

1.INTRODUCTION:

1.1 Overview

Brain tumor identification is a really challenging task in the early stages of life. These days the issue of brain tumor automatic identification is of great interest. A tumor is the unusual growth of the tissues. A brain tumor is a number of unnecessary cells growing in the brain or central spinal canal. It is the unrestrained progress of cancer cells in any portion of the body.

Deep learning techniques can be used in order to detect the brain tumor of a patient using the MRI images of a patient's brain. In this application, we are helping the doctors and patients to classify the type of scan for the specific image given with the help of Neural Networks and store the patient's data.

Brain tumor is one of the main causes of cancer death worldwide. Computer-aided diagnosis systems showed the potential for improving diagnostic accuracy. But early detection and prevention can significantly reduce the chances of death. It is important to detect Brain tumor as early as possible.

The goal is to classify images into two classifications of malignant and benign. As earlydiagnostics significantly increases the chances of correct treatment and survival. In application, we are helping the doctors and patients to classify the Type of Tumour for the specific image given with the help of Neural Networks. We will be using deep learning algorithm CNN, NumPy, TensorFlow, Keras, OpenCV and some other deep learning techniques. We will do Flask Integration and IBM Watson Deployment also.

1.2 Purpose

Since early detection and prevention can significantly reduce the chances of death the earlier, the better. The purpose here is to build a model in Watson Studio and deploy the model in IBM Watson Machine Learning.

2.LITERATURE SURVEY:

2.1 Existing problem

Doctors use many tests to find, or diagnose, brain tumor. They may also do tests to learn if the cancer has spread to a part of the body other than the brain tumor.

If this happens, it is called a metastasis. Doctors may also do tests to learn which treatments could work best. For most types of tumor, a biopsy is the only sure way for the doctor to know if an area of the body has tumor. In a biopsy, the doctor takes a small sample of tissue for testing in a laboratory. This section describes options for diagnosing tumor. Not all tests listed below will be used for every person. Doctors may consider these factors when choosing a diagnostic test:

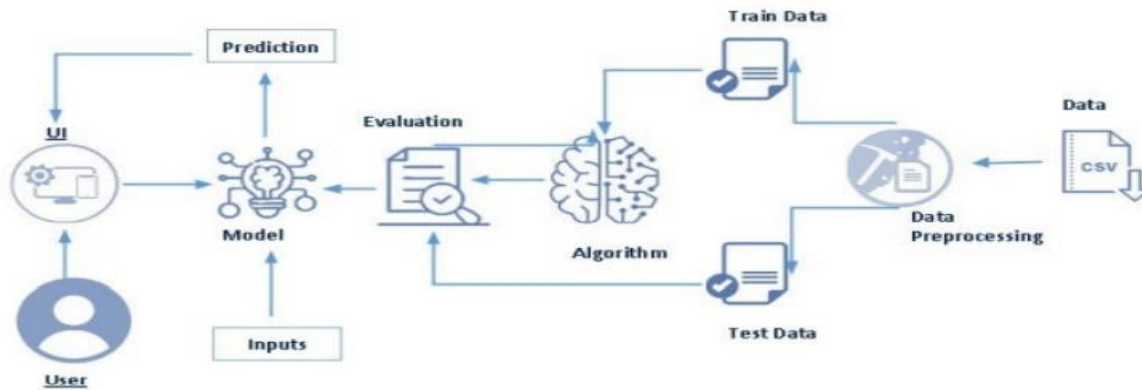
- The type of tumor suspected
 - Your signs and symptoms
 - Your age and general health
 - The results of earlier medical tests
- The series of tests needed to evaluate a possible brain tumor usually begins
- **Diagnostic mammography.** Diagnostic mammography is similar to screening mammography except that more pictures of the tumor. It is often used when a woman is experiencing signs, such as a new lump or nipple discharge. Diagnostic mammography may also be used if something suspicious is found on a screening mammogram.
 - **Ultrasound.** An ultrasound uses sound waves to create a picture of the tumor tissue. An ultrasound can distinguish between a solid mass, which may be tumor, and a fluid-filled cyst, which is usually not cancer.
 - **MRI.** An MRI uses magnetic fields, not x-rays, to produce detailed images of the body. A special dye called a contrast medium is given before the scan to help create a clear picture of the possible tumor. This dye is injected into the patient's vein.

2.2 Proposed Solution:

As can be seen, there is a series of tests and diagnosis to be carried out. A lot of processing and documentation and manual analysis is involved. Instead if all this data is retrieved into a model, early detection and prevention will significantly reduce the chances of death. The purpose here is to build a machine learning and deploy it in Watson Studio by creating an endpoint. To interact with the model, Node-Red and scoring endpoint to be used.

3.THEORITICAL ANALYSIS:

3.1 Block diagram:



3.2 Hardware /Software Design:

To complete this project, you must required following software's, concepts and packages.

1. Anaconda navigator and pycharm: a. Refer the link below to download anaconda navigator

b. Link : <https://youtu.be/1ra4zH2G4o0>

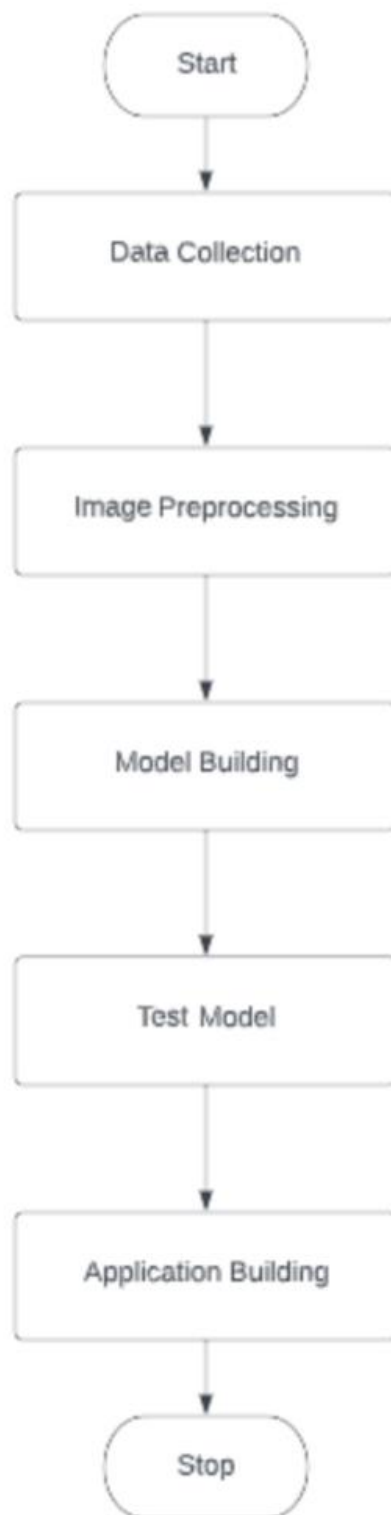
2. Python packages:

- Open anaconda prompt as administrator
- Type "pip install numpy" and click enter.
- Type "pip install pandas" and click enter.
- Type "pip install sklearn" and click enter.
- Type "pip install matplotlib" and click enter.
- Type "pip install seaborn" and click enter.
- Type "pip install Flask" and click enter

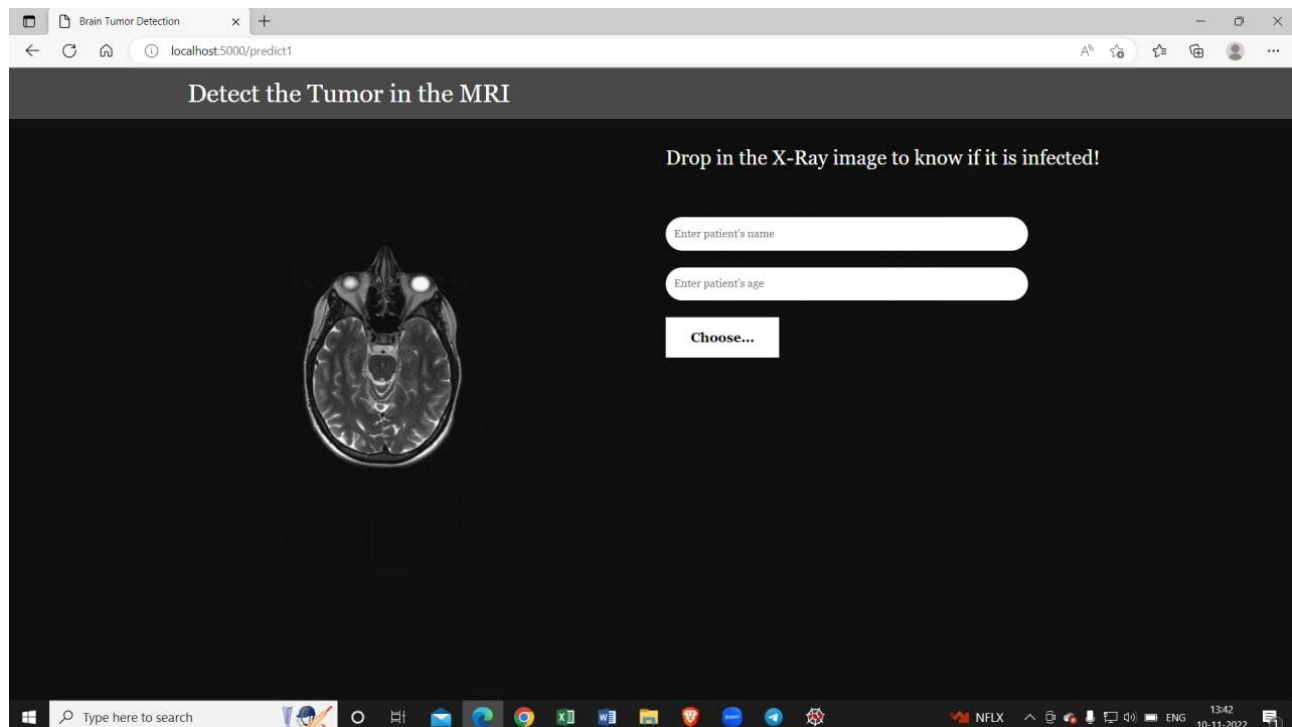
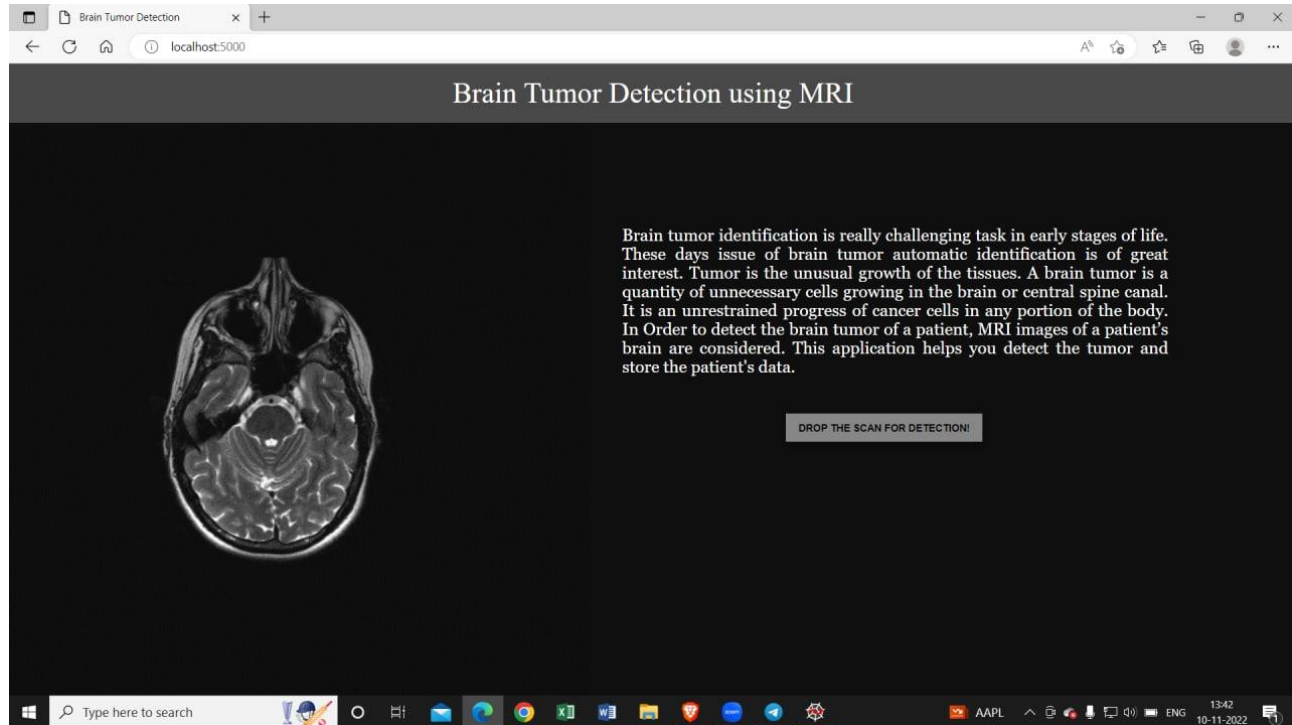
4.EXPERIMENTAL INVESTIGATIONS:

- ML Concepts of
 - **Supervised learning:** <https://www.javatpoint.com/supervised-machinelearning>
 - **Unsupervised learning:** <https://www.javatpoint.com/unsupervisedmachine-learning>
 - [Regression and classification](#)
 - Random forest: <https://www.javatpoint.com/machine-learning-randomforest-algorithm>
 - KNN: <https://www.javatpoint.com/k-nearest-neighbor-algorithm-formachine-learning>
 - Xgboost: <https://www.analyticsvidhya.com/blog/2018/09/an-end-to-endguide-to-understand-the-math-behind-xgboost/>
 - Evaluation metrics: <https://www.analyticsvidhya.com/blog/2019/08/11- important-model-evaluation-error-metrics/> o
 - Flask Basics : https://www.youtube.com/watch?v=lj4l_CvBnt0

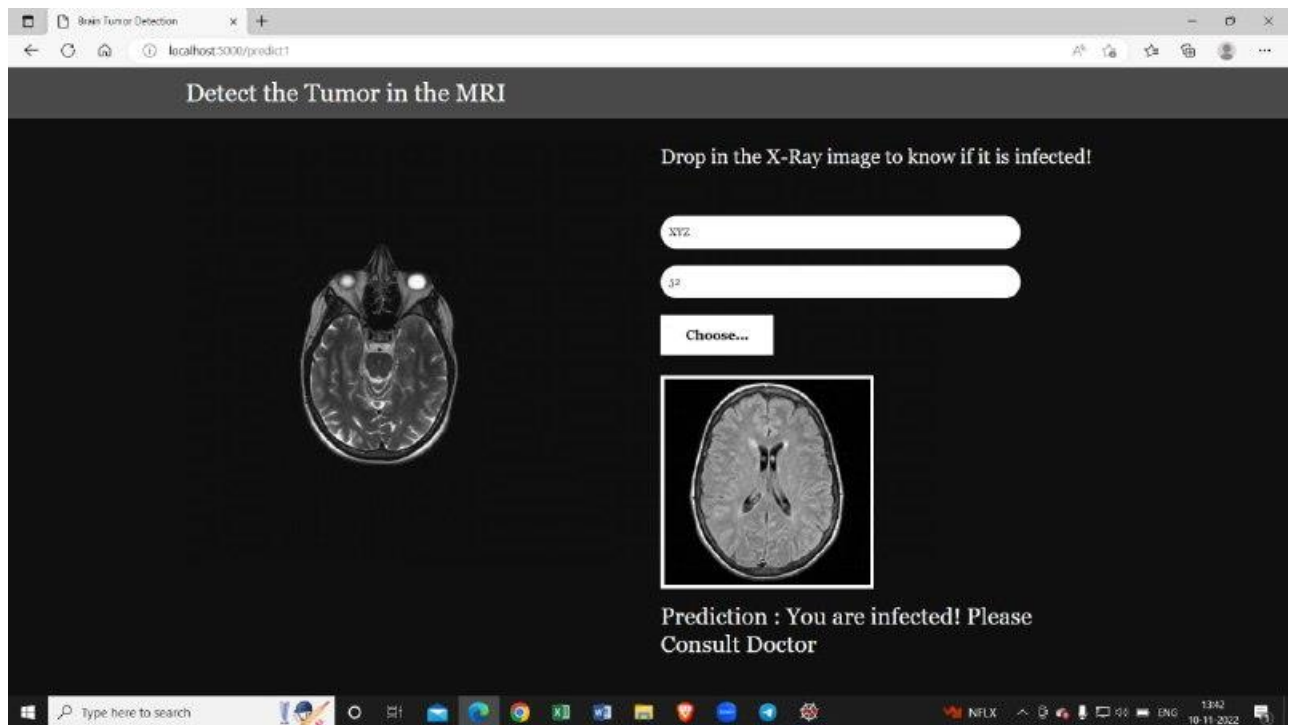
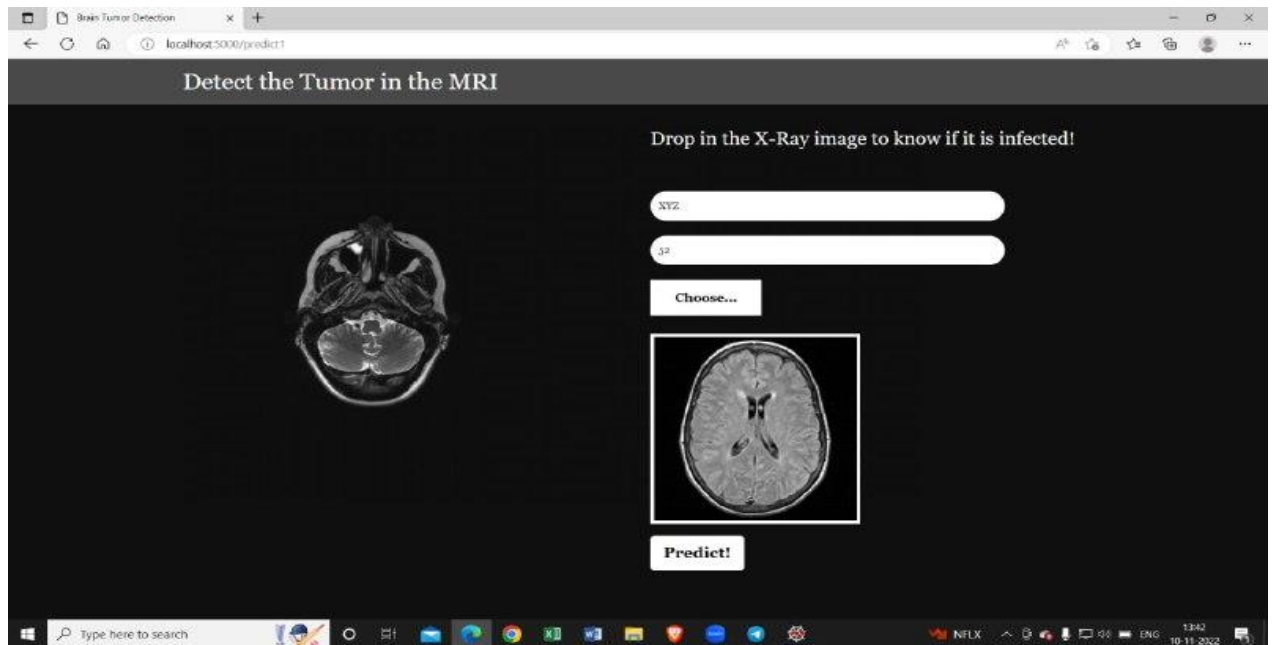
5.FLOWCHART:



RESULT:



RESULT:



7.ADVANTAGES & DISADVANTAGES:

Advantages:

- Increased accuracy for insurance prediction.
- Reduce the time complexity

Disadvantages.

- Data mining techniques does not help to provide effective decision making.

8.APPLICATIONS:

- Deep learning technology can be used for early detection of brain tumor.
- It represents the results obtained by processing input from uploading image.

9.CONCLUSION:

In this project, we have established the application to predict from uploaded image based on the IBM cloud application. Brain tumor prediction can be used as a web app to predict the tumor cell in brain.

10.FUTURE SCOPE:

The project can be further enhanced by deploying the deep learning model obtained using a web application and larger dataset cloud be used for prediction to give higher accuracy and produce better result.

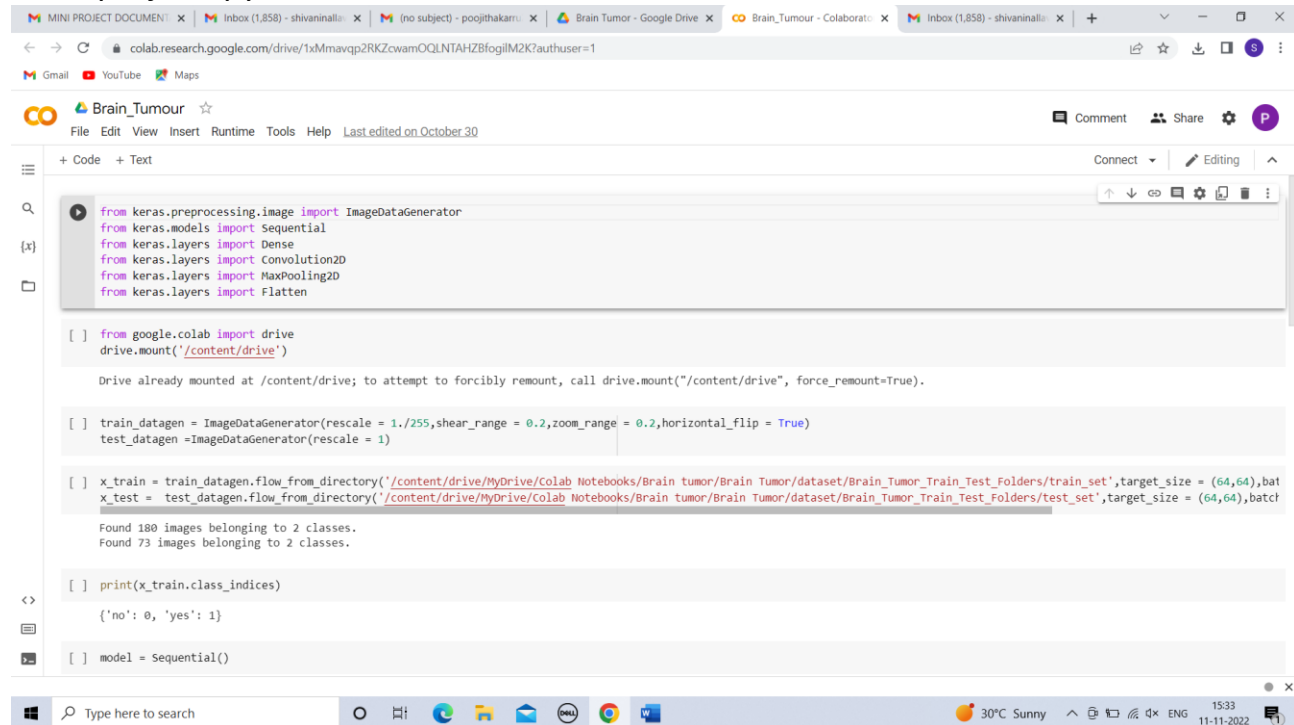
BIBLIOGRAPHY

- Supervised Learning: <https://www.javatpoint.com/supervised-machinelearning>
https://youtu.be/kE5QZ8G_78c
- Unsupervised Learning: <https://www.javatpoint.com/unsupervised-machinelearning>
https://www.youtube.com/watch?v=kE5QZ8G_78c
- Regression, Classification and Clustering: https://www.youtube.com/watch?v=6za9_mh3uTE
<https://www.geeksforgeeks.org/ml-classification-vs-clustering/>
- Artificial Neural Networks: <https://www.youtube.com/watch?v=DKSZHN7jftI>
<https://www.javatpoint.com/artificial-neural-network>
- Convolution Neural Networks (CNN) : https://www.youtube.com/watch?v=umGJ30-15_A
<https://www.geeksforgeeks.org/introduction-convolution-neural-network>

BIBLIOGRAPHY APPENDIX:

SOURCE CODE

Main project.ipynb



```
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten

[ ] from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[ ] train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1)

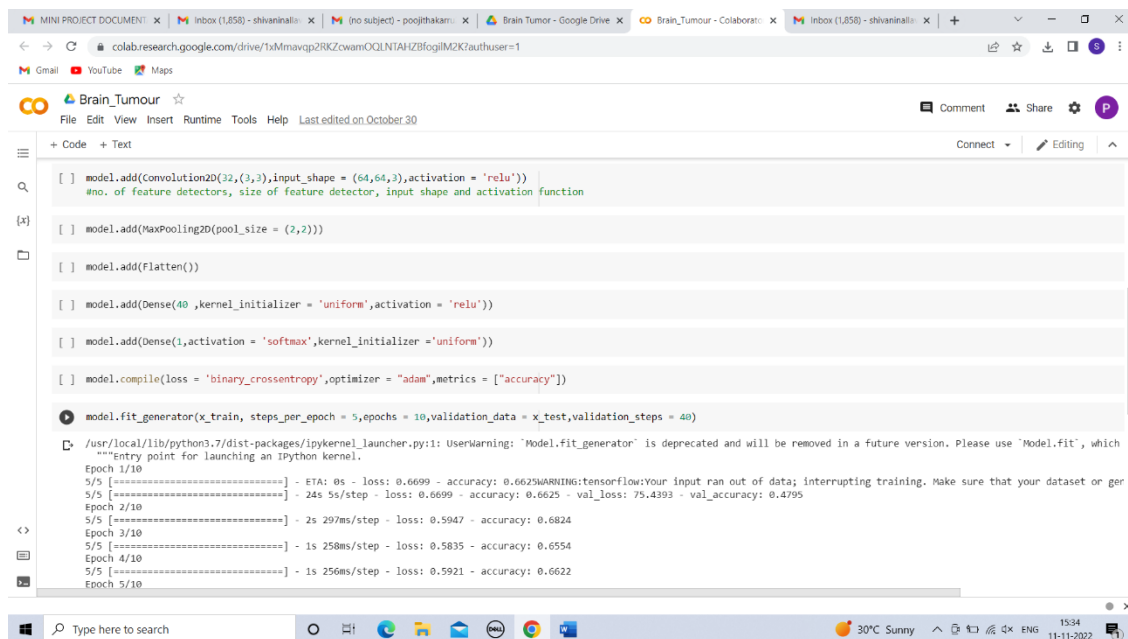
[ ] x_train = train_datagen.flow_from_directory('/content/drive/MyDrive/Colab Notebooks/Brain tumor/Brain Tumor/dataset/Brain_Tumor_Train_Test_Folders/train_set', target_size = (64,64), batch_size = 32)
x_test = test_datagen.flow_from_directory('/content/drive/MyDrive/Colab Notebooks/Brain tumor/Brain Tumor/dataset/Brain_Tumor_Train_Test_Folders/test_set', target_size = (64,64), batch_size = 32)

Found 180 images belonging to 2 classes.
Found 73 images belonging to 2 classes.

[ ] print(x_train.class_indices)

{'no': 0, 'yes': 1}

[ ] model = Sequential()
```



```
[ ] model.add(Convolution2D(32,(3,3),input_shape = (64,64,3),activation = 'relu'))
#no. of feature detectors, size of feature detector, input shape and activation function

[ ] model.add(MaxPooling2D(pool_size = (2,2)))

[ ] model.add(Flatten())

[ ] model.add(Dense(40, kernel_initializer = 'uniform', activation = 'relu'))

[ ] model.add(Dense(1, activation = 'softmax', kernel_initializer = 'uniform'))

[ ] model.compile(loss = 'binary_crossentropy', optimizer = "adam", metrics = ["accuracy"])

[ ] model.fit_generator(x_train, steps_per_epoch = 5, epochs = 10, validation_data = x_test, validation_steps = 40)

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: 'Model.fit_generator' is deprecated and will be removed in a future version. Please use 'Model.fit', which
is the entry point for launching an IPython kernel.
Epoch 1/10
5/5 [=====] - ETA: 0s - loss: 0.6699 - accuracy: 0.6625WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your dataset or generator
can generate enough data.
Epoch 2/10
5/5 [=====] - 2s 297ms/step - loss: 0.5947 - accuracy: 0.6824
Epoch 3/10
5/5 [=====] - 1s 258ms/step - loss: 0.5835 - accuracy: 0.6554
Epoch 4/10
5/5 [=====] - 1s 256ms/step - loss: 0.5921 - accuracy: 0.6622
Epoch 5/10
```

BIBLOGRAPHY

The screenshot shows a Google Colab notebook interface. The browser tabs at the top include 'MINI PROJECT DOCUMENT', 'Inbox (1,858) - shivaninalli', 'no subject - poojithakaru', 'Brain Tumor - Google Drive', 'Brain_Tumour - Colaboratory', and another 'Inbox (1,858) - shivaninalli'. The notebook's address bar shows a Google Drive link. The notebook title is 'Brain_Tumour' with a star icon. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help', with a note 'Last edited on October 30'. The toolbar has 'Comment', 'Share', and a settings icon. The main code editor shows a list of training epochs with their respective loss and accuracy values, followed by a command to save the model as 'braintumor.h5'. The left sidebar contains icons for file explorer, search, and other notebook functions. The bottom status bar shows the Windows taskbar with a search bar, application icons, and system information: '30°C Sunny', '15:34', and '11-11-2022'.

```
[ ] Epoch 5/10  
5/5 [=====] - 1s 268ms/step - loss: 0.5695 - accuracy: 0.6757  
Epoch 6/10  
5/5 [=====] - 1s 253ms/step - loss: 0.5348 - accuracy: 0.6689  
Epoch 7/10  
5/5 [=====] - 1s 249ms/step - loss: 0.5794 - accuracy: 0.6689  
Epoch 8/10  
5/5 [=====] - 1s 281ms/step - loss: 0.5802 - accuracy: 0.6284  
Epoch 9/10  
5/5 [=====] - 1s 248ms/step - loss: 0.5592 - accuracy: 0.6892  
Epoch 10/10  
5/5 [=====] - 1s 286ms/step - loss: 0.5508 - accuracy: 0.6689  
<keras.callbacks.History at 0x7fcd27f06bd0>  
  
[ ] model.save("braintumor.h5")  
  
[ ]
```

App.py

```
1 import numpy as np  
2 import os  
3 from keras.preprocessing import image  
4 import pandas as pd  
5 import cv2  
6 import tensorflow as tf  
7 # Flask utils  
8 from flask import Flask, request, render_template  
9 from werkzeug.utils import secure_filename  
10 from tensorflow.python.keras.backend import set_session  
11 from tensorflow.python.keras.models import load_model  
12  
13 global graph  
14 tf.compat.v1.disable_eager_execution()  
15 sess = tf.compat.v1.Session()  
16  
17  
18 import tensorflow.compat.v1 as tf  
19 tf.disable_v2_behavior()  
20 graph=tf.get_default_graph()  
21  
22 # Define a flask app  
23 app = Flask(__name__)  
24 set_session(sess)  
25 # Load your trained model  
26 model = load_model('braintumor.h5')  
27  
28 print('Model Loaded. Check http://127.0.0.1:5000/')  
29  
30  
31 @app.route('/', methods=['GET'])  
32 def index():  
33     .....
```

BIBLOGRAPHY

```
31 @app.route('/', methods=['GET'])
32 def index():
33     # Main page
34     return render_template('brainintro.html')
35
36 @app.route('/predict1', methods=['GET'])
37 def predict1():
38     # Main page
39     return render_template('brainpred2.html')
40
41
42 @app.route('/predict', methods=['GET', 'POST'])
43 def upload():
44     if request.method == 'POST':
45         # Get the file from post request
46         df= pd.read_csv('patient.csv')
47         f = request.files['image']
48         name=request.form['name']
49         age=request.form['age']
50
51         # Save the file to ./uploads
52         basepath = os.path.dirname(__file__)
53         file_path = os.path.join(
54             basepath, 'uploads', secure_filename(f.filename))
55         f.save(file_path)
56         img = image.load_img(file_path, target_size=(64, 64))
57         x = image.img_to_array(img)
58         x = np.expand_dims(x, axis=0)
```

```
59
60     with graph.as_default():
61         set_session(sess)
62         prediction = model.predict_classes(x)[0][0]
63     print(prediction)
64     if prediction==0:
65         text = "You are perfectly fine"
66         inp = "No tumor"
67     else:
68         text = "You are infected! Please Consult Doctor"
69         inp="Tumor detected"
70
71     df=df.append(pd.DataFrame({'name':[name], 'age':[age], 'status':[inp]
72     df.to_csv('patient.csv',index = False)
73     return text
74
75 if __name__ == '__main__':
76     app.run(debug=False, threaded = False)
77
```