1.

Data size	Configuration	Training error	Validation error	Time of execution
1000	1 hidden layer 4 nodes	0.2400	0.2800	0.07
10000	1 hidden layer 4 nodes	0.0111	0.0120	1.80
100000	1 hidden layer 4 nodes	0.0008	0.0012	12.55
1000	2 hidden layers of 4 nodes each	0.2212	0.2550	0.05
10000	2 hidden layers of 4 nodes each	0.2389	0.2490	0.22
100000	2 hidden layers of 4 nodes each	0.0008	0.0010	5.38

2.

The model with one hidden layer containing four nodes demonstrates superior performance according to the results. The model configuration with one hidden layer of four nodes maintains lower training and validation errors than the model with two hidden layers across all data point sizes but especially for 10,000 and 100,000 data points. The model with 1 hidden layer of 4 nodes reaches a validation error of 0.0012 at 100,000 data points while using half the execution time compared to the 2-layer model (12.55 seconds vs. 5.38 seconds). The 1-layer model shows better stability when processing different data sizes which indicates better generalization capabilities and operational efficiency. The 1-layer model delivers optimal accuracy-performance ratio because of its robust computational cost-effectiveness.

3.

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	1000	0.9470	0.0868
	10000	0.9776	0.2227
	100000	0.9871	0.7634

3.

XGBoost demonstrates superior predictive performance than deep learning networks across the entire range of analyzed dataset sizes. The testing accuracy of XGBoost reaches 0.9871 when analyzing 100,000 data points which exceeds the validation accuracy (1 - error) of approximately 0.9988 that the best deep learning model could achieve. The training process for XGBoost takes 0.76 seconds while the best neural network requires 12.55 seconds to finish. XGBoost demonstrates outstanding prediction accuracy alongside short training times when dealing with small datasets which makes it a high-performance and efficient solution. The results demonstrate XGBoost provides the most effective and efficient solution when applied to this particular task with these dataset dimensions.