ETHZ - Machine Learning Project 1

Team: Orel

Authors: Cristina Bozzo
Swaneet Sahoo
Alessadra Urbinati

Summary:

1)Preprocessing:
First of all, we have implemented a show-interacted procedure to visualize slice of the samples.

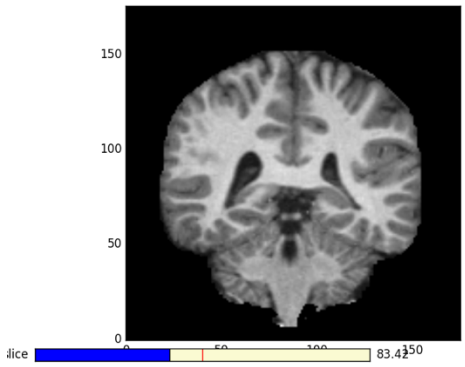


Figure 1: Front-to-back sliding, test_sample_1, age 20

Then we have flattened out the data (6 million dimensions at the beginning) into a matrix whose rows are the sample (train and test) and columns the features.

2)Features extraction:

We have tried different techniques to extract the important features. We have used both naïve approaches like selecting just a cube for each samples or zooming the intensity of the voxels into a lower resolution through simple average; and more complex method like PCA, kernel PCA or SVD, but in this case we were loosing any kind of interpretability, the results were not so good and some of them took too much time. So we have decided to come back to some basic statics, we computed average, second moment, third moment, standard deviation, mode, median, and for each sample histograms of the voxels intensity, we tried to group these in 400, 200, 50 boxes and at the end decided the 200 were the best trade-off between low variance and too much over fitting.
Looking at at the following graph (Figure 2), we have decided to take into account just values of intensity from 1 (excluding the 'black') to 2000, since after that we have assumed there would have been just noise.

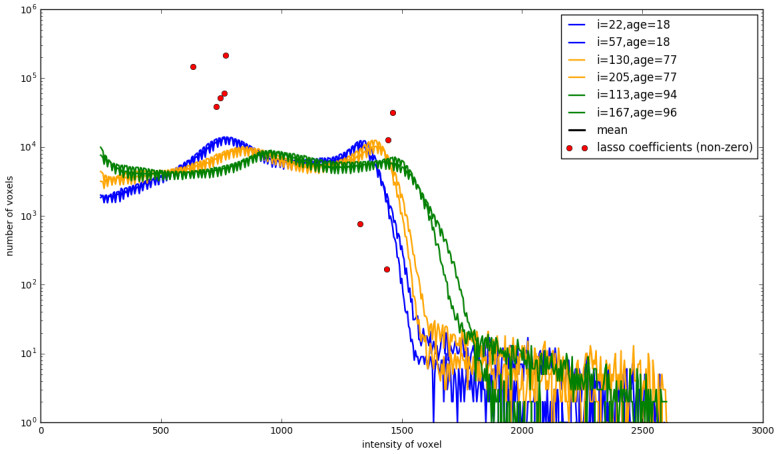
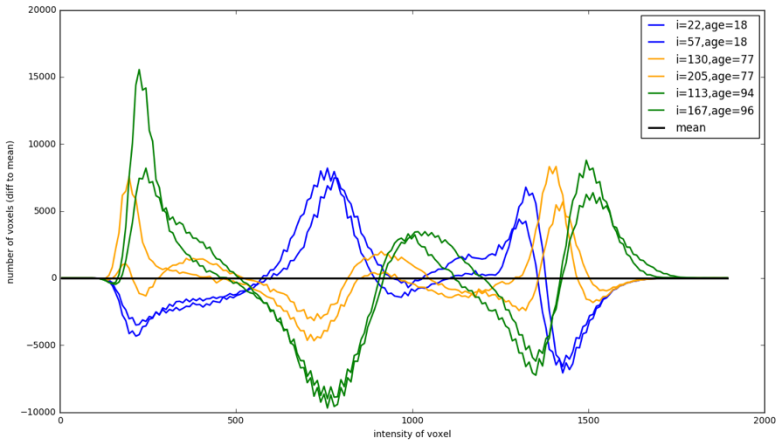


Figure 2.

We have decided to centre the histograms subtracting the intensity mean (we also subtracted the median but the results did not improve). Figure 3.



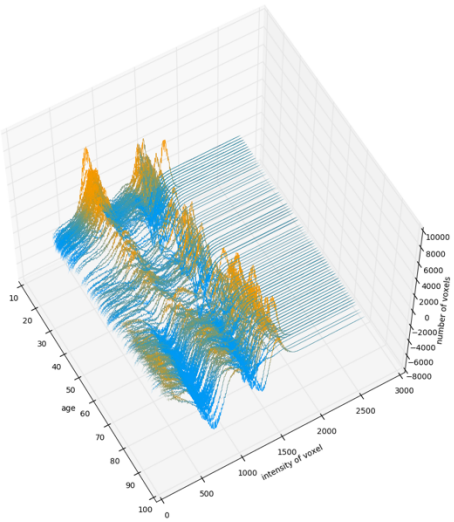


Figure 4: All histograms plotting in the same diagram

3) Train model Y:

We have tried different method according to the various extractions of features. The Lasso regression after the zoom took us at score 90, it was not good enough. With the final extraction (statistics plus histogram) and Lasso we reached 77, it was a big improvement.

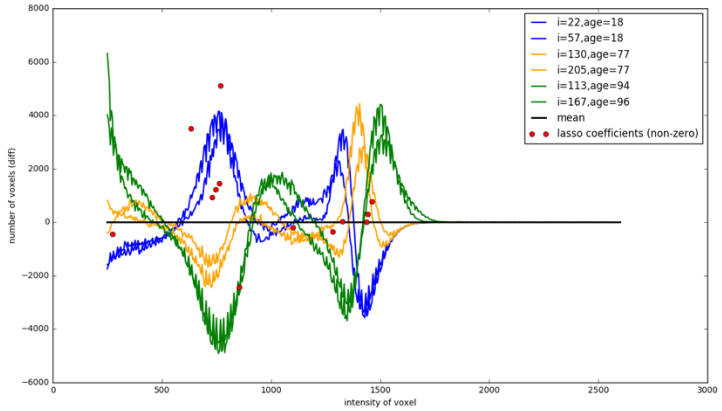


Figure 5.

After learning about boosting and averaging regressors we have tried AdaBoost with Lasso, the score was of 70. But AdaBoost is most effecient with sparse data so we tried BaggingBoost instead, the score was more or less the same. We have substituted Lasso with the default method for Bagging Boost, the Decision Tree, in this case we reached our best score: 58.
Since inside the Decision Tree there is a random state, we have tried different values for the depth of the tree (knowing that more depth could lead to over fitting), the scores were not very different.

4) Postprocessing:
We have saved the predictions in a cvs file and obtained the following plots to check them:

Figure 6: Histogram of the actual ages of training data and the predicted ages for test data.

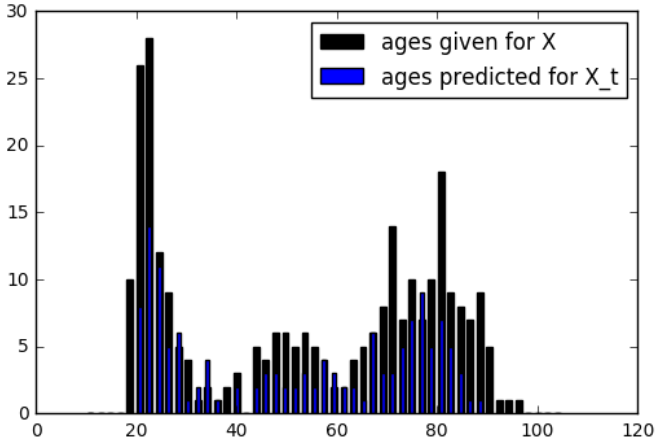


Figure 7: Difference per age prediction to one of score 58.

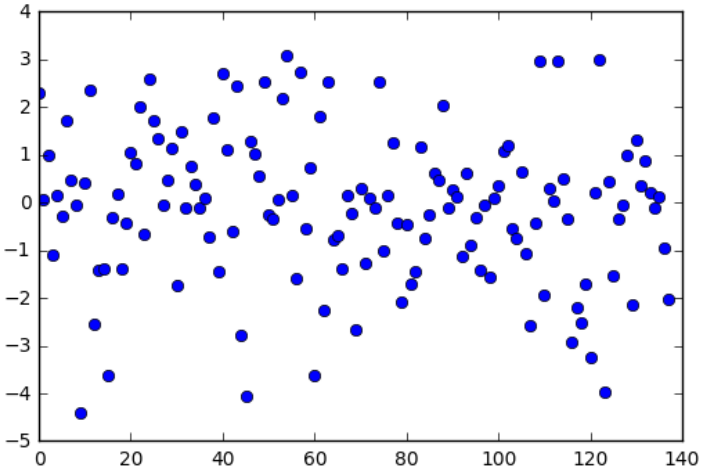


Figure 8: Sorted age predictions for test data (to visualize the predictions)