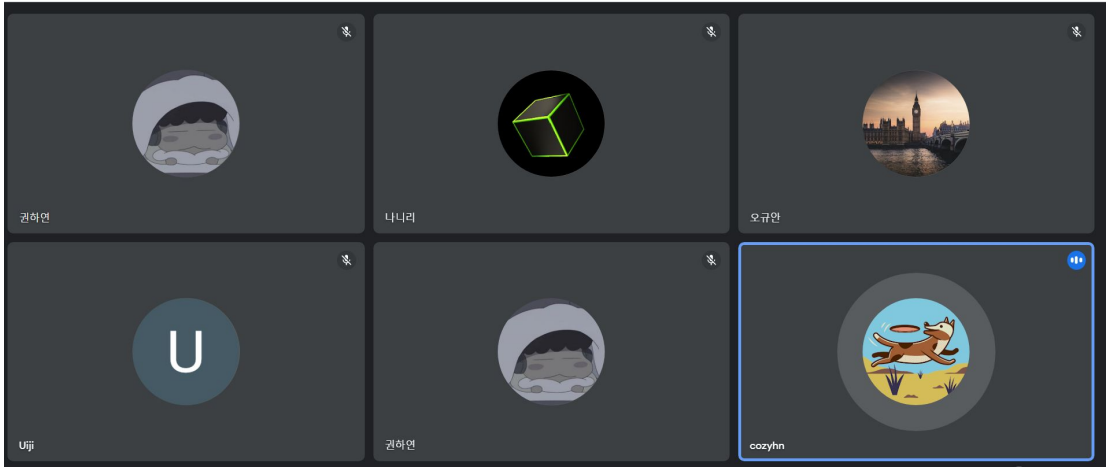
CUAI Kaggle Dacon 스터디

2023.11.07

발표자: 홍사훈

스터디원 소개



권하연
황의지
홍사훈
오규안
고지흔

스터디 소개

kaggle과 Dacon 대회 중 하나를 선택하여 코드리뷰 진행

Competition

Predict Health Outcomes of Horses

Playground

a month ago • 1541 teams

Competition

Regression with a Crab Age Dataset

Playground

5 months ago • 1429 teams

HD현대 AI Challenge

2023.09.25 연습

알고리즘 채용 정형 조선해양 회귀 MAE

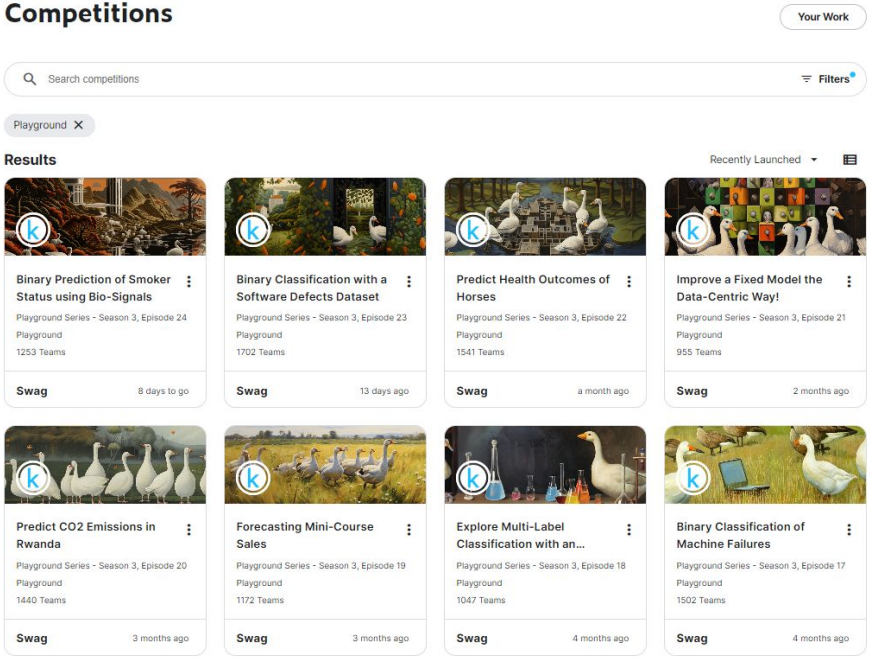
월간 데이콘 쇼츠 - AI vs Human 텍스...

2023.10.27 연습

알고리즘 언어 분류 탐지

Kaggle Playground

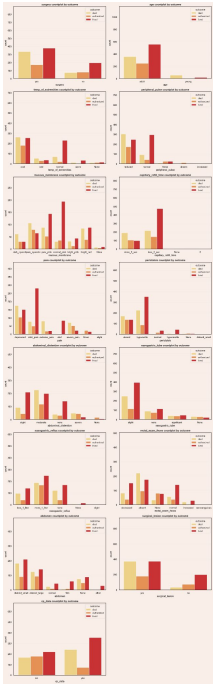
"낮은 이해관계 설정 속 새로운 유형의 문제 연습"

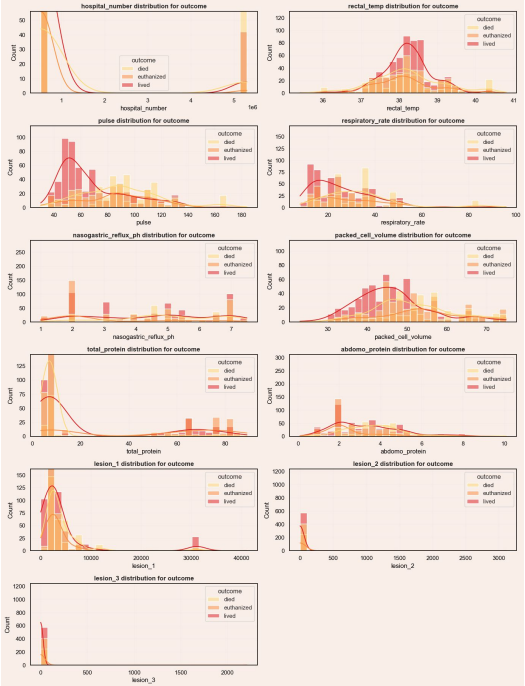


Kaggle Playground

Predict Health Outcomes of Horse

다양한 낯선 요인들을 범주형/연속형으로 분리하여 EDA진행





surgery (수술 여부에 따른 결과)

- 수술을 한 말은 그렇지 않은 말에 비해 사망률이 높다
- 수술을 하지 않은 말은 사망 비율이 상대적으로 낮고 안락사 비율이 미묘하게 높다
- 수술 할만큼 아픈 말들이라 도중에 죽였을 가능성이 높다
- 수술하지 않은 말의 경우 손도 쓰지못할 정도라 그냥 안락사 시킨 경우가 존재할 수 도있다

peristalsis (나이에 따른 결과)

- 어린 말은 안락사를 하지 않는다
- 어른 말이 생존비율이 더 높다
- 사실 어릴때 살았다는 말은 어른말이되었다는건데 이 의미는 잘 모르겠다

pain (통증)

- 심할수록 생존률 높음.
- 없을때는 각성수준, 안락사율이 높음

peris (연동운동)

- 없으면 사망률 급증
- 과다운동도 사망위험 존재
- 보통이거나 저운동성은 사망하지않음

temp_of_extremities (온도)

- 저체온일때 사망률이 가장 높다

peripheral_pulse (펄스)

- 줄어들때 사망률이 높다
- 존재하지 않으면 안락사 시킨다

mucous_membrane (점막)

- 청색증이면 사망률과 안락사비율이 높다
- 창백한 핑크도 사망률이 높은편
- 밝은 빨강이면 생존과 사망 비율이 비슷하고 안락사 비율도 높다
- 밝은 분홍도 사망률이 높음

Kaggle Playground

Regression with a Crab Age Dataset

- **dataset**

	id	Sex	Length	Diameter	Height	Weight	Shucked Weight	Viscera Weight	Shell Weight	Age
0	0	I	1.5250	1.1750	0.3750	28.973189	12.728926	6.647958	8.348928	9
1	1	I	1.1000	0.8250	0.2750	10.418441	4.521745	2.324659	3.401940	8
2	2	M	1.3875	1.1125	0.3750	24.777463	11.339800	5.556502	6.662133	9
3	3	F	1.7000	1.4125	0.5000	50.660556	20.354941	10.991839	14.996885	11
4	4	I	1.2500	1.0125	0.3375	23.289114	11.977664	4.507570	5.953395	8

1.

```
from sklearn.tree import DecisionTreeRegressor, plot_tree
from sklearn.preprocessing import MinMaxScaler, StandardScaler, LabelEncoder
from sklearn.pipeline import make_pipeline # 데이터 전처리 및 모델 학습 단계를 하나의 객체로 묶어주는데 사용됩니다.
from sklearn.linear_model import Ridge, RidgeCV, Lasso, LassoCV
from sklearn.model_selection import KFold, StratifiedKFold, train_test_split, GridSearchCV, RepeatedKFold, RepeatedStratif
from sklearn.metrics import mean_squared_error, mean_absolute_error
from sklearn.inspection import PartialDependenceDisplay  # partial dependence plot
                                                         # 특정 특성(feature)과 타겟 변수(target variable) 간의 관계를 시각
from sklearn.ensemble import RandomForestRegressor, HistGradientBoostingRegressor, GradientBoostingRegressor, ExtraTreesRe
                                                         # 히스토그램

from sklearn.svm import SVR  # Support Vector Regression 비선형 회귀, 마진(포인터 간 간격) 최대화하면서
from lightgbm import LGBMRegressor  # gradient boosting
from xgboost import XGBRegressor
from catboost import CatBoostRegressor
from sklego.linear_model import LADRegression
```

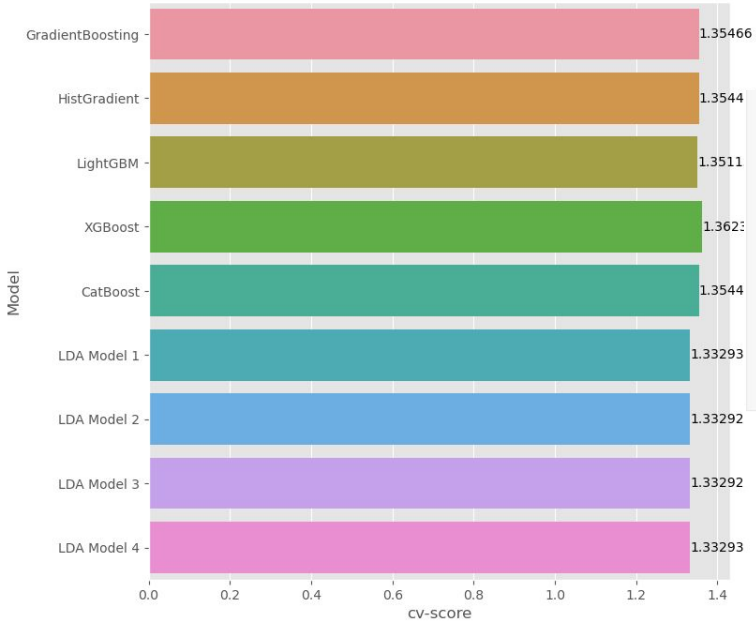
2.

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, roc_auc_score
from sklearn.model_selection import train_test_split, cross_val_predict, KFold
from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
from catboost import CatBoostRegressor, Pool
import lightgbm as lgb
import xgboost as xgb
import optuna

import warnings
warnings.simplefilter("ignore")
```

Kaggle Playground

Regression with a Crab Age Dataset

1.



2.

```
# Evaluate the model
xgb_model = xgb.XGBRegressor(objective='reg:absoluteerror',learning_rate=0.1, max_depth=7, min_chi
_weight=3, n_estimators=200, subsample=0.7, gamma=0.1, reg_lambda=0, reg_alpha=0)
lgb_model = lightgbm.LGBMRegressor(max_depth=5,num_leaves=120, n_estimators=1800,objective='mae',
tric='mean_absolute_error',reg_alpha=0.000012, reg_lambda=0.45)
catboost_model = CatBoostRegressor(loss_function='MAE',eval_metric='MAE',bagging_temperature=2.5,
lsample_bylevel=0.75,learning_rate=0.067,od_wait=40,max_depth=6,l2_leaf_reg=1.575,min_data_in_leaf
8,random_strength=0.55, max_bin=256, logging_level='Silent')


score = evaluate_model_with_stack(X,y, xgb_model, lgb_model, catboost_model)
print("Average MAE after Stacking: ",score)
```

```
Average MAE after Stacking:  1.3386856250830803
```

Dacon

현대 AI Challenge

- 정형 데이터
- MAE
- 선박의 대기시간 예측

Dacon

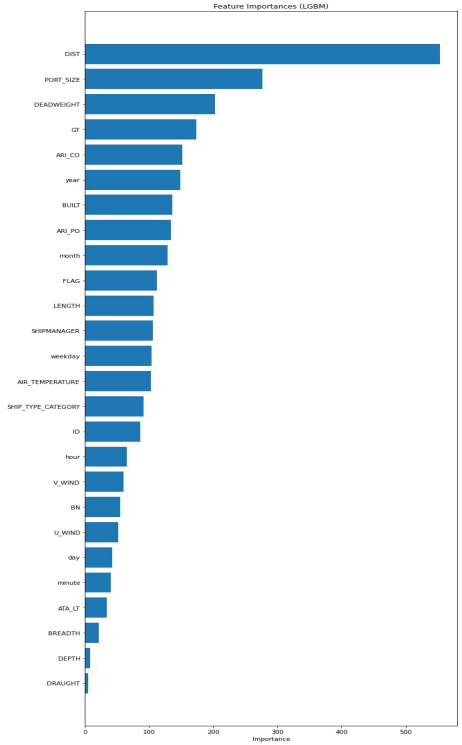
현대 AI Challenge

- 결측치 처리

```
def impute_test_missing_values(train_df, test_df, change_col, missing_col):
    condition_missing = test_df[missing_col].isna()

    replacement_values = (train_df.groupby(change_col)[missing_col].mean().reset_index(name='Imputed_' + missing_

    # Apply the means to the test data where values are missing
    merged_df = (test_df.loc[condition_missing].reset_index().merge(replacement_values, on=change_col, how='left'
    # Replace the missing values in the original test data
    test_df.loc[condition_missing, missing_col] = merged_df.set_index('index')['Imputed_' + missing_col].values

    return test_df
```

```
grouping_columns_list = [['ARI_PO', 'year', 'month', 'day', 'ATA_LT'],['ARI_PO', 'month', 'day', 'ATA_LT'],
                         ['ARI_PO', 'month', 'ATA_LT'],['ARI_PO', 'month'],['ARI_PO'],['month'],['year']]
for grouping_columns in grouping_columns_list:
    train = impute_missing_values(train, grouping_columns,'AIR_TEMPERATURE')
    test = impute_test_missing_values(train,test,grouping_columns,'AIR_TEMPERATURE')

print("temp done")

grouping_columns_list = [['ARI_PO', 'year', 'month', 'day', 'ATA_LT'],['ARI_PO', 'year', 'month', 'day'],
                         ['ARI_PO', 'month', 'ATA_LT'],['ARI_PO', 'year', 'month'],['ARI_PO'],['month'],['year']]
for grouping_columns in grouping_columns_list:
    train = impute_missing_values(train, grouping_columns,'BN')
    test = impute_test_missing_values(train,test,grouping_columns,'BN')

print("bn done")
```

- 특성 추출



Dacon

AI vs Human 텍스트 판별 해커톤

sentence1~4 사람과 기계가 작성한 문장
label - 사람이 작성한 문장의 번호

	id	sentence1	sentence2	sentence3	sentence4	label
1	TRAIN_000	직원들 마음에 들지 않는다...	직원들 진짜 싸가지 없어요...	직원들 정말 싸가지 없네요...	직원들의 태도가 정말 별로...	2
2	TRAIN_001	분위기 최고! 2층 창문이 ...	분위기가 너무 좋아요! 2층...	분위기가 짱!! 2층 창문이 ...	분위기가 너무 좋아요! 2층...	3
3	TRAIN_002	일단, 장사가 잘 되길 바라...	일단 장사가 잘되길 바라는...	일단 저는 장사가 잘되기를...	먼저, 칭찬과 응원의 의미...	2
4	TRAIN_003	1편의 독특함 때문에 살짝 ...	1편의 신선함에 비해 약간 ...	1편의 독특함 때문에 약간...	1편이 워낙 참신했던 탓에 ...	4
5	TRAIN_004	빵점 주고 싶은걸 간신히 ...	빵점을 주고 싶지만 참아냈...	빵점 주고 싶을 정도로 엄...	빵점을 주고 싶었는데 참았...	1
6	TRAIN_005	치킨 반마리 주문 가능. 골...	둘이서 치킨 반마리 주문이...	두 사람이서 반마리의 치킨...	두 사람이서 치킨 반마리 ...	2
7	TRAIN_006	진짜 일하시는 사람들 너무...	진짜 시끄러운데 ... 이런건	진짜 이 직장은 너무 시끄	진짜 일하는 분들이 너무	1
8	TRAIN_007	우와 정말 멋진 스터디 카...	와우, 정말 좋아			
9	TRAIN_008	가시려는 분들은 이전 리뷰...	가시고자 하시			
10	TRAIN_009	라운즈 앱에서 안경 구매 ...	라운즈 앱을 통			

	id	sentence1	sentence2	sentence3	sentence4
1	TEST_0000	배송은 정말 빨랐어요! 집...	배송은 정말 빠르고요~ 집...	배송은 빠르구여~집에서 ...	배송 빨라요~ 속옷 불편한...
2	TEST_0001	하.. 원래도 비싸는데 가격...	가격 상승에 비해서 품질이...	원래 가격이 높았음에도 불...	가격 상승과 함께 품질이 ...
3	TEST_0002	평일 저녁에 방문했을 때, ...	평일 저녁에 갔는데 매우 ...	평일 저녁에 방문했을 때 ...	평일 저녁에 찾아간 카페, ...
4	TEST_0003	직원들의 친절도가 좋지 않...	직원들이 불친절한 것은 물...	직원들이 친절하지 않으며,...	직원들이 친절하지 않은 것...
5	TEST_0004	갈비를 주문했음에도 다른 ...	갈비를 주문했더니 전지가 ...	갈비로 달라고 했는대..... ...	갈비를 주문했지만, 받은 ...
6	TEST_0005	"금희네"는 창동 4동 주공 ...	금희네는 창4동 주공17단...	이곳 금희네는 창4동 주공...	금희네는 창4동 주공17단...
7	TEST_0006	왜 저만이 이곳에서 기분 ...	여기의 경험이 좋지 않았군...	이곳에 갔다가 마음에 상처...	와.. 저만 여기갔다가 맘 상...
8	TEST_0007	사진 보고 맑은 순대국 먹...	사진에서는 맑은 순대국을 ...	사진을 보고 기대했던 맑은...	사진에 설명된 맑은 순대국...
9	TEST_0008	쉐프님 접객 애티튜드 때문...	쉐프님의 고객 서비스 태도...	셰프님의 태도가 살짝 불편...	셰프의 태도 때문에 별 한 ...
10	TEST_0009	1점도 너무 아까워요. 사장...	1점 주시는 건 실망스러워...	1점도 너무너무 아까움. 사...	1점 아깝다. 사장과 주방 ...

Dacon

AI vs Human 텍스트 판별 해커톤

베이스라인

```
tokenizer = AutoTokenizer.from_pretrained('skt/kogpt2-base-v2')
model = AutoModel.from_pretrained('skt/kogpt2-base-v2')
model.to(device)
```

```
with torch.no_grad():
    # 각 테스트 케이스에 대해
    for idx in tqdm(range(len(test_df))):
        row = test_df.iloc[idx]
        best_score = float('-inf')
        best_label = 0

        # train 데이터에서 랜덤하게 문장을 가져옵니다.
        random_row = train_df.sample(1).iloc[0]
        random_answer = random_row['label']
        random_labels = {}
        for i in range(1, 5):
            random_labels[f'sentence{i}'] = 'O' if i == random_answer else 'X'

        # GPT-2에게 제공할 prompt를 작성합니다.
        example_sentence = f"""
        주어진 문장이 사람이 작성한 것이 맞으면 O, 아니면 X를 반환하세요. \
        문장1 : {random_row['sentence1']} -> {random_labels['sentence1']} \
        문장2 : {random_row['sentence2']} -> {random_labels['sentence2']} \
        문장3 : {random_row['sentence3']} -> {random_labels['sentence3']} \
        문장4 : {random_row['sentence4']} -> {random_labels['sentence4']} \

        # 문제
        문장 :
        """

        # 각 문장에 대한 확률값을 구하고, 가장 높은 확률값을 가진 문장을 선택합니다.
        for i in range(1, 5):
            prompt = example_sentence + " " + row[f"sentence{i}"]
            inputs = tokenizer(prompt, return_tensors="pt")
            inputs = inputs.to(device)
            with torch.no_grad():
                outputs = model(**inputs)
                score = outputs[0][:, -1, :].max().item()

            if score > best_score:
                best_score = score
                best_label = i

        preds.append(best_label)
```

Dacon

AI vs Human 텍스트 판별 해커톤

Private-1위

* single model
A) "jhgan/ko-sroberta-multitask" (public: 0.98939, private: 0.97987)
- optimizer=adamw, learning_rate=0.00003, batch_size=64, epochs=10, cv=5, seed=826
B) "kykim/bert-kor-base" (public: 0.98939, private: 0.98896)
- optimizer=adamw, learning_rate=0.00003, batch_size=64, epochs=10, cv=5, seed=826
C) "kykim/funnel-kor-base" (public: 0.98788, private: 0.98961)
- optimizer=adamw, learning_rate=0.00003, batch_size=32, epochs=10, cv=5, seed=826

* ensemble model (soft voting)
A + B 모델을 soft ensemble한 결과는 다음과 같습니다. (public: 0.99091, private: 0.98571)
A + C 모델을 soft ensemble한 결과는 다음과 같습니다. (public: 0.99091, private: 0.98701)
B + C 모델을 soft ensemble한 결과는 다음과 같습니다. (public: 0.99091, private: 0.99416)

* final model (soft voting)
최종 제출은 3가지 모델(A+B+C)을 soft ensemble 하였습니다. (public: 0.99091, private: 0.99156)

Dacon

AI vs Human 텍스트 판별 해커톤

Private-1위

```
#리뷰를 사람이 썼는지 기계가 썼는지 0과 1로만 구분한다. 사람이 썼으면 1. 기계가 썼으면00이다
for i, v in enumerate(train_df["label"]):
    for j in range(1, 5):
        if v==j:
            train_true.append(train_df.iat[i, j-1])
        else:
            train_false.append(train_df.iat[i, j-1])

train_df_true = pd.DataFrame(train_true, columns=["text"])
train_df_false = pd.DataFrame(train_false, columns=["text"])
train_df_true["label"] = 1
train_df_false["label"] = 0

train_df = pd.concat([train_df_true, train_df_false]).reset_index(drop=True)
train_df.shape
```

	id	sentence1	sentence2	sentence3	sentence4	label
1	TRAIN_000	직원들 마음에 들지 않는다...	직원들 진짜 싸가지 없어요...	직원들 정말 싸가지 없네요...	직원들의 태도가 정말 별로...	2
2	TRAIN_001	분위기 최고! 2층 창문이 ...	분위기가 너무 좋아요! 2층...	분위기가 짱!! 2층 창문이 ...	분위기가 너무 좋아요! 2층...	3
3	TRAIN_002	일단, 장사가 잘 되길 바라...	일단 장사가 잘되길 바라는...	일단 저는 장사가 잘되기를...	먼저, 정찬과 응원의 의미...	2
4	TRAIN_003	1편의 독특함 때문에 살짝 ...	1편의 신선함에 비해 약간 ...	1편의 독특함 때문에 약간...	1편이 워낙 참신했던 탓에 ...	4
5	TRAIN_004	행정 주고 싶은걸 간신히 ...	행정을 주고 싶지만 참아냈...	행정 주고 싶을 정도로 엄...	행정을 주고 싶었는데 참았...	1
6	TRAIN_005	치킨 반마리 주문 가능. 골...	홀에서 치킨 반마리 주문이...	두 사람이서 반마리의 치킨...	두 사람이서 치킨 반마리 ...	2
7	TRAIN_006	진짜 일하시는 사람들 너무...	진짜 시끄러운데... 이런거 ...	진짜 이 직장은 너무 시끄...	진짜 일하는 분들이 너무 ...	1
8	TRAIN_007	우와 정말 멋진 스터디 카...	와우, 정말 좋아요! 이 스터...	와우, 정말 좋은 공부 카페...	우와 완전좋아요 스터디카...	4
9	TRAIN_008	가시려는 분들은 이전 리뷰...	가시고자 하시는 분들은 앞...	이 식당을 방문하실 분들께...	요즘은 리뷰를 믿기 힘들죠...	2
10	TRAIN_009	라운즈 앱에서 안경 구매 ...	라운즈 앱을 통해 안경과 ...	라운즈 앱을 통해 안경과 ...	라운즈 앱으로 안경 구매 ...	1

	text	label
0	직원들 진짜 싸가지 없어요 ㅋㅋㅋㅋ 가지 마숑 인터넷이 더 싼거 알면서도 이것저것...	1
1	분위기가 짱!! 2층 창문이 커서 탁 트여있는 느낌이에요 ㅎㅎ 조명도 예쁘고 음료랑...	1
2	일단 장사가 잘되길 바라는 마음에서 별5개 드립니다 간도 맞았고 매운걸 좋아하는 입...	1
3	1편이 워낙 참신했던 탓에 좀 묻힌 감이 있긴 하지만 재미는 여전합니다. 시스템도 ...	1

Dacon

AI vs Human 텍스트 판별 해커톤

Private-1위

model - jhgan/ko-sroberta-multitask

Validate metric	DataLoader 0
val_acc	0.925000011920929
val_loss	0.17511601746082306

Validate metric	DataLoader 0
val_acc	0.9750000238418579
val_loss	0.11738772690296173

Validate metric	DataLoader 0
val_acc	0.9750000238418579
val_loss	0.15472881495952606

Validate metric	DataLoader 0
val_acc	0.9750000238418579
val_loss	0.11738772690296173

Validate metric	DataLoader 0
val_acc	0.925000011920929
val_loss	0.17511601746082306

감사합니다