Hi class,

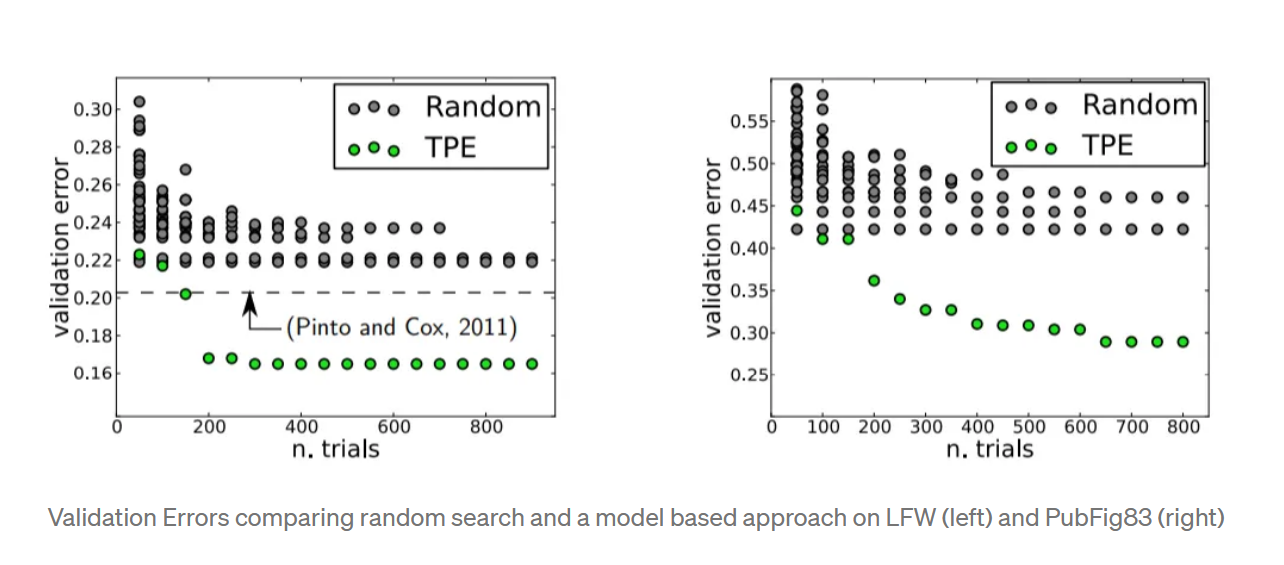
The premise of hyperparameter tuning is to aid in the learning process of an ML/DL model. Grid search, random search, and Bayesian optimization as noted, are three popular optimization methods for hyperparameter tuning.

First at a general, high-level, grid search is a technique used to systematically search for a predefined hyperparameter space to find the optimal hyperparameters for the ML model. This is achieved by specifying a set of values for various hyperparameters, then a brute force like method is implemented where the algorithm trains and evaluates the model on all possible combinations of the hyperparameter values in the search space ‌(Joseph, 2018).

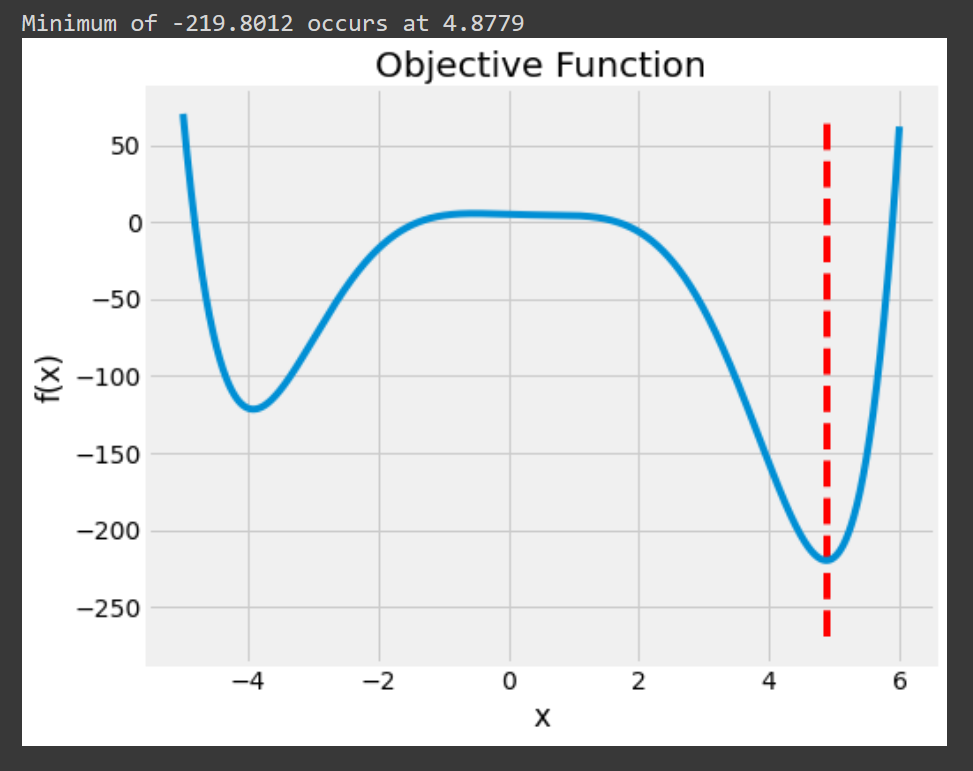
Next, random search in connection with ML hyperparameter tuning works very similarly to grid search, but rather than iterating through the set of possible hyperparameters and evaluating each one, random search only evaluates a certain number of hyperparameter combinations that are selected randomly.

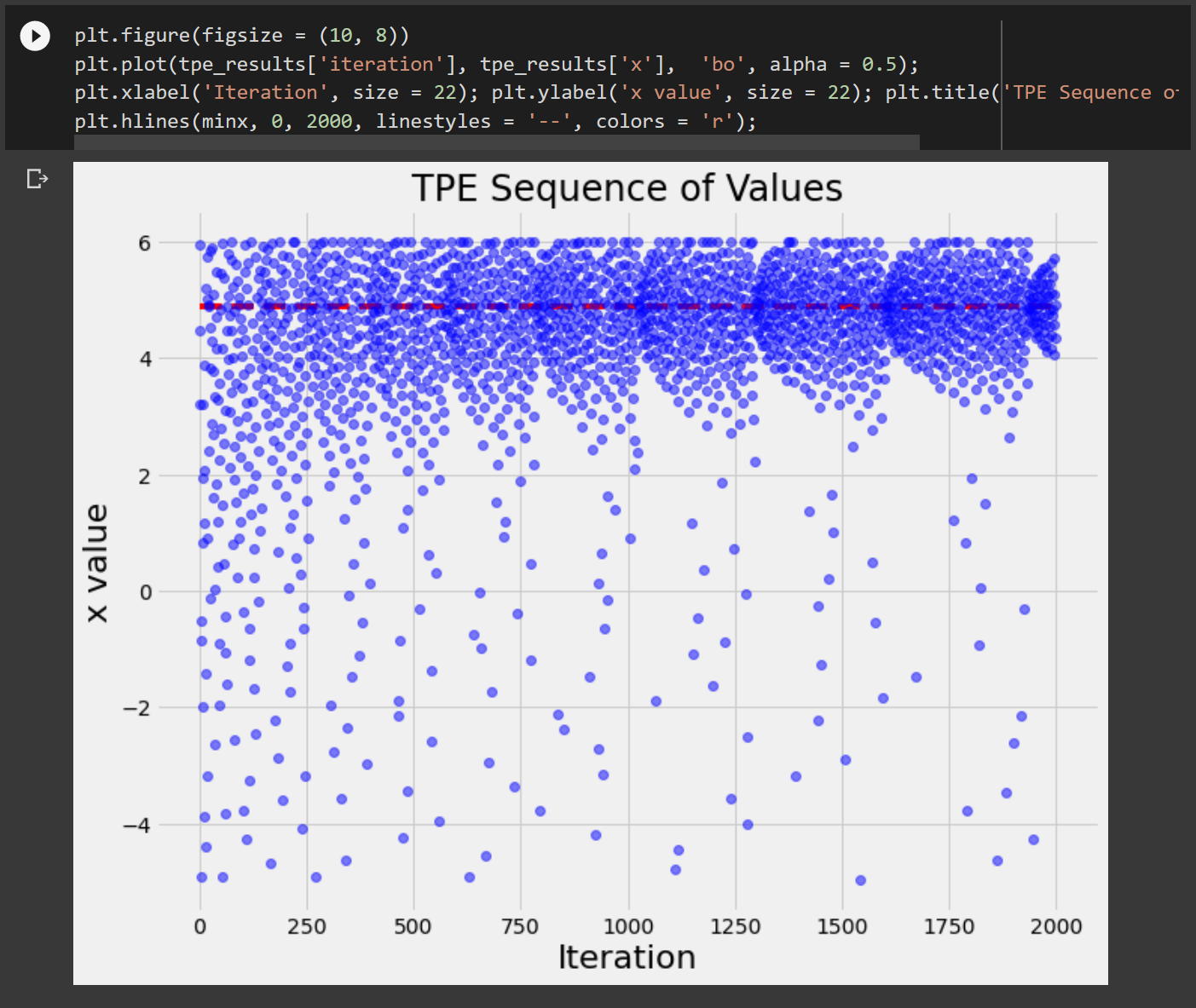
Lastly, Bayesian optimization is premised on a probabilistic model that uses sequential optimization techniques to iteratively select the next iteration or set of hyperparameters based on the previous results from the last iteration and evaluation. Typically, an objective function such as validation accuracy is used to evaluate the model. While there is some variance depending on the type of ML model and neural net that is being applied to solve a given problem, typically, Bayesian optimization takes the cake when it comes to popularity and efficiency. Here is two graphs comparing the validation error for hyperparameter optimzation for an image classification NN -- random search is in gray, and Bayesian optimization using the Tree Parzen Estimator is in green. For context, the lower the number for validation error, the better.

The winner in this comparison is fairly obvious. Bayesian optimization in this image classification NN resulted in smaller validation errors with a smaller amount of trials resulting in less time consumed in the hyperparameter tuning process ‌(Koehrsen, 2018).



Looking at the actual implementation in code with Bayesian Optimization, Hyperopt is a Python library that implements either the Tree-Structured Parzen Estimator or the Random algorithm. After the importation of hyperopt, the history needs to be stored using a Trials object, then the optimization processes can be ran according to the four main parts to the optimization process in general: 10 objective function (what we are trying to minimize); 2) domain space; 3) hyperparameter optimization function; and 4) trials (score, parameter pairs recorded each time the objective function is evaluated). Then matplotlib can be used to visualize various useful data. Here is what it might look like in code and matplotlib.





**References**

Joseph, R. (2018, December 29). Grid Search for model tuning. Medium.<https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e>

Koehrsen, W. (2018, June 24). A Conceptual Explanation of Bayesian Hyperparameter Optimization for Machine Learning. Medium; Towards Data Science. https://towardsdatascience.com/a-conceptual-explanation-of-bayesian-model-based-hyperparameter-optimization-for-machine-learning-b8172278050f