Method Used	Dataset Size	Testing-set predictive performance	Time taken for the model to be fit
XGBoost in Python via scikit-learn and 5-fold CV	100		
	1000		
	10000		
	100000		
	1000000		
	10000000		
XGBoost in R – direct use of xgboost() with simple cross-validation	100	0.75	0.014s
	1000	0.97	0.0288s
	10000	0.9695	0.2117s
	100000	0.9854	1.4050s
	1000000	0.9885	12.9564s
	10000000	0.989	56.2554s
XGBoost in R – via caret, with 5-fold CV simple cross-validation	100	0.918	2.103498s
	1000	0.971	4.314s
	10000	0.984	22.635s
	100000	0.987	102.628s
	1000000	0.990	506.259s
	10000000	0.991	3056.269s

The implementation of xgboost() with basic cross-validation requires less time compared to XGBoost execution through caret with 5-fold cross-validation. The direct implementation of XGBoost completes a 10 million observation run in 56 seconds but caret requires more than 50 minutes (3056 seconds). The predictive performance of caret remains higher than other implementations in every dataset size scenario with performance increases from 0.001 to 0.168 based on dataset dimensions.

The xgboost() function should be used directly with simple cross-validation for most practical applications. The performance advantage from using caret amounts to 0.2% improvement at maximum dataset size while requiring 54x more computational time. The recommendation holds maximum value for big datasets and production environments that need frequent model retraining. Xgboost() delivers outstanding results in computational speed 0.989 at 10M observations with predictable performance thus enabling quicker model testing and modifications for real-world operations.