수상작 리뷰 보고서(11/26)

태그

[의류 제조 회사 생산성 예측]

https://dacon.io/competitions/official/235986/codeshare/7000?page=1&dtype=recent

데이터

1. ID
2. quarter
3. department
4. day
5. targeted_productivity
6. smv
7. wip
8. over_time
9. incentive
10. idle_time
11. idle_men
12. no_of_style_change
13. no_of_workers
14. actual_productivity

코드 흐름

1. 범주형 카테고리 인코딩

```
categorial_features = ['quarter', 'department', 'day']
one_hot_encoder = {}
for categorial_feature in categorial_features:
    features = set(list(df[categorial_feature]))
    for fname in features:
        df[fname] = (df[categorial_feature]==fname)
        test[fname] = (test[categorial_feature]==fname)
    df = df.drop(columns=categorial_feature)
    test = test.drop(columns=categorial_feature)
```

2. 파라미터 튜닝

```
br_params = {
    'n_iter': 304,
    'tol': 0.16864712769300896,
    'alpha_1': 5.589616542154059e-07,
    'alpha_2': 9.799343618469923,
    'lambda_1': 1.7735725582463822,
    'lambda_2': 3.616928181181732e-06
}

lightgbm_params = {
    'num_leaves': 5,
    'max_depth': 16,
    'learning_rate': 0.01,
    'n_estimators': 400
}

ridge_params = {
    'alpha': 631.1412445239156
}
```

3. 모델 적용 (gbr, br, ridge, lgbm)

```
models = {'gbr':GradientBoostingRegressor(),
          'br':BayesianRidge(**br_params),
          'ridge':Ridge(**ridge_params),
```

```
          'lgbm':LGBMRegressor(**lightgbm_params)}

for name, model in models.items():
  model.fit(X, y)
```

4. 결과 예측

```
y_pred = (
    0.25 * models['gbr'].predict(test) +
    0.1 * models['br'].predict(test) +
    0.1 * models['ridge'].predict(test)+
    0.25 * models['lgbm'].predict(test))

sub_df = pd.read_csv('sample_submission.csv')
sub_df['actual_productivity'] = y_pred
sub_df.to_csv('first.csv',index= False)
```

5. lightGBM 회귀 모델(LGBMRegressor)을 다양한 하이퍼파라미터 조합으로 학습시키고, 이를 교차 검증(Cross Validation)을 통해 성능을 평가하는 과정입니다. 이를 통해 가장 성능이 좋은 하이퍼파라미터 조합을 찾음.

```
import pandas as pd
import numpy as np
from sklearn.linear_model import BayesianRidge, HuberRegresso
from sklearn.ensemble import GradientBoostingRegressor
import lightgbm
from lightgbm import LGBMRegressor
from sklearn.model_selection import KFold, cross_val_score
from sklearn.preprocessing import StandardScaler

df = pd.read_csv('train.csv')
df = pd.get_dummies(df,columns=["quarter","department","day"]


X = df.drop(columns=["ID","actual_productivity"])
y = df["actual_productivity"]
print(X.shape)
```

```
scaler = StandardScaler()
#scaler.fit(X)
#X = scaler.transform(X)

max_depth = [5]
num_leaves = [16]
learning_rate = [0.01,0.02,0.005]
n_estimators = [400]

lgbm_paramlist = []

for md in max_depth:
    for nl in num_leaves:
        for lr in learning_rate:
            for ne in n_estimators:
                lgbm_paramlist.append({
    'num_leaves': nl,
    'max_depth': md,
    'learning_rate': lr,
    'n_estimators': ne})

def to_str(param):
    ret = ''
    for name, val in param.items():
        ret += name+"_"+str(val)+" "
    return ret

models = {to_str(lp):LGBMRegressor(**lp) for lp in lgbm_param
score = []

kf = KFold(n_splits=10)
for name, model in models.items():
    model.fit(X, y)
    result = np.sqrt(-cross_val_score(model, X, y, scoring='n
    score.append((np.mean(result),name))
    print(name, np.mean(result))

score.sort()
```

```
for i in range(20):
    print(score[i])
```

차별점, 배울점

- 범주형 변수를 **원-핫 인코딩**(One-Hot Encoding)을 사용하여 수치형 변수로 변환하고. standardscaler를 이용해 데이터를 표준화 진행
- lightgbm의 다양한 파라미터 조합을 생성하고 각 하이퍼파라미터 조합에 해당하는 LightGBM 모델을 값으로 가지는 딕셔너리 만들어 모델을 돌림.
- k-fold를 통해 데이터셋을 10개의 폴드로 나눠 교차 검증을 수행해 정확도를 높임.
- predict 수행시 성능 기준으로 모델마다 가중치를 다르게 주어 정확도를 높임.