Comparison of R/Python for XGBoost

Q1) Compare the accuracy values of XGBoost models fit on the newly created data, for the following sizes of datasets. Along with accuracy, report the time taken for computing the results. Report your results in a table with the following schema.

XGBoost in R – direct use of xgboost() with simple cross-validation		
Sample size	Accuracy	Time taken
100	0.95	0.058
1000	0.925	0.0855
10000	0.971	0.303622
100000	0.983	2.521756
1000000	0.988	32.2056
10000000	0.9929	129.8
XGBoost in R – via caret, with 5-fold CV simple cross-validation		
Sample size	Accuracy	Time taken
100	0.9404261	6.088
1000	0.95	9.944
10000	0.98	42.68
100000	0.98	373.66
1000000	0.98	2806
10000000	0.99	50856
XGBoost in Python via scikit-learn and 5-fold CV		
Sample size	Accuracy	time taken
100	0.91	0.1591
1000	0.95	0.2962
10000	0.97	0.4361
100000	0.98	1.0521
1000000	0.99	8.7808
10000000	0.99	109.88

The tables show the performance of XGBoost implemented in different ways across varying sample sizes. The direct R implementation (using xgboost() with simple CV) demonstrates excellent scalability, maintaining high accuracy (0.95-0.99) while being the fastest option, particularly for large datasets (32.2 seconds for 1M rows). The caret implementation in R with 5-fold CV achieves comparable accuracy but is significantly slower (373.66 seconds for 100K rows vs 2.52 seconds for direct implementation). Python's scikit-learn with 5-fold CV strikes a balance between accuracy (0.91-0.99) and speed, being faster than R's caret but slower than R's direct implementation.

Q2) Based on the results, which approach to leveraging XGBoost would you recommend? Explain the rationale for your recommendation.

I would recommend the direct R implementation using xgboost() for most use cases. The rationale is that it provides comparable accuracy to other methods (often within 0.01-0.02) while being dramatically faster - up to 100x faster than R's caret for larger datasets. The Python implementation is a good alternative if working in a Python ecosystem, as it's faster than R's caret while maintaining similar accuracy. The caret implementation should only be used if its additional features (like automated parameter tuning) are necessary, as its computational overhead is substantial. For large-scale applications where performance matters, the direct R implementation offers the best balance of speed and accuracy.