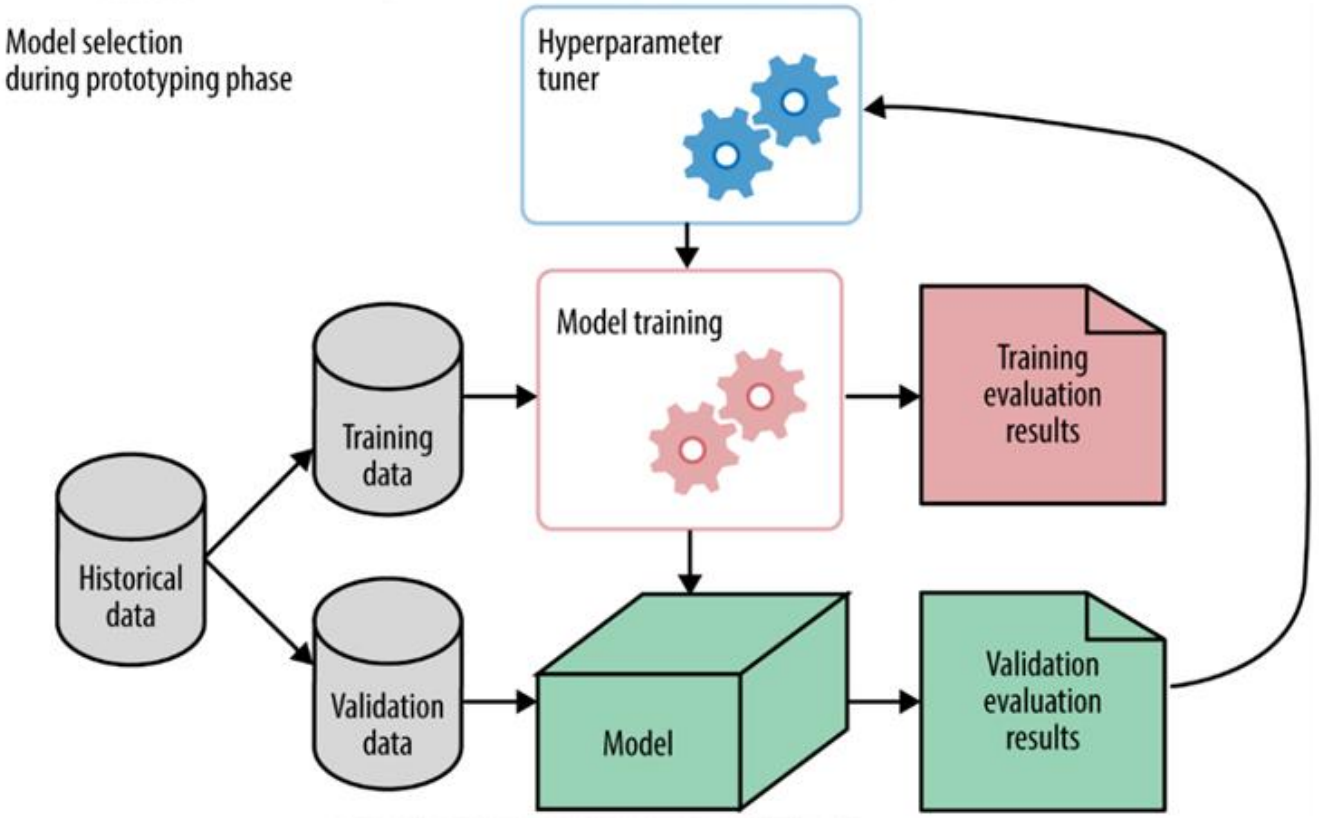
Hyperparameter Tuning with Optuna

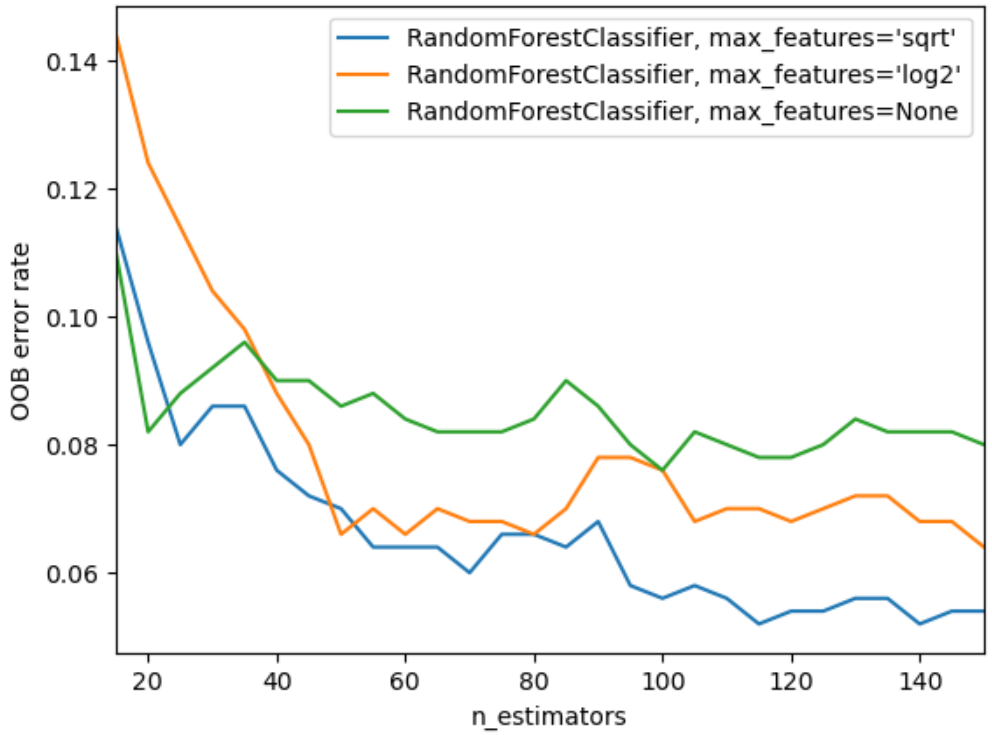
2022.01.23

Akiba, Takuya, et al. "Optuna: A next-generation hyperparameter optimization framework." *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2019.

하이퍼파라미터 튜닝의 과정



하이퍼파라미터 튜닝이 왜 필요할까?



같은 모델을 사용하더라도 하이퍼파라미터에 따라 성능이 다를 수 있다.

Source : https://velog.io/@emseoyk/%ED%95%98%EC%9D%B4%ED%8D%BC%ED%8C%8C%EB%9D%BC%EB%AF%B8%ED%84%B0-%ED%8A%9C%EB%8B%9D

시도 가능한 하이퍼파라미터 튜닝 방법

- Grid search
  - 하이퍼파라미터 후보군 설정 → 모든 조합 중 가장 좋은 성능을 내는 조합 찾기
- Randomized search
  - 하이퍼파라미터 후보군 설정 → 랜덤 조합 설정 → 가장 좋은 성능을 내는 조합 찾기
- Bayesian optimization
  - 하이퍼파라미터 후보군 설정 → 이전에 테스트한 파라미터에 대한 사전 정보 반영 → 가장 좋은 성능을 내는 조합 찾기

자세한 튜닝 방법은 Week3 혜원님 발표 자료를 참고해주세요 ☺
https://github.com/Deep-dive-into-Kaggle-and-ML/Kaggle/blob/main/week03/Hyewon/week03.pdf

Optuna

- Define-by-Run style 동적 하이퍼파라미터 최적화 프레임워크
  - TensorFlow나 Keras : 전체 그래프 정의 → session run → 모델 실행
  - PyTorch : 모델 실행 과정에서 그래프 수정 가능
- 하이퍼파라미터 후보군 설정 → 샘플링과 가지치기 → 가장 좋은 성능을 내는 조합 찾기
- 빠르게 사용 가능하며 튜닝 결과를 쉽게 시각화 가능
- Kaggle에서 하이퍼파라미터 튜닝을 위해 많이 사용하고 있음

Optuna with XGBoost

파라미터별 자세한 의미는 링크 참고

목적함수 정의 (XGBoost)

```python
def objectiveXGB(trial: Trial, X, y):
    params = {
        'n_estimators' : trial.suggest_int('n_estimators', 500, 4000),
        'max_depth' : trial.suggest_int('max_depth', 8, 16),
        'min_child_weight' : trial.suggest_int('min_child_weight', 1, 300),
        'gamma' : trial.suggest_int('gamma', 1, 3),
        'learning_rate' : 0.01,
        'colsample_bytree' : trial.suggest_discrete_uniform('colsample_bytree', 0.5, 1,
        'nthread' : -1,
        'tree_method' : 'gpu_hist',
        'predictor' : 'gpu_predictor',
        'lambda' : trial.suggest_loguniform('lambda', 1e-3, 10.0),
        'alpha' : trial.suggest_loguniform('alpha', 1e-3, 10.0),
        'subsample' : trial.suggest_categorical('subsample', [0.6, 0.7, 0.8, 1.0]),
        'random_state' : 42
    }

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

    y_train = np.array(y_train).reshape(-1, 1)
    y_test = np.array(y_test).reshape(-1, 1)

    model = xgb.XGBRegressor(**params)
    xgb_model = model.fit(X_train, y_train, verbose=False, eval_set=[(X_test, y_test)])
    score = mean_squared_error(xgb_model.predict(X_test), y_test, squared=False)

    return score
```

Optuna with XGBoost

```
study = optuna.create_study(direction='minimize', sampler=TPESampler())
study.optimize(lambda trial : objectiveXGB(trial, X, y), n_trials=100)
print('Best trial: score {},#nparams {}'.format(study.best_trial.values, study.best_trial.params))
```

목적 함수에 기반한 최적화 시도

최적의 조합 찾기

Optuna with XGBoost

- Plotly를 사용하여 상호작용 가능한 그래프 형태로 출력 가능(하나 제 PC에서는 되지 않네요)