머신 러닝 입문

Rep4. 실제 현장 CSV 데이터셋을 이용한 지도학습 프로젝트보고서

컴퓨터시스템과 3-A 202045016 허유진

목차

1. 프로젝트 개요 (문제인식 및 필요성)

프로젝트의 문제인식 및 필요성

와인을 잘 모르는 사람은 어떤 와인이 좋은 와인인지 잘 알지 못해 가격이 비싸면 좋은 와인이라고 생각하는 경향이 크다.

-> 가격을 제외한 레드 와인의 성분을 통해서 와인의 품질을 파악하는 서비스

효과

1. 와인 초심자의 선택의 폭 증가
2. 와인 선택에 대한 고민의 시간 절감
3. 와인에 대한 진입장벽을 낮출 수 있음

1. 프로젝트 개요 (문제인식 및 필요성)

Red Wine Quality

Simple and clean practice dataset for regression or classification modelling

Data Card | Code (1431) | Discussion (22)

About Dataset

Context

The two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine. For more details, consult the reference [Cortez et al., 2009]. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.).

These datasets can be viewed as classification or regression tasks. The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones).

This dataset is also available from the UCI machine learning repository, https://archive.ics.uci.edu/ml/datasets/wine+quality , I just shared it to kaggle for convenience. (If I am mistaken and the public license type disallowed me from doing so, I will take this down if requested.)

Content

For more information, read [Cortez et al., 2009].

Input variables (based on physicochemical tests):

Usability
8.82

License
Database: Open Database, Cont...

Expected update frequency
Not specified

레드 와인 퀄리티 캐글 링크 : https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009

2. 데이터 수집 및 데이터분석(EDA 방법)

	A	B	C	D	E	F	G	H	I	J	K	L
1	fixed acidi	volatile aci	citric acid	residual su	chlorides	free sulfur	total sulfu	density	pH	sulphates	alcohol	quality
2	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
3	7.8	0.88	0	2.6	0.098	25	67	0.9968	3.2	0.68	9.8	5
4	7.8	0.76	0.04	2.3	0.092	15	54	0.997	3.26	0.65	9.8	5
5	11.2	0.28	0.56	1.9	0.075	17	60	0.998	3.16	0.58	9.8	6
6	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
7	7.4	0.66	0	1.8	0.075	13	40	0.9978	3.51	0.56	9.4	5
8	7.9	0.6	0.06	1.6	0.069	15	59	0.9964	3.3	0.46	9.4	5
9	7.3	0.65	0	1.2	0.065	15	21	0.9946	3.39	0.47	10	7
10	7.8	0.58	0.02	2	0.073	9	18	0.9968	3.36	0.57	9.5	7
11	7.5	0.5	0.36	6.1	0.071	17	102	0.9978	3.35	0.8	10.5	5
12	6.7	0.58	0.08	1.8	0.097	15	65	0.9959	3.28	0.54	9.2	5
13	7.5	0.5	0.36	6.1	0.071	17	102	0.9978	3.35	0.8	10.5	5
14	5.6	0.615	0	1.6	0.089	16	59	0.9943	3.58	0.52	9.9	5
15	7.8	0.61	0.29	1.6	0.114	9	29	0.9974	3.26	1.56	9.1	5
16	8.9	0.62	0.18	3.8	0.176	52	145	0.9986	3.16	0.88	9.2	5
17	8.9	0.62	0.19	3.9	0.17	51	148	0.9986	3.17	0.93	9.2	5
18	8.5	0.28	0.56	1.8	0.092	35	103	0.9969	3.3	0.75	10.5	7
19	8.1	0.56	0.28	1.7	0.368	16	56	0.9968	3.11	1.28	9.3	5
20	7.4	0.59	0.08	4.4	0.086	6	29	0.9974	3.38	0.5	9	4
21	7.9	0.32	0.51	1.8	0.341	17	56	0.9969	3.04	1.08	9.2	6

[Winequality-red]

[데이터]

X : fixed acid, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol

Y : quality

2. 데이터 수집 및 데이터분석(EDA 방법)

```
data = pd.read_csv('winequality-red.csv')
```

data.shape

```
(1599, 12)
```

data. isnull().sum()

```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

2. 데이터 수집 및 데이터분석(EDA 방법)

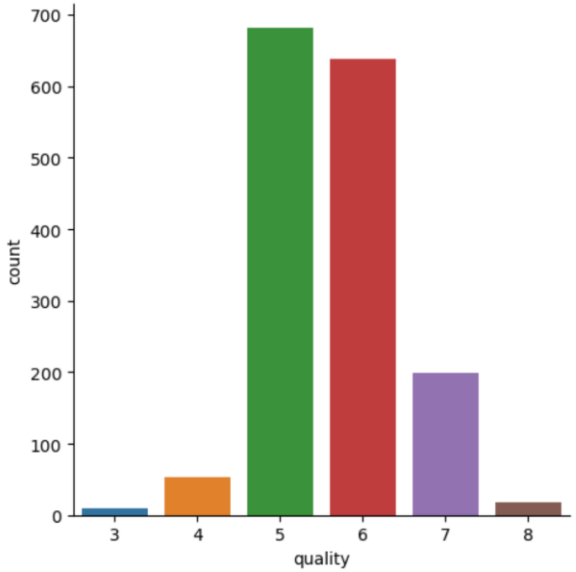
data.describe()

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	0.658149	10.422983	5.636023
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	0.169507	1.065668	0.807569
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.000000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	0.550000	9.500000	5.000000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	0.620000	10.200000	6.000000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	0.730000	11.100000	6.000000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000	2.000000	14.900000	8.000000

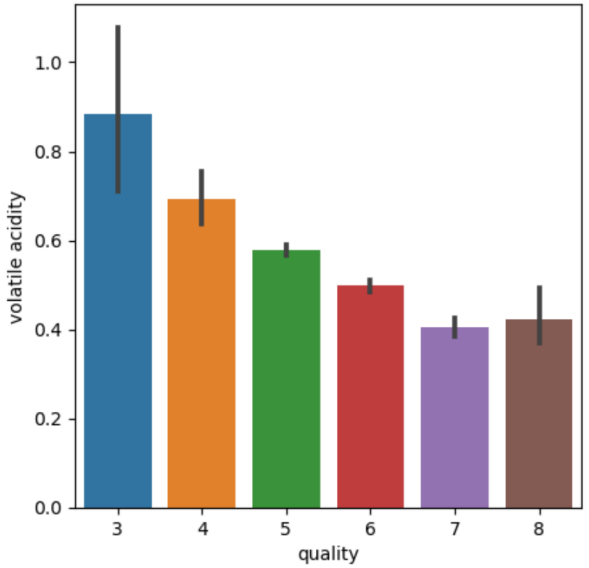
데이터의 크기와 컬럼들 사이에 null이 없는 것, 그리고 열에 대한 간단한 통계를 확인할 수 있습니다.

2. 데이터 수집 및 데이터분석(EDA 방법)

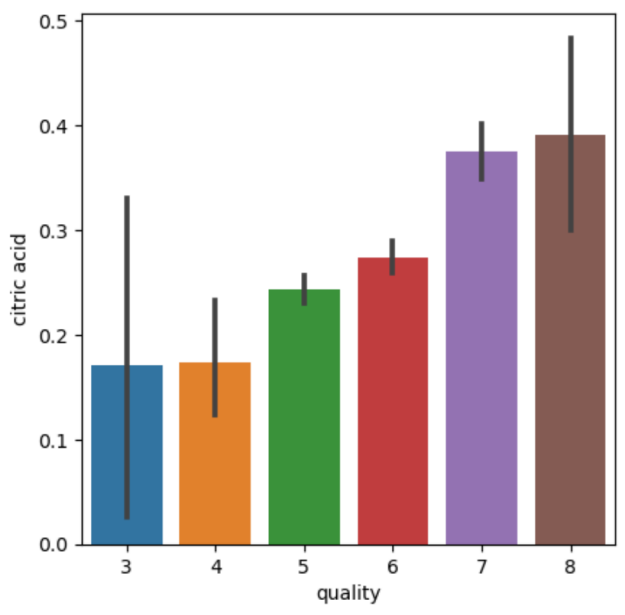
[Quality – Count]



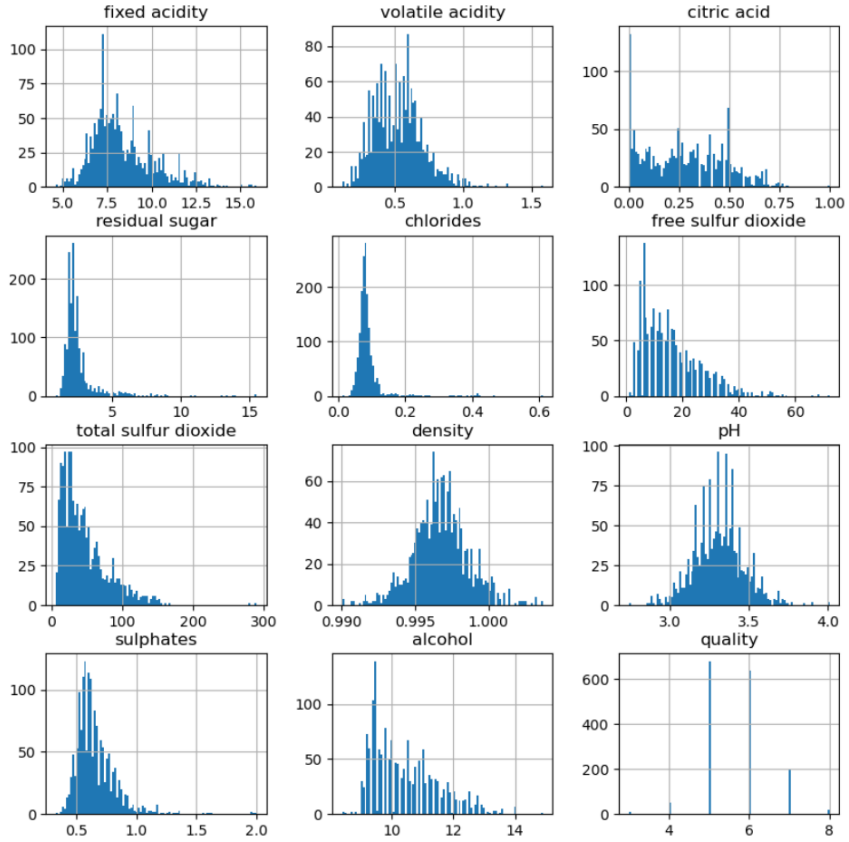
[Quality – Volatile acidity]



[Quality – citric acid]

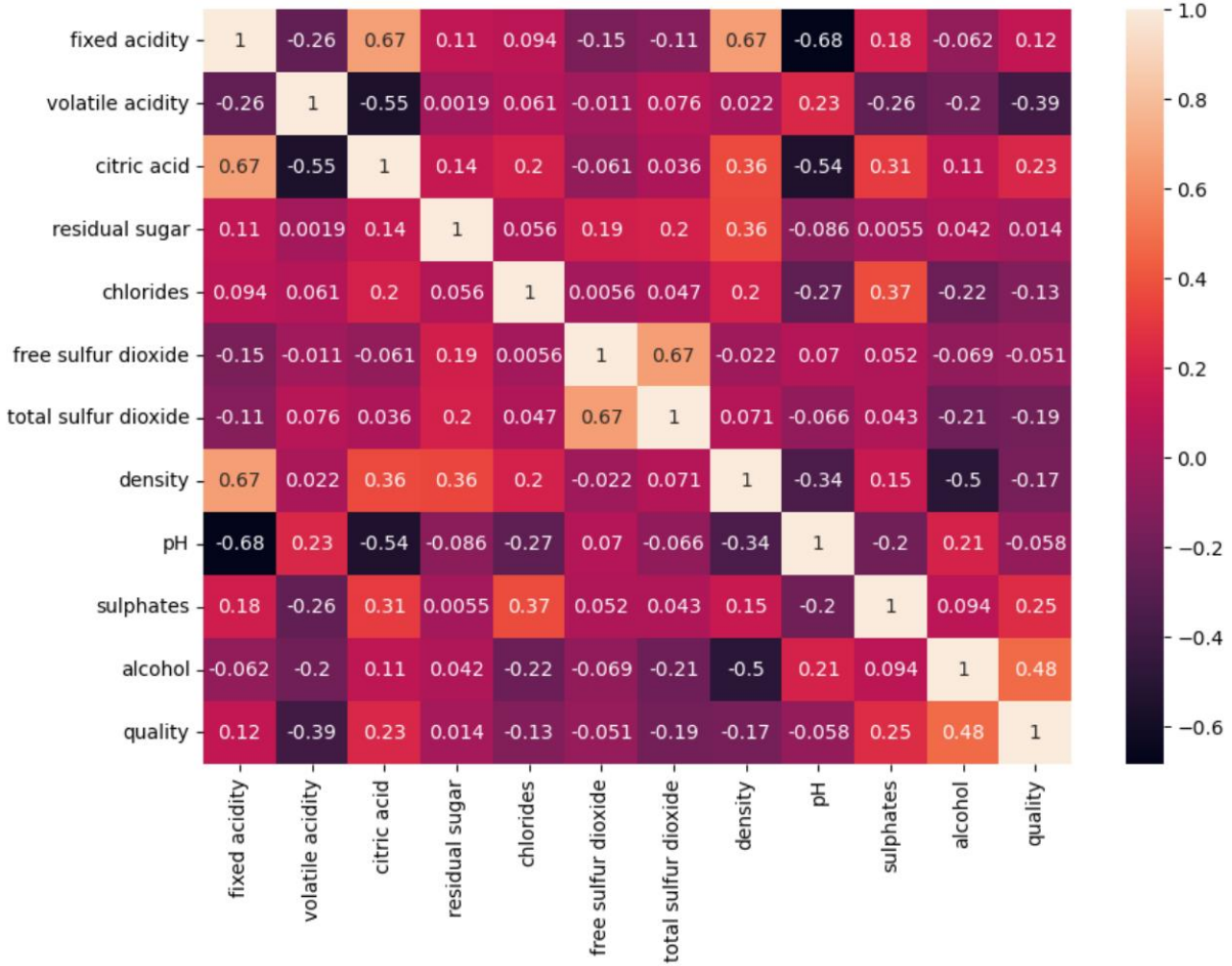


2. 데이터 수집 및 데이터분석(EDA 방법)



X 항목에 따른 분포를 한눈에 볼 수 있는 표

2. 데이터 수집 및 데이터분석(EDA 방법)



data.corr()["quality"].sort_values()

```
volatile acidity        -0.390558
total sulfur dioxide    -0.185100
density                 -0.174919
chlorides               -0.128907
pH                      -0.057731
free sulfur dioxide     -0.050656
residual sugar           0.013732
fixed acidity            0.124052
citric acid              0.226373
sulphates                0.251397
alcohol                  0.476166
quality                  1.000000
Name: quality, dtype: float64
```

3 . 머신 러닝 모델 & 모델 평가

```
train_input, test_input, train_target, test_target = train_test_split(
    X, Y, test_size=0.2, random_state=42)
```

```
print(X.shape, train_input.shape, test_input.shape)
```

```
(1599, 11) (1279, 11) (320, 11)
```

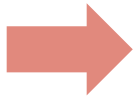
훈련 값과 테스트 값을 분리하여 크기 확인

3 . 머신 러닝 모델 & 모델 평가

[로지스틱 회귀]

```
#Logistic Regression
lg = LogisticRegression()
lg.fit(train_input,train_target)
```

```
▾ LogisticRegression
LogisticRegression()
```



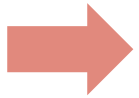
```
l_pred = lg.predict(test_input)
l_acc = accuracy_score(l_pred,test_target)
print("테스트 정확도 : ",l_acc*100)
```

```
테스트 정확도 :  86.5625
```

[결정 트리]

```
#결정트리 모델
tree = DecisionTreeClassifier()
tree.fit(train_input,train_target)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```



```
t_pred = tree.predict(test_input)
t_acc = accuracy_score(t_pred,test_target)
print("테스트 정확도 : ",t_acc*100)
```

```
테스트 정확도 :  88.125
```

3 . 머신 러닝 모델 & 모델 평가

[랜덤 포레스트]

```
#랜덤포레스트(분류)
forest = RandomForestClassifier(n_jobs=-1)
forest.fit(train_input,train_target)
```

```
RandomForestClassifier(n_jobs=-1)
```

```
f_pred = forest.predict(test_input)
f_acc = accuracy_score(f_pred,test_target)
print("테스트 정확도 : ",f_acc*100)
```

```
테스트 정확도 :  91.875
```

3가지의 방법을 사용해본 후 가장 테스트
정확률이 높은 랜덤 포레스트 방법을 채택하였습니다.

3 . 머신 러닝 모델 & 모델 평가

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, mean_squared_error, r2_score
import matplotlib.pyplot as plt
import seaborn as sns
```

```
X = data.drop('quality', axis=1)
y = data['quality'].apply(lambda y_value: 1 if y_value >= 6.5 else 0)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

X와 Y의 데이터를 분리 한 후
Y의 데이터 값은 6.50이상은 1 나머지 값은 0으로 변경하였습니다.

3 . 머신 러닝 모델 & 모델 평가

```
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [5, 10, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

```
Best parameters: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 300}
```

```
rf = RandomForestClassifier()
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=5, n_jobs=-1, verbose=2)
grid_search.fit(X_train, y_train)
print('Best parameters:', grid_search.best_params_)
rf = grid_search.best_estimator_
```

랜덤 포레스트의 정확도를 높이기 위해
가장 좋은 파라미터를 찾은 후 적용하였습니다

3 . 머신 러닝 모델 & 모델 평가

```
# Evaluate model
train_pred = rf.predict(X_train)
train_acc = accuracy_score(train_pred, y_train)
print('Train accuracy:', train_acc)

test_pred = rf.predict(X_test)
test_acc = accuracy_score(test_pred, y_test)
print('Test accuracy:', test_acc)
```

```
Train accuracy: 0.9953088350273651
Test accuracy: 0.90625
```

훈련 결과가 99% 테스트 결과가 90%에 가까워지며
기존의 모델보다 정확한 측정이 가능해 졌습니다.

3 . 머신 러닝 모델 & 모델 평가

```
input_data = (7.3,0.65,0.0,1.2,0.065,15.0,21.0,0.9946,3.39,0.47,10.0)
#input_data = (7.5,0.5,0.36,6.1,0.071,17.0,102.0,0.9978,3.35,0.8,10.5)

input_data_as_numpy_array = np.asarray(input_data)

input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = rf.predict(input_data_reshaped)

print(prediction)

if prediction[0] == 1:
    print("Good Quality Wine!")
else:
    print("Bad Quality Wine")
```

```
[1]
Good Quality Wine!
```

훈련한 모델이 제대로 측정이 되는지 확인하기 위하여
입력 값을 넣어 결과를 예측하였습니다.

3 . 머신 러닝 모델 & 모델 평가

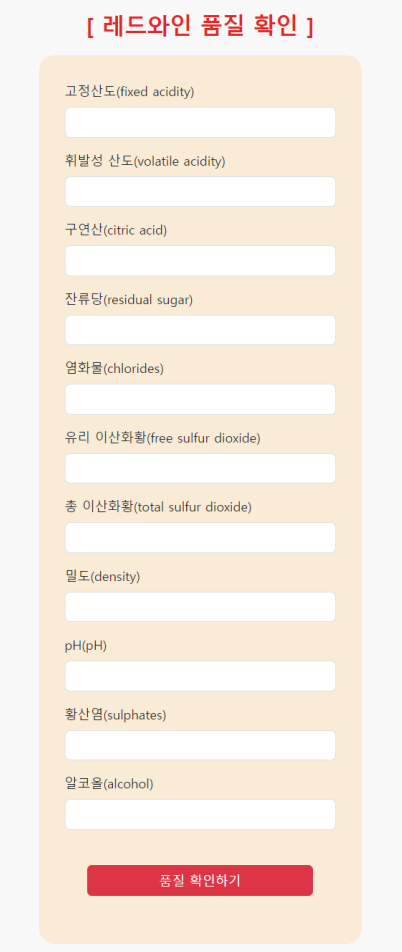
```python
import pickle

# 모델 저장
rf.fit(X_train, y_train)
with open('wine.pickle', 'wb') as f:
    pickle.dump(rf, f)
```

웹을 통해 배포를 하기위해
피클을 이용하여 모델을 저장하였습니다.

4 . 서비스(웹) html,css,Flask.. 사용

[Main.html]



[Good 퀄리티 값 입력]

[Bad 퀄리티 값 입력]

4 . 서비스(웹)

[Good 퀄리티]

와인의 품질

Good Quality

[Bad 퀄리티]

와인의 품질

Bad Quality

Result.html

사용한 모델에 관한 깃 주소

5. 느낀점

실제 현장에서 사용되는 CSV를 통해 모델 구현하는 과정을 진행해보니 책을 보면서 공부하는 것과 달리 어떤 모델이 정확도가 높은 지와 같은 부분들을 직접 적용해 보는 시간을 가졌습니다.

이번 과제를 통해서 머신 러닝에서 어떤 부분이 부족한지 알 수 있는 시간을 가졌으며 부족한 부분을 더 보강하여 프로젝트에 임해야 겠다고 다짐을 하게 되었습니다.

또한 웹을 통해 모델을 확인하는 과정과 직접 모델을 구현하는 과정을 진행하면서 전보다 머신 러닝에 대한 관심을 가지게 되었고 많은 현장 CSV를 통해 머신 러닝을 훈련시키는 방법을 공부하여 프로젝트에서 좋은 결과를 만들 수 있도록 하겠습니다.
감사합니다!

감사합니다