



딥러닝 기반의 수산물 수입가격 예측 모형 도출

Seafood import price prediction model based on deep learning

Work Team Name & Members

W_W(윈터위너)

곽원일(팀장), 송민아, 김유환, 김지훈

Work Schedule

2021.08.24 주제선정, 데이터 수집

2021.08.30 데이터 전처리, 데이터 분석

2021.09.03 모델 설계 및 개발

2021.09.13 모델 평가 및 검증

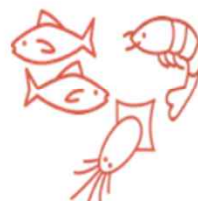
2021.09.24 장고 웹프레임워크 구현

Work Rule

Train Data : 2016년 ~ 2020년 수산물 수입데이터

Test Data : 2021년 1월 ~ 6월 예측

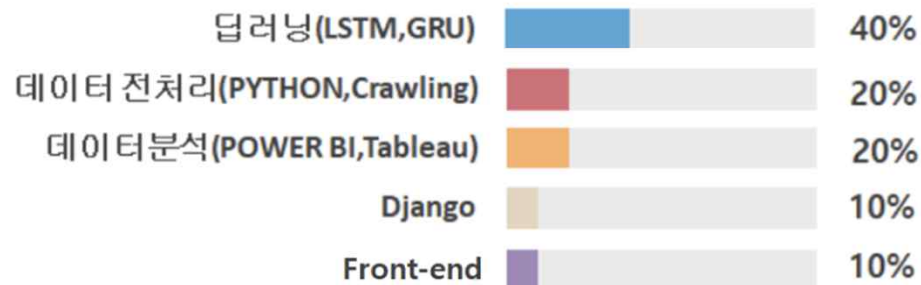
데이터 전처리 방법, 활용 알고리즘 설명



수산물(연어,오징어,흰다리새우) 주별 평균단가 예측 서비스

빅데이터 생태계 조성을 통한 해양수산 뉴딜 정책 실현 및
수산업 이해 관계자의 경영계획 수립 기반 구축

Skills



목 차

01

프로젝트 소개

- 팀 소개 및 개발 환경
- 대회 소개
- RAW DATA
- 도메인 조사
- 분석 방향

04

모델 성능 향상

- 하이퍼파라미터 조정
- LSTM과 GRU 앙상블

02

데이터 전처리

- 이상치 보정
- 유효한 데이터 선정
- 변수 생성

05

시뮬레이터 구현

- UI 구현
- 기능 구현

03

모델 탐색 및 검증

- 모델 탐색
- 모델 검증

06

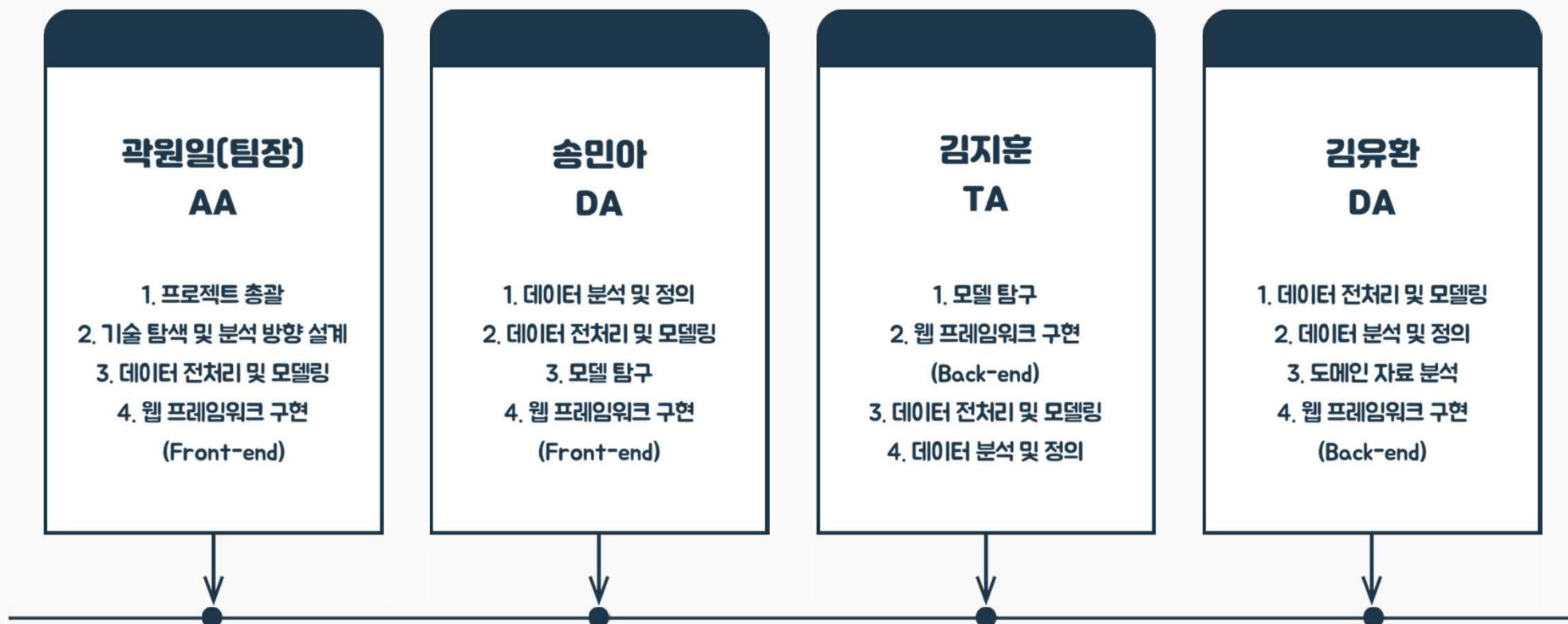
결론 및 느낀점

01. 프로젝트 소개

1. 팀 소개
2. 개발 환경
3. 대회 소개
4. RAW DATA
5. 도메인 조사
6. 분석 방향

01. 프로젝트 소개

1) 팀 소개



01. 프로젝트 소개

2) 개발 환경

개발



python
v 3.8.11



TensorFlow
v 2.6.0

LSTM

GRU



시각화 및 협업도구



Power BI



GitHub

시뮬레이터 구현

django

v 3.2.7



Bootstrap v.5



Chart.js



01. 프로젝트 소개

3) 대회 소개

주관	한국해양수산개발원
주제	수산물 수입가격 예측을 통한 최적의 가격예측 모형 도출
목적	빅데이터 생태계 조성을 통한 해양수산 뉴딜 정책 실현 및 수산업 이해관계자의 경영계획 수립 기반 구축
데이터	2015년 12월 28일 ~ 2020년 12월 28일 약 5년간 수산물 수입평균단가 데이터
예측 시점	2021년 1월 4일 ~ 2021년 6월 28일
예측 품목	연어, 오징어, 흰다리새우
평가 척도	RMSE

✓ 주어진 수산물 데이터를 분석하여 수산물의 수입가격을 예측하고,
최적의 수산물 수입가격 예측 모형 제시

01. 프로젝트 소개

4) RAW DATA

< 수산물 수입평균단가 데이터 >

	REG_DATE	P_TYPE	CTRY_1	CTRY_2	P_PURPOSE	CATEGORY_1	CATEGORY_2	P_NAME	P_IMPORT_TYPE	P_PRICE
0	2015-12-28	수산물	아르헨티나	아르헨티나	판매용	갑각류	새우	아르헨티나붉은새우	냉동	7.48
1	2015-12-28	수산물	바레인	바레인	판매용	갑각류	게	꽃게	냉동	2.92
2	2015-12-28	수산물	바레인	바레인	판매용	갑각류	게	꽃게	냉동,절단	3.36
3	2015-12-28	수산물	칠레	칠레	판매용	패류 멍게류	해삼	해삼	건조,자숙	18.26
4	2015-12-28	수산물	중국	중국	판매용	어류	서대 박대 페루다	서대	냉동	4.79
5	2015-12-28	수산물	중국	중국	판매용	어류	복어	은밀복	냉동	2.17
6	2015-12-28	수산물	중국	중국	판매용	어류	옥돔	옥돔	냉동	5.15
7	2015-12-28	수산물	중국	중국	판매용	어류	복어	까치복	냉동	4.27
8	2015-12-28	수산물	중국	중국	판매용	갑각류	새우	흰다리새우	냉동,살,자숙	7.20
9	2015-12-28	수산물	중국	중국	판매용	갑각류	새우	흰다리새우	냉동,살	6.97
10	2015-12-28	수산물	중국	중국	판매용	패류 멍게류	바지락	바지락	냉장,살	4.70

< 테이블 정의서 >

컬럼ID	컬럼명
reg_date	기준일
p_type	제품구분
ctry_1	제조국
ctry_2	수출국
p_purpose	수입용도
category_1	중분류명 카테고리
category_2	어종
p_name	상세어종
p_import_type	수입형태
p_price	평균단가(\$)

2015년 12월 28일 ~ 2020년 12월 28일까지의

시계열적 데이터

→ 총 51,552개 데이터

10개의 변수

5) 도메인 조사

- 수산물 수입단가에 대한 전반적인 지식과 동향 습득

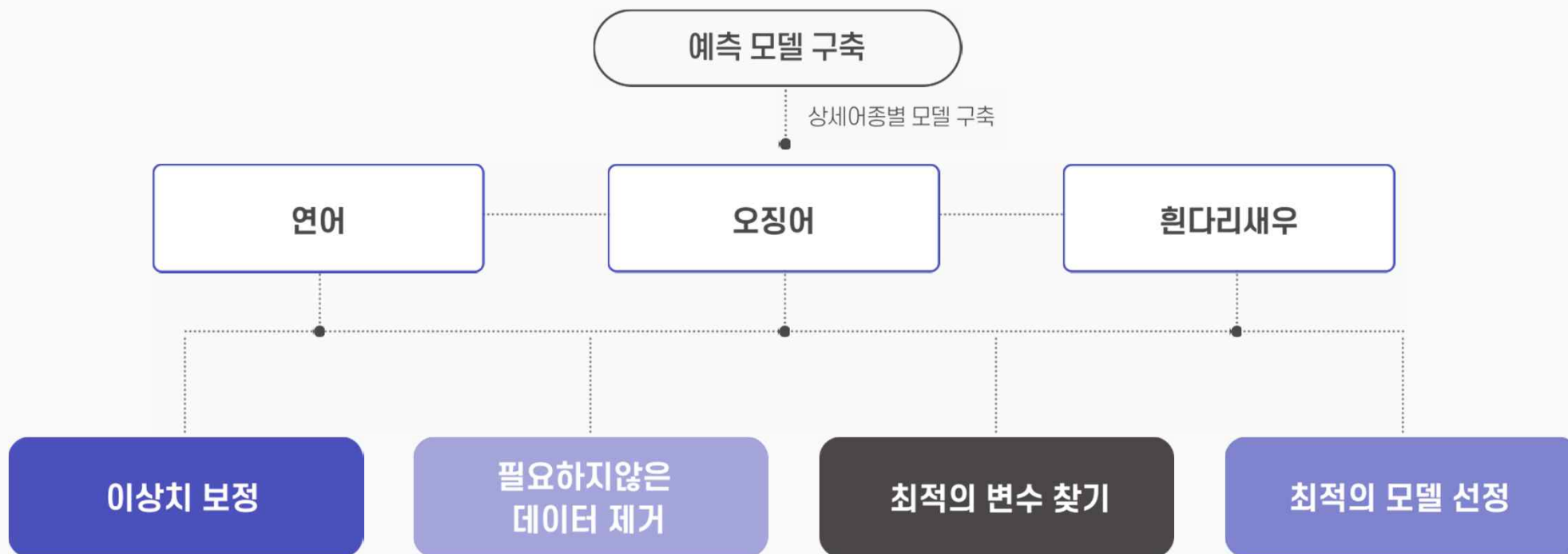
도메인 조사 결과 수입단가는 이상치가 생기는 경우가 빈번

→ 데이터 전처리과정에서 **이상치를 어떻게 처리할 것인지**가 핵심



6) 분석 방향

상세어종별(연어, 오징어, 흰다리새우)로 나누어 데이터 분석 및 모델 구축



02. 데이터 전처리

예측하고자 하는 시점의 평균단가에 무엇이 영향을
미칠 것인지를 중심으로 데이터를 분석 및 유효한 데이터로 정제

1. 데이터 전처리
2. 이상치 보정 및 유효한 데이터 선정
3. 파생변수

02. 데이터 전처리

1) 데이터 전처리 - 상세어종별 분류

< 주어진 수산물 수입평균단가 데이터 >

	REQ_DATE	P_TYPE	CTRY_1	CTRY_2	P_PURPOSE	CATEGORY_1	CATEGORY_2	P_NAME	P_IMPORT_TYPE	P_PRICE
0	2015-12-28	수산물	아르헨티나	아르헨티나	판매용	갑각류	새우	아르헨티나붉은새우	냉동	7.48
1	2015-12-28	수산물	바레인	바레인	판매용	갑각류	게	꽃게	냉동,절단	2.92
2	2015-12-28	수산물	바레인	바레인	판매용	갑각류	게	꽃게	냉동,절단	3.36
3	2015-12-28	수산물	칠레	칠레	판매용	패류 멍게류	해삼	해삼	건조,자숙	18.26
4	2015-12-28	수산물	중국	중국	판매용	어류	서대 박대 폐목	서대	냉동	4.79
5	2015-12-28	수산물	중국	중국	판매용	어류	복어	은밀복	냉동	2.17
6	2015-12-28	수산물	중국	중국	판매용	어류	옥돔	옥돔	냉동	5.15
7	2015-12-28	수산물	중국	중국	판매용	어류	복어	까치복	냉동	4.27
8	2015-12-28	수산물	중국	중국	판매용	갑각류	새우	힌다리새우	냉동,살,자숙	7.20
9	2015-12-28	수산물	중국	중국	판매용	갑각류	새우	힌다리새우	냉동,살	6.97
10	2015-12-28	수산물	중국	중국	판매용	패류 멍게류	바지락	바지락	냉장,살	4.70

< 상세어종 데이터 분할 코드 >

```
df = pd.read_csv('RAW_DATA/수산물_통합데이터.csv')

# 컬럼값과 조건을 비교하여 그 결과를 새로운 변수에 할당
is_shrimp = df['P_NAME'] == '힌다리새우'
# 조건을 충족하는 데이터를 필터링하여 새로운 변수에 저장
shrimp = df[is_shrimp]

# print(shrimp)

shrimp = shrimp.drop('Unnamed: 0', axis=1)
shrimp.reset_index(drop=True, inplace=True)

# print(shrimp)
shrimp.to_csv('RAW_DATA/힌다리새우_RAW_데이터.csv')
```

연어

	REQ_DATE	P_TYPE	CTRY_1	CTRY_2	P_PURPOSE	CATEGORY_1	CATEGORY_2	P_NAME	P_IMPORT_TYPE	P_PRICE
0	2015-12-28	수산물	노르웨이	노르웨이	판매용	어류	연어	연어	냉장, 필렛(F)	12.94
1	2015-12-28	수산물	노르웨이	노르웨이	판매용	어류	연어	연어	냉장, 포장통김, 활생(F)	19.15
2	2015-12-28	수산물	노르웨이	노르웨이	판매용	어류	연어	연어	냉장, 통통(F), 활생	12.08
3	2015-12-28	수산물	노르웨이	노르웨이	자사제품제조용	어류	연어	연어	냉장, 필렛(F)	13.43
4	2015-12-28	수산물	노르웨이	노르웨이	판매용	어류	연어	연어	냉장	8.84
5	2016-01-04	수산물	캐나다	캐나다	판매용	어류	연어	연어	냉장	9.20

오징어

	REQ_DATE	P_TYPE	CTRY_1	CTRY_2	P_PURPOSE	CATEGORY_1	CATEGORY_2	P_NAME	P_IMPORT_TYPE	P_PRICE
0	2015-12-28	수산물	대만	대만	판매용	연체류 해물오류	오징어	오징어	냉동, 통체	1.99
1	2015-12-28	수산물	중국	중국	판매용	연체류 해물오류	오징어	오징어	냉동, 통체	0.79
2	2015-12-28	수산물	페루	페루	판매용	연체류 해물오류	오징어	오징어	냉동, 다리	1.23
3	2015-12-28	수산물	페루	페루	판매용	연체류 해물오류	오징어	오징어	냉동, 통체, 자숙	5.48
4	2015-12-28	수산물	칠레	칠레	판매용	연체류 해물오류	오징어	오징어	냉동, 다리	0.97
5	2015-12-28	수산물	칠레	칠레	판매용	연체류 해물오류	오징어	오징어	냉동, 지느러미	0.76

힌다리새우

	REQ_DATE	P_TYPE	CTRY_1	CTRY_2	P_PURPOSE	CATEGORY_1	CATEGORY_2	P_NAME	P_IMPORT_TYPE	P_PRICE
0	2015-12-28	수산물	중국	중국	판매용	갑각류	새우	힌다리새우	냉동, 살, 자숙	7.20
1	2015-12-28	수산물	중국	중국	판매용	갑각류	새우	힌다리새우	냉동, 살	6.97
2	2015-12-28	수산물	페루	페루	판매용	갑각류	새우	힌다리새우	냉동	6.10
3	2015-12-28	수산물	태국	태국	판매용	갑각류	새우	힌다리새우	냉동, 살, 자숙	13.38
4	2015-12-28	수산물	태국	태국	판매용	갑각류	새우	힌다리새우	냉동, 살	15.05
5	2015-12-28	수산물	사우디아라비아	사우디아라비아	판매용	갑각류	새우	힌다리새우	냉동	6.02


02. 데이터 전처리

1) 데이터 전처리 - 필요하지 않은 기본변수 제거

< 흰다리새우_RAW_데이터 >

	REG_DATE	P_TYPE	CTRY_1	CTRY_2	P_PURPOSE	CATEGORY_1	CATEGORY_2	P_NAME	P_IMPORT_TYPE	P_PRICE
0	2015-12-28	수산물	중국	중국	판매용	갑각류	새우	흰다리새우	냉동,살,자숙	7.20
1	2015-12-28	수산물	중국	중국	판매용	갑각류	새우	흰다리새우	냉동,살	6.97
2	2015-12-28	수산물	페루	페루	판매용	갑각류	새우	흰다리새우	냉동	6.10
3	2015-12-28	수산물	태국	태국	판매용	갑각류	새우	흰다리새우	냉동,살,자숙	13.38
4	2015-12-28	수산물	태국	태국	판매용	갑각류	새우	흰다리새우	냉동,살	15.05
5	2015-12-28	수산물	사우디아라비아	사우디아라비아	판매용	갑각류	새우	흰다리새우	냉동	6.02

 : 상세어종별 공통적인 부분으로 평균단가에 영향 끼치는 요인 x

 : 종류에 따라 평균단가에 영향 끼치는 요인 o

→ 연어, 오징어 또한 공통적으로 적용.

**상세어종별(연어, 오징어, 흰다리새우) 주별 평균단가를 예측하기 위해
주별 평균단가에 영향을 끼치는 변수를 4가지로 추림**

제조국(CTRY_1)

수출국(CTRY_2)

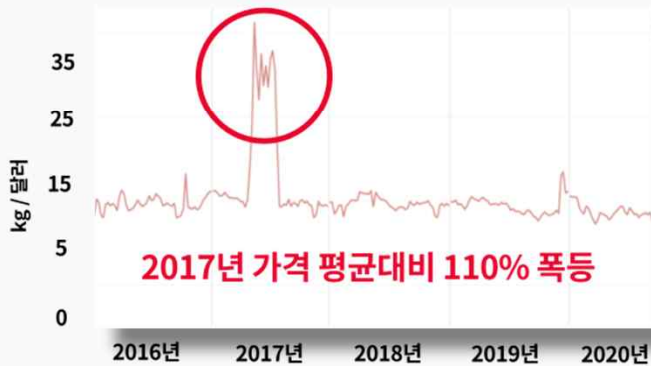
수입용도(P_PURPOSE)

수입형태(P_IMPORT_TYPE)

02. 데이터 전처리

1) 데이터 전처리 - 이상치 보정

< 연어 냉장, 필렛(F) 주별 평균단가 >



< 오징어 냉동, 지느러미 주별 평균단가 >

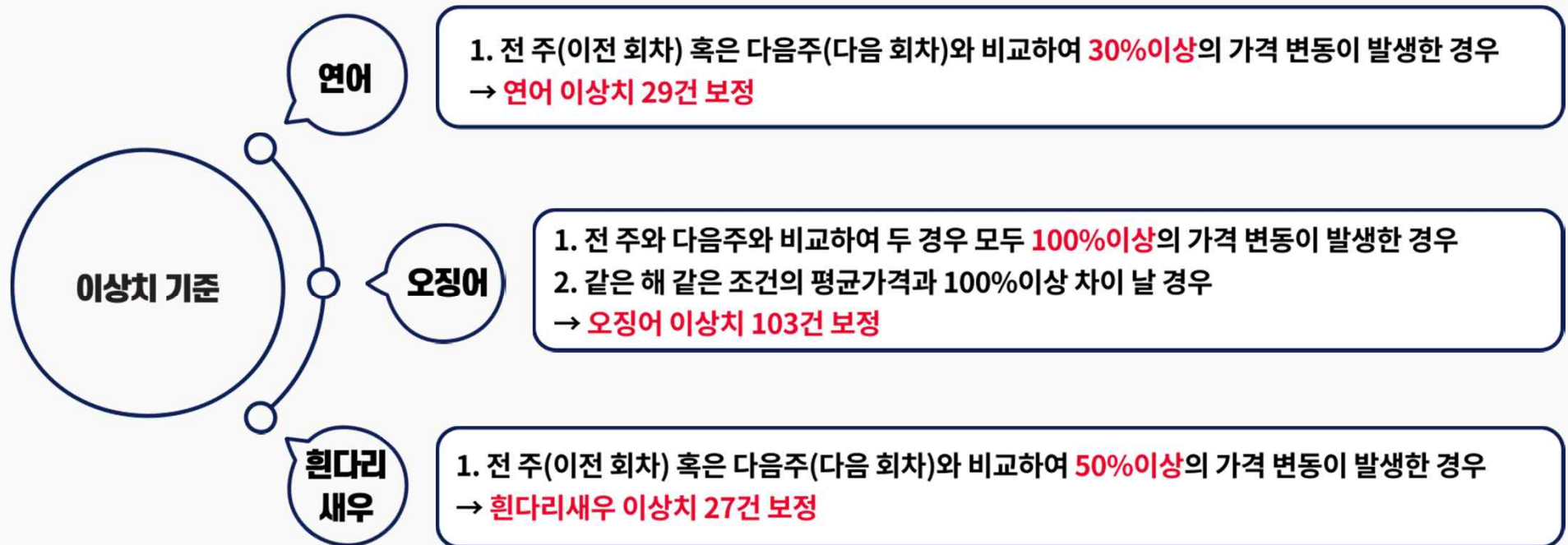


< 흰다리새우 냉동 주별 평균단가 >



이상치로 판단 → 안정적인 예측 모델을 위해 **평균단가로 데이터 보정**

1) 데이터 전처리 - 이상치 보정 결론



02. 데이터 전처리

2) 유효한 데이터 선정

예측시점에 수입되지 않을 데이터를 제거해 예측시점 수입단가의 예측 정확성을 높임

유효한 데이터 선정

- ☑ 가격 변동이 심해 예측에 악영향을 끼치는 과거 데이터
- ☑ 예측시점에 수입되지 않을 것이라 판단되는 데이터
- ☑ 고려하지 않아도 될 적은 비율이지만 주별 평균단가에 영향을 끼치는 데이터

< 특정 조건 데이터 제거 코드 >

```
df = pd.read_csv('/Users/minaworld/Desktop/흰다리새우_데이터_전처리.csv')
print(df)
df.drop('Unnamed: 0', axis=1, inplace=True)

idx = df[(df['CTRY_2'] == '태국') & (df['P_IMPORT_TYPE'] == '냉장')].index
# print(idx)

df.drop(idx, inplace=True)

df.reset_index(drop=True, inplace=True)

print(df)

df.to_csv('흰다리새우_유효한_데이터.csv')
```



연어 데이터 제거: 55건 오징어 데이터 제거: 40건 흰다리새우 데이터 제거: 291건

02. 데이터 전처리

3) 변수 생성

주별 평균단가 예측 **모델 학습에 필요한 변수 생성 과정 필요**

내부 데이터
+
외부 데이터



파생변수 생성



주별 평균단가와
상관계수 절대값 0.4 이상



최종 변수 선정

02. 데이터 전처리

3) 변수 생성(내부 데이터)

<대표적인 변수 생성 코드 예시>

```
import pandas as pd

df = pd.read_csv('RAW_DATA/원다리세우_데이터_최종셋.csv')
df = df.drop('Unnamed: 0', axis = 1)
df['REG_DATE'] = pd.to_datetime(df['REG_DATE'], format='%Y-%m-%d')
```

```
nation_mean = df.groupby(['REG_DATE', 'CTRY_2'])['P_PRICE'].mean()
n_mean = nation_mean.unstack()
```

국가별 평균단가 변수

CTRY_2	일본제사마	베트남	사우디아라비아	레소토	인도	인도네시아	중국	콜롬비아	태국	태완
REG_DATE										
2015-12-28	6.17	10.176000	6.02	6.28	NaN	NaN	NaN	NaN	15.475000	NaN
2016-01-04	7.19	8.347500	5.70	6.50	NaN	NaN	NaN	NaN	15.475000	NaN
2016-01-11	6.03	10.324000	NaN	8.09	NaN	NaN	7.08	NaN	16.230000	NaN
2016-01-18	6.10	10.014000	NaN	8.15	NaN	NaN	6.94	NaN	11.486667	NaN
2016-01-25	6.22	9.882000	6.50	8.05	10.06	6.26	6.86	NaN	12.577500	NaN
...
2020-11-30	6.80	10.063333	NaN	5.72	NaN	NaN	NaN	NaN	12.650000	5.99
2020-12-07	7.33	10.376667	NaN	6.00	NaN	NaN	NaN	NaN	11.370000	6.20
2020-12-14	6.06	10.153333	NaN	6.05	7.25	NaN	NaN	NaN	13.676667	6.17
2020-12-21	8.62	9.136667	NaN	6.04	12.50	NaN	NaN	NaN	12.650000	5.77
2020-12-28	5.91	11.395000	NaN	5.95	NaN	8.59	NaN	NaN	15.480000	6.14

국가별 카운트 변수

```
nation_count = df.groupby(['REG_DATE', 'CTRY_2'])['P_PRICE'].count()
n_count = nation_count.unstack()

for i in n_count.columns:
    n_count.rename(columns=(i+'C'), inplace=True)

n_count
# n_count.to_csv('RAW_DATA/원다리세우_데이터_최종셋.csv')
```

CTRY_2	일본제사마C	베트남C	사우디아라비아C	레소토C	인도C	인도네시아C	중국C	콜롬비아C	태국C	태완C
REG_DATE										
2015-12-28	1.0	6.0	1.0	1.0	NaN	NaN	1.0	NaN	3.0	1.0
2016-01-04	1.0	4.0	1.0	1.0	NaN	NaN	NaN	NaN	2.0	NaN
2016-01-11	1.0	5.0	NaN	1.0	NaN	NaN	1.0	NaN	3.0	NaN
2016-01-18	1.0	5.0	NaN	1.0	NaN	NaN	1.0	NaN	3.0	NaN
2016-01-25	1.0	5.0	1.0	1.0	1.0	1.0	1.0	NaN	4.0	NaN
...
2020-11-30	1.0	3.0	NaN	1.0	NaN	NaN	NaN	NaN	1.0	1.0
2020-12-07	1.0	3.0	NaN	1.0	NaN	NaN	NaN	NaN	4.0	1.0
2020-12-14	1.0	3.0	NaN	1.0	1.0	NaN	NaN	NaN	3.0	1.0
2020-12-21	1.0	3.0	NaN	1.0	1.0	NaN	NaN	NaN	3.0	1.0
2020-12-28	1.0	4.0	NaN	1.0	NaN	1.0	NaN	NaN	3.0	1.0

< 변수 종류 >

날짜별 평균단가 변수

국가별 평균단가 변수

국가별 카운트 변수

총 9가지 경우 중

수입용도별 평균단가 변수

수입용도별 카운트 변수

수입형태별 평균단가 변수

수입형태별 카운트 변수

국가 타입별 평균단가 변수

국가 타입별 카운트 변수

상세어종별 데이터 분석을 바탕으로
변수 생성 및 결측치는 0으로 처리

02. 데이터 전처리

3) 변수 생성(외부 데이터)

▶ 원달러 환율

< 활용 목적 >

1. 환율이 수입단가에 영향을 미칠 것이라 판단
2. 원달러 환율이 가장 대표적인 지표

날짜	통화명	환율
2015.12.28	미 달러화(USD)	1,171.00
2015.12.29	미 달러화(USD)	1,165.00
2015.12.30	미 달러화(USD)	1,167.70
2015.12.31	미 달러화(USD)	1,172.00
2016.01.04	미 달러화(USD)	1,172.00

(출처: 서울외국환중개소)

▶ 유가 두바이유 시세

< 활용 목적 >

1. 수출입 국가간 거리를 고려하여 유가가 수입단가에 영향을 미칠 것이라 판단
2. 두바이유가는 유가 중에서도 가장 대표적인 지표

날짜	종가	오픈	고가	저가	거래량	변동 %
2021년 06월 28일	71.54	71.54	71.54	71.54	-	-0.16%
2021년 06월 25일	71.65	71.65	71.65	71.65	-	0.08%
2021년 06월 24일	71.60	71.60	71.60	71.60	-	0.11%
2021년 06월 23일	71.52	71.52	71.52	71.52	-	0.15%
2021년 06월 22일	71.41	71.41	71.41	71.41	-	-0.09%

(출처: Investing)

02. 데이터 전처리

3) 변수 생성(외부 데이터)

원달러 환율 데이터 전처리

```
df = pd.read_csv('RAW_DATA/변수/외부데이터/환율데이터.csv')
df.drop('Unnamed: 0', axis=1, inplace=True)

df['REG_DATE'] = pd.to_datetime(df['REG_DATE'])
df['날짜'] = pd.to_datetime(df['날짜'])

price_list = df['REG_DATE'].unique()
price_list2 = df['날짜'].unique()

list_price = []
for i in df['환율']:
    list_price.append(i.replace(",",""))

df['환율'] = list_price
# print(df)

dict_price = {}

for i in price_list:
    is_price = df['날짜'] == i
    price_df = df[is_price]
    # print(price_df)
    dict_price[i] = price_df['환율'].sum()

# print(df['환율'])
# print(dict_price)
dict_price = pd.DataFrame(dict_price, index=['환율'])

dict_price = dict_price.transpose()

dict_price['환율'].interpolate()

print(dict_price)

dict_price.to_csv('RAW_DATA/변수/외부데이터/환율데이터전처리.csv')
```

유가 두바이유 데이터 전처리

```
df = pd.read_csv('RAW_DATA/변수/외부데이터/두바이유내역(investing.com).csv')
df1 = pd.read_csv('RAW_DATA/변수/외부데이터/환율데이터.csv')

# print(df.columns)

df.drop('오픈', axis=1, inplace=True)
df.drop('고가', axis=1, inplace=True)
df.drop('저가', axis=1, inplace=True)
df.drop('거래량', axis=1, inplace=True)
df.drop('변동 %', axis=1, inplace=True)

date = []
for i in df['날짜']:
    i = i.replace("년", "-")
    i = i.replace("월", "-")
    i = i.replace("일", "")
    date.append(i)

# print(date)

df['날짜'] = date

df['날짜'] = pd.to_datetime(df['날짜'])
df1['REG_DATE'] = pd.to_datetime(df1['REG_DATE'])

# print(df)

do = {}
for i in df1['REG_DATE']:
    is_date = df['날짜'] == i
    date = df[is_date]
    do[i] = date['종가'].sum()

# print(do)

do_price = pd.DataFrame(do, index=['종가'])
do_price = do_price.transpose()
# print(do_price)

do_price.to_csv('RAW_DATA/변수/외부데이터/두바이유전처리.csv')
```

**최종 데이터 셋과 일치하는 날짜의
원달러 환율 및 유가 두바이유 만을 추출하기 위한
데이터 전처리**

원달러 환율

	환율
2015-12-28	1171.00
2016-01-04	1172.00
2016-01-11	1196.20
2016-01-18	1211.00
2016-01-25	1203.20
...	...
2021-06-07	1117.50
2021-06-14	1111.20
2021-06-21	1132.20
2021-06-28	1128.80

두바이유

	종가
2015-12-28	34.58
2016-01-04	32.10
2016-01-11	27.86
2016-01-18	0.00
2016-01-25	26.38
...	...
2021-06-07	69.84
2021-06-14	70.78
2021-06-21	71.48
2021-06-28	71.54

02. 데이터 전처리

3) 변수 생성 - 최종 변수 선정

주별 평균단가와 변수간의 상관관계 확인 코드

```
squid= pd.read_csv('data/result/초안/squid_result_initial.csv', encoding='cp949')
# squid.head(2)

squid_cloumnns = []

for i in squid.columns:
    squid_cloumnns.append(i)

print(squid_cloumnns)

for i in range(1, len(squid_cloumnns) -1):
    value = squid_cloumnns[i]
    x_squid = squid[value]
    y_squid = squid['P_PRICE']
    corr = np.corrcoef(x_squid, y_squid)[0, 1]
    if corr >= 0.4 or corr <= -0.4:
        print(f'{value}과 P_PRICE의 상관관계수 : {corr}')
```

상관계수 기준: 절댓값 0.4 이상

	REG_DATE	베트남	냉동,살 자숙,포 장함	냉동,살 자숙,포 장함C	냉동,살,자 숙,포장함 감_베트남	냉동,살,자 숙,포장함 감_태국	냉동,살,자 숙,포장함 감_베트남C	냉동,살,자 숙,포장함 감_태국C	P_PRICE
0	2016-07-04	10.150000	8.120000	19.345	2	16.19	22.50	1	10.988750
1	2016-07-11	11.288000	8.857500	16.340	1	16.34	0.00	1	9.633846
2	2016-07-18	10.128000	10.300000	18.265	2	16.91	19.62	1	10.946000
3	2016-07-25	10.866000	9.397500	18.280	1	18.28	0.00	1	9.624167
4	2016-08-01	11.068000	11.500000	15.270	1	15.27	0.00	1	10.422700
...
229	2020-11-30	10.063333	11.695000	0.000	0	0.00	0.00	0	8.764286
230	2020-12-07	10.376667	9.315000	14.855	2	15.78	13.93	1	9.614000
231	2020-12-14	10.153333	8.963333	15.800	2	14.61	16.99	1	9.702000
232	2020-12-21	9.136667	10.483333	17.440	1	0.00	17.44	0	9.889000
233	2020-12-28	11.395000	8.540000	18.905	2	16.77	21.04	1	10.782727

결측치 0으로 변환

```
노르웨이과 P_PRICE의 상관관계수 : 0.8128772225011722
칠레과 P_PRICE의 상관관계수 : -0.4754421454865802
c칠레과 P_PRICE의 상관관계수 : -0.49093647660833467
자사제품제조용과 P_PRICE의 상관관계수 : 0.6612479986189607
판매용과 P_PRICE의 상관관계수 : 0.9220717381393944
냉장과 P_PRICE의 상관관계수 : 0.6977374765283038
냉장, 포장함감, 필렛(F)과 P_PRICE의 상관관계수 : 0.41547842580026867
냉장, 필렛(F)과 P_PRICE의 상관관계수 : 0.6672517359984668
냉장, 필렛(F), 횡감과 P_PRICE의 상관관계수 : 0.40806748502944573
c냉장과 P_PRICE의 상관관계수 : 0.41547842580026867
노르웨이 자사제품제조용과 P_PRICE의 상관관계수 : 0.629977929255789
노르웨이 판매용과 P_PRICE의 상관관계수 : 0.6684619910083915
칠레 판매용과 P_PRICE의 상관관계수 : -0.4754421454865802
호주 판매용과 P_PRICE의 상관관계수 : -0.4229645173194794
c칠레 판매용과 P_PRICE의 상관관계수 : -0.49093647660833467
c호주 판매용과 P_PRICE의 상관관계수 : -0.4251406401486903
노르웨이 냉장과 P_PRICE의 상관관계수 : 0.63736095967937
노르웨이 냉장, 포장함감, 필렛(F)과 P_PRICE의 상관관계수 : 0.4216365857502705
노르웨이 냉장, 필렛(F)과 P_PRICE의 상관관계수 : 0.6672517359984668
노르웨이 냉장, 필렛(F), 횡감과 P_PRICE의 상관관계수 : 0.40806748502944573
```

02. 데이터 전처리

3) 변수 생성 - 최종 변수 선정

<최종 데이터 셋>

연어 : 26개 변수, 235개 데이터

오징어 : 10개 변수, 234개 데이터

흰다리새우 : 10개 변수, 234개 데이터

<Data Split>

Train Data : 2016년 7월 4일 ~ 2020년 5월 25일

Validate Data : 20%


Test Data : 2020년 6월 1일 ~ 2020년 12월 28일

03. 모델 탐색 및 검증

시계열 데이터 예측을 위한 모델 탐색 및 검증

1. 모델 탐색

- SARIMA, XGBoost, LSTM, GRU



2. 모델 검증

03. 모델 탐색 및 검증

1) 모델 탐색

< 모델 선정 기준: 시계열 데이터 예측 모델 >

SARIMA

대표적인 시계열
분석 모형 ARIMA + 계절성 반영

LSTM

장기간에 걸친
데이터간의 연관관계 파악 용이

XGBoost

사용하기에 편리하고
앙상블에 용이

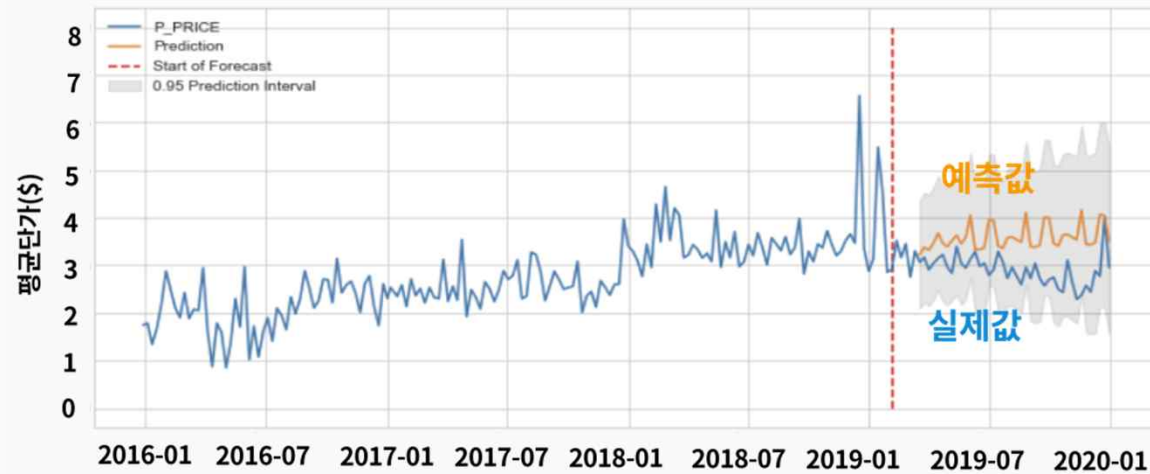
GRU

기존 LSTM의 구조를
간단하게 개선한 모델

03. 모델 탐색 및 검증

2) 모델 검증 - SARIMA

< SARIMA를 통한 오징어 평균단가 예측 그래프 >

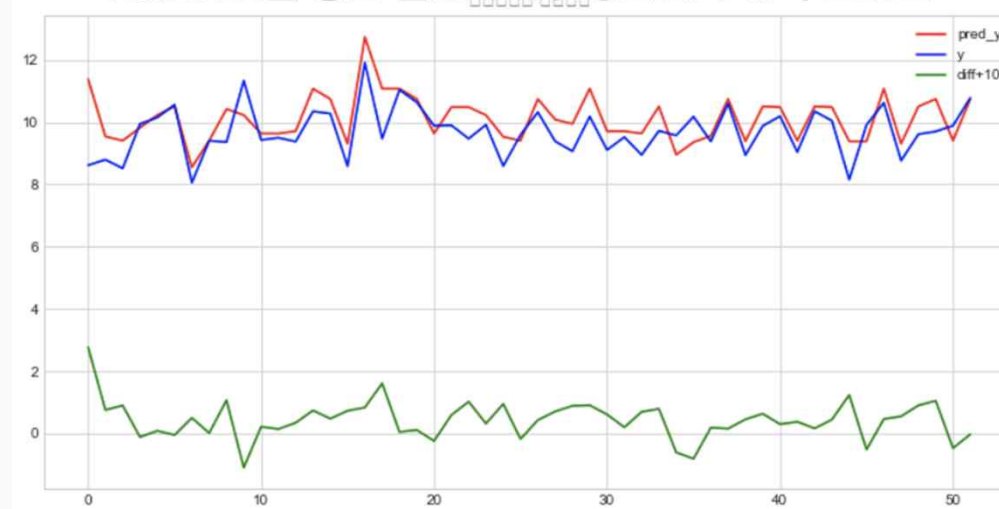


1. **하나의 변수**에 대해서만 작동
2. 모델 돌리는 시간이 **평균 5분 이상**
→ 후보 모델에서 **제외**

03. 모델 탐색 및 검증

2) 모델 검증 - XGBoost

< XGBoost를 통한 힌다리새우 평균단가 예측 그래프 >



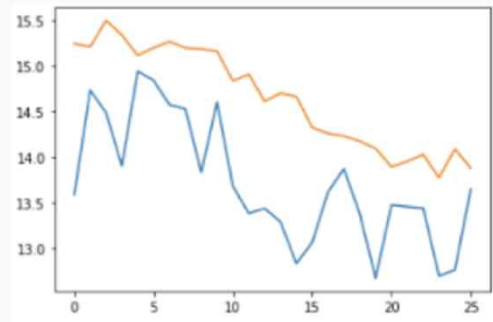
1. Test Data가 주어지지 않음

2. 다 대 다 예측이 불가능

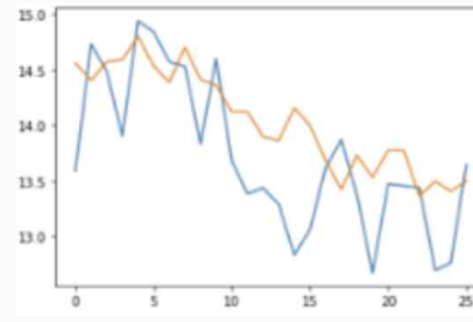
→ 후보 모델에서 제외

03. 모델 탐색 및 검증

2) 모델 검증



LSTM



GRU

[최종 모델로 LSTM, GRU 선정]

04. 모델 성능 향상

1. 하이퍼파라미터 튜닝

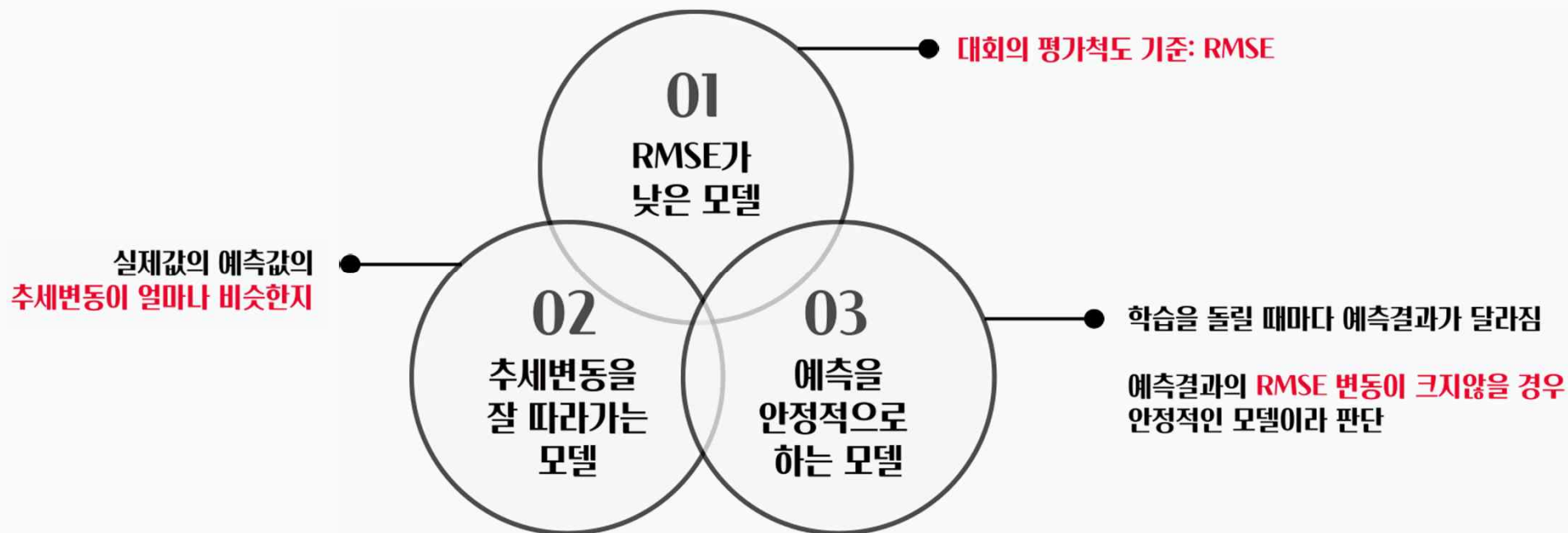
2. 소프트 앙상블

앙상블: 하나의 데이터를 여러개의 분류기를 통해 다수의 학습 모델을 만들어 학습시키고 학습 결과를 결합함으로써 과적합을 방지하고 정확도를 높이는 학습기법

3. 최종 모델

04. 모델 성능 향상

1) 하이퍼파라미터 튜닝 - 기준



* RMSE : 오차의 제곱합을 산술평균한 값의 제곱근으로서 관측값들간의 상호간 편차를 의미.
실제값과 예측값의 차이가 얼마인가를 알려주는데 많이 사용되는 척도
→ 개별 관측값이 중심으로부터 얼마나 머릴 떨어져 있는 정도를 나타냄

04. 모델 성능 향상

1) 하이퍼파라미터 튜닝

LSTM 모델 코드

```
def RMSE(y_test, y_pred):
    return np.sqrt(mean_squared_error(y_test, y_pred))

def split_xy5(dataset, time_steps, y_column):
    x=[]
    y=[]
    for i in range(len(dataset)):
        x_end_number = i + time_steps
        y_end_number = x_end_number + y_column
        if y_end_number > len(dataset):
            break
        tmp_x = dataset[i:x_end_number, :]
        tmp_y = dataset[x_end_number:y_end_number, :]
        x.append(tmp_x)
        y.append(tmp_y)
    return np.array(x), np.array(y)

df = pd.read_csv(r'C:\Users\yuhwan\PycharmProjects\pythonsml\bc_test\
df = df.fillna(0) # nan
df['reg_date'] = pd.to_datetime(df['reg_date'], format="%Y-%m-%d")
df=df.set_index(df['reg_date'])

dataset = df.loc['2016-07':'2020-06'].reset_index(drop=True)
df1 = dataset.copy()
use_col = dataset.columns[1:]
dataset=dataset[use_col]

# 정규화
sc = MinMaxScaler(feature_range=(0,1))
dataset = sc.fit_transform(dataset)
```

```
pred = 26
epoch = 500
window_size = 52
unit= 64
bs = 36
```

* 하이퍼파라미터 튜닝 코드

```
x, y = split_xy5(dataset, window_size, pred)

savey_shape1=y.shape[1]
savey_shape2=y.shape[2]
y=y.reshape(y.shape[0],y.shape[1]*y.shape[2])

model = Sequential()
model.add(LSTM(unit, input_shape=(x.shape[1],x.shape[2]))) # LSTM GRU
model.add(Dense(y.shape[1]))
model.summary()
model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])

early_stopping = EarlyStopping(monitor='loss', patience=10, mode='min')
hist = model.fit(x,y, epochs=epoch, batch_size=bs, validation_split=0.2, callbacks=[early_stopping])

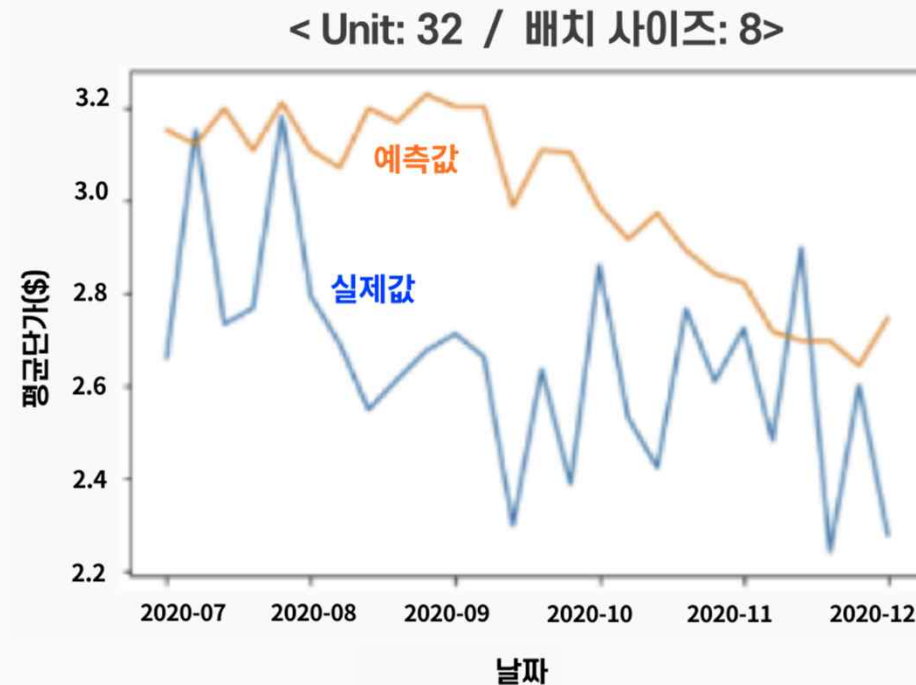
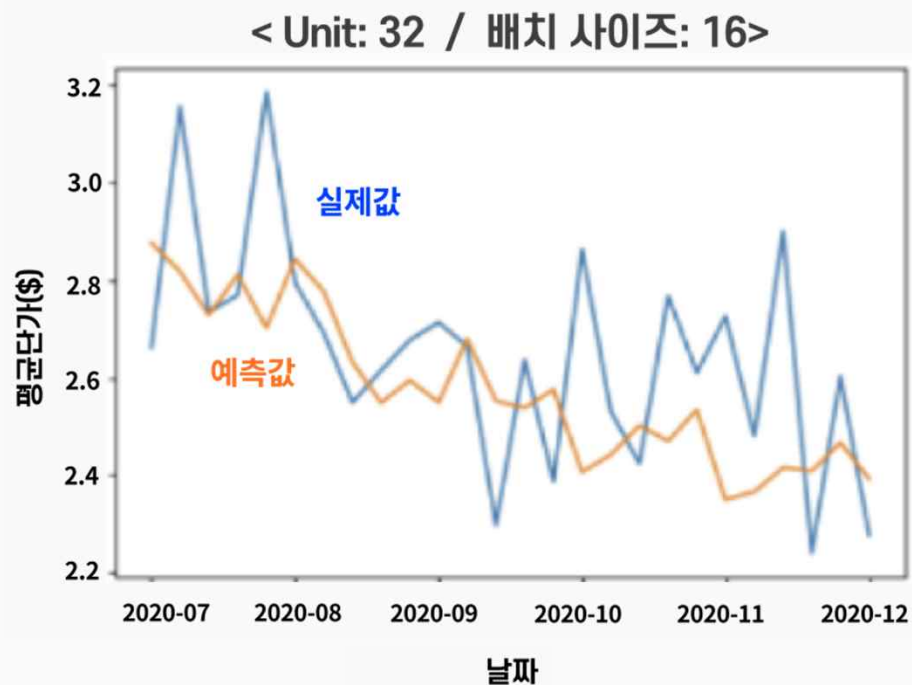
# 예측 # 모델 평가 부분
x_test = dataset[-window_size:] shape: Any
x_test=x_test.reshape(1,x_test.shape[0],x_test.shape[1])
y_pred = model.predict(x_test)
y_test = df.loc['2020-07:'].reset_index(drop=True) # 여기 날짜
y_test = y_test[use_col].values
y_pred=y_pred.reshape(savey_shape1,savey_shape2)
y_pred = sc.inverse_transform(y_pred)
# y_test=y_test.reshape(y_test.shape[0],y.shape[1],y.shape[2])

print("RMSE: ", RMSE(y_test[:-1],y_pred[:-1]))
print("window_size",window_size,"//unit:",unit,"//반복횟수:",epoch,"//배치사이즈",bs)
```

04. 모델 성능 향상

1) 하이퍼파라미터 튜닝

< 오징어 데이터 셋 GRU 예측 결과 >



하이퍼파라미터에 따라 예측결과 다름 → 안정적인 예측을 위해 **최적의 하이퍼파라미터 선정 필요**

04. 모델 성능 향상

1) 하이퍼파라미터 튜닝



< 최적의 모델 하이퍼파라미터 선정 >

연어

	LSTM	GRU
Unit 수	16	64
배치 사이즈	8	64

오징어

	LSTM	GRU
Unit 수	32	64
배치 사이즈	4	4

흰다리새우

	LSTM	GRU
Unit 수	64	16
배치 사이즈	64	16

04. 모델 성능 향상

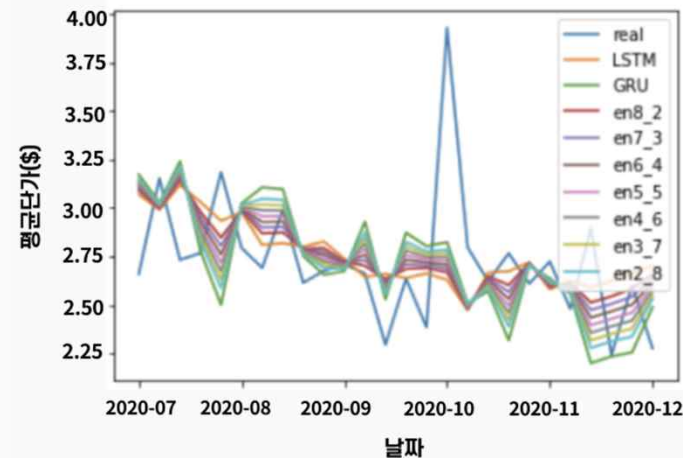
2) 소프트 앙상블

< 최적의 앙상블 비율 선정 코드 >

```
en9_1=[]
for i in range(len(y_pred)):
    en9_1.append((y_pred[:, -1][i]*0.9+y_pred2[:, -1][i]*0.1))
en8_2=[]
for i in range(len(y_pred)):
    en8_2.append((y_pred[:, -1][i]*0.8+y_pred2[:, -1][i]*0.2))
en7_3=[]
for i in range(len(y_pred)):
    en7_3.append((y_pred[:, -1][i]*0.7+y_pred2[:, -1][i]*0.3))
en6_4=[]
for i in range(len(y_pred)):
    en6_4.append((y_pred[:, -1][i]*0.6+y_pred2[:, -1][i]*0.4))
en5_5=[]
for i in range(len(y_pred)):
    en5_5.append((y_pred[:, -1][i]*0.5+y_pred2[:, -1][i]*0.5))
en4_6=[]
for i in range(len(y_pred)):
    en4_6.append((y_pred[:, -1][i]*0.4+y_pred2[:, -1][i]*0.6))
en3_7=[]
for i in range(len(y_pred)):
    en3_7.append((y_pred[:, -1][i]*0.3+y_pred2[:, -1][i]*0.7))
en2_8=[]
for i in range(len(y_pred)):
    en2_8.append((y_pred[:, -1][i]*0.2+y_pred2[:, -1][i]*0.8))
en1_9=[]
for i in range(len(y_pred)):
    en1_9.append((y_pred[:, -1][i]*0.1+y_pred2[:, -1][i]*0.9))
```

```
print("LSTM의 RMSE: ", RMSE(y_test[:, -1], y_pred[:, -1]))
print("GRU의 RMSE: ", RMSE(y_test[:, -1], y_pred2[:, -1]))
print("앙상블모델9:1의 RMSE: ", RMSE(y_test[:, -1], en9_1))
print("앙상블모델8:2의 RMSE: ", RMSE(y_test[:, -1], en8_2))
print("앙상블모델7:3의 RMSE: ", RMSE(y_test[:, -1], en7_3))
print("앙상블모델6:4의 RMSE: ", RMSE(y_test[:, -1], en6_4))
print("앙상블모델5:5의 RMSE: ", RMSE(y_test[:, -1], en5_5))
print("앙상블모델4:6의 RMSE: ", RMSE(y_test[:, -1], en4_6))
print("앙상블모델3:7의 RMSE: ", RMSE(y_test[:, -1], en3_7))
print("앙상블모델2:8의 RMSE: ", RMSE(y_test[:, -1], en2_8))
print("앙상블모델1:9의 RMSE: ", RMSE(y_test[:, -1], en1_9))
```

< 최적의 앙상블 선정을 위한 오징어 그래프 >



LSTM의 RMSE: 0.34510605374424436

GRU의 RMSE: 0.38180010380241586

앙상블 모델9:1의 RMSE: 0.3421938960455942

앙상블 모델8:2의 RMSE: 0.34078241799388853

앙상블 모델7:3의 RMSE: 0.3408902610806306

앙상블 모델6:4의 RMSE: 0.3425159902083994

앙상블 모델5:5의 RMSE: 0.3456381876474604

앙상블 모델4:6의 RMSE: 0.35021683245521257

앙상블 모델3:7의 RMSE: 0.3561957642542435

앙상블 모델2:8의 RMSE: 0.3635058940008092

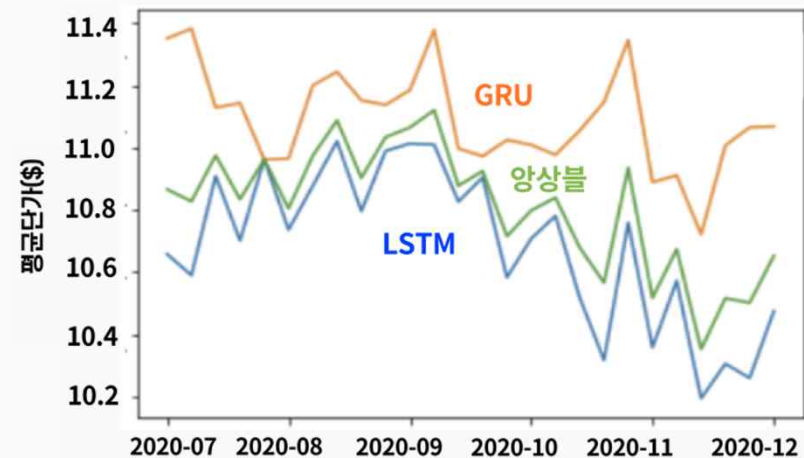
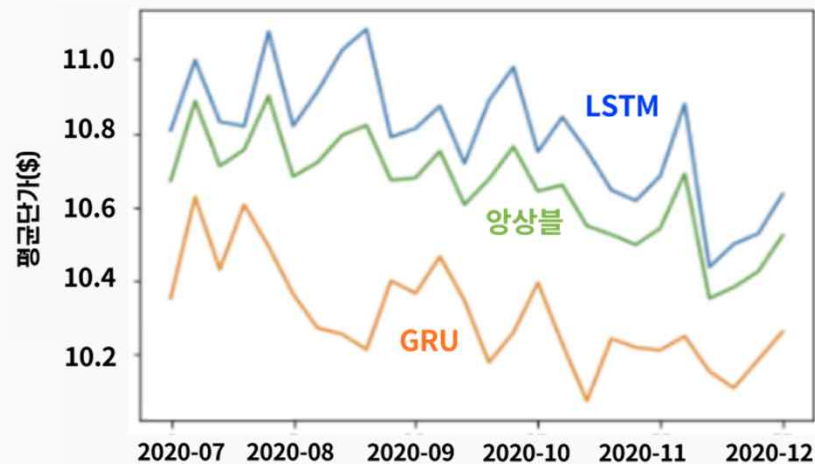
앙상블 모델1:9의 RMSE: 0.372068766811931

RMSE를 기준으로 앙상블 비율 선정

04. 모델 성능 향상

2) 소프트 앙상블

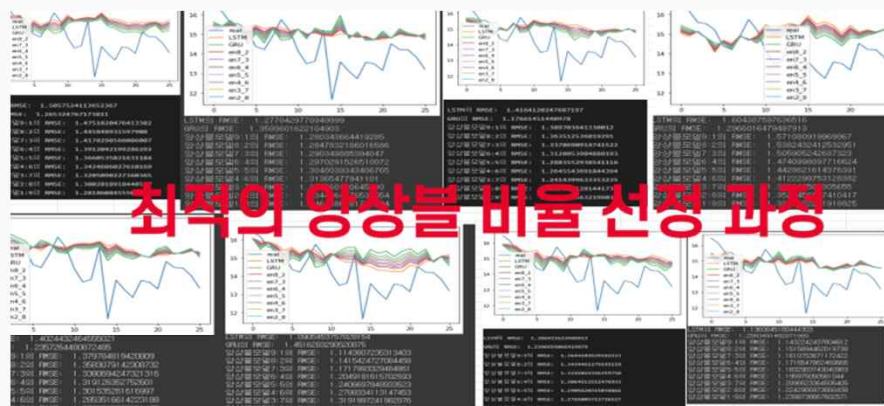
< 흰다리 새우의 2021년 26주 예측 그래프 >



소프트 앙상블을 통해 **안정적인 예측결과**가 나오는 것을 확인

04. 모델 성능 향상

2) 소프트 앙상블



< 최적의 모델 앙상블 비율 선정 >

연어

	LSTM	GRU
Unit 수	16	64
배치 사이즈	8	64
앙상블 비율	8	2

오징어

	LSTM	GRU
Unit 수	32	64
배치 사이즈	4	4
앙상블 비율	5	5

흰다리새우

	LSTM	GRU
Unit 수	64	16
배치 사이즈	64	16
앙상블 비율	3	7

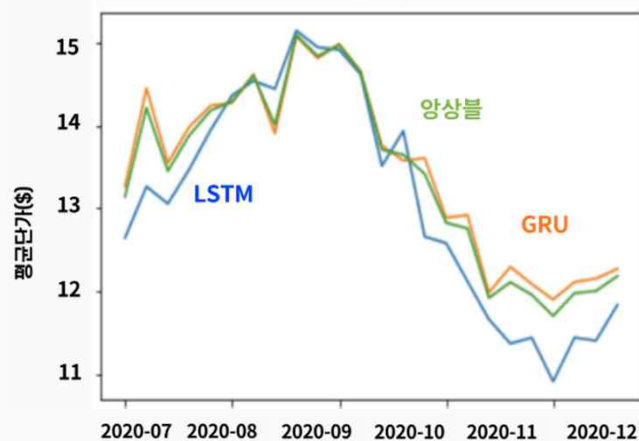
04. 모델 성능 향상

3) 최종 모델

- 상세어종별 최적의 모델 하이퍼파라미터를 선정 → 소프트 앙상블

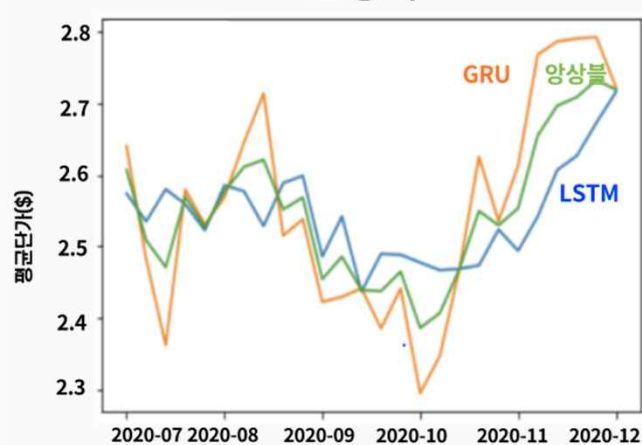
< 2021년 가격 예측 >

< 연어 >



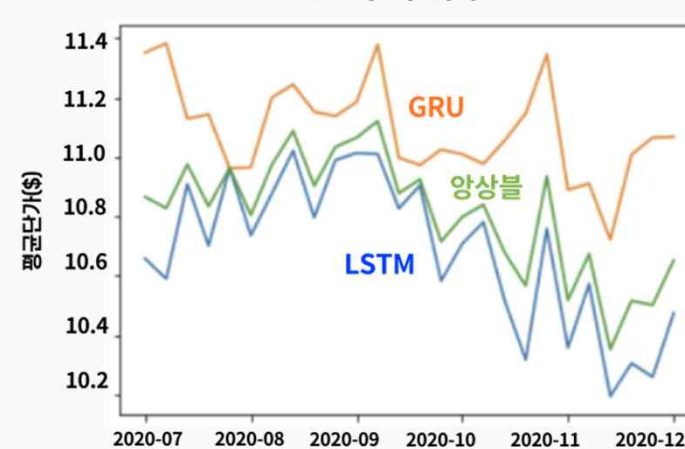
3월까지 가격 상승 후 하락세

< 오징어 >



급등락이 반복

< 흰다리새우 >



완만한 하락세

05. 시뮬레이터 구현

1. UI 구현



2. 기능 구현

05. 시뮬레이터 구현

1) UI 구현

- 시뮬레이터 최종 UI

해양수산데이터연구소

Search for...

예측 단가

수산물 수입단가 예측 시뮬레이터

수산물 가격정보

자유 토론방

뉴스

데이터 제공

그래프

역설

수산물 수입단가 예측 시뮬레이터

어종 선택

연어

학습기간 선택

2016년 07월 ~ 2020년 12월

예측 기간

☐ 1주 ☐ 4주 ☒ 12주 ☐ 24주

모델 선택

☐ LSTM ☐ GRU ☒ LSTM + GRU

양상불 비율 선택

2 : 8

LSTM 하이퍼 파라미터 설정

Epoches

500

Window Size

52

Units

16

Batch Size

8

GRU 하이퍼 파라미터 설정

Epoches

500

Window Size

52

Units

64

Batch Size

64

실행

Reset

그래프 출력

예측 평균단가

실제 평균단가

날짜	예측 평균단가	실제 평균단가
yyyy-mm-dd	\$/kg	\$/kg
yyyy-mm-dd	\$/kg	\$/kg
yyyy-mm-dd	\$/kg	\$/kg
yyyy-mm-dd	\$/kg	\$/kg
yyyy-mm-dd	\$/kg	\$/kg

결과 출력

설정값	Epoches	Window Size	Units	Batch Size	비율
LSTM	None	None	None	None	None
GRU	None	None	None	None	None

RMSE 평가점수

None

모델 저장

모델 불러오기

데이터 저장

05. 시뮬레이터 구현

2) 기능 구현

수산물 수입단가 예측 시뮬레이터

어종 선택
연어

학습기간 선택
2016년 07월 ~ 2020년 12월

예측 기간
☐ 1주 ☐ 4주 ☒ 12주 ☐ 24주

모델 선택
☐ LSTM ☐ GRU ☒ LSTM + GRU

양상물 비율 선택
2 : 8 **조건 설정**

LSTM 하이퍼 파라미터 설정

Epoches	Window Size
500	52
Units	Batch Size
16	8

GRU 하이퍼 파라미터 설정

Epoches	Window Size
500	52
Units	Batch Size
64	64

실행 Reset

```
$(".fish_species").change(function (e) {  
  if (e.target.value === 'salmon') {  
    $('.lstm_epochs').val("500")  
    $('.lstm_windowsize').val("52")  
    $('.lstm_units').val("16")  
    $('.lstm_batchsize').val("8")  
    $('.gru_epochs').val("500")  
    $('.gru_windowsize').val("52")  
    $('.gru_units').val("64")  
    $('.gru_batchsize').val("64")  
    $(".ratio").val("{{2}}_{{8}}")  
  }  
})
```

JQuery 사용하여 기능 19개 구현

05. 시뮬레이터 구현

2) 기능 구현



```
let myChartOne = document.getElementById('myChartOne').getContext('2d');
let lineChart = new Chart(myChartOne, {
  type: 'line',

  data: {
    labels: predict_dates,
    datasets: [{
      label: '예측 평균단가',
      data: predict_price,
      borderColor: 'orange',
      pointHoverBackgroundColor: 'red',
      pointHoverBorderColor: 'red',
      fill: false
    },
    {
      label: '실제 평균단가',
      data: real_price,
      borderColor: 'gray',
      pointHoverBackgroundColor: 'red',
      pointHoverBorderColor: 'red',
      fill: false
    }
  ],
});
```

Chart.js 사용하여 그래프 출력
Table 사용하여 결과 출력
실행, Reset, 모델저장, 데이터저장 버튼 구현

06. 결론 및 느낀점

06. 결론 및 느낀 점

1) 결론 및 느낀점



1. 프로젝트를 진행하면서 데이터 분석부터 전처리, 모델선정, 모델 설계, 시뮬레이터 제작 까지 직접 수행 하며 **실질적 경험**을 쌓을 수 있었습니다.
2. 같은 목표를 가지고 학원에서 처음 만난 교육생들과 **함께 프로젝트를 진행**하면서, 융합적 연구와 팀의 중요성을 크게 배웠습니다.
3. 분석한 것은 많은데 **문서로 정리**를 다 하지 못해 아쉬움이 많이 남는 프로젝트였고, 이후 보완해나갈 계획입니다.

시연 시작하겠습니다.
