Hyperparameter Tuning techniques

Models can have many hyperparameters and finding the best combination of parameters can be treated as a search problem. The two best strategies for Hyperparameter tuning are:

1. GridSearchCV
2. RandomizedSearchCV
3. Bayesian Optimization

1. GridSearchCV

Grid search can be considered as a “brute force” approach to hyperparameter optimization. We fit the model using all possible combinations after creating a grid of potential discrete hyperparameter values. We log each set’s model performance and then choose the combination that produces the best results. This approach is called GridSearchCV, because it searches for the best set of hyperparameters from a grid of hyperparameters values.

An exhaustive approach that can identify the ideal hyperparameter combination is grid search. But the slowness is a disadvantage. It often takes a lot of processing power and time to fit the model with every potential combination, which might not be available.

For example: if we want to set two hyperparameters C and Alpha of the Logistic Regression Classifier model, with different sets of values. The grid search technique will construct many versions of the model with all possible combinations of hyperparameters and will return the best one.

As in the image, for C = [0.1, 0.2, 0.3, 0.4, 0.5] and Alpha = [0.1, 0.2, 0.3, 0.4]. For a combination of *C=0.3 and Alpha=0.2*, the performance score comes out to be 0.726(Highest), therefore it is selected.

Drawback: GridSearchCV will go through all the intermediate combinations of hyperparameters which makes grid search computationally very expensive.

2. RandomizedSearchCV

As the name suggests, the random search method selects values at random as opposed to the grid search method's use of a predetermined set of numbers. Every iteration, random search attempts a different set of hyperparameters and logs the model's performance. It returns the combination that provided the best outcome after several iterations. This approach reduces unnecessary computation.

RandomizedSearchCV solves the drawbacks of GridSearchCV, as it goes through only a fixed number of hyperparameter settings. It moves within the grid in a random fashion to find the best set of hyperparameters. The advantage is that, in most cases, a random search will produce a comparable result faster than a grid search.

Drawback: It's possible that the outcome could not be the ideal hyperparameter combination is a disadvantage

3. Bayesian Optimization

Grid search and random search are often inefficient because they evaluate many unsuitable hyperparameter combinations without considering the previous iterations' results. Bayesian optimization, on the other hand, treats the search for optimal hyperparameters as an optimization problem. It considers the previous evaluation results when selecting the next hyperparameter combination and applies a probabilistic function to choose

the combination that will likely yield the best results. This method discovers a good hyperparameter combination in relatively few iterations.

Data scientists use a probabilistic model when the objective function is unknown. The probabilistic model estimates the probability of a hyperparameter combination's objective function result based on past evaluation results.

$$P(\text{score}(y)|\text{hyperparameters}(x))$$

It is a "surrogate" of the objective function, which can be the root-mean-square error (RMSE), for example. The objective function is calculated using the training data with the hyperparameter combination, and we try to optimize it (maximize or minimize, depending on the objective function selected).

Applying the probabilistic model to the hyperparameters is computationally inexpensive compared to the objective function. Therefore, this method typically updates and improves the surrogate probability model every time the objective function runs. Better hyperparameter predictions decrease the number of objective function evaluations needed to achieve a good result. Gaussian processes, random forest regression, and tree-structured Parzen estimators (TPE) are examples of surrogate models.

The Bayesian optimization model is complex to implement, but off-the-shelf libraries like Ray Tune can simplify the process. It's worth using this type of model because it finds an adequate hyperparameter combination in relatively few iterations. However, compared to grid search or random search, we must compute Bayesian optimization sequentially, so it doesn't allow distributed processing. Therefore, Bayesian optimization takes longer yet uses fewer computational resources.

Drawback: Requires an understanding of the underlying probabilistic model.

Challenges in Hyperparameter Tuning

- **Dealing with High-Dimensional Hyperparameter Spaces: Efficient Exploration and Optimization**
- **Handling Expensive Function Evaluations: Balancing Computational Efficiency and Accuracy**
- **Incorporating Domain Knowledge: Utilizing Prior Information for Informed Tuning**
- **Developing Adaptive Hyperparameter Tuning Methods: Adjusting Parameters During Training**

Applications of Hyperparameter Tuning

- **Model Selection: Choosing the Right Model Architecture for the Task**
- **Regularization Parameter Tuning: Controlling Model Complexity for Optimal Performance**
- **Feature Preprocessing Optimization: Enhancing Data Quality and Model Performance**
- **Algorithmic Parameter Tuning: Adjusting Algorithm-Specific Parameters for Optimal Results**

Advantages of Hyperparameter tuning:

- **Improved model performance**
- **Reduced overfitting and underfitting**
- **Enhanced model generalizability**

- **Optimized resource utilization**
- **Improved model interpretability**

Disadvantages of Hyperparameter tuning:

- **Computational cost**
- **Time-consuming process**
- **Risk of overfitting**
- **No guarantee of optimal performance**
- **Requires expertise**
    -