******Brain Tumor Identification in MRI Images**

**using K-means Clustering and XGBoost**

MS Machine Learning

Columbia University in the City of New York

**Skand Upmanyu**

MS Business Analytics, Columbia University

su2236@columbia.edu

***Abstract*:** Brain tumor is a cancerous or noncancerous mass or growth of abnormal cells in the brain. An estimated 700,000 Americans are living with a brain tumor. Of these, 69.8% tumors are benign and 30.2% tumors are malignant. Moreover, an estimated 87,240 people will receive a primary brain tumor diagnosis in 2020 [1]. Since these cases have an average survival rate for the malignant tumor cases is only 36%, early identification is very crucial for such situations. Brain tumors can be identified using MRI (Magnetic Resonance Imaging) diagnostic tests which is a time-consuming task for the doctors and healthcare specialists. Due to the high volume of MRI tests every year, an automated process of brain tumor identification from these images can help assist the pathology experts in speeding up the process of brain tumor identification. This study proposes the combination of supervised and unsupervised machine learning techniques to classify MRI images based on the presence of brain tumors. This crucial task requires high accuracy and therefore, we will have utilized the state-of-the-art techniques like XGBoost to achieve high specificity as well as high sensitivity.

**I. Introduction**

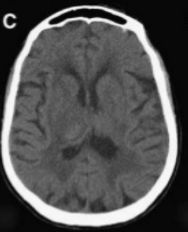
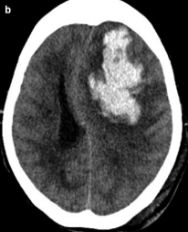
Brain tumors are the fifth-leading cause of cancer-related death in males age 40-59 years [1]. 3,657 children are estimated to be living with a brain tumor in the U.S. [1]. More than any other cancer, brain tumors can have lasting and life-altering physical, cognitive, and psychological impacts on a patient’s life. In most people with primary brain tumors, the cause of the tumor is not clear, but the doctors have identified some factors that may increase your risk of a brain tumor which include Exposure to radiation and Family history of brain tumors [2].

[1] Quick Brain Tumor Facts: <https://braintumor.org/brain-tumor-information/brain-tumor-facts/>

[2] Mayo Clinic: <https://www.mayoclinic.org/diseases-conditions/brain-tumor/symptoms-causes/syc-20350084>

Diagnosis of a brain tumor is done by a neurologic exam (by a neurologist or neurosurgeon), CT (computer tomography scan) and/or magnetic resonance imaging (MRI), and other tests like an angiogram, spinal tap and biopsy. Of these, magnetic MRI is commonly used to help diagnose brain tumors. It uses magnetic fields, not x-rays, to produce detailed images of the body. These tumors are mostly identifiable with the human eye but automating the process of tumor identification would require the use of complex machine learning techniques which we will introduce in this study.

Examples of MRI scans with and without brain tumor are depicted below:



**MRI scan with no brain tumor**

**MRI scan with brain tumor**

**Tumor**

*Figure 1.1 MRI Scans with and without tumor*

Our approach to identify the tumors from MRI images is outlines below:

1. Loading MRI scans

2. Image Augmentation to expand dataset

3. K-means clustering for feature generation

4. XGBoost for classification (tumor/non-tumor)

*Figure 1.2 Process flow*

**II. Data Processing and Image Augmentation**

1. *Data source*

A data source of 200 images was obtained from “Brain MRI Images for Brain Tumor Detection” [3] by Navoneel Chakrabarty. This dataset contains MRI images with the following summary statistics:

|  |  |
| --- | --- |
| **Data** | **Attribute** |
| Total images | 200 |
| Tumor MRI images | 100 |
| Non-tumor MRI images | 100 |
| Image size | Varying |

1. *Image resizing*

Since the images were of varying size, the images were transformed to have the same dimensions of 188 x 232.

1. *Image augmentation to expand dataset*

Due to the limitations in the size of the dataset (200 images), we performed image augmentation to expand our dataset for training. This involved rotating the images across various axes to obtain different versions of the same image which can then be utilized for training our model. Following transformations were performed for image augmentation:

* 1. Horizontal flipping
  2. Vertical flipping
  3. 180 degrees rotation

The image augmentation was carried out in R using the functions *“mirror”* and *“imrotate”* from the *“imager”* package. After obtaining our expanded dataset, we split it using *stratified sampling* into training (70%), validation (15%), and test (15%) sets. The final summary statistics of the data after image augmentations is given below:

|  |  |
| --- | --- |
| **Data** | **Attribute** |
| Total images | 800 |
| Tumor MRI images | 400 |
| Non-tumor MRI images | 400 |
| Sampling method | Stratified sampling |
| Training set | 560 |
| Validation set | 120 |
| Test set | 120 |
| Event rate | 50% |
| Image size | 188 x 232 |

[3] <https://www.kaggle.com/navoneel/brain-mri-images-for-brain-tumor-detection>

An example of image augmentation is depicted below:



1. *Horizontal flipping*
2. *Vertical flipping*
3. *180 degrees rotation*

*Figure 2.1 Image augmentation*

From the above MRI images, we make the following observations about the presence of tumors:

* ***Intensity:*** The tumor has a high color intensity then the rest of the image
* ***Size:*** The high intensity pixels for an image with tumor would he larger in number than an image with no tumor
* ***Centroid:*** The centroid for the pixels with high intensity can be a good indicator of the location of the tumor

Therefore, we use clustering method on these images to create feature sets which would take into account the above information and will then use these feature sets to solve the classification problem.

**III. K-means clustering for feature generation**

1. *Clustering process*

In order to create features like intensity, size, and centroid, we used k-means clustering to cluster the pixels in the MRI images. The process is depicted below:

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| 1 |
| 1 |
| 2 |
| 2 |
| 2 |
| 1 |
| 3 |
| 1 |
| 1 |
| 1 |
| 1 |
| 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 3 | 3 |
| 1 | 3 | 3 | 1 | 1 |
| 2 | 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 2 | 2 |
| 2 | 1 | 1 | 1 | 2 |

Clustered Image

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 3 | 3 |
| 1 | 3 | 3 | 1 | 1 |
| 2 | 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 2 | 2 |
| 2 | 1 | 1 | 1 | 2 |

Visualization for clustered image

Dataframe with cluster information

Original image

…

…

Mean intensity,

cluster size, centroid

Vectorized

Intensity based clustering

*Figure 3.1 Clustering process*

The augmented images were read into R and each image was converted into a matrix of dimensions (height x width of image) containing the intensity values at each pixel. Each of these matrices were then vectorized in order to perform clustering on these images.

**We chose k-means clustering for this purpose in order to:**

* Speed up the process of clustering a total of 800 images and to have faster computations
* Large no. of pixel values (*high n*), hierarchical clustering would require in-memory *n* x *n* x *800* distance matrix

The cluster information from each image was then stored in a dataframe which included:

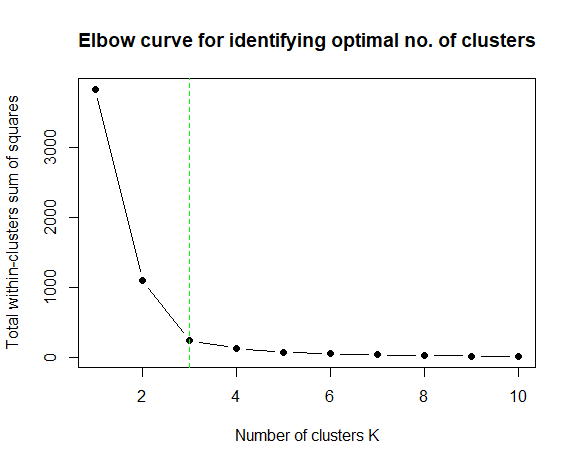
* Mean intensity at each cluster
* Size of each cluster (number of pixels)
* Centroid of each cluster (position in the image: x and y coordinate)
* Presence of tumor (binary: 0 or 1)
* Dataset label (train, validation, or test)

The vectors were then converted back to a matrix with cluster labels as the values and these matrices were then used to visualize the clustered image as depicted in the process above.

1. *Determination of optimal number of clusters*

In order to perform the clustering process efficiently, we use the elbow method to identify the optimal number of clusters. All the images were clustered using numbers of clusters from 1 to 10 and the average of the total within-clusters sum of squares was taken across all the images for each number of clusters. Finally, we plot the mean of total within-clusters sum of squares vs the number of clusters in order to determine the optimal number of clusters.

**Elbow curve for identifying optimal no. of clusters**



*Figure 3.2 Elbow curve for K-means clustering*

***From the elbow curve, we identified the optimal number of clusters as 3.***

A snapshot of the final dataframe obtained using K-means for feature extraction is shown below:

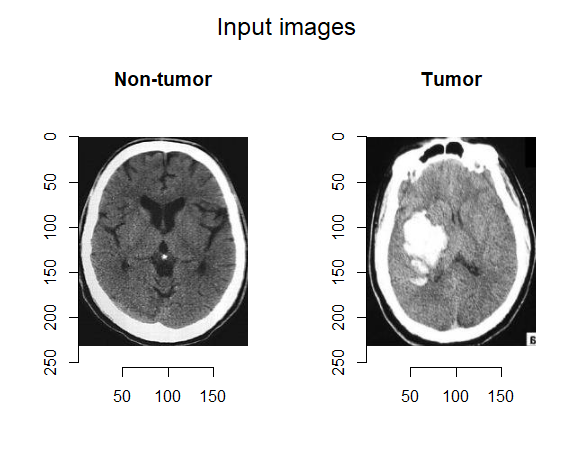
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ﻿data\_set | c1\_size | c2\_size | c3\_size | c1\_intensity |
| train | 17607 | 20955 | 5054 | 0.0348607 |
| train | 16905 | 19751 | 6960 | 0.0655778 |
| train | 13714 | 23038 | 6864 | 0.0463897 |
| train | 10641 | 27462 | 5513 | 0.0217019 |
| train | 14514 | 18560 | 10542 | 0.0409230 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| c2\_intensity | c3\_intensity | ﻿c1\_x\_centroid | c2\_x\_centroid | c3\_x\_centroid |
| 0.3471991 | 0.9332255 | 95.55342 | 93.76683 | 93.87000 |
| 0.3442183 | 0.9425208 | 98.41083 | 92.93332 | 89.44698 |
| 0.3662888 | 0.9529098 | 94.07255 | 94.85298 | 94.16929 |
| 0.4386409 | 0.9407041 | 90.59261 | 95.80537 | 95.53945 |
| 0.3548802 | 0.9439750 | 92.02039 | 95.83109 | 95.57039 |

|  |  |  |  |
| --- | --- | --- | --- |
| c1\_y\_centroid | c2\_y\_centroid | c3\_y\_centroid | tumor |
| 108.5343 | 122.3085 | 120.1672 | 0 |
| 104.9151 | 123.3934 | 125.0761 | 0 |
| 114.7050 | 116.6865 | 119.4605 | 0 |
| 109.5807 | 117.9369 | 122.6980 | 0 |
| 112.0284 | 121.7142 | 113.4764 | 0 |

*Figure 3.3 Clustering features snapshot*

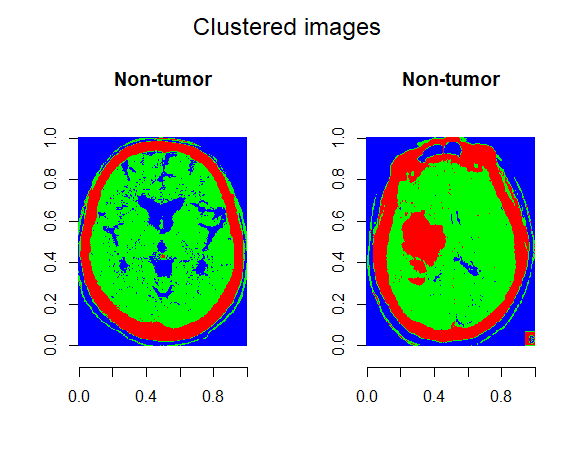
An example of the input and the clustered image for tumor and non-tumor MRI scans is as follows:



**Input images**

**Non-tumor**

**Tumor**



**Non-tumor**

**Tumor**

**Clustered images**

*Figure 3.4 Visualization of clustered images*

**IV. XGBoost for Classification**

After obtaining the features and labels, we now train an eXtreme Gradient Boosting (XGBoost) model to classify images based on the presence of brain tumor. XGBoost is a popular and efficient open-source implementation of the gradient boosted trees algorithm.

**We choose an XGBoost model for this purpose because:**

* This is a purely prediction problem. Our target is to make better predictions without analyzing the process of making these predictions
* XGBoost is known to produce best predictions in terms of accuracy. In healthcare problems, we need to be as accurate as possible in order to serve the patients better
* Since we are dealing with large number of images, we need to choose an algorithm which is hardware efficient. Being a powerful and fast machine learning library, XGBoost pushes the limit of computations resources to optimize the use of computer hardware

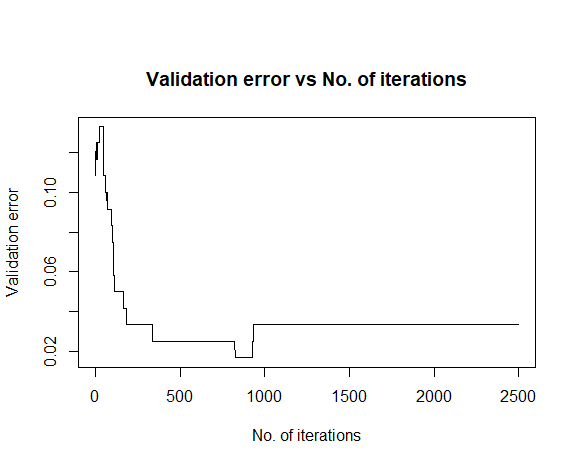
1. *Cross validation for no. of iterations*

One critical parameter which needs to be tuned in XGBoost is the number if iterations. We set the depth of trees to have sufficient interactions and the learning rate to be a low value and tune the number of iterations using cross validation. For this purpose, we trained the data using the training set and plot the validations error vs the number of iterations using the following hyperparameters:

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Learning rate | 0.01 |
| Maximum depth of tree | 6 |
| Maximum iterations | 2500 |
| Objective | binary:logistic |
| Training observations | 560 |
| Validation obervations | 120 |

*Figure 4.1 Hyperparameters for cross validating XGBoost*

The plot of validation error vs the number of iterations is depicted below:



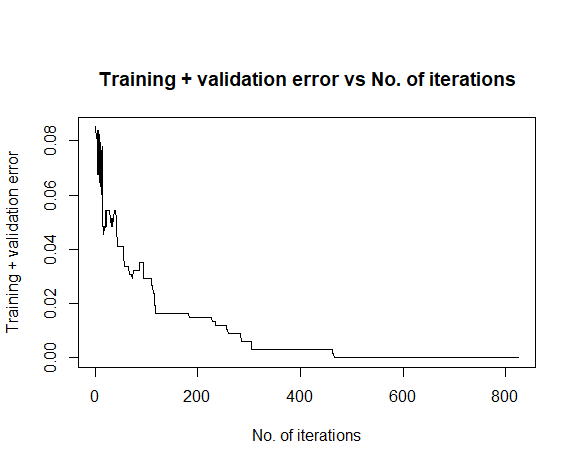
**Validation error vs no. of iterations**

*Figure 4.2 Cross validating no. of iterations*

***From this plot we observe that the minimum validation error occurs at 825 iterations***

1. *Model training using cross validated hyperparameters*

We now retrain the model using the combination of training and validation sets and the optimum number of iterations identified using cross validation (825). The rest of the hyperparameter were set to be the same as mentioned in Figure 4.1. This time, we plot the training error vs. the number of iterations to monitor our XGBoost training process.



**Training + validation error vs no. of iterations**

*Figure 4.3 Training + validation error vs no. of iterations*

**V. Model performance and results**

Finally, we make predictions on our training + validation set and test set and analyze their performance. The confusion matrices and results for both the sets are given below:

1. *Training set*

Training accuracy = 100%

Training Sensitivity = 1.0

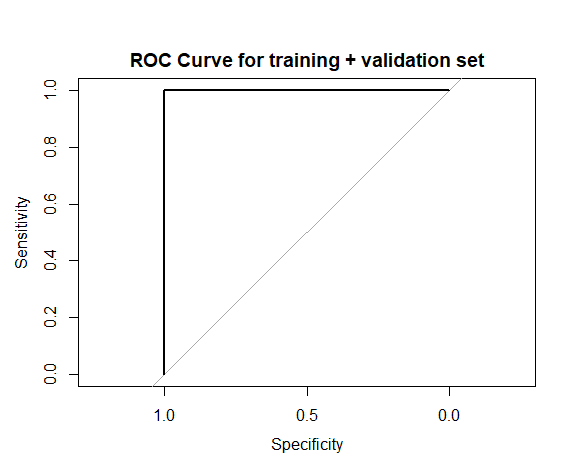
Training Specificity = 1.0

Confusion Matrix:

|  |  |  |
| --- | --- | --- |
| N = 680 | Actual: No | Actual: Yes |
| Predicted: No | 340 | 0 |
| Predicted: Yes | 0 | 340 |

*Figure 5.1 Confusion matrix for training + validation set*

**ROC Curve for training + validation set**



AUC = 1.0

*Figure 5.2 ROC curve for training + validation set*

1. *Test set*

Test accuracy = 99.17%

Test Sensitivity = 0.9833

Test Specificity = 1.0

Confusion Matrix:

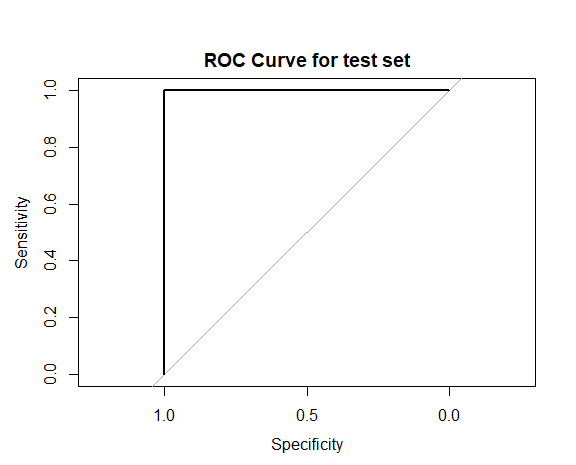
|  |  |  |
| --- | --- | --- |
| N = 120 | Actual: No | Actual: Yes |
| Predicted: No | 59 | 0 |
| Predicted: Yes | 1 | 60 |

*Figure 5.2 Confusion matrix for test set*

**ROC Curve for test set**

AUC = 1.0

*Figure 5.3 ROC curve for test set*



The XGBoost model, known for its high predictive power is able to produce an accuracy of 99.17% on the test set.

**VI. Discussion and Conclusions**

1. *Discussion*

The model performance and results, at first, look too good to be true. An accuracy of 100% on the training set and 99.17% on the test seem very high. But it is important to note these numbers are usually dependent upon the context and the use case. Brain tumors are almost always identifiable with the naked eye and therefore, observing the same using machine learning is not a very difficult task. The actual value being created here is the automation of the process which saves a lot of time and a high accuracy is just a byproduct. The amount of time required to identify tumors in a large number of MRI images can now be reduced to a few seconds using this model. Moreover, the accuracy metrics indicate that the classification is indeed very reliable.

1. *Conclusions and future scope*

From this study, we observed the application of K-means clustering on image data which is very unique. Interestingly, one can infer that the features generated manually for using cluster analysis are similar to the ones generated using Convolutional Neural Networks which extracts image features at both micro and macro levels. In the case of MRI images, we are almost certain about the features which can be used to identify the tumors like cluster centroid, cluster size and intensity and therefore, we are able to create these features manually using K-means. This helps in reducing prediction time and makes the prediction process more intuitive than CNNs. In more complex applications like MRI of Alzheimer's disease, more complicated techniques and CNNs might be required for the classification problem but it would be interesting to observe the contribution of clustering technique in image feature extraction. Moreover, we conclude that eXtreme Gradient Boosting provides us with high prediction powers which cannot be obtained very easily using linear models.

**VII. References**

* T. M. Shahriar Sazzad, K. M. Tanzibul Ahmmed, M. U. Hoque and M. Rahman, "Development of Automated Brain Tumor Identification Using MRI Images," 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), Cox'sBazar, Bangladesh, 2019, pp. 1-4
* W. Chen, X. Qiao, B. Liu, X. Qi, R. Wang and X. Wang, "Automatic brain tumor segmentation based on features of separated local square," 2017 Chinese Automation Congress (CAC), Jinan, 2017, pp. 6489-6493.
* Z. Zulkoffli and T. A. Shariff, "Detection of Brain Tumor and Extraction of Features in MRI Images Using K-means Clustering and Morphological Operations," 2019 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS), Selangor, Malaysia, 2019, pp. 1-5.

**VIII. Resources**

* Github (with code and data): <https://github.com/skandupmanyu/Brain_Tumor_Identification_Kmeans_XGBoost>