Week 3 스터디 발표

Hyperparameter Tuning

Grid Search

모델의 파라미터 후보군 중에서 가장 좋은 성능을 내는 파라미터 조합을 찾는 방법

[특징]
- 파라미터 후보군을 하나하나 테스트하여 성능을 비교한다.
- 후보군의 수가 늘어나면 그에 비례하여 Grid search를 하는 데 소요하는 시간이 늘어난다.

```
model_lgb = lgb.LGBMRegressor()
parameters = {'n_estimators':[100,200,300,400,500,600],
              'objective':['regression'],
              'learning_rate':[0.01,0.05,0.1],
              'max_depth':[8,10,12,14,16,18,20,22],
              'num_leaves':[7,15,20,30,40],
              'num_iterations':[1000,1500,2000]}
grid_lgb = GridSearchCV(model_lgb, param_grid=parameters, cv=3, refit=True)
grid_lgb.fit(train_x, train_y)
print('best parameters : ', grid_lgb.best_params_)
print('best score : ', grid_lgb.best_score_)
```

*출처: https://github.com/Deep-dive-into-Kaggle-and-ML/Kaggle/blob/main/week03/Eunhee/lgbm_regressor.ipynb

Randomized Search

모델의 파라미터 후보군 중 랜덤하게 파라미터 조합을 선택하여 성능을 비교한 후 최적의 조합을 찾아내는 방법

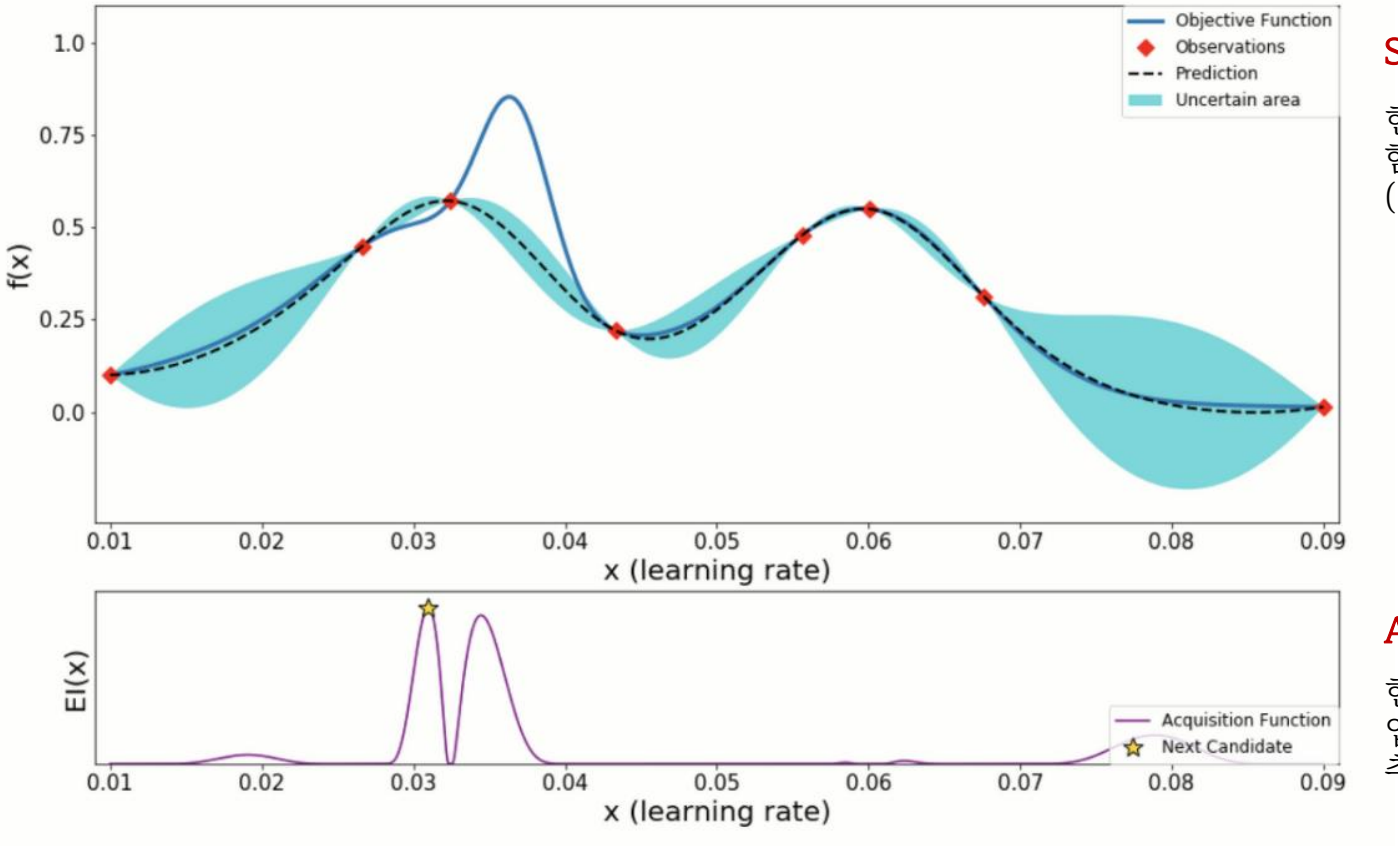
[특징]
- Grid Search에 비해 소요시간이 적다.
- 전체 파라미터 조합의 후보군에 대해 성능 테스트를 하는 것이 아니라, 랜덤하게 선택한 일부에 대해서 테스트를 진행하므로 Randomized search를 통해 찾아낸 파라미터 조합이 최적의 조합이 아닐 수 있다.

→ Grid Search와 Randomized Search 모두 이전에 테스트한 파라미터의 성능에 대한 사전 정보를 갖고 있지 않다.

Bayesian Optimization

Hyperparameter에 대한 조사 수행 시, 이전에 테스트한 파라미터에 대한 사전 정보를 반영하면서 최적의 파라미터 조합을 찾는 optimization 방법론

→ 미지의 목적함수 f(x) 를 최대로 만드는 해를 찾는 것을 목적으로 한다.



Surrogate Model

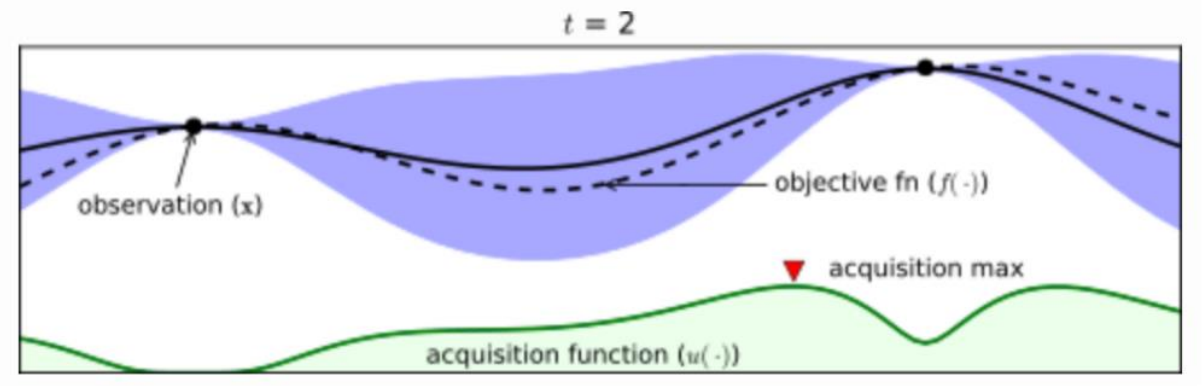
현재까지 조사된 (x, f(x)) 를 바탕으로 미지의 목적 함수에 대한 확률적인 추정을 하는 모델
(이 때 Gaussian Process가 가장 많이 이용됨)

Acquisition function

현재까지 확률적으로 추정된 결과를 바탕으로 최적의 입력값을 찾는 데 가장 유용한 다음 입력값의 후보를 추천하는 함수

Bayesian Optimization

Acquisition function



1. Exploitation: 최적값의 후보가 acquisition max point 주변에 존재할 것이라고 가정하고 탐색
2. Exploration: 표준편차가 가장 큰 점(불확실성이 가장 큰 점) 주변에 최적값이 존재할 것이라고 가정하고 탐색

→ Exploitation과 Exploration의 tradeoff