Assignment 2

Statistical Learning

Felix Wente - 3337731

December 14, 2022

Question 1

The first model used is a Generalized Additive Model. One reason to use Generalized Additive Models (GAMs) for this type of data and prediction problem is that they can handle complex, non-linear patterns in the data. In a dataset with a large number of predictors, it is common for there to be non-linear relationships between the predictors and the outcome, and GAMs are well-suited to modeling these types of relationships. GAMs work by fitting a smooth function to each predictor which captures the non-linear relationship between that predictor and the outcome variable. These smooth functions are estimated using a statistical technique called maximum likelihood estimation which allows the model to account for any non-linearities in the data without making strong assumptions about the functional form of the relationship. One of the key advantages of GAMs is that they can be easily interpreted since the smooth functions provide a visual representation of the relationship between each predictor and the outcome. This can help you identify important predictors and understand how they are impacting the outcome. Additionally GAMs are flexible and can accommodate a wide variety of data types, including both continuous and categorical predictors. This makes them a useful tool for a wide range of prediction problems. Overall given the complexity of the dataset and the need to model non-linear relationships GAMs would be a good choice for this type of prediction problem. The second method to predict the outcome variable was boosting more specifically XGBoost. Boosting is a type of ensemble learning method in which multiple weak learners are trained to make predictions, and the predictions of each weak learner are combined to make a final prediction. This combination of weak learners is typically done in a sequential manner, where each subsequent weak learner is trained to correct the mistakes made by the previous weak learners. XGBoost is a specific implementation of boosting that uses decision trees as the weak learners. XGBoost is appropriate for this type of data and regression problem for several reasons. 1. XGBoost can handle both categorical and numerical predictors which is important in this case since the predictors include both types of variables. 2. XGBoost is able to deal with imbalanced data which may be the case in this dataset since the outcome variable has a minimum value of 0 and a maximum value of 30. 3. XGBoost has been shown to perform well on regression tasks and has been widely used in predictive modeling tasks in various fields. Overall the combination of XGBoost's ability to handle mixed data types imbalanced data and its performance on regression tasks makes it a good choice for this dataset and regression problem.
The last method used was a multilayer perceptron (MLP). An artificial neural network (ANN) is a machine learning model that is inspired by the structure and function of the human brain. It consists of multiple interconnected neurons or nodes that process and transmit information in

the form of signals or weights. A multilayer perceptron (MLP) is a specific type of ANN that has multiple layers of neurons and uses a supervised learning algorithm to adjust the weights of the connections between the neurons in order to make accurate predictions. In the context of this regression problem an MLP would be appropriate because it is capable of modeling complex nonlinear relationships between the predictors and the outcome variable. It can also handle both numerical and categorical data which is important in this dataset as it has both types of predictors. Additionally the relatively large number of observations (1152) allows for sufficient data to train and validate the MLP model.

Question 2,3,4

GAM

The model was built with the pygam library. All the predictors were included in the model. The categorical ones with a factor term (f()) and the numerical ones with a spline term(s()). One of the hyperparameters for this model is the Number of splines to use for the feature function. A spline is a piecewise continuous polynomial function that is used to approximate a set of data points. In a GAM (Generalized Additive Model) the number of splines is a parameter that determines the level of smoothness of the model. The greater the number of splines the more flexible the model will be and the better it will be able to approximate the data. However using a large number of splines can also lead to overfitting where the model fits the noise in the data rather than the underlying trend. It is therefore important to choose the right number of splines for a GAM model based on the complexity of the data and the goal of the analysis.The hyper parameters that were tried were 5, 10, 15, 20 and 30. The best one was 10. Another hyperparameter is lambda. In the context of a generalized additive model (GAM), the term ”lambda” refers to a regularisation parameter that is used to control the amount of smoothness in the model. This parameter is used to prevent overfitting by limiting the complexity of the model. In general a higher value of lambda will result in a smoother, i.e, less complex model while a lower value of lambda will result in a more complex and potentially overfitted model. The appropriate value of lambda for a given GAM model can be determined through model selection techniques such as cross-validation. Thus, gridsearch was used to determine the optimal lambda, 63.0957 (Table 1).

Table 1: GAM output

Feature Function	Lambda	Rank	EDoF	P > x	Sig. Code
f(0)	[63.0957]	3	2.7	5.50e-09	***
f(1)	[63.0957]	2	0.9	4.42e-01	
s(2)	[63.0957]	10	2.8	3.70e-01	
s(3)	[63.0957]	10	2.8	1.15e-01	
s(4)	[63.0957]	10	2.4	1.11e-16	***
s(5)	[63.0957]	10	2.2	2.42e-01	
s(6)	[63.0957]	10	2.4	3.36e-01	
s(7)	[63.0957]	10	2.6	2.87e-01	
f(8)	[63.0957]	2	0.7	3.75e-01	
f(9)	[63.0957]	2	0.8	3.75e-01	
f(10)	[63.0957]	4	1.3	1.36e-06	***
f(11)	[63.0957]	2	0.8	8.41e-01	
f(12)	[63.0957]	2	0.7	5.49e-02	.
f(13)	[63.0957]	2	0.7	7.01e-03	**
f(14)	[63.0957]	2	0.6	7.95e-02	.
s(15)	[63.0957]	10	2.1	3.25e-04	***
f(16)	[63.0957]	2	0.7	5.20e-01	
f(17)	[63.0957]	3	1.3	9.08e-05	***
f(18)	[63.0957]	2	0.7	3.07e-01	
f(19)	[63.0957]	2	0.8	1.22e-11	***
intercept		1	0.0	1.11e-16	***

From the parial dependence plots (Figure 1) with their confidence intervals, as well as the significance levels in the table, we can see which variables mattered the most for predicting the outcome variable. Especially pedigree and AO seem very important since they are continuous with a clear trend (positive and negative respectively). But also PsychTreat and bTypeDep seem to matter. The predictive measures can be seen in Table 4.

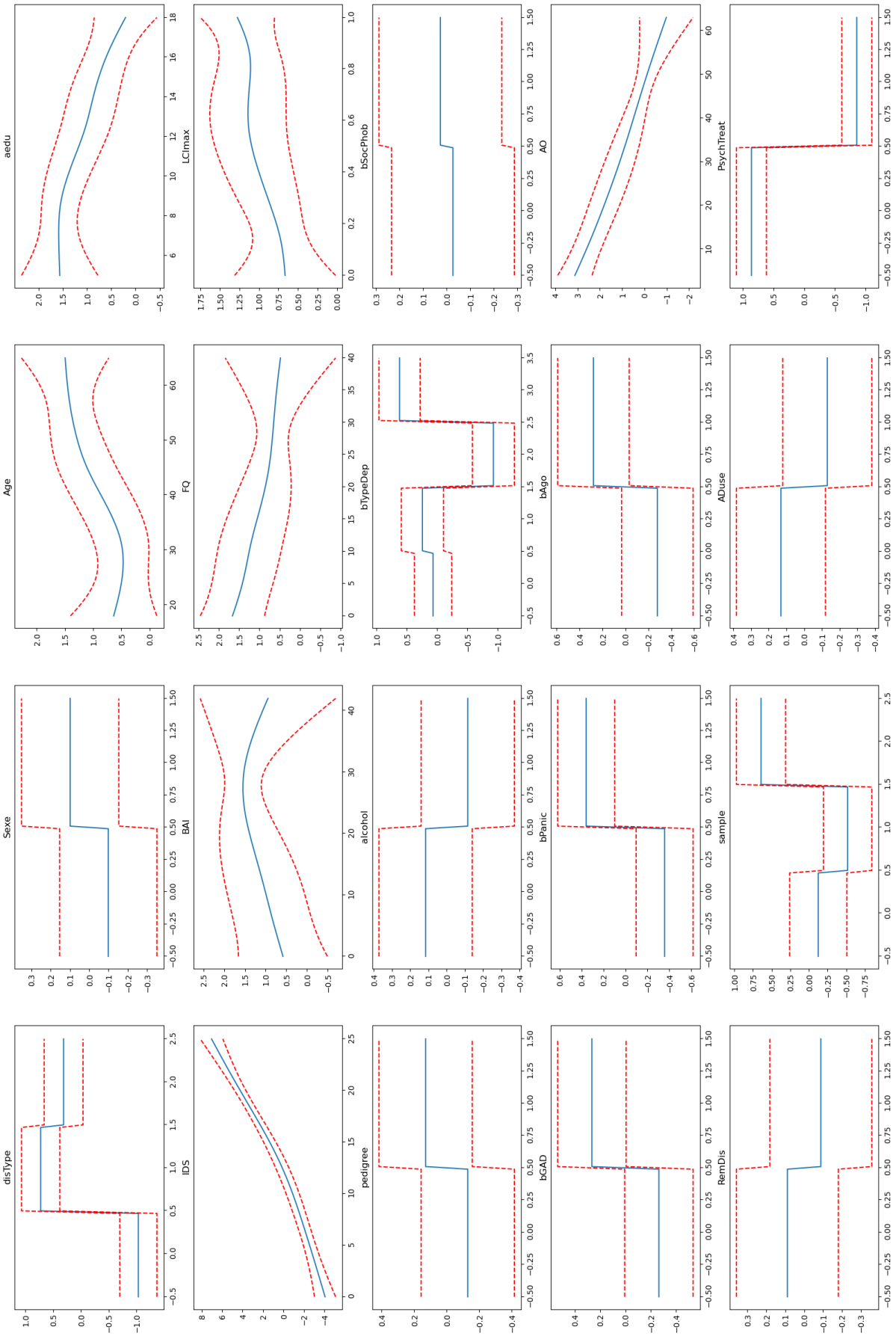

Figure 1: GAM Partial plots

XGBoost

Parameter	Value
subsample	0.7
max_depth	4
max_bin	1024
learning_rate	0.2
gamma	1
colsample_bytree	0.5

Table 2: Best parameters XGBoost

For the XGBoost model we first set up the parameter space to be explored with randomised search Cross-validation. These hyper parameters included: (1)subsample:Subsample ratio of the training instances. Setting it to 0.5 means that XGBoost would randomly sample half of the training data prior to growing trees. and this will prevent overfitting.(2) max_depth:Maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit. (3) max_bin: Maximum number of discrete bins to bucket continuous features. (4) learning_rate: Step size shrinkage used in update to prevent overfitting. After each boosting step we can directly get the weights of new features and learning_rateshrinks the feature weights to make the boosting process more conservative. (5) gamma: Minimum loss reduction required to make a further partition on a leaf node of the tree; The larger it is the more conservative the algorithm will be. (6) colsample_bytree: is the subsample ratio of columns when constructing each tree. Subsampling occurs once for every tree constructed. The chosen values by randomised search Cross-validation can be seen in Table 2 . XGBoost is more of a black box algorithm than GAM and therefore not as interpretable. However, one can see the feature importance of the individual parameters where the importance indicates the number of times feature is used to split data, as seen in the weight plot (Figure 3); or importance indicates the average gain across all splits where feature was used, as seen in the gain plot (Figure 2). The former is is likely going to represent the range variables have and their nonlinear relation ship as this would account for more splits. The latter however represents the relative contribution of the corresponding feature to the model calculated by taking each feature's contribution for each tree in the model. As can be seen, it seems to agree with the GAM that IDS seems to matter a lot, as well as the other variables that the GAM found of importance. The final predictive measures can be seen in Table 4.

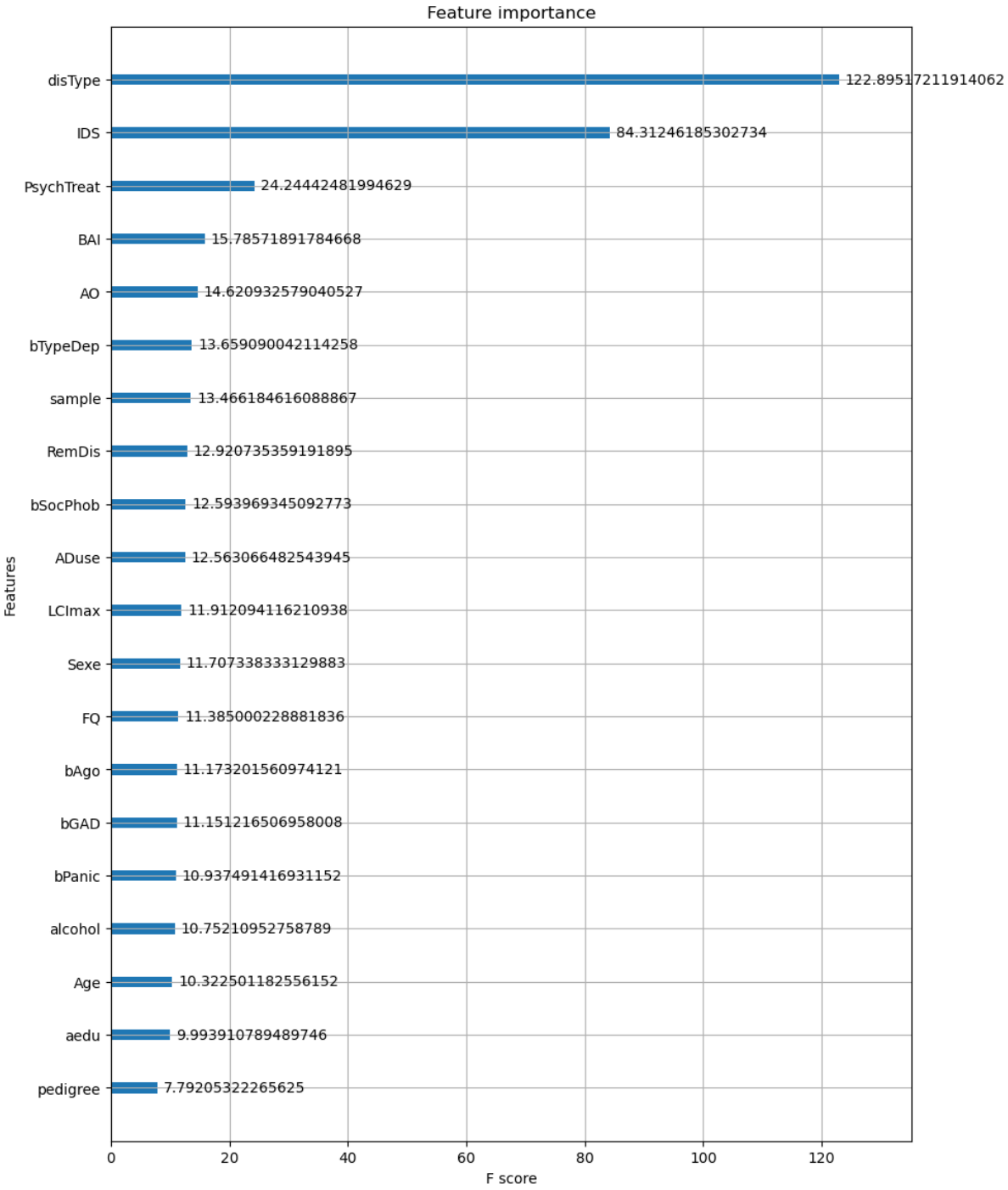

Figure 2: Feature Importance - gain

ANN (MLP)

Hyperparameter	Value	Hyperparameter	Value	Hyperparameter	Value
num_layers	4	units_1	505	activation	relu
dropout_1	False	rate_1	0.75	units_2	30
dropout_2	False	rate_2	0.25	lr	0.01
units_3	480	dropout_3	False	rate_3	0.25
units_4	380	dropout_4	True	rate_4	0.75

Table 3: Best parameters MLP

For the multilayer perceptron there are a great number of hyperparameters to control. For instance, the width and depth,i.e., the number of neurons and number of layers, comprising the main structure. But there are are a lot more such as the type of optimizers and learning rate, activation functions, dropout layers and their rates. To go through this hyperparameter space via randomized gridsearch would be very inefficient and take long.Thus we use the keras tuner and bayesian optimization. Bayesian optimization is a technique for optimizing hyperparameters of a machine learning model. It uses a Bayesian method to optimise the hyperparameters of a model, meaning that it uses past data and previous models to determine potential improvements. In Keras (keras tuner) Bayesian optimization can be used to optimise the hyperparameters of a neural network. This is done by using a search algorithm that takes into account the past results and updates the parameters accordingly. Bayesian optimization has been shown to produce better results than grid search or random search and is especially helpful when optimizing complex models with many hyperparameters. The model was created with the sequential api, using the tuner alongside it to let it determine the structure (width and depth) and the hyperparameters used in the model. The chosen values by Baysian Optimization can be seen in Table 3.Units_"number" means the number of neurons for that specific dense layer, Dropout_"number" means if a dropout layer was chosen after that ense layer, with rate_"number" denoting the dropout rate. The final predictive measures can be seen in Table 4.

Model	MAE	RMSE
GAM	3.010	3.717
XGBoost	3.258	3.888
MLP	3.149	3.888

Table 4: Predictive Performance on Test set

Question 5

The predictive ability of the three models is relatively equal. However, the GAM is slightly better in both statistics. This might be due to the fact that GAMs are flexible and can accommodate a wide variety of data types including both continuous and categorical predictors. This makes them a useful tool for a wide range of prediction problems. In contrast neural networks are typically more effective with continuous data while XGBoost is potentially more effective with very high dimensional and categorical data. Additionally, the GAM is more interpretable while both neural networks and XGBoost are less interpretable since they are based on more complex black-box models. Hence for this specific problem the GAM is best suited. It showed that Pedigree, the Presence of a first-degree relative with an anxiety and/or depressive disorder, had a significant positive relationship, while AO, Age at onset of the disorder, and PsychTreat, whether subject receives psychological treatment for the disorder(s), had a significant negative effect (step function for PsychTreat). In future research other combinations of variables (interaction terms, engineered features) could be tried. For instance, subtracting AO from age to see the years since disorder started. But for that more observations are likely needed, since the degree of freedom for the individual predictors weren't that high already (Table 1).

Question 6

The predicted outcome values , the estimate of the severity of David's depressive symptoms in 12 months time, is given in Table 5. Since all three models have a value of greater or equal to 17, the recommendation would be that David should be referred to the intensive treatment program.

Model	**predicted outcome**
GAM	17.175
XGBoost	18.681
MLP	17.768

Table 5: Predictive Performance on Test set

Appendix

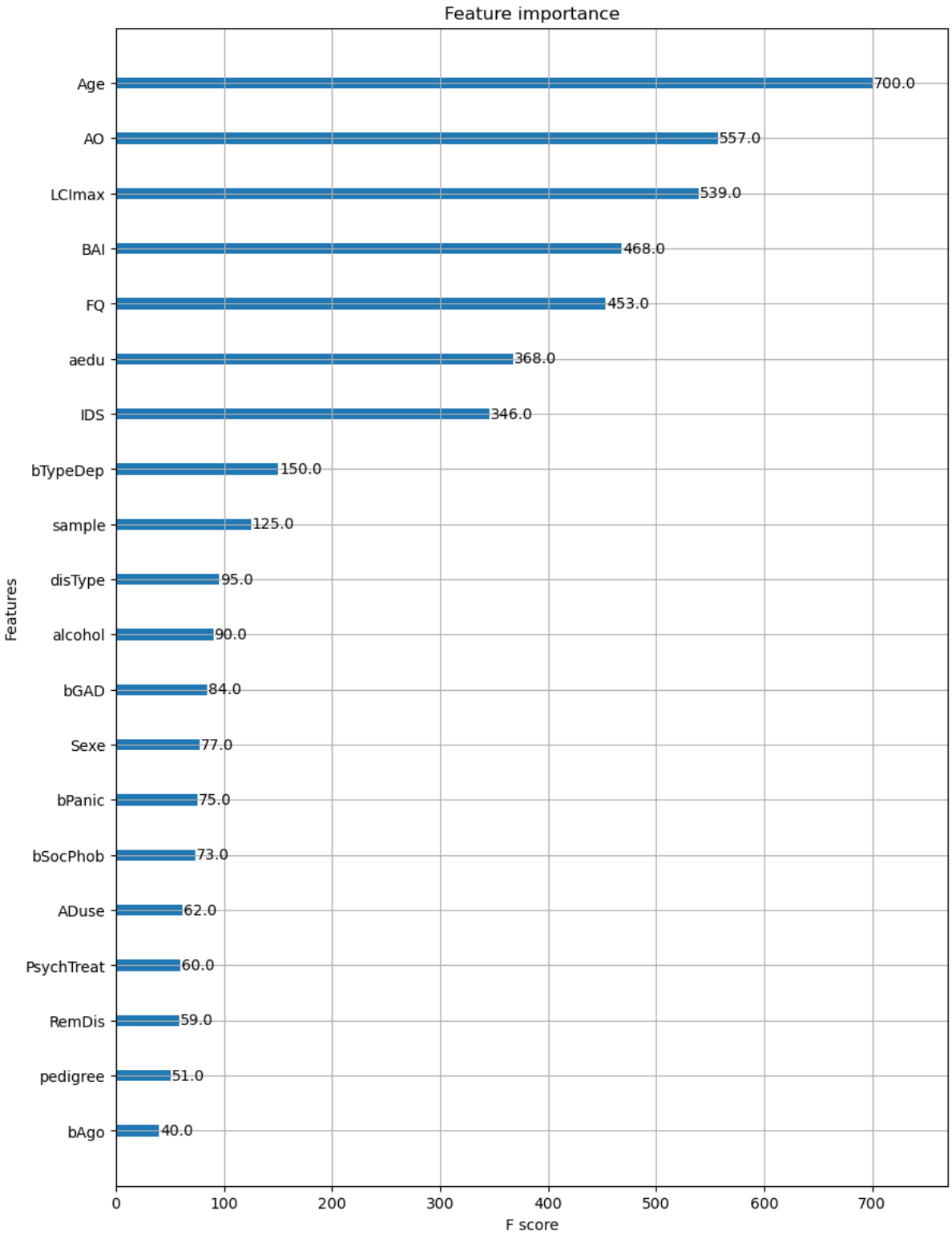

Figure 3: Feature Importance - weights