| 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 | 0.95 | 0.96 |
| | 1000 | 0.97 | 0.65 |
| | 10000 | 0.9820 | 2.03 |
| | 100000 | 0.9866 | 4.06 |
| | 1000000 | 0.9923 | 34.95 |
| | 10000000 | 0.9932 | 339.80 |
| XGBoost in R – direct use of xgboost() with simple cross-validation | 100 | 0.95 | 0.123290 |
| | 1000 | 0.94 | 0.2767 |
| | 10000 | 0.9675 | 0.654 |
| | 100000 | 0.98360 | 2.4151 |
| | 1000000 | 0.98857 | 52.66 |
| | 10000000 | | |
| XGBoost in R – via caret, with 5-fold CV simple cross-validation | 100 | 0.90 | 42.61410 |
| | 1000 | 0.96 | 52.8319 |
| | 10000 | 0.9820 | 87.57598 |
| | 100000 | 0.9905 | 542.53178 |

| Method used | Dataset size | Testing-set predictive performance | Time taken for the model to be fit |
|---|---|---|---|
| | 1000000 | | |
| | 10000000 | | |

XGBoost implementation in R delivers optimal predictive results combined with efficient computation performance in all dataset scale tests. The direct XGBoost implementation in R completed in 52.66 seconds for 1 million observations while achieving 0.98857 accuracy yet the Python scikit-learn implementation needed 339.80 seconds to fit the models yet reached 0.9932 accuracy for 10 million observations. A key advantage of the R direct implementation becomes crucial for massive applications because it accelerates execution by 6.5x without degrading predictive accuracy by more than 0.4% at 1-million observation sizes.

R users should stay away from using caret for big datasets because its poor performance with growing data sets has been proven through testing. The caret implementation in R needed 542.53 seconds to process 100,000 observations and provided negligible accuracy advantages compared to direct R implementation. The system showed unable to finish operations on the largest dataset which demonstrates major constraints for operational deployment. Python provides better scalability than R when using caret although the results fall short of what direct XGBoost running in R can achieve. Real-world applications that need both high performance and efficiency should use direct XGBoost implementation in R because it provides the best solution when working with datasets larger than 100,000 observations.