Week 11 Assignment: XGBoost Model Comparison

Comparison of XGBoost Methods

Method	Dataset Size	Accuracy	Time (s)
Python + sklearn (5-fold CV)	100	0.89	1.39
Python + sklearn (5-fold CV)	1000	0.943	0.28
Python + sklearn (5-fold CV)	10000	0.9722	0.83
Python + sklearn (5-fold CV)	100000	0.9874	4.03
Python + sklearn (5-fold CV)	1000000	0.9919	41.4
Python + sklearn (5-fold CV)	10000000	0.9932	430.01
Direct XGBoost	100	1.0	0.24
Direct XGBoost	1000	1.0	0.4
Direct XGBoost	10000	1.0	0.8
Direct XGBoost	100000	0.9996	2.58
Direct XGBoost	1000000	0.996	22.1
Direct XGBoost	10000000	0.9942	190.74
Caret XGBoost (5-fold CV)	100	0.9109	1.94
Caret XGBoost (5-fold CV)	1000	0.953	2.32
Caret XGBoost (5-fold CV)	10000	0.9727	4.25
Caret XGBoost (5-fold CV)	100000	0.9865	15.57
Caret XGBoost (5-fold CV)	1000000	0.9922	120.6
Caret XGBoost (5-fold CV)	10000000	0.9935	1206.0

Detailed Explanation and Recommendation

Objective:

The goal was to evaluate the performance , accuracy and training time of XGBoost models using:

1. Python using scikit-learn

2. R using direct xgboost()

3. R using caret with 5-fold cross-validation

Python + scikit-learn:

The approach is known for very fast execution due to multi-threading capabilities. It consistently delivers high accuracy, which improves as the dataset size increases. Additionally, it integrates seamlessly with pipelines and grid search tools, making it ideal for production environments. However, one drawback is that hyperparameter tuning must be done manually, unlike R's caret package.

R Direct xgboost():

The xgboost() function in R produced the highest accuracy across all dataset sizes and performed efficiently on small to medium datasets. It is lightweight and fast. However, it does not have built-in support for cross-validation, and the accuracy is evaluated on the training data itself, which may lead to overfitting or an overly optimistic performance estimate. This approach is best suited for quick experimentation or when using fixed hyperparameters.

R Caret + XGBoost (5-fold CV):

The method is user-friendly and allows for easy integration of cross-validation and hyperparameter tuning. The accuracy results from this method are more realistic and unbiased. However, as the dataset size grows, caret becomes significantly slower and more memory-intensive. It is most effective for small to medium datasets where tuning is crucial.

Final Recommendation:

The results indicate that Python with XGBClassifier from scikit-learn should be used to manage large datasets. The model provides both high accuracy and fast processing speed and supports cross-validation. The tool is suitable for production environments because it works with cloud platforms and has broad library integration capabilities. The R environment should use caret for model tuning and xgboost for quick small-scale testing. Python with sklearn represents the most practical and robust solution for scalable real-world applications.