BRAIN TUMOR CLASSIFICATION AND PREDICTION USING CONVOLUTION-AL NEURAL NETWORKS

CAPSTONE PROJECT THREE
DATA SCIENCE CAREER TRACK

SPRINGBOARD

SHAHJAHAN AHMED
JANUARY 02, 2022

Problem Statement

Brain Tumor Classification and Prediction using Convolutional Neural Networks to help automate the diagnostic process which will ensure proper treatment and save lives and resources.

Introduction

- A brain tumor is a collection, or mass of abnormal cells in our brain. Our skull, which encloses our brain, is very rigid. Any growth inside such a restricted space can cause problems.

- Brain tumors can be cancerous (malignant) or noncancerous (benign). When benign or malignant tumors grow, they can cause the pressure inside our skull to increase. This can cause brain damage, and it can be life-threatening.

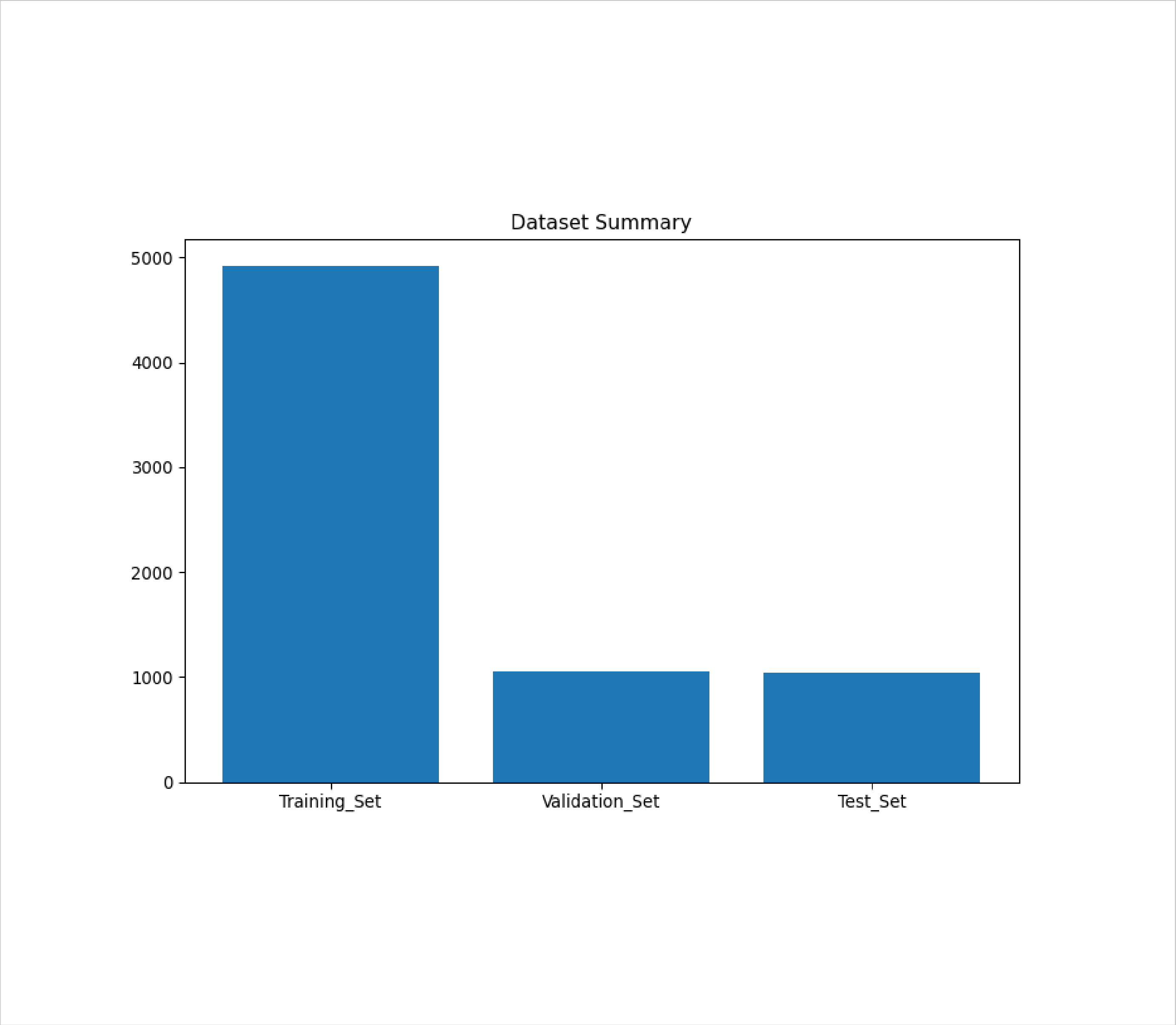
- Early detection and classification of brain tumors is an important research domain in the field of medical imaging and accordingly helps in selecting the most convenient treatment method to save patients life.

Stakeholders

- ❑ Doctors
- ❑ Hospitals
- ❑ Medical Centers
- ❑ Patients

Dataset Summary

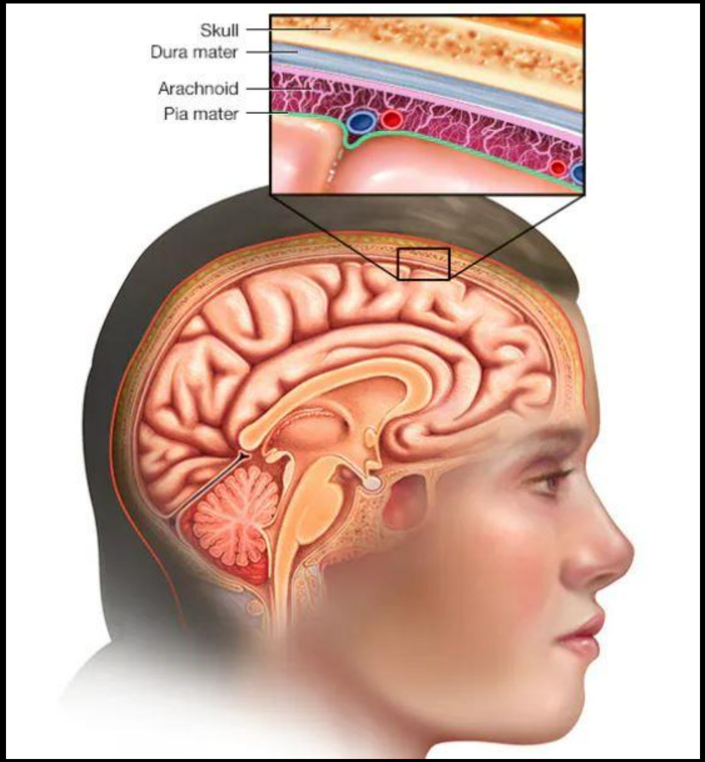
- The dataset contains four separate classes of brain MRI images: Glioma Tumor, Meningioma Tumor, Pituitary Tumor, and Absence of Tumor.
- The total number of images is 7023 and I have divided them into training set, validation set, and test set with a 70:15:15 ratio.
- Each of the training, validation, and test dataset contains all four MRI image classes.

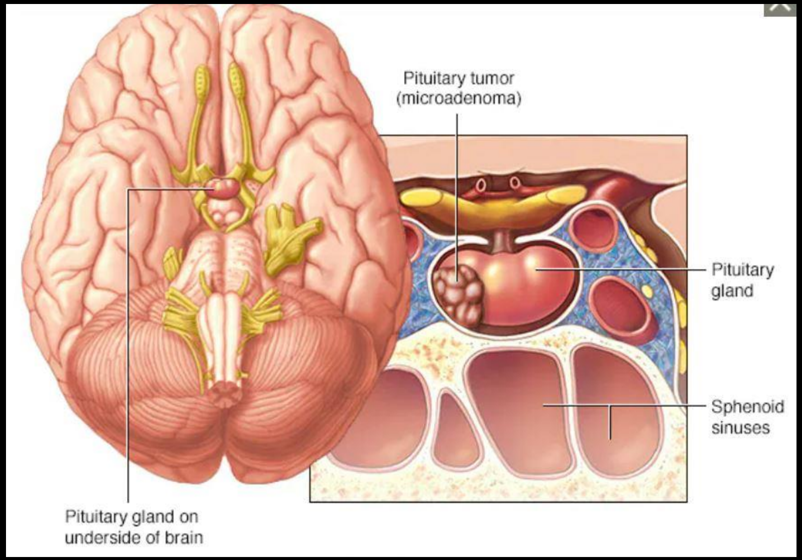


Brain Tumor Classes

- ❑ Glioma Tumor: Glioma is a type of tumor that occurs in the brain and spinal cord. Gliomas begin in the gluey supportive cells (glial cells) that surround nerve cells and help them function.
- ❑ Pituitary Tumor: Pituitary tumors are abnormal growths that develop in our pituitary gland. Some pituitary tumors result in too much of the hormones that regulate important functions of our body.
- ❑ Meningioma Tumor: A meningioma is a tumor that arises from the meninges — the membranes that surround our brain and spinal cord. Meningioma is the most common type of tumor that forms in the head.
- ❑ No Tumor: The MRI image does not contain any kinds of tumor cell.

Brain Tumor Classes



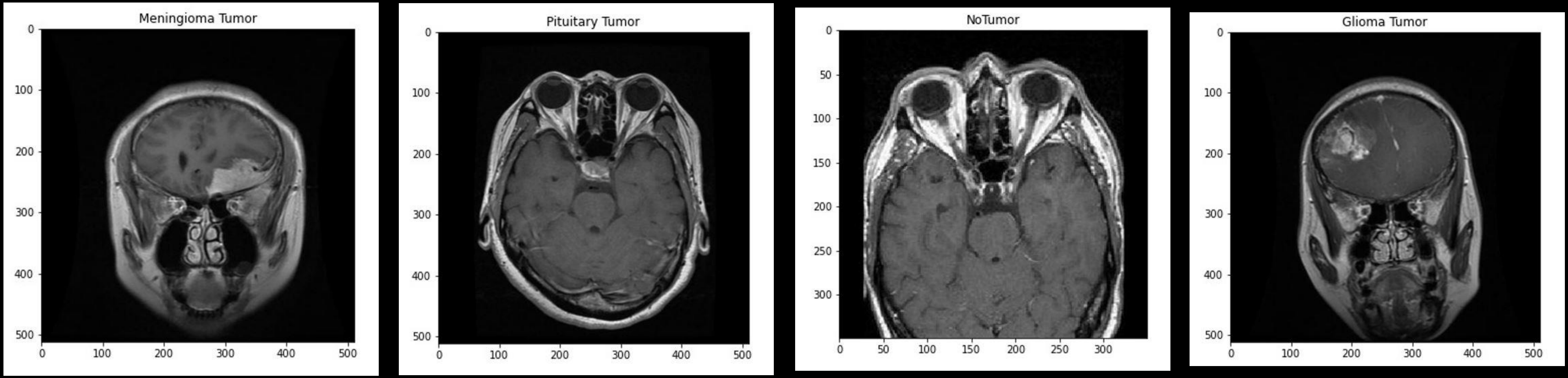


Data Wrangling

- ❑ Observed all the MRI images in training, validation, and test set with their respective file format.
- ❑ Observed whether the dataset contain RGB or grayscale images.
- ❑ Applied different filtering techniques to improve the image quality.
- ❑ Observed the shape of each images in training, validation, and test set for resizing.
- ❑ Converted all the images into gray scale image of size (128, 128, 1).

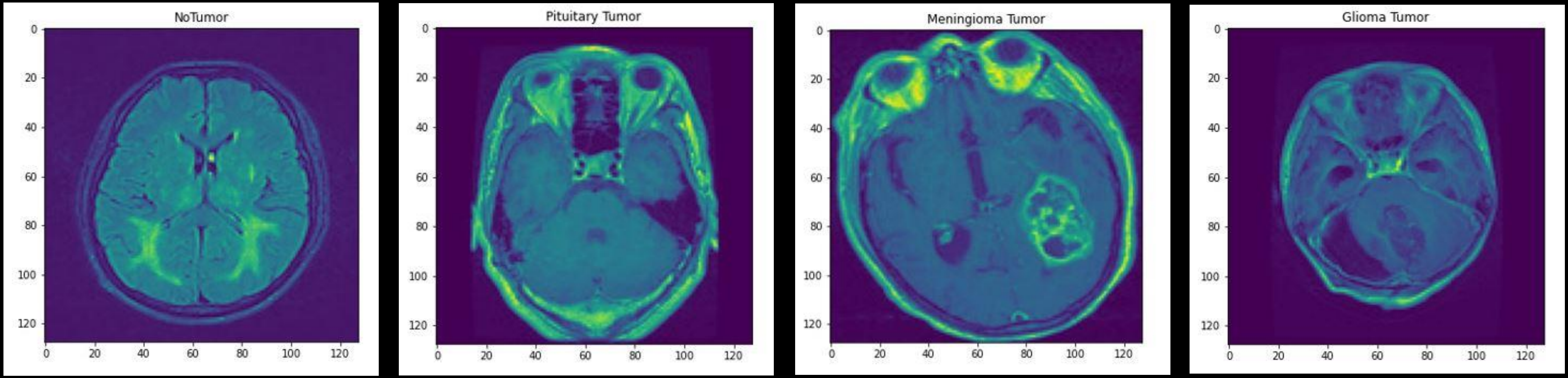
Exploratory Data Analysis Step one

- The dataset contains RGB image with different shapes.
- There are four tumor classes in each training, validation, and test set.
- The figure below depicted the MRI images of four different tumor classes.



Exploratory Data Analysis Step two

- Applied some filtering techniques.
- RGB to grayscale conversion
- Figure below shows the resized images of shape (128,128,1).



Preprocessing

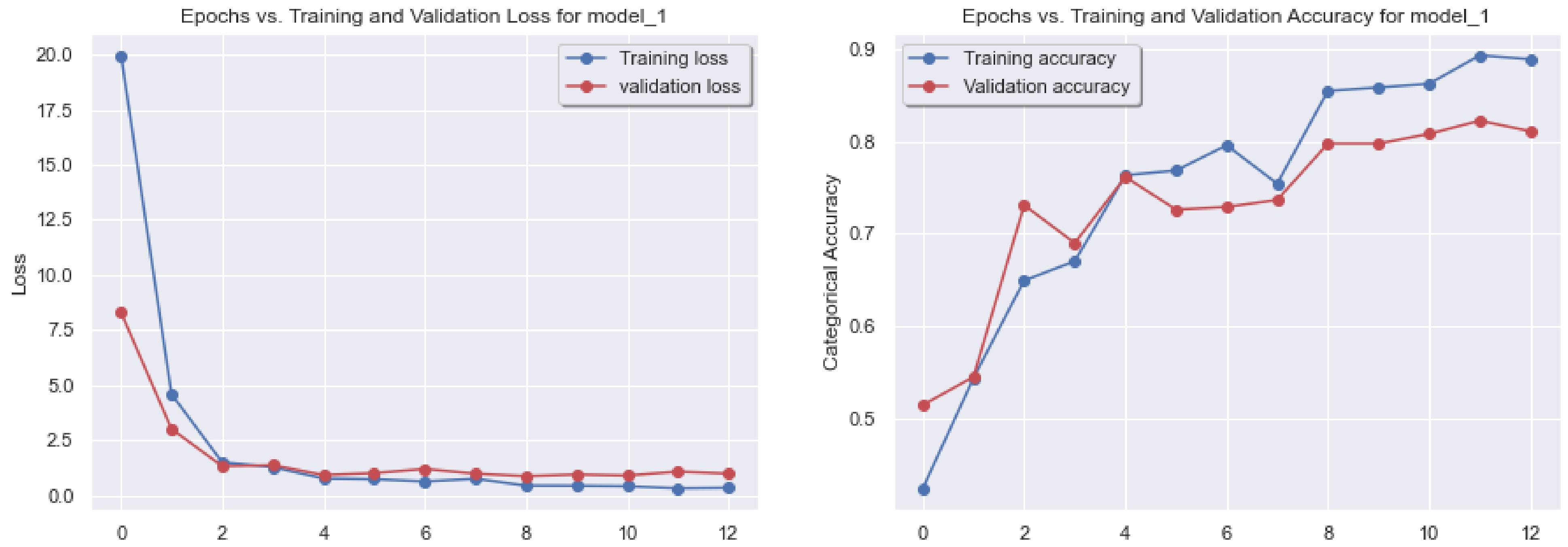
- ❑ Created image arrays and corresponding labels for training, validation, and test dataset.
- ❑ The image arrays are respectively X_train, X_valid, and X_test, and the corresponding labels for image arrays are y_train, y_valid, and y_test.
- ❑ Created one-hot vectors for y_train, y_test, and y_valid

Modeling Steps

- ❑ Implement different Neural Network Architectures for Brain Tumor Classification and Prediction
- ❑ Set up appropriate hyperparameters for each model
- ❑ Compile each model using appropriate optimizer, loss function, and model metrics
- ❑ Train each model using training set and predict the model's performance using test set
- ❑ Evaluate test results of each model using evaluation metrics
- ❑ Compare and find the best model

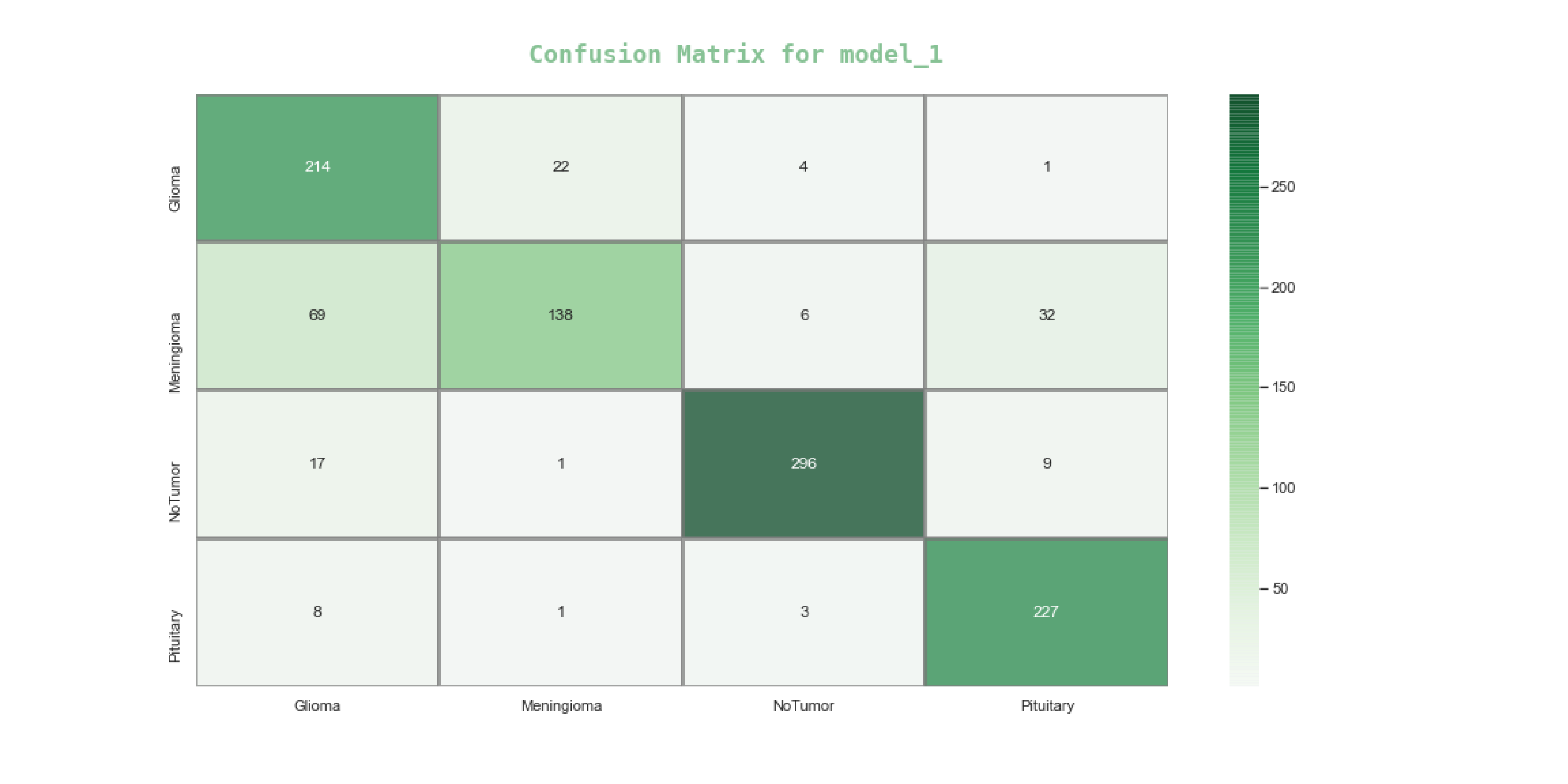
Model 1: Simple neural network with only dense layers

- The model 1 contains an input layer, six Dense layers, and one output layer.
- I have used ‘ReLU’ as an activation function for the Dense layers and ‘softmax’ as an activation function for the output layer
- I have defined Adam as an optimizer, categorical accuracy as a loss function, categorical cross entropy as a model metric.
- No Regularization technique used.



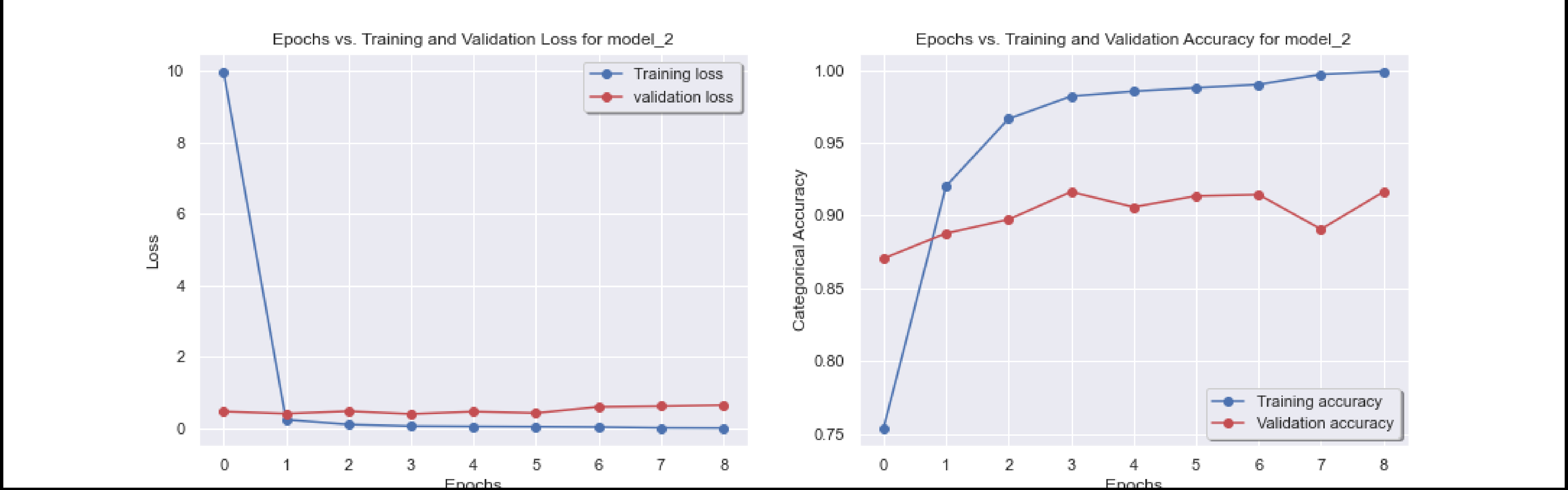
Evaluation Results for Model 1

- Training Accuracy: 0.893
- Validation Accuracy: 0.822
- Test Accuracy: 0.834
- Precision Score: 0.85
- Recall Score: 0.83
- F1 Score: 0.83



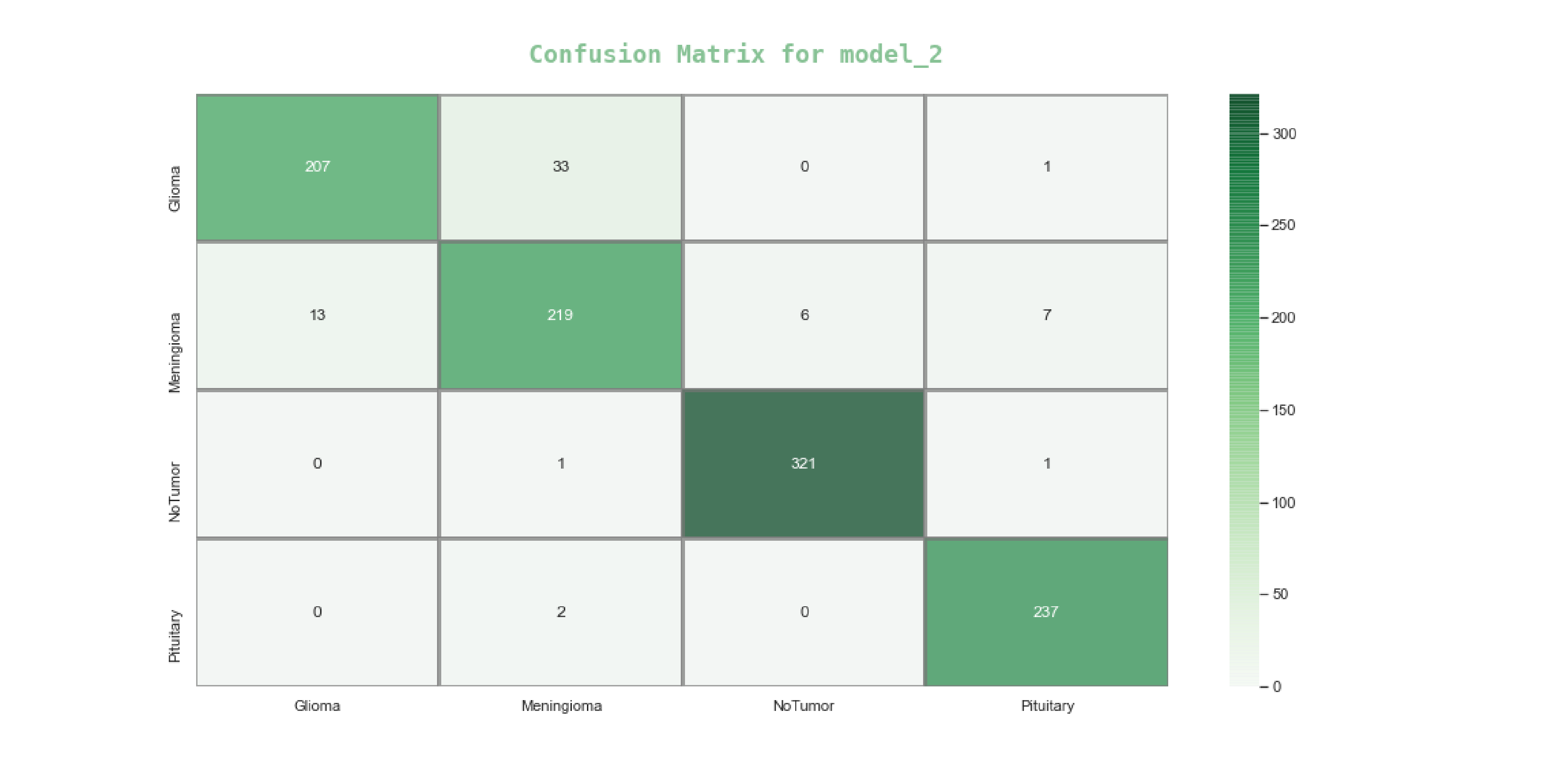
Model 2: Convolutional neural network with two Conv2D layers

- Model 2 contains an input layer, two Conv2D layers with MaxPolling2D layer, one Dense layer, and an output layer.
- I have used ‘ReLU’ as an activation function for the hidden layers and ‘softmax’ as an activation function for the output layer
- I have defined Adam as an optimizer, categorical accuracy as a loss function, categorical cross entropy as a model metric.
- No Regularization technique used.



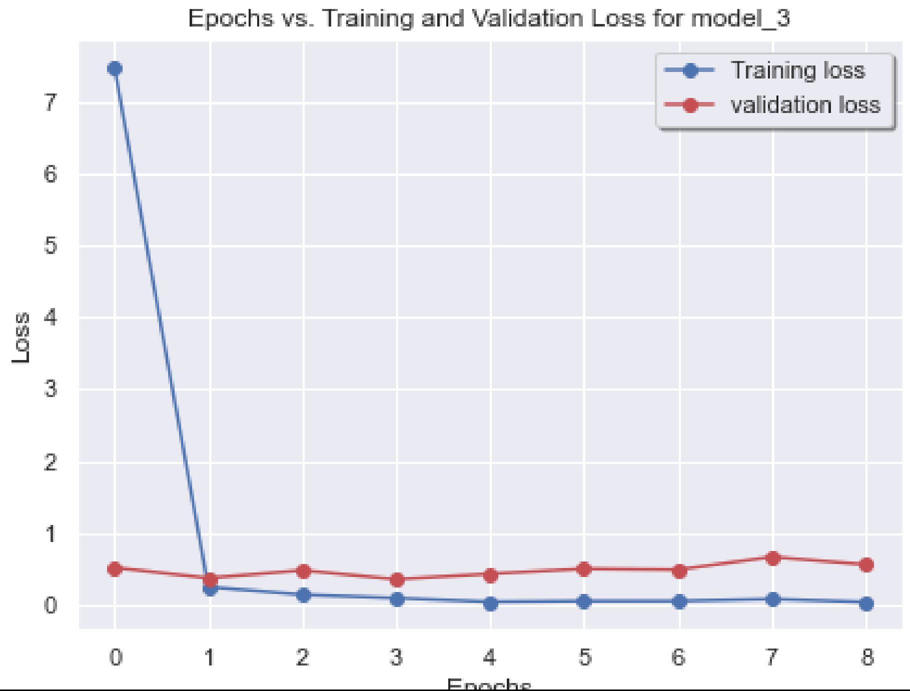
Evaluation Results for Model 2

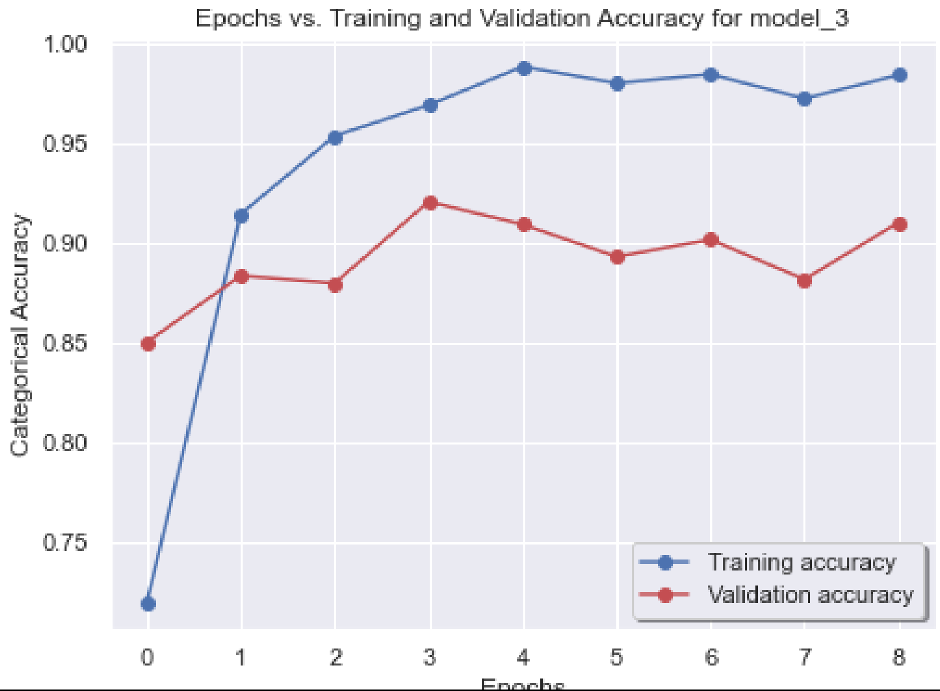
- Training Accuracy: 0.999
- Validation Accuracy: 0.916
- Test Accuracy: 0.938
- Precision Score: 0.94
- Recall Score: 0.94
- F1 Score: 0.94



Model 3: Convolutional neural network with two Conv2D layers and an addition of regularization layers

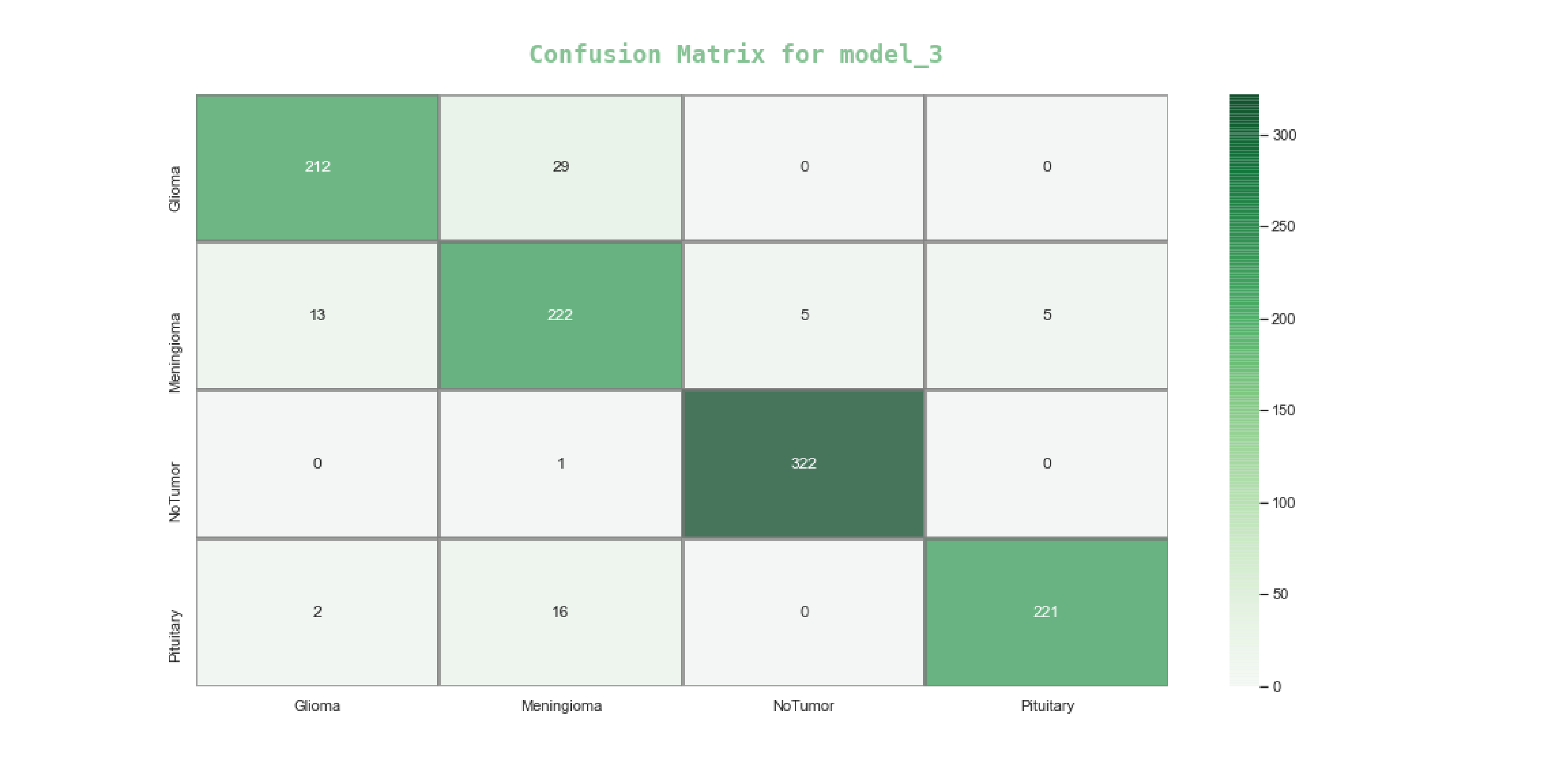
- Model 3 is a copy of model 2 with some additional regularization techniques.
- To overcome the overfitting caused by model 2, I have added BatchNorm2D layer after each convolutional layers and a Dropout layer after the dense layer.
- The optimizer, loss function, model metrics have kept same with previous model.





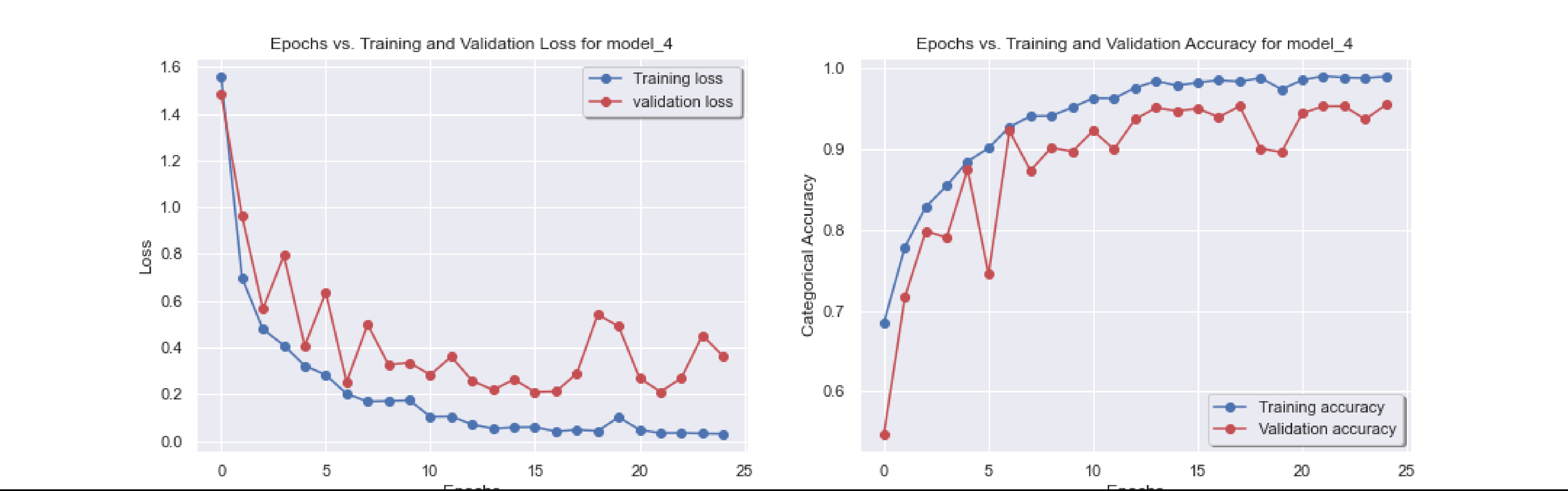
Evaluation Results for Model 3

- Training Accuracy: 0.988
- Validation Accuracy: 0.921
- Test Accuracy: 0.932
- Precision Score: 0.93
- Recall Score: 0.93
- F1 Score: 0.93



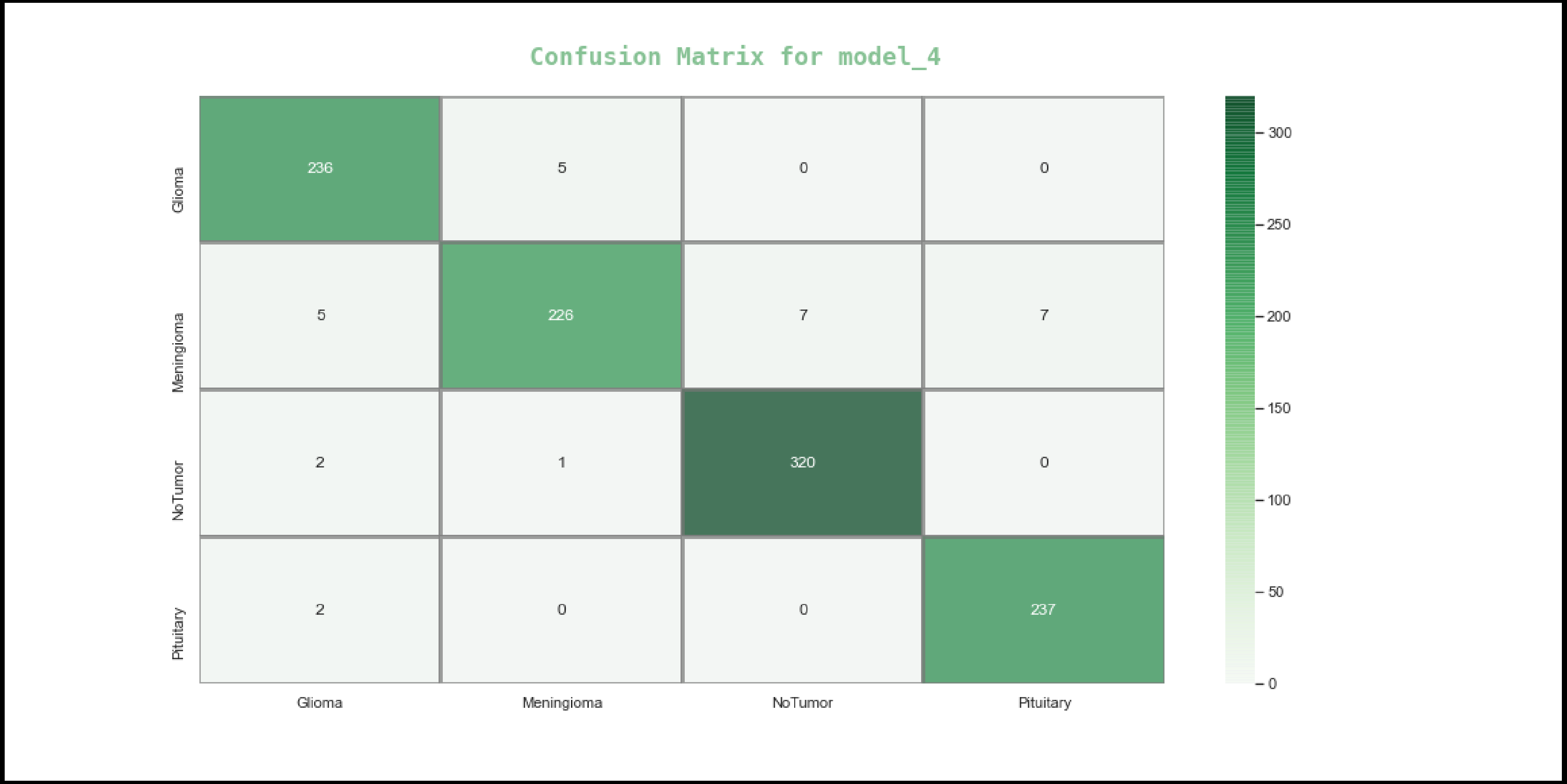
Model 4: Convolutional neural network with three Conv2D layers and regularization layers

- Model 4 contains an input layer, three Conv2D layers with MaxPolling2D and BatchNorm2D layers, two Dense layers, and an output layer.
- The activation function, optimizer, loss function, and model metrics remains same with previous models.
- Additionally, I have introduced learning rate scheduling and EarlyStopping for adaptive training.
- BatchNorm2D, and Dropout layers used for regularization.



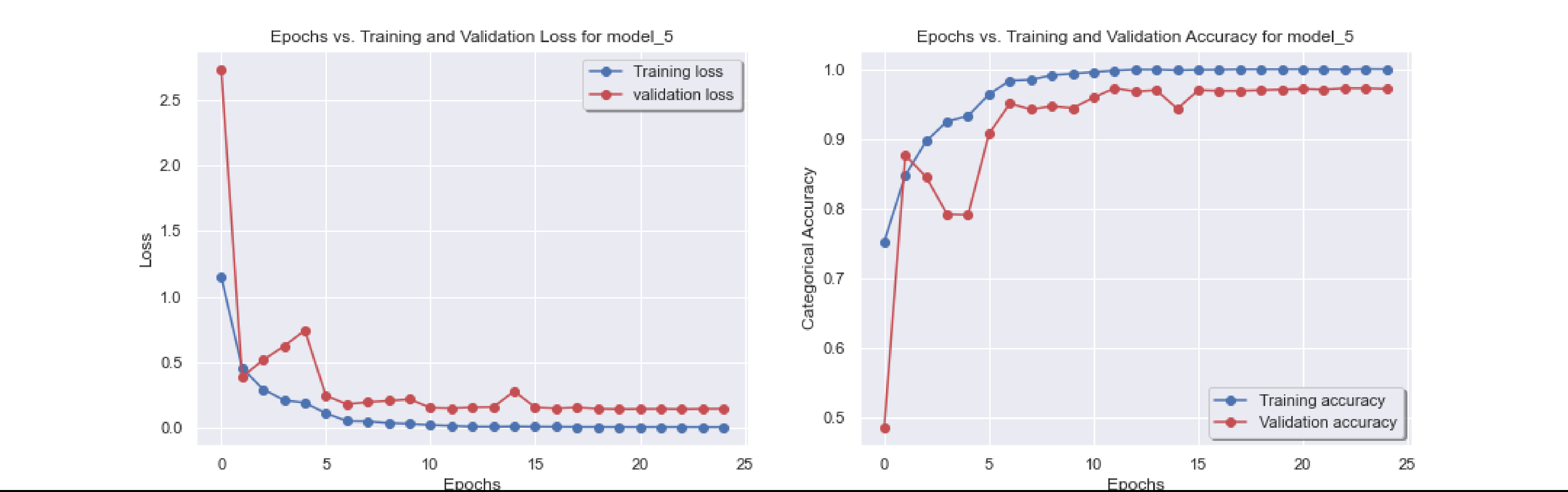
Evaluation Results for Model 4

- Training Accuracy: 0.990
- Validation Accuracy: 0.955
- Test Accuracy: 0.972
- Precision Score: 0.97
- Recall Score: 0.97
- F1 Score: 0.97



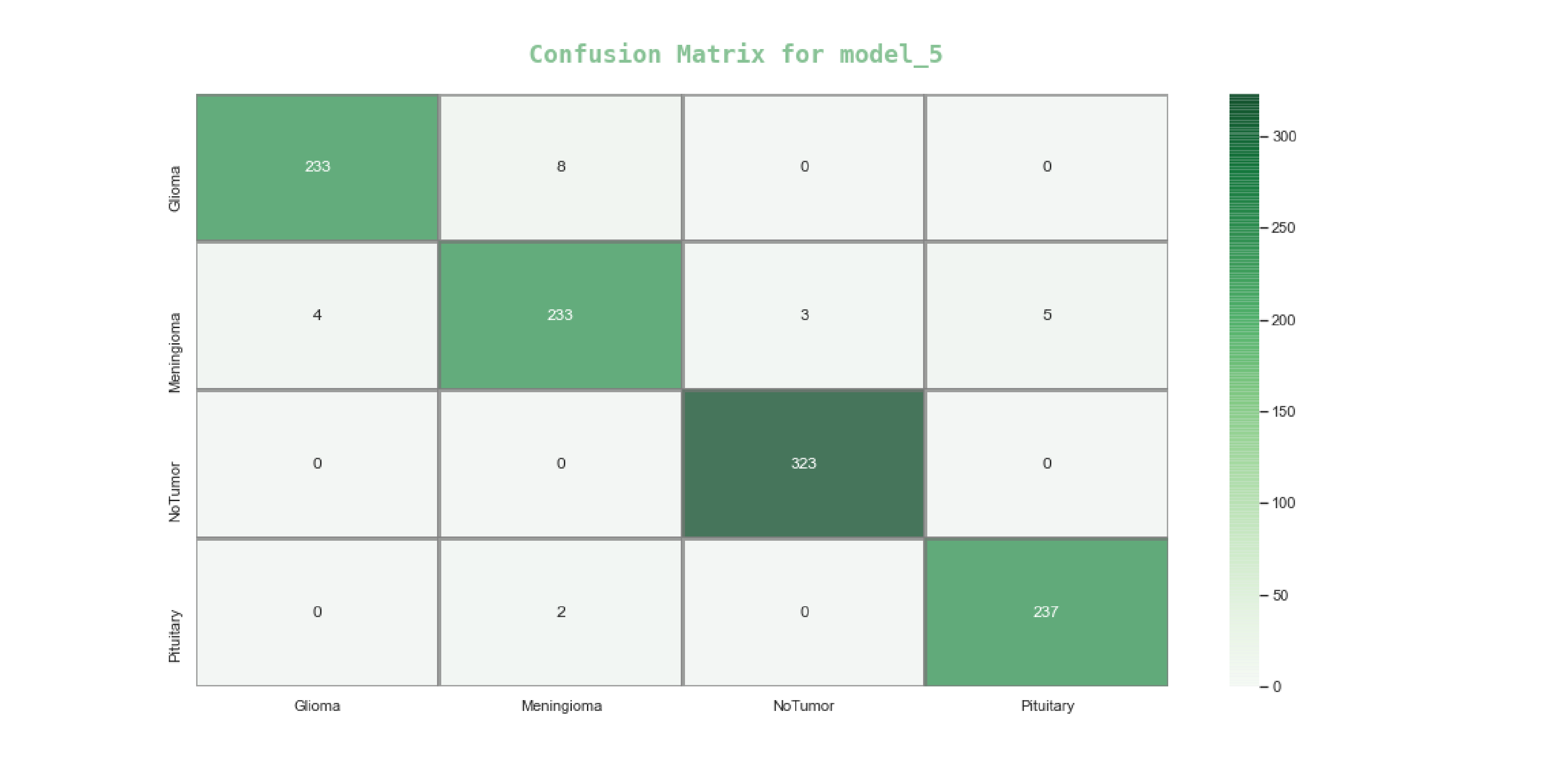
Model 5: Convolutional neural network with four Conv2D layers and regularization layers

- ❑ Model 5 contains an input layer, four Conv2D layers with MaxPolling2D and BatchNorm2D layers, one Dense layer, and an output layer.
- ❑ The activation function, optimizer, loss function remains same with previous models.
- ❑ I have introduced learning rate scheduling and EarlyStopping for adaptive training.
- ❑ BatchNorm2D, and Dropout layers for regularization



Evaluation Results for Model 5

- Training Accuracy: 1.00
- Validation Accuracy: 0.972
- Test Accuracy: 0.979
- Precision Score: 0.98
- Recall Score: 0.98
- F1 Score: 0.98



Results and Findings

- Model 1 results training accuracy of 0.893, validation accuracy of 0.822, and test accuracy of 0.834. The weighted value of precision, recall, and F1-score is 0.85, 0.83, and 0.83, respectively.
- Model 2 results training accuracy of 0.999, validation accuracy of 0.916, and test accuracy of 0.938. The weighted value of precision, recall, and F1-score is 0.94.
- Model 3 results training accuracy of 0.988, validation accuracy of 0.921, and test accuracy of 0.932. The weighted value of precision, recall, and F1-score is 0.93.
- Model 4 results training accuracy of 0.990, validation accuracy of 0.955, and test accuracy of 0.972. The weighted value of precision, recall, and F1-score is 0.97.
- Finally, the optimum model (model 5) results training accuracy of 100%, validation accuracy of 97.2%, and test accuracy of 97.9%. The weighted value of precision, recall, and F1-score is 98%.

Table showing the comparison among all Models Outcome

	Training accuracy	Validation Accuracy	Test Accuracy	Weighted Average		
				Precision	Recall	F1-Score
Model 1	0.89	0.822	0.834	0.85	0.83	0.83
Model 2	0.99	0.916	0.938	0.94	0.94	0.94
Model 3	0.98	0.921	0.932	0.93	0.93	0.93
Model 4	0.99	0.955	0.972	0.97	0.97	0.97
Model 5	1.00	0.972	0.979	0.98	0.98	0.98

Conclusion

- Despite lack of proper computational resources, I have implemented, trained, and tested all my models and obtained good evaluation results.
- The optimum model (model 5) results training accuracy of 100%, validation accuracy of 97.2%, test accuracy of 97.9%, and the weighted value of precision, recall, and F1-score of 98%.
- Model 5 shows a significantly low misclassification: only 23 misclassifications out of 1048 test images.
- More precisely there is only 2.19% misclassification and out of them 1.5% false negatives and 0.57% false positives.

Thank You