



2025 관세청 공모전

딥러닝 기반 자동 밀수입 의심 거래 탐지 시스템

OB 3조 | 고은서 김연우 류호현 박선민 추윤서

Table of contents

01

- 기획 개요

02

- 데이터 병합

03

- EDA

04

- 모의 데이터 제작

05

- 모델링

06

- 웹 서비스 데모

07

- 차별성 및 기대효과



01

아이디어 기획 개요



공모전 소개

관세청 공공데이터 활용 경진대회

공공데이터 포털에 개방되어 있는
관세청 공공데이터 및 관세청 수출입무역통계 누리집,
국가관세종합정보시스템 등 관세청 소관 누리집이나 시스템
등을 통해 개방 중인 공공 데이터를 활용해
신규 비즈니스 또는 서비스 기획

진행 일정

	내용
1주차 (6/23~6/27)	데이터 선정 및 아이디어 도출
2주차 (6/30~7/4)	데이터 전처리 및 EDA
3주차 (7/7~7/11)	모델링 및 아이디어 구체화
4주차 (7/14~7/18)	모델링 및 아이디어 구체화(2)
5주차 (7/21~7/25)	공모전 접수(7/20)
6주차 (7/28~8/1)	발표 자료 작성

 관세청

2025년 관세청 공공데이터 활용 경진대회

2025. **4.1** (화) ~ **7.20** (일)



공모부문

① 아이디어 기획 ② 제품 및 서비스 개발

참가자격

관세청 공공데이터에 관심 있는 누구나
→ 개인 또는 단체(팀) · 예비창업자 · 기업 · 기관 등)
※ 단체의 경우 1개 팀당 최대 5인까지 허용
※ 공모 부문을 선택하여 해당 부문에 1개 작품만 응모 가능
(아이디어 기획 부문 1개 또는 제품 및 서비스 개발 부문 1개 중 선택)

공모일정

접수 4.1. ~ 7.20.	1차 서류평가 7.21. ~ 8.1.	2차 발표평가 8.4. ~ 8.14.	최종 시상 8월 말
--------------------	-------------------------	-------------------------	---------------

※ 추진 일정은 진행 상황에 따라 변경될 수 있음

신청방법

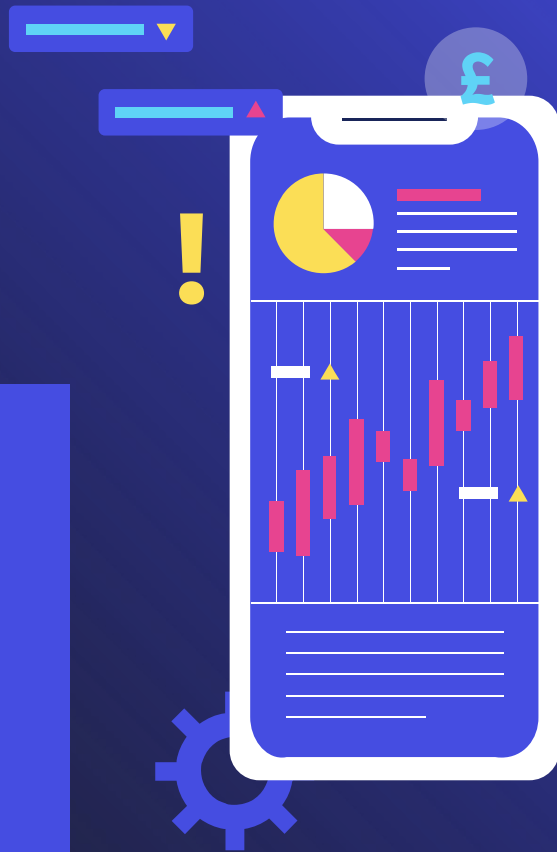
이메일(customsdata@korea.kr) 서류 접수

시상규모

1.1 기획 개요

“ 비지도 학습 기반 이상탐지 모델을 활용한 밀수입 자동 감지 시스템 설계 ”

1. 통관 데이터에 숨겨진 이상 거래를 자동으로 감지하여
다양한 고위험 품목의 밀수입(밀반입)을 **사전에 식별 가능한 시스템 구축**
2. **비지도 학습 기반**의 이상탐지 기법을 활용
→ 기존 정상 거래 흐름과 뚜렷하게 벗어난 수입 패턴을
사전 정의된 라벨 없이도 자동 탐지가능
3. 중량 대비 단가, 운송비용, 수입 국가 등 세부 통관 데이터를 기반으로,
정형화된 정상 흐름과 다른 이상 거래 신호를 감지하도록 설계



1.2 주요 설계 요소



HSK 코드 및 품목별 수입 데이터 분류

수산물 항목 품목별 구조화된 데이터를 활용하여
품목별 특성과 가격대 범위를 반영



이상값 탐지

- 중량 대비 단가, 운송비용, 무역 국가 등을 고려해 이상치 탐지 설계
- 2020년 5월부터 2025년 4월까지의 데이터 활용
(학습용 / 평가용 데이터 구분)
→ 실제 시계열 예측 성능과 이상 거래 탐지 정확도를 함께 검증



비지도 학습 기반 탐지 모델 적용

수입 거래의 다양한 시계열적 특성(계절성, 가격 변동성)
: 시계열 데이터의 장기 의존성 학습 필요

→ LSTM 기반 AutoEncoder 모델 활용:
정상 거래의 재구성 오차를 기반으로 이상 여부 판단



거시경제 요인 반영한 정밀도 향상

국제 유가, 환율, 소비자물가지수(CPI), 생산자물가지수(PPI) 등
외부 경제 변수를 통합하여, 단순한 거래 패턴뿐 아니라
글로벌 경제 여건에 따른 가격 변동과 왜곡 현상도
모델이 학습하고 고려할 수 있도록 구성



1.3 활용 데이터 소개

2020년 5월부터 2025년 4월까지의 기간에 해당하는 데이터 활용

→ 시계열 기반 이상치 탐지 모델 설계에 필요한 충분한 데이터 길이와 계절성 확보, 그리고 최근 추세 반영을 위한 적절한 기간

1) 운송비 데이터

관세청_해상수출입(수입) 운송비용 *

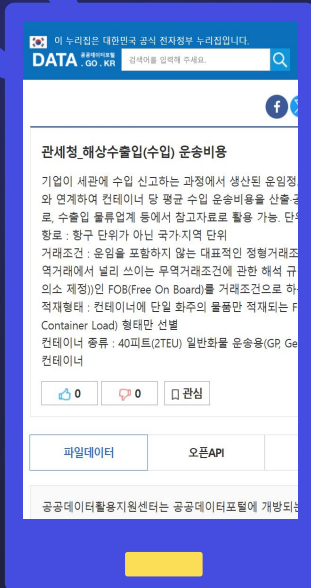
- 수산물의 수입 원가를 구성하는 주요 요소
- 해상 운송 중심으로 분석
- 운송비의 변동이 수입 단가 및 물동량에 미치는 영향을 정밀하게 파악



2) 수출입 데이터

관세청_해상수출입(수입) 운송비용 *

- 관세청 수출입신고 데이터 기반
해양수산부에서 집계한 국가별·품목별 월별 통계
→ 수산물 품목만 사용
- 수입 급증·단가 변화 감지, 이상 거래 탐지,
외부 요인(운송비, 유가 등)과의 연관성 분석



3) 국제 유가

원유의 국제 시장에서의 거래 가격. 한국의 수입 비용 및 물가에 미치는 영향을 보다 정확하게 반영하기 위해 두바이유를 국제유가의 대표 지표로 선택



4) 환율

원화 대비 외국 통화(예: 달러, 위안, 엔)의 상대 가치를 나타내며, 수입 원가 산정과 가격 책정에 직결되는 주요 거시경제 변수

5) 소비자 물가지수(CPI)

가계가 구입하는 상품 및 서비스의 평균 가격 변동을 측정하는 지수로, 국민 생활 물가 수준을 대표하는 경제 지표



6) 생산자 물가지수(PPI)

국내 생산자가 공급하는 상품의 도매 가격 변동을 나타내는 지표로, 산업 원가 및 공급 측 물가 수준을 반영

02

데이터 병합



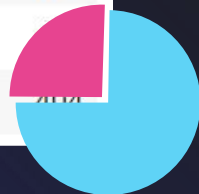
운송비 데이터 설명:

관세청_해상수출입(수입) 운송비용

기업이 세관에 수입 신고하는 과정에서 생산된
운입정보를 화물정보와 연계하여
컨테이너 당 평균 수입 운송비용을 산출·공표하는
데이터로, 수출입 물류 업계 등에 참고 자료로 활용
가능하다.

- 단위: 천원/2 TEU
- 항로: 항구 단위가 아닌 국가·지역 단위
- 거래조건: 운임을 포함하지 않는 대표적인
정형거래조건인 FOB를 거래조건으로 하는
수입신고 건
- 적재형태: 컨테이너에 단일 화주의 물품만
적재되는 FCL 형태만 선별
- 컨테이너 종류: 40피트(2 TEU) 일반화물
운송용 컨테이너

	기간	미국서부	미국동부	유럽연합	중국	일본	베트남
0	2019-01	2065	1830	1136	712	769	423
1	2019-02	1636	1850	1164	690	762	427
2	2019-03	1692	1491	1142	680	892	429
3	2019-04	1758	1815	1369	662	797	
4	2019-05	2552	1620	1429	692	901	



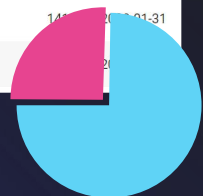
수출입 데이터 설명:

해양수산부_수산물품목별수출입현황

수산물 수출입정보는 국내에 수출 및 수입되는 수산물에 대한 **월별 통계정보**로 관세청에서 보유한 데이터를 집계하여 제공하는 데이터로 **수산물 품목별 수출입 현황**을 제공하는 목록이다.

- 기준년월
- HSK 품목코드
- 수출입구분코드
- 수산물수출입품목명
- 수출입구분명
- 당월수출입중량 (킬로그램)
- 당월수출입미화금액 (달러)
- 당해누계수출입중량 (킬로그램)
- 당해누계수출입미화금액 (달러)
- 데이터기준일자

	기준년월	국가코드	HSK품목코드	수출입구분코드	수출입구분명	국가명	경제권명	수산물수출입품목명	당월수출입중량 (킬로그램)	당월수출입미화금액 (달러)	당해누계수출입중량 (킬로그램)	당해누계수출입미화금액 (달러)	데이터기준일자
0	2020-01	AE	301998000	E	수출	아랍에미리트연합	걸프협력회의	넙치(할어)	392	8670	392	8670	2020-01-31
1	2020-01	AE	302290000	E	수출	아랍에미리트연합	걸프협력회의	기타 넙치류(간장, 어란제외/신선, 냉장/피레트, 어육제외)	3	84	3	84	2020-01-31
2	2020-01	AE	302899090	E	수출	아랍에미리트연합	걸프협력회의	기타(신선 또는 냉장)(삼치,복어,병어,아귀 외 기타)	35	1197	35	1197	2020-01-31
3	2020-01	AE	306339000	E	수출	아랍에미리트연합	걸프협력회의	계(기타)	232	14184	232	14184	2020-01-31
4	2020-01	AE	307119000	E	수출	아랍에미리트연합	걸프협력회의	굴치패 기타(산 것 신선 또는 냉장한 것)	3	49	3	49	2020-01-31



수출입 데이터 설명:

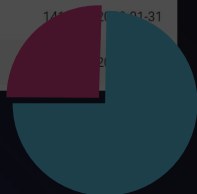
해양수산부 수산물품목별수출입현황

● 운송비 데이터에는 결측치가 이상치가 발견되지 않았다.

- 기준년월
- HSK 품목코드
- 수출입구분코드
- 수산물수출입품목명
- 수출입구분명
- 당월수출입중량
(킬로그램)
- 당월수출입미화금액
(달러)
- 당해누계수출입중량
(킬로그램)
- 당해누계수출입미화금액
(달러)
- 데이터기준일자

- 운송비 데이터에는 결측치가 이상치가 발견되지 않았다.
- 수출입 데이터에는 수입 중량이 0인 품목들이 존재했으며, 이를 확인해본 결과, 당월에 수입이 되지 않아 발생한 값들이었다.
- 국가 코드, 경제권명, 파일기준년월에 결측치가 존재했다.

연도	국가 코드	경제권명	파일기준	년월	결측치	연도	국가 코드	경제권명	파일기준	년월	결측치
2020-01	AE	3011998000	E	수출	아랍에미리트연합	컴퓨터 및 소프트웨어	392	8570	392	8670	2020-01-31
2020-01	AE	3022990000	E	수출	아랍에미리트연합	컴퓨터 및 소프트웨어	3	84	3	84	2020-01-31
2020-01	AE	302899090	E	수출	아랍에미리트연합	컴퓨터 및 소프트웨어	35	1197	35	1197	2020-01-31
2020-01	AE	306339000	E	수출	아랍에미리트연합	컴퓨터 및 소프트웨어	232	14184	232	14184	2020-01-31
2020-01	AE	307119000	E	수출	아랍에미리트연합	컴퓨터 및 소프트웨어	3	49	3	49	2020-01-31



병합 전 문제점 - 1

[수출입 데이터]

그리스, 네덜란드, 덴마크, 독일, 라트비아, 루마니아, 몰타, 벨기에, 불가리아, 스페인,
아일랜드, 에스토니아, 이탈리아, 체코공화국, 크로아티아 등

[운송비 데이터]

‘유럽연합’

미국

‘미국 서부, 동부’

베트남

‘베트남’

일본

‘일본’

중국

‘중국’

```
[ ] # 1. 국가 → 지역 매핑
```

```
country_to_region = {
```

```
    '그리스': '유럽연합',
```

```
    '네덜란드': '유럽연합',
```

```
    '덴마크': '유럽연합',
```

```
    '독일': '유럽연합',
```

```
    '라트비아': '유럽연합',
```

```
    '루마니아': '유럽연합',
```

```
    '몰타': '유럽연합',
```

```
    '벨기에': '유럽연합',
```

```
    '불가리아': '유럽연합',
```

```
    '스페인': '유럽연합',
```

```
    '아일랜드': '유럽연합',
```

```
    '에스토니아': '유럽연합',
```

```
    '이탈리아': '유럽연합',
```

```
    '체코공화국': '유럽연합',
```

```
    '크로아티아': '유럽연합',
```

```
    '포르투갈': '유럽연합',
```

```
    '폴란드': '유럽연합',
```

```
    '프랑스': '유럽연합',
```

```
    '스웨덴': '유럽연합',
```

```
    '슬로바키아': '유럽연합',
```

```
    '오스트리아': '유럽연합',
```

```
    '핀란드': '유럽연합',
```

```
    '리투아니아': '유럽연합',
```

```
    '슬로베니아': '유럽연합',
```

```
    '헝가리': '유럽연합',
```

```
    '미국': '미국',
```

```
    '베트남': '베트남',
```

```
    '일본': '일본',
```

```
    '중국': '중국'
```

```
}
```

[운송비 데이터]

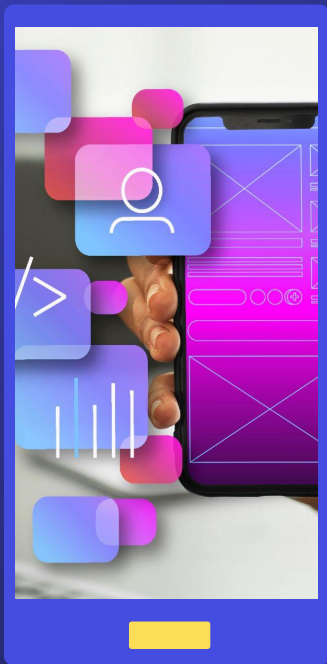
1. 유럽 국가 → 유럽연합으로 모두 매핑

2. 미국 서부와 동부는 국가가 같으므로 거리에 따른
운송비 외에는 적용되지 않을 것으로 간주

→ 미국 서부와 동부 데이터의 평균값을
미국 전체의 운송 비용이라고 한다.

```
import_cost['미국'] = (import_cost['미국서부'] + import_cost['미국동부']) / 2
```

병합 전 문제점 - 2



수출입 데이터의 HSK 품목 코드 **자리수**가 맞지 않는다.



품목코드
0106199000

기타 포유동물



품목코드
106199000

→ ???

```
[ ] # 세 개 컬럼을 문자열로 변환하고 길이 측정
df_filtered['HSK품목코드_자리수'] = df_filtered['HSK품목코드'].astype(str).str.len()

# 고유 자리수 확인
print("✅ df_filtered['HSK품목코드'] 자리수 종류:", df_filtered['HSK품목코드_자리수'].unique())

🔍 ✅ df_filtered['HSK품목코드'] 자리수 종류: [10 9]

[ ] df_filtered['HSK품목코드'] = df_filtered['HSK품목코드'].astype(str).str.zfill(10)

df_filtered['HSK품목코드'].astype(str).apply(len).unique()

🔍 array([10])
```


2.1 데이터 병합

- '수출입구분명'이 '수입'인 것만 추출
- 날짜 포맷 맞추기(pd.to_datetime)
- Left 조인이 아닌 Inner 조인 사용
- '기준년월'과 '지역'을 기준으로 병합

	기준년월	국가코드	HSK품목코드	수출입구분코드	수출입구분명	국가명	경제권명	수산물수출입품목명	당월수출입증량(킬로그램)	당월수출입미화금액(달러)	당해누계수출입증량(킬로그램)	당해누계수출입미화금액(달러)	데이터기준일자	파일기준년월	지역	HSK품목코드_자리수	기간	운송비
0	2020-01-01	AT	2501001020	I	수입	오스트리아	유럽연합/경제협력개발기구	천일염	1739	15767	1739	15767	2020-01-31	NaT	유럽연합	10	2020-01-01	1265.0
1	2020-01-01	AT	2501009010	I	수입	오스트리아	유럽연합/경제협력개발기구	식염	3350	2542	3350	2542	2020-01-31	NaT	유럽연합	10	2020-01-01	1265.0
2	2020-01-01	AT	2501009020	I	수입	오스트리아	유럽연합/경제협력개발기구	순염화나트륨	98000	29985	98000	29985	2020-01-31	NaT	유럽연합	10	2020-01-01	1265.0
3	2020-01-01	BG	1605592090	I	수입	불가리아	유럽연합	소라(조제 또는 저장처리)기타	3520	26240	3520	26240	2020-01-31	NaT	유럽연합	10	2020-01-01	1265.0
4	2020-01-01	BG	1605599090	I	수입	불가리아	유럽연합	연체동물 기타(조제 또는 저장처리)(기타)	31140	345953	31140	345953	2020-01-31	NaT	유럽연합	10	2020-01-01	1265.0

2.2 거시경제변수 결합



환율

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월' 기준 병합



유가

- 날짜 형식 변경
- 'Dubai' 유가만 사용
- Inner 조인 적용
- '기준년월'을 기준으로 병합



소비자·생산자물가지수

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월' 기준 병합



2.2 거시경제변수 결합



환율

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월' 기준 병합



유가

- 날짜 형식 변경
- 'Dubai' 유가
- Inner 조인 적용
- '기준년월' 기준 병합

	날짜	환율
0	May-20	1,228.67
1	Jun-20	1,210.01
2	Jul-20	1,198.90
3	Aug-20	1,186.85
4	Sep-20	1,178.80

2.2 거시경제변수 결합



환율

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월' 기준 병합



유가

- 날짜 형식 변경
- 'Dubai' 유가만 사용
- Inner 조인 적용
- '기준년월'을 기준으로 병합



소비자·생산자물가지수

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월' 기준 병합



2.2 거시경제변수 결합



환율

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월'을 기준으로 병합



유가

- 날짜 형식 변경
- 'Dubai' 유가만 사용
- Inner 조인 적용
- '기준년월'을 기준으로 병합



	월	유종	Dubai	Brent	WTI	Oman
0	05월	NaN	30.47	32.41	28.53	30.75
1	06월	NaN	40.80	40.77	38.31	40.85
2	07월	NaN	43.30	43.22	40.77	43.41
3	08월	NaN	44.00	45.02	42.39	44.22
4	09월	NaN	41.51	41.87	39.63	41.58

지수

2.2 거시경제변수 결합



환율

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월' 기준 병합



유가

- 날짜 형식 변경
- 'Dubai' 유가만 사용
- Inner 조인 적용
- '기준년월'을 기준으로 병합



소비자·생산자물가지수

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월' 기준 병합



2.2 거시경제변수 결합

환율

	시점	전국
0	2020.05	99.44
1	2020.06	99.71
2	2020.07	99.63
3	2020.08	100.19
4	2020.09	100.74

	시점	총지수
0	2020.05	98.96
1	2020.06	99.47
2	2020.07	99.69
3	2020.08	100.18
4	2020.09	100.38

소비자

소비자 • 생산자물가지수

- 날짜 형식 변경
- Inner 조인 적용
- '기준년월' 기준 병합

03

EDA



EDA

데이터 결합을 위해 만들었던 칼럼, 중복되는 칼럼들을 삭제

[남아있는 칼럼]

기준년월 | HSK품목코드 | 국가명 | 수산물수출입품목명 | 당월수출입중량(킬로그램) | 당월수출입미화금액(달러)

당해누계수출입중량(킬로그램) | 당해누계수출입미화금액(달러) | 지역 | 운송비 | 평균환율 | CPI | PPI

이후 결측치를 확인하였으나, 결측치는 존재하지 않았다.

	기준년월	HSK품목코드	국가명	수산물수출입품목명	당월수출입중량 (킬로그램)	당월수출입미화금액 (달러)	당해누계수출입중량 (킬로그램)	당해누계수출입미화금액 (달러)	지역	운송비	평균환율	유가	CPI	PPI
0	2020-05-01	1212211090	오스트리아	김(기타)(식용의 것)	0	0	1	8	유럽연합	1442.0	1,228.67	30.47	99.44	98.96
1	2020-05-01	1604191090	오스트리아	어류조제품(연어,청어,정어리,고등어 등 밀폐용기에넣은것)	0	0	3	73	유럽연합	1442.0	1,228.67	30.47	99.44	98.96
2	2020-05-01	2102204090	오스트리아	스리루리나 효모(기타)	0	0	2	26	유럽연합	1442.0	1,228.67	30.47	99.44	98.96
3	2020-05-01	2501001010	오스트리아	암염(岩鹽)	857	17882	857	17882	유럽연합	1442.0	1,228.67	30.47	99.44	98.96
4	2020-05-01	2501001020	오스트리아	천일염	1500	5679	5699	24589	유럽연합	1442.0	1,228.67	30.47	99.44	98.96

EDA - 기간별 운송비 변동 분석

```
[ ] import matplotlib.pyplot as plt

# 월별 지역별 평균 운송비
monthly_cost = df_ffinal[df_ffinal['운송비'].notnull()].groupby(['기준년월', '지역'])
['운송비'].mean().reset_index()

# 시각화
plt.figure(figsize=(12,6))
for region in monthly_cost['지역'].unique():
    subset = monthly_cost[monthly_cost['지역'] == region]
    plt.plot(subset['기준년월'], subset['운송비'], label=region)

plt.title("기간별 지역별 평균 운송비 추이")
plt.xlabel("기준년월")
plt.ylabel("운송비")
plt.legend()
plt.grid()
plt.show()
```



기간별로 지역별 평균 운송비가 달라진다.

전체적으로 2021년 이후 증가하고 있으며, 2023년도 이후로 감소하는 추세를 보이고 있다.

EDA - 국가별 상위 품목 분석

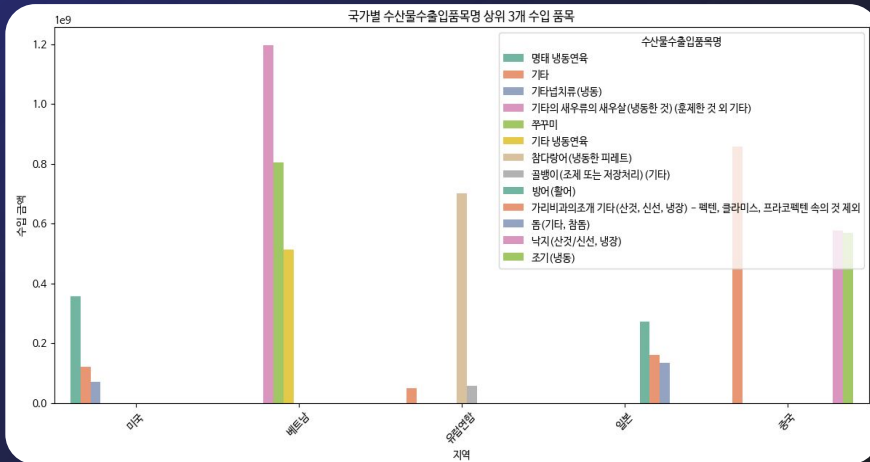
```
[ ] # 국가-HS코드별 수입 금액 합산
hs_imports = df_ffinal.groupby(['지역', '수산물수출입품목명'])
['당월수출입미화금액(달러)'].sum().reset_index()

# 상위 3개씩 추출
top_hs_by_country = (
    hs_imports
    .sort_values(['지역', '당월수출입미화금액(달러)'], ascending=[True, False])
    .groupby('지역')
    .head(3)
)
```

```
[ ] import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 6))
sns.barplot(
    data=top_hs_by_country,
    x='지역',
    y='당월수출입미화금액(달러)',
    hue='수산물수출입품목명',
    palette='Set2'
)

plt.title("국가별 수산물수출입품목명 상위 3개 수입 품목")
plt.ylabel("수입 금액")
plt.xticks(rotation=45)
plt.legend(title="수산물수출입품목명")
plt.tight_layout()
plt.show()
```



국가별로 많이 수입하는 품목이 다르다.

미국 - '명태 냉동연육', 베트남 - '기타의 새우류 새우살(냉동한 것)', 유럽연합 - '참다랑어(냉동한 피레트)', 일본 - '방어(활어)', 중국 - '가리비과의 조개 기타'를 많이 수입한다.

EDA - 품목별 수입 중량의 시간 차이 분석

```
[ ] # 가장 많이 수입된 품목 Top N개 선정
top_items = df_ffinal.groupby('수산물수출입품목명')['당월수출입중량(킬로그램)']
.sum().nlargest(5).index

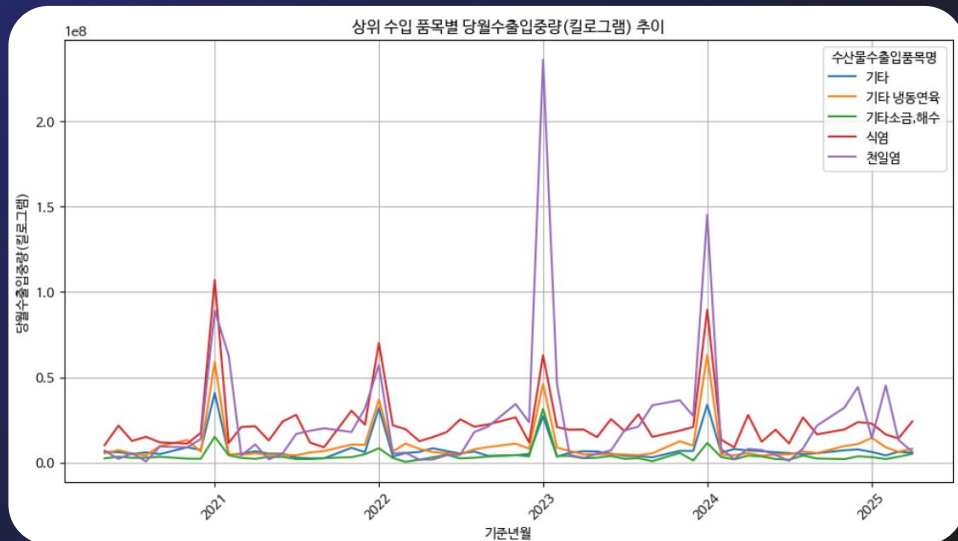
# 기간별 품목별 당월수출입중량(킬로그램)
subset = df_ffinal[df_ffinal['수산물수출입품목명'].isin(top_items)]
monthly_item = subset.groupby(['기준년월', '수산물수출입품목명'])['당월수출입중량(킬로그램)']
.sum().reset_index()

# 시각화
import seaborn as sns

plt.figure(figsize=(12,6))
sns.lineplot(data=monthly_item, x='기준년월', y='당월수출입중량(킬로그램)', hue='수산물수출입품목명')
plt.title("상위 수입 품목별 당월수출입중량(킬로그램) 추이")
plt.xticks(rotation=45)
plt.grid()
plt.show()
```

많이 수입해보는 상위 품목별로
어느 달에 많이 수입해오는지 분석해보았다.

→ 모든 품목이 연초에 수입이 증가하는 경향이 있다.



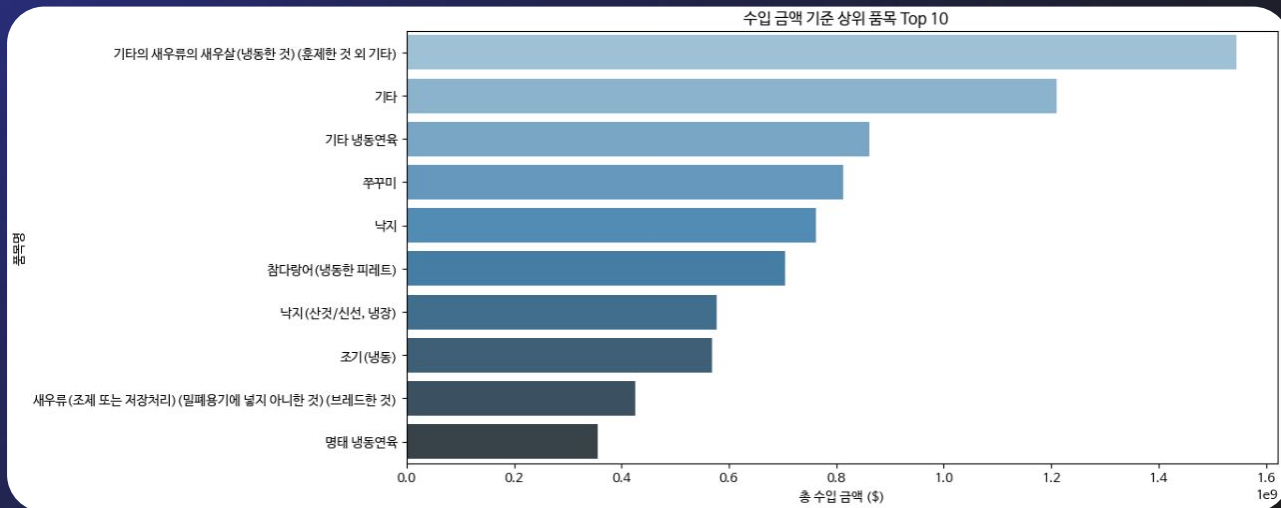
EDA - 수입 상위 품목별 금액

```
[ ] top_products = df_final.groupby('수산물수출입품목명')['당월수출입미화금액(달러)'].sum().sort_values(ascending=False).head(10)

plt.figure(figsize=(12,6))
sns.barplot(x=top_products.values, y=top_products.index, palette='Blues_d')
plt.title('🔥 수입 금액 기준 상위 품목 Top 10')
plt.xlabel('총 수입 금액 ($)')
plt.ylabel('품목명')
plt.show()
```

많이 수입해보는 상위 품목별
금액을 살펴보았다.

→ '새우살'의 총 수입금액이
'명태 냉동연육'보다 비싸다.

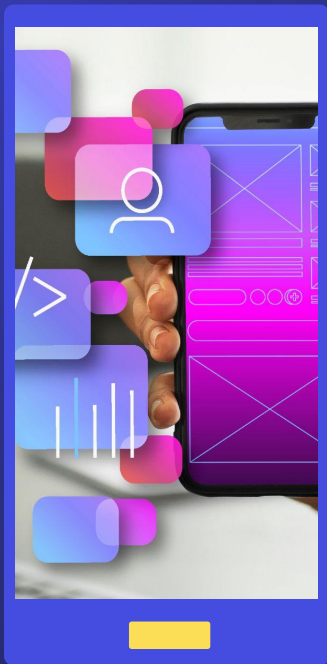


04

모의 데이터 제작



4.1 칼럼 추가



실제 단가 칼럼 제작

(당월 수입된 총 금액 ÷ 당월 수입된 총 중량)으로 실제 단가를 측정한다.

```
[ ] df_ffinal['실제단가'] = df_ffinal.apply(
    lambda row: row['당월수출입미화금액(달러)'] / row['당월수출입중량(킬로그램)']
    if row['당월수출입중량(킬로그램)'] != 0 else 0,
    axis=1
)
```



총 운송비 칼럼 제작

운송비 데이터의 단위는 2 TEU이다.

이것을 킬로그램으로 전환 후, 당월 수입된 총 중량을 곱해 총 운송비를 측정한다.

```
[ ] # 1. kg당 운송비(원) 계산
    df_ffinal['kg당 운송비(원)'] = df_ffinal['운송비'] * 1000 / 40000

    # 2. 총 운송비(원) 계산 (중량이 0이면 0)
    df_ffinal['총 운송비(원)'] = df_ffinal.apply(
        lambda row: 0 if row['당월수출입중량(킬로그램)'] == 0
        else row['당월수출입중량(킬로그램)'] * row['kg당 운송비(원)'], axis=1
    )
```

4.1 칼럼 추가

실제 단가 칼럼 제작

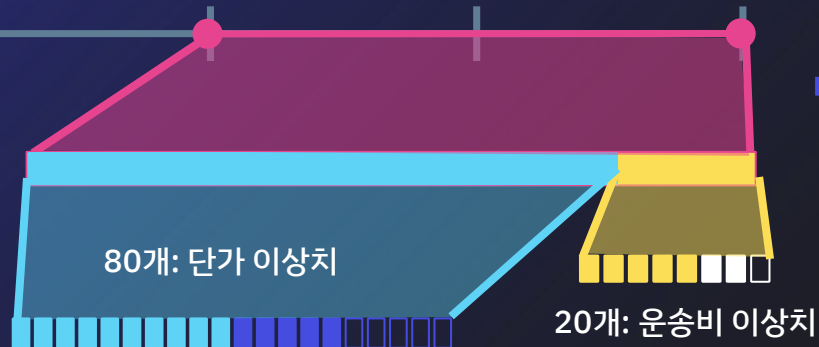
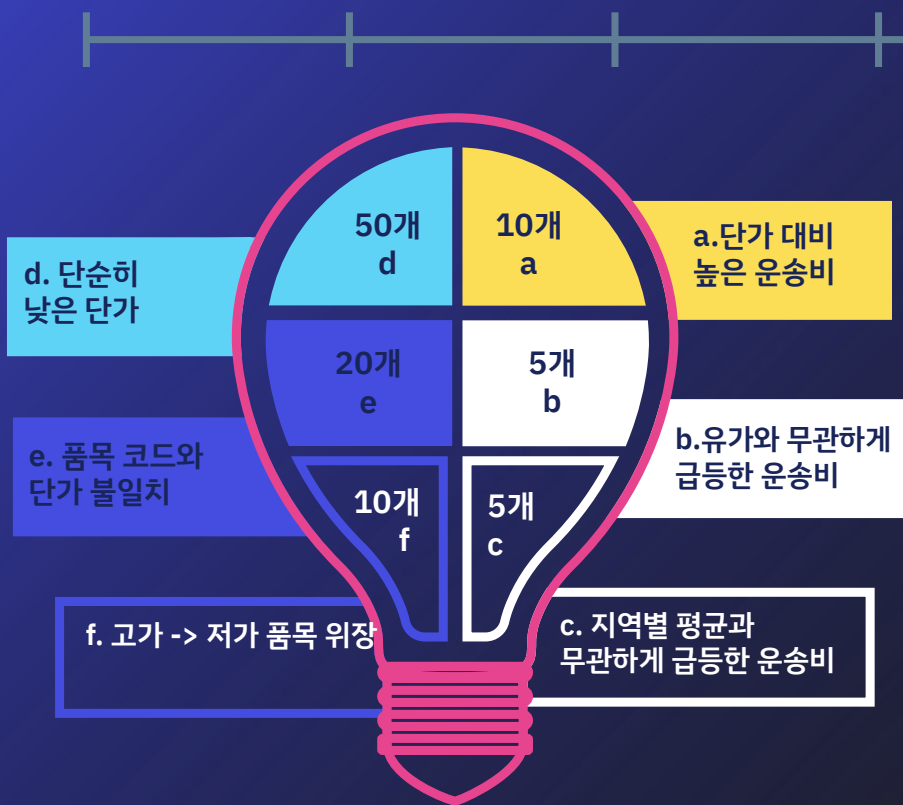
	기준년 월	HSK품목코 드	국가 명	수산물수출입품목 명	당월수출입중 량(킬로그 램)	당월수출입 미화금액(달 러)	당해누계수출 입중량(킬로그 램)	당해누계수출 입미화금액(달 러)	지 역	운송 비	평균환 율	유가	CPI	PPI	실제단가	kg당 운 송비(원)	총 운송비 (원)
0	2020-05-01	1212211090	오스 트리 아	김(기타)(식용의 것)	0	0	1	8	유럽 연합	1442.0	1,228.67	30.47	99.44	98.96	0.000000	36.05	0.00
1	2020-05-01	1604191090	오스 트리 아	어류조제품(연어,청 어,정어리,고등어 등 밀폐용기에넣은것)	0	0	3	73	유럽 연합	1442.0	1,228.67	30.47	99.44	98.96	0.000000	36.05	0.00
2	2020-05-01	2102204090	오스 트리 아	스리루리나 효모(기 타)	0	0	2	26	유럽 연합	1442.0	1,228.67	30.47	99.44	98.96	0.000000	36.05	0.00
3	2020-05-01	2501001010	오스 트리 아	암염(岩鹽)	857	17882	857	17882	유럽 연합	1442.0	1,228.67	30.47	99.44	98.96	20.865811	36.05	30894.85
4	2020-05-01	2501001020	오스 트리 아	천일염	1500	5679	5699	24589	유럽 연합	1442.0	1,228.67	30.47	99.44	98.96	3.786000	36.05	54075.00

```
[ ] # 1. kg당 운송비(원) 계산
df_ffinal['kg당 운송비(원)'] = df_ffinal['운송비'] * 1000 / 40000

# 2. 총 운송비(원) 계산 (중량이 0이면 0)
df_ffinal['총 운송비(원)'] = df_ffinal.apply(
    lambda row: 0 if row['당월수출입중량(킬로그램)'] == 0
    else row['당월수출입중량(킬로그램)'] * row['kg당 운송비(원)'], axis=1)
```

4.2 이상치 데이터 생성

* 1개년(24년 5월 ~ 25년 4월) 데이터 중 100개의 이상치 생성



❖ 운송비 이상치(20개)

- a. 10개
- b. 5개
- c. 5개

❖ 단가 이상치(80개)

- d. 50개
- e. 20개
- f. 10개

4.2 이상치 데이터 생성

* 1개년(24년 5월 ~ 25년 4월) 데이터 중 100개의 이상치 생성

	기준년 월	HSK품목코드	국가명	수산물수출입 품목명	당월수출입 중량(킬로그램)	당월수출입 미화금액 (달러)	당해누계수출 입중량(킬로그램)	당해누계수출 입미화금액(달러)	지역	평균환율	유가	CPI	PPI	실제단가	kg당 운송비 (원)	총 운송비 (원)	운송 비조 작여 부
58660	2024-12-01	305531000	중국	기타 어류(건조)	88054	827945	1105319	10706465	중국	1434.42	73.23	114.91	119.52	9.402696	14.104044	1.241917e+06	a
61093	2025-02-01	305720000	베트남	대구(가두스모르화), 대구 이외	21546	60521	42769	120187	베트남	1445.56	77.92	116.08	120.33	2.808920	4.213381	9.078150e+04	a
62065	2025-04-01	302820000	중국	가오리와 홍어(라지대과)(신선 또는 냉장)	15560	54617	74460	300261	중국	1444.31	67.74	116.38	120.14	3.510090	5.265135	8.192550e+04	a
53746	2024-07-01	2501009010	오스트리아	식염	20000	22438	103005	116995	오스트리아	1383.38	83.83	114.13	119.56	1.121900	1.682850	3.365700e+04	a

f. 고가 -> 저가 품목 위장

c. 지역별 평균과
무관하게 급등한 운송비

c. 5개

❖ 단가 이상치(80개)

d. 50개

e. 20개

f. 10개

05

모델링



5.1 모델 선택



| 기존 연구 자료 참고

- ❖ 정세연·김상철(2022)의 연구
「LSTM 오토인코더를 활용한 축산 환경
시계열 데이터의 이상치 탐지: 경계값 설정에
따른 성능 비교

: 축산 환경 시계열 데이터 이상치 탐지에서
LSTM AutoEncoder를 활용

- Dynamic Threshold 방식으로
99.66%의 높은 정확도를 기록
- 정상 패턴 복원 과정에서
이상 여부에 민감하게 반응

→ 수산물 수입 거래의 정상·이상 거래 구분에도 적합



| 모델의 특징

LSTM

: 장기 의존성을 학습하는데 강점

+

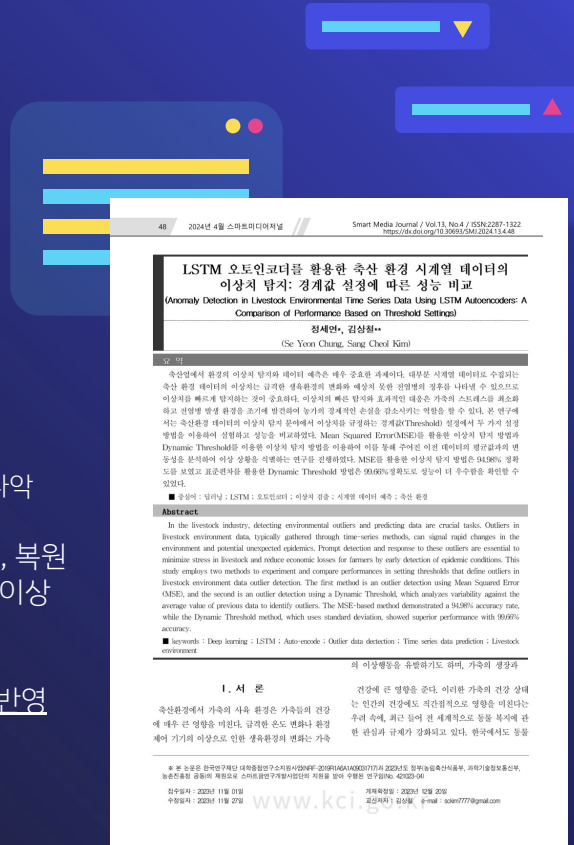
AutoEncoder

: 입력 데이터의 압축, 복원으로 핵심 패턴 파악

- ❖ 정상 거래 데이터 학습해 패턴 압축, 복원
+ 재구성 오차가 기준 이상인 경우 이상
거래로 판단

= 시계열 구조와 이상 패턴 복원 능력 동시 반영

“LSTM 기반 AE”
최종 모델로 선정



5.2 모델링 과정



1) 입력 데이터

:단순 3차원 vs 슬라이딩 윈도우 기법

입력 데이터를 단순 3차원으로 입력하여 모델링하는 방식과
슬라이딩 윈도우 기법을 적용해 품목별 12개월 단위 시퀀스를 구성하는 방식을 비교 실험

→ 거의 모든 모델에서 시퀀스 데이터 형태가 더 좋은 성능(LSTM의 장점을 살림), 채택

+ 시계열 길이 (6개월 / 12개월) 비교실험 → 12개월이 더 안정적인 성능, 채택

```
1 time_steps = 12
2 def build_sequences(df, time_steps):
3     sequences = []
4     end_indices = []
5     for code, group in df.groupby('HSK품목코드'):
6         group = group.sort_values('기준년월')
7         data = group[features].values
8         indices = group.index.values
9         if len(data) >= time_steps:
10             for i in range(len(data) - time_steps + 1):
11                 window = data[i:i+time_steps]
12                 end_idx = indices[i + time_steps - 1]
13                 sequences.append(window)
14                 end_indices.append(end_idx)
15     return np.array(sequences), end_indices
16
17
18 train_df = df[df["기준년월"] < "2024-05-01"]
19 test_df = df[df["기준년월"] >= "2024-05-01"]
20
21 X_train, train_end_idx = build_sequences(train_df, time_steps)
22 X_test, test_end_idx = build_sequences(test_df, time_steps)
23
24 test_df_seq = test_df.loc[test_end_idx].reset_index(drop=True)
```

2) 다양한 방식 시도



1. 이상치 판단 기준(Threshold): MSE 기반 5%/10% 판단하는 직관적 방법, IQR 기반 방법, (선행논문) Dynamic Threshold 방법, Threshold 튜닝 방법
2. 입력피쳐 구성: 수치형 변수만 사용하는 경우, HSK코드도 임베딩 레이어로 포함해 사용하는 경우
3. 정규화 방식: MinMaxScaler, RobustScaler
4. 딥러닝 모델 구조: Dropout, Batch Normalization, 뉴런 수와 층 수 조정 등
5. 전통적 이상치 탐지 모델 기반 방식 시도 (Isolation Forest, LOF, EllipticEnvelope 등)



5.3 최종 모델 A

- ❖ 정규화 방식: MinMaxScaler
- ❖ Threshold: MSE 기반 상위10%
- ❖ 수치형 변수만 사용
- ❖ 룰 기반 이상치 탐지 + LSTM AE
- ❖ Relu, adam, epochs=50, batch_size=64

결과: TP=8 (운송비 이상치 기준 20개 중 8개 감지)

→ 룰 기반 이상치 탐지*

: 설정한 룰 기반으로 이상치를 탐지하는 방식

설정 룰 = 지역/월별 평균 대비 편차 > 100%, 운송비/유가 비율이 상위 15% 초과

```
1 df["지역별_월평균_kg당운송비"] = df.groupby(["국가명", "지역", "기준년월"])[["kg당 운송비(원)"]].transform('mean')
2 df["운송비_평균편차"] = (df["kg당 운송비(원)"] - df["지역별_월평균_kg당운송비"]).abs() / df["지역별_월평균_kg당운송비"]
3 df["운송비_유가비"] = df["kg당 운송비(원)"] / df["유가"]
4 df["운송비_유가비_이상"] = df["운송비_유가비"] > df["운송비_유가비"].quantile(0.85)
5
6 df["운송비_스코어"] = (df["운송비_평균편차"] > 1).astype(int) + df["운송비_유가비_이상"].astype(int)
7 df["운송비_모델예측"] = (df["운송비_스코어"] >= 1).astype(int)
```

```
26 def create_lstm_autoencoder(input_shape):
27     model = models.Sequential()
28     model.add(layers.LSTM(64, activation='relu', input_shape=input_shape, return_sequences=False))
29     model.add(layers.RepeatVector(input_shape[0]))
30     model.add(layers.LSTM(64, activation='relu', return_sequences=True))
31     model.add(layers.TimeDistributed(layers.Dense(input_shape[1])))
32     model.compile(optimizer='adam', loss='mse')
33     return model
34
35 input_shape = X_train.shape[1:]
36 model = create_lstm_autoencoder(input_shape)
37 model.fit(X_train, X_train, epochs=50, batch_size=64, validation_data=(X_test, X_test), verbose=0)
```

```
1 reconstructed = model.predict(X_test)
2 mse = np.mean(np.power(X_test - reconstructed, 2), axis=(1, 2))
3 threshold = np.percentile(mse, 85)
4 lstm_outliers = (mse > threshold).astype(int)
5
6 test_df_seq["is_outlier_lstm"] = lstm_outliers
```

[룰 기반 평가]				
	precision	recall	f1-score	support
0	0.9987	0.9831	0.9908	6867
1	0.0482	0.4000	0.0876	15
accuracy			0.9818	6882
macro avg	0.5239	0.6916	0.5392	6882
weighted avg	0.9966	0.9818	0.9989	6882
룰 기반 TP: 6				
[LSTM AE 평가]				
	precision	recall	f1-score	support
0	0.9988	0.8507	0.9188	6867
1	0.0077	0.5333	0.0153	15
accuracy			0.8500	6882
macro avg	0.5033	0.6920	0.4671	6882
weighted avg	0.9966	0.8500	0.9169	6882
LSTM-AE TP: 8				
[룰 + LSTM 최종 결합 평가]				
	precision	recall	f1-score	support
0	0.9988	0.8405	0.9129	6867
1	0.0073	0.5333	0.0143	15
accuracy			0.8399	6882
macro avg	0.5030	0.6869	0.4636	6882
weighted avg	0.9966	0.8399	0.9109	6882
최종 결합 TP: 8				

5.3 최종 모델 B

- ❖ 정규화 방식: RobustScaler
- ❖ Threshold: IQR 기반 임계값 (Q1, Q3 밖)
- ❖ 칼럼 모두 사용: HSK 코드를 정수 인코딩 + 임베딩 레이어를 LSTM에 병합
- ❖ 인코딩: Ordinal Encoder
- ❖ LSTM AE (입력 데이터: 수치+임베딩-16차원)
- ❖ tanh, adam, epochs=50, batch_size=64

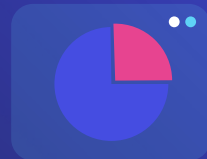
결과: TP=7, 31

운송비 이상치: 20개 중 8개 감지, 단가 이상치: 80개 중 31개 감지
Precision, Recall = (0.003, 0.768), (0.012, 0.754)

```
[ ] 1 # 운송비
2 precision_운송비 = len(tp_운송비) / merged_all[merged_all["is_outlier"] == 1].shape[0]
3 recall_운송비 = len(tp_운송비) / merged_all[merged_all["운송비_실제이상"] == 1].shape[0]
4 print(f"운송비 Precision: {precision_운송비:.3f}, Recall: {recall_운송비:.3f}")
5
6 # 단가
7 precision_단가 = len(tp_단가) / merged_all[merged_all["is_outlier"] == 1].shape[0]
8 recall_단가 = len(tp_단가) / merged_all[merged_all["단가_실제이상"] == 1].shape[0]
9 print(f"단가 Precision: {precision_단가:.3f}, Recall: {recall_단가:.3f}")
```

운송비 Precision: 0.003, Recall: 0.768
단가 Precision: 0.012, Recall: 0.754

```
1 # LSTM Autoencoder 모델 정의 (임베딩 통합)
2 def create_lstm_autoencoder_with_embedding(time_steps, num_features, num_codes, embedding_dim):
3     input_data = Input(shape=(time_steps, num_features), name='수치입력')
4     input_code = Input(shape=(time_steps,), dtype='int32', name='코드입력')
5
6     x_embed = Embedding(input_dim=num_codes + 1, output_dim=embedding_dim)(input_code)
7     x = Concatenate()([input_data, x_embed])
8
9     x_encoded = layers.LSTM(128, activation='tanh', return_sequences=True)(x)
10    x_encoded = layers.LSTM(64, activation='tanh', return_sequences=False)(x_encoded)
11
12    x_decoded = layers.RepeatVector(time_steps)(x_encoded)
13    x_decoded = layers.LSTM(64, activation='tanh', return_sequences=True)(x_decoded)
14    x_decoded = layers.LSTM(128, activation='tanh', return_sequences=True)(x_decoded)
15    output = layers.TimeDistributed(layers.Dense(num_features))(x_decoded)
16
17    model = Model(inputs=[input_data, input_code], outputs=output)
18    model.compile(optimizer='adam', loss='mse')
19    return model
```



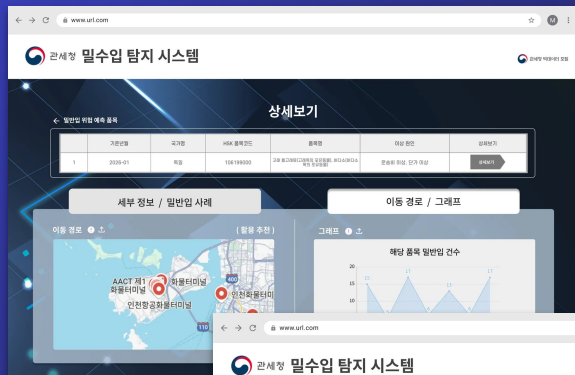
06

웹 서비스 데모

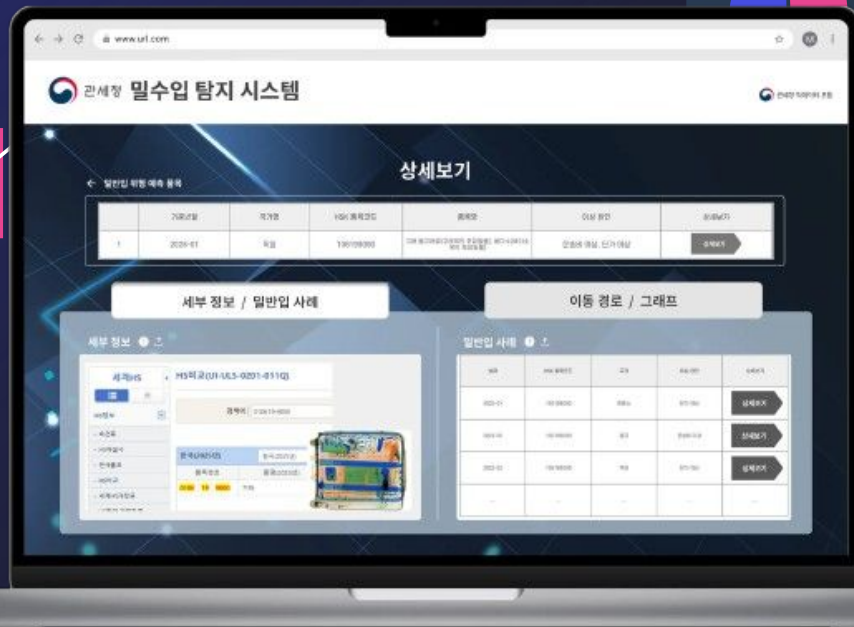


웹 서비스 화면

이후 확장 가능 서비스 (세부 정보)



이후 확장 가능 서비스 (이동 경로)



기준 설정 후 조회 가능

07

차별성 및 기대효과



7.1 기존 서비스와의 차별성



지도학습 대비 **유연성** **높은** 비지도 학습 구조

- 과거 사례에 국한되지 않고 새로운 유형의 밀수 수법에 선제적으로 대응 가능
- 데이터 편향 및 과적합의 위험을 낮춰 다양한 상황에 유연하게 적용 가능



정형 수치 데이터를 활용한 다변량 이상거래 탐지

- 단가, 중량, 운송비, 관세율 등 다양한 수치형 정보 기반 탐지
- 고도화된 정량적 판별을 통해 단순 텍스트 분석보다 신뢰도 향상



범용성과 확장성을 가진 플랫폼 구조

- 본 분석에서 적용된 수산물 분야를 넘어 농산물, 생활용품, 담배, 의류 등 다양한 품목과 국가에 적용 가능
- 장기적으로는 다양한 목적에 활용 가능한 통합형 탐지 시스템 발전 가능



비정형, 시계열 패턴까지 대응하는 정교한 이상탐지

- LSTM 기반 AE 알고리즘으로 명확한 규칙에 부합하지 않는 거래도 탐색 가능
- 단발성 이상뿐 아니라 점진적, 복합적 패턴 변화 포착 가능
- 계절성과 시기별 단가 변동 등 시계열적 특징을 반영해 더욱 정교한 탐지 가능



AI 기반 자동화로 인한 효율성과 비용 절감 효과

- 기존 인력 중심 대응 방식 대비 분석 속도와 정확도 향상 가능, 인적 자원 부담 경감
- 운영 효율성 높여 지속 가능한 관세행정 운영 기반 마련



7.2 기대 효과



- **대응 품목의 확장성과 시스템 유연성 확보**

다양한 품목군에 대한 적용을 염두에 둔 모델 구조. 동일한 구조로 마약, 담배, 의약품, 귀금속 등 다양한 고위험 품목으로 확장 적용 가능

- **통관 행정의 정밀도 및 효율성 향상**

통관 과정 전반에 AI기반 자동화 탐지 체계를 도입하여 기존의 인력 중심 사후 적발 위주의 밀수 대응 체계를 데이터 기반의 선제적 대응 구조로 전환 역할. 기술적으로 정밀하고 신속한 구조로 고도화 가능

- **기술 일자리 창출 및 산업적 파급 효과**

데이터 분석, 머신러닝 모델링, 리스크 진단 등의 전문인력 수요확대. 민간 기술 기업 및 스타트업의 시장 진입 기회 제공





감사합니다 