Original Article

Exploring the Potential of Machine Learning Algorithms to Improve Diffusion Nuclear Magnetic Resonance Imaging Models Analysis

Leonar Steven Prieto-González, Luis Agulles-Pedrós

Department of Physics, Medical Physics Group, National University of Colombia, Campus Bogotá, Bogotá, Colombia

Abstract

Purpose: This paper explores different machine learning (ML) algorithms for analyzing diffusion nuclear magnetic resonance imaging (dMRI) models when analytical fitting shows restrictions. It reviews various ML techniques for dMRI analysis and evaluates their performance on different *b*-values range datasets, comparing them with analytical methods. **Materials and Methods:** After standard fitting for reference, four sets of diffusion-weighted nuclear magnetic resonance images were used to train/test various ML algorithms for prediction of diffusion coefficient (D), pseudo-diffusion coefficient (D*), perfusion fraction (f), and kurtosis (K). ML classification algorithms, including extra-tree classifier (ETC), logistic regression, C-support vector, extra-gradient boost, and multilayer perceptron (MLP), were used to determine the existence of diffusion parameters (D, D*, f, and K) within single voxels. Regression algorithms, including linear regression, polynomial regression, ridge, lasso, random forest (RF), elastic-net, and support-vector machines, were used to estimate the value of the diffusion parameters. Performance was evaluated using accuracy (ACC), area under the curve (AUC) tests, and cross-validation root mean square error ($RMSE_{CV}$). Computational timing was also assessed. **Results:** ETC and MLP were the best classifiers, with 94.1% and 91.7%, respectively, for the ACC test and 98.7% and 96.3% for the AUC test. For parameter estimation, RF algorithm yielded the most accurate results The $RMSE_{CV}$ percentages were: 8.39% for D, 3.57% for D*, 4.52% for f, and 3.53% for K. After the training phase, the ML methods demonstrated a substantial decrease in computational time, being approximately 232 times faster than the conventional methods. **Conclusions:** The findings suggest that ML algorithms can enhance the efficiency of dMRI model analysis and offer new perspectives on the microstructural and functional organization of biological tissues. This paper also discusses the limitations and future directions of ML-based dMRI analysis.

Keywords: Diffusion magnetic resonance imaging, intravoxel incoherent motion, kurtosis, machine learning

Received on: 17-01-2024 **Review completed on:** 27-03-2024 **Accepted on:** 15-04-2024 **Published on:** 25-06-2024

Introduction

Diffusion magnetic resonance imaging (dMRI) has become an increasingly important tool in medical imaging, allowing the study of microstructural changes in biological tissues.[1] The main component in dMRI analysis is the estimation of the apparent diffusion coefficient (ADC). A deeper understanding of tissue microstructure can be made by estimation of other parameters such as pseudo-diffusion coefficient (D*) and perfusion fraction (f) from Intravoxel incoherent motion (IVIM) model,[2-4] and kurtosis (K) from similar advanced models.[5-7] However, estimating these parameters are challenging due to the complexity of the underlying physics and the limitations of standard analytical fitting.[8-12]

Limitations of standard analytical fitting in dMRI models

MRI is a widely used technique for disease characterization. Since MRI signal comes mainly from water molecules hydrogen nuclei, self-diffusion allows one to obtain internal structure information; the random water hydrogen movement

Address for correspondence: Prof. Luis Agulles-Pedrós, Carrera 30 # 45, Physics Department, Building 404, Bogotá, Colombia.
E-mail: lagullesp@unal.edu.co

Access this article online

Quick Response Code:

Website: www.jmp.org.in

DOI: 10.4103/jmp.jmp_10_24

How to cite this article: Prieto-González LS, Agulles-Pedrós L. Exploring the potential of machine learning algorithms to improve diffusion nuclear magnetic resonance imaging models analysis. J Med Phys 2024;49:189-202.

will be restricted depending on the tissue microstructure.[13] The simplest diffusion model was developed by Stejskal and Tanner[14] and is described by the following equation and considers unrestricted isotropic displacements:

$$S(b)/S_0 = exp(-bD) \quad (1)$$

where S is the signal intensity with diffusion weighting, S_0 is the signal intensity without diffusion weighting ($b = 0$), b is a function of the gradient intensity and time during which the gradients are applied (also known as b-value), and D is the self-diffusion coefficient.

This model predicts a signal exponential decay, which depends on the gradient intensity and timing. These parameters are summarized in the b-value, which is usually set up to 500, 1000, or 1500 s/mm^2, and constant during acquisition.[15]

The ADC reflects the modification of the free self-diffusion coefficient due to the restricted movement within intra- and extracellular spaces due to specific tissue characteristics. By applying the Stejskal–Tanner equation (Equation 1) to each voxel, quantitative ADC maps can be generated by fitting to different b-values.[16] In this text, D and ADC are used interchangeably for the sake of simplicity.

Intravoxel incoherent motion (IVIM)

The range of b-values is crucial since it can provide significant insights into tissue microstructure.[2] Lower b-values (usually below 100 s/mm^2) will not allow observation of signal decay due to self-diffusion, even restricted due to the "speed" of diffusivity and cell size. Though self-diffusion and blood microcirculation occur at distinct scales and involve different physical processes, an observed apparent motion randomness arises from the arrangement of the vessel network through which blood flows. In essence, randomness emerges from the collective movement of water molecules within this network, transitioning between capillary segments. This collective motion resembles a pseudo-diffusion process and can be detected for b-values < 200 s/mm^2.[2] The IVIM model was developed by Le Bihan *et al*.[3] It assumes two exponential decays for each diffusion regime; then the signal is split into two addends, one due to self-diffusion restrictions and the other to vessel network movement, as follows:

$$\frac{S}{S_0} = f \cdot exp(-bD^*) + (1-f)exp(-bD) \quad (2)$$

In human tissues, D is the previous ADC value, D^* is the "microcirculation pseudo-diffusion."[3] The parameter f is the proportion of signal detected in both diffusion regimes and is known as perfusion fraction.[2]

Kurtosis

The models previously introduced (Equations 1 and 2) assume a normal distribution of diffusion magnitudes and their directions. This point is vital since it results in an exponential signal dependence.[16] However, this assumption is only the case in some tissues. A non-Gaussian distribution assumes "kurtosis behavior,"[4,5,16] which gives a quadratic-exponential behavior of the signal, as follows:

$$S/S_0 = f \cdot exp(-bD^*) + (1-f)exp(-bD + b^2D^2K/6) \quad (3)$$

where K accounts for the kurtosis factor when the Gaussian model is insufficient to fully characterize the complexity of the diffusion process in some biological tissues.[15]

Recent research has shown that diffusion kurtosis imaging can provide valuable information about the microstructural properties of tissues, such as cellular density, axon diameter, and myelin content, which are essential for a variety of clinical applications, including the diagnosis and monitoring of neurological disorders, cancer, and inflammatory diseases.[4-7]

Standard analytical fitting in dMRI models

Fitting ADC mono-exponential signal decay is usually based on a robust linear least square fitting, revealing consistency in almost all cases.[8,17] However, it lacks higher microstructural details and possible extra diagnostic information. Then, IVIM or kurtosis models may reveal a higher order of internal structure.[12,18]

One of the parameters to take into consideration to identify which diffusion model suits best is the robustness of the fitting.[8] Evaluating the goodness of fit is a useful method for model discrimination. A comprehensive analysis of the image set may be necessary to determine the appropriate goodness of fit and the corresponding value to be considered. Other limitations arise from the D and D^* limit of the b-value in the case of the IVIM model[9,10] or even for kurtosis models,[11] where an agreement between noise and scan time is decisive.[12]

Machine learning in dMRI

The field of machine learning (ML), particularly automatic learning, has seen rapid growth in recent years, demonstrating considerable potential in medical physics applications.[19,20] In the imaging field, ML algorithms have gained practical uses due to their capability to perform tasks that are challenging or impractical with conventional algorithms.[21]

ML-based dMRI analysis is an emerging field that employs artificial neural networks (ANNs) to estimate ADC values.[22-24] ML can also help to address these challenges by providing more accurate and efficient methods for estimating IVIM and kurtosis in dMRI.[25-31] ML algorithms can be trained using large datasets to understand the underlying relationships between dMRI signals and the parameters of interest. Furthermore, ANNs have the capability to discern complex relationships within dMRI signals and the ADC values (among other models). Studies[22-24] have demonstrated that ANNs yield more precise estimates than traditional fitting methods; this aligns with the central objectives of the current work.

ML algorithms have been trained to estimate the parameters of the IVIM model from dMRI signals and have provided more accurate results than traditional fitting methods.[25-27]

ML algorithms have been employed to estimate kurtosis from dMRI signals, providing accurate and efficient results.[28-31]

ML can then perform both model fitting and classification tasks in diffusion MRI. In model fitting, ML algorithms can be used to estimate the parameters of a model that best fits the diffusion MRI signal, such as the IVIM model. This estimation can be achieved using supervised learning algorithms, where the model's parameters as the target variable and the diffusion MRI signal as the input variable.

Due to these achievements, ML algorithms are also able to differentiate between healthy and diseased tissue with higher accuracy (ACC) or distinguish between different types of tissues based on the diffusion MRI signal.[30,31] This is achieved using supervised or unsupervised learning algorithms for pixel or ROI classification applying ML to dMRI, IVIM, and kurtosis model fitting and classification. The studies outlined in Theaud *et al*.[31] and Li *et al*.[32] show how deep learning (DL) techniques, such as convolutional neural networks and recurrent neural networks, can improve the ACC and efficiency of dMRI model fitting.

The use of these techniques, however, also presents challenges that need to be addressed. Recent advancements in artificial intelligence have leveraged DL methods to enhance the quality of diffusion MRI (dMRI) data for clinical applications. These methods are often tailored to the study of specific pathologies.[32] While promising, they have been associated with an increased incidence of false positives and notable biases. Another study by Ahmad *et al*.[33] proposed an automated DL pipeline for automatic artifact detection in dMRI scans, demonstrating positive results in artifact detection across a variety of datasets.

This study focuses on exploring the potential of various ML algorithms for estimating dMRI parameters in ADC and IVIM models, including kurtosis. In addition, this research aims to demonstrate the effectiveness of ML techniques in enhancing the efficiency and ACC of parameter estimation in different dMRI data sets by comparing models. The present work introduces a novel approach that refines diagnostic ACC and efficiency, providing valuable methods to assist researchers and practitioners in effectively translating complex ML algorithms into practical clinical solutions.

Materials and Methods

Theory

Supervised learning

Supervised learning involves constructing an inference function that can predict the output for new input vectors.

The selection and application of a supervised learning algorithm is a complex task that requires careful consideration of several factors. One of the main factors to consider is the heterogeneity of the data.[34] In the case of dMRI, predicting parameters such as diffusion requires the input features to be numerical. Therefore, algorithms such as linear regression, logistic regression, support vector machines (SVMs), ANNs, and k-nearest neighbors (KNN) are commonly used in this context.

In addition, it is important to consider the presence of interactions and nonlinearities in the data. If the features contribute independently to the output, algorithms based on linear functions may be suitable. However, if complex interactions between features exist, algorithms like ANN are more effective because they are specifically designed to discover these interactions. The parameter estimation in ADC and IVIM models involves a mixture of both types of functions; ADC in semi-log space is generally linear, while IVIM may or may not be linear depending on the *b*-values being studied.

Handling missing data

Addressing missing data are a critical step in analyzing dMRI. There are two primary ways to handle missing data: eliminating the specific row or completing the information through imputation.[35] A common imputation method uses the average of the known data, also known as mean imputation, and is mainly used in this study. This method efficiently assigns diffusion, pseudo-diffusion, perfusion fraction, or kurtosis to the parameters since it can be estimated by taking other known data as a reference. In addition, it allows for evaluating this mean's dispersion to assess the assignment's ACC.

Confusion matrix

The confusion matrix enables the calculation of other performance metrics such as true positive rate (sensitivity or TPR), false-positive rate (FPR), ACC, and true-negative rate (specificity or TNR). These metrics provide additional insights into the model's performance and can help identify areas for improvement.[36,37]

Receiver operating characteristic curve

The receiver operating characteristic (ROC) curve is typically represented by a graph with the TPR on the Y-axis and the FPR on the X-axis and is contained within the square $[0, 1] \times [0, 1]$. The ideal classifier would have a point at the upper left corner of the graph, representing a sensitivity of 100% (no false negatives) and a specificity of 100% (no false positives). A random classifier, on the other hand, would tend toward a diagonal line on the graph.[36,38-40]

Algorithms

Various algorithms were employed and quantitatively compared to derive the most accurate estimations of the characterization parameters. Classification algorithms were implemented to determine the presence or absence of diffusion in each voxel. Regression algorithms were utilized to ascertain the value of the diffusion parameters that characterize a voxel.

Logistic regression (LR)

This is a statistical method for binary and multiclass classification problems, also known as logit regression. It models the probabilities describing the possible outcomes of a single trial using a logistic function.[41] This model can be used as a probabilistic classifier for dMRI models.

Random forest

This is an ensemble method that combines multiple decision trees to make predictions. Each tree in the ensemble is built from a sample drawn with replacement from the training set. Random forests (RFs) achieve reduced variance by combining diverse trees, sometimes at the cost of a slight increase in bias.[42] This model can help classify the dMRI models per pixel.

Extratrees classifier

This is an ML technique that improves the predictive ACC of decision trees. It involves creating an ensemble of trees, where each tree is constructed from a random sample of the training dataset. The ETC is beneficial when overfitting is a concern and is known to provide good results in high-dimensional datasets.[42]

Extragradient boost

This is a ML algorithm used for classification and regression tasks. It combines weak predictive models, such as decision trees, to create a robust predictive model. Extragradient boost (XGB) is particularly useful when the goal is to improve the ACC of the predictions, and it is well-known for its high performance and speed.[43]

Multilayer perceptron

This is a type of neural network composed of multiple layers of interconnected "neurons." The input layer comprises a set of neurons representing the input features of the data. Each neuron in the hidden layers transforms the values from the previous layer using a weighted linear sum followed by a nonlinear activation function, such as the hyperbolic tangent function. The output layer receives the values from the last hidden layer and transforms them into output values. The model is trained using backpropagation, a method for efficiently calculating the gradients of the loss function with respect to the model's parameters. This is used to update the network weights to minimize the loss function. For classification, the cross-entropy loss function is commonly used, which gives a vector of probability estimates per sample.[44] This model is attractive since it can consider other features not considered in the standard fitting.

Support vector machines

This is a linear regression model that uses a margin of tolerance to identify the best decision boundary between data points. This margin, known as the epsilon-tube, allows for some variability in the predicted values, making it more robust to outliers and noise in the data.[45]

C-Support Vector Classification

This is a type of SVM algorithm commonly used for classification tasks. The algorithm finds the best hyperplane, a decision boundary that separates the different classes in the data. The best hyperplane is chosen by maximizing the margin, which is the distance between the hyperplane and the closest data points of each class.[46]

Kernel ridge regression (Ridge)

This is a method for solving regression problems. It is similar to Support Vector Regression (SVR) in that it uses a kernel trick to transform the data into a higher dimensional space, where a linear model is applied. However, instead of using the E-insensitive loss function used in SVR, ridge uses the squared error loss function combined with L2 regularization. This results in a model that is not sparse but can be fitted in a closed fashion and is generally faster for medium-sized datasets compared to SVR.[47]

Least absolute shrinkage and selection operator (Lasso)

The Lasso algorithm is a linear regression model that uses a regularization term, known as L1 regularization, to shrink the coefficients of less important features to zero. This results in a sparse solution, where only a subset of the input features are selected for the final model. However, Lasso can be sensitive to the scaling of the input features and can sometimes produce biased estimates.[48]

ElasticNet

This is a linear regression model that combines the L1 and L2 regularization techniques to balance the trade-off between the model's simplicity and the fitting of the training data.[41,44]

Polynomial regression (degree d) (Poly)

Poly is a type of linear regression used to model nonlinear relationships between the input features and the output variable. It is achieved by introducing polynomial terms of the input features, with the degree of the polynomial specified by the user.[49]

Dataset deployment and machine learning implementation

The experiments were conducted on a machine equipped with an AMD Ryzen 7 4700U CPU, 2.00 GHz, 16 GB of RAM, Radeon Graphics and no GPU used. The operating system was Ubuntu 20.04.6 LTS.

The implementation was done in Python 3.9,[50] using TensorFlow 2.5[51] for the ML models and Scikit-learn 0.24.2[52] for the data preprocessing and evaluation metrics.

Additional software dependencies for Python: PyDicom 2.1.2[53] (for parsing DICOM files), Nibabel 5.1.0[54] (for accessing the NIFTI format), Numpy 1.21.0,[55] Scipy 1.6.3,[56] and Matplotlib 3.4.0.[57]

The datasets used for training and testing the models consisted of four distinct sets of 3D dMRI images, each with unique characteristics [Table 1].[58-62] The first set, referred to as "MR701" (confidential), comprises self-generated images of an adult's prostate. The "IVIM" set[58,61] includes images of the human head and neck, while the "CFIN" set[58-60] features images of the human skull. The "PigBrain" set[62] contains *ex vivo* images of a benign neoplasm in a Göttingen minipig's brain. All image sets were initially presented in axial sections. Table 1 provides a summary of each image set's most pertinent characteristics. The IVIM, CFIN, and PigBrain sets can be

Table 1: Relevant characteristics of each set of images: MR701, intravoxel incoherent motion, CFIN, and PigBrain

Images set	MR701	IVIM	CFIN	PigBrain
Number of slices	10	54	19	4
Image pixel size (x, y)	160,160	156,156	96,96	64,128
b-values/(s/mm^2)	0, 20, 40, 60, 80, 100, 120, 140, 180, 200, 300, 400, 500	0, 10, 20, 30, 40, 60, 80, 100, 120, 140, 180, 200, 300, 400, 500, 600, 700, 800, 900, 1000	200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000, 2200, 2400, 2600, 2800, 3000	1075, 2475, 3069, 4000, 5911, 8181, 11,295
Parameters to determine	*D, D*, f*	*D, D*, f*	*D, K*	*D, K*
Dataset source	Confidential	Reference [61]	Reference [62]	Reference [60]

IVIM: Intravoxel incoherent motion, CFIN: Center of functionally integrative neuroscience

freely downloaded from DYPI and Hvidovre Hospital in NIFTI (.nii) and DICOM (.dcm) formats. These formats are easily manageable in Python version 3.x using the NiBabel and PyDICOM packages. Approximately 1 million data points are available in these sets for statistical analysis and subsequent algorithm training/testing processes.

Voxel parameter estimation

The initial phase of the study involves analytically calculating the parameters by fitting, which characterize the maps of diffusion (*D*), pseudo-diffusion (*D**), perfusion fraction (*f*), and kurtosis (K). These calculations serve as a conventional benchmark for comparison, enabling the evaluation of precision and computational speed against non-ML methods and providing a reference framework for the training and testing of ML models. Not all of these parameters are available in all sets of images. A simple diagram of this process is shown in Figure 1. First, a set of images corresponding to a diffusion-weighted magnetic resonance (dMRI) study with all its *b*-values is imported. Then, for each voxel of interest, a graph of the normalized signal as a function of the *b*-value is generated. Fitting these points using monoexponential and biexponential models, with or without kurtosis, allows for estimating the values of *D*, *D**, *f*, and/or *K*. The analytical computation of these parameters provides a baseline for subsequent comparisons. The resulting voxel parametrization is archived and subsequently utilized as input for the training and testing phases of the ML models, depending on the goodness of fit. The same process is repeated for all voxels of interest in each slice and through all slices in the entire study.

Voxel and dataset selection

As established in the theoretical framework, datasets with an adequate range of *b*-values are candidates for calculating the diffusion coefficient (D). Datasets with a sufficient number of *b*-values between 0 mm^2/s and 200 mm^2/s are used to determine the pseudo-diffusion coefficient (*D**) and perfusion fraction (*f*), while kurtosis (*K*) calculations require *b*-values exceeding 1000–1500 mm^2/s. This ensures each parameter is calculated from the optimal data subset described in Table 1.

Figure 2 illustrates the process of characterizing each voxel following the selection criteria similar to the one developed by Agulles-Pedros L, *et al*.[8] Initially, the normalized signal values S/S_0 of the voxel at position i, j are plotted against the *b*-values (in a semi-logarithmic space). For the MR701

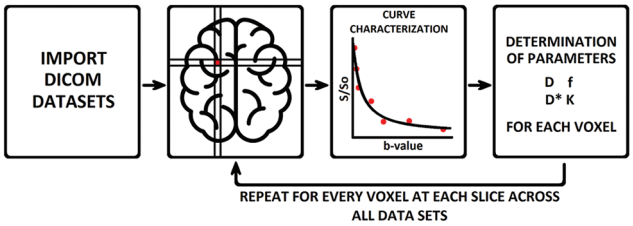

Figure 1: Flowchart for determining each voxel's characteristic parameters (diffusion, pseudo-diffusion, perfusion fraction, and kurtosis)

and IVIM datasets, sufficient self-diffusion data enable *D** and *f* calculation. In the analysis of all image sets, *D* values are calculable for all *b*-values. The mono-exponential model (Equation 1) and the bi-exponential model (Equation 2) fit *D*, *D**, and *f*, with a coefficient of determination (r^2) minimum of 0.95 required for a valid fit. High *b*-value datasets (CFIN and PigBrain) are fitted with a bi-exponential kurtosis model (Equation 3), also demanding an r^2 above 0.95. Voxel data not meeting these thresholds are omitted from the predictive model training set.

In this study, the 0.95 threshold was established for the regression analysis. This threshold corresponds to a correlation coefficient of approximately 0.975, indicating a correlation stronger than the commonly accepted threshold of 0.9 for a strong correlation.[63,64] This is not an absolute threshold, its choice may affect the results and also can vary depending on the context and nature of the data.

The r^2 values of the analyzed voxels are recorded in matrices corresponding to the dimensions of each dataset. These matrices also include the associated *D*, *D**, *f*, and *K* values, when applicable. Upon compiling these data for an entire slice, characteristic maps for each parameter can be generated. However, not all voxels have a sufficiently good coefficient of determination r^2 due to various factors. These can include unintended alterations in image acquisition, reconstruction, artifacts, and even the high threshold set for r^2. Nevertheless, parameters such as *D*, *D**, *f*, and *K* in a voxel should not be dismissed based solely on a poor fit. This is because the tissue within a voxel is expected to share a similar internal structure with its closest neighboring voxels, suggesting that these parameters could still contain relevant information about the tissue.

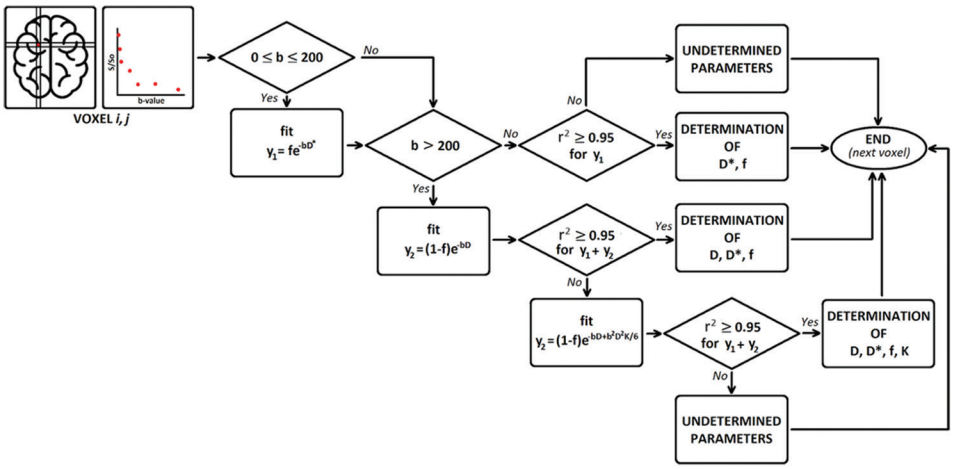

Figure 2: A flowchart that outlines the process for determining the fitting for any specific voxel. It provides a step-by-step visual guide, simplifying the understanding of the fitting determination procedure

To address this issue, a process referred to as a "smoothing filter" is applied. This stage assumes that a voxel should share the characteristics of its nearest neighbors. For instance, when the value of D cannot be determined in voxel i, j, the D value of its neighbors is evaluated. If D has been successfully calculated for all the neighbors, it is assumed that D also exists in voxel i, j. Similarly, the same assumption is made if a voxel has a good r^2 fit but is surrounded by voxels where D cannot be determined with a sufficiently high goodness of fit. This process is repeated until there are no isolated voxels with states that conflict with those of their neighbors, or the total size of the structure under study is not modified beyond a conservative criterion of 5%. The term "smoothing filter" is used here to describe the process of iteratively refining the voxel states to ensure consistency with their neighbors.

The resulting maps are modified-r^2-maps that consider the states of regions or sets of several voxels within the same slice and the individual voxel characteristics, as shown in Figure 3. These modified maps provide a more accurate representation of the diffusion properties of the studied tissue.

Once the modified-r^2-maps are obtained, the S/S_0 values of a single voxel and its nearest neighbors are stored in an input vector for training and testing, and the values of D, D^*, f, and K of the voxels of interest are stored in an expected output vector.

One input vector and one output vector are obtained for each voxel characterized, and they are used to train/test the ML regression models, as shown in Figure 4. Following the validation method (cross-validation), 80% of the randomly selected data are used for training the models, and the remaining 20% are used as test data[30,31] in different random packages or blocks to obtain a cross-validated root mean square error ($RMSE_{CV}$). Upon completion of the training process, each algorithm generates a prediction. These predicted outcomes are then compared with the expected output vector, which has been derived through analytical calculations. Finally, the difference between the predicted

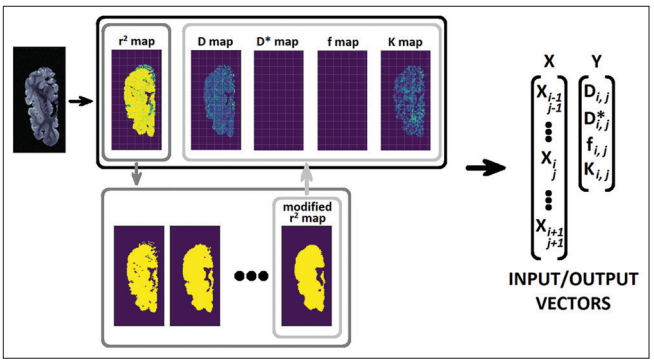

Figure 3: Illustration of the process for deriving input and output vectors from the r^2 and modified-r^2 maps. The modification is implemented through a proposed "smoothing filter," which enhances the diversity of the data quality. The figure visually depicts the derivation of these vectors, which are subsequently used for training and testing the machine learning models

and expected results is statistically evaluated using several metrics. These include mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), and the coefficient of determination (R^2) for prediction. In addition, the cross-validated root mean squared error ($RMSE_{CV}$) is used, which is a measure of the prediction error that incorporates cross-validation to provide a more robust estimate of model performance. For classification algorithms, ACC measurements from confusion matrices and the area under the ROC curves area under the curve (AUC) are used. The ROC curve is a graphical representation of the performance of a binary classifier, and AUC is a single scalar value that represents the expected performance of the classifier. A higher AUC indicates better classifier performance.

Algorithms setup and parameters

All the models mentioned in this work were implemented using the Scikit-learn library, a powerful and flexible Python library for ML. Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface. It includes the implementation of all these models: Logistic

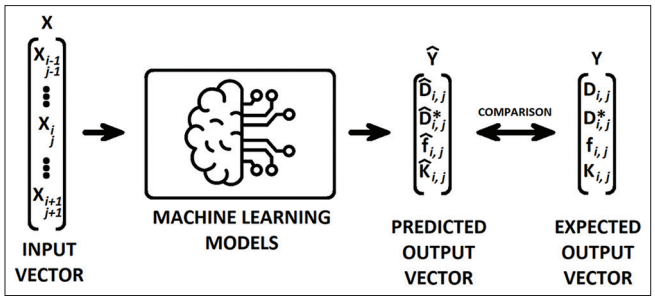

Figure 4: Comprehensive evaluation of machine learning (ML) models used in this study. The training and testing of the models are conducted using input–output vectors, which are analytically derived beforehand. The effectiveness of the models is demonstrated by comparing the predicted results with the respective ML outcomes

Regression (LG), RF, ETC, XGB, Multilayer Perceptron (MLP), SVMs, C-Support Vector Classification (SVC), ridge regression, lasso regression, ElasticNet (EN), and d-degree polynomial regression (Poly).

One of the key features of Scikit-learn is that it allows for easy hyperparameter tuning and model optimization. All parameters in scikit-learn's implementations can be directly modified when the model is instantiated or using the set_params method. This makes it straightforward to conduct hyperparameter optimization using techniques such as grid search, detailed further in this work.

The LR model was implemented with a cross-entropy loss function and an Adam optimizer. Regularization techniques such as L1 and L2 were applied to prevent overfitting obtaining better $RMSE_{CV}$ with L2. A Grid Search was conducted for hyperparameters optimization with a range of values between 0 and 10, finding an optimal "*c*-value" which is the inverse of the regularization strength of 3.4 for "L2" penalty.

The RF model was implemented using entropy as the criterion for the quality of a split. A grid search was conducted for hyperparameters optimization with a range of values between 100 and 1000 for "n_estimators," between 0 and 1 for "min_samples_split" and "min_samples_leaf," and between 1 and 100 for "max_depth." The optimal values found were 549 for "n_estimators," 0.021 for "min_samples_split," 0.029 for "min_samples_leaf," and 38 for "max_depth." "max_features" was set to "auto."

The ETC model was implemented using the default criterion for the quality of a split. A grid search was conducted for hyperparameters optimization with a range of values between 100 and 1000 for "n_estimators" and between 1 and 100 for "max_depth," finding optimal values of 284 for "n_estimators" and 83 for "max_depth" "max_features" set in "auto."

The XGB model was implemented using cross-entropy as the objective function. A grid search was conducted to optimize the hyperparameters "n_estimators," "learning_rate," and "max_depth," with a range of values between 100 and 1000 for "n_estimators," between 0.01 and 0.30 for "learning_rate," and between 1 and 10 for "max_depth." The optimal values found were 354 for "n_estimators," 0.09 for "learning_rate," and 7 for "max_depth."

The MLP model was implemented using a neural network architecture, with a cross-entropy loss function and an Adam optimizer. The dropout regularization technique was applied to prevent overfitting. The model was trained for 300 epochs with a batch size of 128. A grid search was conducted to optimize the hyperparameters "hidden_layer_sizes," "activation, and "solver," with the following ranges: (50), (100), and (50, 50) for "hidden_layer_sizes;" "relu" and "tanh" for "activation;" and "adam" and "sgd" for "solver." The optimal values found were (50, 50) for "hidden_layer_sizes," "relu" for the activation function, and "adam" for the solver.

The SVR model was implemented using an ε-insensitive loss function and a 'sigmoid' kernel function. The "C" parameter was used for regularization L2, with a range of values between 0.1 and 10 considered during a grid search. The optimal "*C*-value" found was 5.1 for a "sigmoid" kernel.

The SVC model was implemented for the classification problem using a Hinge loss function and a stochastic gradient descent optimizer. L2 regularization was applied to prevent overfitting, controlled by the "C" parameter. A grid search was conducted to optimize the "C" parameter, with a range of values between 0.1 and 10. The optimal "*C*-value" found was 3.9, using a "sigmoid" kernel.

The ridge regression model was trained using the Adam optimization algorithm with a MSE as the loss function. To mitigate overfitting, L2 regularization, an implicit regularization technique, was employed. Hyperparameter tuning was performed using a one-dimensional grid search method over a range of "alpha" values from 0 to 1. The optimal "alpha" value was found to be 0.79.

The Lasso regression model was trained using the Adam optimization algorithm with a MSE as the loss function. To mitigate overfitting, L1 regularization, an implicit regularization technique, was employed. Hyperparameter tuning was performed using a one-dimensional grid search method over a range of "alpha" values from 0 to 1. The optimal "alpha" value was found to be 0.71.

The EN model was trained using the Adam optimization algorithm with a MSE as the loss function. To mitigate overfitting, L1 regularization was employed. Hyperparameter tuning was performed using a grid search method over a range of "alpha" and "L1_ratio" values from 0 to 1. The optimal "alpha" value was found to be 0.75 and the optimal "L1_ratio" was 0.96.

The Poly model was trained using the Adam optimization algorithm with a MSE as the loss function. To mitigate overfitting, L2 regularization was employed. The "degree" hyperparameter was varied from 2 to 6 during the model

training process. The optimal degree, which resulted in the lowest ($RMSE_{CV}$), was found to be 4.

Results

Results for r^2 classification and parameters forecasting are obtained separately. Initially, binary classification algorithms are implemented to determine which voxels in each set of images might contain information on diffusion. Then, for voxels previously characterized as suitable for diffusion parameters forecasting, regression algorithms are used to determine *D*, *D**, *f*, and *K*. Finally, a comparison of execution time is conducted. This involves comparing the computation times to find the characterization parameters for the same amount of voxels. On one side, there are the classical analytical (fitting) algorithms used for finding parameters *D*, *D**, *f*, and *K*. On the other side, there are the best-trained ML forecasting methods. This approach provides a thorough evaluation of the efficiency and effectiveness of the various methods.

r^2 classification

The results of the modified r^2 classification algorithms are presented as normalized confusion matrices, as they provide a clear visual representation of the performance of each algorithm. This allows for a quick comparison of their effectiveness [Figure 5]. The main diagonal of these matrices corresponds to the correct predictions made by each algorithm. As these values are close to 1, the algorithm performs better in ACC.

In the case of binary classification, such as modified-r^2, the AUC of the ROC curves [Figure 6] is considered. This approach helps to account for any biased data. It also aids in avoiding overfitting to a single class. This is particularly useful when the goal is to optimize the relationship between false positives and true positives in a model.

Figure 6 shows the comparative ROC graph of the different models implemented. For the set of algorithms shown, ACC and AUC are summarized in Table 2. This shows the algorithms in order according to their AUC value. All models have AUC values of at least 90%; however, the best results are obtained from the ETC with 98.4% and the MLP with 96.3%.

The performance of each algorithm is indicated in Table 2, where the highest precision, reflecting the algorithms' ability to correctly classify positive cases while minimizing false positives, is observed with the multi-layer perceptron and ETC algorithm.

Forecast of *D*, *D**, *f*, and *K*

To evaluate the performance of the prediction algorithms, a variety of metrics to assess the errors in the predictions was employed, including the MAE, the mean square error (MSE), its root mean square error (RMSE), the coefficient of determination (R2), and the cross-validated RMSE ($RMSE_{CV}$).[65-67] Here, R^2, is used to assess ML models and should not be confused with r^2, which measures the fit of monoexponential or biexponential regressions to the diffusion models discussed in the theory section.

Figure 7 and Table 3 present the $RMSE_{CV}$ values for the implemented algorithms. A lighter color indicates a lower $RMSE_{CV}$, signifying better algorithm performance; a darker color represents a higher $RMSE_{CV}$ and less efficient performance. It is worth noting that the color scale applies only within the same group and does not indicate any relationship between different figures.

Table 3 shows the numerical results of the calculated errors, ordered according to the $RMSE_{CV}$. All values were normalized to unity, and the regression metrics were calculated with respect to the expected error when predicting the values of diffusion, pseudo-diffusion, perfusion fraction, and kurtosis. It is important to note that the values of MAE, MSE, RMSE, and $RMSE_{CV}$ were normalized to the maximum values: *D* (4.88×10^{-3} mm²/s), *D** (9.89×10^{-2} mm²/s), *K* (1.95), and the perfusion fraction *f* (0.91), respectively. Although the free water auto-diffusion coefficient at body

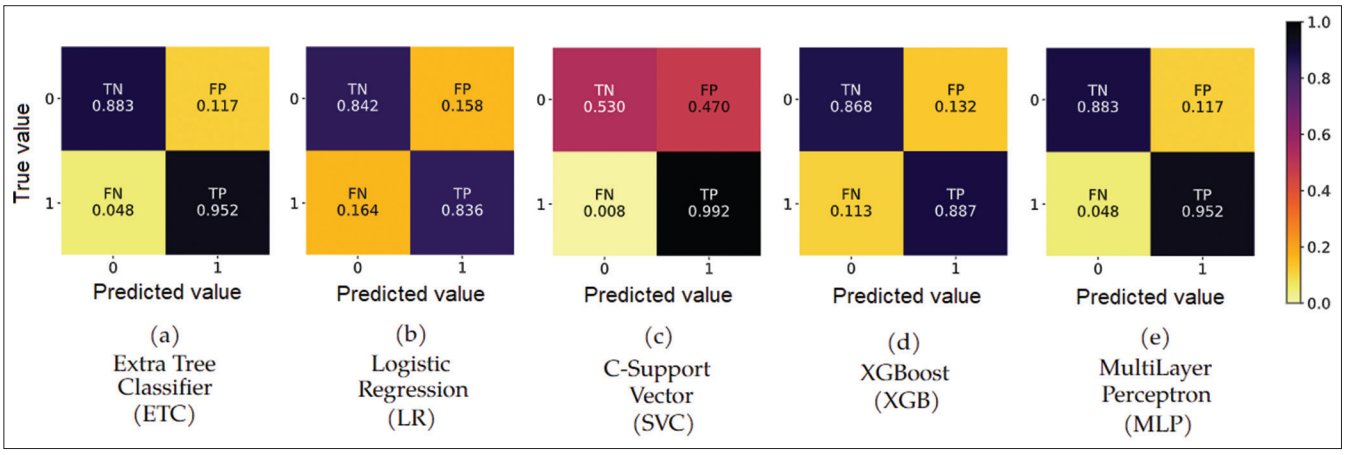

Figure 5: Normalized confusion matrices of the algorithms implemented for classification based on the r^2 value. Color code is used for faster visual analysis. ETC: Extra-tree classifier, LR: Logistic regression, SVC: C-Support Vector Classification, XGB: Extragradient boost, MLP: Multilayer perceptron, TN: True negative, TP: True positive, FP: False positive, FN: False negative

temperature is approximately 3×10^{-3} mm^2/s, some D values may be larger due to fitting and algorithm characteristics. However, it was decided not to set a strong restriction below the auto-diffusion coefficient, as this may have artificially shifted the results.

Processing time

The time required to determine the parameters D, D^*, f, and K for 100,000 voxels using conventional fitting algorithms was calculated and is presented in Table 4. In contrast, Table 4 shows the computation time of the ML methods presented in this study after they have been trained to individually find each of these parameters in the same number of voxels. This is because ML methods are designed to efficiently locate one parameter at a time from a voxel of any slice or image set. The results were obtained from the average processing time of 100,000 random voxels analysis, and the process was repeated 10 times; the standard deviation was calculated from this average.

Table 2: Accuracy and area under the curve values obtained for the implemented algorithms, sorted according to the area under the curve value

Algorithm	ACC	AUC
ETC	0.941	0.984
MLP	0.917	0.963
XGB	0.838	0.939
LR	0.877	0.921
SVC	0.761	0.900

The table uses color coding as a visual guide to algorithm performance: blue font indicates superior performance, while red numbers indicate less optimal outcomes

When considering all image sets, the average calculation time for the four main parameters using conventional methods is 4408 ± 351 s. However, once trained, the computation time for 100,000 voxels using the proposed ML algorithms to

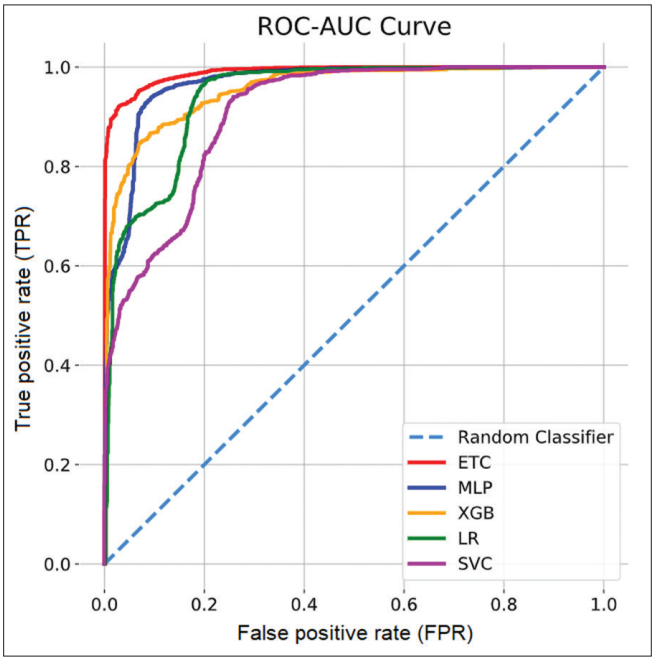

Figure 6: Receiver operating characteristic curves of the implemented classification algorithms for the classification of r^2, compared with a random classifier. ETC: Extra-tree classifier, ROC: Receiver operating characteristic, AUC: Area under the curve, LR: Logistic regression, SVC: C-Support Vector Classification, MLP: Multilayer perceptron, XGBL: ExtraGradient Boost

Table 3: Mean absolute error, mean square error, root mean square error, regression coefficient of determination, and root mean square error cross-validated of each of the algorithms implemented for the (top left) diffusion, (top right) pseudo-diffusion, (bottom left) perfusion fraction and (bottom right) kurtosis data

Model	MAE	MSE	RMSE	R^2	$RMSE_{cv}$
RF	0.0462	0.0071	0.0845	0.292	0.0843
XGB	0.0465	0.0073	0.0856	0.274	0.0858
Ridge	0.0542	0.0084	0.0917	0.166	0.0942
LinR	0.0543	0.0084	0.0918	0.166	0.0942
Poly. (d=2)	0.0577	0.0160	0.1266	0.587	0.0942
Lasso	0.0512	0.0101	0.1004	0.00003	0.1032
ElasticNet	0.0512	0.0100	0.1005	0.00003	0.1032
SVR	0.7430	0.0125	0.1118	0.239	0.1148

Model	MAE	MSE	RMSE	R^2	$RMSE_{cv}$
RF	0.0159	0.00117	0.0342	0.973	0.0357
XGB	0.0213	0.00149	0.0385	0.965	0.0402
LinR	0.0382	0.00314	0.0560	0.926	0.0571
Poly. (d=2)	0.0166	0.00112	0.0335	0.974	0.0571
Ridge	0.0385	0.00313	0.0560	0.926	0.0572
SVR	0.0458	0.00580	0.0761	0.864	0.0727
Lasso	0.118	0.0426	0.2064	0.000006	0.2165
ElasticNet	0.117	0.0426	0.2064	0.000005	0.2165

Model	MAE	MSE	RMSE	R^2	$RMSE_{cv}$
RF	0.0182	0.0023	0.0477	0.766	0.0454
XGB	0.0024	0.0024	0.049	0.753	0.0465
Ridge	0.0337	0.0034	0.0585	0.648	0.0555
LinR	0.0337	0.0034	0.0585	0.647	0.0555
Poly. (d=2)	0.0242	0.0024	0.0491	0.751	0.0555
SVR	0.0477	0.0042	0.0645	0.571	0.0651
Lasso	0.0542	0.0097	0.0986	0.0004	0.0991
ElasticNet	0.0542	0.0097	0.0987	0.0003	0.0992

Model	MAE	MSE	RMSE	R^2	$RMSE_{cv}$
RF	0.0186	0.0013	0.0355	0.980	0.0352
XGB	0.0197	0.0012	0.0351	0.980	0.0353
SVR	0.0364	0.0031	0.0555	0.951	0.0577
Ridge	0.0368	0.0034	0.0581	0.946	0.0595
LinR	0.0368	0.0034	0.0581	0.946	0.0595
Poly. (d=2)	0.0174	0.0013	0.0366	0.979	0.0595
Lasso	0.161	0.063	0.0251	0.0001	0.2522
ElasticNet	0.161	0.062	0.0251	0.0001	0.2522

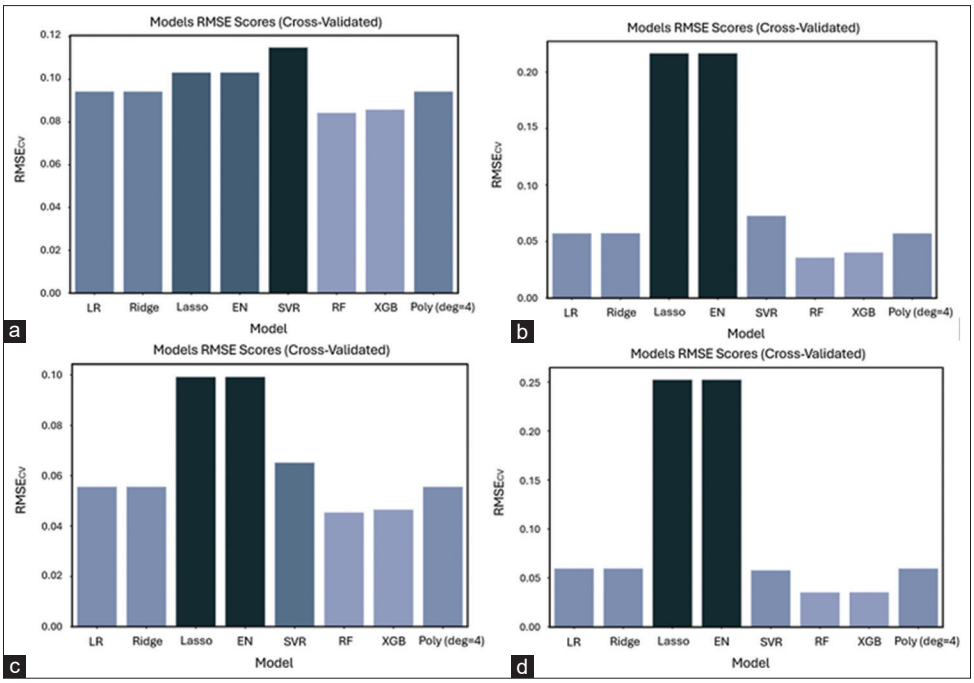

Figure 7: Root mean square error cross-validated scores for every algorithm implemented for the prediction of (a) diffusion coefficient, (b) pseudo-diffusion coefficient, (c) perfusion fraction, and (d) kurtosis. LR: Logistic regression, SVR: Support vector regression, RF: Random forest, XGB: Extragradient boost, EN: ElasticNet, RMSE: Root mean square error

Table 4: Average computation times for characterization parameters of 100,000 voxels in each image set using conventional fitting methods, and average processing times for diffusion, pseudo-diffusion, perfusion fraction, and kurtosis computation for 100,000 voxels using machine learning methods

Set of images	Average processing time of 100.000 voxels
MR701	5984 (394) s≈99 (6) min
IVIM	4512 (428) s≈75 (7) min
CFIN	3416 (301) s≈57 (5) min
PigBrain	3723 (279) s≈62 (5) min
Parameter	**Average processing time of 100.000 voxels**
Diffusion (*D*)	4.835 (28) s
Pseudo-diffusion (*D**)	5.632 (53) s
Perfusion fraction (*f*)	4.545 (42) s
Kurtosis (*K*)	3.986 (12) s

determine the four parameters was significantly reduced to 18.998 ± 0.135 s, making the faster ML algorithms

Analysis

While all models have AUC values of at least 90%, the ACC of the algorithms varies, mainly for the SVC model, being below 80% [Table 2]. Notably, although the XGB model has an AUC value of 93.9%, its precision is lower than that of the LR model, with 83.8% and 87.7%, respectively. This discrepancy could be attributed to the inherent differences in the algorithms. For instance, XGB, a gradient-boosting model, is designed to sequentially correct the mistakes of the weak learners in the ensemble, thereby optimizing the overall model performance. This optimization process, which focuses on reducing both bias and variance, might lead to a higher AUC. However, this could come at the cost of ACC, especially in cases where the data distribution is imbalanced. The model might become too complex, fitting not only the signal but also the noise in the data, which could lead to a higher number of false positives and false negatives, thereby reducing the ACC. On the other hand, LR, being a simpler model, might be less prone to overfitting, leading to better ACC. LR makes fewer assumptions about the data and is less likely to model the noise, especially when the problem is linearly separable or nearly linearly separable. Therefore, even though its AUC might be lower than that of XGB, its ACC could be higher as it might make fewer prediction errors. Given the thorough analysis of the data, the observed performance differences among the models can be attributed to the inherent characteristics and assumptions of each model.

Even though the absolute values of *D*, *D**, *f*, and *K* are the ultimate goal of dMRI, their estimation lies beyond the scope of this work, as it aims to test and compare the use of ML algorithms against conventional fitting. Note that the dataset used in this work covers different anatomical regions and even distinct species, which makes the absolute values less valuable. The importance of the dataset lies in the wide range of *b*-values and the differences in anatomical regions and species, i.e., different microstructures. However, it is important to mention that realistic results have been obtained, as this represents the initial step in validating the results.

Among the models implemented in this work, ADC maps present more consistent diffusion values in terms of the small dispersion of the data for magnitude D [Figure 7a]. On the other hand, according to the theory and literature,[2-4] it is verified that the pseudo-diffusion D^* presents values of an order of magnitude larger than D. However, it is a parameter with a large range of values since some voxels were estimated significantly outside this bounds due to practical acquisition methods of the images. The perfusion fraction f, as expected and according to its definition, yields values in the interval between 0 and 1, although with a skewed distribution to the left and a mode around 0.22. Finally, kurtosis K has been estimated only for image sets with b-values large enough to detect the effect ($b \geq 3000$ s/mm^2). The distribution of these values varies between images, being close to a Gaussian behavior with a mean value of around 1 for the PigBrain set. While the CFIN set's mean is also around 1, the data are slightly biased to larger values (up to 1.5).

The predicted diffusion coefficients, D, are relatively consistent among the various algorithms tested since fewer missing values exist in the image sets used for this parameter [Figure 7a]. However, the other predicted parameters, such as perfusion fraction, f, pseudo-diffusion, D^*, and kurtosis, K, show a large variability [Figure 7b-d] since these parameters have more missing values across the image sets. Some algorithms may perform better when less data are available, while others may require more data to achieve accurate results.

The RF and XGB algorithms exhibit lower $RMSE_{CV}$ values than the others, indicating that they are the most effective for predicting D, D^*, f, and K [Table 3 and Figure 7]. The effectiveness of both these algorithms in predicting diffusion parameters as demonstrated in this study is in line with the findings of other studies.[68,69] While these referenced studies validate the utility of RF and XGB in MRI image classification tasks, they do not focus on the prediction of the specific diffusion parameters. Therefore, the results of this study not only corroborate the effectiveness of these algorithms but also extend their applicability in diffusion MRI analysis, particularly in predicting specific diffusion parameters.

Conversely, the EN and Lasso algorithms exhibit the highest errors, making them less suitable for predicting pseudo-diffusion, perfusion, and kurtosis values [Figure 7b-d].

Regarding [Table 3], the results of the MAE regression metric indicate the error or difference that would be obtained after predicting a value between 0 and 1, while the MSE measures the deviation from said prediction. The RMSE represents the estimation residuals and is typically slightly higher than the MAE, as it is more sensitive to outliers and considers potential biases. In addition, the $RMSE_{CV}$ is expected to be similar to the RMSE, as it averages these errors over multiple iterations of the same training set with different samples. Lastly, the coefficient of determination (R^2) demonstrates the goodness of fit of the tested methods. However, it should be noted that even if the other errors are minor, this value may not be an accurate indicator if the training data are dispersed in relation to the prediction function.

Discussion

The experimental characterization of the signal decay in individual voxels is the first step in determining the parameters that physically describe a dMRI, as it allows for the verification and quantification of the physical phenomenon. The following models were considered to characterize these decay curves: the ADC model (mono-exponential) and the IVIM model (bi-exponential); the latter as a simple model and with kurtosis. Each model allows for determining the parameters of interest (D, D^*, f, and K) using linear or polynomial regressions voxel by voxel for four distinct image sets. It is important to note that while a correlation coefficient of 0.9 is generally accepted in the literature as indicative of a strong correlation, the square of this value (0.81) would be considered a strong fit in terms of the r^2. However, given the quality of the data in this study and the need for a robust fit for the predictions, a more stringent threshold of 0.95 for r^2 was deemed appropriate. This choice can influence results and is context-dependent.

Various factors, including image acquisition alterations, reconstruction, artifacts, and a high r^2 threshold, can result in insufficient r^2 coefficients for some voxels. Parameters such as D, D^*, f, and K in a voxel should not be disregarded due to poor fit, as they may still hold relevant tissue information due to expected structural similarity with neighboring voxels. This is the reason why the implementation of smoothing filters on modified-r^2 images is also an important step in the preprocessing of the images since inhomogeneities may be introduced in the information and the training data. This can happen when considering data that would have been discarded initially due to a lack of precision in the regression. On the other hand, the smoothing filters corrected possible errors that occurred during the characterization of the attenuation curves of the voxels.[70]

Classification algorithms have been implemented to determine where the diffusion can be found based on modified r^2 maps and where D, D^*, f, or K can be found with a confidence level equal to or >90% [Figures 5 and 6]. Among the classification algorithms tested, the ETC methods and MLP reported the best precision results (94.1% and 91.7%, respectively) and area under the ROC curve test (98.4% and 96.3%, respectively), [Table 2]. This work's results demonstrate the effectiveness of both the ETC and MLP algorithms in MRI classification tasks, aligning with findings from Soltaninejad *et al*.[71] and Yun *et al*.,[72] respectively. However, it is important to note that these studies focused on different tasks within the realm of MRI classification, and did not specifically address the identification of diffusion parameters D, D^*, f, and K. Despite these differences, the robust performance of ETC and MLP in predicting these specific parameters in this study underscores their potential applicability in advancing diffusion MRI analysis.

Regarding the regression algorithms for D, D^*, f, and K, RF, and XGB have been, in all cases, the methods with the lowest errors with respect to the $RMSE_{CV}$ for a k-fold division of 5 samples [Figure 7].

Finally, the computing time of ML algorithms is estimated to find the values of the parameters D, D^*, f, and K versus the time it takes to find these parameters using conventional methods, over 100,000 voxels. Using conventional methods, this time is 4408 ± 351 s while ML is 18.998 ± 0.135 s [Table 4]. It should be noted that the ML computation time has been calculated after training. The training time has not been computed since it strongly varies among methods and depends on the training data set and computer capabilities.

Conclusions

This work tested some of the most commonly used models for characterizing diffusion.[19] However, other models allow for the detection and differentiation of various effects. For instance, the characterization of attenuation through the triexponential model[73] or through "signal spoiling"[74] could be considered. In addition, to improve the precision of the parameters by conventional methods, the effect of noise on the images should be taken into account.[75]

Several ML algorithms can be used for model fitting and classification in MRI, including LR,[76] RF,[77] SVR,[77] MLP,[78] and ETC.[79] The choice of algorithm will depend on the nature of the data, the complexity of the model, and the desired ACC of the results.

While the Akaike Information Criterion (AIC) was initially considered for comparing the implemented algorithms,[80-82] it was ultimately not used due to its limitations.[83] These include its assumptions of normally distributed model errors and linearity in parameters, which may not always hold true for all the models. In addition, AIC does not account for the computational complexity of models, interpretability of models, or provide a measure of uncertainty in model estimation.[84,85] There is also a risk of overfitting as AIC can favor more complex models, and it may not be suitable for large datasets as it tends to select more complex models as dataset size increases.[86] Therefore, these limitations led to the decision not to estimate AIC in this study.

It is important to note that model fitting and classification in dMRI can be challenging due to the complex relationships between the MRI signal and the underlying biological structure. As a result, careful preprocessing of the data and thorough evaluation of the results is crucial to ensure that the ML algorithms provide accurate and reliable results.

Financial support and sponsorship

Nil.

Conflicts of interest

There are no conflicts of interest.

References

1. Tax CM, Bastiani M, Veraart J, Garyfallidis E, Okan Irfanoglu M. What's new and what's next in diffusion MRI preprocessing. Neuroimage 2022;249:118830.
2. Le Bihan D. What can we see with IVIM MRI? Neuroimage 2019;187:56-67.
3. Le Bihan D, Breton E, Lallemand D, Grenier P, Cabanis E, Laval Jeantet M. MR imaging of intravoxel incoherent motions: Application to diffusion and perfusion in neurologic disorders. Radiology 1986;161:401-7.
4. Rosenkrantz AB, Padhani AR, Chenevert TL, Koh DM, De Keyzer F, Taouli B, *et al.* Body diffusion kurtosis imaging: Basic principles, applications, and considerations for clinical practice. J Magn Reson Imaging 2015;42:1190-202.
5. Jensen JH, Helpern JA, Ramani A, Lu H, Kaczynski K. Diffusional kurtosis imaging: The quantification of non-Gaussian water diffusion by means of magnetic resonance imaging. Magn Reson Med 2005;53:1432-40.
6. Maiter A, Riemer F, Allinson K, Zaccagna F, Crispin Ortuzar M, Gehrung M, *et al.* Investigating the relationship between diffusion kurtosis tensor imaging (DKTI) and histology within the normal human brain. Sci Rep 2021;11:8857.
7. Bingbing G, Yujing Z, Yanwei M, Chunbo D, Weiwei W, Shiyun T, *et al.* Diffusion kurtosis imaging of microstructural changes in gray matter nucleus in Parkinson disease. Front Neurol 2020;11:252.
8. Agulles-Pedros L, Humm J, Sala E, Vargas H, Mazaheri Y. Error Reduction in Parameter Estimation from the Segmented Intravoxel Incoherent Motion Bi-exponential Model. J Clin Res Radiol 2018;1:10101. doi: 10.33309/2639-913X.010101.
9. Hu YC, Yan LF, Han Y, Duan SJ, Sun Q, Li GF, *et al.* Can the low and high b-value distribution influence the pseudodiffusion parameter derived from IVIM DWI in normal brain? BMC Med Imaging. 2020;20:14. doi: 10.1186/s12880-020-0419-0. PMID: 32041549; PMCID: PMC7011602.
10. Perucho JA, Chang HC, Vardhanabhuti V, Wang M, Becker AS, Wurnig MC, *et al.* B-value optimization in the estimation of intravoxel incoherent motion parameters in patients with cervical cancer. Korean J Radiol 2020;21:218-27.
11. Chuhutin A, Hansen B, Jespersen SN. Precision and accuracy of diffusion kurtosis estimation and the influence of b-value selection. NMR Biomed 2017;30:10.1002/nbm.3777. doi: 10.1002/nbm.3777. Epub 2017 Aug 25. PMID: 28841758; PMCID: PMC5715207.
12. Liu Y, Wang X, Cui Y, Jiang Y, Yu L, Liu M, *et al.* Comparative study of monoexponential, intravoxel incoherent motion, kurtosis, and IVIM-kurtosis models for the diagnosis and aggressiveness assessment of prostate cancer. Front Oncol 2020;10:1763.
13. Callaghan PT. Principles of Nuclear Magnetic Resonance Microscopy. Oxford, UK: Oxford University Press; 1993.
14. Stejskal EO, Tanner JE. Spin diffusion measurements: Spin echoes in the presence of a time-dependent field gradient. J Chem Phys 1965;42:288-92.
15. Tang L, Zhou XJ. Diffusion MRI of cancer: From low to high b-values. J Magn Reson Imaging 2019;49:23-40.
16. Le Bihan D, Iima M. Diffusion magnetic resonance imaging: What water tells us about biological tissues. PLoS Biol 2015;13:e1002203.
17. Palumbo P, Martinese A, Antenucci MR, Granata V, Fusco R, De Muzio F, *et al.* Diffusion kurtosis imaging and standard diffusion imaging in the magnetic resonance imaging assessment of prostate cancer. Gland Surg 2023;12:1806-22.
18. Żurowska A, Pęksa R, Grzywińska M, Panas D, Sowa M, Skrobisz K, *et al.* Comparison of diffusion kurtosis imaging and standard mono-exponential apparent diffusion coefficient in diagnosis of significant prostate cancer-a correlation with gleason score assessed on whole-mount histopathology specimens. Diagnostics (Basel) 2023;13:173.
19. Avanzo M, Trianni A, Botta F, Talamonti C, Stasi M, Iori M. Artificial intelligence and the medical physicist: Welcome to the machine. Appl Sci 2021;11:1691.

20. El Naqa I, Das S. The role of machine and deep learning in modern medical physics. Med Phys 2020;47:125-6.
21. Hadjiiski L, Cha K, Chan HP, Drukker K, Morra L, Näppi JJ, *et al.* AAPM task group report 273: Recommendations on best practices for AI and machine learning for computer-aided diagnosis in medical imaging. Med Phys 2023;50:e1-24.
22. Lim JP, Blumberg SB, Narayan N, Epstein SC, Alexander DC, Palombo M, *et al* (2022). Fitting a directional microstructure model to diffusion-relaxation MRI data with self-supervised machine learning. In Computational Diffusion MRI (pp. 77-88). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-21206-2_7.
23. Novak J, Zarinabad N, Rose H, Arvanitis T, MacPherson L, Pinkey B, *et al.* Classification of paediatric brain tumours by diffusion weighted imaging and machine learning. Sci Rep 2021;11:2987.
24. Chen G, Hong Y, Huynh KM, Yap PT. Deep learning prediction of diffusion MRI data with microstructure-sensitive loss functions. Med Image Anal 2023;85:102742.
25. Barbieri S, Gurney Champion OJ, Klaassen R, Thoeny HC. Deep learning how to fit an intravoxel incoherent motion model to diffusion-weighted MRI. Magn Reson Med 2020;83:312-21.
26. Ye C, Xu D, Wang L, Wang R, Zhu Y. Application of machine learning in optimizing b-value acquisition strategy of diffusion Magnetic Resonance Imaging. J Phys Conf Ser 2020;1682:012005. doi: 10.1088/1742-6596/1682/1/012005.
27. Bertleff M, Domsch S, Weingärtner S, Zapp J, O'Brien K, Barth M, *et al*. Diffusion parameter mapping with the combined intravoxel incoherent motion and kurtosis model using artificial neural networks at 3 T. NMR Biomed 2017;30:10.1002/nbm.3833. doi: 10.1002/nbm.3833. Epub 2017 Sep 28. PMID: 28960549.
28. Jiang L, Zhou L, Ai Z, Xiao C, Liu W, Geng W, *et al.* Machine learning based on diffusion kurtosis imaging histogram parameters for glioma grading. J Clin Med 2022;11:2310.
29. Tong Q, Gong T, He H, Wang Z, Yu W, Zhang J, *et al*. A deep learning-based method for improving reliability of multicenter diffusion kurtosis imaging with varied acquisition protocols. Magn Reson Imaging 2020;73:31-44.
30. Theaud G, Edde M, Dumont M, Zotti C, Zucchelli M, Deslauriers Gauthier S, *et al.* DORIS: A diffusion MRI-based 10 tissue class deep learning segmentation algorithm tailored to improve anatomically-constrained tractography. Front Neuroimaging 2022;1:917806.
31. Li Z, Fan Q, Bilgic B, Wang G, Wu W, Polimeni JR, *et al.* Diffusion MRI data analysis assisted by deep learning synthesized anatomical images (DeepAnat). Med Image Anal 2023;86:102744.
32. Aja Fernández S, Martín Martín C, Planchuelo Gómez A, Faiyaz A, Uddin N, Schifitto G, *et al*. Validation of deep learning techniques for quality augmentation in diffusion MRI for clinical studies. NeuroImage Clin 2023;39:103483.
33. Ahmad A, Parker D, Dheer S, Samani ZR, Verma R. 3D-QCNet – A pipeline for automated artifact detection in diffusion MRI images. Comput Med Imaging Graph 2023;103:102151.
34. Mehryar M, Afshin R, Ameet T. Foundations of Machine Learning. 2nd ed. Massachusetts, USA: The MIT Press; 2018.
35. Brodley CE, Friedl MA. Identifying mislabeled training data. J Artif Intell Res 2011;11:131-67.
36. Powers D. Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. Mach Learn Technol 2008;2.
37. Machine Learning Crash Course: Clasificación: Curva ROC y AUC; 2022. Available from: https://www.developers.google.com/machine-learning/crash-course/classification/roc-and-auc?hl=es-419. [Last retrieved on 2022 Dec 03].
38. del Valle Benavides AR. Curvas ROC (Receiver-Operating-Characteristic) y sus aplicaciones. [master's thesis]. Universidad de Sevilla; 2017. Seville, Spain.
39. Zou KH, O'Malley AJ, Mauri L. Receiver-operating characteristic analysis for evaluating diagnostic tests and predictive models. Circulation 2007;115:654-7.
40. Brown CD, Davis HT. Receiver operating characteristics curves and related decision measures: A tutorial. Chemom Intell Lab Syst 2006;80:24-38. doi: 10.1016/j.chemolab.2005.05.004.
41. Scikit-Learn. (2007-2023). Machine Learning in Python: Linear Models. Available from: https://www.scikit-learn.org/stable/modules/linear_model.html. [Last retrieved on 2023 Jan 17].
42. Scikit-Learn. (2007-2023). Machine Learning in Python: Ensembles. Available from: https://www.scikit-learn.org/stable/modules/ensemble.html. [Last retrieved on 2023 Jan 17].
43. InteractiveChaos-Educational Resources. (n.d.). Tutorial de Machine Learning: Gradient Boosting. Available from: https://www.interactivechaos.com/es/manual/tutorial-de-machine-learning/gradient-boosting. [Last retrieved on 2023 Jan 17].
44. Scikit-Learn. Machine Learning in Python: Neural Network Models; 2023. Available from: https://www.scikit-learn.org/stable/modules/neural_networks_supervised.html. [Last retrieved on 2023 Jan 17].
45. Scikit-Learn. (2023). Machine Learning in Python: Support Vector Machines. Available from: https://www.scikit-learn.org/stable/modules/svm.html [Last retrieved on 2023 January 17]
46. Novakovic J, Veljovic A. C-support vector classification: Selection of kernel and parameters in medical diagnosis. In: 2011 IEEE 9th International Symposium on Intelligent Systems and Informatics. Subotica, Serbia: IEEE; 2011. p. 465-70.
47. Scikit-Learn. Machine Learning in Python: Kernel Ridge Regression. 2023. Available from: https://www.scikit-learn.org/stable/modules/kernel_ridge.html. [Last retrieved on 2023 Jan 17].
48. Kim SJ, Koh K, Lustig M, Boyd S, Gorinevsky D. An interior-point method for large-scale ℓ1-regularized least squares. IEEE J Sel Top Signal Process 2007;1:606-17.
49. Scikit-Learn. Machine Learning in Python: Polynomial Features; 2023. Available from: https://www.scikit-learn.org/stable/modules/preprocessing.html#polynomial-features. [Last retrieved on 2023 Jan 17].
50. Python Software Foundation. Python 3.9.0 Documentation; 2021. Available from: https://docs.python.org/release/3.9.0/index.html. [Last retrieved on 2024 Mar 24].
51. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, *et al*. TensorFlow: Large-scale machine learning on heterogeneous systems. TensorFlow; 2015. Available from: https://www.tensorflow.org/. [Last retrieved on 2024 Mar 24].
52. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, *et al*. Scikit-learn: Machine learning in Python. J Mach Learn Res 2011;12:2825-30. Available from: https://www.scikit-learn.org/stable/. [Last retrieved on 2024 Mar 24].
53. Mason D, Contributors P. PyDicom: An Open Source DICOM library; 2008. Available from: https://www.pydicom.github.io/pydicom/. [Last retrieved on 2024 Mar 24].
54. Brett M, Markiewicz CJ, Hanke M, Côté MA, Cipollini B, McCarthy P, *et al.* nipy/nibabel: 3.2.1. Zenodo; 2020. Available from: https://www.nipy.org/nibabel/. [Last retrieved on 2024 Mar 24].
55. Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, *et al*. Array programming with NumPy. Nature 2020;585:357-62. Available from: https://www.numpy.org/. [Last retrieved on 2024 Mar 24].
56. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, *et al*. SciPy 1.0: Fundamental algorithms for scientific computing in Python. Nat Methods 2020;17:261-72. Available from: https://www.scipy.org/. [Last retrieved on 2024 Mar 24].
57. Hunter JD. Matplotlib: A 2D graphics environment. Comp Sci Eng 2007;9:90-5. Available from: https://www.matplotlib.org/. [Last retrieved on 2024 Mar 24].
58. Garyfallidis E, Brett M, Amirbekian B, Rokem A, van der Walt S, Descoteaux M, Nimmo-Smith I; Dipy Contributors. Dipy, a library for the analysis of diffusion MRI data. Front Neuroinform 2014;8:8. doi: 10.3389/fninf.2014.00008. PMID: 24600385; PMCID: PMC3931231.
59. Henriques R, Rokem A, Garyfallidis E, St-Jean S, Peterson E, Correia M. [Re] Optimization of a free water elimination two-compartment model for diffusion tensor imaging. ReScience 2017;3:1-8. doi: 10.5281/zenodo.495237.
60. Dyrby TB, Baaré WF, Alexander DC, Jelsing J, Garde E, Søgaard LV. An *ex vivo* imaging pipeline for producing high-quality and high-resolution

diffusion-weighted imaging datasets. Hum Brain Mapp 2011;32:544-63.
61. Peterson E. IVIM dataset. 2016 May. doi: 10.6084/m9.figshare.3395704.v1. Available from: https://figshare.com/articles/dataset/IVIM_dataset/3395704.
62. Hansen B, Jespersen SN. Data for evaluation of fast kurtosis strategies, b-value optimization and exploration of diffusion MRI contrast. Sci Data 2016;3:160072.
63. Akoglu H. User's guide to correlation coefficients. Turk J Emerg Med 2018;18:91-3.
64. Schober P, Boer C, Schwarte LA. Correlation coefficients: Appropriate use and interpretation. Anesth Analg 2018;126:1763-8.
65. DeGroot MH, Schervish MJ. Probability and Statistics. 4th ed., Vol. 1. Addison-Wesley, London, Toronto, Boston; 2011.
66. Mood A, Graybill F, Boes D. Introduction to the Theory of Statistics. New York, USA: McGraw-Hill; 1974.
67. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, *et al*. Scikit-learn: Machine learning in python. J Mach Learn Res 2011;12:2825-30.
68. Alexander DC, Zikic D, Zhang J, Zhang H, Criminisi A. Image quality transfer via random forest regression: Applications in diffusion MRI. In: Golland P, Hata N, Barillot C, Hornegger J, Howe R, editors. Medical Image Computing and Computer-assisted Intervention – MICCAI 2014. Boston, MA, USA: Springer; 2014. p. 231-8.
69. Tang W, Zhou H, Quan T, Chen X, Zhang H, Lin Y, *et al.* XGboost prediction model based on 3.0T diffusion kurtosis imaging improves the diagnostic accuracy of MRI BiRADS 4 Masses. Front Oncol 2022;12:833680.
70. Prieto-Gonzalez LS. Análisis de Modelos de Difusión por Imágenes de Resonancia Magnética Nuclear con Machine Learning. [master's thesis]. Universidad Nacional de Colombia; 2022. Bogotá, Colombia.
71. Soltaninejad M, Yang G, Lambrou T, Allinson N, Jones TL, Barrick TR, *et al*. Automated brain tumor detection and segmentation using superpixel-based extremely randomized trees in FLAIR MRI. Int J Comput Assist Radiol Surg 2017;12:183-203.
72. Yun J, Park JE, Lee H, Ham S, Kim N, Kim HS. Radiomic features and multilayer perceptron network classifier: A robust MRI classification strategy for distinguishing glioblastoma from primary central nervous system lymphoma. Sci Rep 2019;9:5746.
73. Zeng Q, Shi F, Zhang J, Ling C, Dong F, Jiang B. A modified tri-exponential model for multi-b-value diffusion-weighted imaging: A method to detect the strictly diffusion-limited compartment in brain. Front Neurosci 2018;12:102.
74. Corbin N, Callaghan MF. Imperfect spoiling in variable flip angle T (1) mapping at 7T: Quantifying and minimizing impact. Magn Reson Med 2021;86:693-708.
75. Basu S, Fletcher T, Whitaker R. Rician noise removal in diffusion tensor MRI. Med Image Comput Comput Assist Interv 2006;9:117-25.
76. Gajula S, Rajesh V. An MRI brain tumor detection using logistic regression-based machine learning model. Int J Syst Assur Eng Manage 2024;15:124-34.
77. O'Dwyer L, Lamberton F, Bokde AL, Ewers M, Faluyi YO, Tanner C, *et al.* Using support vector machines with multiple indices of diffusion for automated classification of mild cognitive impairment. PLoS One 2012;7:e32441.
78. Joo Y, Namgung E, Jeong H, Kang I, Kim J, Oh S, *et al.* Brain age prediction using combined deep convolutional neural network and multi-layer perceptron algorithms. Sci Rep 2023;13:22388.
79. Chen Z, Pawar K, Ekanayake M, Pain C, Zhong S, Egan GF. Deep learning for image enhancement and correction in magnetic resonance imaging-state-of-the-art and challenges. J Digit Imaging 2023;36:204-30.
80. Jerome NP, Periquito JS. Analysis of renal diffusion-weighted imaging (dwi) using apparent diffusion coefficient (ADC) and intravoxel incoherent motion (IVIM) models. Methods Mol Biol 2021;2216:611-35.
81. Ngrisch M, Sourbron S, Reiser MF, Peller M. Model selection in dynamic contrast enhanced MRI: The akaike information criterion. In: Dössel O, Schlegel WC, editors. World Congress on Medical Physics and Biomedical Engineering, September 7 – 12, 2009, Munich, Germany. IFMBE Proceedings, Vol 25/4. Berlin, Heidelberg: Springer; 2009.
82. Zhang X, Johnson TD, Little RJ, Cao Y. Quantitative magnetic resonance image analysis via the em algorithm with stochastic variation. Ann Appl Stat 2008;2:736-5.
83. Cavanaugh JE, Neath AA. The Akaike information criterion: Background, derivation, properties, application, interpretation, and refinements. WIREs Comput Stat. 2019;11:e1460. doi: 10.1002/wics.1460.
84. Myung JI, Tang Y, Pitt MA. Evaluation and comparison of computational models. Methods Enzymol 2009;454:287-304.
85. Rigdon E, Sarstedt M, Moisescu O. Quantifying model selection uncertainty via bootstrapping and akaike weights. Int IJC 2023;47:1596-1608. [doi: 10.1111/ijcs. 12906].
86. Wu J, Zheng H, Zhao B, Li Y, Yan B, Liang R, *et al*. Large-Scale Datasets for Going Deeper in Image Understanding. In: 2019 IEEE International Conference on Multimedia and Expo (ICME); 2019. p. 1480-5. doi: 10.1109/ICME.2019.00256