**WiDS Datathon++ 2025 University Challenge***Age Prediction from 2D Functional Brain Networks*

# 1. Introduction

This project focuses on predicting age using 2D functional brain networks (connectomes) from fMRI data, provided by the Healthy Brain Network (HBN). The goal is to build a regression model to predict age based on brain connectivity and metadata while exploring sex-specific differences in brain development.

# 2. Data Preprocessing

## 2.1 Features Used:

# Brain Connectivity Features: The dataset included connectivity matrices representing pairwise correlations between 200 brain regions. Each matrix was flattened into a long-format row, resulting in 19,900 unique features (upper triangle of the matrix, excluding the diagonal).

# Metadata Features: Demographic and clinical metadata such as age, sex, BMI, ethnicity, race, handedness, and study site were included. Additional features included psychological factors (e.g., internalizing, externalizing, and attention scores).

## 2.2 Handling Missing Values:

# Numerical Features: Missing values in numerical features (e.g. psychological scores) were imputed using both the mean and KNN imputation. KNN imputation was used for more complex cases where the mean might not capture the underlying patterns.

# Categorical Features: Missing values in categorical features (e.g., ethnicity, race) were filled with 'Unknown'

## 2.3 Handling Outliers:

# The values that were over the third standard deviation were eliminated, removing the outliers and remaining with 95% of the total data.

## 2.4 Feature Exclusion:

# BMI Exclusion: Individuals without a BMI value were excluded from the training set to ensure data quality, due to the high importance of the feature.

# Parent Education Exclusion: The parent education features (parent\_1\_education and parent\_2\_education) were excluded from the model due to their high percentage of missing values and lack of relevance to the prediction task.

## 2.5 Feature Encoding:

# Categorical variables (e.g., sex, ethnicity, race) were one-hot encoded to make them suitable for regression models.

# 3. Tested Algorithms and Results

| Model | Kaggle RMSE | Notes |
| --- | --- | --- |
| Elastic Net (Best Model) | 1.81073 | Best-performing model with hyperparameter tuning. |
| Bayesian Regression | 1.83686 | Performed well but slightly worse than Elastic Net |
| LASSO (k=11) | 1.83884 | Good performance with k=11 splits. |
| LASSO (k=5) | 1.87295 | Slightly worse than LASSO with k=11. |
| XGBoost | 1.95789 | Moderate performance, worse than Elastic Net. |
| Random Forest | 2.25605 | Poor performance compared to Elastic Net. |
| SVR (Support Vector Regression) | 2.33567 | Poor performance compared to Elastic Net. |
| Linear Regression | 2.34491 | Baseline model with moderate performance. |
| Stepwise Regression | 2.41402 | Poor performance, similar to Linear Regression. |
| DECISIONAL TREE | 3.22916 | Worst-performing model, likely due to overfitting. |

# 

# 4. Hyperparameter Tuning

# To optimize model performance, several hyperparameter tuning techniques were employed:

# GridSearchCV: This method was used to exhaustively search over a specified parameter grid. It performed the best among the tuning methods, particularly for the Elastic Net model.

# RandomizedSearchCV: This method was used to sample a fixed number of parameter combinations from a specified distribution. It was useful for exploring a wide range of hyperparameters efficiently.

# HalvingGridSearchCV: This newer method was tested, which uses successive halving to eliminate poor-performing parameter combinations early in the search process. While promising, it did not outperform GridSearchCV in this project.

# 5. Elastic Net Regression

# The Elastic Net model was the best-performing model, achieving a Kaggle RMSE of 1.81073. Below is a detailed explanation of the model and its implementation:

## 5.1 Model Overview

# The Elastic Net model combines L1 (Lasso) and L2 (Ridge) regularization to prevent overfitting and handle multicollinearity in the data. It was implemented using the ElasticNet class from scikit-learn.

## 5.2 Hyperparameter Tuning

# Alpha: Controls the overall strength of regularization. Values were tested in a logarithmic range from 0.001 to 10.

# L1 Ratio: Determines the mix between L1 and L2 regularization. Values ranged from 0.1 (mostly L2) to 1 (pure Lasso).

# Cross-Validation: A range from 5 to 15 fold cross-validation was used to ensure the model generalizes well to unseen data, best performance was on 11 folds.

## 5.3 Results

# The best hyperparameters were found using GridSearchCV, which was optimized for the lowest root\_mean\_absolute\_error.

# The model achieved an RMSE of 1.81073 on the Kaggle test set, making it the best-performing model in this project.

# 6. Conclusion

# The Elastic Net model outperformed all other models, achieving the lowest RMSE on the Kaggle test set. The success of this model can be attributed to its ability to handle multicollinearity and its robust hyperparameter tuning process. Future work could explore combining multiple models or incorporating additional data sources (e.g., structural brain networks) to further improve performance.