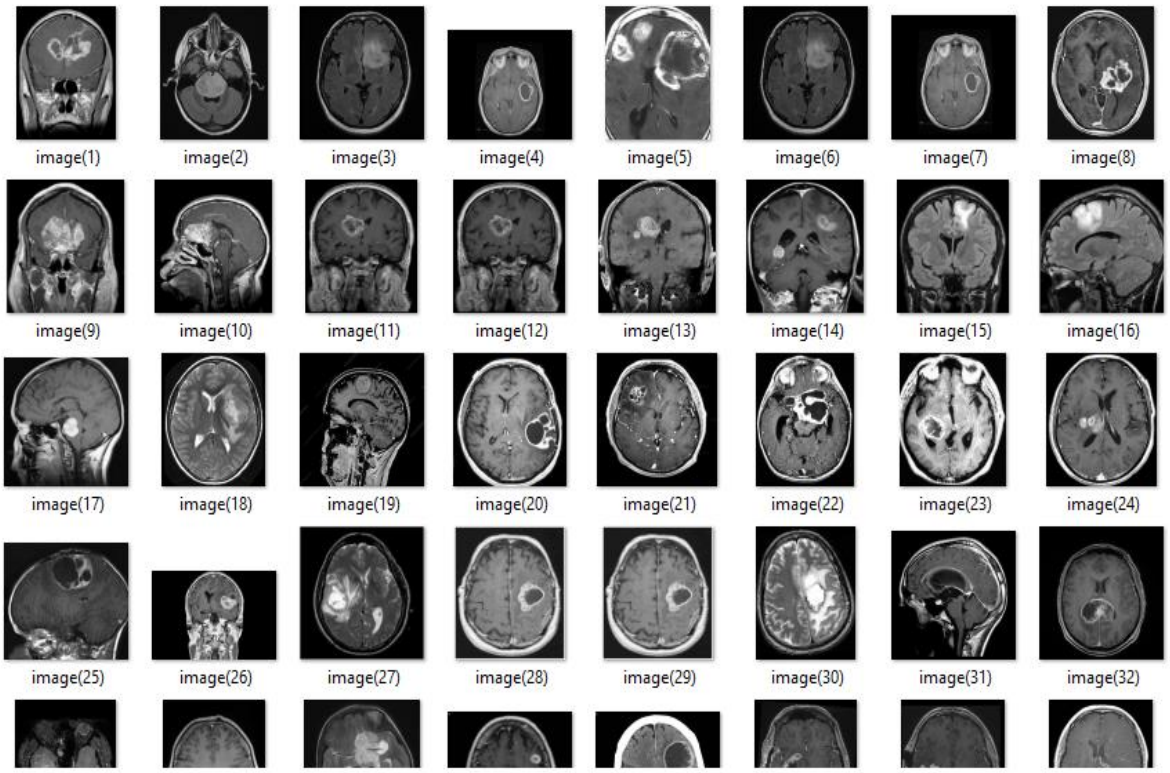


Fig.3.1 Dataset: Types of brain tumor.

RESEARCH METHODOLOGY

4. RESEARCH METHODOLOGY

The "new assessment" of any piece of investigation is suggested as exploration system. It's about how an investigator plans a pack in an organized way to ensure authentic and trustworthy results that meet the investigation's goals and targets.

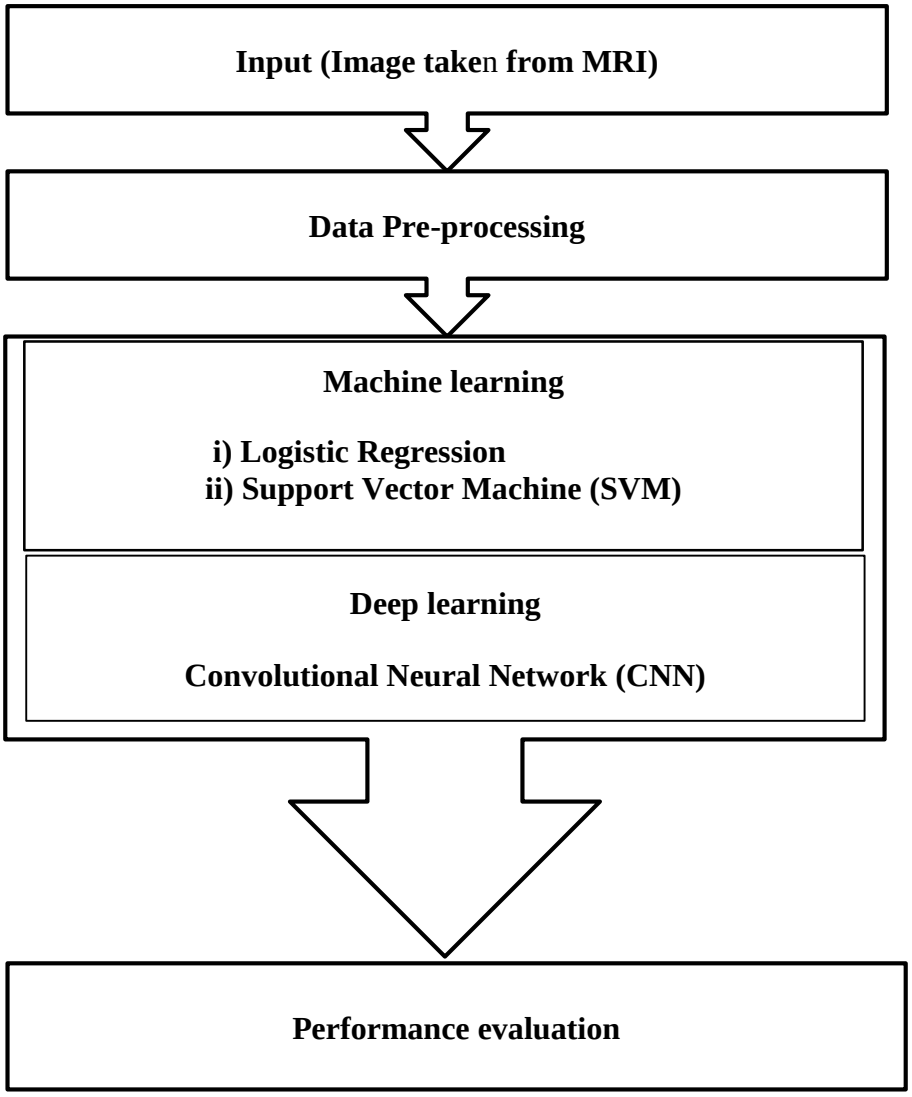


Fig.4.1 Overall methodology

The expression "research technique" or "examination configuration" alludes to, not entirely settled, and sane assortment of information with the end goal of dissecting as well as acquiring data to respond to explore questions. The method of choice is determined by the research problem and purpose, and those methods cannot be described as more appropriate. The overall methodology of the proposed work is shown in above Fig. 4.1.

Input

The patient is assumed to be in good health and capable of undergoing an MRI scan with the assistance of the doctor. The input for this study is a patient's brain MRI images.

4.1 Data Preprocessing

- For the simple understanding of the information, the information ought to be changed from its crude state. The information pre-handling steps we considered are:
- TensorFlow, Numpy, pandas, matplotlib, os, and Scikit-learn are likewise imported. Image information expansion is a strategy for expanding the quantity of images in a dataset by utilizing methods like revolution to make various variants of the image.
- The augmented data is updated to convert photographs to grayscale. Noise is eliminated through a series of erosions and dilations. Pixels from the image's edges are eroded. In contrast to erosion, dilation adds pixels to the image's edges. Utilizing the Gaussian haze method, the image is smoothed. The image is convolved with the Gaussian channel when the Gaussian haze activity is performed. The Gaussian channel is a low-pass channel that disposes of high-repeat picture parts. The picture is in this manner smoothed.
- Recognize the largest contour; a contour is a shape's dividing line or highlight. In the wake of finding the shaped image's outrageous focuses, all increased images were resized to 255x255 for Logistic Regression, Support Vector Machine, and 288x288 for CNN, and all images before pre-handling were of changing sizes. The images were edited at the limits. At long last, the data is partitioned into two classifications. At long last, the data is separated into two classifications. They are preparing and testing information, with the preparation dataset containing 2296 images and the testing dataset containing 574 images.

4.2 Algorithms used

- Logistic regression, Support Vector Machine (SVM), and Convolutional Neural Network are the algorithms used in the proposed work (CNN).

Logistic Regression:

- An administered learning plan computation used to anticipate the probability of an objective variable is known as essential backslide. Since the possibility of the goal or ward variable is dichotomous, there are only two likely classes.

- To put it another way, the reliant variable is arranged in nature, with information coded as 1 or 0.

- P(Y=1) as a part of X is normal numerically by a fundamental lose the faith model. It's perhaps the most major AI estimation, and it might be used to handle an arrangement of portrayal issues like spam revelation, diabetes assumption, and infection acknowledgment.

Types of Logistic Regression

- By and large, strategic relapse alludes to paired calculated relapse with parallel objective factors, however it can likewise foresee two extra classifications of factors. Calculated relapse is ordered into the accompanying sorts in view of the quantity of classes:

Binary or Binomial

- In paired demand, a reliant variable will have just two potential sorts either 1 or 0. For instance, these components could address achievement or disappointment, yes or no, win or fiasco, and so on.

Multinomial

- The dependent variable in a multinomial gathering can have no less than three possible unordered sorts or types with no quantitative significance. These variables could actually imply "Type A," "Type B," or "Type C," for example.

Ordinal

- In ordinal classification, a dependent variable may have three or more ordered types, each of which has a quantitative significance. These factors could, for instance, address

"poor" or "great," "awesome," or "magnificent," with scores going from 0,1,2,3 for every class.

Logistic Regression Assumptions

- Prior to plunging into the execution of strategic relapse, we should know about the accompanying presumptions about the equivalent:
- The objective factors in double calculated relapse should generally be parallel, and the ideal result is addressed by factor level 1.
- The model ought to be liberated from multi-collinearity, and that implies the free factors should be autonomous of each other in our model incorporate significant factors
- For strategic relapse, pick an enormous example size.

Regression Models

- Matched Logistic Regression Model − the most un-complex kind of essential backslide is twofold or binomial determined backslide in which the goal or ward variable can have only 2 potential sorts either 1 or 0.
- Multinomial Logistic Regression Model: In multinomial key backslide, the goal or ward variable can have no less than three expected unordered sorts, none of which have any quantitative significance.

Support Vector Machine (SVM)

- Support Vector Machine (SVM) is a renowned Supervised Learning computation for Classification and Regression issues. In any case, it is basically used in Machine Learning for course of action issues.
- The target of the SVM estimation is to find the best line or decision limit for secluding n-layered space into classes with the objective that new data centers can be helpfully situated in the right order later on. A hyperplane is the most ideal decision limit.
- SVM picks the hyperplane-outlining crazy centers/vectors. Support vectors are the outrageous cases, and the calculation is named after them. SVM calculation can be utilized for Face identification, Image arrangement, Text order, and so on.

Types of SVM

Linear SVM:

- Direct SVM is utilized for clearly discernable information, which is portrayed as information that can be accumulated into two classes utilizing a solitary straight line.

Non-linear SVM:

- Non-Linear SVM is used for non-straightforwardly disconnected data, and that truly means that if a dataset can't be described using a straight line, it is seen as non-direct data, and the classifier used is called Non-direct SVM.

Hyperplane and Support Vectors in the SVM calculation:

Hyperplane:

- In n-layered space, different lines/choice limits can be utilized to isolate the classes, however the objective is to track down the best choice limit for ordering the data of interest. The hyperplane of SVM alludes to this best limit.

- The hyperplane's not set in stone by the quantity of highlights in the dataset; in the event that there are just two elements, the hyperplane will be a straight line. Hyperplane will be a two-layered plane assuming there are three highlights.

- To make a hyperplane with a most extreme edge (the distance between relevant pieces of information).

Support Vectors:

- The information of interest or vectors that are the closest to the hyperplane and which impact the spot of the hyperplane are named as Support Vector. Since these vectors support the hyperplane, hereafter called a Support vector.

Model building for Machine learning

Machine learning models are incredible assets for performing basic assignments and taking care of complicated issues rapidly and actually. Machine learning models are fit to be conveyed in an assortment of ventures because of the remarkable development of information in the advanced world. These models have a wide scope of utilizations, incorporating machine learning in medical services and the machine learning model building steps are shown in below Fig.4.2.

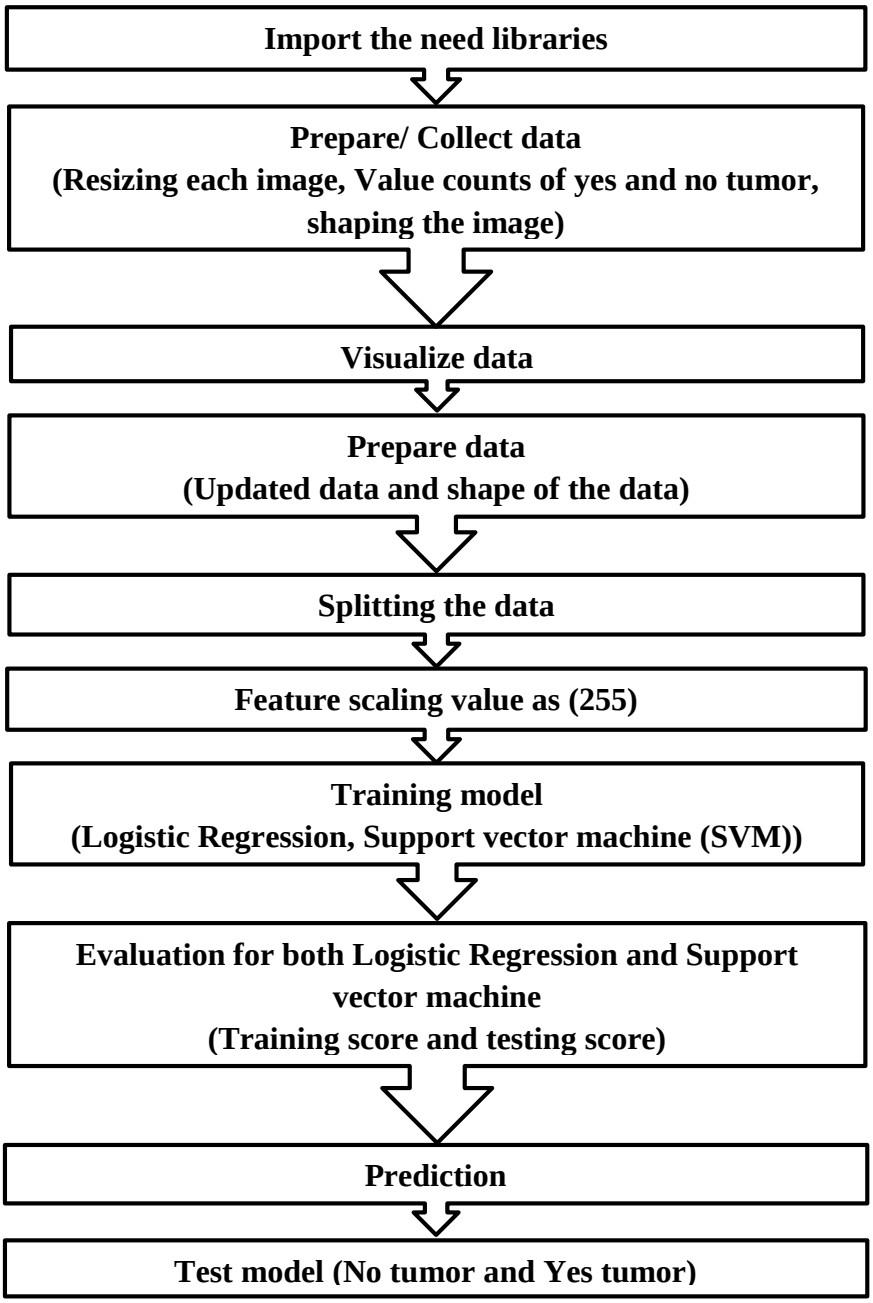


Fig.4.2 Brain tumor detection model using Machine learning techniques.

Importing the libraries os, keras, PIL, numpy, pandas, matplotlib, and sklearn. Setting path for respective folder each classes and classes have the images which contain a tumor or not are shown in below Fig.4.3. Resized image can train faster and append them into x data to do the training with data same thing will be added to the non-value means the images that doesn't contain tumor because model need to see both images with tumors ad images without tumors. Then to label the data 1 means yes and 0 means no in an array, the series value counts how may are tumors (1) and non-tumors (0) are shown in below Fig.4.4

Fig.4.3 Import the need libraries

Fig.4.4 Prepare/ Collect data

The shape property is regularly used to recover a cluster's ongoing shape, however it can likewise be utilized to reshape it set up by passing it a tuple of exhibit aspects. Use plt.imshow() with the cmap parameter set to 'grey' to display a grayscale image in matplotlib not are shown in below Fig.4.5. The train-test parting is a procedure for assessing the exhibition of an machine learning framework that includes reshaping a cluster without changing the information. Since the informational index is parted into two sets: a preparation set and a testing set, it is called Train score and Test score. Training accounts for 80% of the budget, while testing accounts for 20%. By scaling the data within the training and testing phases, the data takes on a new shape (0, 1) are shown in below Fig.4.6.

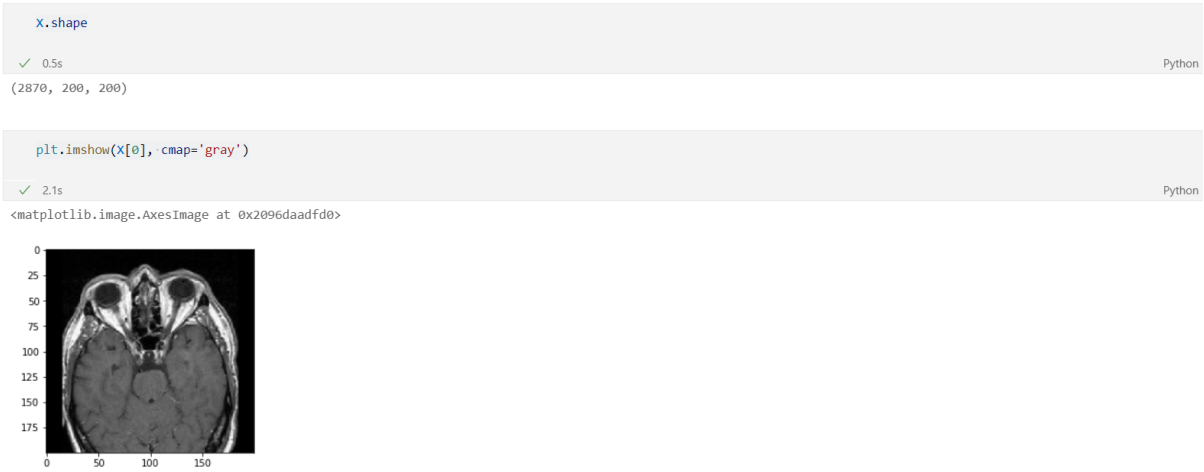


Fig.4.5 Visualize data

Fig.4.6 Shaped data are updated and splitting the data for training, testing.

Importing PCA from sklearn, naming xtrain as pca_train and xtest as pca-test. Here logistic regression and support vector machine as the model that are going to train by using sklearn. The model coefficients shrink as c decreases, according to logistic regression. All the coefficients are zero until c=0.001. The regularisation penalty is becoming more prominent, which has this effect are shown in below Fig.4.7. The reason for the Support Vector Classifier is to fit the information and return a "best fit" hyper plane that partitions or arranges the information. A method to measure the accuracy of a model is to use train and test scores. For logistic regression in an array, use the model to predict the test data are shown in below Fig.4.8.

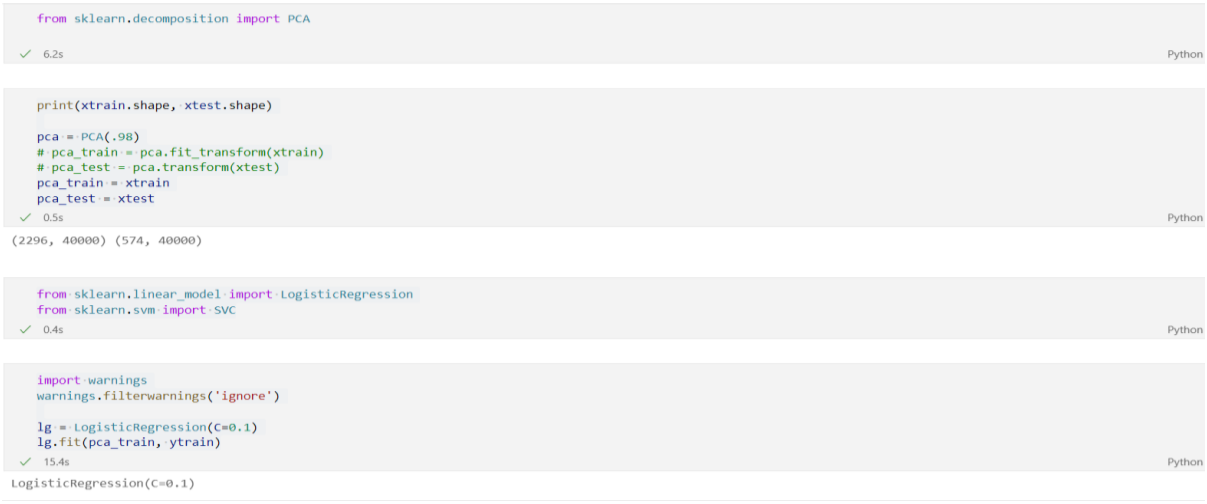


Fig.4.7 Importing logistic regression for training.

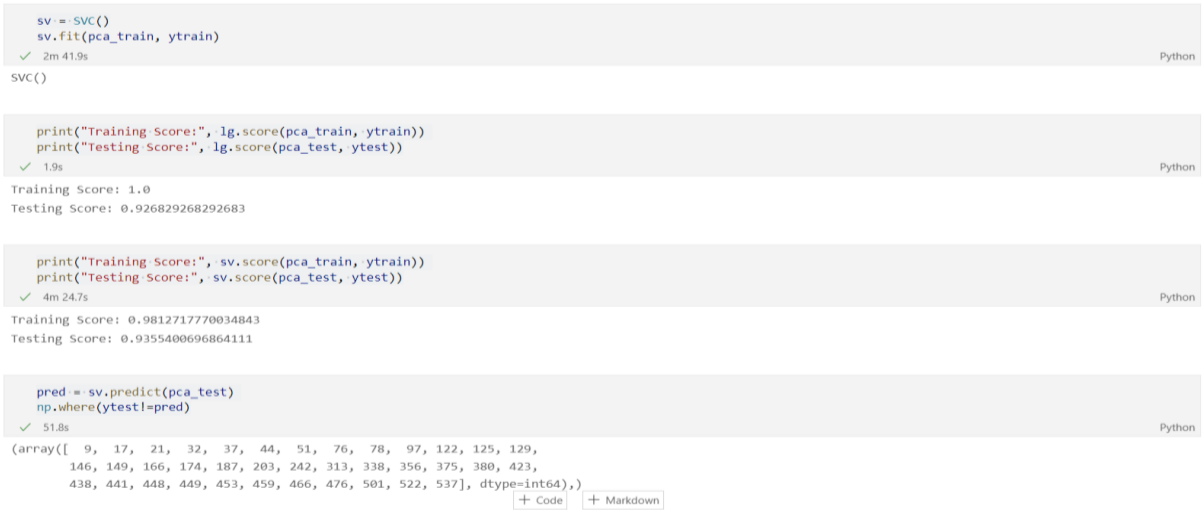


Fig.4.8 Importing support vector machine for training and evaluation for both Logistic Regression and Support vector machine (Training score and testing score).

Use the model to test data for support vector machine in array and predict the ytest. Declaring no tumor as 0, positive tumor, giloma tumor, and meningioma tumor as 1 are shown in below Fig.4.9. Plotting no tumor prediction of support vector machine for first nine images in testing data are shown in below Fig.4.10.

```
pred = lg.predict(pca_test)
np.where(ytest!=pred)
```

✓ 0.1s Python

```
(array([  8,   9,  21,  32,  33,  44,  51,  76,  78,  97, 122, 140, 142,
        146, 149, 171, 174, 177, 181, 183, 187, 203, 242, 274, 305, 338,
        342, 374, 375, 380, 409, 423, 438, 441, 453, 459, 476, 501, 512,
        522, 525, 560], dtype=int64),)
```

```
pred[10]
```

✓ 0.7s Python

```
1
```

```
ytest[10]
```

✓ 0.9s Python

```
1
```

```
dec = {0:'No Tumor', 1:'Positive Tumor',2:'glioma_tumor',3:'meningioma_tumor'}
```

✓ 0.5s Python

Fig.4.9 Prediction values in arrays.

```
plt.figure(figsize=(12,8))
p = os.listdir('Testing')
c=1
for i in os.listdir('Testing/no_tumor/')[:9]:
    plt.subplot(3,3,c)

    img = cv2.imread('Testing/no_tumor/'+i,0)
    img1 = cv2.resize(img, (200,200))
    img1 = img1.reshape(1,-1)/255
    p = sv.predict(img1)
    plt.title(dec[p[0]])
    plt.imshow(img, cmap='gray')
    plt.axis('off')
    c+=1
```

✓ 7.9s Python

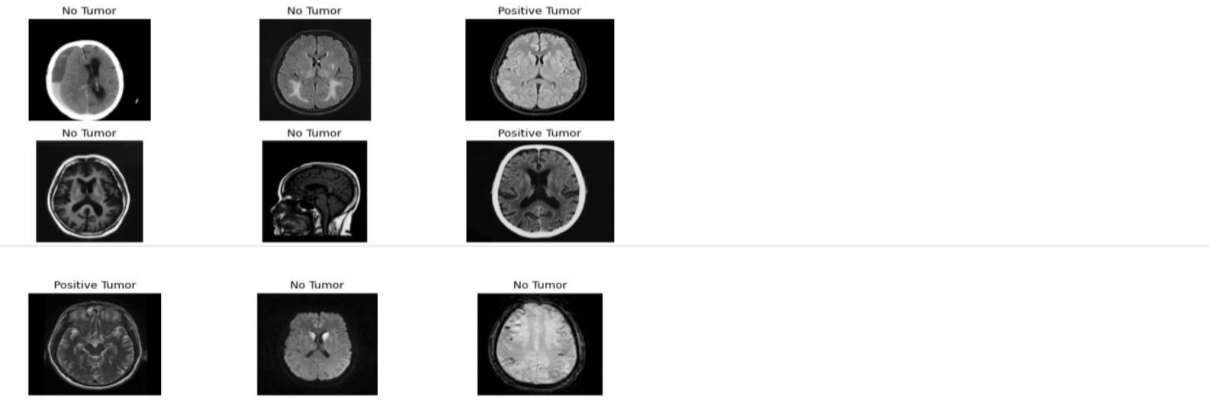


Fig.4.10 Testing the model for No tumor prediction.

Plotting Positive tumor prediction of support vector machine for first sixteen images in testing data are shown in below Fig.4.11.

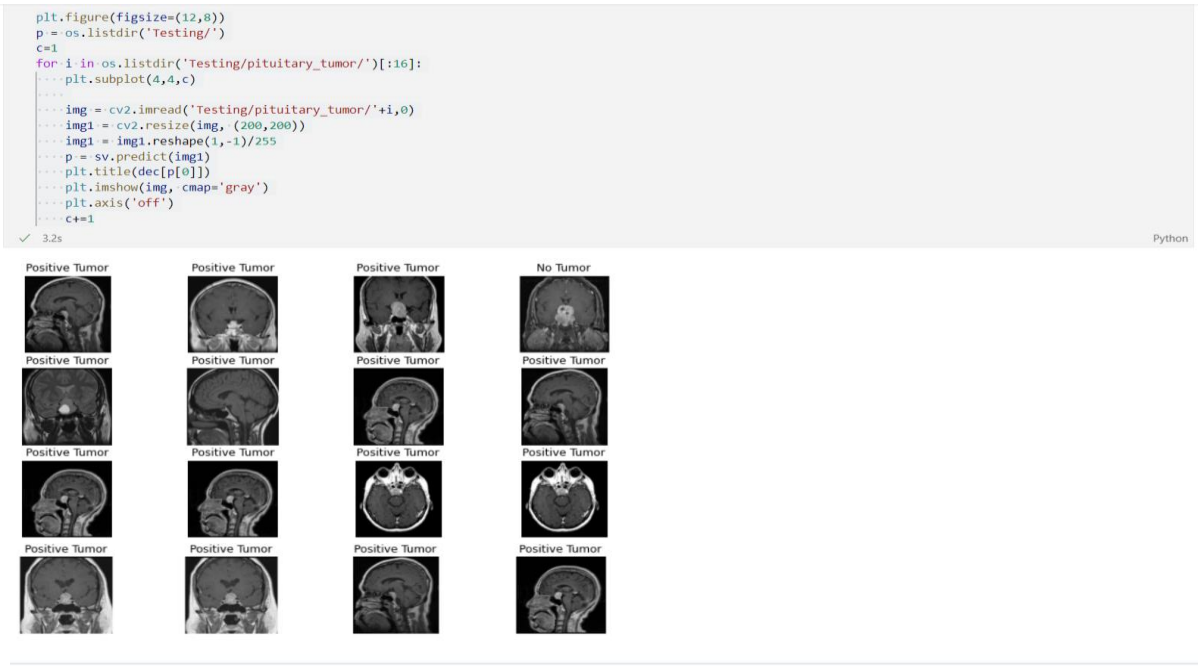


Fig.4.11 Testing the model for Yes (positive) tumor prediction.

Convolutional Neural Network (CNN)

- Computer Vision errands can now be performed with an assortment of cell phone applications. Computer Vision is a subset of Deep Learning that helps with learning a Computer vision to have something like a human.
- It's not easy to recreate our perception of the world with just a few lines of code. Objects and faces that pass through our vision are constantly recognized, segmented, and inferred. The human eye is a marvel within itself, subconsciously absorbing information. The study of human vision and perception through a variety of 'tasks,' including, but not limited to:
- Object recognition
- Image grouping
- Image remaking

- Face acknowledgment
- Semantic division.
- Image characterization is the most common way of partner a picture with a bunch of class names. An machine learning calculation is taken care of a bunch of pre-marked preparing information in this managed gaining issue from the preparation images, this calculation attempts to get familiar with the visual elements related with each name and groups unlabeled images likewise. It's a typical errand that the Keras Open-Source Deep Learning Library will explore today.
- The process of associating an image with a set of labels is based on image classification. It's an issue of supervised learning where a machine learning algorithm is fed pre-labeled training data. This algorithm learns the visual features associated with each label from the training images and then classifying unlabeled images accordingly. With the Keras Open-Source Deep Learning Library, look into a common assignment today.

Forward Propagation:

- A series of channels connect the Input, Hidden, and Output layers, allowing data to flow into a network within these buried layers of neurons, the values of the input data are altered.
- Review that each neuron in the association helps input through related redirects from all neurons in the past layer. This information is the weighted measure of the overall large number of burdens at all of these affiliations copied by the outcome vector of the past layer. This weighted total is given into an Activation Function, which creates the outcome for a specific neuron while moreover filling in as the commitment for the accompanying layer. The data is incited forward for each layer until it shows up at the Output layer, where the amount of layers is extended.

Back propagation:

- Once the Output layer is reached, the model's projected class is the neuron with the highest activation. The greatness of progress that needs to happen in the result layer to limit the misfortune is processed in light of the organization's result.
- Neural networks try to enhance the output node's value based on the correct class. Back propagation is used to accomplish this. Back engendering expands the worth of the right result hub by utilizing the subordinate (for example slopes) of the misfortune work

concerning each secret layer's loads. Slope plummet endeavors to lessen the general misfortune determined for the organization's forecasts.

Convolutional Neural Networks

- Recall how ordinary brain networks work. A progression of completely associated secret layers engenders the upsides of the info information forward. A brain organization's feedback layer is comprised of N hubs, where is the length of the information vector. To make a prediction, the Output layer performs additional computations and transforms the inputted data. The number of features in a network is a crucial factor to consider. A 32x32x3 pixel picture if smooth grid into a solitary information vector, you'll get a variety of N hubs with related loads.
- While this may be a sensible contribution for an ordinary organization, our natural product bowl picture probably won't be. Values would come about because of leveling the aspects. For image order, time and computational power basically don't lean toward this methodology.
- Convolutional Neural Networks (CNNs) have arisen as a potential arrangement. Almost every computer vision application uses a deep neural network of this type. Almost anyone with a smartphone these days can take high-resolution images.

Architecture:

CNN models have layers. The layers take a 3D volume as information, change it utilizing differential conditions, and result one more 3D volume for each situation. A few layers need hyper boundaries changed, while others don't. . In below Fig.4.12 how layers work in each steps are shown.

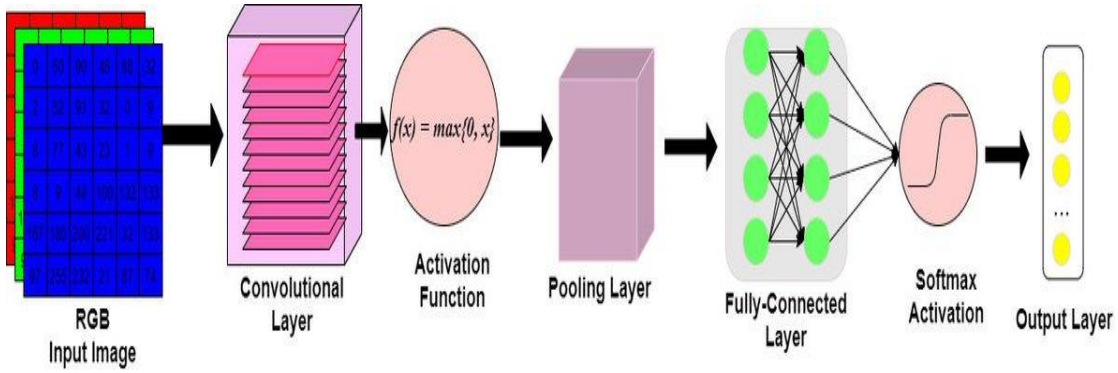


Fig.4.12 Architecture of Convolutional Neural Network.

0. Input Layer:

- A images crude pixel values are addressed by a 3D lattice. WxHxD, with profundity approaching the quantity of variety directs in the image.

1. Convolutional Layer:

- Conv. Layers will enroll the aftereffect of center points related with neighborhood region of the information organization.
- Spot items are determined utilizing a bunch of loads and values related with a neighborhood area of the info.

2. Activation Layer:

- The outcome volume of the Conv. layer is dealt with into a part by-part inception work, most normally a Rectified-Linear Unit (ReLu).
- The ReLu layer will close whether a data center point will 'fire' considering the data.
- Whether the convolution layer's channels have perceived a visual component is shown by this 'ending.'
- The most extreme (0, x) work is applied with an edge of 0.
- The parts of the volume don't change.

3. Pooling Layer:

- To reduce the output volume's width and height, a down-sampling strategy is used.

4. Fully-Connected Layer:

- The result volume, or convolved highlights, is gotten by a Fully-Connected Layer of hubs.

- Particularly like in standard mind associations, every center point in this layer is related with every center point in the volume of features being feed-forward.

- The class probabilities are figured and taken care of in a three-layered show of perspectives [1x1xK], where K is the amount of classes.

Model building for Deep learning

Deep learning models are fabricated utilizing brain organizations. A brain network takes in inputs, which are then dealt with stealthily layers using loads that are changed during getting ready. Then, the model lets out a gauge. The heaps are adjusted to find plans to further develop gauges. Deep learning model building steps are shown in below Fig.4.13.

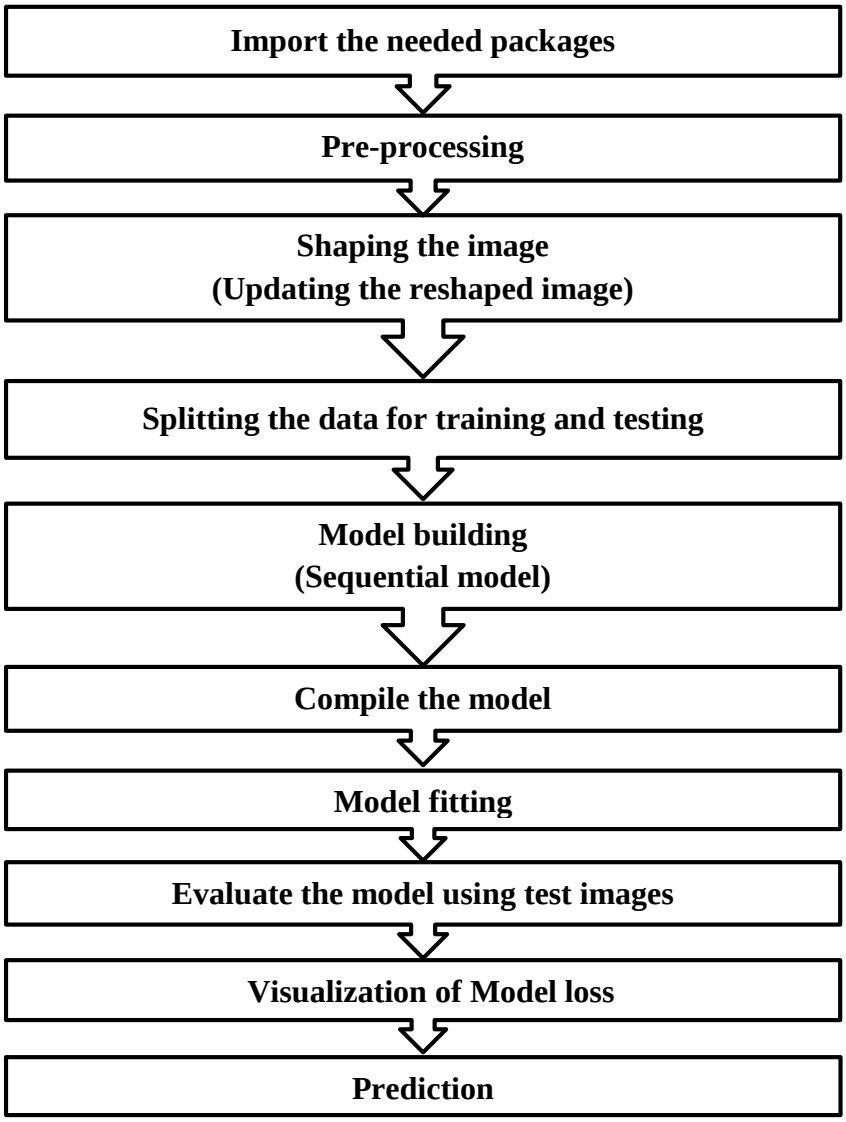


Fig.4.13 Brain tumor detection model using Deep learning techniques.

Import the libraries os, keras, PIL, numpy, pandas, and sklearn. The process of converting categorical data variables into machine code, which improves prediction performance and accuracy rate, is one example of hot encoding are shown in below Fig.4.14. Setting directories for respective folder each classes and classes containing a tumor, no tumor, pituitary tumor, and malignant tumor. Updating results list for images without tumor are shown in below Fig.4.15.

```
import os
import keras
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, BatchNormalization
from PIL import Image
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('dark_background')
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
```

✓ 0.4s Python

```
encoder = OneHotEncoder()
encoder.fit([[0], [1]])
```

✓ 0.6s Python

```
OneHotEncoder()
```

```
data = []
paths = []
result = []

for r, d, f in os.walk(r'Training/pituitary_tumor'):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))
```

Fig.4.14 Import the libraries, Pre-processing and updating results list for images with tumor.

```
paths = []
result = []

for r, d, f in os.walk(r'Training/pituitary_tumor'):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))

for path in paths:
    img = Image.open(path)
    img = img.resize((128,128))
    img = np.array(img)
    if(img.shape == (128,128,3)):
        data.append(np.array(img))
        result.append(encoder.transform([[0]]).toarray())
```

✓ 7.5s Python

```
# This cell updates result list for images without tumor

paths = []
for r, d, f in os.walk(r"Training/no_tumor"):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))

for path in paths:
    img = Image.open(path)
    img = img.resize((128,128))
    img = np.array(img)
    if(img.shape == (128,128,3)):
        data.append(np.array(img))
        result.append(encoder.transform([[1]]).toarray())
```

✓ 2.9s Python

Fig.4.15 Updating results list for images without tumor.

Divide the x dataset into two parts: x train and x test. Similarly, divide the y dataset into two parts: y train and y test. A Sequential model has one hidden layer for the model, a convolutional input layer followed by a Maxpooling layer, a hidden layer with another convolutional layer and flatten the outputs to dimension reduction, and create an output layer with a dense relu and softmax layer are shown in below Fig.4.16. Utilize an Adamax advancement strategy to streamline the model and a misfortune capacity to quantify the misfortune. The name and sort of each layer, the result shape for each layer, and the quantity of weight boundaries for each layer, and the absolute number of teachable and non-teachable boundaries are even remembered for the model's outline are shown in below Fig.4.17.

```
data = np.array(data)
data.shape
```
✓ 0.1s Python

```
(1222, 128, 128, 3)
```

```
result = np.array(result)
result = result.reshape(1222,2)
```
✓ 0.4s Python

```
x_train,x_test,y_train,y_test = train_test_split(data, result, test_size=0.2, shuffle=True, random_state=0)
```
✓ 0.1s Python

```
model = Sequential()

model.add(Conv2D(32, kernel_size=(2, 2), input_shape=(128, 128, 3), padding = 'Same'))
model.add(Conv2D(32, kernel_size=(2, 2), activation ='relu', padding = 'Same'))


model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, kernel_size = (2,2), activation ='relu', padding = 'Same'))
model.add(Conv2D(64, kernel_size = (2,2), activation ='relu', padding = 'Same'))

model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())
```

Fig.4.16 Splitting the data for train, test and Model building (Sequential model).

```
model.add(Flatten())

model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))

model.compile(loss = "categorical_crossentropy", optimizer='Adamax')
print(model.summary())
```
✓ 4.0s Python

```
Output exceeds the size limit. Open the full output data in a text editor
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 128, 128, 32)      416

 conv2d_1 (Conv2D)           (None, 128, 128, 32)      4128

 batch_normalization (BatchN  (None, 128, 128, 32)     128
 ormalization)

 max_pooling2d (MaxPooling2D  (None, 64, 64, 32)       0
 )

 dropout (Dropout)           (None, 64, 64, 32)        0

 conv2d_2 (Conv2D)           (None, 64, 64, 64)        8256

 conv2d_3 (Conv2D)           (None, 64, 64, 64)        16448

 batch normalization 1 (Batc  (None, 64, 64, 64)       256
```

Fig.4.17 Compile the model.

The history is returned by fit () function, which is used to train the model. The model will then be fitted to training data, trained in batches of 40 with 30 epochs, and 20% of the data will be used for validation are shown in below Fig.4.18. Each epoch is measured for training loss and validation loss are shown in below Fig.4.19.

```
conv2d_3 (Conv2D)           (None, 64, 64, 64)        16448

batch_normalization_1 (Batc  (None, 64, 64, 64)       256
hNormalization)

max_pooling2d_1 (MaxPooling  (None, 32, 32, 64)       0
2D)
...
Total params: 33,585,602
Trainable params: 33,585,410
Non-trainable params: 192
_________________________________________________________________
None
```

```
y_train.shape
```
✓ 0.7s Python

```
(977, 2)
```

```
history = model.fit(x_train, y_train, epochs = 30, batch_size = 40, verbose = 1,validation_data = (x_test, y_test))
```
✓ 23m 8.2s Python

```
Output exceeds the size limit. Open the full output data in a text editor
Epoch 1/30
25/25 [==============================] - 48s 2s/step - loss: 10.8488 - val_loss: 56.9921
Epoch 2/30
25/25 [==============================] - 45s 2s/step - loss: 0.7210 - val_loss: 24.3927
Epoch 3/30
25/25 [==============================] - 45s 2s/step - loss: 0.1922 - val_loss: 8.8624
Epoch 4/30
```

Fig.4.18 Model fitting.

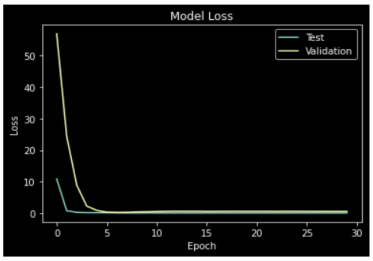
```
Epoch 1/30
25/25 [==============================] - 48s 2s/step - loss: 10.8488 - val_loss: 56.9921
Epoch 2/30
25/25 [==============================] - 45s 2s/step - loss: 0.7210 - val_loss: 24.3927
Epoch 3/30
25/25 [==============================] - 45s 2s/step - loss: 0.1922 - val_loss: 8.8624
Epoch 4/30
25/25 [==============================] - 45s 2s/step - loss: 0.1069 - val_loss: 2.2004
Epoch 5/30
25/25 [==============================] - 45s 2s/step - loss: 0.0876 - val_loss: 0.8508
Epoch 6/30
25/25 [==============================] - 46s 2s/step - loss: 0.0808 - val_loss: 0.2699
Epoch 7/30
25/25 [==============================] - 47s 2s/step - loss: 0.0332 - val_loss: 0.1854
Epoch 8/30
25/25 [==============================] - 46s 2s/step - loss: 0.0139 - val_loss: 0.2292
Epoch 9/30
25/25 [==============================] - 46s 2s/step - loss: 0.0086 - val_loss: 0.3448
Epoch 10/30
25/25 [==============================] - 46s 2s/step - loss: 0.0339 - val_loss: 0.3969
Epoch 11/30
25/25 [==============================] - 45s 2s/step - loss: 0.0092 - val_loss: 0.4625
Epoch 12/30
25/25 [==============================] - 47s 2s/step - loss: 0.0067 - val_loss: 0.5262
Epoch 13/30
...
25/25 [==============================] - 48s 2s/step - loss: 0.0033 - val_loss: 0.5053
Epoch 29/30
25/25 [==============================] - 50s 2s/step - loss: 0.0053 - val_loss: 0.4957
Epoch 30/30
25/25 [==============================] - 47s 2s/step - loss: 0.0011 - val_loss: 0.5006
```

Fig.4.19 Training loss and validation loss.

To understand the variation in accuracy, plot the accuracy of the training and validation sets for each epoch. Plot the loss and validation loss at the same time are shown in below Fig.4.20. The classification model is build for various classes the above predicted output as no tumor 100% accurately are shown in below Fig.4.21.

```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Test', 'Validation'], loc='upper right')
plt.show()
```

✓ 1.2s Python




```
def names(number):
    if number==0:
        return 'Its a Tumor'
    else:
        return 'No, Its not a tumor'
```

✓ 0.7s Python

Fig.4.20 Visualization of Model loss.

```
from matplotlib.pyplot import imshow
img = Image.open(r"Training/no_tumor/1.jpg")
x = np.array(img.resize((128,128)))
x = x.reshape(1,128,128,3)
res = model.predict_on_batch(x)
classification = np.where(res == np.amax(res))[1][0]
imshow(img)
print(str(res[0][classification]*100) + '% Confidence This Is ' + names(classification))
```

✓ 1.8s Python

```
100.0% Confidence This Is No, Its not a tumor
```

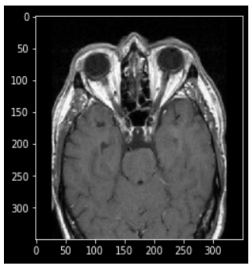


Fig.4.21 Evaluate the model using test image for No tumor.

The classification model is build for various classes the above predicted output it's a tumor 100% accurately are shown in below Fig.4.22. The classification model is build using keras for prediction, the outcome of tested image is a tumor 100% accurately are shown in below Fig.4.23.

```
from matplotlib.pyplot import imshow
img = Image.open(r"Training/pituitary_tumor/p (1).jpg")
x = np.array(img.resize((128,128)))
x = x.reshape(1,128,128,3)
res = model.predict_on_batch(x)
classification = np.where(res == np.amax(res))[1][0]
imshow(img)
print(str(res[0][classification]*100) + '% Confidence This Is A ' + names(classification))
```

✓ 0.3s Python

```
100.0% Confidence This Is A Its a Tumor
```

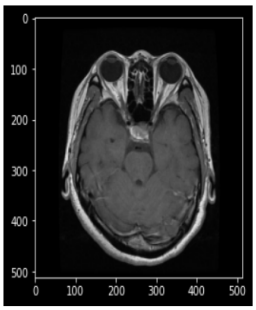


Fig.4.22 Evaluate the model using test image for a tumor.

```
from matplotlib.pyplot import imshow
img = Image.open(r"Training/pituitary_tumor/p (1).jpg")
x = np.array(img.resize((128,128)))
x = x.reshape(1,128,128,3)
res = loaded_model.predict_on_batch(x)
classification = np.where(res == np.amax(res))[1][0]
imshow(img)
print(str(res[0][classification]*100) + '% Confidence This Is A ' + names(classification))
```

✓ 0.5s Python

```
100.0% Confidence This Is A Its a Tumor
```

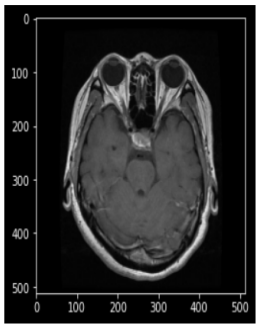


Fig.4.23 Evaluate the model using keras for a tumor prediction.

4.3 Performance evaluation:

The system has already been trained to spot tumor cells during a patient's MRI and thus to predict whether or not the patient seems to possess a tumor.

SVM is a somewhat new machine learning calculation for order, though LR is an old standard measurable grouping technique. While contrasting help vector machine and calculated relapse as far as testing, support vector machine beats strategic relapse in this review are shown in below Fig.4.24. This work is finished by utilizing CNN, CNN execution is determined by involving both misfortune and approval of misfortune for exactness. When contrasted with Logistic Regression and Support Vector Machines, CNN gives the most elevated level of precision. Thus, CNN is utilized to convey a flask framework for prediction are shown in below Fig.4.25.

Fig.4.24 Performance evaluation for machine learning.

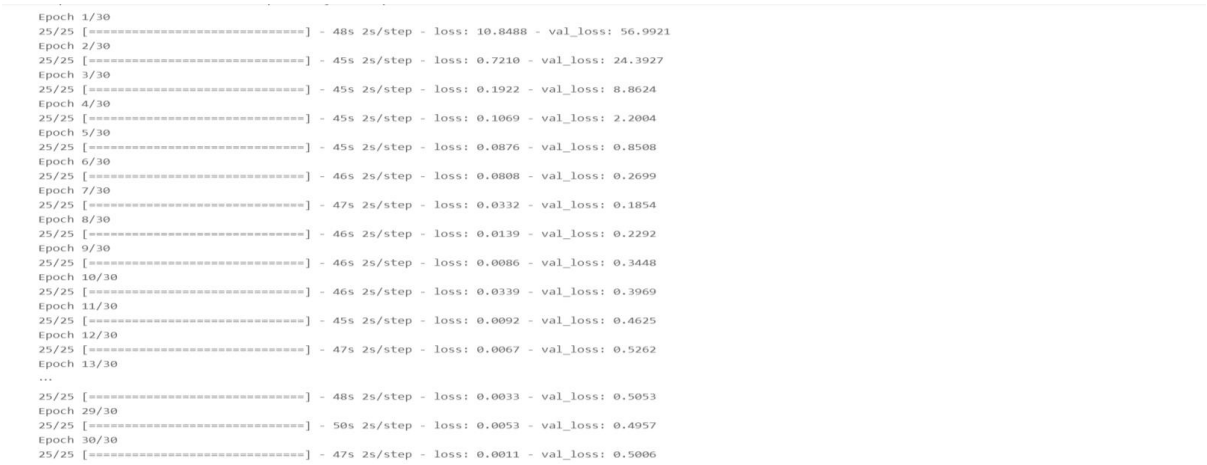


Fig.4.25 Performance evaluation for deep learning.

RESULTS AND DISCUSSION

5. RESULTS AND DISCUSSION

The results of machine learning models and deep learning model have been evaluated and the accuracy is shown in table.5.1.

Table.5.1 Accuracy of machine learning models and deep learning models.

ALGORITHMS	ACCURACY
Logistic Regression	92%
Support Vector Machine	93%
Convolutional Neural Network	99.99%

From the above table its evident that CNN gives the highest accuracy level, when compared with Logistic Regression and Support Vector Machines, so CNN is used for deploying a flask system for prediction.

Home page for brain tumor prediction through MRI images using CNN in keras as shown in below Fig.5.1. Select a file for brain tumor prediction are shown in below Fig.5.2.

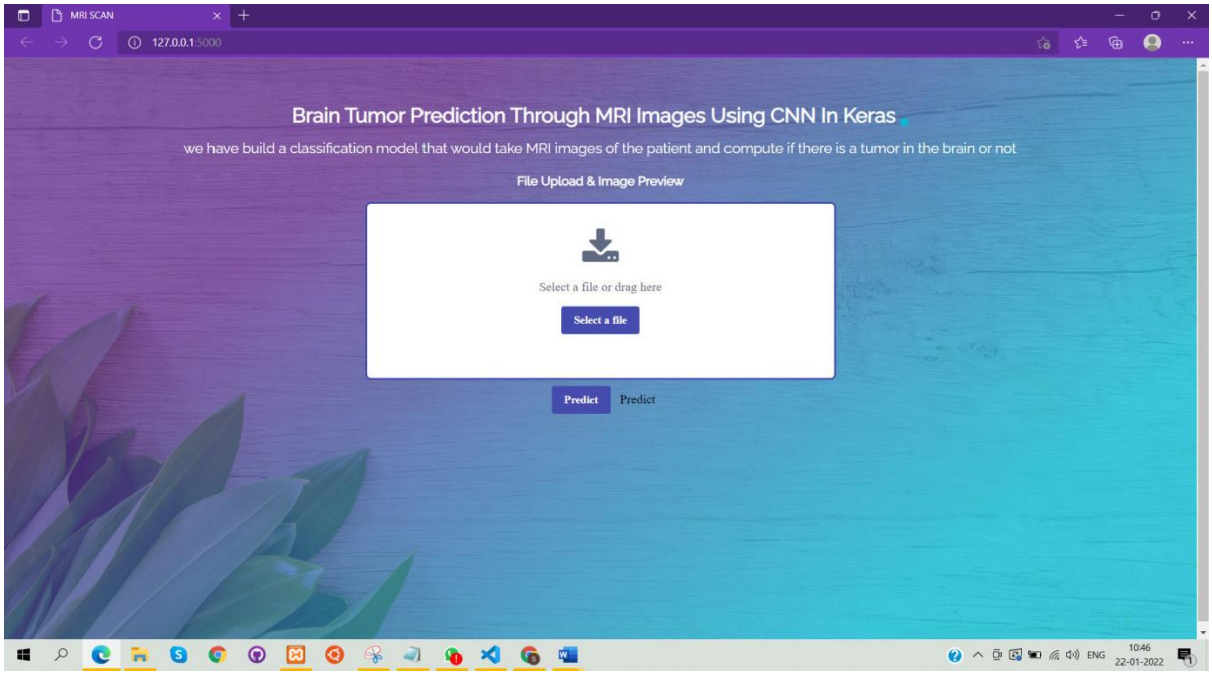


Fig.5.1 Home page.

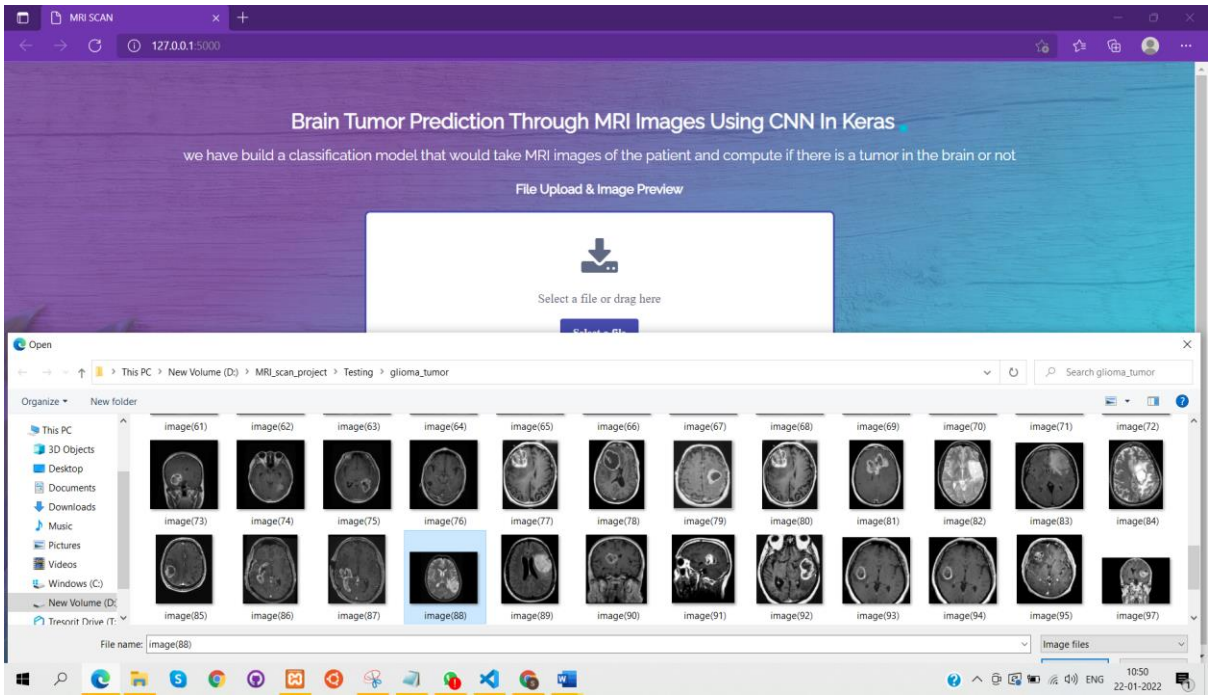


Fig.5.2 Select a file.

Selected a file for brain tumor prediction is uploaded are shown in below Fig.5.3. Uploaded file is previewed for brain tumor prediction. From the uploaded MRI image, no tumor can be predicted with 100% accuracy are shown in below Fig.5.4.

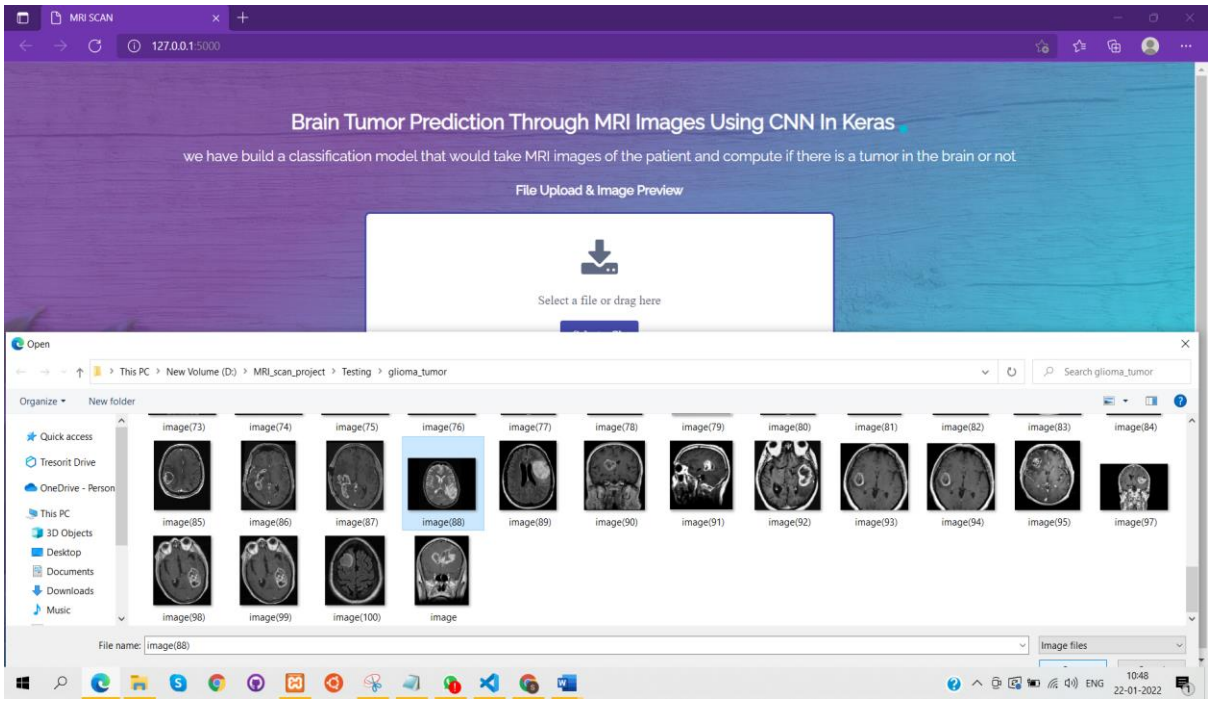


Fig.5.3 File uploaded.

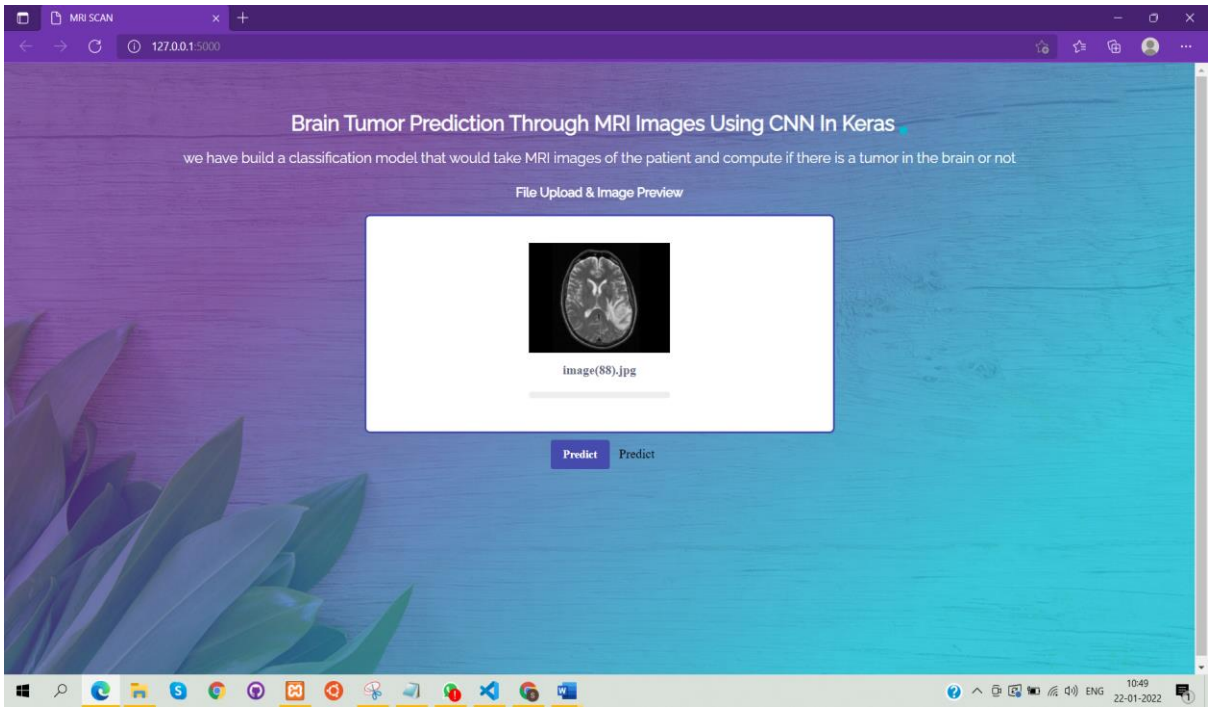


Fig.5.4 Uploaded file is previewed for the prediction.

From the uploaded MRI image, no tumor can be predicted with 100% accuracy are shown in below Fig.5.5. Using the various class folders, choose a file for prediction are shown in below Fig.5.6.

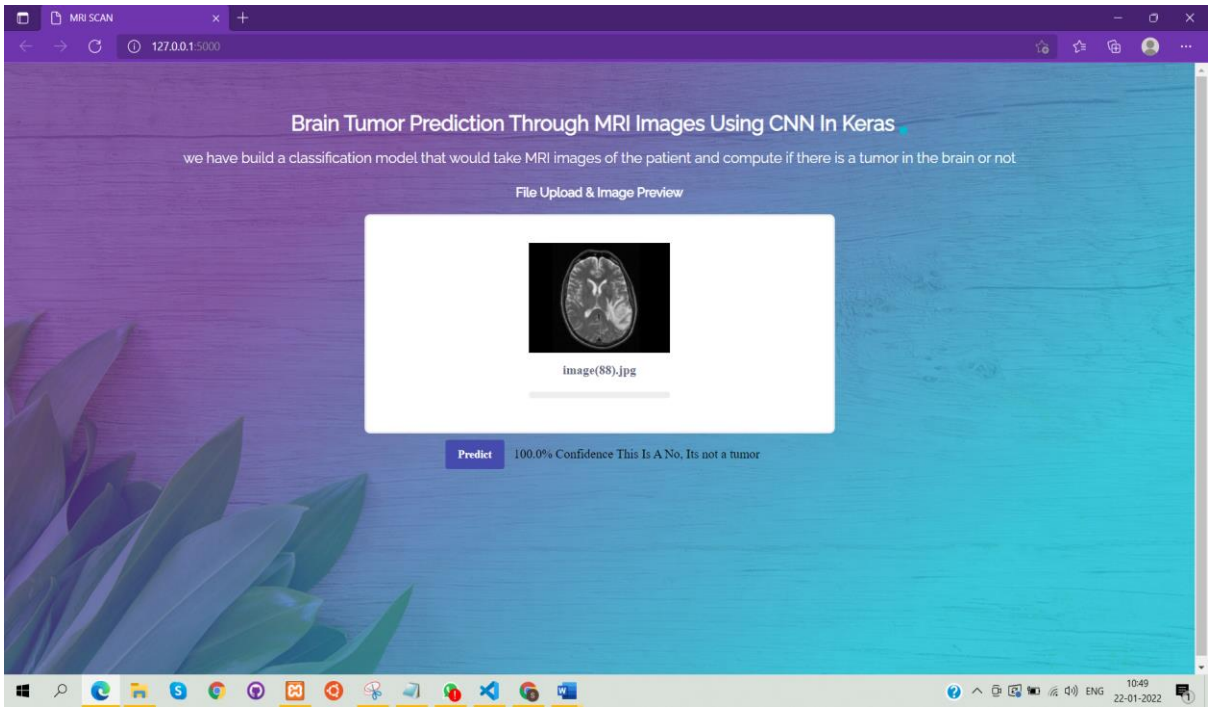


Fig.5.5 No tumor is predicted with 100% accuracy.

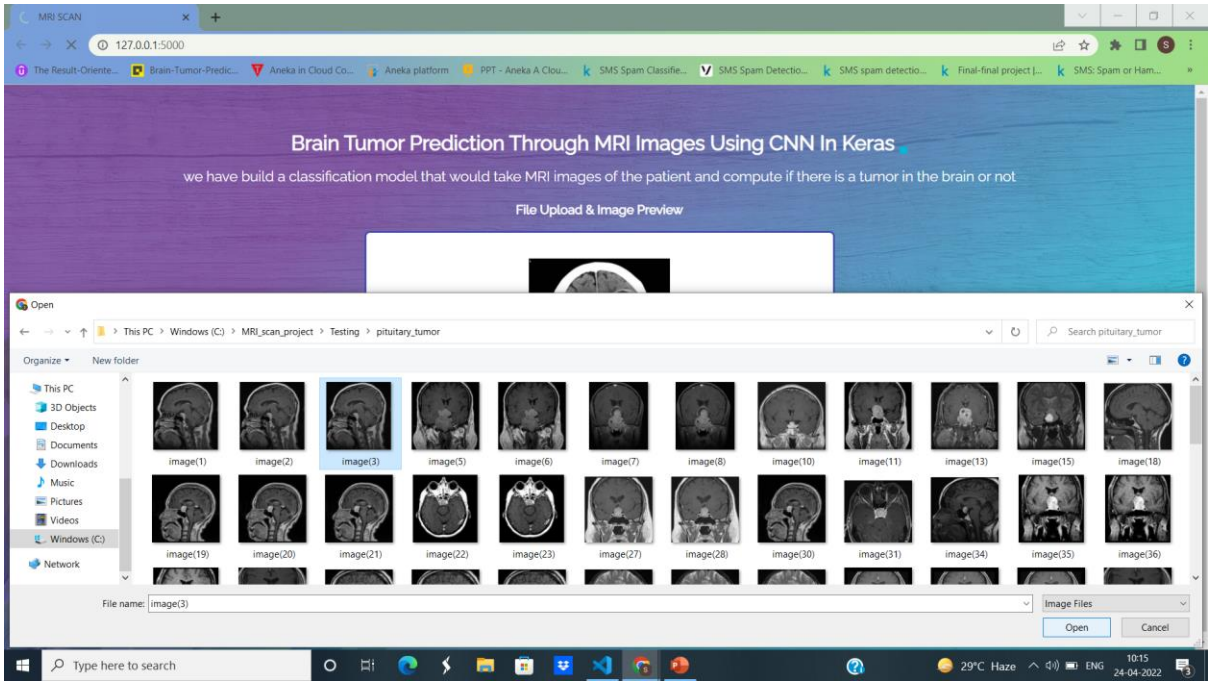


Fig.5.6 Choose a file for prediction again using the various class folders.

A file prediction for brain tumor is uploaded after a file is selected are shown in below Fig.5.7. The uploaded file is being previewed for prediction are shown in below Fig.5.8.

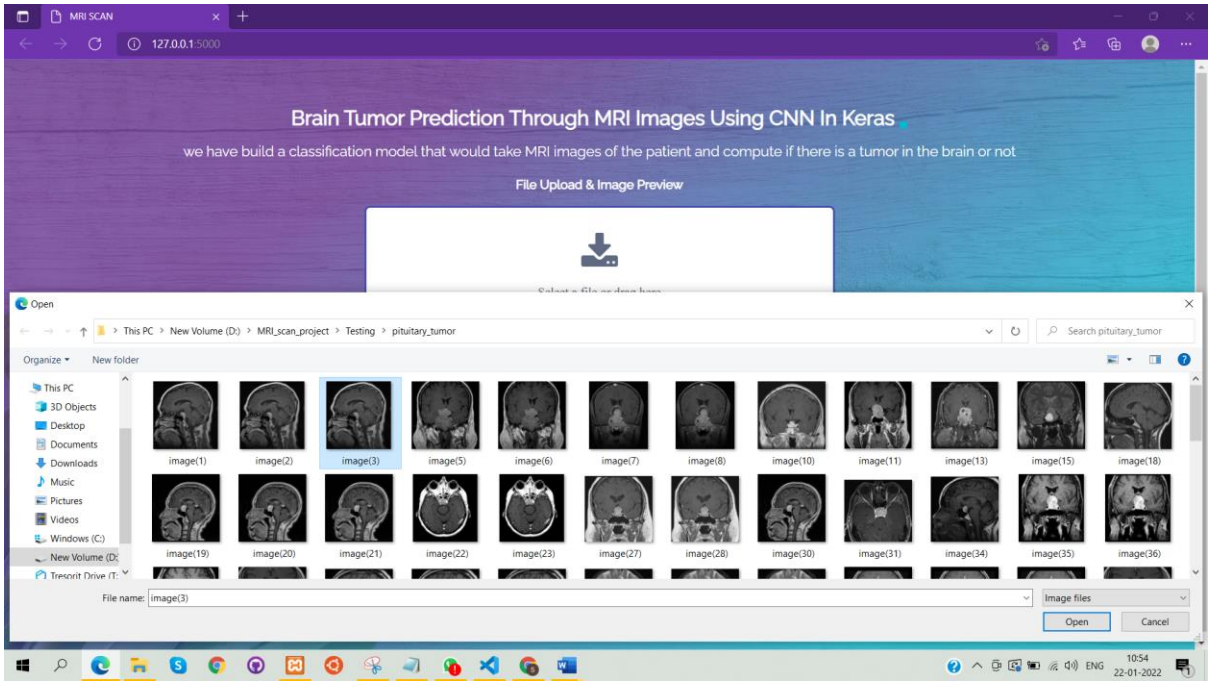


Fig.5.7 File is selected for prediction.

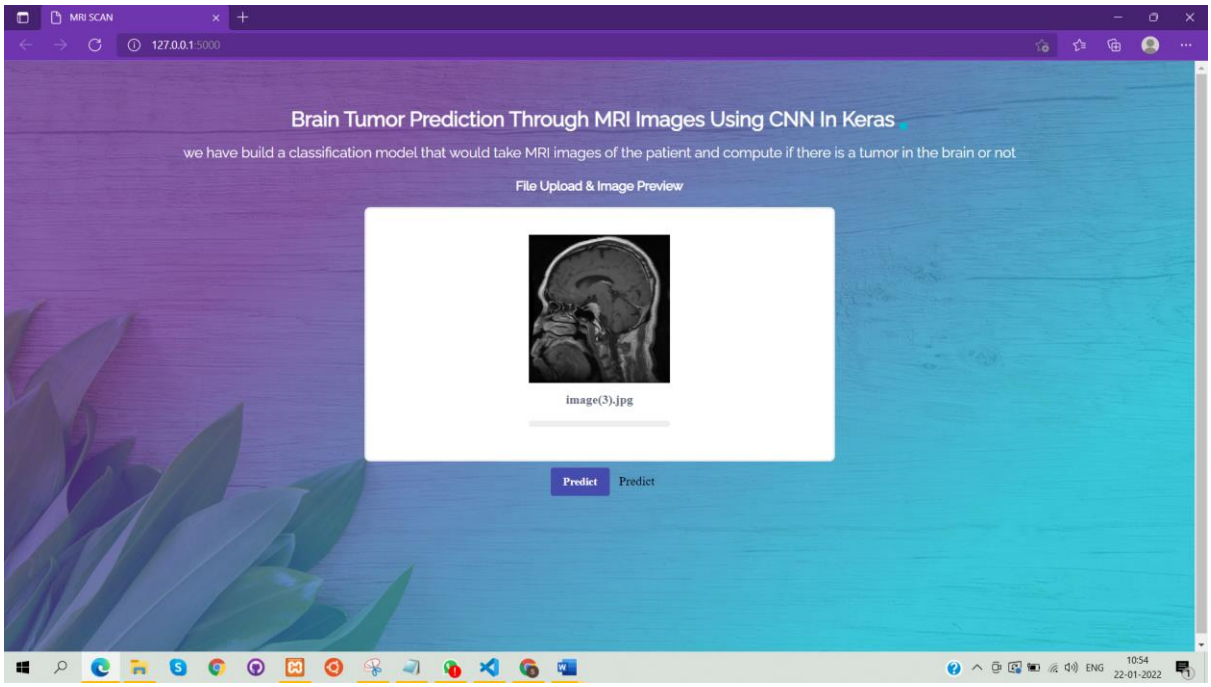


Fig.5.8 Image previewed for uploaded file.

Yes tumor is predicted 100% accurately from the uploaded MRI image prediction are shown in below Fig.5.9.

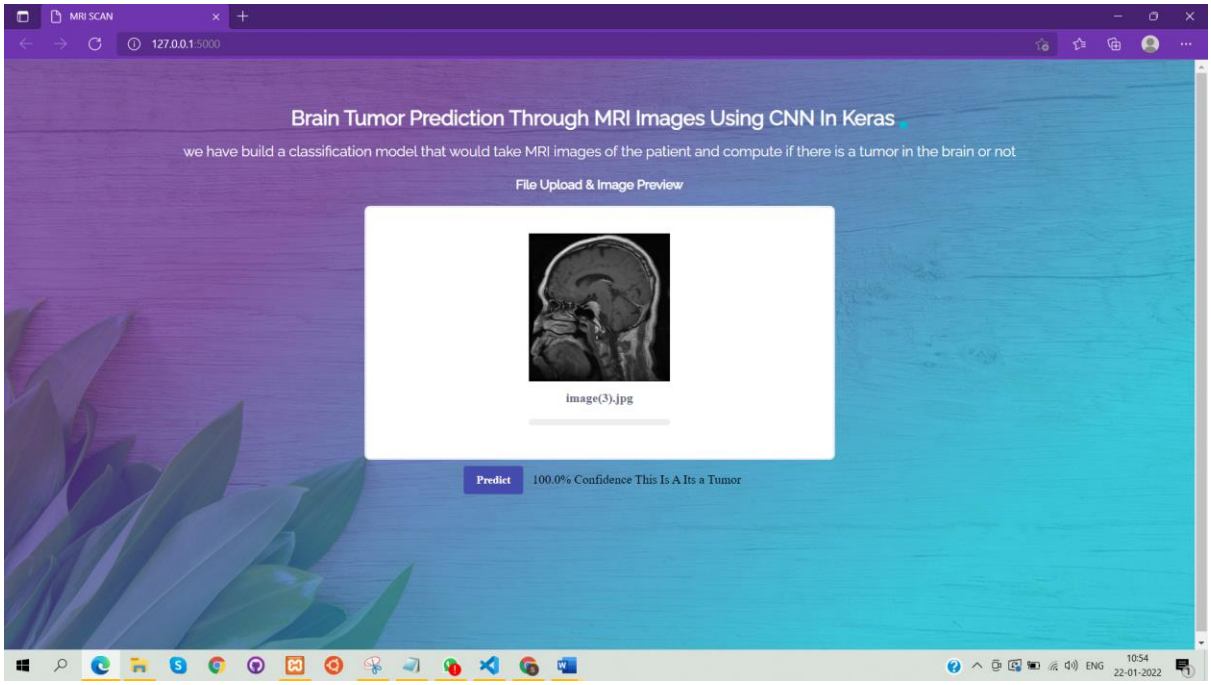


Fig.5.9 Yes tumor is predicted 100% accurately.

CONCLUSION

6. CONCLUSION

Medical procedure can be used to find out tumors in the brain, yet chemo and radiotherapy can delay the life expectancy of dangerous tumors after medical procedure. Early location and treatment are fundamental for mind tumor recognizable proof, proposed action, Machine learning, for example, Support Vector Machine (SVM), Logistic Regression (LR) and Advanced Reading as CNN are utilized for MRI imaging to go through the above technique. Likewise, Logistic Regression and Support Vector Machine requires more PC time and memory than CNN however, but creates satisfactory outcomes. Since the clinical field creates and safeguards a lot of information, top to bottom review will assume a significant part in information examination.

BIBLIOGRAPHY

7. BIBLIOGRAPHY

REFERENCES

[1]. Abiwinanda N, Hanif M, Hesaputra ST, Handayani A, Mengko TR [2019] Convolutional neural network for brain tumour classification. 183–189 in IFMBE Proc.

[2]. Ayadi W, Elhamzi W, Charfi I, and Atri M [2021] Deep CNN is used to classify brain tumours. 53(1):671–700 in Neural Process Lett.

[3]. Badža MM and MC Barjaktarovi [2020] A convolutional neural network was used to classify brain tumours from MRI images. Appl Sci 10(6):1–13.

[4]. Cinar and Yildirim M [2020] The hybrid convolutional neural network architecture was used to detect tumours on brain MRI images. 109684, Medical Hypotheses.

[5]. Deepak S, Ameer P [2019]. Brain tumour classification via transfer learning using deep CNN features. Comput Biol Med 111:103345.

[6]. Ertosun MG and Rubin DL [2015] Automated grading of gliomas in digital pathology images using deep learning: a modular approach using an ensemble of convolutional neural networks 1899–1908 in Annu Symp Proc AMIA Symp 2015.

[7]. Kabir Anaraki A, Ayati M, and Kazemi F [2019] Convolutional neural networks and genetic algorithms are used to classify and grade brain tumours based on magnetic resonance imaging. 39(1):63–74 in Biocybern Biomed Eng.

[8]. Khan HA, Jue W, Mushtaq M, Mushtaq MU [2020] Convolutional neural networks are used to classify brain tumours in MRI images. 17(5):6203–6216 in Math Biosci Eng.

[9]. Khawaldeh S, Pervaiz U, Rafiq A, Alkhawaldeh R S [2017] Noninvasive glioma tumour grading using convolutional neural networks and magnetic resonance imaging. Appl Sci 8(1):1–17.

[10]. Mehrotra R, Ansari MA, Agrawal R, and Anand RS [2020] A Transfer Learning Approach for AI-based Brain Tumor Classification. 2(9):1–12.

[11]. Mohsen H, El-Dahshan ESA, El-Horbaty ESM, Salem ABM [2018] Brain tumour classification using deep learning neural networks 68–71 in Future Comput Informat J.

[12]. Mzoughi H, Njeh I, Wali A, Slima M, A. Ben BenHamida, C. Mhiri, K. Mahfoudhe [2020] A deep multi-scale 3D convolutional neural network (CNN) was used to classify MRI gliomas brain tumours. 903–915.

[13]. Pereira S, R Meier, V Alves, M Reyes, CA Silva [2018] Automatic brain tumour grading and quality assessment from MRI data using convolutional neural networks. Machine learning in medical image computing applications: understanding and interpretation 106–11 in Springer, Cham.

[14]. Rehman A, Naz S, Razzak M, Akram F, and Imran M [2020] A transfer learning-based Deep Learning framework for automatic brain tumour classification. 757–775 in Circuits, Systems, and Signal Processing.

[15]. Talo M, Baloglu UB, Yldrm Y, Rajendra Acharya U [2019] Deep transfer learning for automated brain abnormality classification using MR images. 176–188 in Cogn Syst Res 54(12).

[16]. Yang Y, Yan LF, Zhang X, Han Y, Nan HY, Hu YC, Hu B, Yan SL, Zhang J, Cheng DL, Ge XW [16]. Yang Y, Yan LF, Zhang X, Han Y, Nan HY, Hu (2018) A deep learning study with transfer learning on glioma grading on conventional MR images. 804 in Front Neurosci.

SOURCE CODE

8. SOURCE CODE

HTML CODING

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>MRI SCAN </title>

  <link rel="stylesheet" href="{{ url_for('static', filename='css/mriscan.css') }}">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">

  </script>

</head>

<body>

  <h2>Brain Tumor Prediction Through MRI Images Using CNN In Keras <span class="dot"></span> </h2>

  <p class="lead">we have build a classification model that would take MRI images of the patient and compute if there is a tumor in the brain or not

    <br><br><b>File Upload & Image Preview</b></p>

  <!-- Upload  -->

  <form id="file-upload-form" class="uploader">

    <input id="file-upload" type="file" name="fileUpload" accept="image/*" />
```

```
    <label for="file-upload" id="file-drag">
      <img id="file-image" src="#" alt="Preview" class="hidden">
      <div id="start">
        <i class="fa fa-download" aria-hidden="true"></i>
        <div>Select a file or drag here</div>
        <div id="notimage" class="hidden">Please select an image</div>
        <span id="file-upload-btn" class="btn btn-primary">Select a file</span>
      </div>
      <div id="response" class="hidden">
        <div id="messages"></div>
        <progress class="progress" id="file-progress" value="0">
          <span>0</span><span>%</span>
        </progress>
      </div>
    </label>
    <span id="predict-btn" class="btn btn-primary">Predict</span>
    <span id="Result" >Predict</span>
  </form>
  <script src="{{ url_for('static', filename='js/mriscan.js') }}"></script>
</body>
</html>
```

APP PY CODING

```
import json

# from models.TSV import TSV

from flask import Flask, render_template, url_for, redirect, jsonify, request

import os

import base64

import urllib.request

import numpy as np

import keras

from keras.models import Sequential

from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, BatchNormalization

from PIL import Image

from keras.models import load_model

loaded_model = load_model("network.h5")

app = Flask(__name__, static_url_path='/static')

# gene_panel_json = {}

# gene_json = {}

# def creatFlatFile(filename='Gene_Panel_v10.2.tsv'):

#     global gene_panel_json, gene_json

#     obj = TSV(filename)

#     del obj
```

```
#     gene_panel_file = open('flat_data/gene_panel.json', 'r')
#     gene_file = open('flat_data/gene.json', 'r')
#     gene_panel_json = json.loads(gene_panel_file.read())
#     gene_json = json.loads(gene_file.read())
#     gene_panel_file.close()
#     gene_file.close()
# @app.route("/gene_panel")
# def gene_panel():
#     res = list(gene_panel_json.keys())
#     return jsonify(res)
# @app.route("/gene_panel_list", methods=['POST'])
# def gene_panel_table():
#     query = json.loads(request.data)
#     res = {}
#     for key in query['data']:
#         key = key.replace('©', '&copy;')
#         res[key] = gene_panel_json[key]
#     return jsonify(res)
# @app.route("/get_genes")
# def get_genes():
#     res = list(gene_json.keys())
```

```
#     return jsonify(res)
def names(number):
    if number==0:
        return 'Its a Tumor'
    else:
        return 'No, Its not a tumor'
@app.route("/predicted", methods=['POST'])
def predicted():
    files = request.files.getlist('files[]')
    query = files[0]
    res = {}
    img = Image.open(query)
    # img = Image.open(query)
    # image = base64.b64encode(query).decode("utf-8")
    print(img)
    x = np.array(img.resize((128,128)))
    x = x.reshape(1,128,128,3)
    res = loaded_model.predict_on_batch(x)
    classification = np.where(res == np.amax(res))[1][0]
    result=str(res[0][classification]*100) + '% Confidence This Is A ' +
names(classification)
    # for gene in query['data']:
```

```
    #     gene_panel = gene_json[gene]
    #     for value in gene_panel:
    #         panel = value[0]
    #         status = value[1]
    #         res[panel] = []
    #         for i in gene_panel_json[panel]:
    #             if i[0] == status:
    #                 res[panel].append(i)
    return jsonify(result)
@app.route("/")
def home():
    return render_template("index.html")
if __name__ == '__main__':
    # creatFlatFile()
    app.run(debug=True)
```

MRI CODING

```
import os
import keras
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, BatchNormalization
```

```
from PIL import Image
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import os
path = os.listdir('Training')
classes = {'no_tumor':0, 'pituitary_tumor':1,'glioma_tumor':1,'meningioma_tumor':1}
import cv2
X = []
Y = []
for cls in classes:
    pth = 'Training/'+cls
    for j in os.listdir(pth):
        img = cv2.imread(pth+'/'+j, 0)
        img = cv2.resize(img, (200,200))
        X.append(img)
        Y.append(classes[cls])
X = np.array(X)
Y = np.array(Y)
```

```
np.unique(Y)

pd.Series(Y).value_counts()

X.shape

plt.imshow(X[0], cmap='gray')

X_updated = X.reshape(len(X), -1)

X_updated.shape

xtrain, xtest, ytrain, ytest = train_test_split(X_updated, Y, random_state=10,test_size=.20)

xtrain.shape, xtest.shape

print(xtrain.max(), xtrain.min())

print(xtest.max(), xtest.min())

xtrain = xtrain/255

xtest = xtest/255

print(xtrain.max(), xtrain.min())

print(xtest.max(), xtest.min())

from sklearn.decomposition import PCA

print(xtrain.shape, xtest.shape)

pca = PCA(.98)

# pca_train = pca.fit_transform(xtrain)

# pca_test = pca.transform(xtest)

pca_train = xtrain

pca_test = xtest
```

```
from sklearn.linear_model import LogisticRegression

from sklearn.svm import SVC

import warnings

warnings.filterwarnings('ignore')

lg = LogisticRegression(C=0.1)

lg.fit(pca_train, ytrain)

sv = SVC()

sv.fit(pca_train, ytrain)

print("Training Score:", lg.score(pca_train, ytrain))

print("Testing Score:", lg.score(pca_test, ytest))

print("Training Score:", sv.score(pca_train, ytrain))

print("Testing Score:", sv.score(pca_test, ytest))

pred = sv.predict(pca_test)

np.where(ytest!=pred)

pred = lg.predict(pca_test)

np.where(ytest!=pred)

pred[10]

ytest[10]

dec = {0:'No Tumor', 1:'Positive Tumor',2:'glioma_tumor',3:'meningioma_tumor'}

plt.figure(figsize=(12,8))

p = os.listdir('Testing')
```

```
c=1
for i in os.listdir('Testing/no_tumor/')[:9]:
    plt.subplot(3,3,c)
    img = cv2.imread('Testing/no_tumor/'+i,0)
    img1 = cv2.resize(img, (200,200))
    img1 = img1.reshape(1,-1)/255
    p = sv.predict(img1)
    plt.title(dec[p[0]])
    plt.imshow(img, cmap='gray')
    plt.axis('off')
    c+=1
plt.figure(figsize=(12,8))
p = os.listdir('Testing/')
c=1
for i in os.listdir('Testing/pituitary_tumor/')[:16]:
    plt.subplot(4,4,c)
    img = cv2.imread('Testing/pituitary_tumor/'+i,0)
    img1 = cv2.resize(img, (200,200))
    img1 = img1.reshape(1,-1)/255
    p = sv.predict(img1)
    plt.title(dec[p[0]])
```

```
    plt.imshow(img, cmap='gray')
    plt.axis('off')
    c+=1
```

CNN CODING

```
import os
import keras
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, BatchNormalization
from PIL import Image
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('dark_background')
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
encoder = OneHotEncoder()
encoder.fit([[0], [1]])
data = []
paths = []
result = []
```

```
for r, d, f in os.walk(r'Training/pituitary_tumor'):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))
for path in paths:
    img = Image.open(path)
    img = img.resize((128,128))
    img = np.array(img)
    if(img.shape == (128,128,3)):
        data.append(np.array(img))
        result.append(encoder.transform([[0]]).toarray())
# This cell updates result list for images without tumor
paths = []
for r, d, f in os.walk(r"Training/no_tumor"):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))
for path in paths:
    img = Image.open(path)
    img = img.resize((128,128))
    img = np.array(img)
```

```
    if(img.shape == (128,128,3)):
        data.append(np.array(img))
        result.append(encoder.transform([[1]]).toarray())
data = np.array(data)
data.shape
result = np.array(result)
result = result.reshape(1222,2)
x_train,x_test,y_train,y_test = train_test_split(data, result, test_size=0.2, shuffle=True, random_state=0)
model = Sequential()
model.add(Conv2D(32, kernel_size=(2, 2), input_shape=(128, 128, 3), padding = 'Same'))
model.add(Conv2D(32, kernel_size=(2, 2),  activation ='relu', padding = 'Same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, kernel_size = (2,2), activation ='relu', padding = 'Same'))
model.add(Conv2D(64, kernel_size = (2,2), activation ='relu', padding = 'Same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2)))
model.add(Dropout(0.25))
model.add(Flatten())
```

```
model.add(Dense(512, activation='relu'))

model.add(Dropout(0.5))

model.add(Dense(2, activation='softmax'))

model.compile(loss = "categorical_crossentropy", optimizer='Adamax')

print(model.summary())

y_train.shape

history = model.fit(x_train, y_train, epochs = 30, batch_size = 40, verbose = 1,validation_data = (x_test, y_test))

loss, accuracy = model.evaluate(x_test, y_test)

plt.plot(history.history['loss'])

plt.plot(history.history['val_loss'])

plt.title('Model Loss')

plt.ylabel('Loss')

plt.xlabel('Epoch')

plt.legend(['Test', 'Validation'], loc='upper right')

plt.show()

def names(number):

    if number==0:

        return 'Its a Tumor'

    else:

        return 'No, Its not a tumor'

from matplotlib.pyplot import imshow
```

```
img = Image.open(r"Training/no_tumor/1.jpg")

x = np.array(img.resize((128,128)))

x = x.reshape(1,128,128,3)

res = model.predict_on_batch(x)

classification = np.where(res == np.amax(res))[1][0]

imshow(img)

print(str(res[0][classification]*100) + '% Confidence This Is ' + names(classification))

from matplotlib.pyplot import imshow

img = Image.open(r"Training/pituitary_tumor/p (1).jpg")

x = np.array(img.resize((128,128)))

x = x.reshape(1,128,128,3)

res = model.predict_on_batch(x)

classification = np.where(res == np.amax(res))[1][0]

imshow(img)

print(str(res[0][classification]*100) + '% Confidence This Is A ' + names(classification))

from keras.models import load_model

model.save("network.h5")

loaded_model = load_model("network.h5")

# loss, accuracy = loaded_model.evaluate(x_test, y_test)

from matplotlib.pyplot import imshow

img = Image.open(r"Training/pituitary_tumor/p (1).jpg")
```

```
x = np.array(img.resize((128,128)))

x = x.reshape(1,128,128,3)

res = loaded_model.predict_on_batch(x)

classification = np.where(res == np.amax(res))[1][0]

imshow(img)

print(str(res[0][classification]*100) + '% Confidence This Is A ' + names(classification))
```