[Dacon] 심리 성향 예측 AI 경진대회 - Auto ML 하는 방법 — 나무늘보의 개발 블로그

노트북:	첫 번째 노트북
만든 날짜:	2021-06-21 오후 11:27
URL:	https://continuous-development.tistory.com/247

Data scientist/Machine Learning

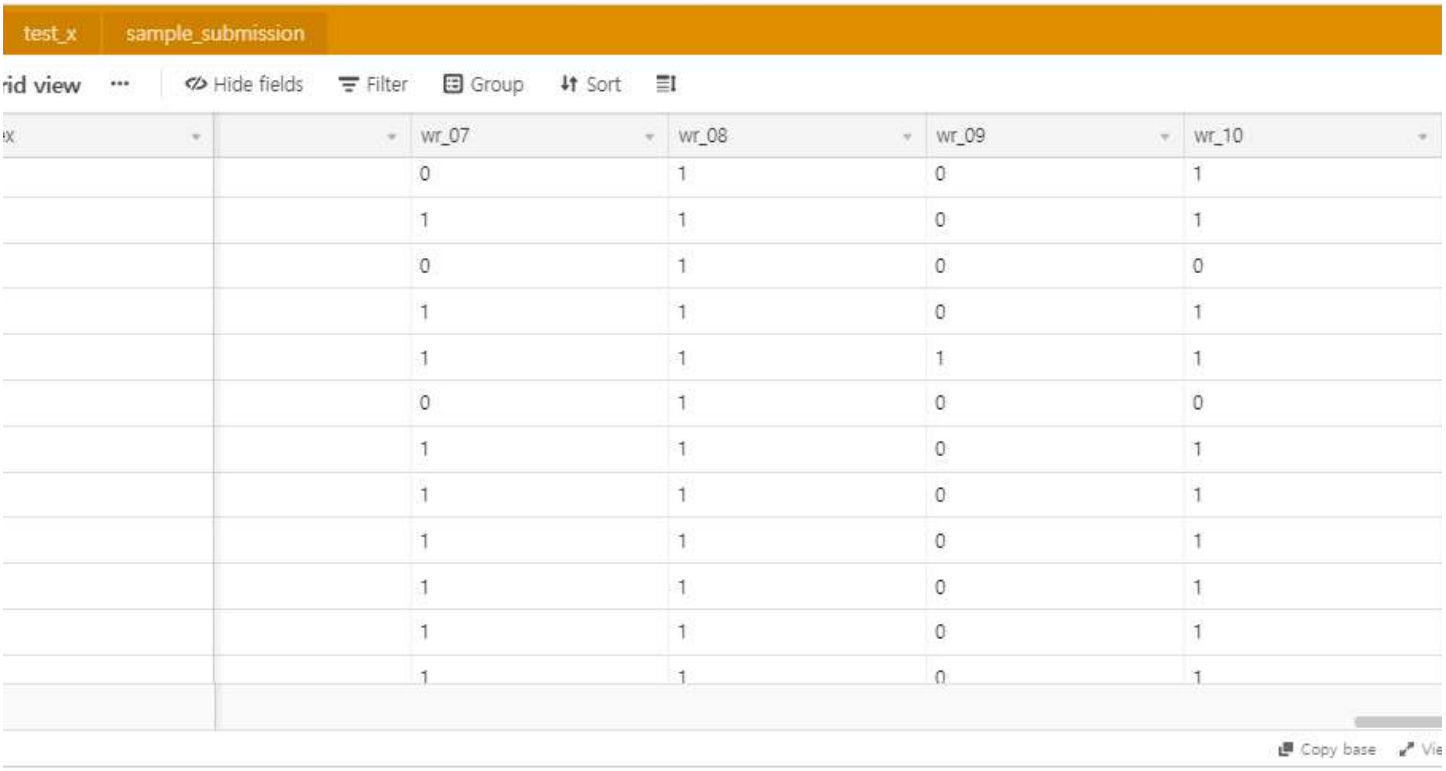
[Dacon] 심리 성향 예측 AI 경진대회 - Auto ML 하는 방법

2021. 6. 21. 16:54 수정 삭제 공개

심리 성향 예측 AI 경진대회

이 대회는 Dacon에서 진행한 대회로서 **심리학 테스트 분석 알고리즘 개발** 하는 대회였다. (https://dacon.io/competitions/official/235647/overview/description)
이 대회에서 어떤 착한분이 간단하게 AutoML를 하는 소스에 대해서 공유를 해줬는데 AutoML를 경험해보는데 있어서 좋은 경험이 될 것 같아서 가져왔다.

데이터



test_x sample_submission

id view ··· Hide fields Filter Group Sort

x		wr_07	wr_08	wr_09	wr_10
		0	1	0	1
		1	1	0	1
		0	1	0	0
		1	1	0	1
		1	1	1	1
		0	1	0	0
		1	1	0	1
		1	1	0	1
		1	1	0	1
		1	1	0	1
		1	1	0	1
		1	1	0	1

Copy base Vie

평가

심사 기준: AUC

소스

경로 설정 (Define your path)

```
path = 'data/'

import os
os.listdir(path)
```

데이터 불러오기 (Read Data)

```
import pandas as pd
train = pd.read_csv(path + 'train.csv')
```

```
test = pd.read_csv(path + 'test_x.csv')
submission = pd.read_csv(path + 'sample_submission.csv')
```

머신을 돌릴 때 앞부분의 소스는 이런식으로 고정해 놓고 path를 바꾸는 형식으로 하면 소스 템플릿 처럼 사용하기 편 할 것 같다.

데이터 구조 확인 (Checking the shapes of data)

```
print(train.shape)
print(test.shape)
print(submission.shape)
```

PyCaret 패키지 설치 (Install PyCaret)

```
!pip install pycaret
```

우리가 autoML를 하는데 있어 사용하는 패키지는 pycaret이다.

분류 작업에 필용한 함수 불러오기 (Import methods for classification task)

```
from pycaret.classification import *
```

실험 환경 구축 (Setup the environment)

PyCaret에서는 모델 학습 전 실험 환경을 구축 해주어야 합니다. setup 함수를 통해 환경을 구축할 수 있습니다.

setup 단계에서는 PyCaret이 자동으로 컬럼 형태를 인식합니다. 그 후 사용자에게 제대로 인식되었는지 확인을 받게 됩니다. 그 때 enter를 눌러주시면 됩니다.

또한 주어진 데이터의 얼마를 사용하여 train / validation을 구축할지 묻게 되는데, 전체 데이터를 사용하고 싶다면 enter 눌러주시면 됩니다.

```
# 'voted' 컬럼이 예측 대상이므로 target 인자에 명시
# 'voted' column is the target variable
```

```
clf = setup(data = train, target = 'voted')
```

	Description	Value
0	session_id	6636
1	Target Type	Binary
2	Label Encoded	1: 0, 2: 1
3	Original Data	(45532, 78)
4	Missing Values	False
5	Numeric Features	42
6	Categorical Features	35
7	Ordinal Features	False
8	High Cardinality Features	False
9	High Cardinality Method	None
10	Sampled Data	(45532, 78)
11	Transformed Train Set	(31872, 201)
12	Transformed Test Set	(13660, 201)
13	Numeric Imputer	mean
14	Categorical Imputer	constant
15	Normalize	False
16	Normalize Method	None

이런식으로 각 컬럼이 어떤 형태인지 확인을 한다. setup을 통해 어떤 데이터를 가지고 할지를 정한다. 거기서 data를 train으로 정하고 그 타겟 대상(맞춰야하는 대상)을 voted라고 명시한다.

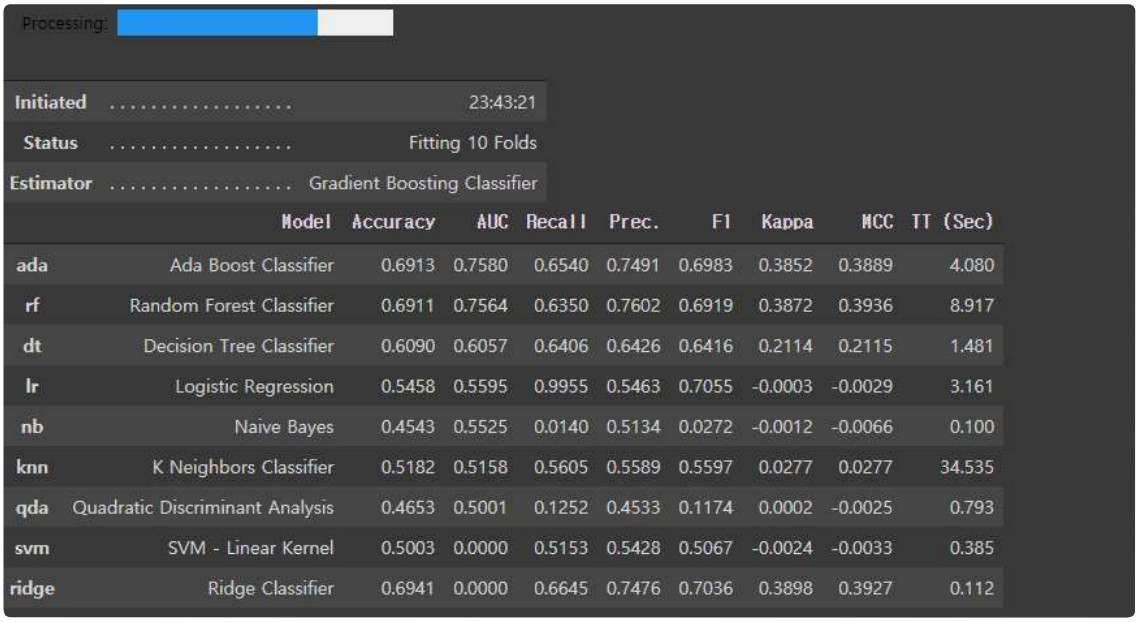
모델 학습 및 비교 (Train models and compare)

환경 구축을 했으니 PyCaret에서 제공하는 기본 모델에 대해 학습하고 비교해보겠습니다.

compared_models 함수를 통해 15개의 기본 모델을 학습하고 성능을 비교할 수 있습니다.

AUC 기준으로 성능이 가장 좋은 3개의 모델을 추려내어 저장해보겠습니다. 본 대회 평가지표가 AUC이기 때문에 AUC 기준으로 모델을 선정합니다.

```
best_3 = compare_models(sort = 'AUC', n_select = 3)
```



	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
ada	Ada Boost Classifier	0.6913	0.7580	0.6540	0.7491	0.6983	0.3852	0.3889	4.080
rf	Random Forest Classifier	0.6911	0.7564	0.6350	0.7602	0.6919	0.3872	0.3936	8.917
dt	Decision Tree Classifier	0.6090	0.6057	0.6406	0.6426	0.6416	0.2114	0.2115	1.481
lr	Logistic Regression	0.5458	0.5595	0.9955	0.5463	0.7055	-0.0003	-0.0029	3.161
nb	Naive Bayes	0.4543	0.5525	0.0140	0.5134	0.0272	-0.0012	-0.0066	0.100
knn	K Neighbors Classifier	0.5182	0.5158	0.5605	0.5589	0.5597	0.0277	0.0277	34.535
qda	Quadratic Discriminant Analysis	0.4653	0.5001	0.1252	0.4533	0.1174	0.0002	-0.0025	0.793
svm	SVM - Linear Kernel	0.5003	0.0000	0.5153	0.5428	0.5067	-0.0024	-0.0033	0.385
ridge	Ridge Classifier	0.6941	0.0000	0.6645	0.7476	0.7036	0.3898	0.3927	0.112

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
0	CatBoost Classifier	0.6931		0.6581	0.7501	0.7011	0.3885	0.3921	
1	Gradient Boosting Classifier		0.7636	0.6406		0.6957			
2	Light Gradient Boosting Machine	0.6919	0.7625	0.6457	0.7554	0.6962	0.3876	0.3926	
3	Linear Discriminant Analysis	0.6914	0.7606	0.6630	0.7447	0.7014	0.3843	0.3871	
4	Extra Trees Classifier	0.6885	0.7576	0.6467	0.7493	0.6942	0.3803	0.3846	
5	Ada Boost Classifier	0.6882	0.7545	0.6534	0.7451	0.6962	0.3788	0.3823	
6	Extreme Gradient Boosting	0.6732	0.7432	0.6633	0.7178	0.6894	0.3458	0.3471	
7	Random Forest Classifier	0.6509	0.7090	0.6019	0.7147	0.6534	0.3070	0.3116	
8	Decision Tree Classifier	0.6101	0.6070	0.6398	0.6446	0.6421	0.2138	0.2139	
9	Naive Bayes	0.4535	0.5110	0.0102	0.5177	0.0199	-0.0013	-0.0064	
10	K Neighbors Classifier	0.5139	0.5102	0.5547	0.5556	0.5551	0.0194	0.0194	
11	Quadratic Discriminant Analysis	0.4532	0.5001	0.0000	0.0000	0.0000	0.0000	0.0000	
12	Logistic Regression	0.5466	0.4783		0.5468		-0.0001	0.0000	
13	SVM - Linear Kernel	0.5028	0.0000	0.5803	0.5428	0.5560	-0.0106	-0.0116	
14	Ridge Classifier	0.6915	0.0000	0.6634	0.7445	0.7016	0.3844	0.3872	

여기서는 모델을 비교한다. 여러가지의 모델을 넣고 그 중 3개를 선택하는 것이라고 생각하면 될 것 같다. 그중 3가지를 추려내 best_3 라는 변수명에 저장한다.

모델 앙상블 (Model Ensemble)

학습된 3개의 모델을 앙상블 시키도록 하겠습니다. 본 대회는 score 최적화를 위해 확률 값을 예측해야 하므로 soft vote ensemble을 진행하겠습니다.

```
blended = blend_models(estimator_list = best_3, fold = 5, method = 'soft')
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.6985	0.7716	0.6569	0.7593	0.7044	0.4000	0.4044
1	0.6907	0.7607	0.6388	0.7575	0.6931	0.3858	0.3915
2	0.6895	0.7603	0.6428	0.7532	0.6936	0.3829	0.3879
3	0.6961	0.7677	0.6568	0.7554	0.7027	0.3950	0.3991
4	0.6939	0.7664	0.6374	0.7638	0.6949	0.3928	0.3993
Mean							
SD	0.0033	0.0043	0.0086	0.0036	0.0048	0.0062	0.0059

이전에 뽑았던 3가지를 가지고 vote를 하는데 있어서 soft 한 방식을 사용한다. ensemble중 대표적인 voting을 사용한 것 같다. voting에는 soft / hard한 방식으로 나눠진다.
hard는 각각의 모델들이 결과를 예측하면 단순하게 가장 많은 표를 얻은 결과를 선택하는 것이다.
soft는 각 class별로 모델들이 예측한 probability를 합산해서 가장 높은 class를 선택하는 것이다.

모델 예측 (Prediction)

구축된 앙상블 모델을 통해 예측을 해보겠습니다.

setup 환경에 이미 hold-out set이 존재하므로 해당 데이터에 대해 예측을 하여 모델 성능을 확인하겠습니다.

```
pred_holdout = predict_model(blended)
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Voting Classifier	0.7001	0.7725	0.6471	0.7679	0.7024	0.4045	0.4105

이제 만들어진 모델을 통해 예측을 한다.

전체 데이터에 대한 재학습 (Re-training the model on whole data)

현재까지 실험은 주어진 train 데이터를 다시 한 번 train / validation으로 나눠서 실험을 한 것이므로, 전체 train 데이터에 학습되어 있지 않습니다.

최적의 성능을 위해 전체 데이터에 학습을 시켜주도록 하겠습니다.

```
final_model = finalize_model(blended)
```

대회용 test set에 대한 예측 (Predicting on test set for the competition)

predict_model 함수를 통해 재학습된 모델을 대회용 test set에 대해 예측해보겠습니다.

We will now use the re-trained model on the test set for the competition

```
predictions = predict_model(final_model, data = test)
predictions
```

정리

1. pycaret
2. 모델 앙상블

후기

처음 autoML를 보고 사용 했을때 되게 신세계였다. 여러가지 사람이 하는 작업들을 기계가 알아서 한다니 몬가 이해는 되긴하지만 사람의 공수가 점점 더 적어지는 것 같다. 나중에는 사람은 알고리즘에 대한 기본적인 이해 없이도 이런것들을 사용하는 방법만 익혀도 AI를 하는데 지장이 없을 것 같다. 몬가 이해보다는 사용법을 아는게 더 중요해질 것 같다.
다른 사람들도 이것을 한 번 해봤으면 좋겠다.

'Data scientist > Machine Learning' 카테고리의 다른 글

[Dacon] 심리 성향 예측 AI 경진대회 - Auto ML 하는 방법
[ML/DL] 베이지안 최적화(Bayesizan Optimization)란?
[ML/DL] 회귀(Regression)의 정의와 구현
[ML/DL] 군집화의 정의와 종류 및 구현
[ML/DL] XGboost의 정의와 구현 및 hyper parameter 설정
[ML/DL] 앙상블 학습 (Ensemble Learning): 3.Boosting(부스팅)이란?

Auto ML 오토 머신러닝

나아무늘보
혼자 끄적끄적하는 블로그 입니다.