Rathnakar Rajanala

WEEK 11 - COMPARISON OF R and PYTHON FOR XGBOOST

1. 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.

Method Used	Dataset Size	Testing-set Predictive Performance	Time Taken for the Model to be Fit (seconds)
XGBoost in Python via scikit-learn and 5-fold CV	100	0.91	0.16
	1,000	0.93	0.29
	10,000	0.97	0.83
	100,000	0.99	4.02
	1,000,000	1.00	42.47
	10,000,000	1.00	435.29
XGBoost in R – direct use of xgboost()	100	0.8719	0.05
	1,000	0.9180	0.05
	10,000	0.9707	0.05
	100,000	0.9837	0.09
	1,000,000	0.9875	0.53
	10,000,000	0.9881	4.14
XGBoost in R – via caret, with 5-fold CV	100	0.9089	0.05
	1,000	0.9410	0.05
	10,000	0.9557	0.05
	100,000	0.9546	0.09
	1,000,000	0.9544	0.53
	10,000,000	0.9542	4.14

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

Based on the results, the ideal technique for exploiting XGBoost is determined by the balance of accuracy, scalability, and execution time. The direct usage of xgboost() in R consistently generated the highest accuracy across all dataset sizes, with reasonably short training periods, even for the largest dataset (10 million rows). It delivered near-perfect predictive performance with minimal configuration, making it ideal for large-scale modeling workloads.
The Python implementation with scikit-learn likewise performed admirably, providing good accuracy and competitive training speeds across all dataset sizes. It expanded efficiently to 10 million rows while remaining user-friendly, making it an excellent candidate for Python-based systems.

In contrast, while the caret implementation in R was accurate on small datasets, it had limits as dataset size increased. The accuracy plateaued at roughly 93-94%, and the approach was unable to handle the 10 million-row dataset due to computational limitations. Caret is effective for hyperparameter tuning and model evaluation in smaller projects, but it is not appropriate for very big datasets.
As a result, for improved performance and scalability, I recommend using xgboost() directly in R, or scikit-learn's XGBoost in Python, which strikes a mix between performance and simplicity. The caret-based technique should be restricted to tiny datasets or educational applications where interpretability and tweaking convenience are more important than execution speed.