Brain Sort: An Automatic Analysis Toolkit for AD Neuroimaging Data

1 Abstract:

Alzheimer's disease (AD), as the most frequent cause of dementia characterized by a progressive decline in cognitive function (especially memory function), has received increasing attention in recent years. The exorbitant health-care fees for AD add on to the significance of early stage diagnosis. In the same time, many studies have shown the capability of multi-modal imaging in combination with machine learning methods to differentiate healthy subjects from AD, MCI, even SCD patients. Although the development of internet technology enables various big data studies in the field of neuroimaging, we are still in lack of easy-to-use machine learning toolkit for neuroimaging analysis. Most machine learning toolboxes could only be used by command lines, adding difficulties for doctors and other users who didn't have programming basis but have easy accessibility of dataset to conduct in-time analyses. Using Python language, we hereby developed a toolkit called Brain Sort with a friendly graphical user interface (GUI). With this toolkit, brain image data can be automatically analyzed and machine learning models can be trained for further classification study or disease diagnosis.

2 Introduction:

Characterized by insidious onset and progressive impairment of episodic memory, Alzheimer's disease (AD) has become a common neurodegenerative brain disease among elderly people over the past few years. According to a report published by Alzheimer's Disease International, there are around 44 million dementia patients worldwide, and this number will reach 76 million by 2030 and 135 million by 2050. Of all dementia patients, AD account for 50% to 75% of the case. Despite of the vacuum of a thorough cure medicine for AD, some medications have been used to delay the onset of certain symptoms (such as memory loss) and reduce its psychological impact on the patients. Therefore, accurate diagnosis of AD or MCI patients in the early stage is of great importance.

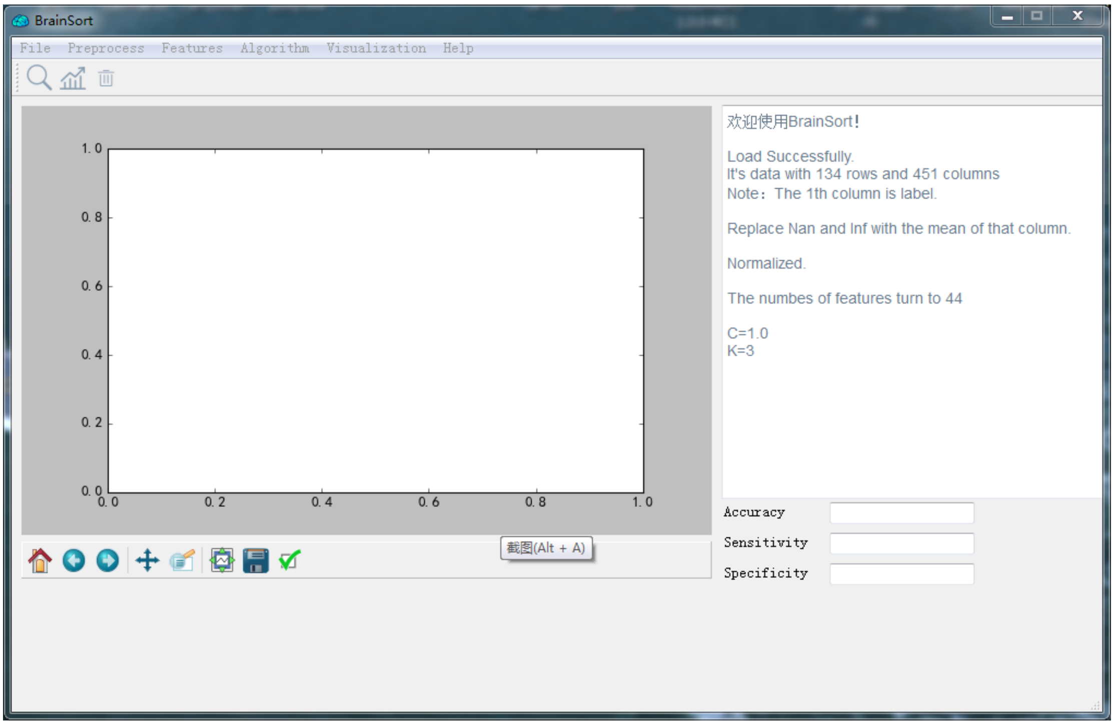
In the field of neuroimaging, with the advancement of internet technology, different structural and functional neuroimaging methods play increasingly important roles in early diagnosis of neurodegenerative diseases by controlling the progression of the disease and assisting the initiation of disease-modifying therapy. In recent years, magnetic resonance imaging (MRI) as a non-invasive practice has become a powerful diagnostic tool that provides clinicians and researchers with structural and functional information of human brains. Facilitated by these tools, there have been many studies modelling the brain as complex networks through neuroimaging data (e.g., MRI and EEG) of neural units (e.g., neurons and brain regions) linked by structural connectivity (i.e., structural wiring) and functional connectivity (i.e., coherent temporal activities). Many important findings emerged from these studies, such as the prominent small-world organization of human brain networks.

Moreover, there has been growing interest within the neuroimaging community in the use of analytical methods that enable such inference of the obtained information. One method is supervised machine learning (ML), an area of artificial intelligence concerned with the development of algorithms and techniques that can automatically extract information from the data. Compared to traditional methods of analysis based on the general linear model, the advantages of applying supervised ML to neuroimaging data are twofold: firstly, supervised ML methods allow characterization down to the level of individuals, therefore yielding results with a potentially high level of clinical translation; secondly, as inherently multivariate approaches, supervised ML methods are sensitive to spatially distributed and subtle effects in the brain that would be otherwise undetectable using traditional univariate method which focuses otherwise on gross differences at group level.

Support vector machine (SVM) is a specific type of supervised ML method that aims to classify data points by maximizing the margin between classes in a high-dimensional space. The optimum algorithm is developed through a ‘training’ phase in which training data are used to develop an algorithm able to discriminate between groups previously defined by the operator (e.g., patients vs. controls), and a ‘testing’ phase in which the algorithm is used to blind-predict the group to which a new observation belongs.

Recently, several freely available toolkits for extracting brain network topological properties have emerged, including Brain Connectivity Toolbox (BCT), eConnectome , Graph-Analysis Toolbox (GAT), Pipeline for Analyzing braiN Diffusion imAges (PANDA), Graph-theoRETical Network Analysis toolkit (GRETNA), NetworkX (http://networkx.lanl.gov/index.html), and Brainwaver (http://cran.r-project.org/web/packages/brainwaver/index.html). They have greatly assisted the investigation of the brain connectome. However, machine learning toolkits for analyzing the brain connectome is still lacking. Although there are some existing toolboxes, most of them can only be used by command lines, inconvenient for non-developers.

Here, we developed a toolkit named Brain Sort, with a friendly graphical user interface (GUI), to provide an efficient and flexible platform for analyzing brain connectome data by training model and visualizing the results, including the coefficient of different brain regions during model training. In this toolkit, the brain connectome topology parameters matrixes will be required as input.



欢迎使用BrainSort!

Load Successfully.
It's data with 134 rows and 451 columns
Note：The 1th column is label.

Replace Nan and Inf with the mean of that column.

Normalized.

The numbes of features turn to 44

C=1.0
K=3

Accuracy

Sensitivity

Specificity

FIGURE 1 | A snapshot of the GUI of Brain Sort.

3 Materials and methods:

3.1 Toolboxes

Brain Sort was developed by Python environment. Python is a widely-used high-level programming language for general-purpose programming with a large, comprehensive standard library. In this part, we will introduce some important toolboxes we have used in developing this toolkit. Figure 1 shows the GUI of Brain Sort.

3.1.1 Numpy

Numpy provides the “ndarray” data type to python, an efficient n-dimensional data representation for array-based numerical computation, similar to that used in Matlab. It handles efficient array persistence (input and output) and provides basic operations such as dot product. Most scientific Python libraries, including scikit-learn, use Numpy arrays as input and output data type.

3.1.2 Matplotlib

Matplotlib is a plotting library tightly integrated into the scientific python stack. It offers publication-quality figures in different formats and is used to generate the histogram in this paper.

3.1.3 Seaborn

Seaborn is a library for making statistical graphics in Python. It is built on top of matplotlib and tightly integrated with the PyData stack, including support for numpy and pandas data structures and statistical routines from scipy and statsmodels. Seaborn aims to make visualization a central part of exploring and understanding data.

3.1.4 Scikit-learn

Scikit-learn is a general purpose machine learning library written in Python. It provides efficient implementations of state-of-the-art algorithms, accessible to non-machine learning experts, and reusable across scientific disciplines and application fields. It also takes advantage of Python interactivity and modularity to supply fast and easy prototyping.

3.1.5 Pyqt4

It is more than a GUI toolkit. It includes abstractions of network sockets, threads, Unicode, regular expressions, SQL databases, SVG, OpenGL, XML, a fully functional web browser, a help system, a multimedia framework, as well as a rich collection of GUI widgets.

3.1.6 Mayavi

Mayavi is a general purpose, cross-platform tool for 3-D scientific data visualization. It has some excellent features: (1) easy scriptability using Python; (2) saving rendered visualization in a variety of image formats; and (3) convenient functionality for rapid scientific plotting via mlab.

3.2 Brain Sort Processing Procedures:

The main procedure of Brain Sort is shown in Figure 2.

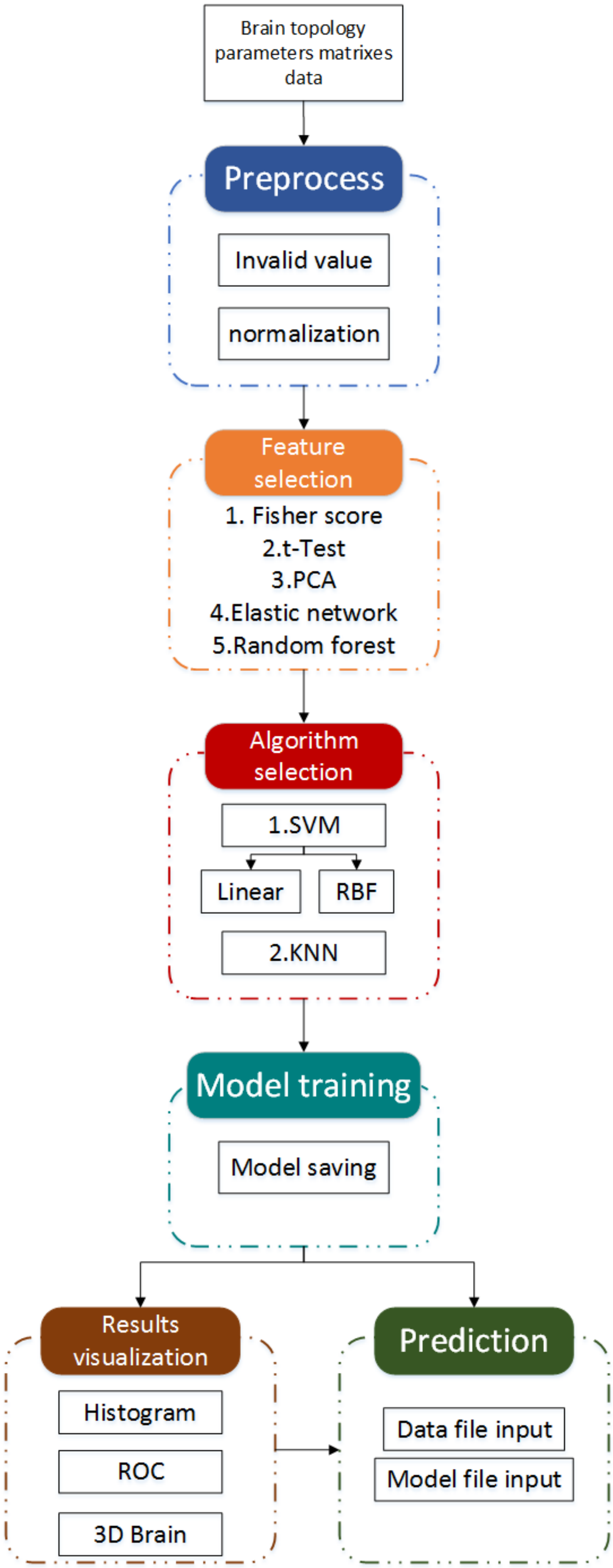

Brain topology parameters matrixes data
Preprocess
Invalid value
normalization
Feature selection
1. Fisher score
2.t-Test
3.PCA
4.Elastic network
5.Random forest
Algorithm selection
1.SVM
Linear
RBF
2.KNN
Model training
Model saving
Results visualization
Histogram
ROC
3D Brain
Prediction
Data file input
Model file input

FIGURE 2 | Main processing procedure in Brain Sort. The procedure includes six parts: (1) preprocess; (2) feature selection; (3) algorithm selection; (4) model training (5) result visualization; and (6) prediction.

3.2.1 Load the input file:

In this step, users need to input an M*N matrix data in the format of ASCII text file. M denotes the number of subjects including patients and healthy controls. N denotes the number of features for model training, which has 90*K columns. K denotes the input data consisting of K kinds of topology parameters about the brain connectome (e.g. clustering coefficients, characteristic path length, degree, local efficiency, and global efficiency). 90 is the number of brain regions defined by Automated Anatomical Labeling (AAL) atlas. In addition to the data, the subject label corresponding to the data is required. We provide two options for label information: users can input an M*1 vector in the format of ASCII text file; or they can input the number of the first class if they don't have the label txt file, but it can only fit the binary classification problem.

3.2.2 Preprocess option:

After data input, we need to perform a normalization step on each feature vector to obtain unit norm vector. In our toolkit, we use L2-norm regularization which is widely used.

3.2.3 Feature selection option:

Feature selection is an important step in bioinformatics analysis, because in many cases the number of features are much more than that of subjects. The number of bases spanning linear space is at most equivalent to the number of independent vectors. Accordingly, that there are more features than subjects inevitably lead to redundancy. Although dimensional reduction is often used to reduce redundancy, it is far from true redundancy elimination as reconstructed bases are usually the linear combination of all features, which is not always necessary for spanning the entire linear space.

In our toolkit, we provide four kinds of feature selection methods, including principal component analysis (PCA), Two Sample t-Test, Fisher-score, elastic network, and random forest.

3.2.3.1 PCA:

The central idea of principal component analysis (PCA) is to reduce the dimensionality of data set consisting of a large number of interrelated variables, while retaining as much as possible of the variance present in the data set. This end is achieved by transforming the original data to a new set of uncorrelated variables, the principal components (PCs), that are ordered such that the first few retain most of the variance present in all of the original variables.

3.2.3.2 Two Sample t-Test:

Welch’s t-test t-statistic is an adaptation of Student’s t-test. That is, it has been derived with the help of Student's t-test and is more reliable when the two samples have unequal variances and unequal sample sizes. This test is used only under the assumption that the two distributions have the same variance. The test decision is from the null hypothesis that the data in vectors x and y come from independent random samples of normal distributions with equal means and equal but unknown variances. In this option, users need to input a predefined threshold and features with p-values smaller than the threshold will be selected.

3.2.3.3 Fisher score:

Fisher score is a univariate filter method that is commonly employed to determine the discriminatory power of individual features between two classes of equal probability. Moreover, it is independent of the class distribution. Therefore, if the data didn’t content normal distributions, users can choose this option instead.

3.2.3.4 Elastic Network:

A promising technique called the lasso was proposed by Tibshirani (1996). The lasso is a penalized least squares method imposing an L1-penalty on the regression coefficients. Due to the nature of L1-penalty, the lasso performs both continuous shrinkage and automatic variable selection simultaneously. Moreover, Hui Zou’s team proposed a new regularization technique called the elastic network, similar to lasso but with fewer limitations and thus more widely used. In our toolkit, we apply this one as a method of feature selection. It should be noted that users need to input four parameters. Alpha: constant that multiplies the penalty terms and defaults to 1.0. L1 ratio: the elastic net mixing parameter. Max iter: the maximum number of iterations. Tolerance: the tolerance for the optimization.

3.2.3.5 Random Forest:

Random forests (RF) is a popular technique for classification, prediction, variable importance evaluation, variable selection, and outlier detection. There are numerous applications of RF in a variety of fields. In our study, we apply it as a method for feature selection. In this option, users need to input three parameters. N trees: the number of trees in the forest. Min split: The minimum number of samples required to split an internal node. K save: the number of feature that they want to retain.

3.2.4 Algorithm option:

In this step, users need to choose the model training kernel algorithm. We provide two options: SVM (support vector machines) and k-NN (k-nearest neighbors).

3.2.4.1 SVM:

Within ML, there are two main approaches that one can take: supervised and unsupervised learning. In supervised learning, one seeks to develop a function which maps two or more sets of observations with two or more, operator defined, categories through an iterative procedure which gradually reduces the difference between the predicted and expected result; subsequently, the algorithm can then be used to assign new, previously unseen, data to one of the predefined categories with a given accuracy. By contrast, in unsupervised learning, one seeks to determine how the data are organized without the availability of a priori information supplied by the operator; with the primary objective of discovering unknown, but potentially useful structure in the data.

One specific form of supervised pattern recognition algorithm is that used for classification, concerned with the automatic discovery of regularities in the data that can be used to classify the data into different predefined categories. Using this approach, individuals (represented by their brain scan for example) are referred to as 'examples' and the categories to which they might belong, 'labels'. The aim is to generate the decision function or 'classifier' that most accurately captures the relationship between each example and its respective label. Examples of multivariate techniques for pattern recognition include, but are not limited to, artificial neural networks, decision trees, Gaussian process classification, and SVM. Of these the most popular technique in neuroimaging literatures is SVM, which to date has been applied in various studies of neurological and psychiatric disorder, allowing the classification of individual observations (e.g. scans) into distinct groups or classes (e.g. diagnostic categories) based on data in high-dimensional space. In addition to performing linear classifications, SVM can also efficiently perform non-linear classification using kernel functions, implicitly mapping their inputs into high-dimensional feature spaces.

In our toolkit, we provide two kernel functions, including linear kernel and radial basis function (RBF) kernel. Notably, we recommend linear kernel in consideration of visualization of the result and we will talk about the details later in the result section. After selecting the kernel functions, users need to input two parameters: C (cost): The parameter C of C-SVC. K: the parameter of k-fold cross validation mode.

3.2.4.2 *k*-NN (*k*-nearest neighbors):

In pattern recognition, the k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regression. It is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms. In this option, users need to input 4 parameters: K: The number of neighbors to use for k neighbor queries. Distance: The distance metric to use for the tree. The default metric is minkowski, and with p=2 is equivalent to the standard Euclidean metric. See the documentation of the Distance Metric class for a list of available metrics. W (weight): The weight function used in prediction. Possible values: 'uniform': uniform weights. All points in each neighborhood are weighted equally. 'distance': weight points by the inverse of their distance. In this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.

3.2.5 Results visualization and model saving:

After finishing the operation, if linear kernel functions are chosen, the toolkit will measure the contribution of each brain region and display it on the right side of the toolkit interface. You can also build a histogram with the first n regions and display a 3D brain model marked with the location of ROI. And in fact, in our toolkit only linear kernel can output the contribution coefficient.

In addition to result visualization, we can also save the classifier model if you want to apply this model for further classification study.

3.2.6 Application of the model:

If users want to apply the model for further study, they need to input the model file and the data they want to predict which didn't consist of label information. After that, Brain Sort will predict the subject label and generate a result file.

3.2.7 Testing the AD MRI data by using Brain Sort:

3.2.7.1 Subjects.

To demonstrate the effects of this toolbox on real data, we analyzed the dataset from the memory clinic of the Neurology Department of Xuan Wu Hospital in Beijing. There are 132 subjects enrolled including 72 NC subjects and 60 AD subjects.

3.2.7.2 Image acquisition.

All MR scans were performed on a 3.0 Tesla MR system (Siemens Magnetom Trio Tim MRI system, Germany) using a standard head coil. During the entire scanning procedure, cushions and headphones were used to reduce subject motion and scanner noise. Structural DTI data and T1-weighted data was available in all participants.

Diffusion Tensor Imaging. DTI data was collected using an echo planar imaging (EPI) sequence with following parameters for three times: in 31 independent, non-collinear directions of a b-value = 1000 s/mm^2 and one additional image with no diffusion weighting (b = 0), slices = 60, TR= 11000 ms, TE =98 ms, flip angle =90°, FOV = 256 mm×232 mm, acquisition matrix= 128×116, and thickness= 2 mm with reversed k-space read-out. The resulting T1-weighted images and cortical models were linearly aligned to the space of the diffusion weighted imagings (DWIs). DWIs as well as resulting tracts were further elastically registered to the T1-weighted image to account for susceptibility artifacts (we assume that the T1-weighted scan serves as a relatively undistorted anatomical reference).

Anatomical T1. In addition, a T1-weighted image was acquired for anatomical reference. T1-weighted MR images were obtained by a 3D magnetization-prepared rapid gradient echo (MPRAGE) with following parameters: Slices = 176, TR = 1900 ms, TE = 2 ms, inversion time (TI) = 900 ms, flip angle=9°, field of view (FOV) = 224×256 (mm)^2, acquisition matrix = 448×512, no gap, and thickness = 1.0 mm.

3.2.7.3 Image pre-processing and network construction.

The image pre-processing and network construction was conducted by PANDA, a MATLAB toolbox for fully automated processing of brain diffusion images. It can perform a series of steps to process DICOM/NIfTI to diffusion metrics. It can also finish the construction of anatomical brain networks for all subjects.

After we get the connectome matrixes, we apply a toolbox named GRETNA for network analysis. It can compute various network topology parameters and we will use five kinds of them, including clustering coefficient of each node (CP), shortest path length of each node (LP), local efficiency of each node (Eloc), global efficiency of each node (Eg), and the degree number of each node (Deg). And because of the AAL atlas we mentioned before, each subject has 90 brain regions regarded as nodes, so for each kind of topology parameters there are 90 values, which means each subject has 450 features in total.

After all the steps above we can input this matrix into our toolkit to process. In this part, we choose Fisher-score in feature selection option and SVM with linear kernel in kernel algorithm option.

4 Result

4.1 A Machine Learning Toolkit for Brain Connectome Data: Brain Sort

We developed a toolkit named Brain Sort. It's an efficient toolkit for analyzing brain network topology parameters data with machine learning methods, which is an open-source toolkit and is freely available at www.bit.edu.cn. Technical supports and updates will be constantly provided by the developers.

This toolkit can analyze the brain topology parameters data based on AAL atlas and can visualize the results according to the contribution during model training. Moreover, users can locate the most important regions in the three-dimension brain model, thus helping them to understand the mechanism of human brain.

Users can also apply the model they trained before for subject label prediction which can help doctors and other researchers to diagnose some diseases or to evaluate the risk of disease.

4.2 The performance of classifier.

Comparison of different methods was performed via k-fold cross-validation due to the limited numbers of available samples. The performance was summarized in Table 1. We use Fisher-score to evaluate the significance in the feature selection step and 3-fold cross-validation. As we can see, a classification accuracy of 82.6% was achieved using linear kernel, with sensitivity of 78.3% and specificity of 86.1%, which can give evidence to the most discriminant region. It's worth noting that only linear kernel can get the coefficients of various brain regions. We can also show the ROC curve as in

Figure 3 and calculate the area under it (AUC), as already shown in Table 1, which measures the probability when one positive and one negative samples are drawn at random, the decision function assigns a higher value to the positive than to the negative sample. Figure 4 is the top 10 of the most discriminant regions during model training. The X-axis means the serial number of brain regions in the automated anatomical labeling (AAL) atlas and the Y-axis means the coefficient that evaluates the weights of different regions during model-training process. These regions' names were shown in Table 2. And significant regions such as cingulum and frontal orbital are highly consistent with previous findings. We can also mark these regions on the 3D brain AAL atlas model as Figure 5 has shown, and the contribution coefficient of each region during model training decides the size of balls marking the location.

The above-shown results for this specific study prove the usability and validity of Brain Sort.

Table 1 Classification performance of different methods

Method	ACC(%)	SEN(%)	SPE(%)	AUC
linear	82.6	78.3	86.1	0.86
RBF	85.6	73.3	95.8	0.91
KNN	80.3	75.0	84.7	0.80

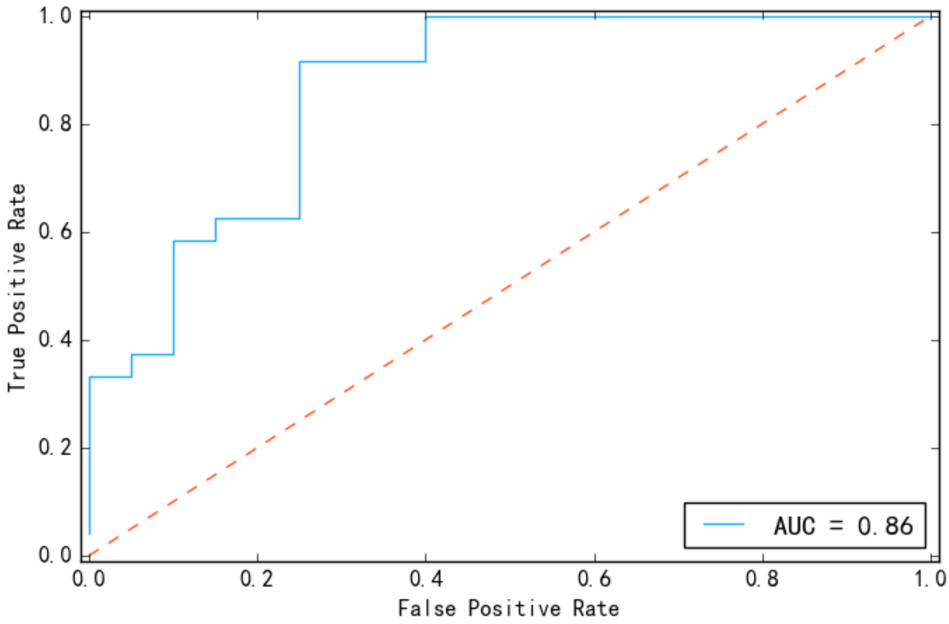


FIGURE 3 | Receiver operating characteristic (ROC) curve.

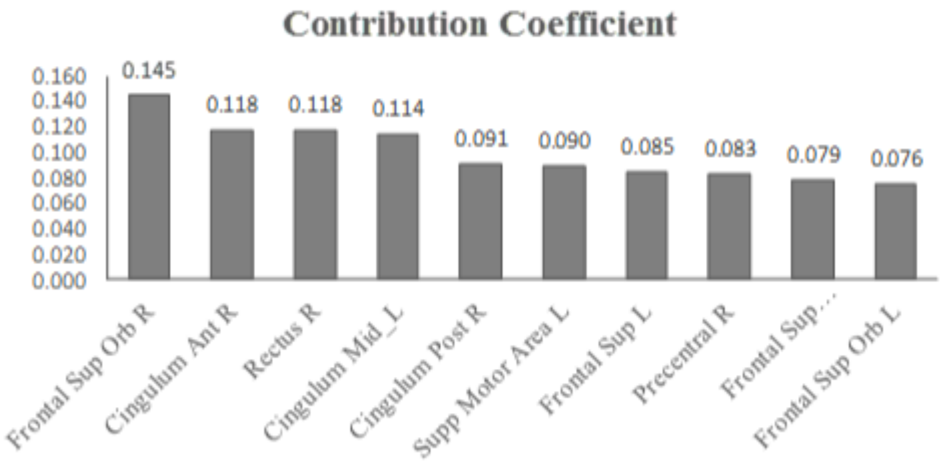


FIGURE 4 | The most discriminant regions during model training.

Table 2 Region name

serial number	name of region
6	Frontal Sup Orb R
32	Cingulum Ant R
28	Rectus R
33	Cingulum Mid L
36	Cingulum Post R
19	Supp Motor Area L
3	Frontal Sup L
2	Precentral R
24	Frontal Sup Medial R
5	Frontal Sup Orb L

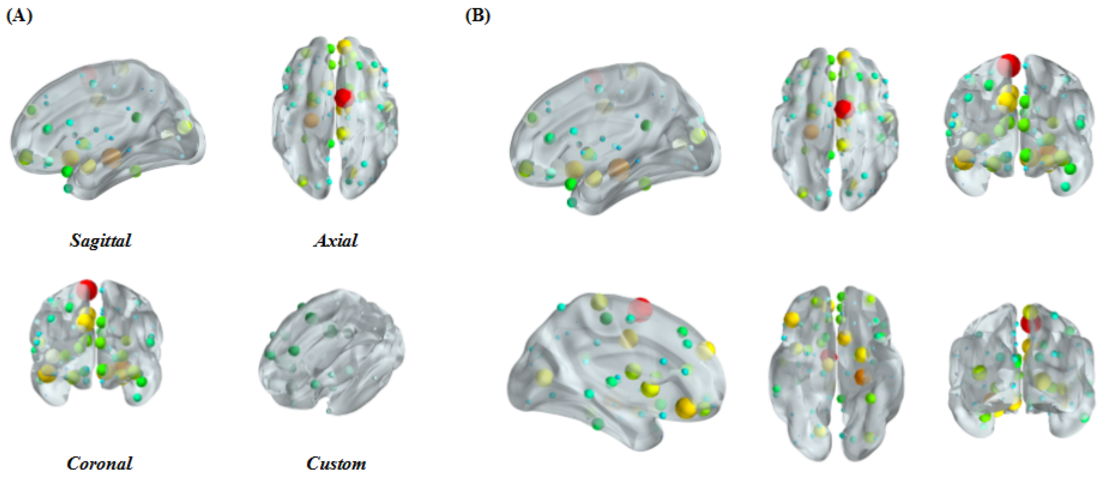


FIGURE 5 | 3D brain model

5 Discussion

We developed Brain Sort as a free toolkit for analyzing brain image data with machine learning methods, which achieved the following major functions: (1) analyze the brain image data with machine learning methods and select the most important regions during model training; (2) visualize the results and build various figures including the histogram of the coefficients that evaluate the contribution of different regions during model-training process, the receiver operating characteristic (ROC) curve and the 3D brain model marked with the ROI; and (3) apply the model users have trained for further classification study or disease diagnosis.

In the past few years, an increasing cohort of studies have used SVM or other pattern recognition methods to investigate possible neuroanatomical biomarkers of neurological and psychiatric disorders. Alzheimer's disease (AD) is probably the mostly studied application of supervised learning in neuroimaging. Advances in medical imaging and medical image analysis have provided useful means to generate and extract valuable neuroimaging information. Automatic classification techniques provide tools to analyze this information and to observe inherent disease-related patterns in the data. In particular, these classifiers have been used to discriminate AD patients from healthy control subjects and to predict conversion from mild cognitive impairment to AD. All these studies have indicated the advantages of machine learning methods in AD study.

With the advent of brain connectome studies, several toolboxes were developed to construct and analyze macro-scale brain networks, including PANDA, BCT, GAT, GRETNA, BrainNet Viewer, Brainwaver, and eConnectome. These toolboxes provide measurements of brain connectome features or the visualization for biological findings but lack the application and integration of these results. Libsvm (http://www.csie.ntu.edu.tw/cjlin/libsvm) and Scikit-learn are popular machine learning library but they don’t provide GUI so that it is not easy for users who don’t have programming basis to use them in their studies.

Although Brain Sort tries to build a new perspective for brain connectome, a few methodological considerations and directions require future study. First, this toolkit can only input data file in “.txt” format. But in this field, many other format data files are widely used, such as “.mat”, “.exl” or “.csv”. Secondly, we only use AAL atlas to define brain regions. Adding more atlas and templates would increase the universality of our toolkits for different kinds of studies.

6 Acknowledgments

We sincerely thank all the developers of Numpy, Matplotlib, Seaborn, Scikit-learn, Pyqt4, and Mayavi. In addition, we thank the support of the memory clinic of the Neurology Department of Xuan Wu Hospital in Beijing.

7 Reference

Abraham, A., F. Pedregosa, M. Eickenberg, P. Gervais, A. Mueller, J. Kossaifi, A. Gramfort, B. Thirion, and G. Varoquaux. "Machine Learning for Neuroimaging with Scikit-Learn." Front Neuroinform 8 (2014): 14.

Alves, G. S., V. O. Knochel, C. Knochel, A. F. Carvalho, J. Pantel, E. Engelhardt, and J. Laks. "Integrating Retrogenesis Theory to Alzheimer's Disease Pathology: Insight from Dti-Tbss Investigation of the White Matter Microstructural Integrity." [In English]. Biomed Research International (2015).

Arbabshirani, M. R., S. Plis, J. Sui, and V. D. Calhoun. "Single Subject Prediction of Brain Disorders in Neuroimaging: Promises and Pitfalls." [In English]. Neuroimage 145 (Jan 15 2017): 137-65.

Bassett, D. S., and E. T. Bullmore. "Human Brain Networks in Health and Disease." [In English]. Current Opinion in Neurology 22, no. 4 (Aug 2009): 340-47.

———. "Small-World Brain Networks." [In English]. Neuroscientist 12, no. 6 (Dec 2006): 512-23.

———. "Small-World Brain Networks Revisited." Neuroscientist (Sep 21 2016).

Cui, Z. X., S. Y. Zhong, P. F. Xu, Y. He, and G. L. Gong. "Panda: A Pipeline Toolbox for Analyzing Brain Diffusion Images." [In English]. Frontiers in Human Neuroscience 7 (Feb 21 2013).

Cui, Z., S. Zhong, P. Xu, Y. He, and G. Gong. "Panda: A Pipeline Toolbox for Analyzing Brain Diffusion Images." Front Hum Neurosci 7 (2013): 42.

He, B., Y. K. Dai, L. Astolfi, F. Babiloni, H. Yuan, and L. Yang. "Econnectome: A Matlab Toolbox for Mapping and Imaging of Brain Functional Connectivity." [In English]. Journal of Neuroscience Methods 195, no. 2 (Feb 15 2011): 261-69.

Hosseini, S. M., F. Hoeft, and S. R. Kesler. "Gat: A Graph-Theoretical Analysis Toolbox for Analyzing between-Group Differences in Large-Scale Structural and Functional Brain Networks." PLoS One 7, no. 7 (2012): e40709.

Hunter, J. D. "Matplotlib: A 2d Graphics Environment." [In English]. Computing in Science & Engineering 9, no. 3 (May-Jun 2007): 90-95.

Khazaee, A., A. Ebrahimzadeh, and A. Babajani-Feremi. "Application of Advanced Machine Learning Methods on Resting-State Fmri Network for Identification of Mild Cognitive Impairment and Alzheimer's Disease." [In English]. Brain Imaging and Behavior 10, no. 3 (Sep 2016): 799-817.

Kloppel, S., A. Abdulkadir, C. R. Jack, N. Koutsouleris, J. Mourao-Miranda, and P. Vemuri. "Diagnostic Neuroimaging across Diseases." [In English]. Neuroimage 61, no. 2 (Jun 2012): 457-63.

Liao, X., A. V. Vasilakos, and Y. He. "Small-World Human Brain Networks: Perspectives and Challenges." Neurosci Biobehav Rev 77 (Apr 05 2017): 286-300.

Lu, Z. Q. J. "The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition." [In English]. Journal of the Royal Statistical Society Series a-Statistics in Society 173 (2010): 693-94.

Nir, T. M., N. Jahanshad, J. E. Villalon-Reina, A. W. Toga, C. R. Jack, M. W. Weiner, P. M. Thompson, and Alzheimer's Dis Neuroimaging Initi. "Effectiveness of Regional Dti Measures in Distinguishing Alzheimer's Disease, Mci, and Normal Aging." [In English]. Neuroimage-Clinical 3 (2013): 180-95.

Orru, G., W. Pettersson-Yeo, A. F. Marquand, G. Sartori, and A. Mechelli. "Using Support Vector Machine to Identify Imaging Biomarkers of Neurological and Psychiatric Disease: A Critical Review." [In English]. Neuroscience and Biobehavioral Reviews 36, no. 4 (Apr 2012): 1140-52.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, et al. "Scikit-Learn: Machine Learning in Python." [In English]. Journal of Machine Learning Research 12 (Oct 2011): 2825-30.

Pereira, F., T. Mitchell, and M. Botvinick. "Machine Learning Classifiers and Fmri: A Tutorial Overview." [In English]. Neuroimage 45, no. 1 (Mar 2009): S199-S209.

Rubinov, M., and O. Sporns. "Complex Network Measures of Brain Connectivity: Uses and Interpretations." [In English]. Neuroimage 52, no. 3 (Sep 2010): 1059-69.

Ruxton, G. D. "The Unequal Variance T-Test Is an Underused Alternative to Student's T-Test and the Mann-Whitney U Test." [In English]. Behavioral Ecology 17, no. 4 (Jul-Aug 2006): 688-90.

Stoessl, A. J. "Neuroimaging in the Early Diagnosis of Neurodegenerative Disease." Transl Neurodegener 1, no. 1 (Jan 13 2012): 5.

Taguchi, Y. H., M. Iwadate, and H. Umeyama. "Principal Component Analysis-Based Unsupervised Feature Extraction Applied to in Silico Drug Discovery for Posttraumatic Stress Disorder-Mediated Heart Disease." BMC Bioinformatics 16 (Apr 30 2015): 139.

van der Walt, S., S. C. Colbert, and G. Varoquaux. "The Numpy Array: A Structure for Efficient Numerical Computation." [In English]. Computing in Science & Engineering 13, no. 2 (Mar-Apr 2011): 22-30.

Wang, J. H., X. D. Wang, M. R. Xia, X. H. Liao, A. Evans, and Y. He. "Gretna: A Graph Theoretical Network Analysis Toolbox for Imaging Connectomics." [In English]. Frontiers in Human Neuroscience 9 (Jun 30 2015).

Yilmaz, E. "An Expert System Based on Fisher Score and Ls-Svm for Cardiac Arrhythmia Diagnosis." [In English]. Computational and Mathematical Methods in Medicine (2013).

Zou, H., and T. Hastie. "Regularization and Variable Selection Via the Elastic Net." [In English]. Journal of the Royal Statistical Society Series B-Statistical Methodology 67 (2005): 301-20.