



감귤 가격 예측

AI 모델, 웹 어플리케이션 개발 중간발표



팀장 김정석
팀원 최현나
팀원 임유정

Index

1. 데이터 수집, 모델 생성 🍃
2. 모델 성능 평가 🍃
3. 웹 어플리케이션 개발 🍃
4. 추가 개발, 보완점 🍃



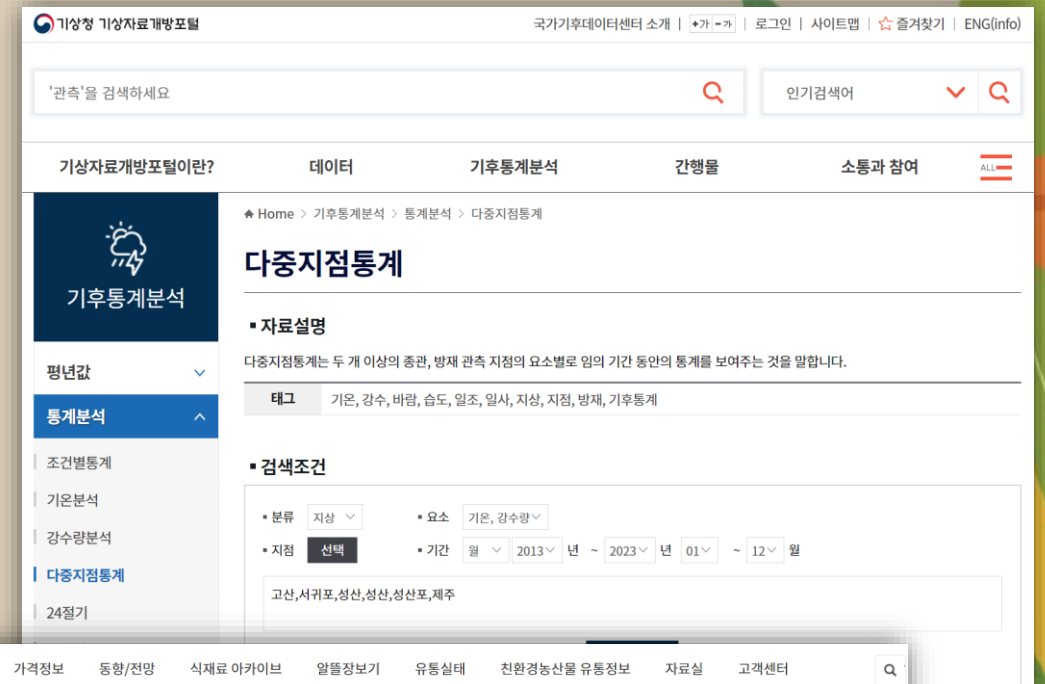
데이터 수집

기상데이터

- 기상청 기상자료 개방 포털
- <https://data.kma.go.kr/>

감귤 가격 데이터

- 농산물 유통정보 KAMIS
- <https://www.kamis.or.kr/>



기상청 기상자료 개방포털

국가기후데이터센터 소개 | [*가](#) | 로그인 | 사이트맵 | 즐겨찾기 | ENG(info)

'관측'을 검색하세요

인기검색어

기상자료개방포털이란? 데이터 기후통계분석 간행물 소통과 참여

Home > 기후통계분석 > 통계분석 > 다중지점통계

다중지점통계

■ 자료설명

다중지점통계는 두 개 이상의 종관, 방재 관측 지점의 요소별로 임의 기간 동안의 통계를 보여주는 것을 말합니다.

태그 기온, 강수, 바람, 습도, 일조, 일사, 지상, 지점, 방재, 기후통계

■ 검색조건

• 분류 지상 • 요소 기온, 강수량
• 지점 선택 • 기간 월 2013 ~ 2023 년 01 ~ 12 월

고산,서귀포,성산,성산,성산포,제주



aT KAMIS 가격정보 동향/전망 식재료 아카이브 알뜰장보기 유통실태 친환경농산물 유통정보 자료실 고객센터

가격정보

대형마트, 전통시장 등에서 소비자에게 판매하는 가격

기간별

일간가격 반순별가격 순별가격 월간가격 연간가격

기간	지역	부류	품목	품종	등급
시작일: 2022-04-15	전체	식량작물	포도	전체	전체
2022년 4월	서울	채소류	감귤	노지	상품
1 2	부산	특용작물	단감	시설	중품
3 4 5 6 7 8 9	대구	과일류	바나나	감귤	M과
10 11 12 13 14 15 16	인천	축산물	참다래		
17 18 19 20 21 22 23	광주	수산물	파인애플		
24 25 26 27 28 29 30					
종료일: 2023-04-06					

kg 단위환산 (동그라미 체크)

조회하기

데이터 전처리

1. 강수량 결측치

Out[4]:

	year	avgTemp	maxTemp	minTemp	rainFall
3764	2023-04-23	15.3	18.8	13.1	NaN
3765	2023-04-24	13.8	15.4	12.0	2.2
3766	2023-04-25	12.9	13.9	11.8	4.4
3767	2023-04-26	15.0	20.2	10.1	NaN
3768	2023-04-27	14.1	17.8	10.6	NaN



Out[5]:

	year	avgTemp	maxTemp	minTemp	rainFall
0	2013-01-01	7.2	10.7	3.5	0.0
1	2013-01-02	6.2	10.9	1.3	0.0
2	2013-01-03	2.3	5.5	0.1	0.0
3	2013-01-04	3.4	6.7	0.3	0.0
4	2013-01-05	5.1	8.8	2.5	0.0
...
3764	2023-04-23	15.3	18.8	13.1	0.0
3765	2023-04-24	13.8	15.4	12.0	2.2
3766	2023-04-25	12.9	13.9	11.8	4.4
3767	2023-04-26	15.0	20.2	10.1	0.0
3768	2023-04-27	14.1	17.8	10.6	0.0

3769 rows × 5 columns

Pandas fillna() -> 강수량 결측치 0으로 처리

데이터 전처리

2. 기상 데이터 결측치

감귤 수확 시기 : 보통 9월~이듬해 3월
가격 데이터와 기상데이터 간의 차이 존재



```
In [20]: df_merge2 = pd.merge(df1_new, df2)
         print(df_merge2)
```

	year	avgTemp	maxTemp	minTemp	rainFall	avgPrice
0	2013-01-02	6.2	10.9	1.3	0.0	2,434
1	2013-01-03	2.3	5.5	0.1	0.0	2,501
2	2013-01-04	3.4	6.7	0.3	0.0	2,483
3	2013-01-07	6.6	8.6	3.8	0.0	2,491
4	2013-01-08	7.9	10.8	5.1	0.0	2,491
...
1091	2023-03-23	17.4	19.3	16.5	1.9	7,144
1092	2023-03-24	14.7	16.8	12.6	11.4	7,036
1093	2023-03-27	13.0	16.9	10.1	0.0	6,941
1094	2023-03-28	13.9	18.2	9.4	0.0	6,941
1095	2023-03-29	14.2	18.1	11.0	0.0	7,356

[1096 rows x 6 columns]

```
In [21]: df_merge2.to_excel(excel_writer = 'c:/tangerine_price_prediction/innerjoin.xlsx')
```

pandas -> Inner join merge

선형회귀 모델

- 선형회귀(Linear Regression): 종속 변수 y 와 한 개 이상의 독립 변수 x 와의 선형 상관관계를 모델링하는 회귀 분석 기법

독립 변수 x : 평균 기온, 최대 기온, 최저 기온, 강수량(기상 데이터)
종속 변수 y : 감귤 가격

$$H(x_1, x_2, x_3, x_4) = x_1w_1 + x_2w_2 + x_3w_3 + x_4w_4 + b$$

선형회귀 모델

```
import tensorflow.compat.v1 as tf

import numpy as np

from pandas.io.parsers import read_csv

tf.disable_v2_behavior()

model = tf.global_variables_initializer();

data = read_csv('C:\\tangerine_price_prediction\\data\\innerjoin.csv', sep=',')

xy = np.array(data, dtype=np.float32)
```

- 입력 변수(기상 데이터 4개) 및 출력 변수(가격) 설정
- 가중치(W)와 편향(b)을 위한 변수 정의

- TensorFlow 및 NumPy 라이브러리 import
- TensorFlow 변수를 초기화
- CSV 파일을 읽어 데이터 로드
- NumPy 배열로 데이터 변환

```
x_data = xy[:, 1:-1]

y_data = xy[:, [-1]]

X = tf.placeholder(tf.float32, shape=[None, 4])

Y = tf.placeholder(tf.float32, shape=[None, 1])

W = tf.Variable(tf.random_normal([4, 1]), name="weight")

b = tf.Variable(tf.random_normal([1]), name="bias")
```

선형회귀 모델

```
hypothesis = tf.matmul(X, W) + b

cost = tf.reduce_mean(tf.square(hypothesis - Y))

optimizer = tf.train.GradientDescentOptimizer(learning_rate=0.000005)

train = optimizer.minimize(cost)
```

```
for step in range(100001)

    cost_, hypo_, _ = sess.run([cost, hypothesis, train], feed_dict={X: x_data, Y: y_data})

    if step % 500 == 0:

        print("#", step, " 손실 비용: ", cost_)

        print("- 감귤 가격: ", hypo_[0])

saver = tf.train.Saver()

save_path = saver.save(sess, "./saved.cpkt")

print('학습된 모델을 저장했습니다.')
```

- 가설(hypothesis) 정의
- 비용함수(cost function) 정의
- 최적화 함수와 훈련 오퍼레이션 정의
(learning rate = 0.000005로 설정)
- 학습 수행 및 학습 저장
(100001번 학습 수행 및
매 500번마다 학습 결과
출력 함으로써 확인)

성능평가 🍊

- MAE(Mean Absolute Error): 평균 절대 잔차

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| = \frac{1}{n} \sum_{i=1}^n |e_i|$$

- MSE(Mean Squared Error): 평균 제곱 오차

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

- RMSE(Root Mean Squared Error): 평균 제곱근 오차

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$



성능평가

- 전체 데이터 Suffle
- 훈련 데이터와 테스트 데이터를 8:2 비율로 나눈 후 학습 진행한 결과

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

train_indices = np.arange(X_train.shape[0])
np.random.shuffle(train_indices)
X_train = X_train[train_indices]
y_train = y_train[train_indices]

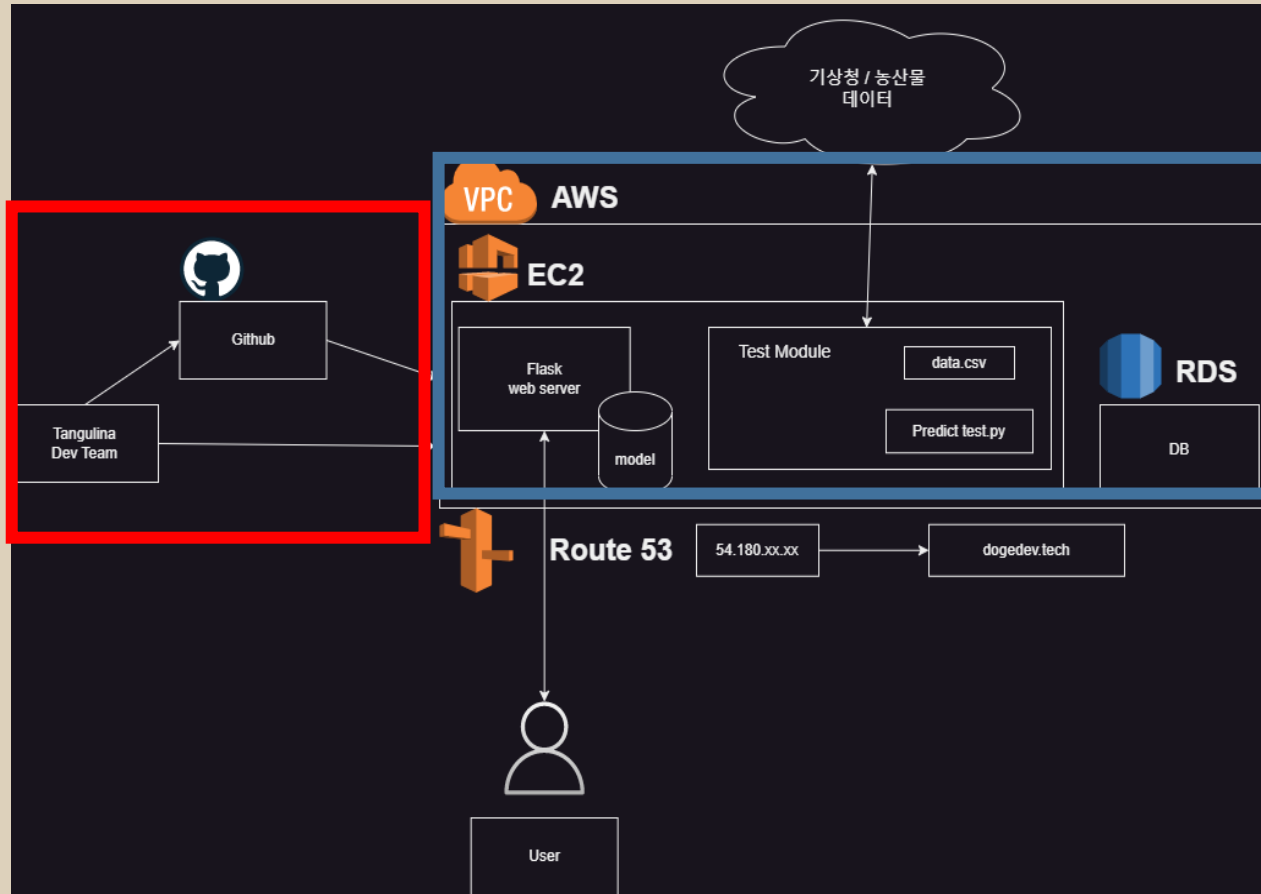
test_indices = np.arange(X_test.shape[0])
np.random.shuffle(test_indices)
X_test = X_test[test_indices]
y_test = y_test[test_indices]
```



```
MAE: 952.5477083333334
MSE: 1351785.7066666668
RMSE: 1162.6631957134734
```

```
mae = np.sum(np.abs(test_y_data - sess.run(hypothesis, feed_dict={X: test_x_data}))) / len(test_y_data)
mse = np.sum(np.square(test_y_data - sess.run(hypothesis, feed_dict={X: test_x_data}))) / len(test_y_data)
rmse = np.sqrt(mse)
```

서버 아키텍처 구조



개발

VSC : Github

SSH 연결

배포

AWS



EC2 : Ubuntu 22.04 LTS



RDS : MariaDB



Route53 : DNS 설정

- **/predict** : Method : post

```
11 # 플레이스 홀더를 설정합니다.
12 X = tf.placeholder(tf.float32, shape=[None, 4])
13 Y = tf.placeholder(tf.float32, shape=[None, 1])
14 W = tf.Variable(tf.random_normal([4, 1]), name="weight")
15 b = tf.Variable(tf.random_normal([1]), name="bias")
16
17 # 가설을 설정합니다.
18 hypothesis = tf.matmul(X, W) + b
19
20 # 저장된 모델을 불러오는 객체를 선언합니다.
21 saver = tf.train.Saver()
22 model = tf.global_variables_initializer()
23
24 # 세션 객체를 생성합니다.
25 sess = tf.Session()
26 sess.run(model)
27
28 # 저장된 모델을 세션에 적용합니다.
29 save_path = "./model/saved.cpkt"
30 saver.restore(sess, save_path)
31
```

```
@app.route('/predict', methods=['POST'])
def predict():
    # 파라미터를 전달 받습니다.
    avg_temp = float(request.form['avg_temp'])
    min_temp = float(request.form['min_temp'])
    max_temp = float(request.form['max_temp'])
    rain_fall = float(request.form['rain_fall'])
    # 감귤 가격 변수를 선언합니다.
    price = 0
    # 입력된 파라미터를 배열 형태로 준비합니다.
    data = ((avg_temp, min_temp, max_temp, rain_fall), (0, 0, 0, 0))
    arr = np.array(data, dtype=np.float32)
    # 입력 값을 토대로 예측 값을 찾아냅니다.
    x_data = arr[0:4]
    dict = sess.run(hypothesis, feed_dict={X: x_data})
    # 결과 감귤 가격을 저장합니다.
    price = dict[0]
    return render_template('index.html', price=price)
```

Sever.py



- **/data** : DB에서 월별 가격정보 조회

```
@app.route('/data')
def data():
    connection = pymysql.connect(**DB_CONFIG)

    cursor = connection.cursor()
    cursor.execute('SELECT DATE_FORMAT(weather_data.date, "%Y%m") AS month, AVG(citrus_price_data.avgPrice) AS avg_monthly_price '
                  'FROM citrus_price_data '
                  'JOIN weather_data ON citrus_price_data.weather_id = weather_data.weather_id '
                  'GROUP BY month '
                  'ORDER BY month')
    results = cursor.fetchall()

    data = {
        'months': [row[0] for row in results],
        'avgPrices': [row[1] for row in results]
    }

    connection.close()
    return jsonify(months=data['months'], avgPrices=data['avgPrices'])
```

Chart.js



데이터 시각화

- Chart.js 자바스크립트 라이브러리 사용
- Fetch로 데이터 불러오기

```
window.onload = function() {  
  var ctx = document.getElementById('myChart').getContext('2d');  
  var months = [];  
  var avgPrices = [];  
  
  fetch('/data')  
    .then(function(response) {  
      return response.json();  
    })  
    .then(function(data) {  
      months = data.months;  
      avgPrices = data.avgPrices;  
    })  
  
  var myChart = new Chart(ctx, {  
    type: 'line',  
    data: {  
      labels: months,  
      datasets: [{  
        label: '월 평균 감귤 가격',  
        data: avgPrices,  
        backgroundColor: 'rgba(255, 99, 132, 0.2)',  
        borderColor: 'rgba(255, 99, 132, 1)',  
        borderWidth: 1  
      }]  
    },  
    options: {  
      scales: {  
        yAxes: [{  
          ticks: {  
            beginAtZero: true  
          }  
        }]  
      }  
    }  
  });  
}
```



- ~/.ssh/config

```
Host aws2
    HostName 54.180.53.50
    User ubuntu
    IdentityFile ~/.ssh/ec2-mykey.pem
```

```
ubuntu@ip-172-31-36-224: ~$ ssh aws2
* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

System information as of 2023. 05. 19. (금) 11:10:06 KST

System load:  0.080078125      Processes:           105
Usage of /:   42.2% of 28.89GB Users logged in:      1
Memory usage: 44%             IPv4 address for eth0: 172.31.36.224
Swap usage:   0%

* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

  https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

*** System restart required ***
Last login: Thu May 18 10:37:08 2023 from 121.189.108.88
ubuntu@ip-172-31-36-224:~$ ls
tangerine_price_prediction
ubuntu@ip-172-31-36-224:~$
```



AWS 배포

- **DB 객체 분리**: 배포시 DB정보 유출 방지를 위해 config.py 저장
- **AWS EC2 인바운드 설정** : 특정 IP에서만 접근 가능

```
config.py > ...  
1 DB_CONFIG = {  
2     'host': 'localhost',  
3     'user': 'root',  
4     'password': 'root',  
5     'db': 'tangerine_market_info'  
6 }  
7
```

EC2 > 보안 그룹 > sg-0f12f2e051ce8ab61 - launch-wizard-2 > 인바운드 규칙 편집

인바운드 규칙 편집 정보

인바운드 규칙은 인스턴스에 도달하도록 허용된 수신 트래픽을 제어합니다.

보안 그룹 규칙 ID	유형 <small>정보</small>	프로토콜 <small>정보</small>	포트 범위 <small>정보</small>	소스 <small>정보</small>
sgr-0995dbb97cb6887b8	HTTPS	TCP	443	사용자 지정 0.0.0.0/0 X
sgr-0e348dfdb7b99ae8e	HTTP	TCP	80	사용자 지정 0.0.0.0/0 X
sgr-047e20866238da070	사용자 지정 TCP	TCP	5000	사용자 지정 0.0.0.0/0 X
sgr-0b24762ebee4204ea	SSH	TCP	22	내 IP 61.42.88.226/32 X

규칙 추가

- 포트포워딩

Iptable : port 5000 -> port 80

: 80번 포트로 들어오는 요청 5000번으로 리디렉션

```
sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 5000
```





- <http://dogedev.tech>



감귤 가격예측

평균온도

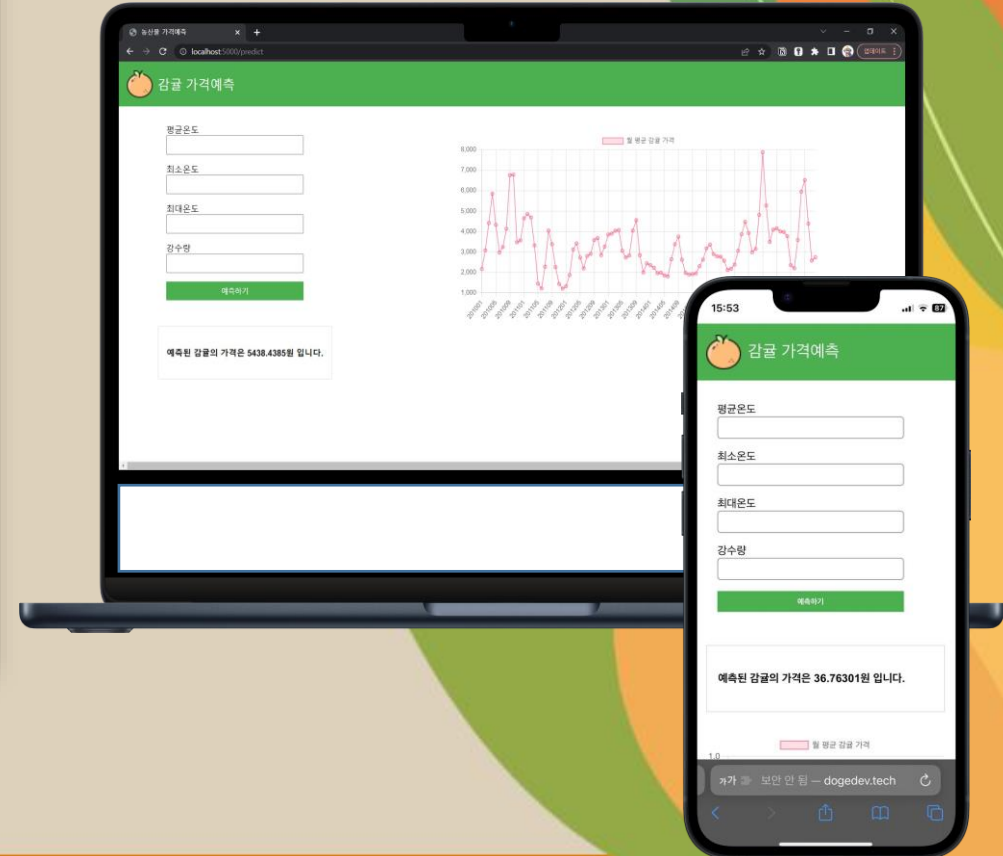
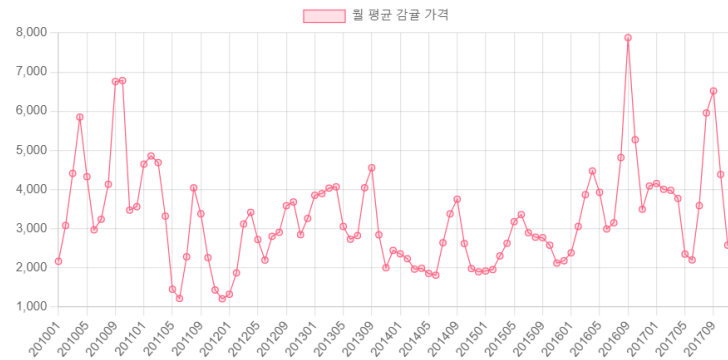
최소온도

최대온도

강수량

예측하기

예측된 감귤의 가격은 5438.4385원 입니다.



추가개발 / 보완사항

1. 입력 값 검증 로직 추가

- 클라이언트 + 서버

2. 날씨 추가정보 차트 추가

- 날씨 추세 조회

3. 모델 성능 개선

- 이슈 사항 해결





Thank
YOU