Brain Tumor Recognition

Work done by:

- Luís Silva 88888
- Pedro Amaral 93283
- Pedro Tavares 93103

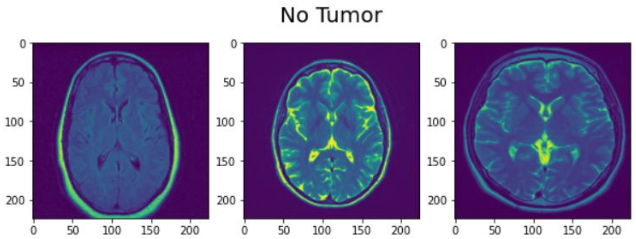
Aprendizagem Automática

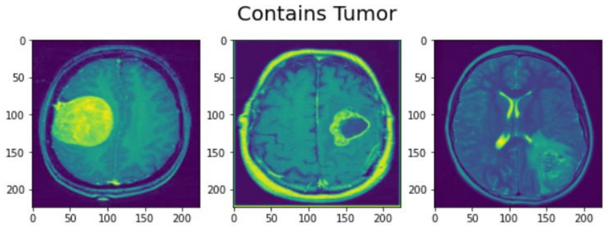
Introduction

- Brain tumor detection is a process performed every day by specialized doctors.
- It is also a classification problem whose objective is to analyze the presence or absence of tumors in the brain through various imaging techniques.

Data Set Analysis

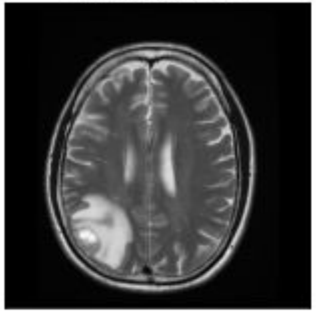
- Our data set contains 253 brain magnetic resonance images.
- They are divided into 2 classes: “No Tumor”, “Contains Tumor”.
- Our dataset contains more images from the class “Contains Tumor”.

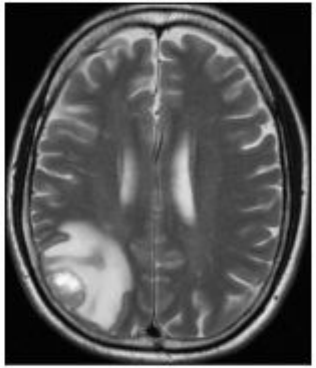




Pre-processing Data

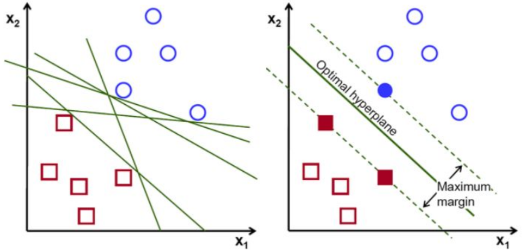
- Read in Grayscale
- Cropping
- Resizing
- Normalizing
- Shuffling





Support Vector Machine

- Supervised learning model
- Hyperplane acts as decision boundary for classification
- Adjustable parameters
    - C - Significance given to the misclassification of an example
    - Gamma - How fast similarity drops to 0
    - Kernel - Mathematical function that maps lower-dimensional data into higher dimensional-data



C	0.1	1	10	100
Gamma	0.0001	0.001	0.1	1
Kernel	rbf	poly	linear	

Convolutional Neural Network

- Deep supervised learning model with multiple layers.
- Feature extraction from groups of pixels through convolutional layers and pooling layers.
- Dropout to avoid overfitting.

```
Layer (type)                Output Shape              Param #
=================================================================
zero_padding2d (ZeroPadding  (None, 228, 228, 3)      0
2D)

activation (Activation)     (None, 228, 228, 3)       0

conv2d (Conv2D)             (None, 225, 225, 64)      3136

max_pooling2d (MaxPooling2D  (None, 56, 56, 64)       0
)

dropout (Dropout)           (None, 56, 56, 64)        0

conv2d_1 (Conv2D)           (None, 53, 53, 128)       131200

max_pooling2d_1 (MaxPooling  (None, 13, 13, 128)      0
2D)

flatten (Flatten)           (None, 21632)             0

dense (Dense)               (None, 512)               11076096

dense_1 (Dense)             (None, 2)                 1026

=================================================================
Total params: 11,211,458
Trainable params: 11,211,458
Non-trainable params: 0
_________________________________________________________________
```

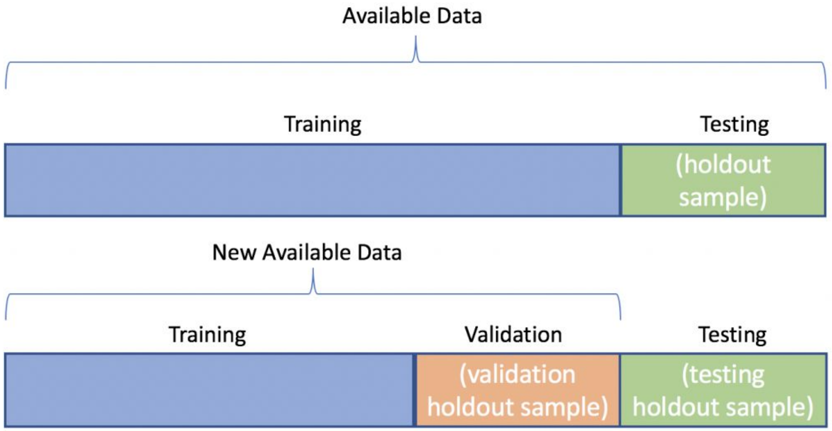
Transfer Learning

- Transfer Learning uses a trained model on a similar dataset.
- We used the VGG16 model as part of our model.
- The layers belonging to VGG16 were set as not trainable since training them led to a worse performance.

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 7, 7, 512)         14714688

 global_average_pooling2d (G  (None, 512)              0
 lobalAveragePooling2D)

 dropout (Dropout)           (None, 512)               0

 dense (Dense)               (None, 4096)              2101248

 dense_1 (Dense)             (None, 4096)              16781312

 dense_2 (Dense)             (None, 2)                 8194

=================================================================
Total params: 33,605,442
Trainable params: 18,890,754
Non-trainable params: 14,714,688
_________________________________________________________________
```

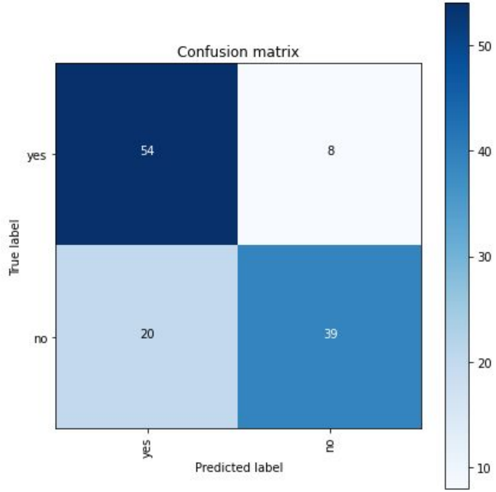
Model Training

- Training, validation and test set.
- Data Augmentation.
- Oversampling and undersampling.



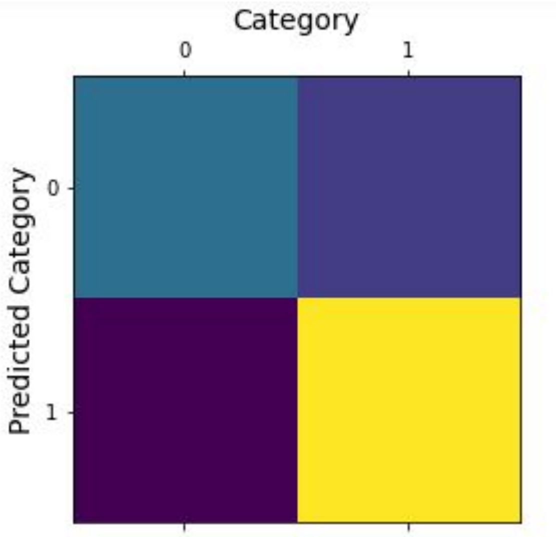
Support Vector Machine (Result)

```
Test Accuracy = 0.77
```



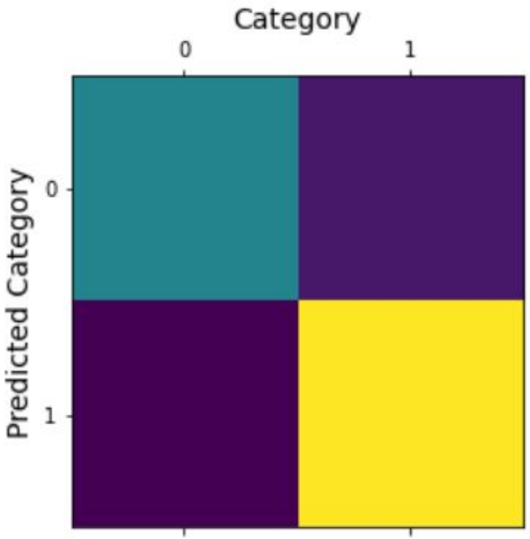
Train Accuracy	0.77
Test Accuracy	0.77
Precision	0.78
Recall	0.77
F1 Score	0.77
Kernel	rbf (Gaussian radial basis function)
C	1
Gamma	0.001

Convolutional Neural Network (Result)



Train Accuracy	0.9007
Validation Accuracy	0.7255
Test Accuracy	0.8824
Train Loss	0.3005
Validation Loss	0.7022
Test Loss	0.4898
Precision	0.92
Recall	0.88
F1 Score	0.89

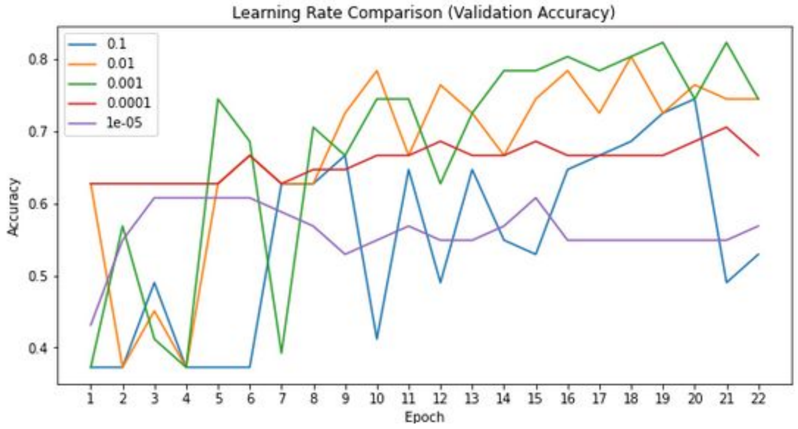
Transfer Learning (Result)

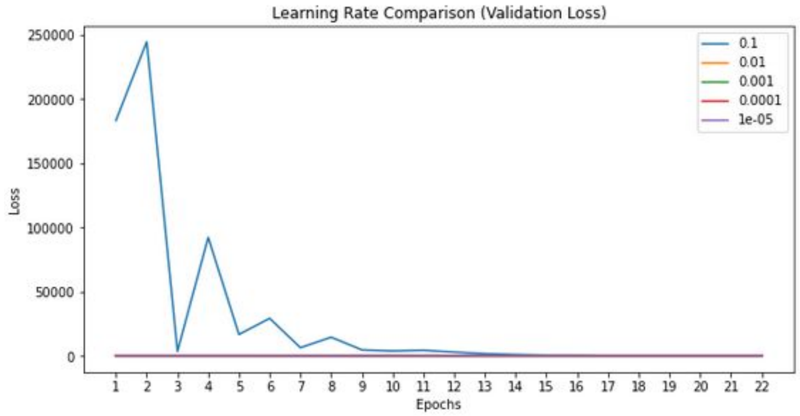


Train Accuracy	0.9216
Validation Accuracy	0.9020
Test Accuracy	0.9216
Train Loss	0.2470
Validation Loss	0.3105
Test Loss	0.2470
Precision	0.93
Recall	0.92
F1 Score	0.92

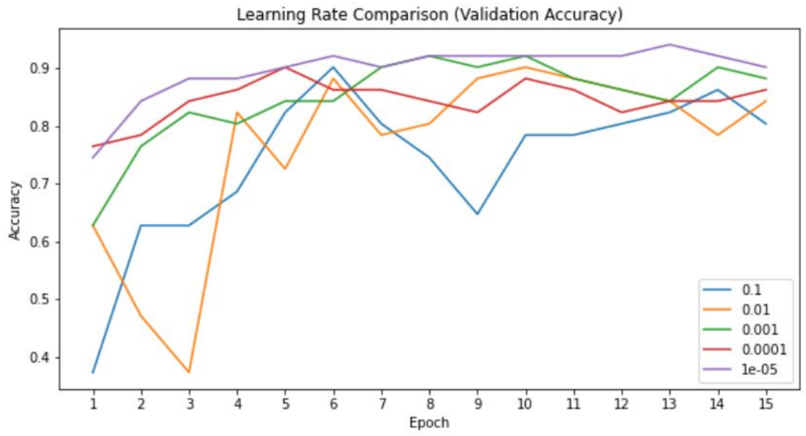
Fig

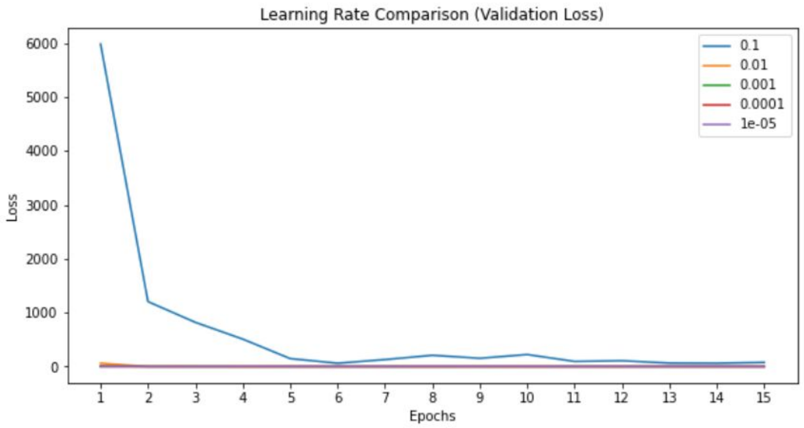
CNN Learning Rate Selection



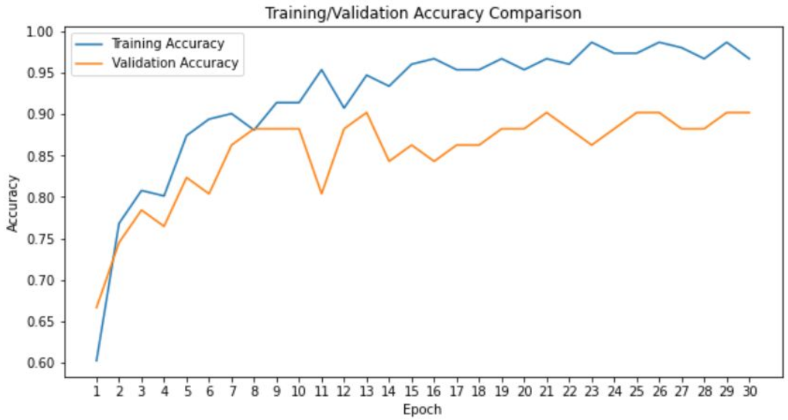


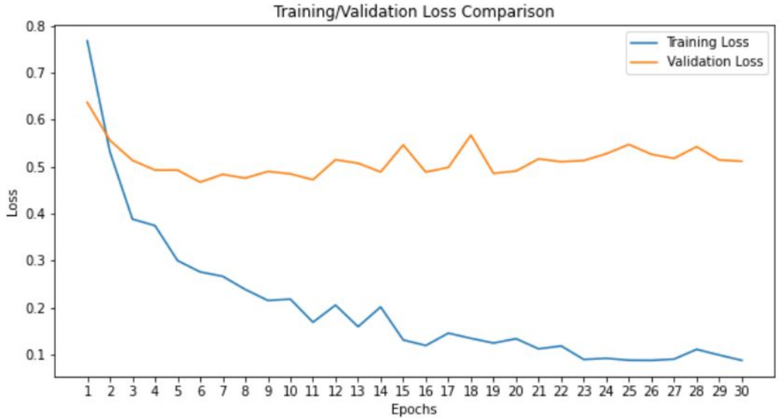
TL Learning Rate Selection





TL Epoch Number Selection





Comparing models

Measure	SVM Model	CNN Model	Transfer Learning
Test Accuracy	0.77	0.8824	0.9216
Test Loss		0.4898	0.2470

Given these numbers, Transfer Learning is the best model.

Conclusion

- 3 models were implemented.
- New approaches with oversampling, undersampling and a more customized data augmentation.
- The results were lower than the projects we based ourselves from.
- Relying on keras for data augmentation should increase our models' performance.