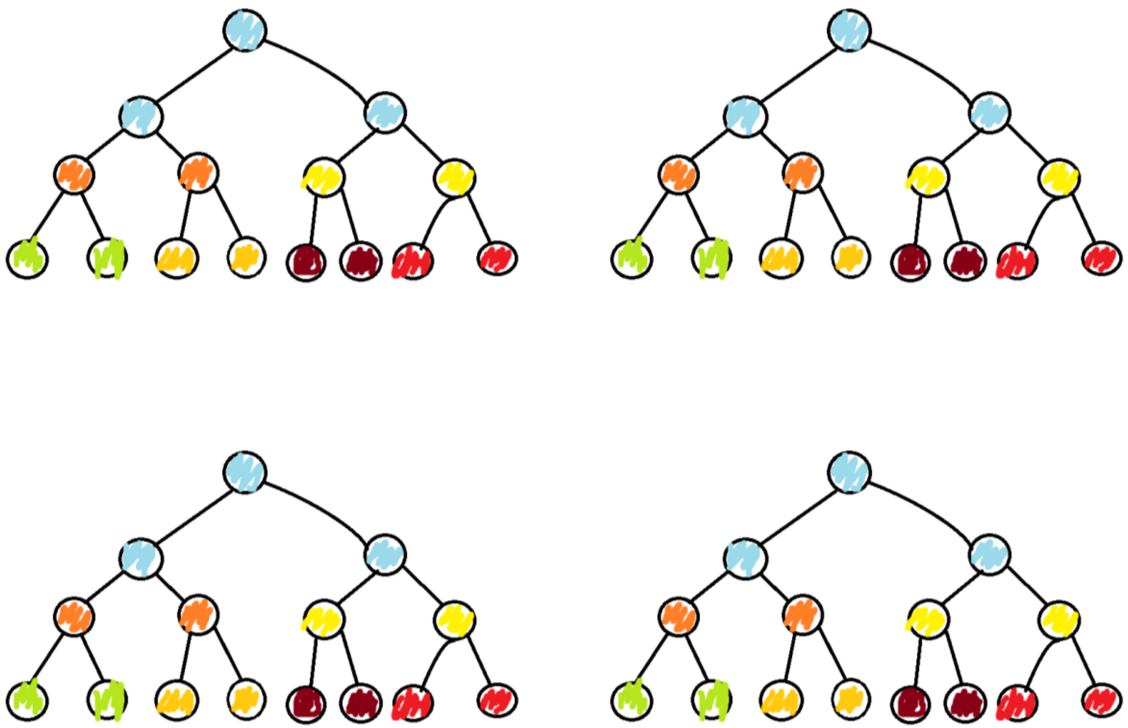
Train In Data

Hyperparameter Optimization

Parameters vs Hyperparameters

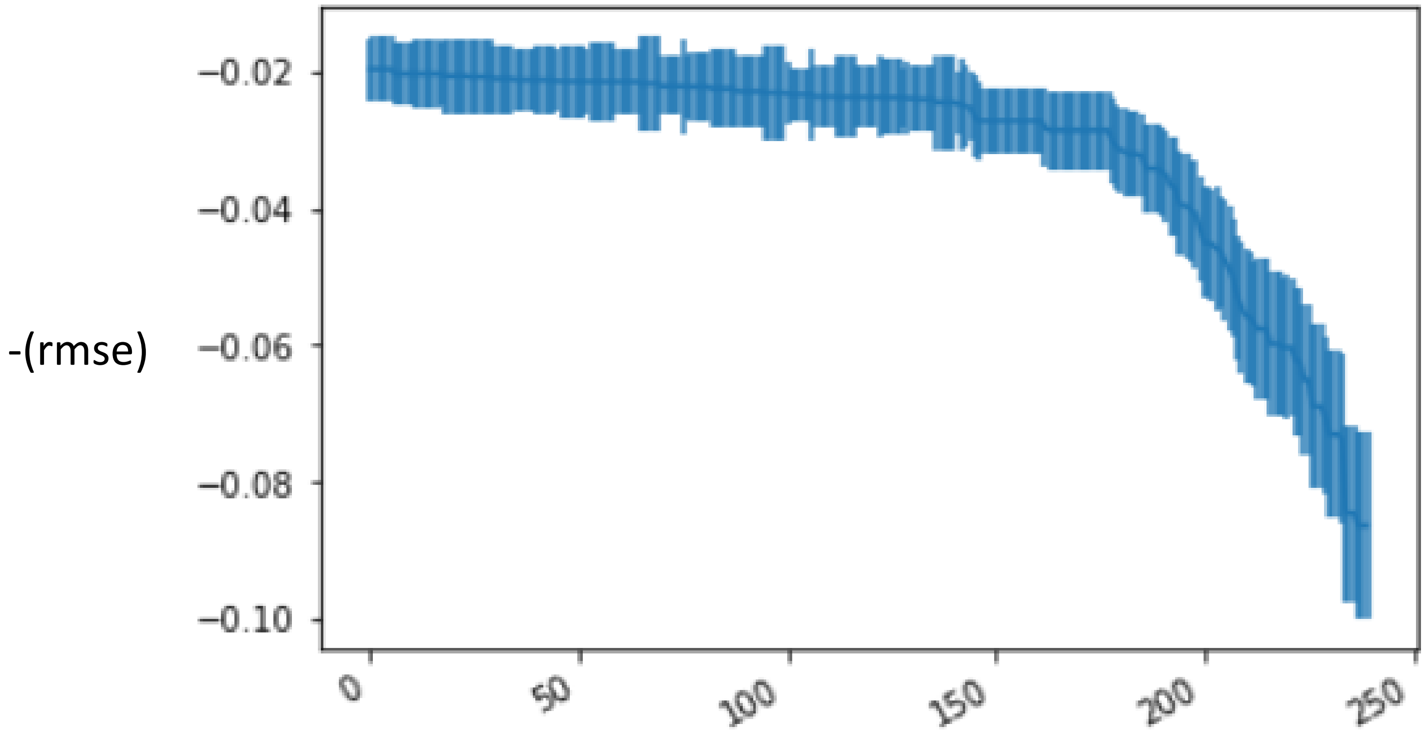
Parameters	Hyperparameters
• Intrinsic to model equation • Optimized **during** training	• Defined **before** training • Constrain the algorithm.

Random Forests and GBMs -Hyperparams



- Number of trees
- The depth of the tree
- Learning rate (GBMs)
- The metric of split quality
- The number of features to evaluate at each node
- The minimum number of samples to split the data further

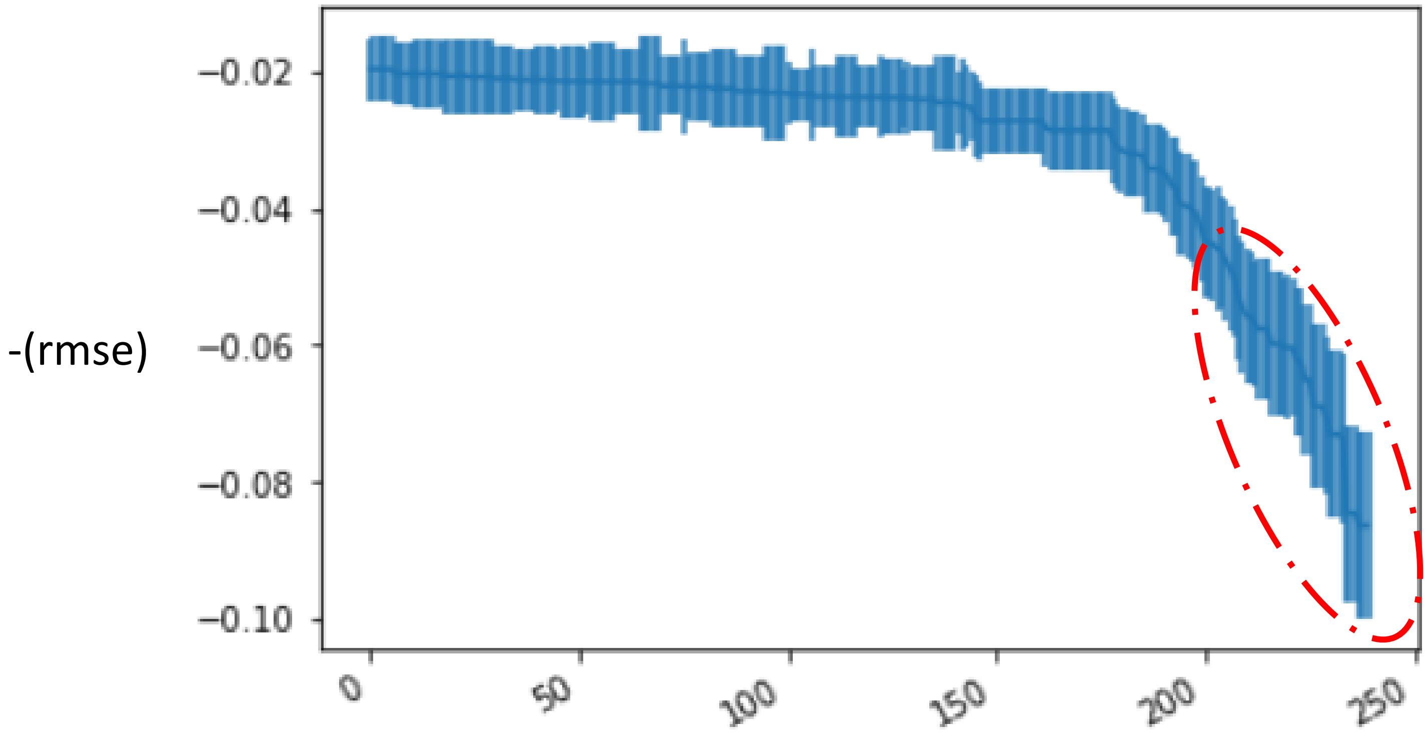
Effect of Hyperparameters



GBMs with different sets of hyperparameters

- Fit several GBMs with different hyperparameters
- Measure each model performance ➔ rmse

Effect of Hyperparameters



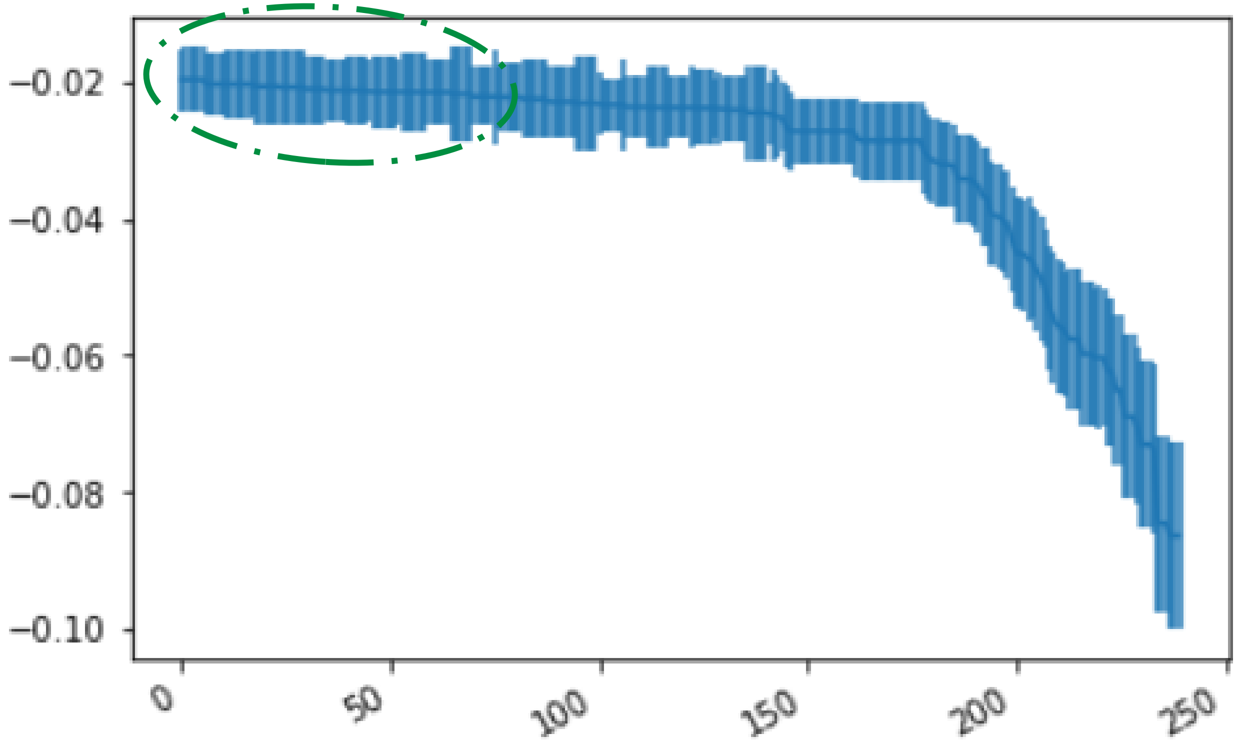
GBMs with different sets of hyperparameters

- Fit several GBMs with different hyperparameters
- Measure each model performance ➔ rmse

Certain hyperparameters return models with decreased performance

Effect of Hyperparameters

-(rmse)



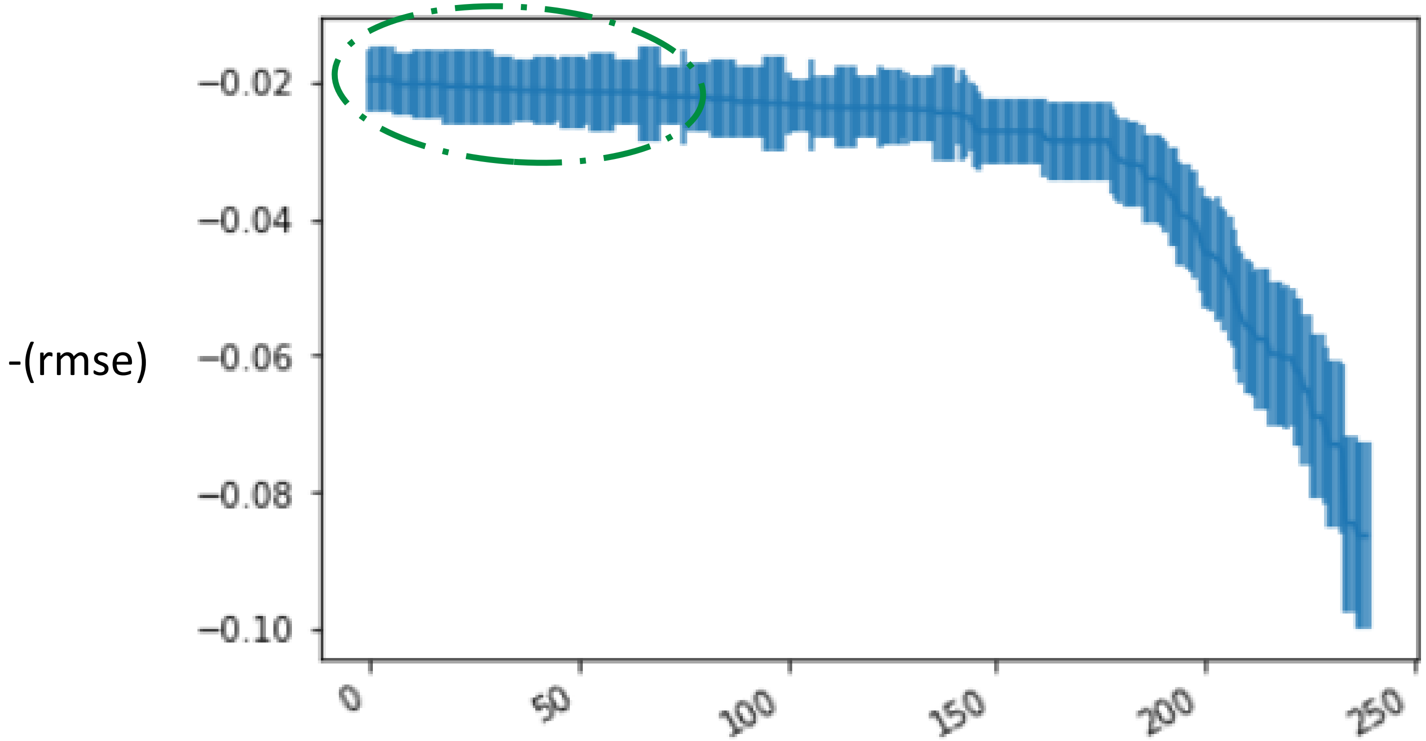
GBMs with different sets of hyperparameters

- Fit several GBMs with different hyperparameters
- Measure each model performance ➔ rmse

More than 1 combination of hyperparameters return a good fit

Effect of Hyperparameters

Low effective dimension



GBMs with different sets of hyperparameters

- Fit several GBMs with different hyperparameters
- Measure each model performance ➔ rmse

More than 1 combination of hyperparameters return a good fit

Hyperparameter Optimization

- The process of finding the best Hyperparameters for a given dataset is called **Hyperparameter Optimization** or **Hyperparameter Tuning**.

- Method to choose the hyperparameters that minimize the generalization error (not necessarily the loss)

Hyperparameter Tuning: Challenges

- We can't define a formula to find the hyperparameters
- Try different combinations of hyperparameter and evaluate model performance
- The critical step is to choose **how many** different hyperparameter combinations we are going to test.

↑ the number of hyperparameter combinations → ↑ the chance to get a better model → ↑ computational cost

Hyperparameter Tuning: Methods

⬆ the number of hyperparameter combinations ➡ ⬆ the chance to get a better model ➡ ⬆ computational cost

- How do we find the hyperparameter combinations to maximise performance while diminishing computational costs
- Different hyperparameter optimization strategies

Hyperparameter Tuning: Methods

- Manual Search
- Grid Search
- Random Search
- Bayesian Optimization
- Others

Hyperparameter Tuning: Search

A search consist of:

- Hyperparameter space
- A method for sampling candidate hyperparameters
- A cross-validation scheme
- A performance metric to minimize (or maximize)

Hyperparameter Tuning: Search

A search consist of:

- Hyperparameter space (here)
- A method for sampling candidate hyperparameters
- A cross-validation scheme (section 4)
- A performance metric to minimize (or maximize) (section 3)

Hyperparameter response surface

Find the hyperparameters that minimize (or maximize) a performance metric

Hyperparams = min(performance metric)

$$
\begin{aligned}
\lambda^{(*)} &\approx \operatorname*{argmin}_{\lambda \in \Lambda} \operatorname*{mean}_{x \in \mathcal{X}^{(\text{valid})}} \mathcal{L}\left(x; \mathcal{A}_\lambda(\mathcal{X}^{(\text{train})})\right). \\
&\equiv \operatorname*{argmin}_{\lambda \in \Lambda} \Psi(\lambda) \\
&\approx \operatorname*{argmin}_{\lambda \in \{\lambda^{(1)} \ldots \lambda^{(S)}\}} \Psi(\lambda) \equiv \hat{\lambda}
\end{aligned}
$$

Hyperparameter response surface

Find the hyperparameters that minimize (or maximize) a performance metric

Hyperparams = min(performance metric)

$$\lambda^{(*)} \approx \underset{\lambda \in \Lambda}{\operatorname{argmin}} \underset{x \in \mathcal{X}^{(\text{valid})}}{\operatorname{mean}} \mathcal{L}\left(x; \mathcal{A}_{\lambda}(\mathcal{X}^{(\text{train})})\right).$$

$$\equiv \underset{\lambda \in \Lambda}{\operatorname{argmin}} \Psi(\lambda)$$

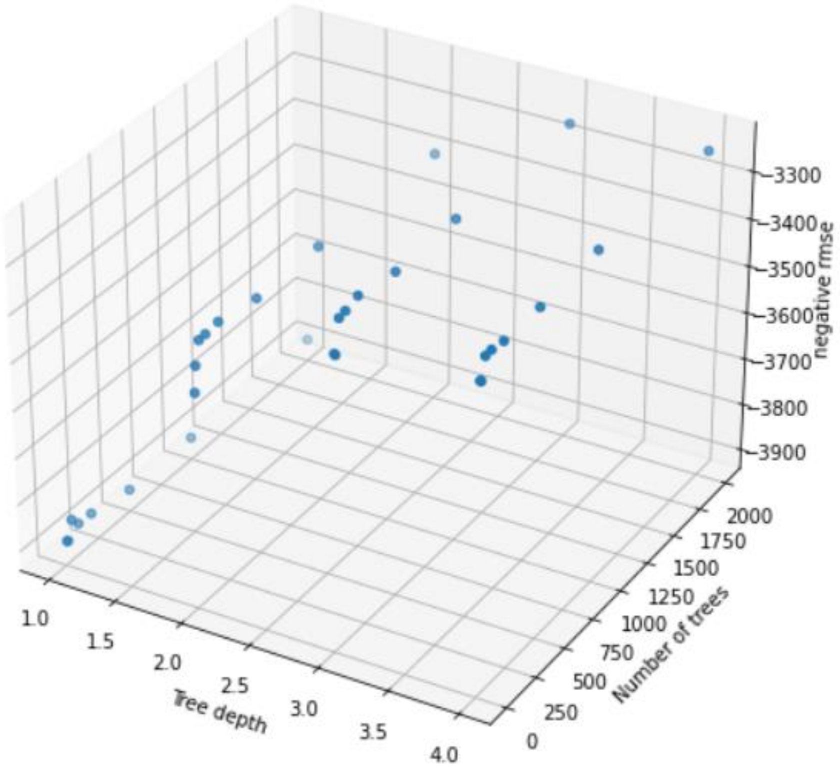
$$\approx \underset{\lambda \in \{\lambda^{(1)} \ldots \lambda^{(S)}\}}{\operatorname{argmin}} \Psi(\lambda) \equiv \hat{\lambda}$$

Response surface

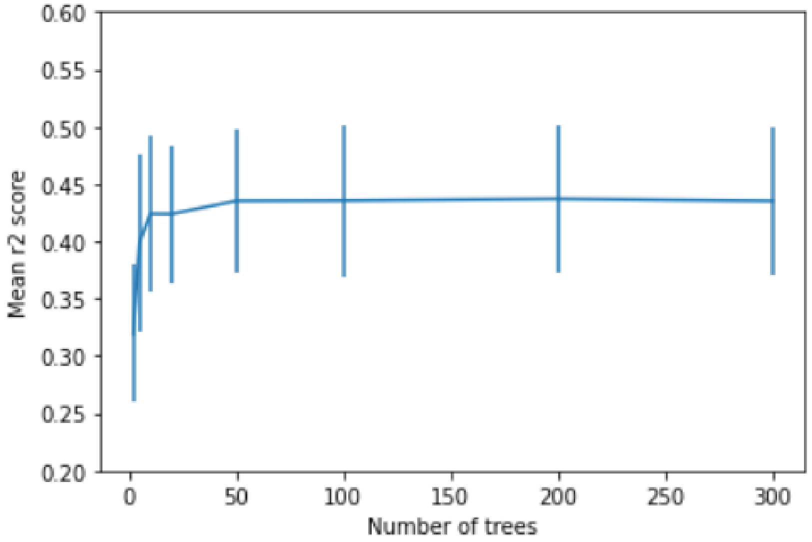
- Algorithm
- Hyperparameters
- Dataset
- Metric

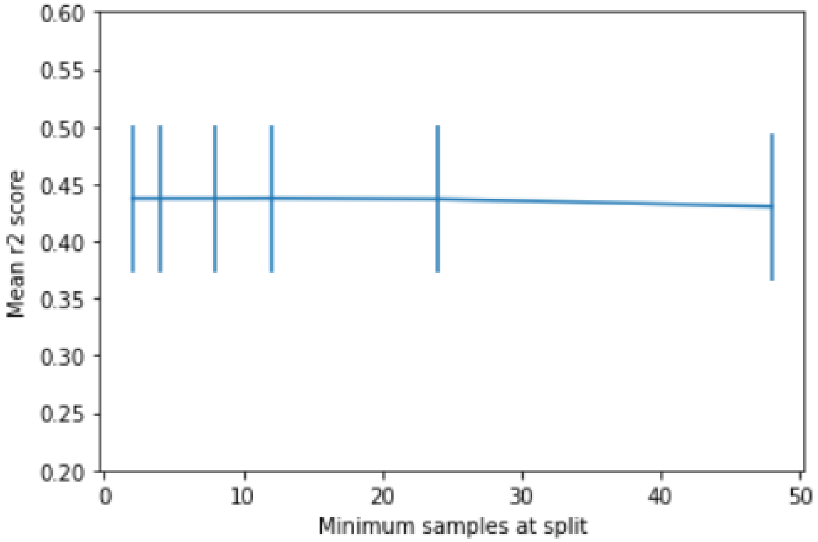
Hyperparameter response surface

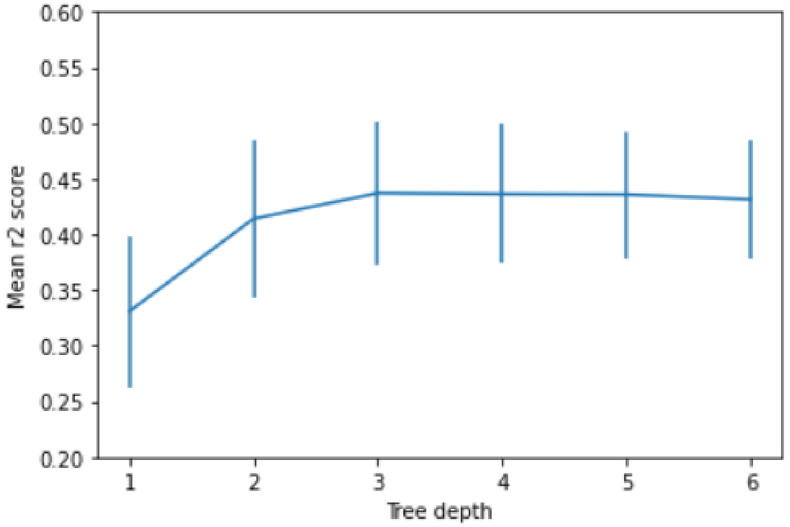
```
# random forests
rf_model = RandomForestRegressor(
    n_estimators=100, max_depth=1, random_state=0, n_jobs=4)

# hyperparameter space
rf_param_grid = dict(
    n_estimators=[10, 20, 50, 100, 200, 500, 1000, 2000],
    max_depth=[1, 2, 3, 4],
)

# search
reg = GridSearchCV(rf_model, rf_param_grid,
                   scoring='neg_mean_squared_error', cv=5)

search = reg.fit(X, y)
```



Low effective dimension







- $\Psi(\lambda)$ are more sensitive to changes in some dimensions
- Most parameters do not matter much

THANK YOU

www.trainindata.com