1) Compare the accuracy values of XGBoost models fit on the newly created data

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	0.87	0.18
	1000	0.949	0.31
	10000	0.9746	0.78
	100000	0.9871	2.74
	1000000	0.9917	24.65
	10000000	0.9931	253.12
XGBoost in R – direct use of xgboost() with simple cross-validation	100	1	0.0123
	1000	1	0.0222
	10000	0.98890	0.0986
	100000	0.98110	1.0984
	1000000	0.9794	9.099
	10000000	0.96523	103.26
XGBoost in R – via caret, with 5-fold CV simple cross-validation	100	1	2.10
	1000	1	4.64
	10000	0.99540	25.81
	100000	0.99477	241.81
	1000000	0.99052	964.23
	10000000	0.98456	2456.23

2. **Based on the results, which approach to leveraging XGBoost would you recommend?**

Explain the rationale for your recommendation.

Based on the results, I recommend using XGBoost in R – direct use of xgboost() with simple cross-validation.

The comparison shows that this method is significantly faster than both XGBoost via scikit-learn and XGBoost via caret. For example, when training on 10 million records, xgboost() in R took around 103 seconds, while Python's scikit-learn needed 253 seconds, and caret took a much longer

2456 seconds. As dataset size increased, caret's training time became impractical for real-world use.

In terms of accuracy, all three methods delivered excellent results. Predictive performance across all dataset sizes was very high (generally between 96% and 99%), and differences between methods were minor. However, because the gain in accuracy was minimal while caret's training time was extremely high, it is not an efficient choice for large datasets.

Python’s XGBoost implementation also performed well and would be a strong choice if the project environment is already based in Python. But from a pure efficiency perspective, direct R XGBoost is the best: it provides nearly the same (or slightly better) predictive performance with much faster execution.

Thus, for practical machine learning on large datasets, direct use of xgboost() in R offers the best combination of speed, efficiency, and accuracy, and is the method I would recommend.