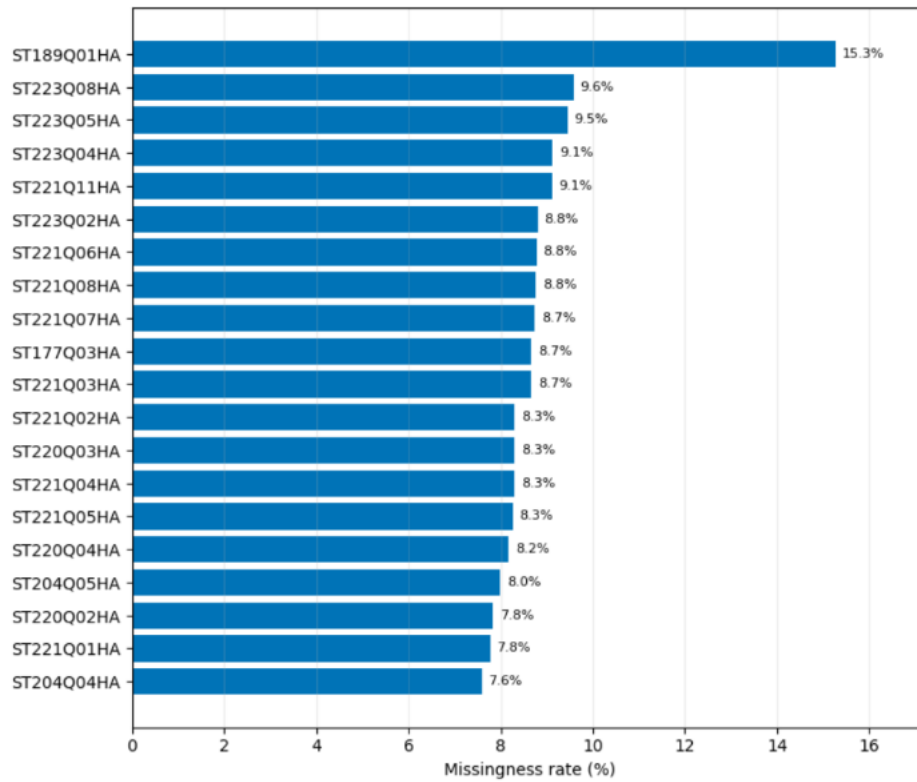
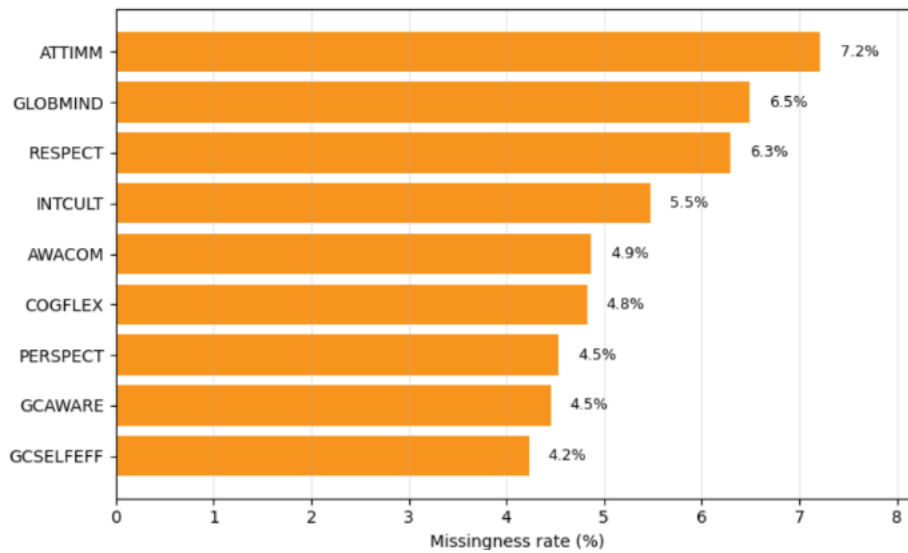


Figure S1. Missingness Rate of Predictors and Indices

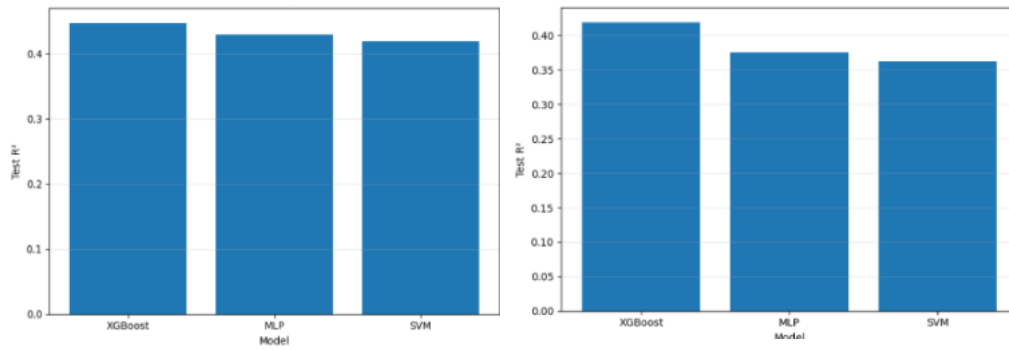


(a) Top 20 Question Variables by Missingness Rate before Imputation



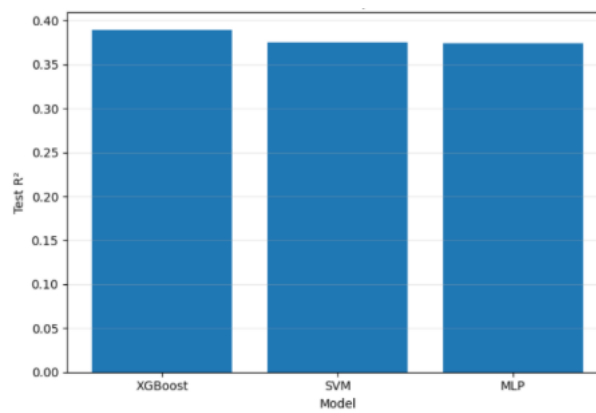
(b) Missingness Rate of Index Variables before Imputation

Figure S2. Model comparison across candidate algorithms



(a) COGFLEX Model Comparison Test R^2

(b) AWACOM Model Comparison Test R^2



(c) PERSPECT Model comparison Test R^2

COGFLEX Model

	R^2			MSE			MAE		
	XGBoost	SVM	MLP	XGBoost	SVM	MLP	XGBoost	SVM	MLP
Training	0.4953	0.3741	0.4733	0.3970	0.4923	0.4143	0.4676	0.5148	0.4780
Testing	0.4469	0.4185	0.4294	0.4411	0.4637	0.4550	0.5030	0.5126	0.5176

AWACOM Model

	R^2			MSE			MAE		
	XGBoost	SVM	MLP	XGBoost	SVM	MLP	XGBoost	SVM	MLP
Training	0.5394	0.4212	0.4099	0.4045	0.5083	0.5183	0.4437	0.4753	0.5063
Testing	0.4195	0.3629	0.3753	0.5460	0.5993	0.5876	0.4946	0.5201	0.5254

PERSPECT Model

	R^2			MSE			MAE		
	XGBoost	SVM	MLP	XGBoost	SVM	MLP	XGBoost	SVM	MLP
Training	0.5110	0.4018	0.4534	0.4247	0.5195	0.4747	0.4860	0.5364	0.5177
Testing	0.3896	0.3753	0.3741	0.5925	0.6064	0.6076	0.5672	0.5750	0.5772

Table S1. Cross-validated mean and standard deviations for R², MSE, and MAE

COGFLEX				
Metric	Train mean	Train stddev	Test mean	Test stddev
R ²	0.5003	0.0036	0.4029	0.0418
MSE	0.3936	0.0034	0.4702	0.0403
MAE	0.4671	0.0018	0.5060	0.0186
AWACOM				
Metric	Train mean	Train stddev	Test mean	Test stddev
R ²	0.5411	0.0038	0.3900	0.0503
MSE	0.4059	0.0058	0.5402	0.0686
MAE	0.4427	0.0028	0.5055	0.0245
PERSPECT				
Metric	Train mean	Train stddev	Test mean	Test stddev
R ²	0.5084	0.0034	0.4141	0.0302
MSE	0.4319	0.0035	0.5136	0.0399
MAE	0.4896	0.0020	0.5295	0.0225

Table S2. Hyperparameters search field for the XGBoost grid search

Hyperparameters	Range/set of values considered
<u>Number of boosting trees</u>	<u>100, 300, 500</u>
<u>Maximum depth of each tree</u>	<u>2, 3, 4, 5</u>
<u>Learning reate</u>	<u>0.01, 0.03, 0.05, 0.1</u>
<u>Fraction of observations sampled for each tree</u>	<u>0.7, 0.9, 1.0</u>
<u>Fraction of features sampled for each tree</u>	<u>0.7, 0.9, 1.0</u>
<u>L1 regularization term (alpha)</u>	<u>0, 0.01, 0.1</u>
<u>L2 regularization term (lambda)</u>	<u>1, 5, 10</u>

Table S3. Software and Reproducibility

Software/Package	Version	Use in the Study
Python	3.12.13	Main programming environment
NumPy	2.4.4	Numerical operations and array processing
pandas	3.0.2	Data cleaning, preparation, and tabulation
scikit-learn	1.8.0	Train-test split, cross-validation, preprocessing pipeline, KNN imputation, MLP, SVM, and performance metrics
XGBoost	3.2.0	Final gradient-boosted tree regression models
SHAP	0.51.0	Model interpretation and predictor importance analysis
Matplotlib	3.10.8	Visualization of model performance, missingness, and SHAP results

All analyses were conducted in Python. Data processing and numerical operations were performed using pandas and NumPy, while machine-learning model development, preprocessing, train-test splitting, cross-validation, and performance evaluation were implemented using scikit-learn. XGBoost was used for the final gradient-boosted tree models, SHAP was used for model interpretation, and Matplotlib was used for visualization.

To improve reproducibility, fixed random seeds were used for train-test splitting, cross-validation, and model initialization where applicable. The final modeling workflow used a machine-learning pipeline in which standardization, KNN imputation, and model fitting were implemented jointly within the cross-validation procedure. This ensured that preprocessing parameters were estimated only from the training portion of each fold.

The software versions used in the analysis were recorded from the computational environment and are reported below. The analysis code can be made available upon reasonable request, subject to data-use and reproducibility requirements.