Distributional Transformation improves Decoding Accuracy when Predicting Chronological Age from Structural MRI

Joram Soch [1,2,3,*]

[1] *Berlin Center for Advanced Neuroimaging, Charité – Universitätsmedizin Berlin, corporate member of Freie Universität Berlin, Humboldt-Universität zu Berlin, and Berlin Institute of Health, Berlin, Germany*
[2] *Berlin Center for Computational Neuroscience, Berlin, Germany*
[3] *German Center for Neurodegenerative Diseases, Göttingen, Germany*

Correspondence*:
BCCN Berlin, Philippstraße 13, Haus 6, 10115 Berlin, Germany
joram.soch@charite.de, joram.soch@bccn-berlin.de, Joram.Soch@DZNE.de

ABSTRACT

When predicting a certain subject-level variable (e.g. age in years) from measured biological data (e.g. structural MRI scans), the decoding algorithm does not always preserve the distribution of the variable to predict. In such a situation, distributional transformation (DT), i.e. mapping the predicted values to the variable's distribution in the training data, might improve decoding accuracy. Here, we tested the potential of DT within the 2019 Predictive Analytics Competition (PAC) which aimed at predicting chronological age of adult human subjects from structural MRI data. In a low-dimensional setting, i.e. with less features than observations, we applied multiple linear regression, support vector regression and deep neural networks for out-of-sample prediction of subject age. We found that (i) when the number of features is low, no method outperforms linear regression; and (ii) except when using deep regression, distributional transformation increases decoding performance, reducing the mean absolute error (MAE) by about half a year. We conclude that DT can be advantageous when predicting variables that are non-controlled, but have an underlying distribution in healthy or diseased populations.

Keywords: structural neuroimaging, machine learning, decoding, prediction, structural MRI, chronological age, distributional transformation, continuous variables

1 INTRODUCTION

In recent years, probabilistic modelling (Stephan and Mathys, 2014) and machine learning (Rutledge et al., 2019) have been increasingly applied to psychiatric populations and problems, leading to the creation of a whole new field of research called ”Computational Psychiatry” (Huys et al., 2016).

The prediction of human age from biological data holds a large promise for computational psychiatry, because biologically predicted age may serve as an important biomarker for a number of processes associated with increasing age, such as declining memory performance in Alzheimer’s Disease (AD).

The 2019 Predictive Analytics Competition[1] (PAC), held before the the 25th Annual Meeting of the Organization for Human Brain Mapping[2] (OHBM), addressed exactly this question by asking teams to predict chronological age of human subjects from raw or preprocessed structural magnetic resonance imaging (sMRI) data using a self-chosen machine learning (ML) approach.

Because brain structure changes significantly when becoming older, age can be predicted from sMRI with considerable precision (Cole et al., 2017), usually quantified as mean absolute error (MAE). Brain-predicted age difference (BPAD), i.e. the difference between age predicted from sMRI and actual age, can either be a sign of ”accelerated” (BPAD > 0) or ”decelerated” (BPAD < 0) brain aging. Accelerated brain aging has been associated with lower levels of education and physical exercise (Steffener et al., 2016) and less meditation (Luders et al., 2016).

Recently, attention has been brought to the fact that ML-style decoding algorithms tend to reproduce the distribution of a learned target variable, in the sense that the distribution of test set predictions will be closer to the distribution of training set samples than to a uniform distribution, expected when purely guessing the target variable (Brodersen et al., 2010, Fig. 1). While this is a general problem of backward decoding algorithms, as opposed to interpretable forward models (Haufe et al., 2014), it is clear that the distribution of predicted values of the target variable will not be exactly identical to the distribution of those values learned from.

In this situation, distributional transformation (DT) might further improve decoding accuracy. Applied to out-of-sample ML prediction, DT operates by transforming the distribution of predicted values into the distribution of learned values of the variable of interest. In this way, prediction of the target variable is not only achieved by reconstructing it from the test set features, but additionally aided by looking at the training set samples, such that predictions are more likely to be in a realistic range for that particular target variable (e.g. age).

In this study, we apply DT to PAC 2019 data, while predicting chronological age using either multiple linear regression, support vector regression (SVR) and deep neural networks (DNN). In summary, we find that (i) multiple linear regression outperforms all other methods in a low-dimensional feature space and (ii) distributional transformation reduces prediction error for linear regression and SVR, but not DNN regression.

[1] https://www.photon-ai.com/pac2019

[2] https://github.com/ohbm/OpenScienceRoom2019/issues/10

2 METHODS

2.1 Structural MRI data

Data supplied within the 2019 Predictive Analysis Competition (PAC) included pre-processed grey matter (GM) and white matter (WM) density images from $n_1 = 2640$ training set and $n_2 = 660$ validation set subjects (for details, see Cole et al., 2017). Covariates included subjects' gender and site of image acquisition; data were acquired at 17 different sites. Subjects' age in years was supplied for the training set (2640 values), but not shared and only after the competition released for the validation set (660 values). The ratio of training to validation set size is 4:1 (see Table 1).

2.2 Feature extraction

The Automated Anatomical Labeling (AAL; Tzourio-Mazoyer et al., 2002) atlas parcellates the human brain into 90 cortical and 26 cerebellar regions. We used the AAL label image (supplied with MRIcroN[3] and also available from the TellMe package[4]) and resliced it to the first pre-processed GM image in order to match image dimensions and voxel size. We then extracted average GM and WM density from all 116 regions from the pre-processed structural images for each subject.

Acquisition site information was transformed into 17 indicator regressors and subject gender information was transformed into a $+1/-1$ regressor. Together with the extracted GM and WM densities, this constituted design matrices for training and validation data having $p = 2 \times 116 + 17 + 1 = 250$ columns (see Figure 1).

2.3 Decoding algorithms

Let y_1 and y_2 be the $n_1 \times 1$ and $n_2 \times 1$ training and validation data vector and let X_1 and X_2 be the $n_1 \times p$ and $n_2 \times p$ training and validation design matrix.

2.3.1 Multiple linear regression

Multiple linear regression proceeds by estimating regression coefficients via ordinary least squares (OLS) from the training data

$$\hat{\beta}_1 = (X_1^{\mathrm{T}} X_1)^{-1} X_1^{\mathrm{T}} y_1 \tag{1}$$

and generating predictions by multiplying the design matrix with estimated regression coefficients in the validation data

$$\hat{y}_2 = \hat{f}_1(X_2) = X_2 \hat{\beta}_1 \ . \tag{2}$$

2.3.2 Support vector regression

Support vector regression (SVR) was implemented in MATLAB using `fitrsvm`. A support vector machine was calibrated using the training data and then used to predict age in the validation data:

$$\begin{aligned} \hat{g}_1 &\leftarrow \texttt{fitrsvm}(X_1, y_1) \\ \hat{y}_2 &= \hat{g}_1(X_2) \end{aligned} \tag{3}$$

[3] https://people.cas.sc.edu/rorden/mricron/install.html

[4] https://github.com/JoramSoch/TellMe

2.3.3 Deep neural network regression

Deep neural network (DNN) regression was implemented in MATLAB using `trainNetwork`. Before training, non-indicator regressors in X_1 and X_2 were z-scored, i.e. mean-subtracted and divided by standard deviation:

$$x_{1j}^* = \frac{x_{1j} - \bar{x}_{1j}}{\hat{\sigma}_{1j}}, \quad x_{2j}^* = \frac{x_{2j} - \bar{x}_{2j}}{\hat{\sigma}_{2j}}, \quad j = 1, \ldots, p \,. \tag{4}$$

The network consisted of six layers (see Table 2) following a MathWorks tutorial[5] on deep learning for linear regression and was solved in training using the Adam optimizer. The number of epochs was set to 100, with a mini batch size of 20, an initial learning rate of 0.01 and a gradient threshold of 1. Similarly to SVR, training and prediction proceeded as follows:

$$\begin{aligned} \hat{h}_1 &\leftarrow \texttt{trainNetwork}(X_1^*, y_1, \texttt{layers}, \texttt{options}) \\ \hat{y}_2 &= \hat{h}_1(X_2^*) \end{aligned} \tag{5}$$

2.4 Distributional transformation

Because the distribution of predicted age values will not exactly match the distribution of validation set age and likely also deviates from the distribution of training set age, one can apply an additional distributional transformation (DT) step after prediction.

DT uses cumulative distribution functions[6] (CDFs). Let X and Y be two random variables. Then, X is distributionally transformed to Y by replacing each observation of X by that value of Y which corresponds to the same quantile as the original value, i.e.

$$\tilde{x} = F_Y^{-1}\left(F_X(x)\right) \tag{6}$$

where F_X is the CDF of X and F_Y^{-1} is the inverse CDF of Y. Note that DT preserves the complete ordering of X, but changes its CDF to that of Y (see Appendix A).

Here, we apply DT to the predicted ages $\hat{y}_2$, with the goal of mapping them to the distribution of the training ages y_1 by calculating

$$\tilde{y}_{2i} = F_1^{-1}\left(\hat{F}_2(\hat{y}_{2i})\right), \quad i = 1, \ldots, n_2 \tag{7}$$

where $\hat{F}_2$ is the empirical CDF of $\hat{y}_2$ and F_1^{-1} is the inverse empirical CDF of y_1, obtained in MATLAB using `ecdf` (see Appendix B).

After the transformation, the ranks of all predictions $\hat{y}_{2i}$ are still the same, but the empirical CDF of $\tilde{y}_2$ matches that of y_1. In other words, we receive something that looks like the training age values in terms of age distribution, but is still predicted from the validation brain data.

[5] https://de.mathworks.com/help/deeplearning/ug/sequence-to-sequence-regression-using-deep-learning.html, accessible in MATLAB as `openExample('nnet/SequencetoSequenceRegressionUsingDeepLearningExample')`

[6] https://statproofbook.github.io/D/cdf

The rationale behind this is that, if training and validation set are unbiased, representative and unsystematic samples from the underlying population, then sampling from the training data should in itself be a good prediction strategy for the validation data. For example, because it can be suspected that mean age has been controlled for when dividing into training and validation data, the age distributions in training and validation data should be close to each other.

2.5 Performance assessment

After generating predictions for the validation set, we assessed decoding accuracy using multiple measures of correlation (see Table 3) between predicted ages $\hat{y}_2$ and actual ages y_2. During the PAC 2019, Objective 1 was to minimize the mean absolute error. Objective 2 was to minimize Spearman's rank correlation coefficient between y_2 and $(y_2 - \hat{y}_2)$, as it is desirable that the brain-predicted age difference (BPAD) is not correlated with age.

The complete data analysis as well as resulting decoding accuracies can be reproduced using the contents of a GitHub repository (see Data Availability Statement).

3 RESULTS

3.1 Influence of decoding algorithm

Qualitatively, prediction performance can be assessed via scatterplots of predicted age against actual age in the validation set (see Figure 2). Quantitatively, the ranking is similar across all measures of correlation (see Figure 3): multiple linear regression performs best ($r = 0.91$, MAE = 5.07 yrs), but only mildly outperforms deep neural network regression ($r = 0.89$, MAE = 5.18 yrs) and strongly outperforms support vector regression ($r = 0.83$, MAE = 6.82 yrs). This is true for measures which are to be maximized (R^2, R^2_{adj}, r, r_{SC}) as well as for measures which are to be minimized (MAE, RMSE, Obj. 2).

3.2 Influence of distributional transformation

When comparing predicted against actual age, one can see that SVR and DNN predictions deviate quite some amount from the actual distribution (see Figure 2, top-middle and top-right), especially by not predicting very high ages (SVR: $\max(\hat{y}_2) = 71.25$ yrs; DNN: $\max(\hat{y}_2) = 75.68$ yrs), whereas linear regression creates a more homogeneous picture (see Figure 2, top-left), but also predicts very low ages ($\min(\hat{y}_2) = 1.06$ yrs).

DT improves the decoding accuracy of linear regression (MAE: 5.07 $\rightarrow$ 4.58 yrs) and SVR (MAE: 6.82 $\rightarrow$ 6.11 yrs) by reducing their MAE by about half a year. For DNN, the error actually goes up (MAE: 5.18 $\rightarrow$ 5.42 yrs). Similar results are observed when considering other measures. Distributional transformation especially benefits Objective 2 of PAC 2019, as the Spearman correlation of brain-predicted age difference with age itself goes down considerably when applying DT (see Figure 3, right), thus increasing the independence of prediction error from predicted variable.

We also calculate the empirical Kullback-Leibler (KL) divergence[7] of the distribution of actual ages from the distributions of predicted ages. The KL divergence is a non-negative distance measure for distributions; the more similar two distributions are, the more closer it is to zero. When not applying DT (see Figure 4, top row), DNN yields the smallest KL divergence, with predictions almost being in the correct range ($17 \leq y_2 \leq 89$), where linear regression achieves lower correspondence and SVR suffers from making a lot medium-age predictions ($40 \leq \hat{y}_2 \leq 60$) and there not being a lot middle-aged subjects in the training and validation sample.

When applying DT (see Figure 4, middle row), all methods give rise to the same histogram of predicted ages and have the same and minimally possible distance to the actual distribution. Still, despite having the same distribution, prediction performance differs between methods (see Figure 3).

3.3 Influence of regression coefficients

In order to see which features aided successful prediction of age when using multiple linear regression, we report parameter estimates and compute confidence intervals (see Figure 5), computed after concatenating training and validation data (because no out-of-sample testing is needed for this analysis). These results show that (i) there was no effect of gender on age, putatively because gender was controlled when splitting the data; (ii) there were only mild site effects, putatively because the whole age range was sampled at each site; and (iii) regional GM and WM densities both contributed to the predictions, as variables from both groups have significant effects on subject age (see Figure 5).

[7] https://statproofbook.github.io/D/kl

4 DISCUSSION

In the present study, we have shown that, when operating in a low-dimensional setting with fewer features than observations (here: by extracting from AAL regions), combining distributional transformation (DT) with relatively simple decoding algorithms (e.g. linear regression or SVR) might be a promising way to reach acceptable decoding accuracies in short time. Notably, DT does not increase prediction precision when the decoding algorithm (e.g. DNN regression) generates test set predictions that already have a similar distribution as the training set samples.

The rationale behind distributional transformation for ML prediction is simple:

1. A lot of target variables have a natural range into which their values must fall:
   a. Human age cannot be smaller than zero (or at least, smaller than -9 months), is rarely larger than 100 years and has thus far not exceeded 122 years.
   b. Intelligence quotients (IQ) are (by construction) normally distributed with mean 100 and standard deviation 15.
   c. Physical parameters such as weight and height fall into typical ranges which are differing by gender and age.
   d. Probabilities and frequencies, e.g. the proportion of correct responses, are bounded to be in the interval $[0, 1]$.
2. When associations between target variable and feature space are learned by sending training samples through a complex machinery of linear and non-linear optimizations, some test set predictions will likely be outside these areas, thereby violating the natural range of the target variable.
3. Distributional transformation brings the predictions back into the natural range by putting them in reference to the training set samples, but preserving the ranks obtained when reconstructing from the test set features.

DT can be particularly advantageous when predicting variables which are hard to control experimentally (esp. biological phenotypes), but the distribution of which is known through the availability of training data. We have provided an easy algorithm for DT (see Appendix) and explored its potential for predicting chronological age from structural MRI. Future studies may investigate whether the DT methodology might be beneficial in other areas of computational psychiatry.

5 APPENDIX

A Proof of distributional transformation

Let X and Y be two random variables. Then, X is distributionally transformed to Y by replacing each observation of X by that value of Y which corresponds to the same quantile as the original value (see Section 2.4), i.e.

$$\tilde{x} = F_Y^{-1}\left(F_X(x)\right) \ . \tag{A.1}$$

where $F_X(x)$ is the cumulative distribution function (CDF) of X and $F_Y^{-1}(y)$ is the inverse CDF of Y. Consequently, the CDF of $\tilde{X}$ follows as

$$\begin{aligned} F_{\tilde{X}}(y) &= \Pr(\tilde{x} \leq y) \\ &= \Pr\left(F_Y^{-1}\left(F_X(x)\right) \leq y\right) \\ &= \Pr\left(F_X(x) \leq F_Y(y)\right) \\ &= \Pr\left(x \leq F_X^{-1}\left(F_Y(y)\right)\right) \\ &= F_X\left(F_X^{-1}\left(F_Y(y)\right)\right) \\ &= F_Y(y) \end{aligned} \tag{A.2}$$

which shows that $\tilde{X}$ and Y have the same CDF and are thus identically distributed.

B Code for distributional transformation

The following code distributionally transforms `x` to `y` in MATLAB:

```
function xt = MD_trans_dist(x, y)
% _
% Distributional Transformation
% FORMAT xt = MD_trans_dist(x, y)
%     x  - source data, data to be transformed
%     y  - reference data, target for transformation
%     xt - transformed data, x mapped to y

% calculate CDFs
[f1, x1] = ecdf(x);
[f2, x2] = ecdf(y);

% transform x
xt = zeros(size(x));
for i = 1:numel(x)
    j1 = find(x1==x(i));
    j1 = j1(end);
   [m, j2] = min(abs(f2-f1(j1)));
    xt(i)  = x2(j2);
end;
clear m j1 j2
```

CONFLICT OF INTEREST STATEMENT

The author has no conflict of interest, financial or otherwise, to declare.

AUTHOR CONTRIBUTIONS

JS conceived, implemented and performed data analysis; created figures and tables; and wrote the paper.

FUNDING

This work was supported by the Bernstein Computational Neuroscience Program of the German Federal Ministry of Education and Research (BMBF grant 01GQ1001C).

ACKNOWLEDGMENTS

The author would like to thank Carsten Allefeld for discussing the linear regression analysis and the distributional transformation method.

We acknowledge support from the German Research Foundation (DFG) and the Open Access Publication Fund of Charité – Universitätsmedizin Berlin.

DATA AVAILABILITY STATEMENT

Data analyzed in this study were available online during the 2019 Predictive Analytics Competition. For this work, no raw images, but only pre-processed maps were used (see Section 2.1). Requests for data release should be directed to Tim Hahn[8] and Ramona Leenings[9].

MATLAB code for (i) feature extraction from pre-processed data, (ii) decoding analyses underlying the results presented in this paper and (iii) results display to reproduce the figures shown in this paper can be found in an accompanying GitHub repository[10].

[8] mailto:hahnt@wwu.de

[9] mailto:leenings@uni-muenster.de

[10] https://github.com/JoramSoch/PAC_2019

REFERENCES

Brodersen, K. H., Ong, C. S., Stephan, K. E., and Buhmann, J. M. (2010). The Balanced Accuracy and Its Posterior Distribution. In *2010 20th International Conference on Pattern Recognition* (Istanbul, Turkey: IEEE), 3121–3124. doi:10.1109/ICPR.2010.764

Cole, J. H., Poudel, R. P., Tsagkrasoulis, D., Caan, M. W., Steves, C., Spector, T. D., et al. (2017). Predicting brain age with deep learning from raw imaging data results in a reliable and heritable biomarker. *NeuroImage* 163, 115–124. doi:10.1016/j.neuroimage.2017.07.059

Haufe, S., Meinecke, F., Görgen, K., Dähne, S., Haynes, J.-D., Blankertz, B., et al. (2014). On the interpretation of weight vectors of linear models in multivariate neuroimaging. *NeuroImage* 87, 96–110. doi:10.1016/j.neuroimage.2013.10.067

Huys, Q. J. M., Maia, T. V., and Frank, M. J. (2016). Computational psychiatry as a bridge from neuroscience to clinical applications. *Nature Neuroscience* 19, 404–413. doi:10.1038/nn.4238

Luders, E., Cherbuin, N., and Gaser, C. (2016). Estimating brain age using high-resolution pattern recognition: Younger brains in long-term meditation practitioners. *NeuroImage* 134, 508–513. doi:10.1016/j.neuroimage.2016.04.007

Rutledge, R. B., Chekroud, A. M., and Huys, Q. J. (2019). Machine learning and big data in psychiatry: toward clinical applications. *Current Opinion in Neurobiology* 55, 152–159. doi:10.1016/j.conb.2019.02.006

Steffener, J., Habeck, C., O'Shea, D., Razlighi, Q., Bherer, L., and Stern, Y. (2016). Differences between chronological and brain age are related to education and self-reported physical activity. *Neurobiology of Aging* 40, 138–144. doi:10.1016/j.neurobiolaging.2016.01.014

Stephan, K. E. and Mathys, C. (2014). Computational approaches to psychiatry. *Current Opinion in Neurobiology* 25, 85–92. doi:10.1016/j.conb.2013.12.007

Tzourio-Mazoyer, N., Landeau, B., Papathanassiou, D., Crivello, F., Etard, O., Delcroix, N., et al. (2002). Automated Anatomical Labeling of Activations in SPM Using a Macroscopic Anatomical Parcellation of the MNI MRI Single-Subject Brain. *NeuroImage* 15, 273–289. doi:10.1006/nimg.2001.0978

TABLES

out-of-sample prediction	**k-fold cross-validation**	**number of subjects**
training	training	2376
	testing	264
validation	reporting	660

Table 1. Data dimensions and cross-validation. During the competition (second column), the model was developed using 10-fold cross-validation within the training data, before performance was reported on the withheld data set. In the context of this paper (first column), age is predicted out-of-sample in the validation data without cross-validation.

MATLAB command	**Description**	**Parameters**
`sequenceInputLayer`	inputs 1D data into network	250 features
`lstmLayer`	long short-term memory (LSTM) layer; learns long-range dependencies between features	125 hidden units
`fullyConnectedLayer`	multiplies with weight matrix and adds bias	50 output units
`dropoutLayer`	sets elements to zero with given probability	$p = 0.5$
`fullyConnectedLayer`	multiplies with weight matrix and adds bias	1 output unit
`regressionLayer`	outputs scalar prediction from network	–

Table 2. Layers of the deep neural network. The network employed for DNN regression consisted of six layers which were designated for using deep learning on regression problems.

Measure	**Description**
R^2	coefficient of determination (”R-squared”)
R^2_{adj}	adjusted coefficient of determination
r	Pearson correlation coefficient
r_{SC}	Spearman’s rank correlation coefficient
MAE	mean absolute error (Objective 1)
RMSE	root mean squared error
Obj. 2	Objective 2 from PAC 2019

Table 3. Measures of prediction performance. For PAC 2019 Objectives, see main text.

FIGURES

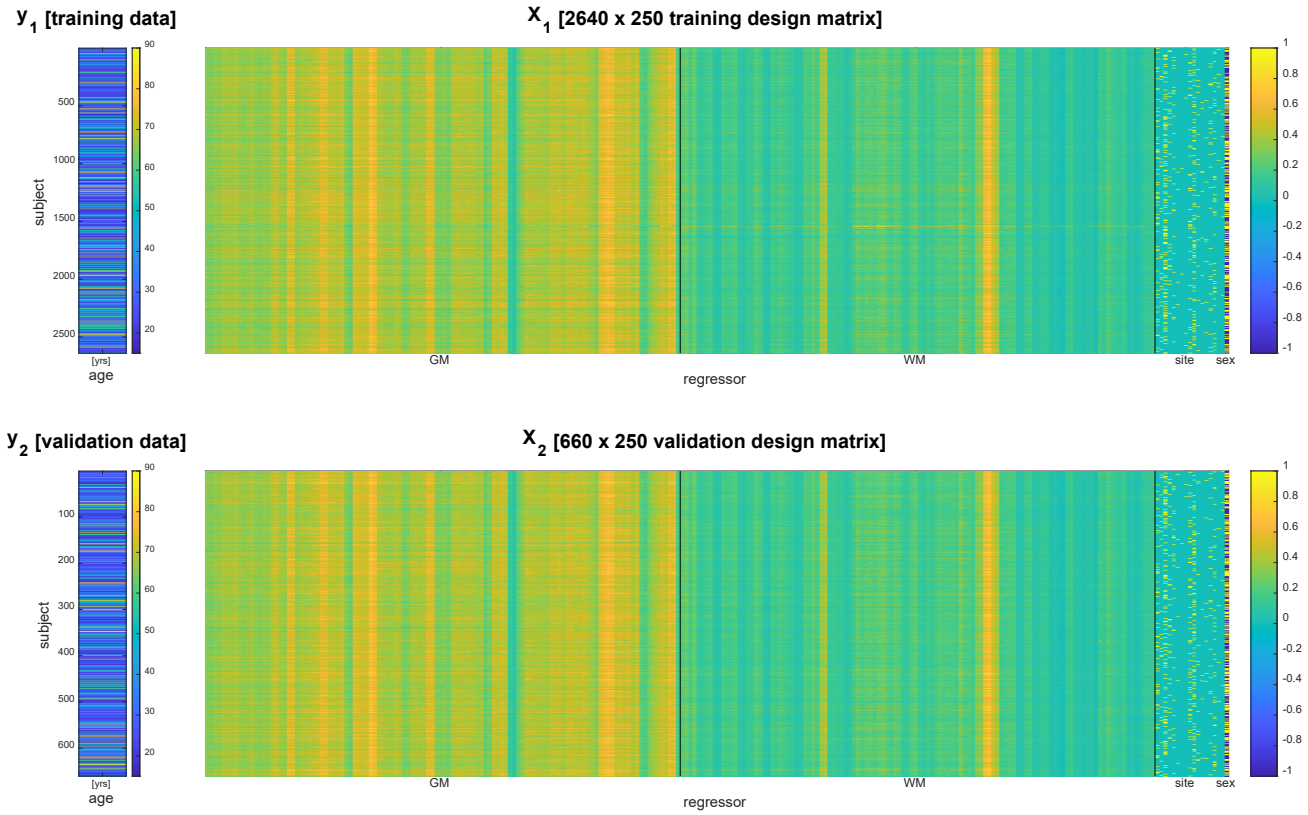

Figure 1. Data vector (left) and design matrix (right) for training set (top) and validation set (bottom). In the design matrices, rows correspond to subjects and columns correspond to brain regions and covariates. Regressors specifying grey matter (GM) and white matter (WM) densities as well as site covariates are separated by vertical black lines.

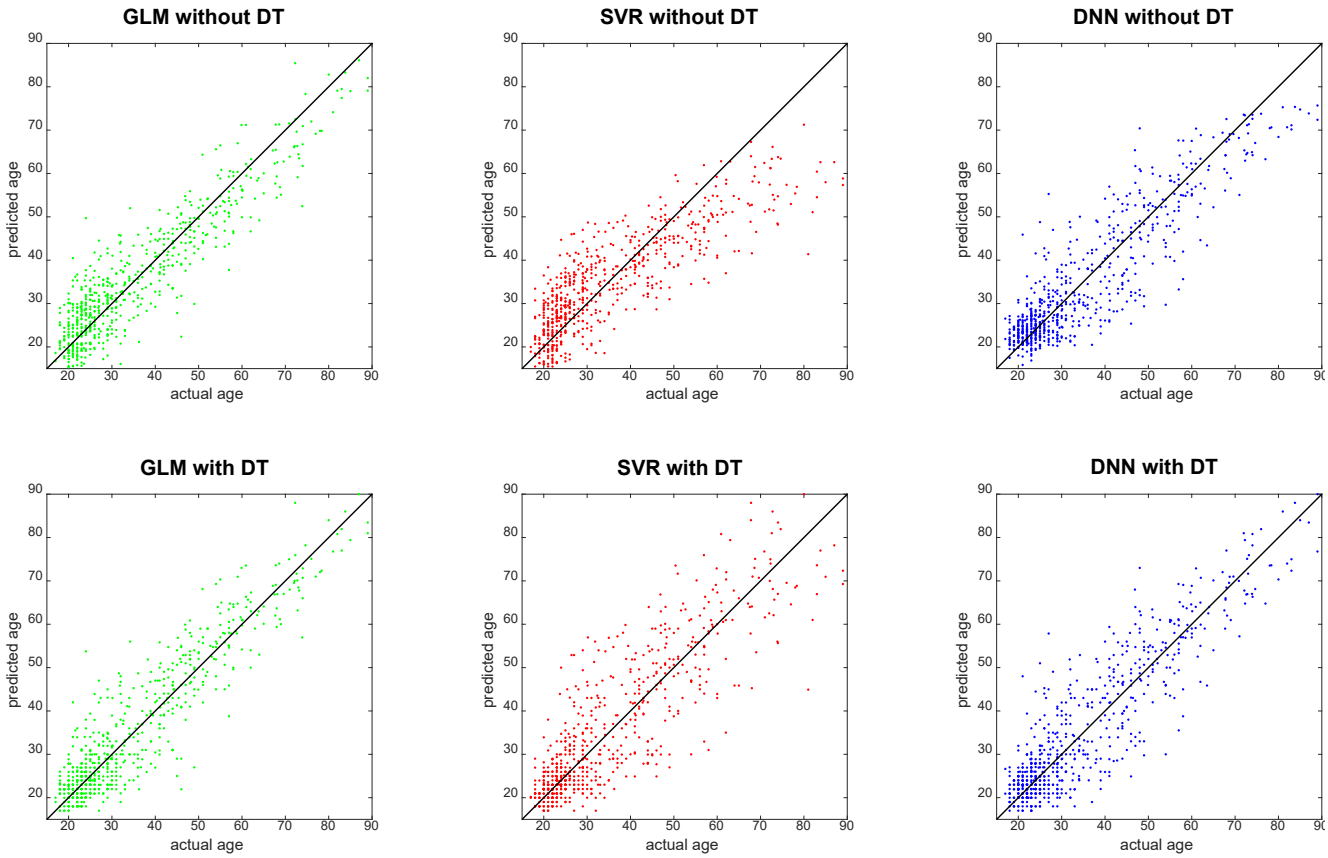

Figure 2. Predicted vs. actual age for validation set subjects. Abbreviations: GLM = multiple linear regression; SVR = support vector regression; DNN = deep neural network regression; DT = distributional transformation.

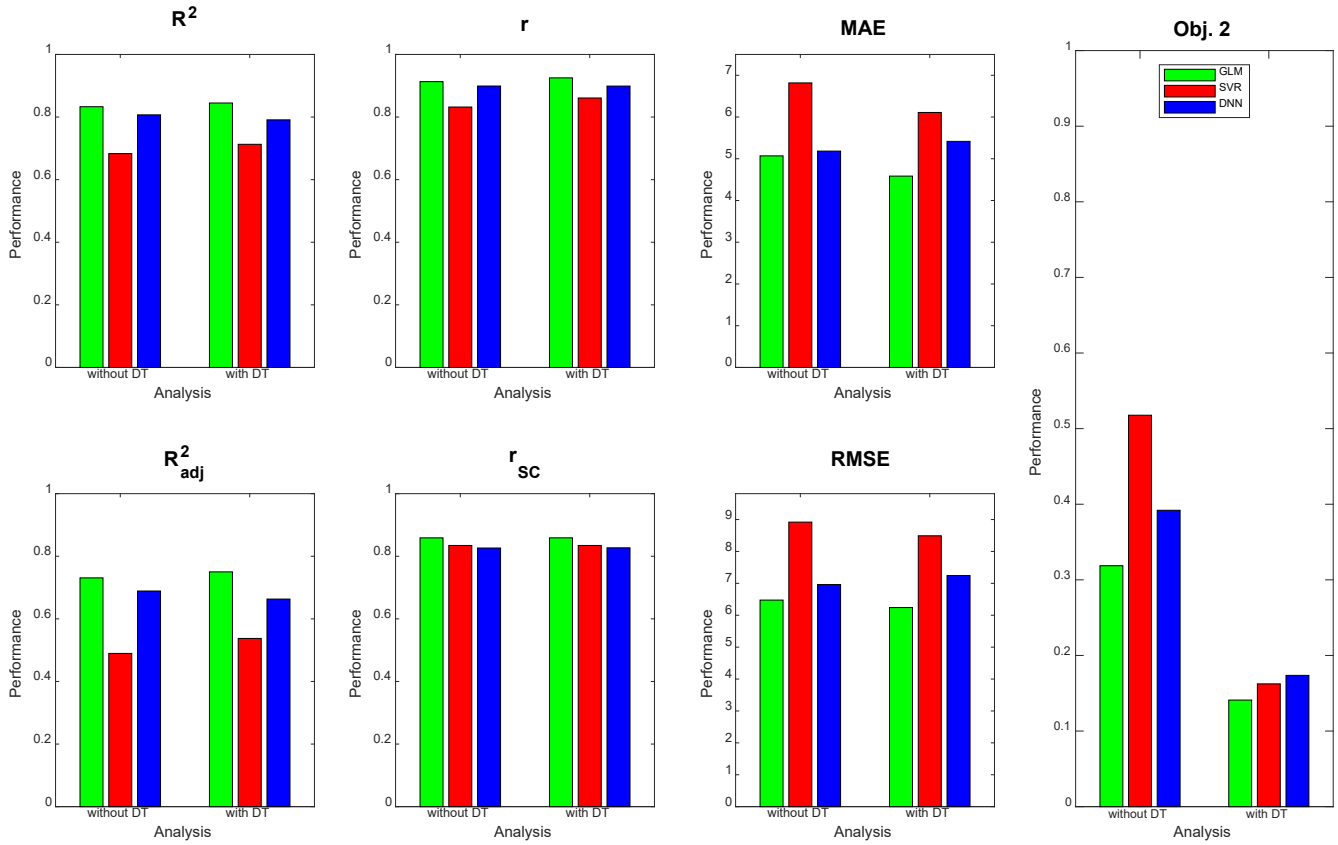

Figure 3. Prediction performance across models, methods and measures. For each performance measure, decoding accuracy is given for multiple linear regression (green), support vector regression (red) and deep neural network regression (blue), with and without distributional transformation (DT). Abbreviations: see Table 3.

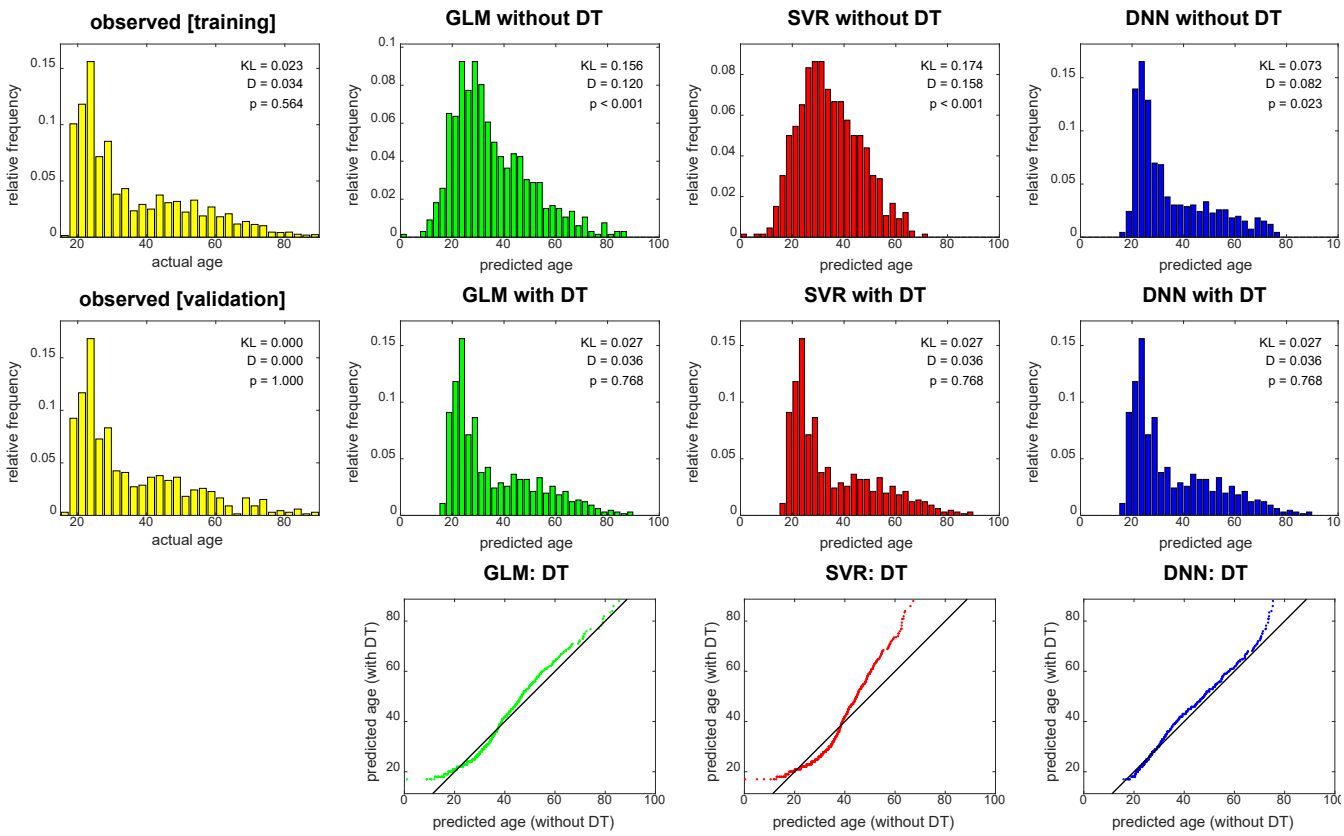

Figure 4. Empirical distributions of actual age (yellow), predicted age (RGB, top) and transformed age (RGB, middle). Each panel shows relative frequencies, i.e. number of occurences divided by total number of subjects. The applied distribution transformation is shown for each prediction model (bottom row). The empirical KL divergence was always computed relative to the validation age distribution (middle row, left), thus this distribution has $\mathrm{KL} = 0$. Also reported are test statistic D and p-value from a two-sample Kolmogorov-Smirnov test against the null hypothesis that predicted values and validation agse are from the same continuous distribution, thus $D = 0$ and $p = 1$ for validation set age. Abbreviations: GLM = multiple linear regression; SVR = support vector regression; DNN = deep neural network regression; DT = distributional transformation; KL = Kullback-Leibler divergence.

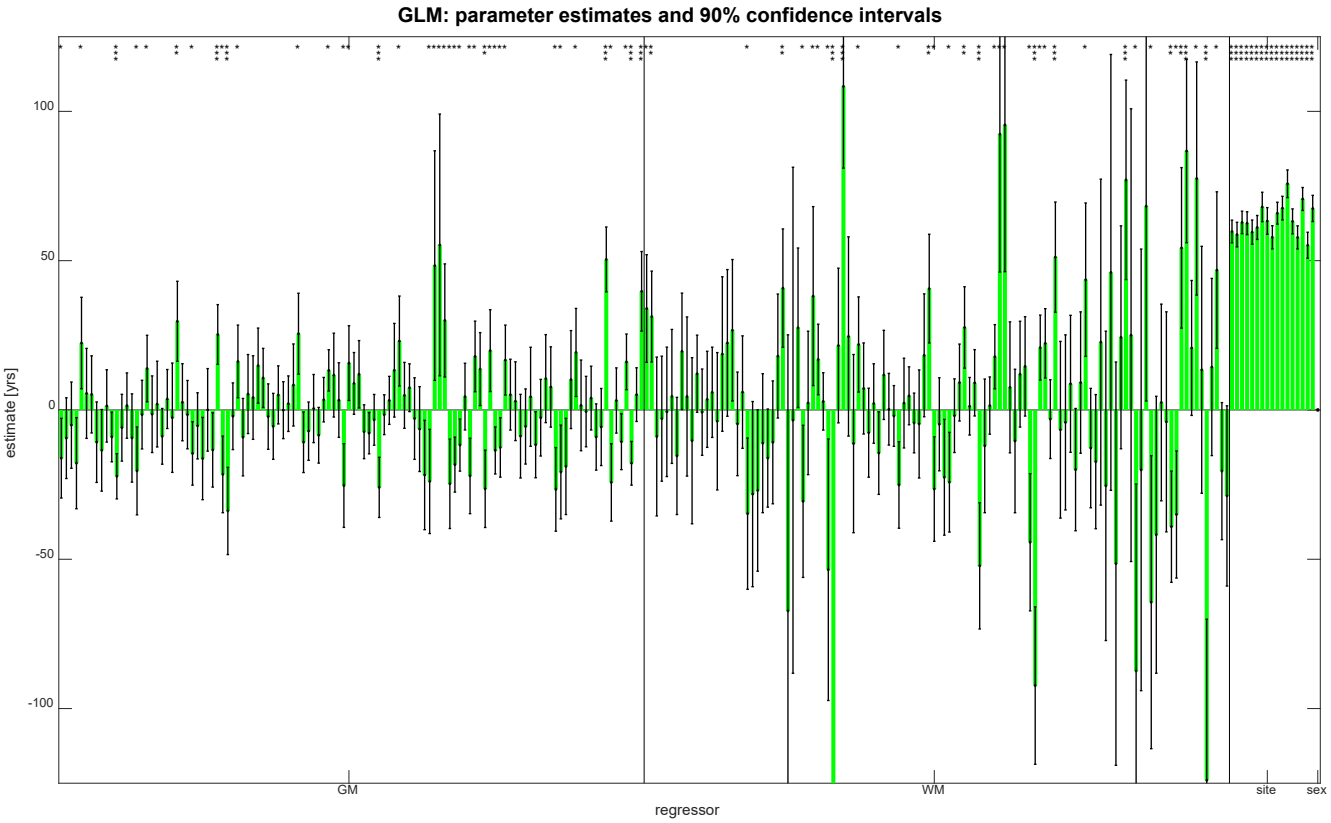

Figure 5. Parameter estimates and confidence intervals from linear regression, obtained when combining training and validation data into one data set. For each regressor, the estimated coefficient is plotted as a bar and a 90 % confidence interval is reported as well. Model parameters are also reported as being significantly different from zero (* $p < 0.05$; ** $p < 0.001$; *** $p < 0.05/250$).