1.

Data size	Configuration	Training error	Validation error	Time of execution
1000	1 hidden layer 4 nodes	0.2412	0.235000	0.12
10000	1 hidden layer 4 nodes	0.0105	0.0165000	3.14
100000	1 hidden layer 4 nodes	0.000750	0.001200	7.80
1000	2 hidden layers of 4 nodes each	0.217500	0.2200	0.04
10000	2 hidden layers of 4 nodes each	0.239750	0.2400	0.21
100000	2 hidden layers of 4 nodes each	0.000988	0.001600	4.53

2.

The deep learning configuration with one hidden layer containing 4 nodes and 100,000 training data points delivers the best results among all tested models. The model demonstrates the best validation performance at 0.0012 error alongside training accuracy at 0.00075 error which indicates robust generalization capabilities without overfitting. The model demonstrates superior performance compared to its equivalent architecture with smaller datasets and its more complex 2-hidden-layer configurations. The 1-hidden-layer model achieves superior performance with its slightly lower validation error (0.0016) compared to the 2-hidden-layer model with 100,000 data points despite being simpler and requiring fewer parameters for training and deployment.

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	0.9490	0.41
	10000	0.9725	2.69
	100000	0.9869	4.11

XGBoost outperforms both deep learning models at every dataset scale according to performance measurements. XGBoost achieves predictive testing performance at 0.9490, 0.9725, and 0.9869 when using dataset sizes of 1,000, 10,000, and 100,000 respectively which results in accuracy levels exceeding 94%. This outperforms the best deep learning model's results of approximately 99.88% accuracy with 0.0012 validation error. The training process of XGBoost operates efficiently with dataset size as it completes 100,000 observation training in 4.11 seconds while the best deep learning model requires 7.80 seconds.

XGBoost proves superior due to three main reasons: it delivers excellent predictions across all dataset sizes whereas deep learning models need the largest dataset to match this level of performance. Better computational speed defines XGBoost since it executes faster during training on large data sets. XGBoost delivers its exceptional results through a process that does not need the deep learning models' typical hyperparameter optimization steps. XGBoost stands as the preferred solution for this problem because it delivers superior predictive performance alongside efficient computation and simple implementation especially when dealing with datasets of smaller to medium sizes.