Comparing Machine Learning Classification Methods for Biological Tracking Data

David Shackelford

A thesis

submitted in partial fulfillment of the

requirements for the degree of

Master of Science

University of Washington

2020

Committee:

Elizabeth Nance

David Beck

Program Authorized to Offer Degree:

Chemical Engineering

University of Washington

Abstract

Comparing Machine Learning Classification Methods for Biological Tracking Data

David Shackelford

Chair of the Supervisory Committee:

Elizabeth Nance

Department of Chemical Engineering

Multiple particle tracking (MPT) has been increasingly used to characterize and probe biological environments. The additional use of machine learning (ML) methods has previously been proven successful in classifying both single particle tracking and MPT data for many biological systems. In order to further utilize collected MPT data for studies focused on nanoparticle diffusion within biological environments, a new predictive package, referred to as diff_predictor was developed in this study. This package uses feature and trajectory datasets, along with prediction methods such as XGBoost, recurrent neural networks, and random forest decision trees, to make predictions and classifications about the biological environment and nanoparticle behavior. To apply diff_predictor, I took a threefold approach: First, an extreme gradient boosting decision tree (XGBoost) was applied to a study of diffusion in a developing region within the brain to predict age based on spatial features. The features used in classification were analyzed for importance using Shapley Additive Explanations (SHAP) values. Second, a long short-term memory recurrent neural network algorithm was used on the temporal data from the same dataset and the results were compared with XGBoost and random forest models.

Finally, demonstration of the versatility of the diff_predictor package showed we can classify regional differences in diffusion in the brain using a new different experimental dataset. The models produced by diff_predictor had accuracies up to 54.56% for prediction in age-based data, and accuracies up to 91.35% for prediction in regional-based data. Furthermore, SHAP values produced by the decision tree methods in diff_predictor provided useful information toward nanoparticle environment and interactions.

Acknowledgements

I would like to thank Dr. Nance for giving me the opportunity to work in her lab for the past two years. When deciding which University to attend, I had a goal to find a PI who would care about me and my career. One of the major reasons I decided to attend UW was because I saw the amazing work she had already accomplished in her career and how well she treated her students. I have never met a professor who was so caring about her graduate and undergraduate students and who put in so much work to help them get further in life. I am extremely grateful to be a part of her group and I appreciate everything she has done for me.

I thank Dr. Beck for teaching me about data science. I am incredibly grateful for his push for data science in the Chemical Engineering Department and his dsmcer/seds classes that he taught. I have learned so much and his inspiration has really helped me find a new passion and career path going forward.

I thank my family for being there for me through the ups and downs, and for helping me get through graduate school. I would like to thank my mom for always being there for me when I was stressed from classes and for pushing me to always do better. I want to thank my dad for inspiring me to go further in academia. And I want to thank my best friend and brother who was always there to cheer me on.

I thank all the awesome graduates and undergraduate students in the Nance Lab. Every day I came to the lab, I was always greeted with friendly faces. I loved having conversations with all of you and I will miss the fun lab retreats and happy hours we had together.

I thank my former mentor, Dr. Ken Carlson of Colorado State University, who was a great inspiration to me. I am extremely grateful for everything he has done for me, and I will always remember the great times I have had in his lab.

Thank you all for the love and support over the past two years of my MS graduate studies.

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

Current brain-affiliated diseases and injuries such as stroke, Alzheimer's disease, and traumatic brain injury have high mortality rates and high socioeconomic burden.[1-4] Traumatic brain injury alone can cost an individual as much as $133,000 for severe injuries and $54,000 for four-year follow up charges. These medical expenses account for a projected $8 billion over the course of the first four years for treatment within the United States.[5] Potential therapies for central nervous system (CNS) diseases take greater than 10 years to reach clinical use and have a higher failure rate than non-CNS therapeutics.[6, 7] Even after a long testing period, drugs for major neurodegenerative diseases have a clinical failure rate for disease modifying treatments of 100%, and current approved treatments focus on palliative symptomatic relief rather than curative to stop disease progression.[7] Many therapies fail partly because of the difficulties faced when penetrating and transporting through the blood brain barrier (BBB) and the brain parenchyma.[7-9] These problems highlight the importance of finding cheaper and better ways to diagnose and treat brain injuries and to find better therapeutic pathways for drug delivery.

Since the 1960s, clinical studies have utilized nanoparticles to develop and deliver therapeutics and increase diagnostic performance.[10] The application of single particle tracking (SPT) and multiple particle tracking (MPT) has increased in recent years as recognized methods to observe nanoparticle behavior within biological microenvironments and help inform the development of therapeutics. MPT is a method that uses microscopy to capture the diffusion of thousands of fluoresced nanoparticles simultaneously while still maintaining similar resolutions to SPT trajectories. MPT can be used in many biological applications including measurement of rheological aspects in microenvironments,[11] pore size within the brain extracellular space (ECS),[8] and uptake and penetration through biological barriers.[12] MPT has also increasingly been employed as a viable way to quantify nanoparticle transport in improving

overall efficacy for nanotherapeutic platforms and drug delivery within biological systems and the brain.[13, 14]

Data collected from MPT methods can quickly become extremely large, with some experiments containing more than a terabyte of information. Given the significant amount of data collected, machine learning can be effectively utilized to show any hidden trends in datasets and to classify datapoints for prediction. Machine learning methods have successful classified both SPT and MPT datasets for many biological systems.[15-17] However, when using MPT, a lot of the data collected is not used, resulting in a high overhead for little return. In order to derive more information out of datasets, Wagner *et al.*, Helmuth *et al.*, and Curtis *et al.* calculated geometric features of each trajectory, such as boundedness, straightness, and mean square displacement (MSD) ratio, based on MPT data. The results of their approaches led to better classification of particles and more insights into the nanoparticle-environment interaction space.[17-20]

In order to further utilize collected MPT data for studies focused on nanoparticle diffusion within the brain, I developed a new predictive package, referred to as diff_predictor. This package uses feature and trajectory datasets to make predictions and classifications about the biological environment and nanoparticle behavior. To apply diff_predictor, I took a threefold approach: First, I applied an extreme gradient boosting decision tree (XGBoost) to diffusion data obtained from the developing cortex of different age brains to predict age based on spatial features. The features used in classification are analyzed for importance using Shapley Additive Explanations (SHAP) values. Second, I used a long short-term memory (LSTM) recurrent neural network (RNN) algorithm on temporal data from the same dataset and compared the results with XGBoost. Finally, I demonstrated the versatility of the diff_predictor package by classifying regional differences using diffusion data obtained from the hippocampus, thalamus, ganglia, striatum, and cortex.

2 EXPERIMENTAL SETUP AND METHODS

Due to time constraints and limited lab access resulting from the COVID-19 pandemic, brain slicing, nanoparticle preparation, and multiple particle tracking (MPT) techniques were performed by Mike McKenna in the Nance Lab. I used the resulting datasets from MPT to calculate trajectory features, develop and analyze machine learning models, and apply predictive models. I developed diff_predictor, a machine learning package to accomplish these objectives. All methods in every stage of the process are detailed as follows.

2.1 Brain Slices

Brain slices were prepared from male Sprague-Dawley (SD) rat pups at varying ages, depending on the specific study. All animal experiments were carried out at the University of Washington in accordance with National Institute of Health guidelines and local Institutional Animal Care and Use Committee regulations. Following euthanasia, brains were extracted, immersed in room temperature (22°C) dissection media, and cut into hemispheres with a razor blade. 300 μm slices were then prepared from each hemisphere using a McIlwain tissue chopper. Individual slices were plated on 30-mm cell culture inserts in non-treated 6-well plates. Prior to plating, 6-well plates were filled with 1 mL SCM. Slices were incubated in sterile conditions at 37°C and 5% CO_2.

2.2 Multiple Particle Tracking

All MPT studies were performed within 24 h of slice preparation. Slices were imaged in a temperature-controlled incubation chamber maintained at 37°C, 5% CO_2, and 80% humidity. 30 minutes (min) prior to video acquisition, 40nm polystyrene nanoparticles conjugated with poly(ethylene-glycol) (PS-PEG) were diluted in 1x phosphate-buffered saline (PBS) to a concentration of ~0.014%. Nanoparticles were injected into each slice using a 10 μL glass syringe (model 701, cemented needle, 26-gauge, Hamilton Company, Reno, NV). Videos were

collected at 33 frames-per-second and 100x magnification for 651 frames via fluorescent microscopy using a cMOS camera (Hamamatsu Photonics, Bridgewater, NJ) mounted on a confocal microscope. Nanoparticle trajectories and trajectory mean square displacements (MSDs) were calculated via diff_classifier (https://github.com/Nance-Lab/diff_classifier), a Python package developed within Nance Lab.[20]

2.3 Feature Extraction

To analyze spatial data, diff_classifier was utilized to calculate 23 different geometric features for every trajectory in the given dataset. In addition, these geometric features were expanded with additional statistical features by calculating the local average of every trajectory feature within 9μm x 9μm rectangular regions for every video taken. The additional features added to the calculated diff_classifier include progression, total distance traveled, turning angle variance, and convex hull of the trajectory.[21] In total, 39 features are calculated for every trajectory. These features are detailed in the Appendix (Table A-1).

2.4 XGBoost and Random Forest

Two ensemble decision tree predictive models are implemented in diff_predictor: XGBoost and Random Forest. XGBoost (https://github.com/dmlc/xgboost) is a boosted decision tree algorithm which builds itself sequentially using multiple weak learner prediction models. Every tree produced in XGBoost during training is fitted to randomly selected datapoints taken from a modified weighted version of the original dataset. Sequential model building continues until either a set number of learners has been created or until the model converges within the exponential loss function. A prediction result is then made by calculating the weighted average of all created learners.[22] XGBoost differs from most boosted decision trees in that XGBoost incorporates regularization into its algorithm to control overfitting data during training. XGBoost also uses a unique objective function which encourages simple models, decreasing variance.[23] For each XGBoost model created, the observed dataset is first cleaned, balanced so that each

class has the same number of datapoints, and tuned for hyperparameters using a cross-validation step to minimize log-loss value. A custom cross-validation method is applied which bins data into as even as possible "folded" datasets in a specific manner to ensure there is no data leakage from the statistical feature sets between the testing and training folds. Each fold is then selected as a testing set while the other folds are used as a training set. The log-loss results of each fold are weighted based on their dataset size and a mean and standard deviation are calculated. Hyperparameters that are tuned for an XGBoost model are in Table 2-1:

Table 2-1. Selected hyperparameters for XGBoost predictive model tuning with their descriptions.

Hyperparameter	**Description**
max_depth	Maximum depth allowed for a created tree. A larger depth will result in a more complex tree.
eta	Step size shrinkage used at the end of boosting process to prevent overfitting. Used in process to shrink feature weights to make boosting more conservative.
min_child_weight	Minimum sum of the instance weight needed in a child node for every node within the tree.
gamma	Minimum loss reduction required to make a further partition on a leaf node of a tree being boosted.
subsample	Sample fraction of the training data that is randomly taken prior to growing a tree in each boosting round.
colsample_bytree	Subsample ratio of columns when constructing a tree each boosting round.

Random Forest (RF) is a bagging decision tree method which makes predictions based on the parallel creation of many weak learning models. These weak learners are created using a random sampling of the calculated features and are traditionally fed with a bootstrapped sample of the given dataset. A classification is then made by aggregating the predictions of the produced trees. Usually for random forest, an estimate of the error can be taken using the out-of-bag (OOB) data, which are the data not in the bootstrap sample during training.[24] For diff_predictor, bootstrapped sampling and OOB are not used due to the additional non-independent statistical features within the dataset. Instead, a customized cross-validation

method is implemented for hyperparameter tuning in the same manner as XGBoost to prevent data leakage between testing and training folds. Hyperparameters for RF are in Table 2-2.[25]

Table 2-2. Selected hyperparameters for random forest predictive model tuning with their descriptions.

Hyperparameter	**Description**
n_estimators	The number of trees in a RF model. The more trees, the more complex the model is.
min_samples_split	Splitting rule. The minimum number of samples that are required within an internal tree node to split it.
min_samples_leaf	The minimum number of samples that are required to be at a leaf node.
max_depth	The maximum allowed depth of a tree. Larger depth is a more complex an individual tree is.

For both methods, after training is finished, a testing set is predicted using the trained model. Precision, recall, f1-score are taken with a single model. In addition, the amount of true-positive, true-negative, false-positive, and false-negative values are calculated. The model is then used to for 50 predictions using a random sampling of the testing set. From this, an average and standard deviation accuracy is calculated.

2.5 Feature Analysis Using SHAP

Shapley Additive Explanations (SHAP) is a method used to explain individual predictions using Shapley Values (https://github.com/slundberg/shap). A Shapley value, ϕ, of a feature, j, on the prediction, $\hat{f}(x)$, is the contribution of a feature value to the difference between the actual prediction and the mean prediction:

$$\Phi_j(\hat{f}) = \beta_j x_j - E(\beta_j X_j) \qquad \textbf{(1)}$$

where $\beta_j x_j$ is the actual prediction and $E(\beta_j X_j)$ is the mean prediction.[26]

Shapley values are difficult to interpret, so SHAP attempts to better understand the individual contribution by explaining predictions of an instance. This explanation is done by

computing the contribution of each feature to the prediction using coalitional game theory. The primary tools used in SHAP are summary plots showing each feature importance, and dependency plots which plot feature values with respect to SHAP values. SHAP importance is different than traditional feature importance in that SHAP importance is based on the magnitude of feature attribution and is calculated by averaging the absolute Shapley values for every feature in the data as follows:

$$I_j = \sum_{i=1}^{n} \left|\Phi_j^i\right| \qquad \textbf{(2)}$$

The advantages of using SHAP over other methods are that prediction is fairly distributed among feature values (even when features are correlated), SHAP uses contrastive explanations which compare the actual prediction with the average prediction, and SHAP has shown to have a faster implementation for tree-based predictive models than other methods that are computationally slower.[27]

Additionally, SHAP values are in the log-odds $f(x)$ space given by the equation:

$$f(x) = \log \frac{x}{1-x} \qquad \textbf{(3)}$$

where x is the true probability of a value belonging to a certain class. This means that the more positive a SHAP value is, the more likely that value belongs to the given class while the more negative a SHAP value is, the more unlikely that value belongs to the given class. A SHAP value of 0 reflects uncertainty in prediction.[28]

2.6 LSTM RNN

Long short-term memory (LSTM) recurrent neural networks (RNNs) were developed using the Keras API for TensorFlow. RNNs are a type of neural network (NN), which can use previous output data recurrently in order to learn patterns. This is done by applying a loop to each neuron which feeds output data back into the input. This creates a chain-like model containing multiple copies of the same neural network. This is useful for series-like data and has been used in

many different applications such as speech recognition,[29] spelling correction,[30] and trajectory modeling.[31] LSTMs are a type of RNN that implement gates which regulate the LSTM and decide what information gets through each LSTM node. Use of these gates, along with pointwise operations allow the LSTM to learn long-term dependencies.[32, 33]

Like decision tree models, LSTMs have hyperparameters which need to be tuned using training data. One of the main hyperparameters is the number of epochs to use. To train an LSTM, training data is broken up into batches. Each batch size is a random sampling of the data. This data is then fed through an LSTM in order to update NN weights to fit the entire dataset. A single pass of data through the NN is known as an epoch. In order to fully train for the entire dataset, multiple epochs must be applied to the NN in order to minimize the NN's cost function using gradient descent. This cost function is measured as the loss value. For experiments the loss value will be measured as cross-entropy loss, or log-loss.[34] Hyperparameters that are tuned for LSTM are in Table 2-3.

Table 2-3. Selected hyperparameters for LSTM RNN predictive model tuning with their descriptions.

Hyperparameter	**Description**
RNN Nodes	Number of RNN nodes in the LSTM model. More nodes means a more complex model.
epochs	Number of epochs to use in training.
batch_size	Number of datapoints to use in a single batch
dropout	Regularization parameter to randomly ignore layer outputs to prevent overfitting
l1	Lasso regression which acts as a penalty to loss function and can reduce coefficients to 0.
l2	Ridge regression which acts as a penalty to loss function and can reduce coefficients to a smaller value.

2.7 Process of Creating Predictive Model

Diff_predictor was created using the coding language, Python, and can be downloaded from the GitHub repository (https://github.com/dash2927/diff_predictor). Diff_predictor uses data taken from Amazon Simple Storage Service (S3) located on Amazon Web Service (AWS). Trajectory data are downloaded as a dataframe using the AWS interface package, boto3, and MSDs and geometric features are calculated using diff_classifier. Using diff_predictor, all required data in an experiment are appended and a tag representing the prediction target is applied to each tracking datapoint. For age-prediction, the target is the age of rat pup from which the data are derived. For region-prediction, the target is the region of the rat pup brain that the data are taken. After the tag is applied, the dataframe is cleaned of all values that were not able to be calculated correctly. The dataframe is then balanced using undersampling so that all target categories have an equal amount of datapoints.

To prevent data leakage between testing and training sets when creating ensemble decision tree models, the datapoints are binned such that all datapoints that were used in the same statistical feature calculation are in the same bin. The datapoints are then split into training and testing sets based on a desired split. For example, a 0.7 training/testing split would have 70% of all data in the training set and 30% of data in the testing set. For binned feature data, datasets are split in a way that no data within bins are separated. Hyperparameters are then set and cross-validation is applied to the desired model for the training set. A random grid-search is applied at this time to choose which parameters perform the best. At the end of cross-validation, the model with the best performance and parameters will then be selected. This selected model will then be used with the test set that was derived earlier. For ensemble decision trees, SHAP is then used to analyze features.

3 PROJECTS

3.1 Using XGBoost for Classifying Age-Based Data

Aim of Study

Multiple particle tracking (MPT) is increasingly utilized as a powerful tool for tracking multiple nanoparticle mean-square displacement (MSD) values over time in both micro and macro environments. Nance *et al.* have used these data in their experiments to observe and develop nanotherapeutics for disease models in the brain parenchyma. To further enhance the usability of these data, Nance *et al.* have developed diff_classifier, which produces spatial features based on the collected temporal tracking data. Further analysis of these data show that there may be potential to characterize aspects of the nanoparticle attributes or the environment the nanoparticle is in using machine learning (ML) methods. Therefore, diff_predictor was created to characterize produced MPT and feature data by creating predictive models. XGBoost, a boosted decision tree, is one of the major predictive models used with diff_classifier generated data. In this project, data from an age-dependent diffusion study are characterized in postnatal day 14 (P14), postnatal day 21 (P21), postnatal day 28 (P28), and postnatal day 35 (P35) rat pups using an XGBoost predictive model. Shapley Additive Explanations (SHAP) are used to observe feature importance.

Results

A total of 50,444 MPT datapoints were used in the creation of the XGBoost age-based model. Each datapoint used represents calculated spatial features for a single particle trajectory. From these data, 6,416 datapoints were collected from a P14 pup, 20,665 datapoints were collected from a P21 pup, 6,194 datapoints were collected from a P28 pup, and 17,169 datapoints were collected from a P35 pup. To balance these data, each age category was undersampled using

random sampling until every category contained 6,194 trajectory datapoints. For this model, a 50/50 split of training/testing datasets resulted in the best accuracy.

Table 3-1. Calculated hyperparameter values for XGBoost age-based predictive model.

Hyperparameter	Value
max_depth	3
eta	0.005
min_child_weight	0
gamma	5
subsample	0.15
colsample_bytree	0.8

Hyperparameters were then found using a 5-fold cross-validation procedure on the training set while minimizing mean log-loss for varying boosting rounds. The log-loss value for the validation set successfully converged to a minimum value of 1.07. The calculated hyperparameters are in Table 3-2.

Table 3-2. Evaluation results for XGBoost age-based model. As shown, results of precision, recall, and f1-score for a single evaluation dataset are listed with their weighted average based on number of datapoints (support). An average accuracy score is also given based on prediction of 50 sets of data randomly sampled with replacement from a larger dataset.

	precision	recall	f1-score	support
P14	0.6733	0.7749	0.7206	1524
P21	0.4737	0.5250	0.4980	1543
P28	0.4825	0.3124	0.3793	1546
P35	0.5428	0.5952	0.5678	1534
weighted average	0.5426	0.5510	0.5407	6147
accuracy (avg. 50)			0.5429	6147

With hyperparameters applied, the model was evaluated using the testing set. The results of this evaluation are found in Table 3-2. The precisions of P14, P21, P28, and P35 pups were found to be 0.67, 0.47, 0.48, and 0.54 respectively. The lower precisions were similar and belonged to the P21 and P28 pups, whereas the P14 and P35 pups had the higher precisions. The recall values of P14, P21, P28, and P35 pups were found to be 0.77, 0.53, 0.31, and 0.59

respectively with the minimum recall belonging to P28 pups. The total accuracy of 50 run models for testing datasets with randomly selected data was found to be 54.29 ± 0.46% with a weighted f1-score of 0.54.

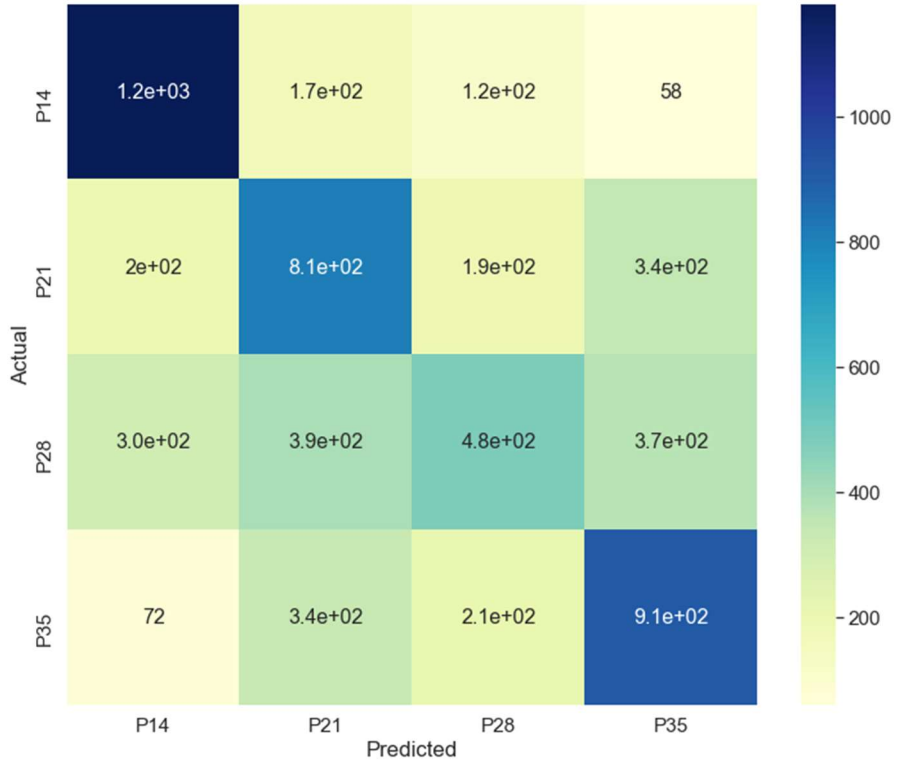

Figure 3-1. Confusion matrix for evaluation results of XGBoost age-based predictive model.

As shown in the confusion matrix (Figure 3-1), most false positive and false negative predictions belonged to the middle P21 and P28 groups. P14 had the highest true positive value of $1.2x10^3$ with a specificity of 0.79. The second-best performing class was P35 with a true positive value of 913 and a specificity of 0.76. The worst performing class was P28 with only 483 true positive values and a specificity of 0.85. The full table showing all true positive, true negative, false positive, false negative values can be found in Appendix (Table A-2).

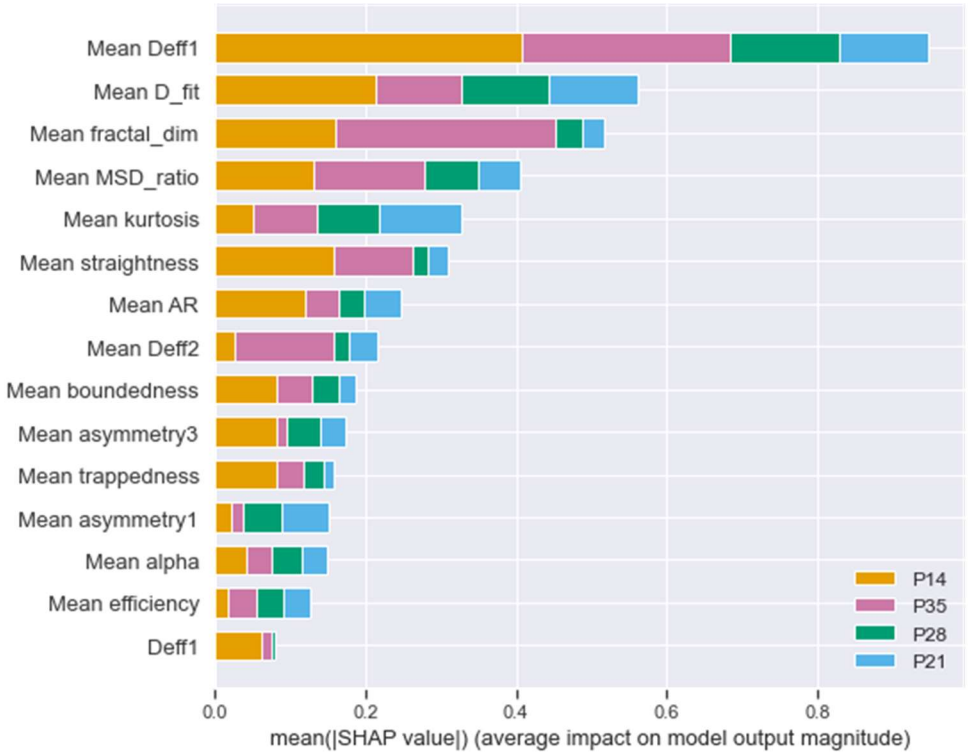

Figure 3-2. SHAP summary plot for feature importance of XGBoost age-based predictive model. Measured in magnitude SHAP value representing the average impact on model output.

To further analyze how features affect classification, a SHAP summary plot was created detailing the top 15 most important features in the model prediction (Figure 3-2). In addition, the SHAP summary plot shows how important each feature is toward predicting an age class. As shown, the top five total most important features labeled as mean Deff1, mean D_fit, mean fractal_dim, mean MSD_ratio, and mean kurtosis. The only important non-statistical calculated feature in the summary plot is Deff1.

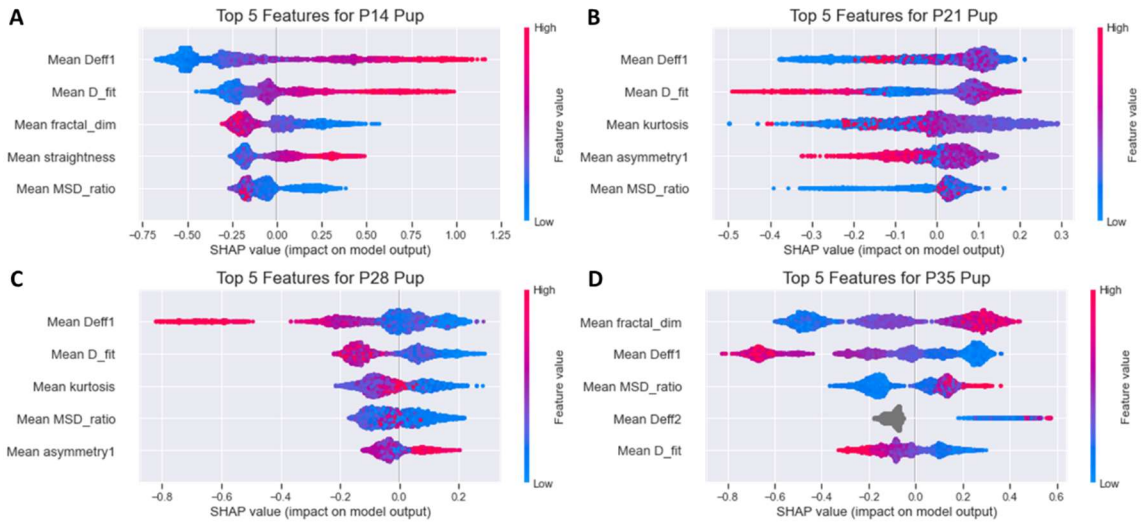

Figure 3-3. Individual SHAP summary plots for P14 (A), P21 (B), P28 (C), and P35 (D). A SHAP value further away from 0.0 represents a higher impact on prediction for that category. A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

For the five most important features, individual SHAP summary plots were created to show how the relative feature importance changes for varying values at each age class (Figure 3-3). For example, for mean Deff1 and mean D_fit, large diffusion values are more positively associated with P14 pups, while lower diffusion values are more positively associated with P28 and P35 pups. These plots also show how indecisive predictions are for P21 and P28 pups, as shown by the SHAP values being closer to 0.0, and the purple points indicate values that are neither high nor low. Additional plots, which better show the SHAP distribution over feature value, can be found in the Appendix (Figure A-1 to Figure A-5). These plots show how prediction SHAP value importance changes with varying values in each feature.

Discussion

I explored further utility of MPT datasets through predictive analysis. Using an XGBoost classifier to observe age dependency in data, I developed a model that could predict the age of four individual classes with a 54.56 ± 0.46% accuracy (Table 3-2), which has a better accuracy relative to random guessing by 30.10%. While this model was able to predict extreme ages of P14 and P35 pups relatively well with a precision of 0.67 and 0.54, respectively, performance

was much lower for the two middle ages. This study was limited to rats that were only 7 days apart and within a 21-day window of early post-natal development. This lower precision may be due to the experimental window being too small and the datapoints of each category overlapping too much for an accurate prediction. To add, recall was also very low for the middle groups, especially for the P28 age group at 0.31, which suggests that the model was not annotating the P28 category properly. To alleviate this, adding more data to the dataset may be useful. However, P28 did perform better in specificity with a specificity of 0.85 (Table A-2). This result means that the model performed very well in identifying true negatives that are correctly identified as true negatives.

An advantage of decision tree classifier models is the ability to analyze feature importance. From produced SHAP summary plots, the five most important trajectory features were found to be mean D_{eff} at 0.33s (mean Deff1), mean fitted diffusion coefficient (mean D_fit), mean fractal dimension (mean fractal_dim), mean MSD ratio (mean MSD_ratio), and mean kurtosis (Figure 3-2). These important feature values provide significant insight into the type of interactions that nanoparticles are having within the brain parenchyma. For example, the prediction for the P14 pups show a higher correlation with both diffusion coefficients. This can be seen better from results in the Appendix Figure A-1 and Figure A-2 in that nanoparticles with diffusion coefficients above 0.003 $\mu m^2/s$ have a positive SHAP value, becoming more positively correlated with P14. In contrast, a diffusion coefficient greater than 0.003 $\mu m^2/s$ for P21, P28, and P32 age categories have a negative SHAP value and, therefore, are negatively correlated with their respective age classification. Lushnikov *et al.* suggested that a large kurtosis may imply non-linear interactions within biological environments such as a particle diffusing on a membrane within a compartmentalized-like region.[35] Based on this paper, for the P28 aged pups, lower amounts of interactions are associated with this age group due to its correlation with a lower kurtosis. Guigas *et al.* showed that fractal dimension can give clues as to a trajectory's

Brownian diffusion and determine how complete a space is explored by a nanoparticle.[36] Distributions of fractal dimensions on SHAP plots for P14 and P35 may give some insight into how nanoparticles are moving within the ECM.

Another observation is that out of the 15 top features in the SHAP summary plots, only one non- statistical feature, i.e., the D_{eff} at 0.33s is important for prediction (Figure 3-2). This result can exemplify the importance of MPT as a solution for classifying biological systems. The fact that MPT tracks thousands of nanoparticles at once allows for a higher amount of information to be collected based both on the environment containing the nanoparticles and the nanoparticles themselves relative to other solutions that cannot be observed as readily.

3.2 Developing an RNN for Temporal-Based Tracking Data

Aim of Study

An ensemble decision tree model such as XGBoost requires that geometric spatial features are created as inputs to the predictive model. This requirement creates a set of easily interpretable features that can help characterize individual particle trajectories. To make a prediction more robust, statistical features are calculated by taking the local average of every trajectory feature within 9μm x 9μm rectangular region of the source video. This method allows the predictive model to achieve accuracies as shown in the previous project. However, calculation of these spatial features can be computationally expensive. For a set of 20 trajectory files with an average of 500 trajectories per file, 35 minutes were required to compute all trajectories. This computational inefficiency can become overwhelming when computing features of large trajectory experiments, which can require processing up to 100,000 individual trajectories at a time. In addition, while XGBoost is an excellent predictor, an accuracy of only 54.56% for the previous project shows a need for improved predictive capability. To try and alleviate these problems, I used diff_predictor to implement two long short-term memory (LSTM) recurrent

neural networks (RNN). One network only uses previously calculated X, Y, and mean square displacement (MSD) data for model prediction. The other network uses X, Y, MSD, and non-statistical spatial features for model prediction. These predictors will be tested with the same age-related diffusion data from the previous project. The prediction models will then be compared to an XGBoost and a Random Forest predictive model.

Results

Two LSTM RNN predictive models were created using the Keras API for TensorFlow. The first model created is a sequential model which uses an LSTM layer, followed by a dropout layer, and two dense neural net layers. The second model is not sequential, and uses an LSTM layer, followed by a dropout layer and dense neural network layer. The output of the dense layer is then concatenated with auxiliary data fed with non-statistical spatial geometric feature data. This concatenated data is then fed through three dense neural network layers. The models use log-loss as a loss function and an Adam algorithm for optimization with a learning rate of 0.001.[37] The models also implement optional L1 and L2 bias regularizers in the LSTM layers which reduce likelihood of overfitting the model.

This experiment used the temporal data belonging to the same dataset that the XGBoost experiment used. Both models used 50,444 MPT trajectories with 651 timestep datapoints each for creation. 6,416 trajectories were collected from a P14 pup, 20,665 trajectories were collected from a P21 pup, 6,194 trajectories were collected from a P28 pup, and 17,169 trajectories were collected from a P35 pup. To balance this data, each age category was undersampled using random sampling until every category contained 6,194 trajectories. With 651 timesteps per trajectory, the entire dataset totaled to 16,129,176 datapoints. Datasets were transformed into an array that could be fed into an RNN classifier and was split into training and testing sets. The models were trained to minimize log-loss with varying RNN nodes, epochs, batch sizes, dropout ratios, and L1/L2 regularization values.

The regular LSTM was created with an 80/20 split of training/testing data. Log-loss was reduced to a converging value of 1.15. The modified auxiliary LSTM also had an 80/20 split of training/testing data. Log-loss converged to a value of 1.17. Graphs of the loss and accuracy convergence over model epochs can be found in the Appendix (Figure A-5 and Figure A-6). Optimal parameters using the training set are shown in Table 3-3.

Table 3-3. Calculated hyperparameter values for LSTM RNN age-based predictive model.

Hyperparameter	LSTM	LSTM-AUX
RNN Nodes	100	300
epochs	50	50
batch_size	175	175
dropout	0.0	0.0
l1	0.40	0.40
l2	0.50	0.50

The LSTM models were compared with the previously used XGBoost model and a created random forest (RF) model (Figure 3-4). The RF model was trained in a similar manner to XGBoost and resulting hyperparameters can be found in Appendix Table A-2. Models were trained and 50 predictions were made for each produced model using randomly selected values from the testing dataset. Results show that XGBoost outperformed all other tested predictive models with an average accuracy of 54.56 ± 0.46%. The regular LSTM performed the worst with an average accuracy of 49.91 ± 1.08%. The modified LSTM performed only slightly better with an accuracy of 50.69 ± 0.93%.

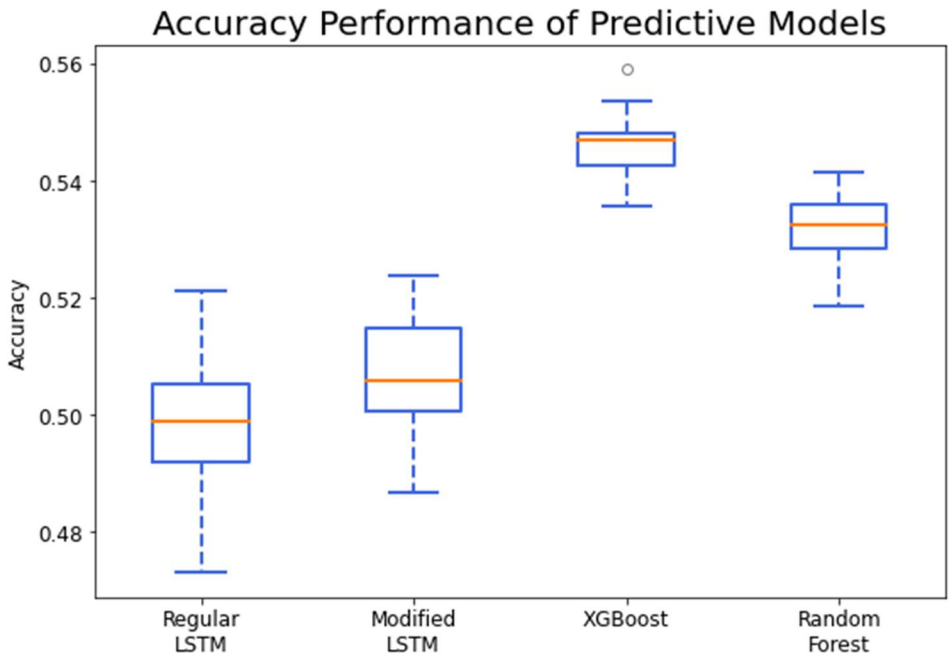

Figure 3-4. Accuracy boxplot from comparing four different predictive models: A regular LSTM, a modified LSTM with added auxiliary data, an XGBoost model, and a random forest model.

Discussion

A comparative study of four predictive models showed that XGBoost performed the best for the given dataset (Fig 3-4). The created RNN model did not provide a better prediction and had a slightly lower accuracy of 49.91 ± 1.08%. In addition, the LSTM models have a larger range of predictive accuracies than the decision tree models. Regardless, all models performed reasonably well with accuracies greater than 50%. In addition, the created LSTM used only the time-series data that included X, Y, and MSD values. In application, this method would save time by alleviating the need of computing spatial and statistical features for every produced trajectory. A modified LSTM was created which used only non-statistical spatial data. This only performed slightly better with an accuracy of 50.69 ± 0.93% (Figure 3-4). This result shows the importance of using MPT statistical spatial features in prediction models. The major issue with LSTM models is that these models do not produce any kind of feature importance for a prediction. This outcome makes the model harder to interpret and does not allow for further

analysis past prediction metrics. In all, LSTM can be a viable option if computational time is important, but for the tested dataset, the model did not perform better.

3.3 Using XGBoost for Classifying Region-Based Data

Aim of Study

The previous two studies showed that predictive models can be used to predict and analyze multiple particle tracking (MPT) datasets, however both had limitations. Predictions of age could be calculated based on both time-series and spatial data with accuracies up to 54%, but predictions with higher accuracies were not possible. The LSTM model, while potentially useful for the fact that it does not need feature data, did not result in a better prediction than XGBoost. Diff_predictor needs to be a robust package, and algorithms in diff_predictor need to work for a wide range of datasets and prediction targets. Literature has shown regional based differences within the brain in capillary density and vascularity differences,[38] microglia transcription within microglia-proliferating brain regions,[39] and capillary permeability.[40] Therefore, a region dependent dataset can be a potentially good match for predictive analysis. In this project, I used region-based data generated from cultured brain slices exposed to oxygen-glucose deprivation (OGD).[41] Data was obtained in the cortex, ganglia, hippocampus, striatum, and thalamus and analyzed using diff_predictor's XGBoost predictive model. In addition, SHAP was used to observe feature importance.

Results

Two datasets were combined to create a total of 24,519 MPT trajectories to use for region-based model creation. From the total data, 6,575 trajectories were collected from the cortex, 17,401 trajectories were collected from the striatum, 233 trajectories were collected from the ganglia, 124 trajectories were collected from the hippocampus, and 186 trajectories were collected from the thalamus. These data were highly unbalanced, with the lowest number of

trajectories belonging to the hippocampus regional class with 124 trajectory datapoints and the highest number belonging to the striatum with 17,401 trajectory datapoints. In order to balance the data, all region classes were undersampled using random sampling to 124 trajectory datapoints. For this model, an 80/20 split of training/testing datasets resulted in the best accuracy.

Table 3-4. Evaluation results for XGBoost region-based model with cortex, ganglia, hippocampus, striatum, and thalamus classes. As shown, results of precision, recall, and f1-score for a single evaluation dataset are listed with their weighted average based on number of datapoints (support). An average accuracy score is also given based on prediction of 50 sets of data randomly sampled with replacement from a larger dataset.

	precision	**recall**	**f1-score**	**support**
cortex	0.6429	0.8182	0.7200	11
ganglia	0.5000	0.5000	0.5000	16
hippocampus	0.6250	0.5000	0.5556	10
striatum	0.8333	0.7692	0.8000	13
thalamus	0.4667	0.4667	0.4667	15
weighted average	0.6024	0.6000	0.5981	65
accuracy (avg. 50)			0.6200	65

Hyperparameters were then found using a 5-fold cross-validation procedure on the training set while minimizing mean log-loss for varying boosting rounds. Log-loss value for the validation set successfully converged to a minimum value of 0.97. These hyperparameters are shown in the Appendix in Table A-4.

The results of using the testing set are found in Table 3-4. Precision of cortex, ganglia, hippocampus, striatum, and thalamus classes were found to be 0.64, 0.50, 0.63, 0.83, and 0.47 respectively. The lowest precision belonged to the thalamus, ganglia, and hippocampus classes while the highest precision belonged to the cortex and the striatum classes. The recall of cortex, ganglia, hippocampus, striatum, and thalamus classes were found to be 0.82, 0.50, 0.50, 0.77, and 0.47, respectively, with the minimum recall belonging to thalamus and ganglia classes. The

total mean accuracy of 50 run models for testing datasets with randomly selected data was found to be 62.00 ± 4.05% with a weighted f1-score of 0.55.

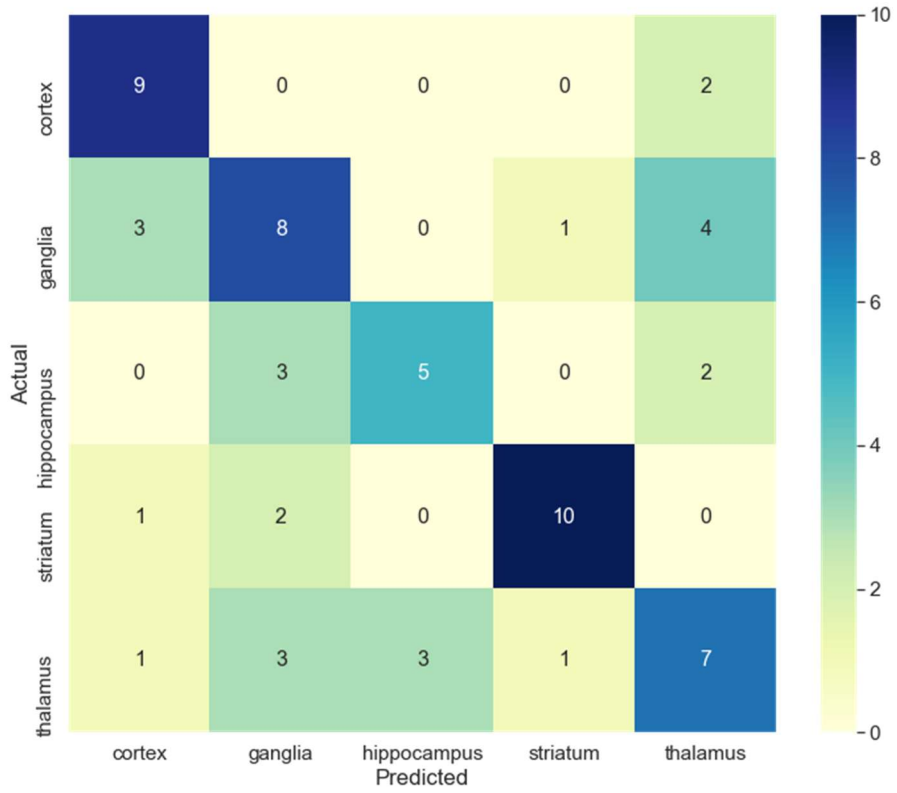

Figure 3-5. Confusion matrix for evaluation results of XGBoost region-based predictive model.

As shown in the confusion matrix (Figure 3-5), most false positive and false negative predictions belonged to the thalamus. The striatum had the highest true positive value of 10 with a specificity of 0.93. The second-best performing class was cortex with a true positive value of 9 and a specificity of 0.86. The worst performing class was hippocampus with only 5 true positive values and a specificity of 0.91. The full table showing all true positive, true negative, false positive, false negative values can be found in Appendix (Table A-6).

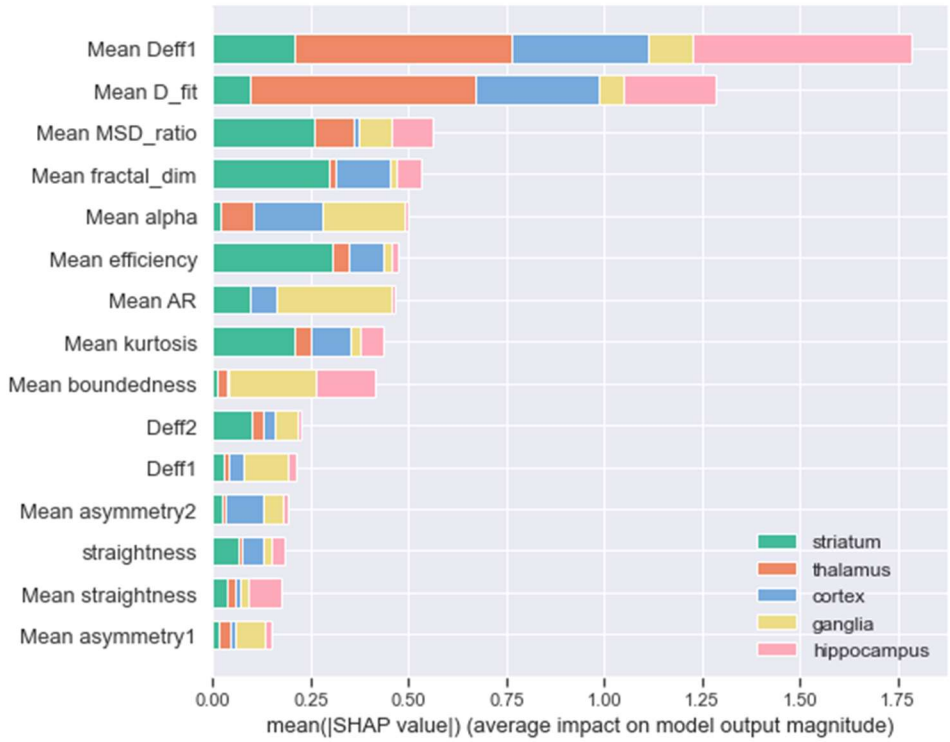

Figure 3-6. SHAP summary plot for feature importance of XGBoost region-based predictive model. Measured in magnitude SHAP value representing the average impact on model output.

SHAP summary plots were created detailing the top 15 most important features in model prediction. In addition, the SHAP summary plot shows how important each feature is toward predicting an age class. As shown in Figure 3-6, the top five total most important features categories were mean Deff1, mean D_fit, mean MSD_ratio, mean fractal_dim, mean MSD_ratio, and mean alpha. The two major discerning features that are most important to prediction are mean Deff1 and mean D_fit.

As a result of the highly unbalanced data, a second binary predictive model was created using only the cortex and striatum region classes. This dataset uses a total of 23,976 MPT trajectories with 6,575 trajectories collected from the cortex and 17,401 trajectories were collected from the striatum. The data are undersampled using random selection to 6,575 datapoints for each class. The model with the best accuracy had an 80/20 training/testing split.

Table 3-5. Evaluation results for XGBoost region-based model with cortex and striatum classes only. As shown, results of precision, recall, and f1-score for a single evaluation dataset are listed with their weighted average based on number of datapoints (support). An average accuracy score is also given based on prediction of 50 sets of data randomly sampled with replacement from a larger dataset.

	precision	**recall**	**f1-score**	**support**
cortex	0.8979	0.9230	0.9103	753
striatum	0.9153	0.8881	0.9015	706
weighted average	0.9064	0.9061	0.9060	1459
accuracy (avg. 50)			0.9135	1459

Hyperparameters were then found using a 5-fold cross-validation procedure on the training set while minimizing mean log-loss for varying boosting rounds. Log-loss value for the validation set successfully converged to a minimum value of 0.97. These hyperparameters are shown in Table A-5 in the Appendix.

The evaluation results of the testing set are found in Table 3-5. Precision of cortex and striatum were found to be 0.90 and 0.92 with a recall of 0.92 and 0.89, respectively. The total mean accuracy of 50 run models for testing datasets with randomly selected data was found to be 91.35 ± 0.57% with a weighted f1-score of 0.91. The confusion matrix in Figure 3-7 shows that false positives and false negatives are extremely low for each class. There are only 54 and 71 false negative and false positive values out of a total of 1,459 values predicted, respectively for cortex prediction relative to striatum.

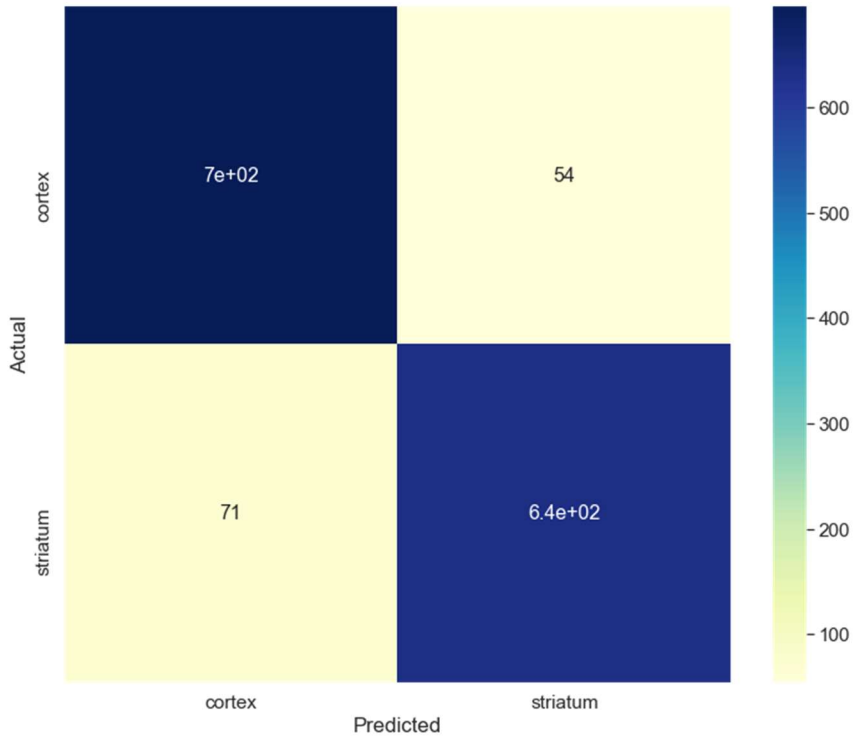

Figure 3-7. Confusion matrix for evaluation results of XGBoost region-based predictive model with only cortex and striatum classes.

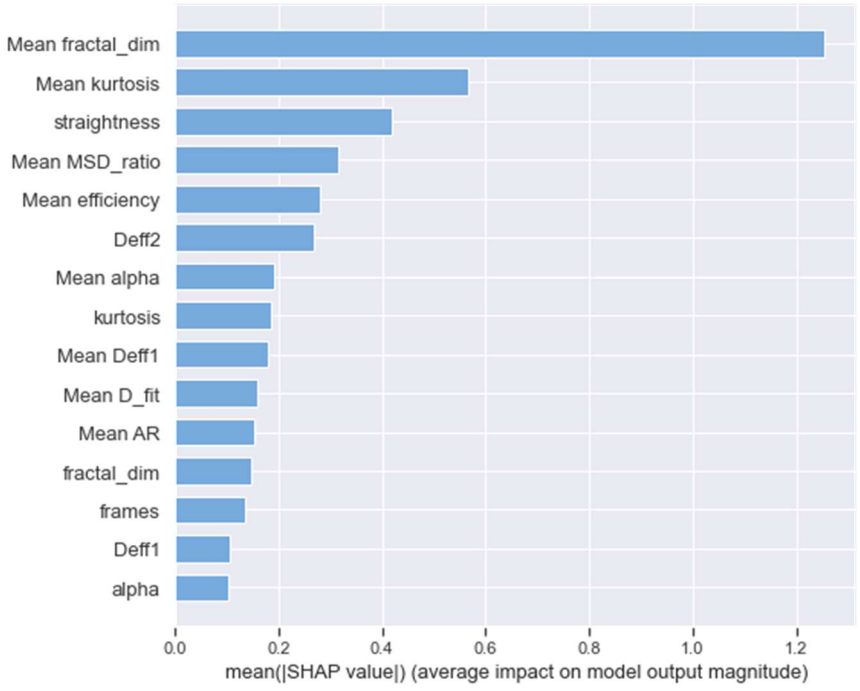

Figure 3-8. SHAP summary plot for feature importance of XGBoost region-based binary predictive model for cortex and striatum classes. Measured in magnitude SHAP value representing the average impact on model output.

The SHAP summary plot (Figure 3-8) shows vastly different top features for importance relative to the previous model (Figure 3-6). Mean fractal dimension is the largest deciding feature when it comes to predicting cortex class over striatum. The other four features with high importance in using the model are mean kurtosis, straightness, mean MSD_ratio, and mean efficiency.

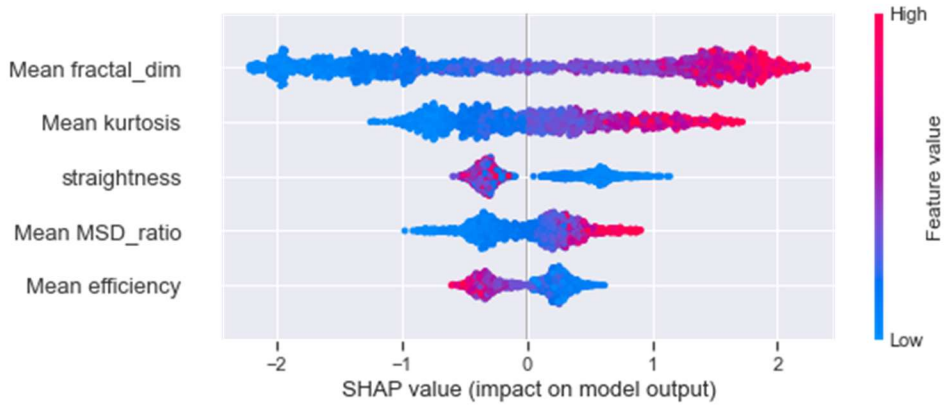

Figure 3-9. Individual SHAP summary plot for cortex prediction. A SHAP value further away from 0.0 represents a higher impact on prediction. A more positive SHAP value is correlated with a higher impact of a datapoint being classified as cortex while a more negative value is correlated with a higher impact of a datapoint being classified as striatum.

For the individual SHAP summary plot (Figure 3-9), clear distributions can be seen in the top five features. Prediction of the cortex is strongly positively correlated with large mean fractal dimension and kurtosis, while negatively correlated with straightness and mean efficiency.

Discussion

Classifying regional dependency in data resulted in a much better performing model than with the age dependent dataset. Using an XGBoost classifier resulted in a model that could predict the region of five individual classes with a 62.00 ± 4.05% accuracy (Table 3-4), which is much better than random guessing (20%) by a large margin. In addition, this dataset was very unbalanced, and a lot of data had to be discarded to maintain equal class sizes. Because of this, there is potential to have a more accurate prediction model if datasets can be increased in

size for ganglia, hippocampus, and thalamus regions. When the model decreased to only predicting two classes, the produced predictive model was 91.35 ± 0.57% accurate. This accuracy can be seen in the confusion matrix (Table 3-7) where only 125 out of 1,459 datapoints in the testing set were falsely predicted.

The performance of the binary prediction model produced is caused by a distinct correlation of mean fractal dimension (mean fractal_dim) and mean kurtosis between the cortex and striatum (Figure 3-9 and Figure A-13). As previously stated, kurtosis may suggest some type of complexity in the movement of a particle in a biological environment such as compartmentalized movement on a membrane.[35] Likewise, fractal dimension can suggest aspects of Brownian motion such as how complete a space in an environment is explored by a nanoparticle.[36] To add, high values of the third most important feature, straightness, is correlated with the striatum. These factors may suggest that there is a much more complex nanoparticle movement within the cortex, while particle movement in the striatum is straighter and simpler.

4 OTHER WORK

Diff_Classifier

In addition, the work for diff_predictor, I have also worked on multiple other projects in Nance Lab. One of my bigger projects involved work on diff_classifier. In this, I created easier-to-understand functions for non-coding members of the lab. I also removed the dependency of some methods on Amazon Web Service. I added multiple features to the feature list in order to improve characterization including mean turning angle, mean turning angle magnitude, turning angle variance, total distance, net distance, and progression.

D. Shackelford, C. Curtis, E. Nance. Exploring the Interface of Trajectory Datasets and Machine Learning in Biomedical Biological Applications. (in progress)

Abstract:

Researchers in biomedical and life sciences are increasingly turning to multiple particle tracking (MPT) as a method of data collection. Despite the application of MPT to a variety of problems in biomedical fields, most papers share a common formula: MPT is used to quantify a magnitude of transport such as a diffusion coefficient, and an ensemble measure of central tendency is reported. In terms of the ratio of raw data to the amount of data used, this approach has a low rate of return. However, machine learning is one toolset that can potentially be leveraged by researchers to expand the utility of MPT datasets. In this review, we summarize the ways machine learning methods have been implemented using trajectory datasets in the biomedical and life sciences literature. We also provide a brief overview of novel machine learning techniques implementing trajectory datasets from the artificial intelligence and computer science fields, including trajectory pre-processing steps, feature extraction, and machine learning algorithms. These tools have been leveraged to answer a number of non-biological classification problems based on trajectory datasets, including determining maritime vessel type, identifying suspicious behavior in surveillance video, and classifying human actions like waving or jumping. In the

biological setting, trajectories obtained from biological entities could be classified based on properties of the moving agent such as cell phenotype, nanoparticle surface functionality, or cell receptor activity, or based on aspects of the tissue microenvironment, such as state of disease or type of tissue. Integrating machine learning toolkits into biomedical sciences can allow analyses of many biological transport datasets, including trafficking of virus, protein, organelle, receptor, cell, and nanomaterials.

M. McKenna, **D. Shackelford**, H. Ferreira Pontes, B. Ball, T. Gao, E. Nance. Multiple particle tracking detects changes in brain extracellular matrix structure and predicts neurodevelopmental age. (in progress)

Abstract:

Brain extracellular matrix (ECM) structure mediates many aspects of neuronal function. Probing changes in ECM structure could provide insights into aging and neurological disease. Herein, we demonstrate the ability to characterize changes in brain ECM structure using multiple particle tracking (MPT). MPT was carried out in organotypic rat brain slices to detect induced and naturally occurring changes in ECM structure. Induced degradation of neural ECM led to a significant increase in nanoparticle diffusive ability in the brain extracellular space. For structural changes that occur naturally during development, an inverse relationship existed between age and nanoparticle diffusion. Using the age-dependent dataset, we applied extreme gradient boosting (XGBoost) to generate models capable of classifying nanoparticle trajectories. Collectively, this work demonstrates the utility of MPT combined with machine learning for measuring changes in brain ECM structure and predicting associated complex features such as developmental age.

5 SUMMARY, CONCLUSIONS, AND FUTURE WORK

In summary, a new predictive package, referred to as diff_predictor, was developed in this study. This package uses feature and trajectory datasets, along with prediction methods, such as XGBoost, recurrent neural networks, and random forest decision trees to make predictions and classifications about the biological environment and nanoparticle behavior. The developed package was then evaluated for application using a threefold approach. First, an extreme gradient boosting decision tree (XGBoost) was applied to a study of diffusion in a developing region within the brain to predict age based on spatial features that were analyzed for importance using values of Shapley Additive Explanations (SHAP). Second, a long short-term memory recurrent neural network algorithm was used on the temporal data from the same dataset and the results were compared with those based on the XGBoost and random forest models. Finally, demonstration of the versatility of the diff_predictor package was demonstrated using a new, different experimental dataset.

In conclusion, the methods within diff_predictor were shown to be able to further utilize MPT data within the brain parenchyma. In the first project, XGBoost within diff_predictor was shown to be able to predict Sprague-Dawley (SD) rat pups at varying ages based on MPT spatial features with a prediction accuracy of 54.29 ± 0.46%. It was then shown how feature analysis could be applied to MPT data using SHAP plots. This application provided the ability to look at both feature importance and trends in feature data for prediction of each age. In the second project, the MPT data were predicted using an LSTM-RNN using only temporal data. The results of the prediction show that the LSTM was able to produce similar accuracies to other methods (49.91 ± 1.08%) while also avoiding calculation of spatial features. In the third project, I showed the versatility in predicting MPT data by using XGBoost to make predictions based on five different regions of the brain was shown. Prediction was able to achieve an accuracy of 62.00 ± 4.05%, which was better than the accuracy for age-based data that

involved more classes to predict. Additionally, binary prediction between the two largest classes (striatum and cortex) performed very well with a prediction accuracy of 91.35 ± 0.57%. Overall, the three projects performed in this study show how prediction using diff_predictor can make the use of MPT data in nanoparticle studies more robust.

Future Work - Regional Data:

From regional based prediction of cortex, ganglia, hippocampus, striatum, and thalamus, a good predictive model was able to be created with an accuracy of 62% (Table 3-4). However, the datasets used in model creation were found to be extremely unbalanced, and datasets had to be undersampled to only 124 trajectories per class. As a result, the produced model could have been potentially improved. Adding data for thalamus, hippocampus, and ganglia data may improve the performance of this model and allow it to predict with higher accuracy.

In addition to this, other features of the brain environment can be measured to get more information out of prediction. One aspect of Shapley Additive Explanations (SHAP), is that it can show how a feature affects prediction after accounting for another individual feature effect. For example, a region-based study could be conducted in which pup sex is used as an additional feature. Using a SHAP interaction plot, changes in other features in each region when accounting for sex might give insight as to whether there are any differences in extracellular matrix structure between male and female pup brains. i.e. The SHAP value for prediction in the cortex might associate with lower or higher D_{eff} depending on sex. The same can be done for features such as size and shape of a nanoparticle, and age of the pup.

Future Work - Other Machine Learning Models:

Diff_predictor contains model tools for XGBoost, random forest, and LSTM RNNs. However, there are many other predictive models which can be used for MPT datasets. Logistic regression, which uses a logistic function to model binary variables.[42] This can be used with

spatial features to get class predictions. One problem with logistic regression, however, is that models do not perform well with many features. CNNs can also be utilized for MPT data. X and y data could be fed through an CNN to get any non-apparent trends. CNNs are very good at finding patterns in data that are in close proximity.[16]

6 CV

David C. Shackelford

dash2927@uw.edu
(970) 231-6492
4337 15th Ave. Northeast
Seattle, WA 98105

I. EDUCATION

Master of Science in Chemical Engineering-Data Science **08/18 – 08/20**
University of Washington, Seattle, WA
Thesis title: Utilizing Multiple Particle Tracking (MPT) Data and Machine Learning to Create Inferences in Particle Environment

Bachelor of Science in Chemical and Biological Engineering **01/12 – 05/16**
Colorado State University, Fort Collins, CO

Lockheed Martin Engineering Management Program **08/17 – 07/18**
University of Colorado, Boulder, CO

II. ENGINEERING EXPERIENCE

Research Assistant; Advisor: Prof. Elizabeth Nance, University of Washington, Seattle, WA **2018 – Present**

- Developed algorithms to extract spatial feature data in MPT x-y datasets of nanoparticles in postnatal rat brain slices.
- Used feature data in ensemble decision tree models (XGBoost, Random Forest) to predict nanoparticle environment and surface chemistry.
- Developed long-short-term-memory recurrent neural network (LSTM-RNN) using Tensorflow and Keras to predict diffusion characteristics based on temporal x-y datasets.
- Used developed predictors to observe regional-based diffusion patterns within the brain cortex, hippocampus, ganglia, striatum, and thalamus.
- Compared LSTM method to other methods in order to find optimal data pipeline.

Research Associate; Colorado State University, Fort Collins, CO **2016-2018**

- Performed bench-scale tests on a moving bed biofilm reactor (MBBR) system to simulate Kruger's AnitaMox process.
- Maintained MBBR to control growth of anaerobic ammonium oxidation (Anammox) bacteria, ammonia oxidizing bacteria (AOB), and nitrite oxidizing bacteria (NOB)to increase efficiency of deammonification.
- Performed RNA and DNA extraction on seeded media for downstream rt-PCR and qPCR analysis.
- Created master spreadsheets, coordinated site visits, conference calls Denver Metropolitan Wastewater Reclamation.

Research Assistant; Colorado State University, Fort Collins, CO **2016-2018**

- Reviewed design report for filter press dewatering system waste disposal at Joint Base Elmendorf-Richardson (JBER).
- Traveled on site to oversee and evaluate performance dewatering system with consultant ECC and vendor Watermark.
- Developed preliminary report summarizing system details and problems during operation.
- Developed final report detailing the design, implementation, and tech transfer of final project.

Research Assistant; Colorado State University, Fort Collins, CO **2014-2016**

- Tested and designed solutions to reduce chemical oxygen demand of water contaminated with airfield deicing fluid for JBER project and Center for Environmental Management of Military Lands (CEMML).
- Practiced safe lab techniques when handling hazardous biological and chemical materials.
- Coordinated with team to formulate plan to resolve obstacles and to meet deadlines during testing process.

III. PUBLICATIONS AND SOFTWARE PACKAGES

1. **D. Shackelford**, C. Curtis, E. Nance. Exploring the Interface of Trajectory Datasets and Machine Learning in Biomedical Biological Applications. (in preparation)
2. M. McKenna, **D. Shackelford**, H. Ferreira Pontes, B. Ball, T. Gao, E. Nance. Multiple particle tracking detects changes in brain extracellular matrix structure and predicts neurodevelopmental age. (in revision)
3. Diff_Predictor (https://github.com/dash2927/diff_predictor)

IV. POSTER PRESENTATIONS

1. Helmbrecht H., Lin R., McKenna M., **Shackelford D**., Nance E. (2020). Data science for developing analysis and prediction tools for neural images and videos. University of Washington Chemical Engineering Annual Graduate Student Poster Session.
2. Curtis C., Helmbrecht H., McKenna M., **Shackelford D**., Zhang M., Nance E. (2019) Implementing data science approaches at the interface of multi-particle tracking and histology datasets. University of Washington Chemical Engineering Annual Graduate Student Poster Session.
3. Boese J., Helmbrecht H., Scheiwiller S., **Shackelford D.** (2019). Protein Characterization and Classification in Pre-Clinical Neural Images. University of Washington Data Intensive Research Enabling Clean Technologies Symposium.

V. SKILLS AND CERTIFICATIONS

Programming: Proficient with C, MathCAD, Matlab, Microsoft Office, Python, R, SQLite, Java, bash

Data Science: AWS (EC2, S3), Hypothesis testing, Machine Learning, Data Analysis, Pandas, Numpy, Scikit-Learn, Tableau, TensorFlow/Keras, LSTM, XGBoost

Certifications: E.I.T. Certification (2017)

VI. PROFESSIONAL ORGANIZATIONS

ACES Master Student Representative **2019**
Tau Beta Pi Engineering Honor Society **2014 - Present**
American Institute for Chemical Engineers (AIChE) **2013 – Present**

7 BIBLIOGRAPHY

1. Ernst, R.L. and J.W. Hay, *The US Economic and Social Costs of Alzheimer's Disease Revisited.* American Journal of Public Health, 1994. **84**: p. 1261-1264.
2. Brookmeyer, R., et al., *Forecasting the Global Burden of Alzheimer's Disease.* Alzheimer's & Dementia: The Journal of the Alzheimer's Association, 2007. **3**(3): p. 186-191.
3. Legler, J.M., et al., *Brain and Other Central Nervous System Cancers: Recent Trends in Incidence and Mortality.* NCI: Journal of the National Cancer Institute, 1999. **91**(16): p. 1382-1390.
4. S, A., et al., *Changing Trends of Atherosclerotic Risk Factors Among Patients With Acute Myocardial Infarction and Acute Ischemic Stroke.* The American Journal of Cardiology, 2017. **119**(10): p. 1532-1541.
5. Brooks, C., et al., *Cost of medical care for a population-based sample of persons surviving traumatic brain injury.* The Journal of Head Trauma Rehabilitation, 1995. **10**(4): p. 1-13.
6. SM, P., et al., *How to improve R&D productivity: the pharmaceutical industry's grand challenge.* Nature Reviews: Drug Discovery, 2010. **9**.
7. VK, G. and K. LK, *The need for new approaches in CNS drug discovery: Why drugs have failed, and what can be done to improve outcomes.* Nanopharmacology, 2017. **120**: p. 11-19.
8. Nance, E.A., et al., *A Dense Poly(Ethylene Glycol) Coating Improves Penetration of Large Polymeric Nanoparticles Within Brain Tissue.* Science Translational Medicine, 2012. **4**(149).
9. B, W., *Drug targeting strategies into the brain for treating neurological diseases.* Journal of Neuroscience Methods, 2018. **311**: p. 133-146.
10. Kreuter, J., *Nanoparticles—A historical perspective.* International journal of pharmaceutics, 2007. **331**: p. 1-10.
11. Xu, Q.G., et al., *Nanoparticle diffusion in, and microrheology of, the bovine vitreous ex vivo.* Journal of Controlled Release, 2013. **167**(1): p. 76-84.
12. Labouta, H.I., et al., *Gold Nanoparticle Penetration and Reduced Metabolism in Human Skin by Toluene.* Pharmaceutical Research, 2011. **28**(11): p. 2931.
13. Chen, C. and J. Suh, *Real-time particle tracking for studying intracellular transport of nanotherapeutics.* Organelle-Specific Pharmaceutical Nanotechnology, 2010: p. 161.
14. Suh, J., M. Dawson, and J. Hanes, *Real-time multiple-particle tracking: applications to drug and gene delivery.* Advanced Drug Delivery Reviews, 2005. **57**(1): p. 63-78.
15. Dosset, P., et al., *Automatic detection of diffusion modes within biological membranes using back-propagation neural network.* Bmc Bioinformatics, 2016. **17**.
16. Granik, N., et al., *Single-Particle Diffusion Characterization by Deep Learning.* Biophysical Journal, 2019. **117**(2): p. 185-192.
17. Curtis, C., et al., *Predicting in situ nanoparticle behavior using multiple particle tracking and artificial neural networks.* Nanoscale, 2019.
18. Helmuth, J.A., et al., *A novel supervised trajectory segmentation algorithm identifies distinct types of human adenovirus motion in host cells.* Journal of Structural Biology, 2007. **159**(3): p. 347-358.
19. Wagner, T., et al., *Classification and Segmentation of Nanoparticle Diffusion Trajectories in Cellular Micro Environments.* Plos One, 2017. **12**(1).
20. Curtis, C., A. Rokem, and E. Nance, *diff_classifier: Parallelization of multi-particle tracking video analyses.* Journal of open source software, 2019. **4**(36): p. 989.
21. Kimmel, J., A. Brack, and W. Marshall, *Deep convolutional and recurrent neural networks for cell motility discrimination and prediction.* IEEE/ACM Trans Comput Biol Bioinform, 2019.
22. Trevor Hastie, R.T., Jerome Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Vol. 2. 2009: Springer.

23. Orzechowski, P., W. La Cava, and J. Moore, *Where are we now? A large benchmark study of recent symbolic regression methods*. 2018.
24. Liaw, A. and M. Wiener, *Classification and regression by randomForest.* R news, 2002. **2**(3): p. 18-22.
25. Probst, P., M.N. Wright, and A.L. Boulesteix, *Hyperparameters and tuning strategies for random forest.* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2019. **9**(3): p. e1301.
26. Shapley, L.S., *A value for n-person games.* Contributions to the Theory of Games, 1953. **2**(28): p. 307-317.
27. Lundberg, S.M. and S.-I. Lee. *A unified approach to interpreting model predictions*. in *Advances in neural information processing systems*. 2017.
28. Zhang, H. and L.T. Maloney, *Ubiquitous log odds: a common representation of probability and frequency distortion in perception, action, and cognition.* Frontiers in neuroscience, 2012. **6**: p. 1.
29. Miao, Y., M. Gowayyed, and F. Metze. *EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding*. in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. 2015. IEEE.
30. Li, H., et al., *Spelling error correction using a nested rnn model and pseudo training data.* arXiv preprint arXiv:1811.00238, 2018.
31. Wu, H., et al. *Modeling trajectories with recurrent neural networks*. 2017. IJCAI.
32. Hochreiter, S. and J. Schmidhuber, *Long short-term memory.* Neural computation, 1997. **9**(8): p. 1735-1780.
33. Gers, F.A., J. Schmidhuber, and F. Cummins, *Learning to forget: Continual prediction with LSTM.* 1999.
34. Moore, R. and J. DeNero. *L1 and L2 regularization for multiclass hinge loss models*. in *Symposium on machine learning in speech and language processing*. 2011.
35. Lushnikov, P.M., P. Šulc, and K.S. Turitsyn, *Non-Gaussianity in single-particle tracking: Use of kurtosis to learn the characteristics of a cage-type potential.* Physical Review E, 2012. **85**(5): p. 051905.
36. Guigas, G. and M. Weiss, *Sampling the cell with anomalous diffusion—the discovery of slowness.* Biophysical journal, 2008. **94**(1): p. 90-94.
37. Kingma, D.P. and J. Ba, *Adam: A method for stochastic optimization.* arXiv preprint arXiv:1412.6980, 2014.
38. Cavaglia, M., et al., *Regional variation in brain capillary density and vascular response to ischemia.* Brain research, 2001. **910**(1-2): p. 81-93.
39. Furube, E., et al., *Brain Region-dependent Heterogeneity and Dose-dependent Difference in Transient Microglia Population Increase during Lipopolysaccharide-induced Inflammation.* Scientific Reports, 2018. **8**(1): p. 2203.
40. Blasberg, R., et al., *Regional blood flow, capillary permeability, and glucose utilization in two brain tumor models: preliminary observations and pharmacokinetic implications.* Cancer Treatment Reports, 1981. **65**: p. 3-12.
41. Joseph, A., et al., *Nanoparticle-microglial interaction in the ischemic brain is modulated by injury duration and treatment.* Bioengineering & Translational Medicine: p. e10175.
42. Wright, R.E., *Logistic regression.* 1995.

8 APPENDICES

Table A-1. Calculated spatial features

Feature	Model Abbreviation	Description	How Feature is Determined
alpha (α)	alpha	Exponent of the anomalous diffusion equation.	Fitting the anomalous diffusion equation: $MSD = 4D_{fit}\tau^{\alpha}$ using non-linear least squares regression and input MSD vs. lag time (τ) data.
Effective diffusion coefficient (D_{fit})	D_fit	Coefficient of the anomalous diffusion equation.	Fitting the anomalous diffusion equation: $MSD = 4D_{fit}\tau^{\alpha}$ using non-linear least squares regression and input MSD vs lag time (τ) data.
Kurtosis (K)	kurtosis	The fourth moment of the projected positions on the dominant eigenvector of the radius of gyration tensor. Can show nonlinearity in trajectory such as compartmentalization.	$K = \frac{1}{N}\sum_{i=0}^{N-1}\frac{\left(x_i^p - \overline{x^p}\right)^4}{\sigma_{x^p}^4}$
Asymmetry1 (a_1)	asymmetry1	Characterizes asymmetry of trajectory. Equals 0 for circularly symmetric trajectory and 1 for linear trajectory	$a = \frac{\left(\lambda_1^2 - \lambda_2^2\right)^2}{\left(\lambda_1^2 + \lambda_2^2\right)^2}$, where $\lambda_1$and λ_2 are the eigenvalues of radius of gyration tensor T: $T = \begin{pmatrix} \frac{1}{N}\sum_{j=1}^{N}\left(x_j - \langle x\rangle\right)^2 & \frac{1}{N}\sum_{j=1}^{N}(x_j - \langle x\rangle)(y_j - \langle y\rangle) \\ \frac{1}{N}\sum_{j=1}^{N}(x_j - \langle x\rangle)(y_j - \langle y\rangle) & \frac{1}{N}\sum_{j=1}^{N}\left(y_j - \langle y\rangle\right)^2 \end{pmatrix}$
Asymmetry2 (a_2)	asymmetry2	Ratio of smaller to larger principal radius of gyration	$a_2 = \frac{\lambda_2}{\lambda_1}$
Asymmetry3 (a_3)	asymmetry3	An asymmetry feature that accounts for non-cylindrical symmetric point distributions	$a3 = -\log\left(1 - \frac{(\lambda_1 - \lambda_2)^2}{2(\lambda_1 - \lambda_2)^2}\right)$
Aspect Ratio (AR)	AR	Ratio of the long and short side of a trajectory based on the minimum and maximum.	$AR = \begin{cases} \frac{\lvert x_{\max} - x_{min}\rvert}{\lvert y_{\max} - y_{min}\rvert}, & \lvert y_{\max} - y_{min}\rvert < \lvert x_{\max} - x_{min}\rvert \\ \frac{\lvert y_{\max} - y_{min}\rvert}{\lvert x_{\max} - x_{min}\rvert}, & \lvert y_{\max} - y_{min}\rvert \geq \lvert x_{\max} - x_{min}\rvert \end{cases}$
Elongation	elongation	An estimation of amount of extension a trajectory has from its centroid.	$Elongation = 1 - \left(\frac{1}{AR}\right)$
Boundedness (B)	boundedness	Quantifies how much a particle with diffusion coefficient, D_{eff} is restricted by a circular confinement of radius r when diffusing for a period of time $N\Delta t$	$B = \frac{D_{eff}N\Delta t}{r^2}$
Fractal Dimension (D_f)	fractal_dim	Measure of how "complicated" a trajectory is. How complete a space is filled by a nanoparticle	$D_f = \frac{\log(N)}{\log(NdL^{-1})}$
Trappedness (p_t)	trappedness	The probability that a particle with diffusion coefficient (D_{eff}) is trapped in a region (r_0) for a period of time $N\Delta t$	$p_t = 1 - \exp\left(0.2048 - 0.25117\left(\frac{D_{eff}N\Delta t}{r_0^2}\right)\right)$

Efficiency (E)	efficiency	The ratio of squared net displacement to the sum of squared step lengths	$E = \frac{((x_{N-1} - x_0)^2 + (y_{N-1} - y_0)^2)}{\sum_{i=1}^{N-1}(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}$
Straightness (S)	straightness	The ratio of net displacement to the sum of step lengths	$S = \frac{\sqrt{(x_{N-1} - x_0)^2 + (y_{N-1} - y_0)^2}}{\sum_{i=1}^{N}\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}}$
MSD Ratio (MSD)	MSD_ratio	Characterizes the shape of the MSD curve. 0 for Brownian motion, < 0 for restricted motion, > 0 for directed motion.	$MSD_{n_1,n_2} = \frac{MSD_{n_1}}{MSD_{n_2}} - \frac{n_1}{n_2}$
Frames	frames	Total number of frames in trajectory	$Frames = N$
Mean Turning Angle (θ_{mean})	angle_mean	Trajectory mean of the turning angle which is the counterclockwise angle from one point to another	$\frac{1}{N}\sum_{i=1}^{N}\theta_{i,i+1}$
Mean Turning Angle Magnitude (θ_{mean})	angle_mag_mean	Trajectory mean of the magnitude of turning angle which is the counterclockwise angle from one point to another	$\frac{1}{N}\sum_{i=1}^{N}\lvert\theta_{i,i+1}\rvert$
Turning Angle Variance (θ_{var})	angle_var	Trajectory variance of the turning angle which is the counterclockwise angle from one point to another	$\frac{\sum_{i=1}^{N}(\theta_{i,i+1} - \bar{\theta})^2}{N}$
Total Distance (d_{tot})	dist_tot	Total distance particle travels throughout trajectory	$d_{total} = \sum_{i=1}^{N}\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}$
Net Distance (d_{net})	dist_net	Net distance particle has traveled throughout trajectory	$d_{net} = \sqrt{(x_N - x_0)^2 + (y_N - y_0)^2}$
Progression	progression	Ratio of the net distance traveled, and the total distance traveled	$progression = \frac{d_{net}}{d_{total}}$
Effective diffusion at 0.33s (D_{eff})	Deff1	Effective diffusion coefficient at 0.33 s	$Deff1 = \frac{MSD_{\tau=0.1}}{4 * 0.1}$
Effective diffusion at 3.3s (D_{eff})	Deff2	Effective diffusion coefficient at 3.3 s	$Deff2 = \frac{MSD_{\tau=1}}{4 * 1}$
Mean values were calculated for based on a window of 120 pixels in which every trajectory within the window gets statistical calculations for alpha, D_fit, kurtosis, asymmetry1, asymmetry2, asymmetry3, AR, elongation, boundedness, fractal_dim, trappedness, efficiency, straightness, MSD_ratio, Deff1, and Deff2.			

Table A-2. True positive (TP), true negative (TN), false positive (FP) and false negative (FN) counts for evaluation of XGBoost age predictive model for P14, P21, P28, and P35 classes.

	TP	TN	FP	FN
P14	1153	2184	584	2226
P21	814	2523	894	1916
P28	478	2859	571	2239
P35	892	2445	761	2049

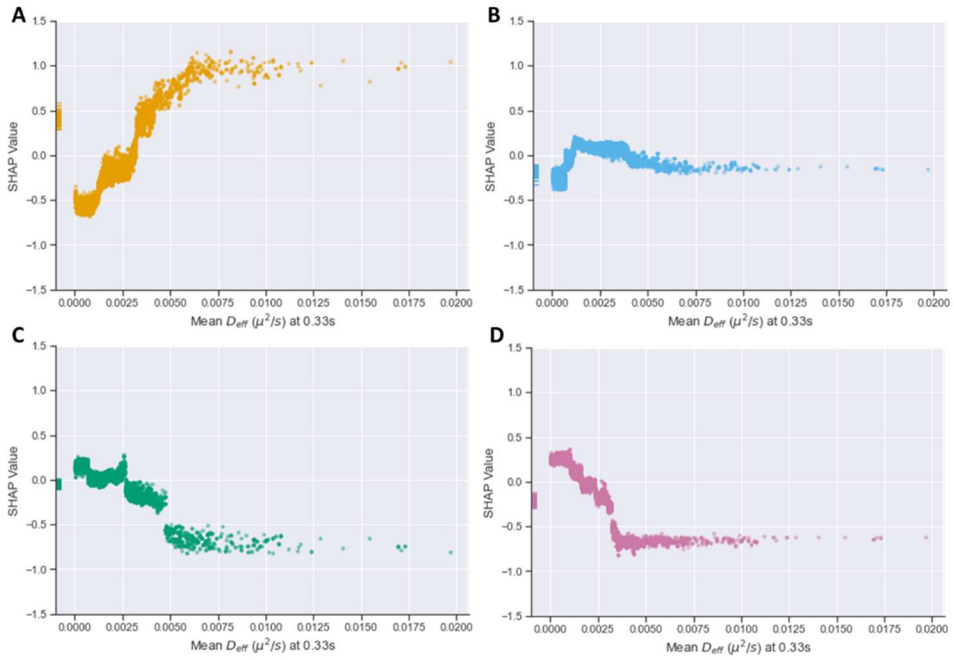

Figure A-1. SHAP dependency plot for mean effective diffusion at 0.33s for P14 (A), P21 (B), P28 (C), and P35 (D). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

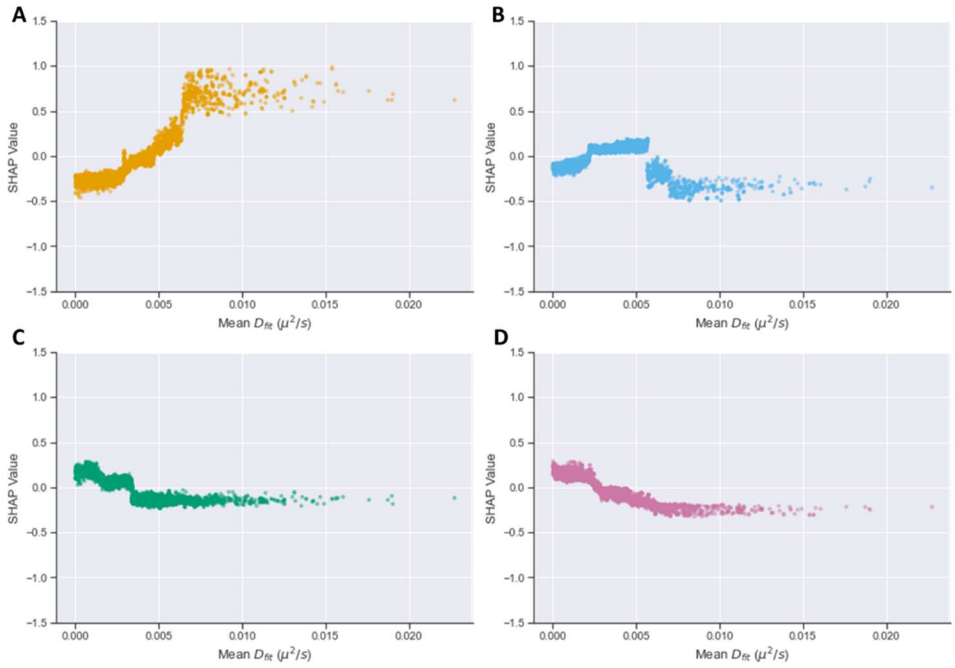

Figure A-2. SHAP dependency plot for mean fitted diffusion for P14 (A), P21 (B), P28 (C), and P35 (D). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

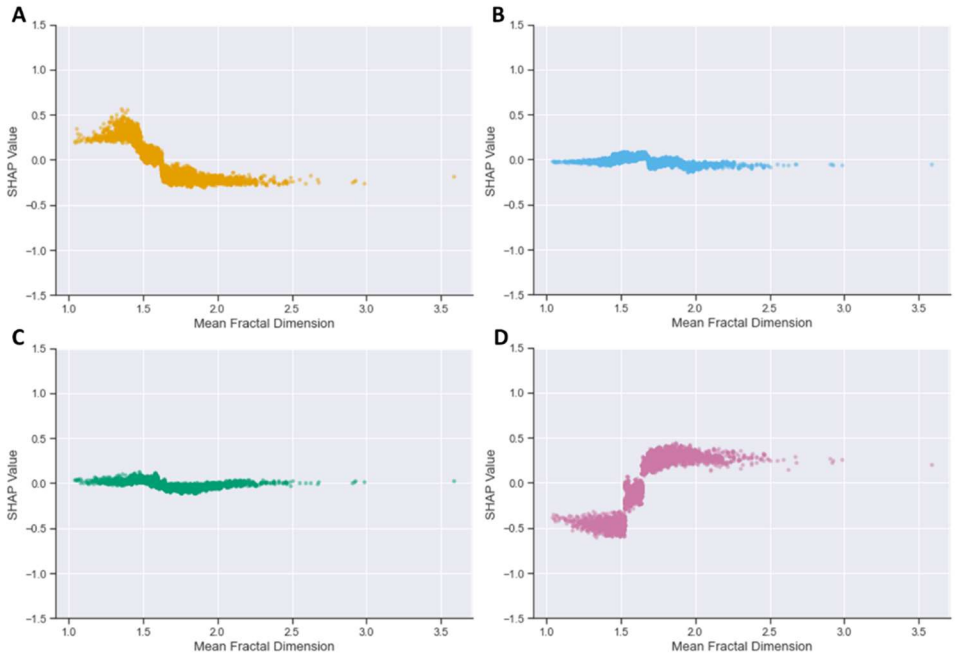

Figure A-3. SHAP dependency plot for mean fractal diffusion for P14 (A), P21 (B), P28 (C), and P35 (D). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

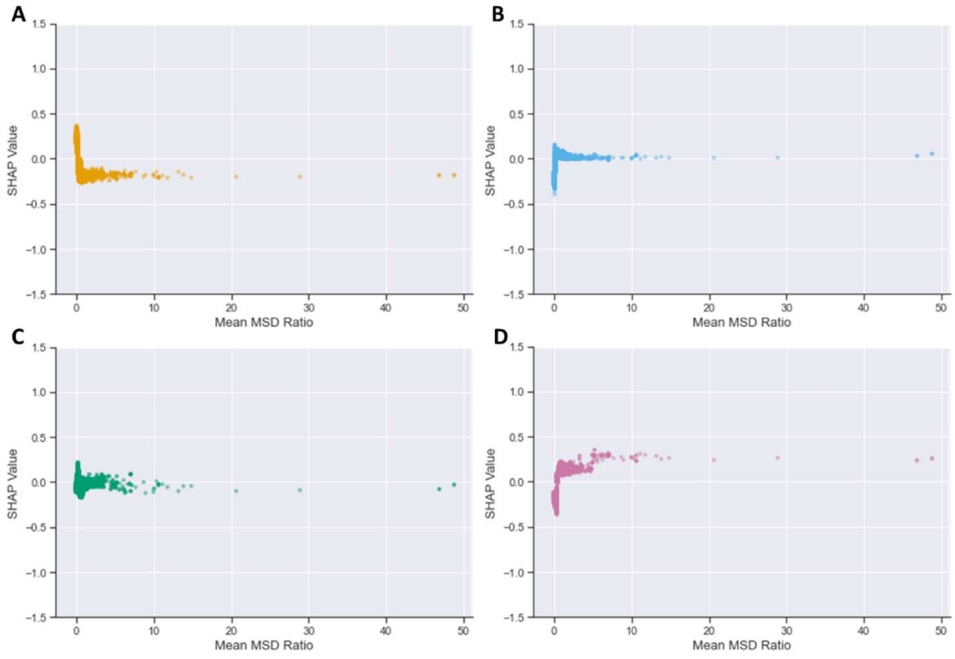

Figure A-4. SHAP dependency plot for mean MSD ratio for P14 (A), P21 (B), P28 (C), and P35 (D). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

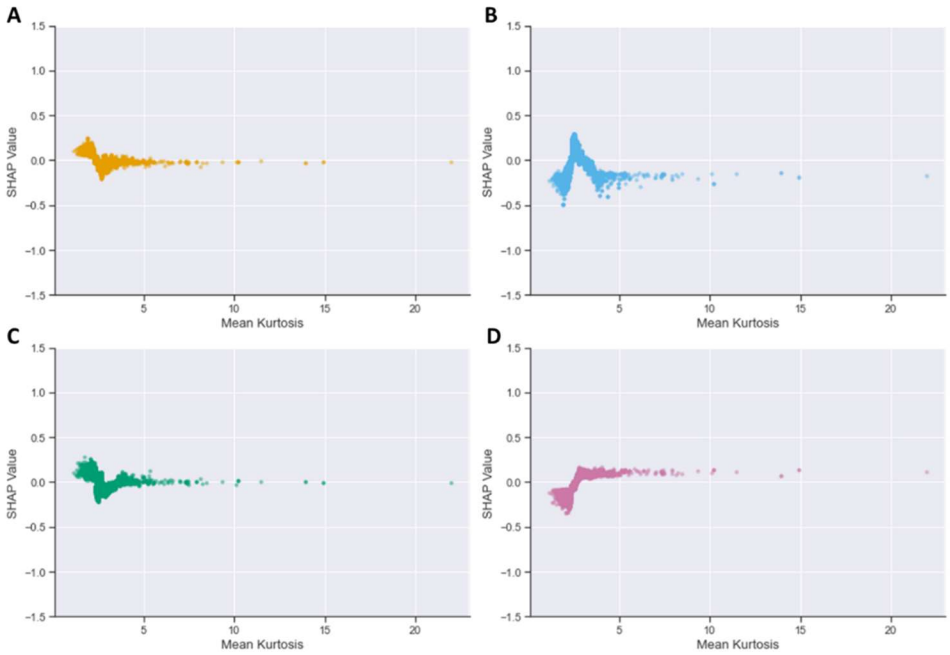

Figure A-5. SHAP dependency plot for mean kurtosis for P14 (A), P21 (B), P28 (C), and P35 (D). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

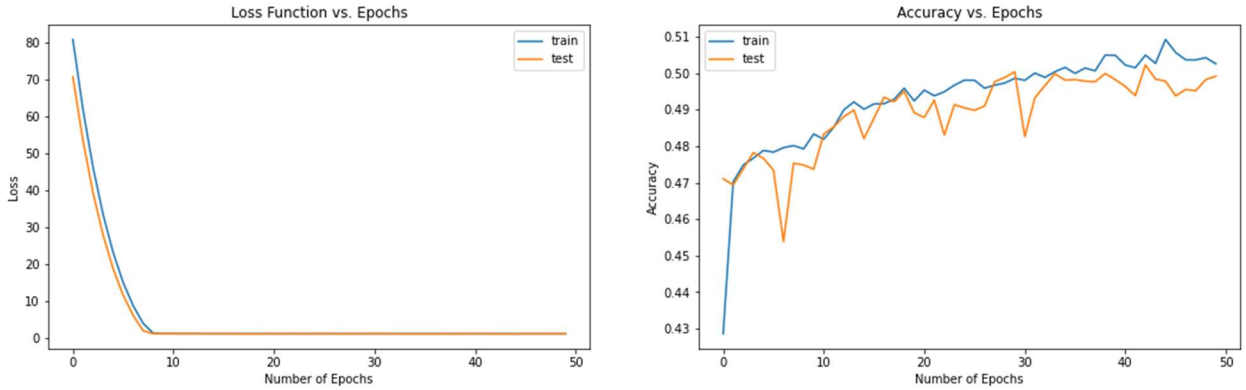

Figure A-6. Loss and accuracy values for training and evaluation sets over epoch when training for regular LSTM RNN predictive model on age-based data.

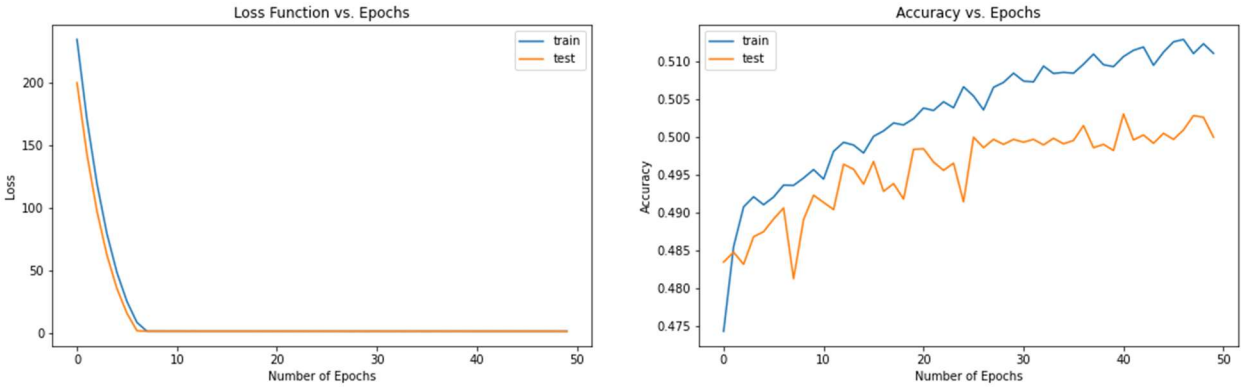

Figure A-7. Loss and accuracy values for training and evaluation sets over epoch when training for modified LSTM RNN predictive model on age-based data.

Table A-3. Calculated hyperparameter values for random forest age-base predictive model

Hyperparameter	Value
n_estimators	1100
min_samples_split	2
min_samples_leaf	4
max_depth	6

Region-Based-Analysis

Table A-4. Calculated hyperparameter values for XGBoost region-based predictive model when predicting for cortex, ganglia, hippocampus, striatum, and thalamus.

Hyperparameter	Value
max_depth	3
eta	0.05
min_child_weight	3
gamma	0
subsample	0.7
colsample_bytree	0.6

Table A-5. Calculated hyperparameter values for XGBoost region-based binary predictive model when predicting for cortex and striatum.

Hyperparameter	Value
max_depth	7
eta	0.05
min_child_weight	9
gamma	0.2
subsample	0.6
colsample_bytree	0.5

Table A-6. True positive (TP), true negative (TN), false positive (FP) and false negative (FN) counts for evaluation of XGBoost region predictive model for cortex, ganglia, hippocampus, striatum, and thalamus.

	TP	TN	FP	FN
Cortex	9	30	5	21
Ganglia	8	31	8	18
Hippocampus	5	34	3	23
Striatum	10	29	2	24
Thalamus	7	32	8	18

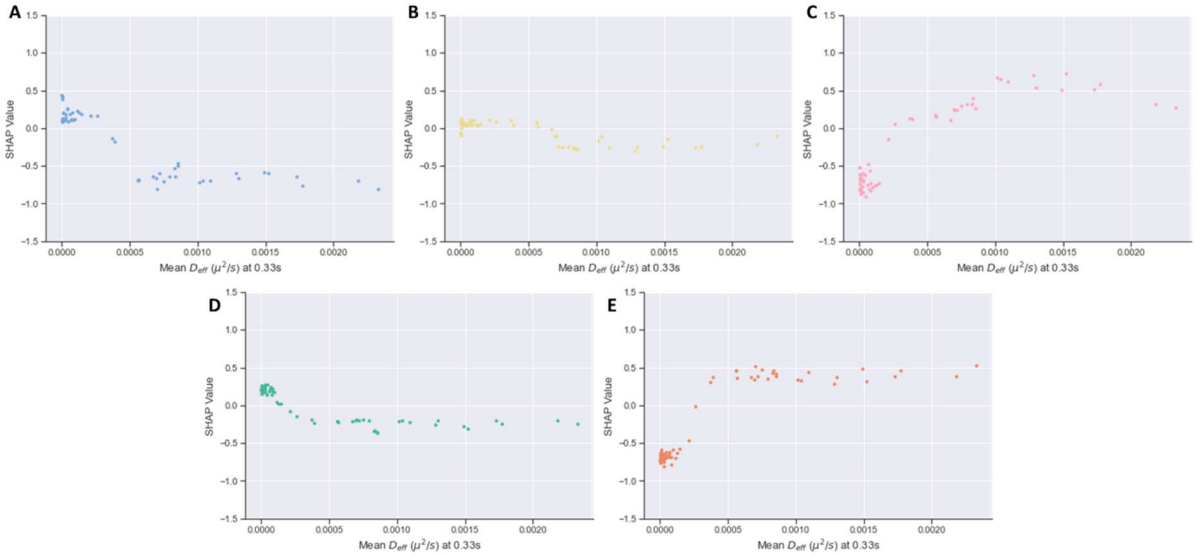

Figure A-8. SHAP dependency plot for mean effective diffusion at 0.33s for cortex (A), ganglia (B), hippocampus (C), striatum (D), and thalamus (E). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

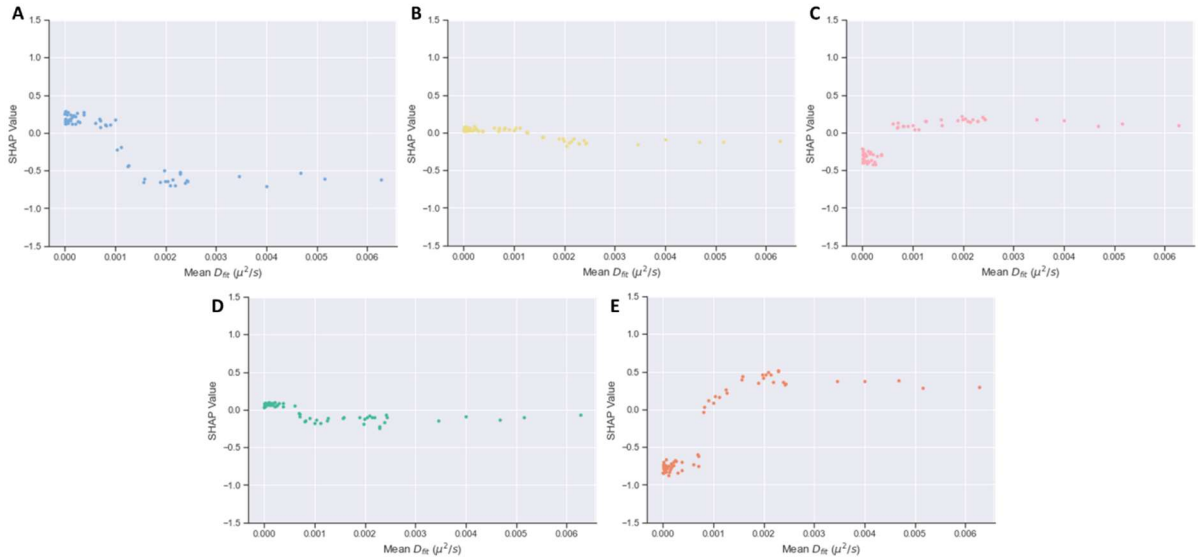

Figure A-9. SHAP dependency plot for mean fitted diffusion for cortex (A), ganglia (B), hippocampus (C), striatum (D), and thalamus (E). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

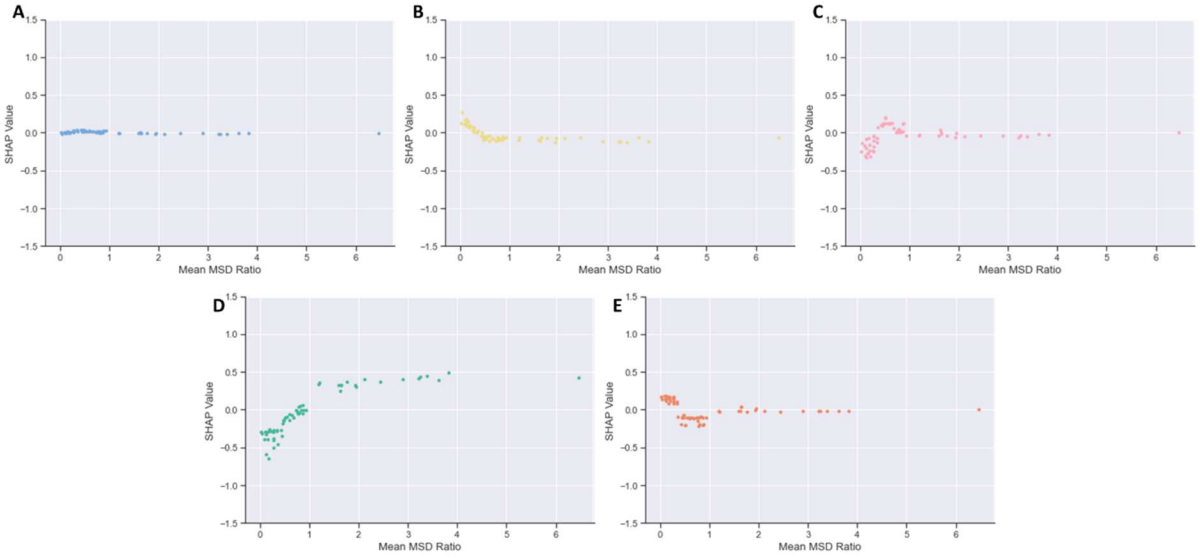

Figure A-10. SHAP dependency plot for mean MSD ratio for cortex (A), ganglia (B), hippocampus (C), striatum (D), and thalamus (E). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

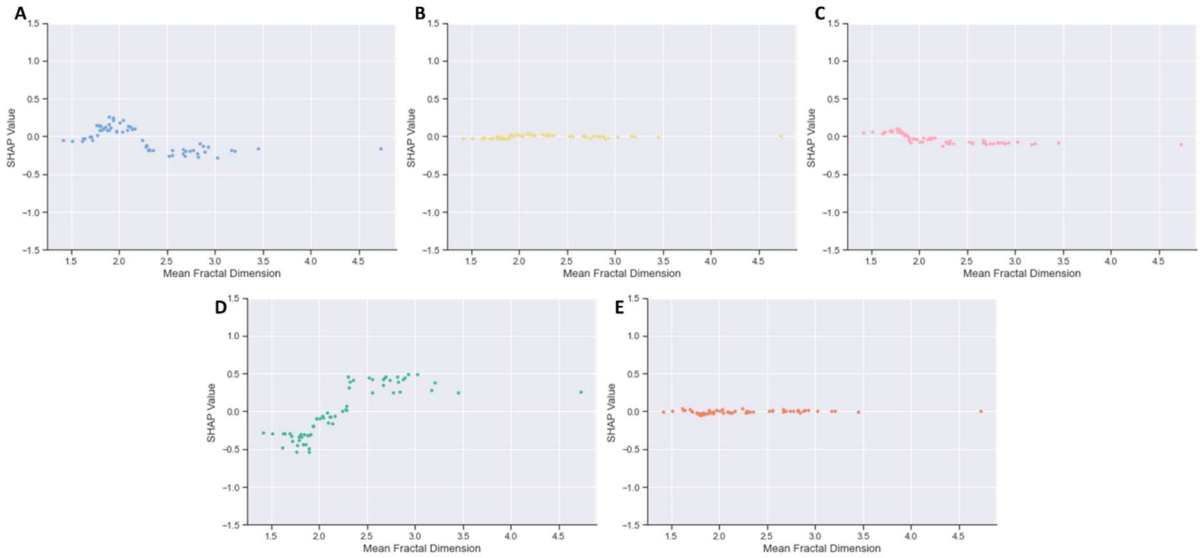

Figure A-11. SHAP dependency plot for mean fractal dimension for cortex (A), ganglia (B), hippocampus (C), striatum (D), and thalamus (E). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

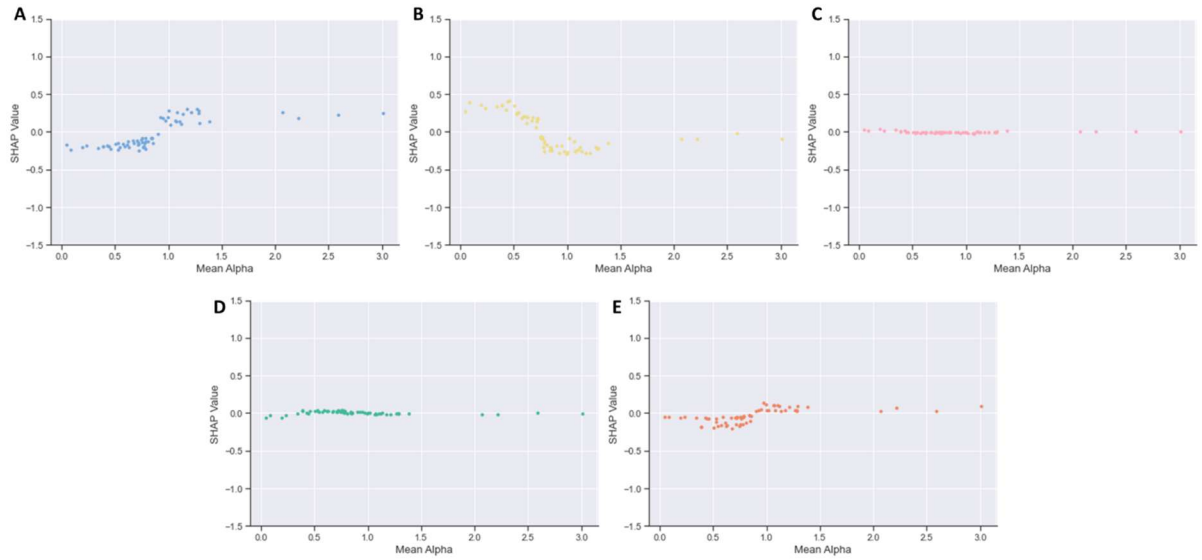

Figure A-12. SHAP dependency plot for mean alpha for cortex (A), ganglia (B), hippocampus (C), striatum (D), and thalamus (E). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as the specified class while a more negative value is correlated with a higher impact of a datapoint not being classified as the specified class.

Table A-7. True positive (TP), true negative (TN), false positive (FP) and false negative (FN) counts for evaluation of XGBoost region-based binary predictive model for cortex and striatum.

	TP	TN	FP	FN
Cortex	696	638	71	54
Striatum	638	696	54	71

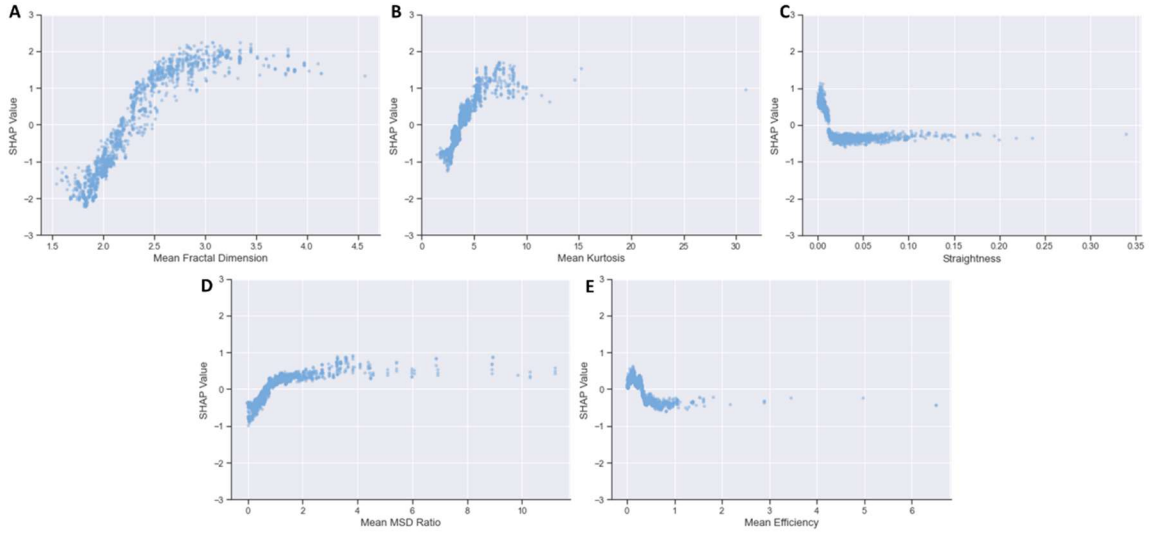

Figure A-13. SHAP dependency plot for cortex when observing the top five features for prediction: mean fractal dimension (A), mean kurtosis (B), straightness (C), MSD ratio (D), and mean efficiency (E). A more positive SHAP value is correlated with a higher impact of a datapoint being classified as cortex while a more negative value is correlated with a higher impact of a datapoint not being classified as cortex.