

# Proiect – Etapa 1

## 1. Cerinta 1

Am definit clasa “BrainTumorDataset”, ce contine constructorul “\_\_init\_\_” si metodele “\_\_len\_\_” si “\_\_getitem\_\_”.

### 1.1. \_\_init\_\_

In dataset-uri am decis sa retin ca argumente calea catre director si doua liste, ce contin calea catre imagini si label-urile corespunzatoare (0 pentru “glioma\_tumor”, 1 pentru “meningioma\_tumor”, 2 pentru “no\_tumor” si 3 pentru “pituitary\_tumor”). Fac acest lucru astfel: iterez prin fiecare folder al directorului dat ca argument si atasez calea imaginii si label-ul corespunzator in cele 2 liste.

Am implementat o functionalitate in plus, care este specifica cerintei 3, si va fi discutata in cadrul capitolului corespunzator.

### 1.2. \_\_len\_\_

Returneaza numarul total de imagini din dataset.

### 1.3. \_\_getitem\_\_

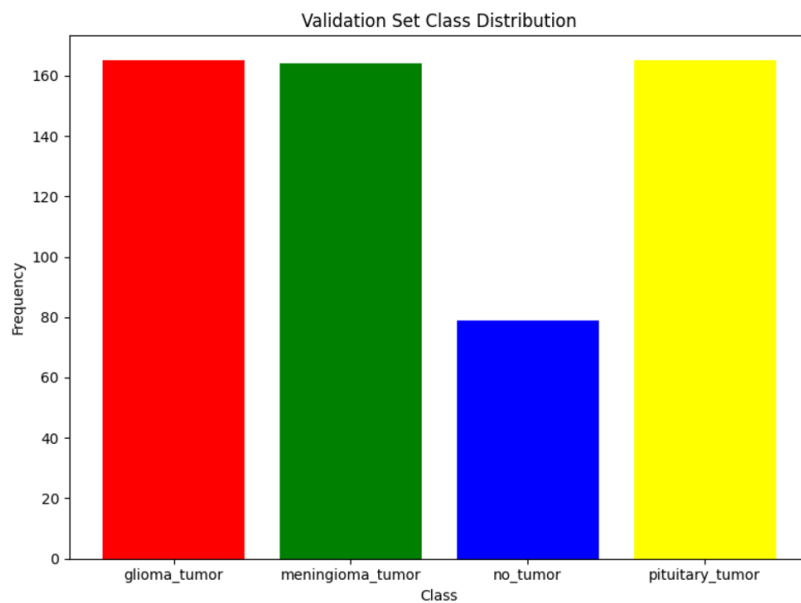
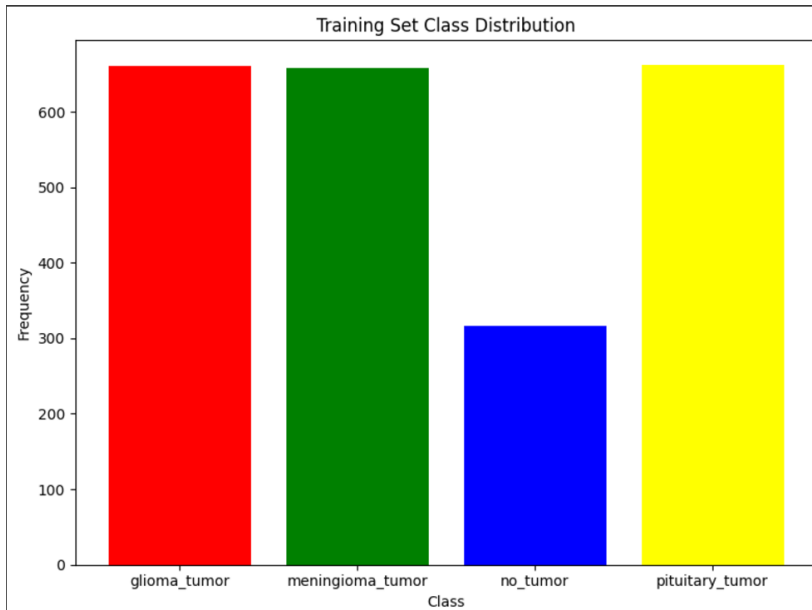
Incarca si returneaza imaginea si label-ul corespunzator pe baza pozitiei in lista de fisiere. Aici am implementat, de asemenea, si posibilitatea de a afisa imaginile, avand aplicate diversele transformari, cerute in viitoarele cerinte. Acestea vor fi discutate in cadrul cerintelor corespunzatoare (Cerintele 5 si 6).

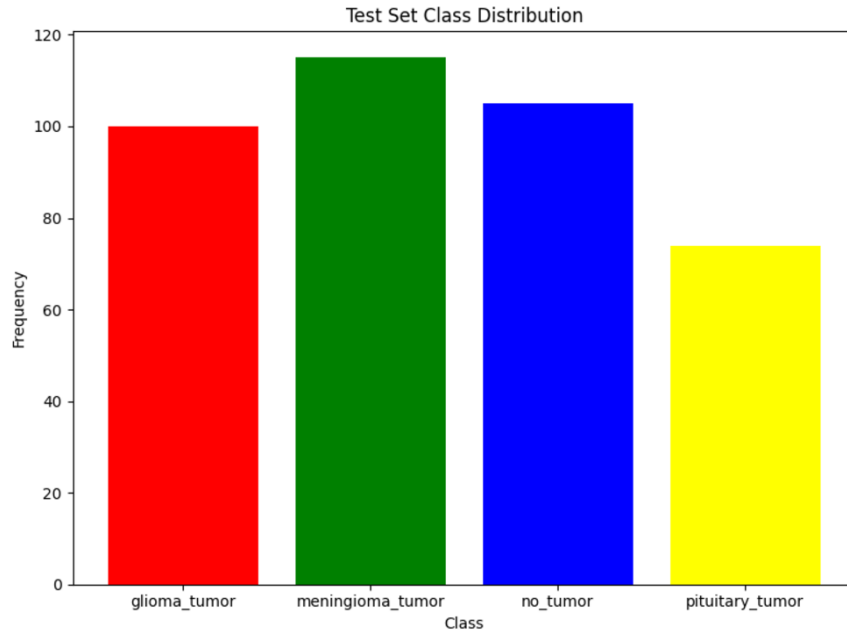
## 2. Cerinta 2

Am definit functia “split\_dataset” si i-am dat ca argumente datasetul ce trebuie impartit, proportiile pentru antrenare (80%) si validare (20%) si optiunea de a amesteca datasetul de validare sau nu (shuffle = False). Am considerat ca este important sa amestec intotdeauna setul de date de antrenare. Pentru impartirea efectiva, iterez prin tipurile de tumori (0, 1, 2, 3) din dataset, dupa care colectez toti indicii care respecta acel tip de tumora, calculez numarul de imagini necesar pentru setul de validare (20% din numarul total de indici selectati) si selectez in mod aleator din indicii pentru a reprezenta noul meu dataset de validare, iar restul indicilor vor reprezenta datasetul de antrenare. Dupa ce creez dataseturile corespunzatoare, le atasez argumentele necesare: calea catre director si cele 2 liste pentru calea catre imagine si label-ul. La final, functia returneaza dataseturile de antrenare si validare. In cadrul acestei cerinte creez si DataLoader-ele corespunzatoare dataseturilor.

### 3. Cerinta 3

Am definit functii “visualize\_class\_distribution” ce are ca argumente setul de date si titlul histogramei. Histogramele sunt realizate pe baza numarului de label-uri, iar initial, am obtinut urmatoarele rezultate:

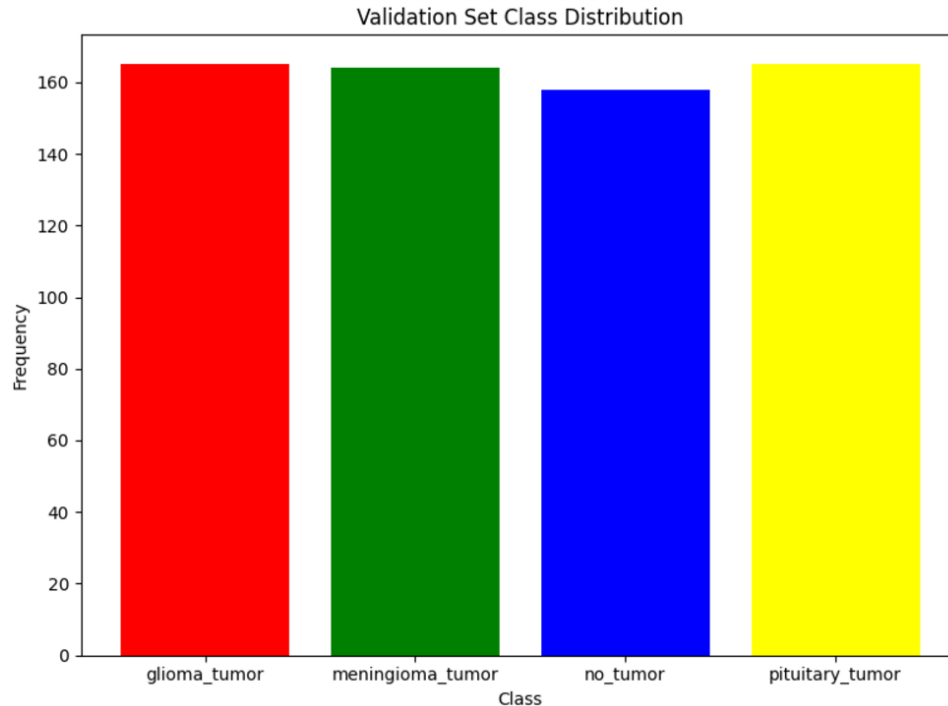




Am observat ca este subreprezentata clasa “no\_tumor” a dataseturilor de antrenare si validare, iar solutia pe care am propus-o a fost sa dublez numarul de label-uri. Am implementat acest lucru in constructorul clasei “BrainTumorDataset” astfel: Deoarece setul de antrenare si validare au in comun directorul “Training”, am pus conditia ca daca calea catre directorul de baza este “Training”, selectez indicii clasei no\_tumor si dublez listele cu label-uri si cai catre imagini cu ajutorul indicilor.

Rezultatul obtinut dupa aceasta implementare:





Datasetul de testare va ramane neschimbat, intrucat clasa “pituitary\_tumor” nu este suficient de sub-reprezentata pentru a face modificari.

Echilibrul setului de date este foarte important deoarece ne ajuta sa evitam clasificari eronate si ajuta la vizualizarea unor sabloane esentiale in modul de antrenare, intrucat vrem sa ne marim sansele de a nu diagnostica un MRI cu un fals negativ.

#### 4. Cerinta 4

Aceasta cerinta a fost implementata in functii “show\_image”, ce primeste ca argument un dataset si alege primele 5 imagini din dataset, in functii de tipul de tumora. Deoarece amestec si selectez aleator imaginile, daca rulez cea de a doua cerinta din nou, atunci se vor afisa alte seturi de imagini. Daca se doreste afisarea acelorlasi imagini de fiecare data, se poate face acest lucru folosind functia “\_\_getitem\_\_”.

Inainte de a discuta imaginile, voi clasifica fiecare tip de tumora si voi prezenta si cateva caracteristici ale acestora:

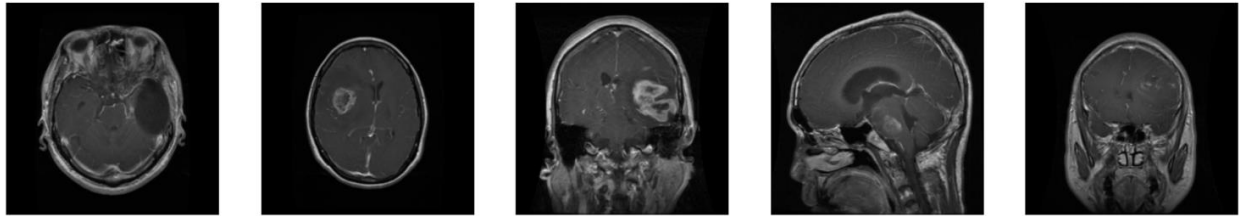
4.1. Tumorile gliom se formeaza din celule gliale si se pot localiza **oriunde in creier sau in maduva spinarii**.

4.2. Tumorile meningioame se dezvoltă din meninge, membranele care invellesc creierul si maduva spinarii si se localizeaza aproape de **maduva spinarii sau suprafata creierului**.

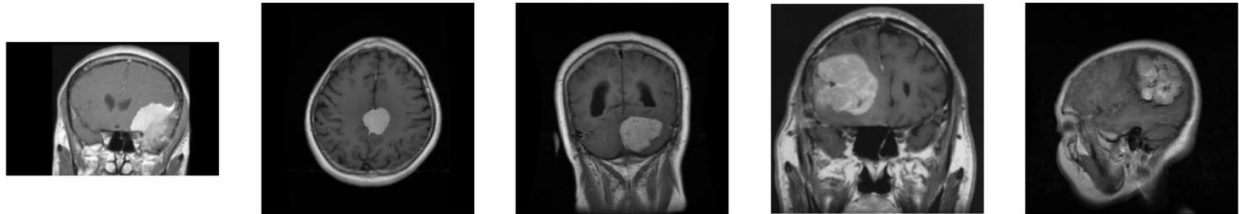
4.3. Tumorile pituitare se dezvoltă in glanda pituitara, responsabila pentru reglarea hormonilor organismului si se localizeaza la **baza creierului**.

Acestea sunt rezultatele obtinute la ultima rulare:

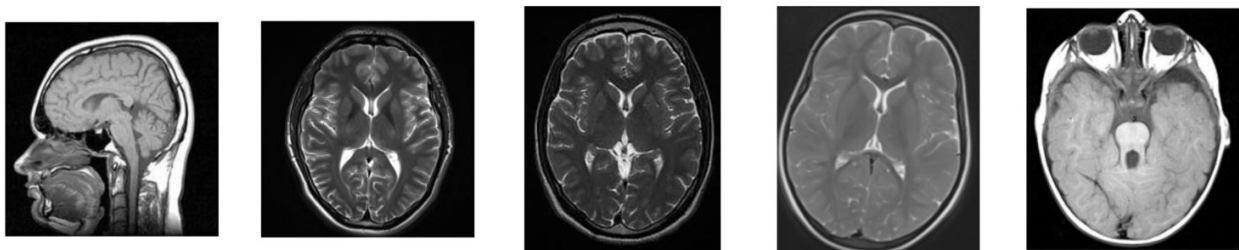
glioma\_tumor



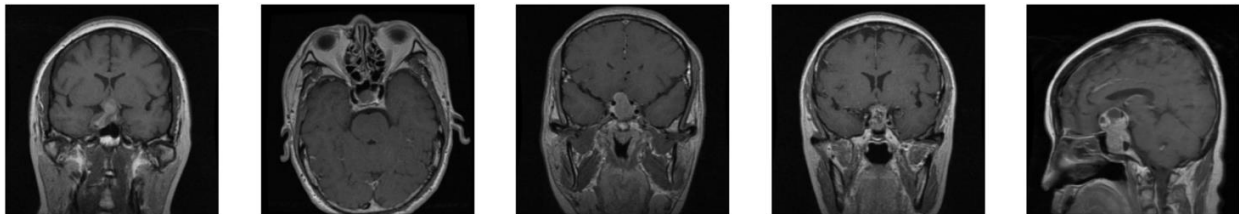
meningioma\_tumor



no\_tumor



pituitary\_tumor



În urma acestor imagini constat că tumoarea pituitară se diferențiază foarte mult de celelalte două tipuri de tumori, deși este mai greu de văzut cu ochiul liber, însă, cu ajutorul transformărilor implementate în cerința 5 și 6, acestea sunt vizibile și ușor de distins.

Cel mai greu de distins pare a fi tumora gliom, deoarece poate fi găsită în 2 locuri diferite, iar, din imaginile obținute, se observă o diferență în culoare în cazul tumorilor prezente în creier, datorită naturii tumorii (benignă sau malignă).

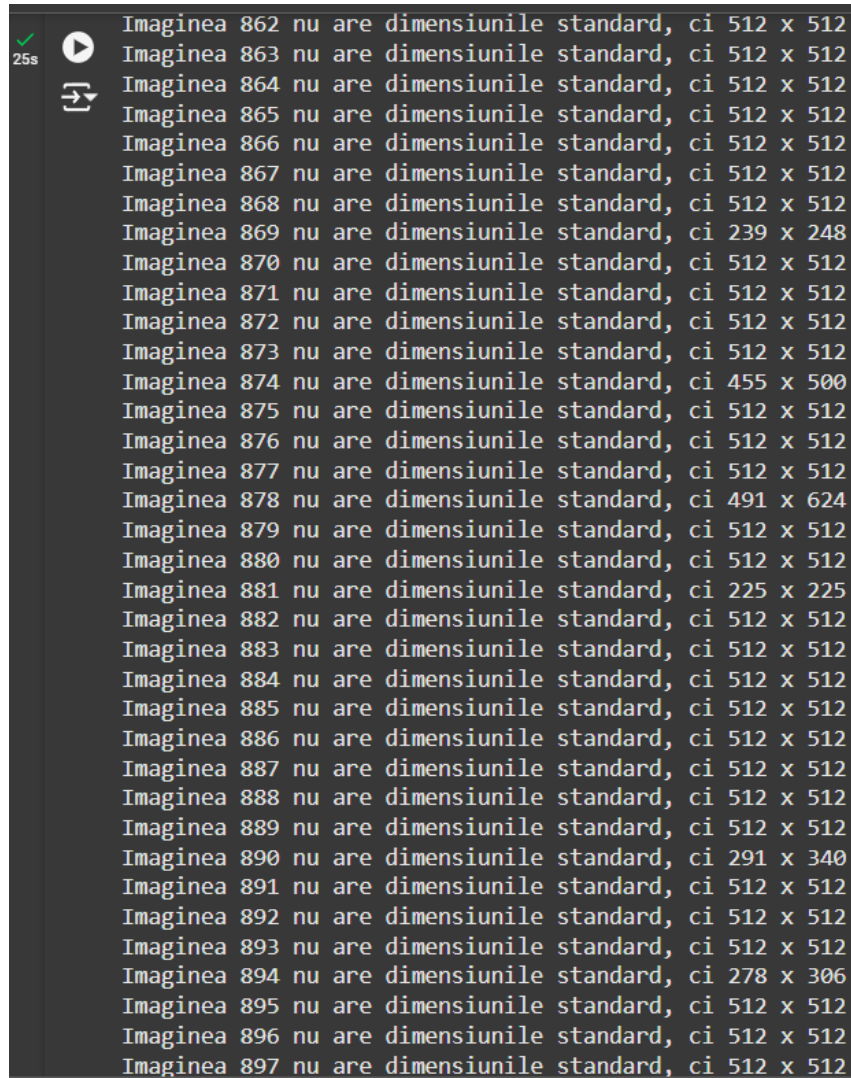
O problemă de distingere ar putea apărea între tumorile gliom din creier și meningioame, întrucât ambele se găsesc în creier și pot avea localizări și forme foarte similare. În imaginile afișate de mine se observă o asemănare foarte mare între cea de a doua poză de la tumorile gliom și cea de a doua poză de la tumorile meningioame, însă există o mică diferență, anume că cea de a doua este mai plină, pe când prima este mai goală înăuntru.

Cu toate acestea, in cazul tumorilor meningioame si gliom, este destul de greu un diagnostic fals negativ, insa, cu ochiul liber, e mai usor de diagnosticat o tumora pituitara ca un fals negativ, ceea ce nu este dorit. Ar putea fi usor de diagnosticat ca fals negativ si tumorile gliom din maduva spinarii, intrucat nu sunt in creier, iar zona respectiva este destul de aglomerata si greu de distins pentru modelul de antrenare, in comparatie cu creierul, care ar trebui sa aibe o suprafata consistenta in culoare.

## 5. Cerinta 5

In cadrul acestei cerinte am implementat 2 functii:

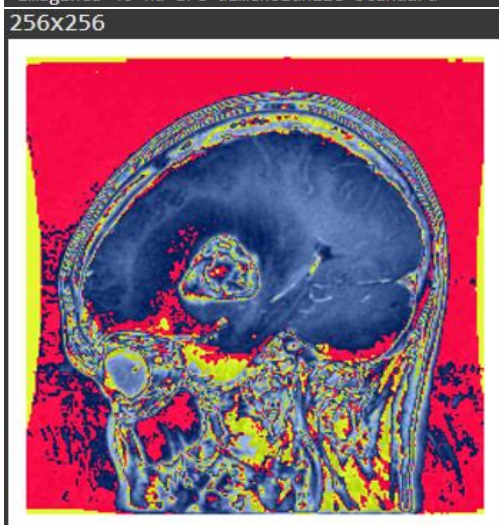
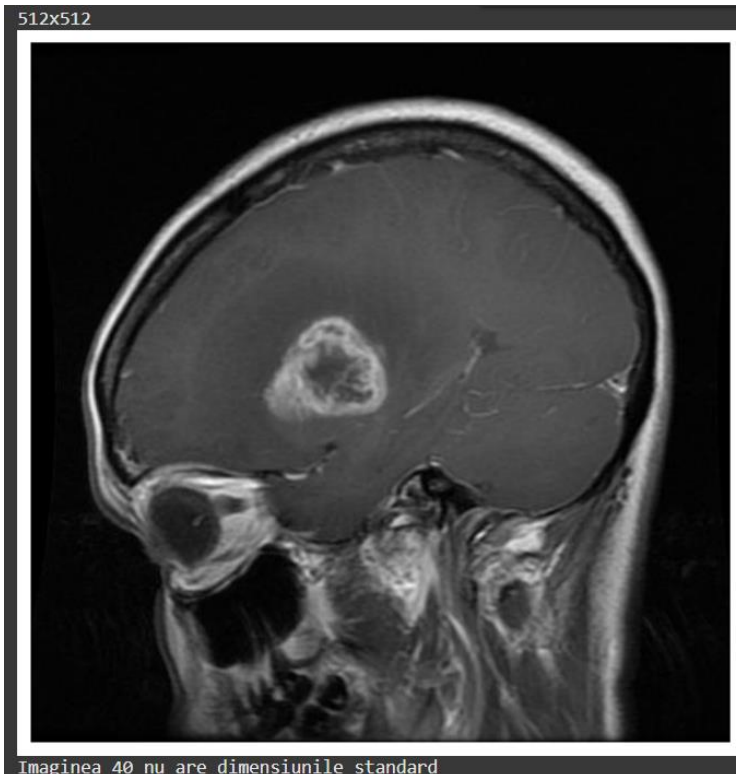
5.1. "verify\_all", ce verifica ca toate imaginile din setul de date dat ca argument sa respecte criteriile mentionate in cerinta. In urma rularii acestei functii, am constatat ca toate imaginile sunt RGB si au valorile pixelilor in parametrii normali, insa sunt foarte multe care au dimensiuni variabile. O parte din output-ul acestei functii:



```
Imaginea 862 nu are dimensiunile standard, ci 512 x 512
Imaginea 863 nu are dimensiunile standard, ci 512 x 512
Imaginea 864 nu are dimensiunile standard, ci 512 x 512
Imaginea 865 nu are dimensiunile standard, ci 512 x 512
Imaginea 866 nu are dimensiunile standard, ci 512 x 512
Imaginea 867 nu are dimensiunile standard, ci 512 x 512
Imaginea 868 nu are dimensiunile standard, ci 512 x 512
Imaginea 869 nu are dimensiunile standard, ci 239 x 248
Imaginea 870 nu are dimensiunile standard, ci 512 x 512
Imaginea 871 nu are dimensiunile standard, ci 512 x 512
Imaginea 872 nu are dimensiunile standard, ci 512 x 512
Imaginea 873 nu are dimensiunile standard, ci 512 x 512
Imaginea 874 nu are dimensiunile standard, ci 455 x 500
Imaginea 875 nu are dimensiunile standard, ci 512 x 512
Imaginea 876 nu are dimensiunile standard, ci 512 x 512
Imaginea 877 nu are dimensiunile standard, ci 512 x 512
Imaginea 878 nu are dimensiunile standard, ci 491 x 624
Imaginea 879 nu are dimensiunile standard, ci 512 x 512
Imaginea 880 nu are dimensiunile standard, ci 512 x 512
Imaginea 881 nu are dimensiunile standard, ci 225 x 225
Imaginea 882 nu are dimensiunile standard, ci 512 x 512
Imaginea 883 nu are dimensiunile standard, ci 512 x 512
Imaginea 884 nu are dimensiunile standard, ci 512 x 512
Imaginea 885 nu are dimensiunile standard, ci 512 x 512
Imaginea 886 nu are dimensiunile standard, ci 512 x 512
Imaginea 887 nu are dimensiunile standard, ci 512 x 512
Imaginea 888 nu are dimensiunile standard, ci 512 x 512
Imaginea 889 nu are dimensiunile standard, ci 512 x 512
Imaginea 890 nu are dimensiunile standard, ci 291 x 340
Imaginea 891 nu are dimensiunile standard, ci 512 x 512
Imaginea 892 nu are dimensiunile standard, ci 512 x 512
Imaginea 893 nu are dimensiunile standard, ci 512 x 512
Imaginea 894 nu are dimensiunile standard, ci 278 x 306
Imaginea 895 nu are dimensiunile standard, ci 512 x 512
Imaginea 896 nu are dimensiunile standard, ci 512 x 512
Imaginea 897 nu are dimensiunile standard, ci 512 x 512
```

Pentru a afisa toate imaginile care nu respecta dimensiunea de 256x256, este necesar un timp de executie de 25 de secunde, ceea ce este destul de mult in comparative cu timpul de executie al celorlalte functii (0-2 secunde). Din acest motiv am comentat apelul acestei functii. Pentru a testa veridicitatea informatiilor observate de mine, se poate decomenta linia de cod care apeleaza aceasta functie si se poate rula de la capat rezolvarea pentru aceasta cerinta. Din cauza multitudinii de imagini care s-ar fi afisat, am ales sa afisez doar informatiile relevante cerintei pentru tot setul de date.

5.2. “verify\_one”, ce verifica ca imaginea de pe pozitie “idx” din setul de date dat ca argument sa respecte criteriile mentionate in cerinta. Am luat la intamplare imaginea cu  $idx = 40$  din setul de date de antrenare si am obtinut urmatorul rezultat:



Am ales sa afisez imaginea originala si cea modificata pentru a discuta diferentele.

In primul rand, avantajele pe care le pot deduce din aceasta transformare sunt ca, deoarece majoritatea imaginilor sunt mai mari, imaginile redimensionate reduc costul computational, accelerand viteza de antrenare, iar faptul ca existau imagini care aveau dimensiuni diferite fata de celelalte putea face modelul sa invete mai greu.

In al doilea rand, din toate criteriile mentionate in cerinta, s-au aplicat doar redimensionarea si normalizarea. Normalizarea m-a ajutat sa observ mai bine adancimea imaginii, lucru care este important in detectia tumorilor. Dupa cum se vede in poza finala, tumoarea este mult mai vizibila si se distinge foarte bine de restul creierului, fiind vizibila dimensiunea si forma acesteia mult mai clar in comparative cu imaginea originala.

Toate verificarile si transformarile pentru aceasta cerinta au fost facute in cadrul functiei “\_\_getitem\_\_” astfel: am dat ca parametru o lista de string-uri “more\_transforms”, iar daca aceasta contine “integrity”, verifica daca imaginea e RGB, daca pixelii sunt intr-un interval asteptat si daca are dimensiunea de 256x256, dupa care aplic transformarile necesare imaginii si returnez imaginea modificata si label-ul. Am considerat ca pixelii imaginilor sa nu aibe valori negative sau peste 255, sau ca imaginile sa nu fie prea intunecate (sa aibe pixeli ce au valoarea maxima 30) in momentul in care am verificat scala pixelilor manual, insa normalizarea a ajutat la uniformizarea imaginii, dupa cum am discutat anterior.

## 6. Cerinta 6

In cadrul acestei cerinte am implementat functiile:

6.1. “image\_processing”, ce primeste ca argument pozitia in lista de fisiere a unei imagini (idx) si returneaza imaginea originala si alte 5 imagini pe care am aplicat separat cele 5 operatii de preprocesare, pentru a observa mai bine efectele.

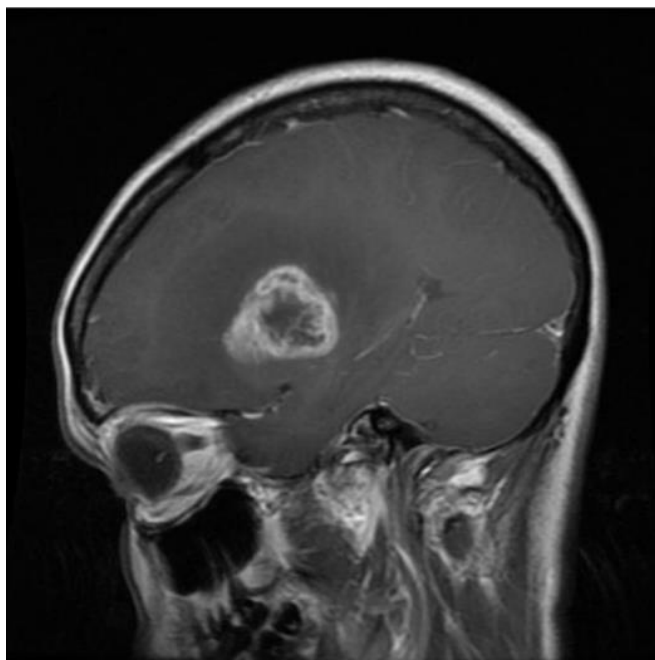
6.2 “image\_processing\_all”, ”, ce primeste ca argument pozitia in lista de fisiere a unei imagini (idx) si returneaza imaginea originala, impreuna cu alte 4 imagini in care aplic, pe rand, fiecare operatie.

Transformarile au fost implementate folosind functia “\_\_getitem\_\_”, cu ajutorul argumentului “more\_transforms”, folosindu-ma de biblioteca OpenCV.

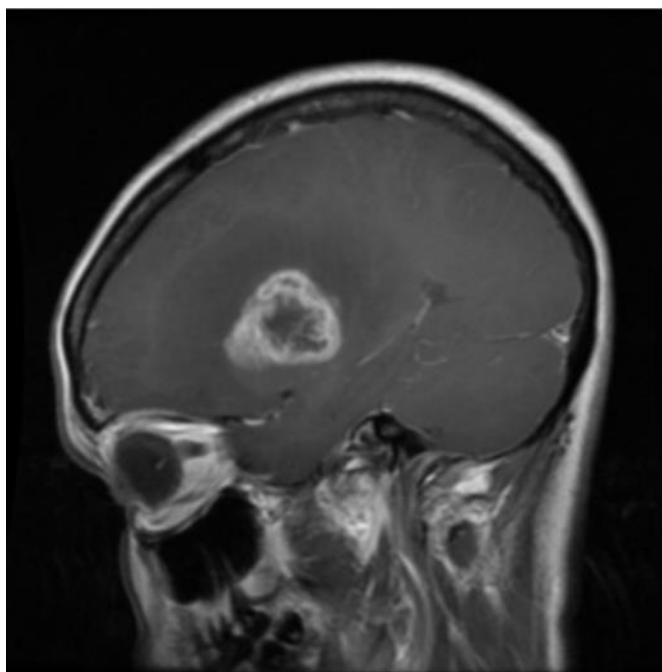
Voi discuta, pe rand, fiecare operatie si efectele pe care aceasta le-a avut asupra imaginii originale.

### 6.3. Imaginea Originala:



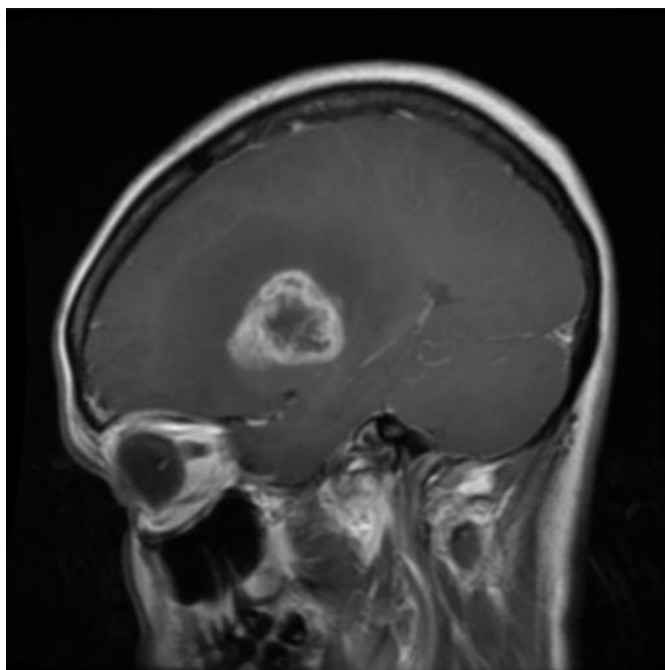


6.4. Imaginea cu filtru Gaussian:



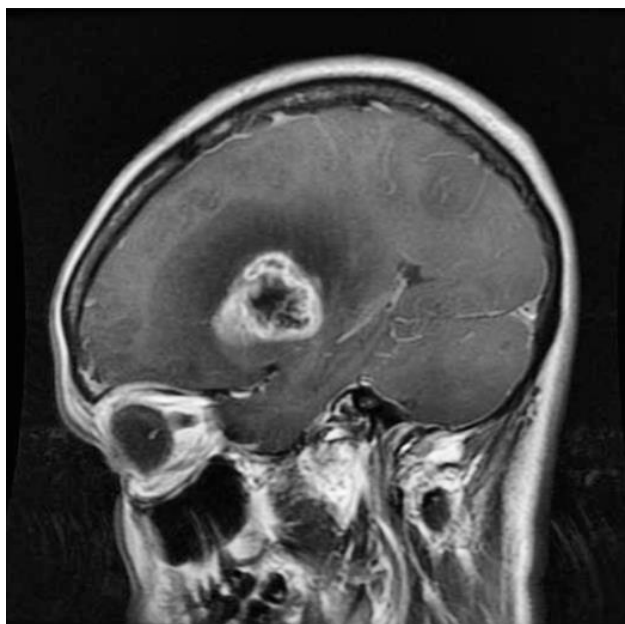
Aceasta transformare a avut ca effect netezirea imaginii si reducerea zgomotului si este utila deoarece elimina detaliile mai putin relevante, pastrand informatiile critice din imagine, imbunatatind astfel performanta modelului de antrenare.

6.5. Imaginea cu filtru Gaussian, normalizata:



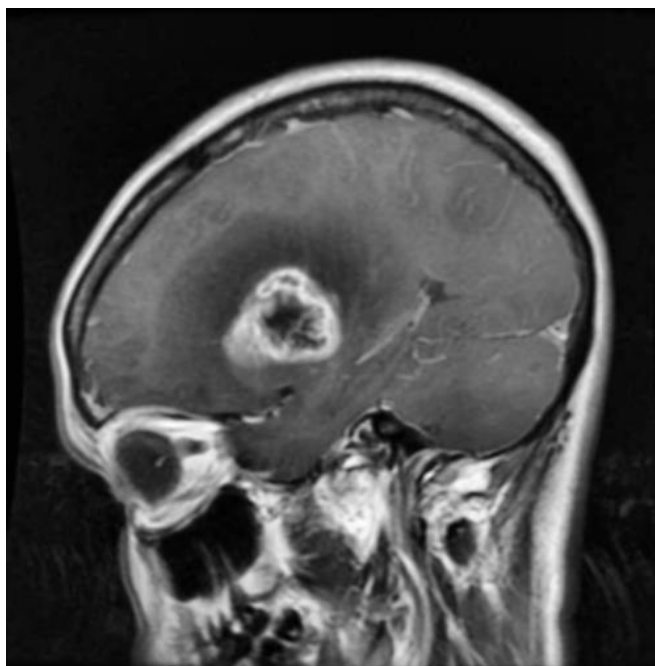
Despre efectele si beneficiile normalizarii am discutat la cerinta anterioara, in sa de aceasta data am folosit normalizarea furnizata de biblioteca OpenCV, ce are ca scop scalarea intensitatii globale a imaginii, folosindu-se de scalarea MINMAX.

6.6. Imaginea cu filtru CLAHE:



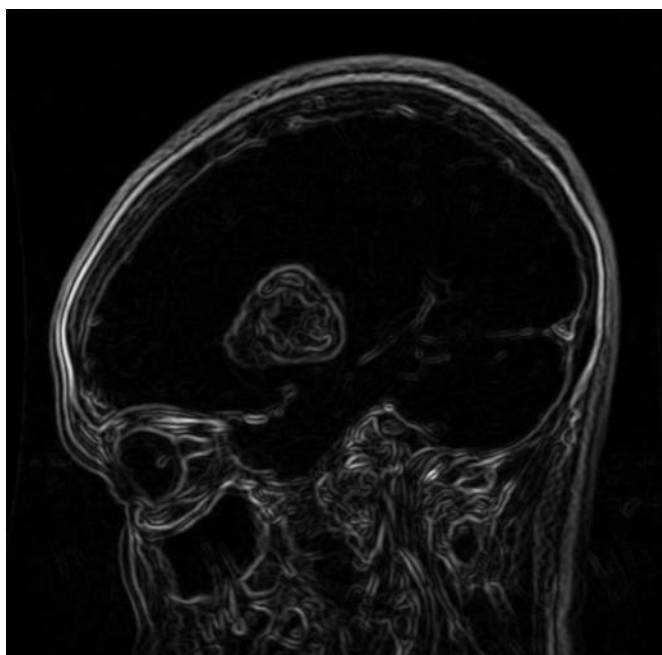
Efectul acestui filtru a fost de a imbunatatii contrastul si vizibilitatea tumorii.

6.7. Imaginea cu filtru Gaussian si CLAHE, normalizata:



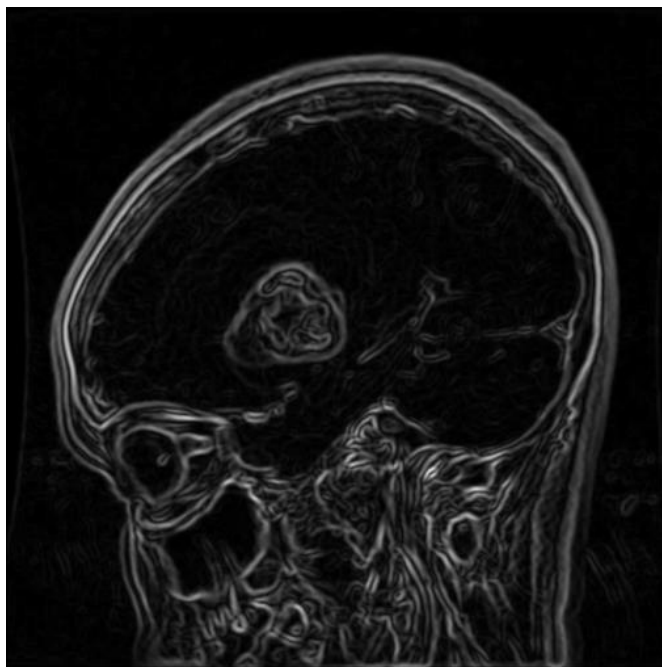
Se poate observa cum filtrul Gaussian a blurat, de exemplu, detaliile creierului, iar filtrul CLAHE a imbunatatit contrastul dintre tumora si creier, astfel imbunatatind sansele modelului de a invata si a detecta tumori.

6.8. Imaginea cu filtru Sobel:



Filtrul acesta a avut rolul de a evidentia marginile, delimitand foarte bine componentele anatomice, fiind mult mai clara prezenta tumorii.

6.9. Imaginea cu filtru Gaussian, CLAHE si Sobel, normalizata:



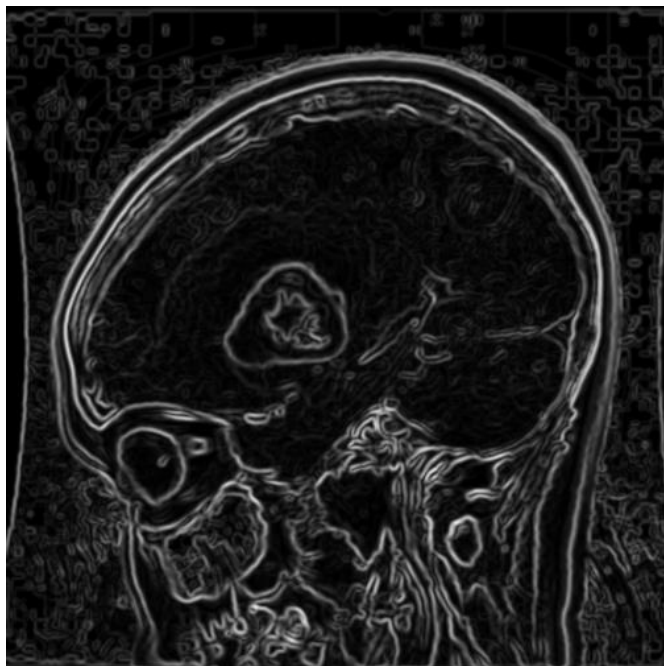
Marginile sunt acum mult mai luminoase (datorita filtrului CLAHE) si mai ingrosate (datorita filtrului Gaussian), fiind mult mai clara delimitarea dintre componentele anatomice.

6.10. Imaginea egalizata:



Acest filtru distribuie uniform intensitatea pixelilor si imbunatateste contrastul global, fiind util in cazul tumorilor ce sunt greu de diferentiat de fundal.

6.9. Imaginea cu filtru Gaussian, CLAHE si Sobel, normalizata si egalizata:



Se observa cum, in ciuda zgomotului din afara capului, marginile sunt mult mai bine definite si mai luminoase.