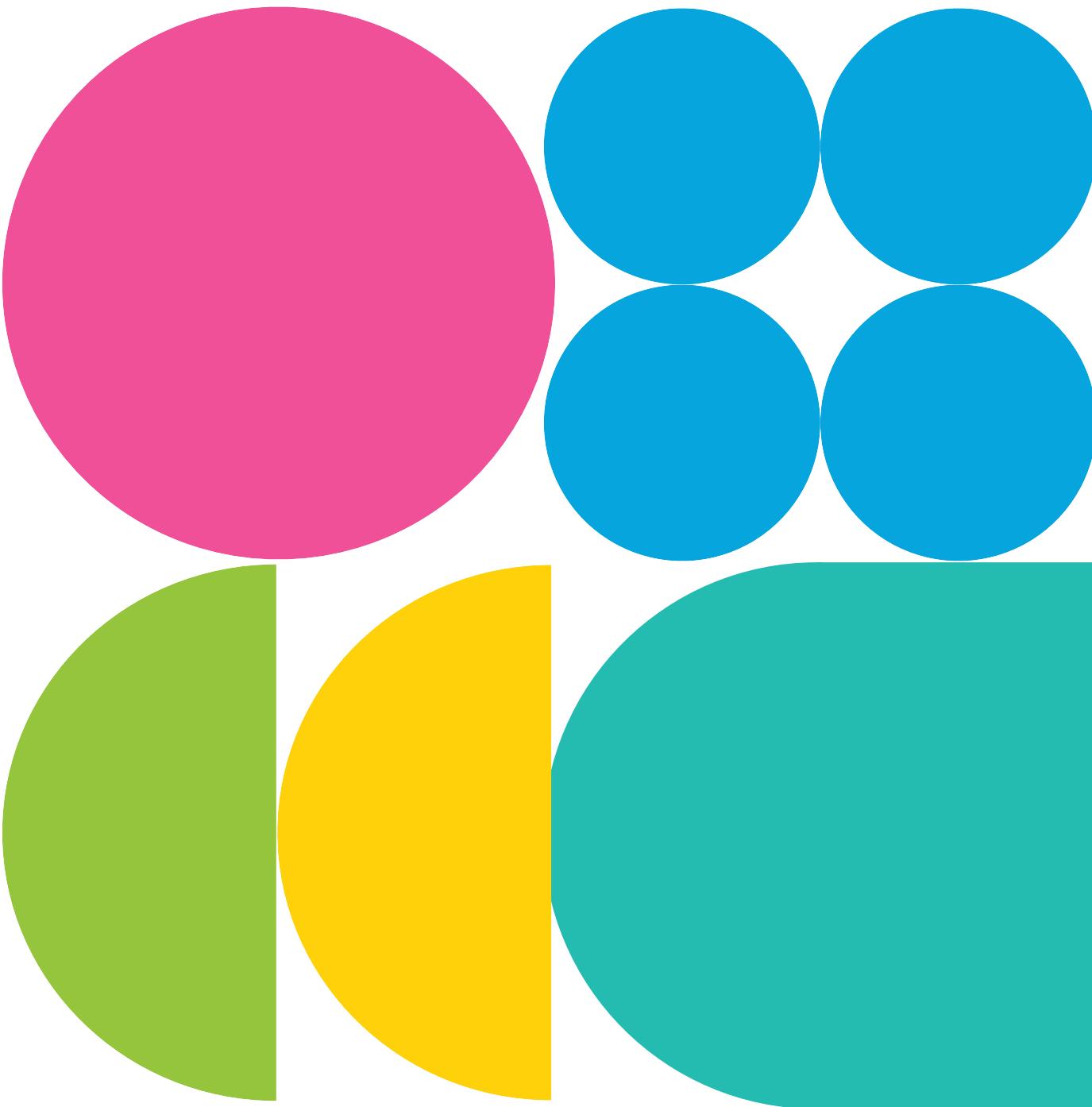


# Sports Drink

## 점유율 예측

고정진 백영현 조소현 천고은 최덕진 최성준



# CONTENTS

01 | 프로젝트 개요

04 | 데이터 시각화

02 | 데이터 수집 & 저장

05 | 결론 및 한계점

03 | 데이터 분석 & 모델링

06 | Q & A

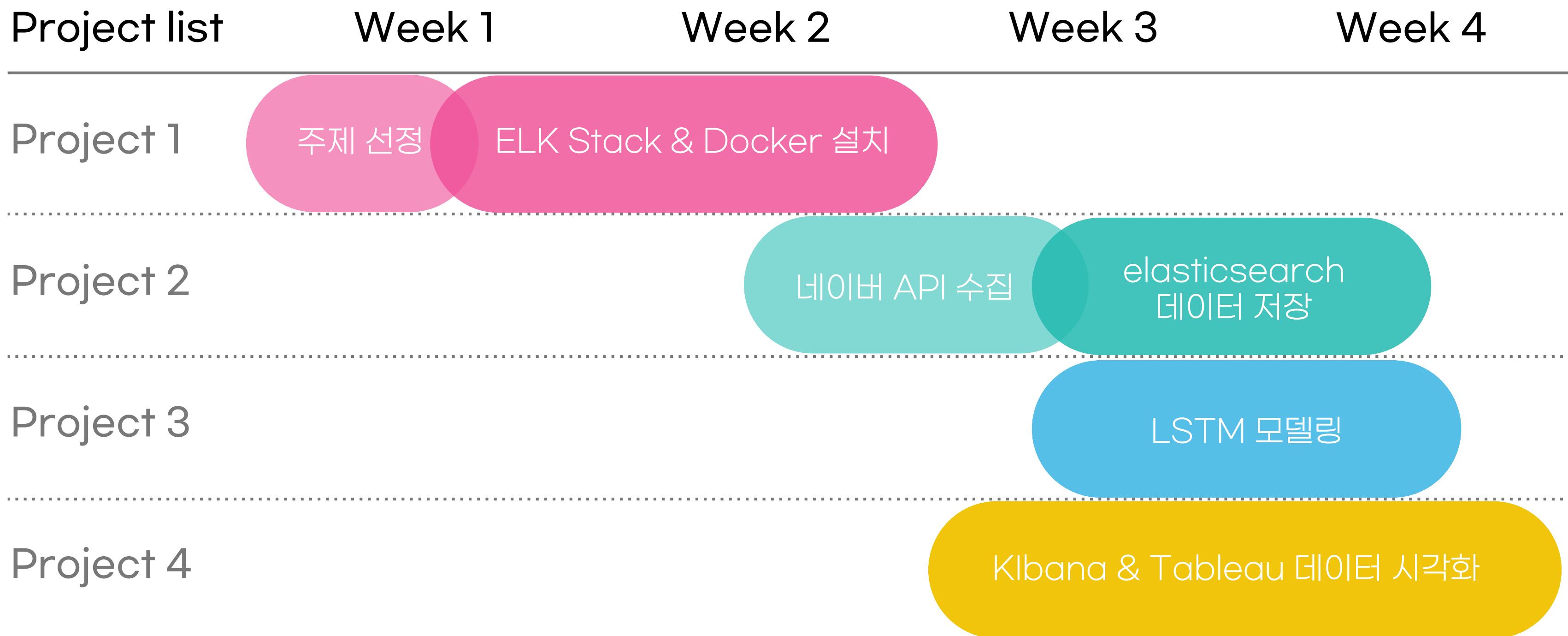


01

# 프로젝트 개요

PLAN

# 프로젝트 계획



## OBJECTIVES

# 프로젝트 목표

## 네이버 검색 트렌드 분석

2024년 동안의 네이버 검색 트렌드를 분석하여 스포츠 음료 브랜드별, 성별, 나이대별 검색량의 패턴을 파악



## LSTM을 활용한 검색량 예측

2025년 스포츠 음료 검색량을 예측하고 기상청 날씨 데이터를 연계하여 예측 정확도를 향상

## ELK기반 데이터 저장 & 시각화

네이버 검색량 API 데이터를 수집하고 Elasticsearch에 저장하여 Kibana로 실시간 데이터를 시각화하고 모니터링

## 심층 분석 및 인사이트 도출

Tableau를 활용하여 데이터를 심층 분석하고, 시각적 인사이트를 도출하여 비즈니스 의사결정에 활용 가능한 정보를 제공

## OVERVIEW

# 프로젝트 개요

데이터 수집 & 저장

**NAVER** OpenAPI

검색량 데이터 수집

 **logstash**

데이터 변환 및 필터링

 **elasticsearch**

데이터 저장

데이터 분석 & 모델링

**LSTM** 모델

미래 검색량 데이터 예측

 **기상청 기상자료개방포털**

날씨 데이터 연계

 **slack**

검색량 변화 알림 메시징 플랫폼

데이터 시각화

 **kibana**

데이터 시각화  
로그 데이터 모니터링

 **+ a b | e a u**

다양한 데이터 소스와  
연결 가능

## OVERVIEW

# 데이터 선정 이유

## 초기 계획과 문제점

- 밀키트 브랜드별 검색량 분석 시도
- 브랜드명 검색 부족 문제 발견
- 사용자 검색 패턴 분석 필요성 인식

## 해결 방안 및 과정

- 이온음료 브랜드로 분석 대상 변경
- 네이버 검색 API 활용 데이터 수집
- 브랜드, 성별, 연령별 데이터 확보

## OVERVIEW

# ELK Stack 선정 이유



1

## 대용량 데이터 수집 및 전처리의 용이

Elasticsearch는 분산형 검색 엔진으로, 대규모 데이터셋에서 빠른 검색과 분석이 가능



2

## 다양한 분석 및 인사이트 도출

LSTM 모델을 통해 예측한 결과와 결합하여 실시간으로 변동성 있는 데이터를 분석



3

## 실시간 모니터링 및 대시보드

Kibana는 실시간 대시보드를 통해 데이터를 시각화하고 모니터링할 수 있는 기능을 제공

## OVERVIEW

# Docker로 ELK Stack 실행

## # 설치 및 설정의 용이성

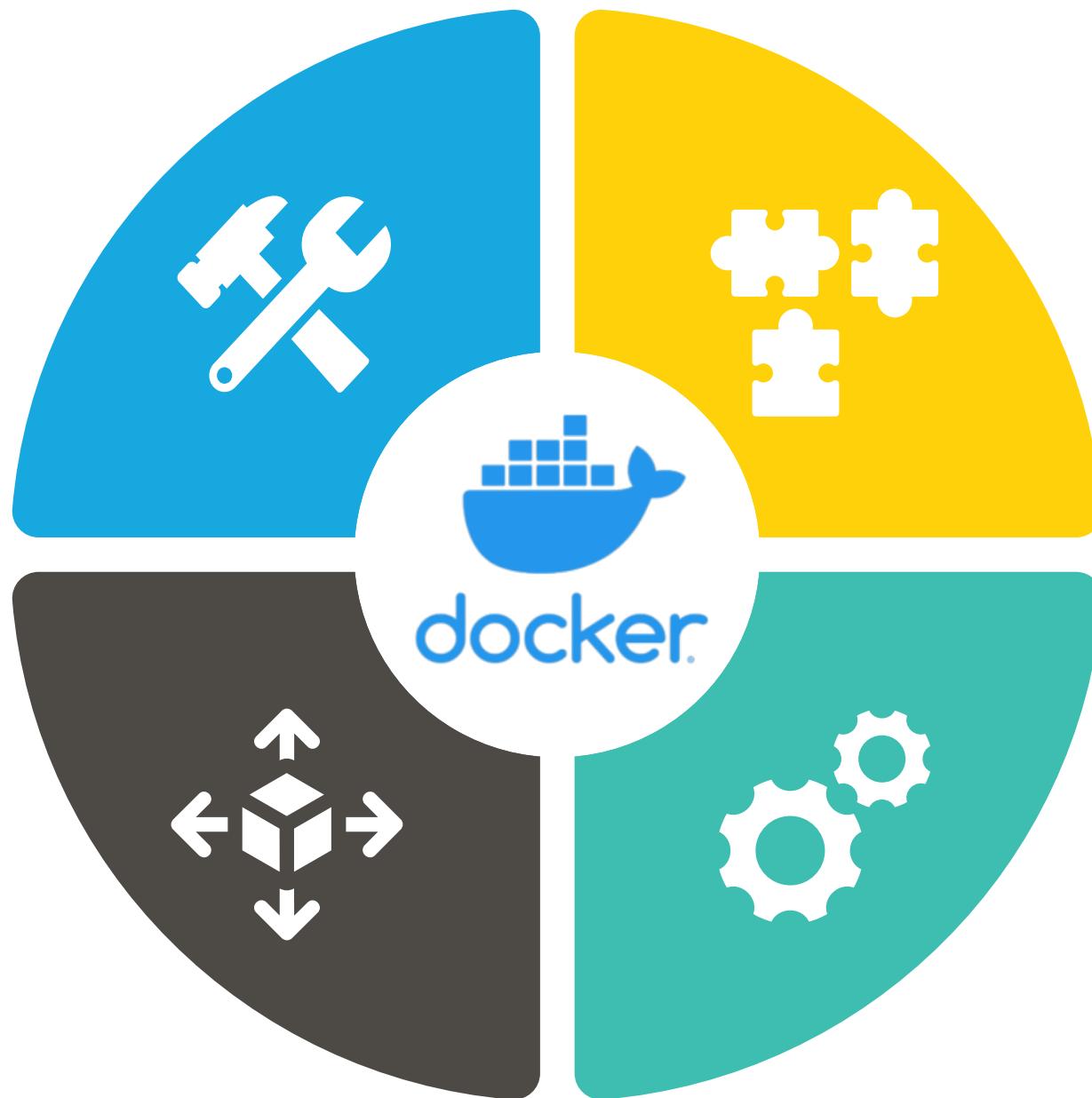
ELK 스택을 빠르고 간단하게 설치하고 설정  
Docker 이미지와 docker-compose로  
여러 컨테이너를 손쉽게 관리

## # 이식성

운영 체제에 관계없이 동일한 환경을 유지  
Windows, macOS, Linux 등 다양한 환경에서  
ELK 스택을 쉽게 이식하고 실행

## # 환경 격리와 일관성

Docker는 각 구성 요소를 독립적인 컨테이너에  
서 실행하여 격리된 환경을 제공하고,  
개발, 테스트, 프로덕션 환경을 일관되게 유지



## # 확장성과 관리 용이성

Docker는 ELK 스택을 쉽게 스케일링하고 관리  
할 수 있으며, 여러 컨테이너를 클러스터로 확장

The screenshot shows the Docker Desktop interface with the following details:

- Left Sidebar:** Contains links for Containers, Images, Volumes, Builds, Docker Hub, Docker Scout, and Extensions.
- Top Bar:** Includes a search bar, a "Ctrl+K" button, notification icon (3 notifications), settings gear, and a "Sign in" button.
- Central Area:** Displays the "docker-elk" stack at path `C:\ITWILL\Final_project\elk\docker-elk`. It lists four containers:
  - docker-elk-setup-1**: docker-elk-setup, status: Running, ports: 50000:50000, Show all ports (4)
  - docker-elk-logstash**: docker-elk-logstash, status: Running, ports: 50000:50000, Show all ports (4)
  - docker-elk-kibana-1**: docker-elk-kibana, status: Running, ports: 5601:5601
  - docker-elk-elasticsearch**: docker-elk-elasticsearch, status: Running, ports: 9200:9200, Show all ports (2)
- Terminal Window:** Shows log output for the kibana-1 container, indicating successful tasks and fleet usage reporting.
- Bottom Status Bar:** Shows "Engine running", RAM/CPU/Disk usage, and a "Terminal" button.

02

# 데이터 수집 & 저장

데이터 수집

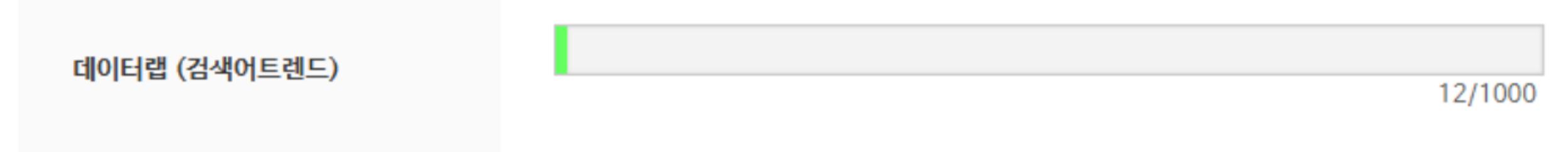
# 네이버 검색 API Application 등록

## 애플리케이션 정보

Client ID	x6bPpM45bpUQUgXrMdzx
Client Secret	..... <a href="#">보기</a>

## 비로그인 오픈 API 당일 사용량

API호출량/일일허용량



## 데이터 수집

# 네이버 API를 통한 데이터 수집

```
# 네이버 API에서 데이터 수집
def fetch_data(gender, ages):
    results = {}
    for batch in [sports_drink[i:i + 5] for i in range(0, len(sports_drink), 5)]:
        body = json.dumps({
            "startDate": "2024-01-01",
            "endDate": today,
            "timeUnit": "date",
            "keywordGroups": batch,
            "device": "",
            "ages": ages,
            "gender": gender
        })

        request = urllib.request.Request(url)
        request.add_header("X-Naver-Client-Id", client_id)
        request.add_header("X-Naver-Client-Secret", client_secret)

        try:
            response = urllib.request.urlopen(request, data=body.encode("utf-8"))
            if response.getcode() == 200:
                data = json.loads(response.read().decode('utf-8'))
                ...
                results[period][group["title"]] += ratio
        except Exception as e:
            print(f"Error occurred: {e}")

    return results
```

## 데이터 수집

# 네이버 API를 통한 데이터 수집

```
# 네이버 API에서 데이터 수집
def fetch_data(gender, ages):
    results = []
    for batch in sports_drink[::5] for i in range(0, len(sports_drink) - 5, 5):
        body = json.dumps({
            "startDate": "2024-01-01",
            "endDate": today,
            "timeUnit": "date",
            "keywordGroups": batch,
            "device": "",
            "ages": ages,
            "gender": gender
        })
        request = urllib.request.Request(url)
        request.add_header("X-Naver-Client-Id", client_id)
        request.add_header("X-Naver-Client-Secret", client_secret)

        try:
            response = urllib.request.urlopen(request, data=body)
            if response.getcode() == 200:
                data = json.loads(response.read().decode('utf-8'))
                ...
                results[period][group["title"]] += ratio
        except:
            pass
    return results
```

## # API에 보내는 주요 파라미터

startDate : 데이터 시작날짜  
endDate : 데이터를 가져올 종료 날짜  
timeUnit : 구간 단위  
keywordGroups : 주제어와 검색어 묶음  
device : 검색환경에 따른 조건  
gender : 사용자의 성별  
ages : 사용자의 연령

데이터 수집

# 네이버 API를 통한 데이터 수집

```
# 네이버 API에서 데이터 수집
def fetch_data(gender, ages):
    results = []
    for batch in range(1, 10):
```

# 스포츠 음료 키워드 그룹

```
sports_drink = [
    {"groupName": "포카리스웨트", "keywords": ["포카리", "포카리스웨트", "포카리 스웨트"]},
    {"groupName": "게토레이", "keywords": ["게토레이", "게토 레이", "Gatorade", ]},
    {"groupName": "파워에이드", "keywords": ["파워에이드", "파워 에이드", "Power Ade"]},
    {"groupName": "토레타", "keywords": ["토레타", "토레타!", "Toreta"]},
    {"groupName": "링티", "keywords": ["링티", "Lingtea", "LINGTEA", "lingtea"]}
]
```

...

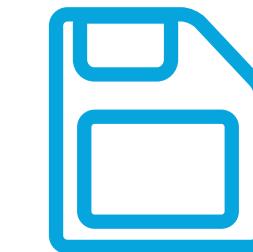
```
        results[period][group["title"]] += ratio
return results
```

데이터 저장

# Elasticsearch에 API 데이터 저장

```
# Elasticsearch 클라이언트 설정
es = Elasticsearch("http://localhost:9200")
index_name = "sports_drink_search" # 저장할 인덱스 이름

# Elasticsearch 저장 함수
def save_to_elasticsearch(index, period, gender,
                           age_group, group_ratios):
    for brand, ratio in group_ratios.items():
        doc = {
            "period": period,
            "gender": gender,
            "age_group": age_group,
            "brand": brand,
            "ratio": ratio,
            "timestamp": datetime.now().isoformat()
        }
        es.index(index=index, document=doc)
```



Data view sports\_drink\_search ✓

```
{
  "brand": ["포카리스웨트"],
  "period": ["2024-01-01T00:00:00.000Z"],
  "gender": ["male"],
  "age_group": ["30대"],
  "ratio": [31.39],
  "timestamp": ["2025-02-14T15:27:28.601Z"],
}
```

데이터 저장

# Elasticsearch에 기상 데이터 저장

# 기상관측.csv 데이터 전처리

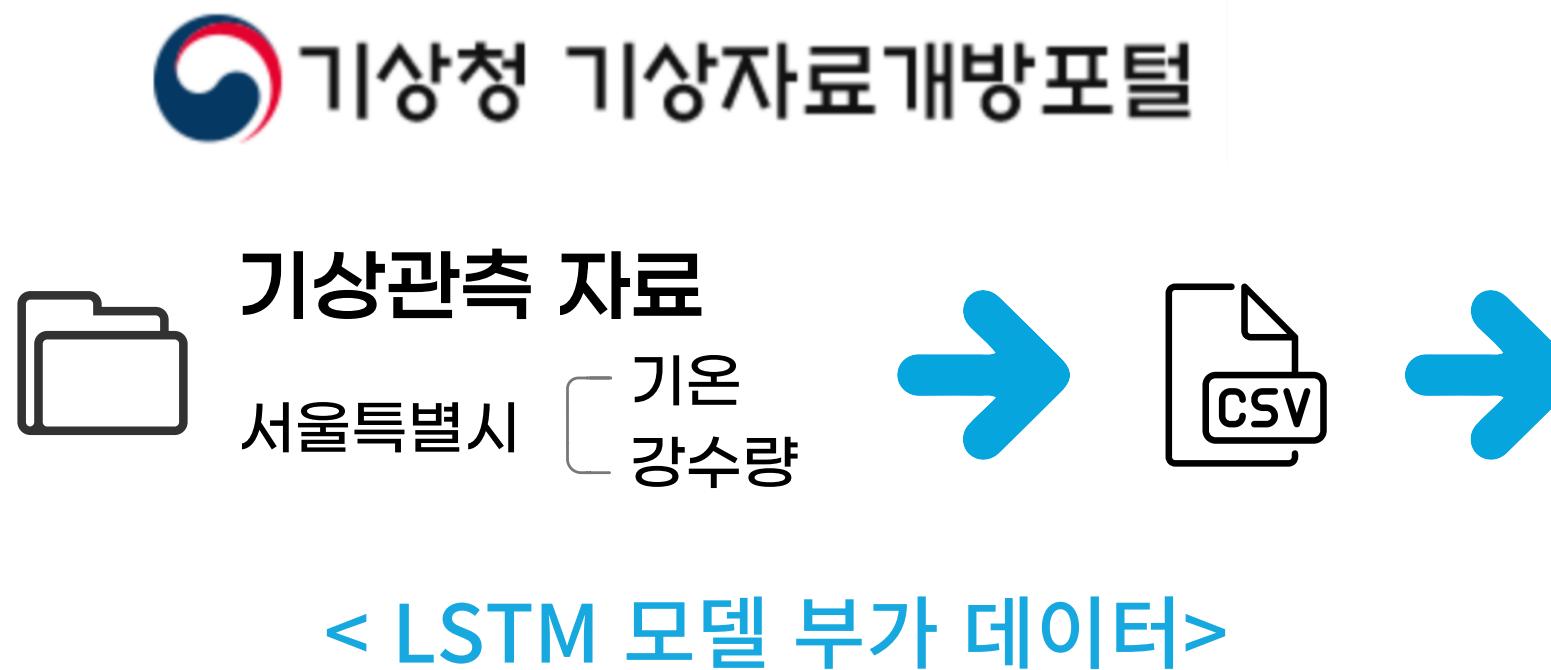
```
def load_weather_data(file_path):
    try:
        df = pd.read_csv(file_path)

        # 컬럼명 변경
        df = df.rename(columns={
            "period": "period",
            ...
        })[["period", "temp_avg", "rainfall"]]

        # 날짜 변환 (Period → 문자열 YYYY-MM-DD)
        df["period"] = pd.to_datetime(df["period"]).dt.strftime("%Y-%m-%d")

        # 결측값 처리
        df["rainfall"].fillna(0, inplace=True)
        ...
        df["rainfall"] = df["rainfall"].astype(float)

        logging.info("기상 데이터 로드 및 전처리 완료")
        print("기상 데이터 로드 및 전처리 완료")
        return df
    ...
```



데이터 저장

# Elasticsearch에 기상 데이터 저장

```
# Elasticsearch 연결 설정
ES_HOST = "http://localhost:9200"
weather_index = "sports_drink_weather"

# Elasticsearch에 데이터를 업로드
def upload_to_elasticsearch(es, df, index_name):
    try:
        records = df.to_dict(orient="records")
        actions = [{"_index": index_name,
                    "_source": record} for record in records]
        helpers.bulk(es, actions)
        logging.info(f"{index_name} 인덱스에 데이터 업로드 완료!")
    ...

```



Data view sports\_drink\_weather ▾

```
{
  "period": ["2024-01-01T00:00:00.000Z"],
  "rainfall": [0],
  "temp_avg": [3.3],
}
```

데이터 수집

# 미래 중기예보 API를 통한 데이터 수집

오픈API 상세



XML JSON 기상청\_중기예보 조회서비스

활용신청

중기전망, 중기육상예보, 중기기온, 중기해상예보 정보를 조회하는 서비스

15 0 관리

다른 사용자들이 활용한 데이터

일반 인증키  
(Encoding)

k7YYvo%2FQpscTH00rba7pGcV17DkFi0awdPCD3ywZ%2FINCtPO86pwz8xc1zdlo%2BCA%2BAhZRmMy5RfKE15O27onr1w%3D%3D

데이터 수집

# 미래 중기예보 API를 통한 데이터 수집

## # API 인증키

```
SERVICE_KEY=unquote("k7YYvo%2FQpscTH00rba7pGcV17DkFi0awdPCD3ywZ%2FINCt  
PO86pwz8xc1zdlo%2BCA%2BAhZRmMy5RfKE15O27onrlw%3D%3D")
```

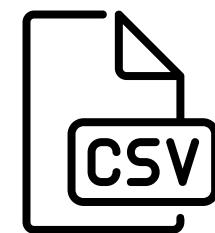
## # 요청 파라미터

```
params_ta = {  
    "serviceKey": SERVICE_KEY,  
    "numOfRows": 10,  
    "pageNo": 1,  
    "dataType": "JSON"  
}
```

```
params_land = {  
    "serviceKey": SERVICE_KEY,  
    "numOfRows": 10,  
    "pageNo": 1,  
    "dataType": "JSON"  
}
```

## # API 요청

```
response_ta = requests.get(BASE_URL_TA, params=params_ta)  
response_land = requests.get(BASE_URL_LAND, params=params_land)
```



future\_weather  
\_forecast.csv

03

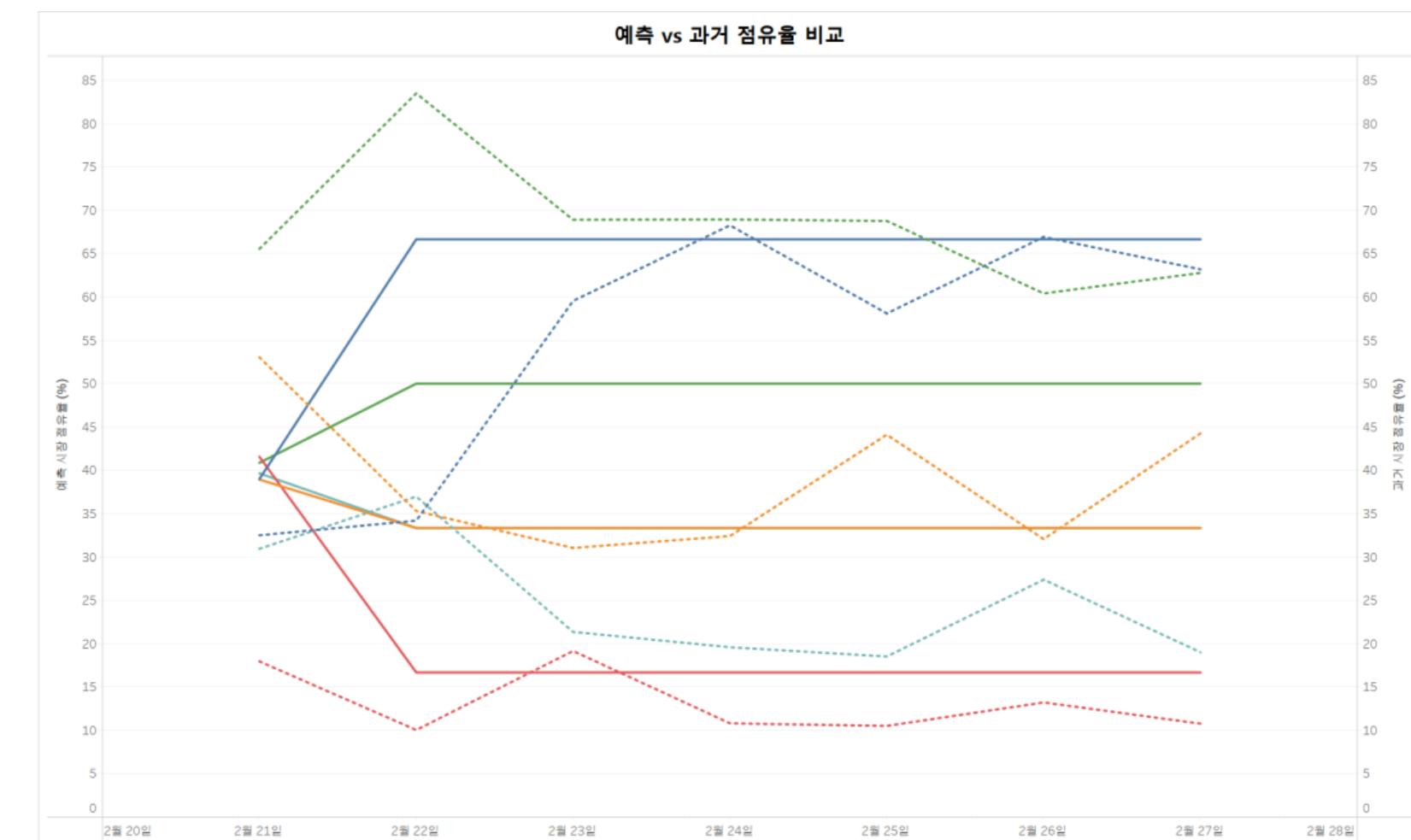
# 데이터 분석 & 모델링

**데이터 분석**

# LSTM 모델

## 시계열 데이터 예측에 특화된 딥러닝 모델

순환 신경망(RNN)의 한 종류인 LSTM은 장기적인 의존성을 학습할 수 있는 구조로 **과거의 정보가 중요한 예측에 효과적으로 사용**될 수 있도록 설계되었습니다. 이 모델은 기본 RNN의 기울기 소실 문제를 해결하여 긴 시퀀스 데이터를 다루는 데 강점을 지닙니다.



스포츠 음료 검색량 예측 결과

## 데이터 분석

# LSTM 모델을 활용한 검색량 예측



## LSTM 모델 선정 이유

검색량은 시간에 따라 변화하는 시계열 데이터로, LSTM 모델은 시간에 따른 검색량의 트렌드와 변화를 효과적으로 예측합니다.



## 기상관측 데이터 연계 이유

기상 데이터는 검색량에 영향을 미칠 수 있는 중요한 외부 변수입니다. LSTM 모델에 연계하여 예측 정확도를 높이고 더 정교한 결과를 도출할 수 있습니다.

데이터 분석

# LSTM 모델 학습



## elasticsearch

Data view sports\_drink\_search ▾

Data view sports\_drink\_weather ▾

### # LSTM 모델 주요 설정

LSTM 유닛 수 : 첫 번째 레이어 128, 두 번째 레이어 64

활성화 함수 : ReLU (비선형성 추가)

드롭아웃 : 30% 드롭아웃 (과적합 방지)

최적화 알고리즘 : Adam (학습률 0.0005)

손실 함수 : MSE (회귀 문제에 적합)

시퀀스 길이 : 7일

배치 크기 : 32

에포크 수 : 100

스케일러 : MinMaxScaler

데이터 분석

# LSTM 모델 최적화 요소

추가 기능	효과
유닛 수 조정 : (128 → 64, 64 → 32)	가벼운 모델로 성능 유지하면서 학습 속도 향상
Dense(32, activation='relu')	비선형 관계 학습을 강화하여 더 정교한 예측 가능
ReduceLROnPlateau	학습 속도가 너무 빠를때 자동으로 학습률 감소 -> 더 나은 최적화
ModelCheckpoint	가장 좋은 모델 가중치를 저장하여 최상의 결과 유지
EarlyStopping	불필요한 학습 방지 -> 학습 시간 단축 + 과적합 방지

## 데이터 분석

# 데이터 크기에 따른 최적의 비율

데이터	추천 비율	이유
100,000개 이상 (빅데이터)	80%:10%:10%	훈련 데이터를 최대한 활용해 모델 성능 향상
10,000개 이상 (중간 규모)	70%:15%: 15%	모델 학습과 평가의 균형 유지
1,000~10,000개(작은 데이터)	60%:20%:20%	검증/데스트 데이터를 충분히 확보하여 모델평가 신뢰성 유지
1,000개 이하(매우 작은 데이터)	50%:25%:25%	데이터가 적으로 일반화 성능을 평가하는 것이 중요

## 데이터 분석

# 최종 독립변수

## # 최종 독립변수 리스트 구성

```
feature_cols = [col for col in df.columns if col.startswith("ratios.") or col in ["temp_avg", "rainfall"]]
feature_cols += [col for col in df.columns if "dayofweek_" in col] # 요일 변수 유지
feature_cols += ["lag_7", "lag_14", "lag_30"] # 시차 변수 추가
```

그룹	포함된 변수	의미
비율 데이터 & 날씨 데이터	ratios.* , temp_avg , rainfall	점유율 비율 & 기상 데이터
계절성 정보	month_* , quarter_* , dayofweek_*	월/분기/요일 원-핫 인코딩
시차 변수(Lag Features)	lag_7, lag_14 , lag_30	7일/14일/30일 전 점유율

데이터 분석

# LSTM 모델 저장

```
# LSTM 모델 학습 함수
def train_lstm_model(df, feature_cols, save_path):
    scaler = MinMaxScaler()
    ...
    model = Sequential([
        LSTM(128, activation='relu', return_sequences=True, ...)),
        ...
        Dense(len(feature_cols))
    ])
    model.compile(optimizer=Adam(learning_rate=0.0005), loss='mse')
    ...
    model.fit(X, y, epochs=100, batch_size=32, verbose=1, ...)
    with open(os.path.join(save_path, "scaler.pkl"), "wb") as f:
        pickle.dump(scaler, f)
    model.save(os.path.join(save_path, "lstm_model.h5"))
    print(f"{save_path} 모델 및 스케일러 저장 완료!")
```

# 학습된 모델과 스케일러 저장



getorei\_10dae\_male



lstm\_model.h5



scaler.pkl

•

•

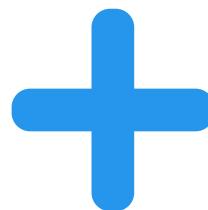


toreta\_60dae\_isang\_male

데이터 분석

# 저장된 LSTM 모델로 점유율 예측

getorei\_10dae\_male  
lstm\_model.h5  
scaler.pkl



future\_weather  
\_forecast.csv



# 예측 결과

date	brand	age	gender	predicted
2025-02-21	게토레이	10대	남성	17.44
2025-02-21	링티	10대	남성	16.28
2025-02-21	포카리스웨트	10대	남성	22.67
2025-02-21	토레타	10대	남성	26.16
		...		
2025-02-27	토레타	60대 이상	여성	16.67

데이터 분석

# Slack 연동 검색량 변화량 알림

LSTM 분석을 통해  
예측한 7일 검색량

VS

Elasticsearch에서 가져온  
2024년 검색량

```
def compare_with_2024_avg(df_yesterday, avg_2024):
    alerts = []

    for _, row in df_yesterday.iterrows():
        key = (row["brand"], row["gender"], row["age_group"])
        yesterday_ratio = row["ratio"]
        avg_ratio = avg_2024.get(key, None)

        if avg_ratio is not None:
            diff_percent = ((yesterday_ratio - avg_ratio) / avg_ratio) * 100
            if abs(diff_percent) >= 150: # 변화 감지 기준
                alerts.append(f"⚠️ {row['brand']} ({row['gender']}),
                            {row['age_group']}) 검색량이 {diff_percent:.2f}%
                            변화! ({avg_ratio:.2f}% → {yesterday_ratio:.2f}%)")

    if alerts:
        send_slack_message("\n".join(alerts))
    else:
        print("✅ 검색량 변화 없음!")
```

## 데이터 분석

# Slack 연동 검색 변화량 알림

The screenshot shows a Slack interface with a search bar at the top containing the query "sport\_drink\_search 검색". The search results are displayed in a channel named "# 스포츠-이온음료-점유율-분석". The results list various search terms along with their search volume changes and predicted future values. A specific message is highlighted with a blue background, showing a 13% increase in search volume for the term "포카리스웨트 (여성, 60대 이상)".

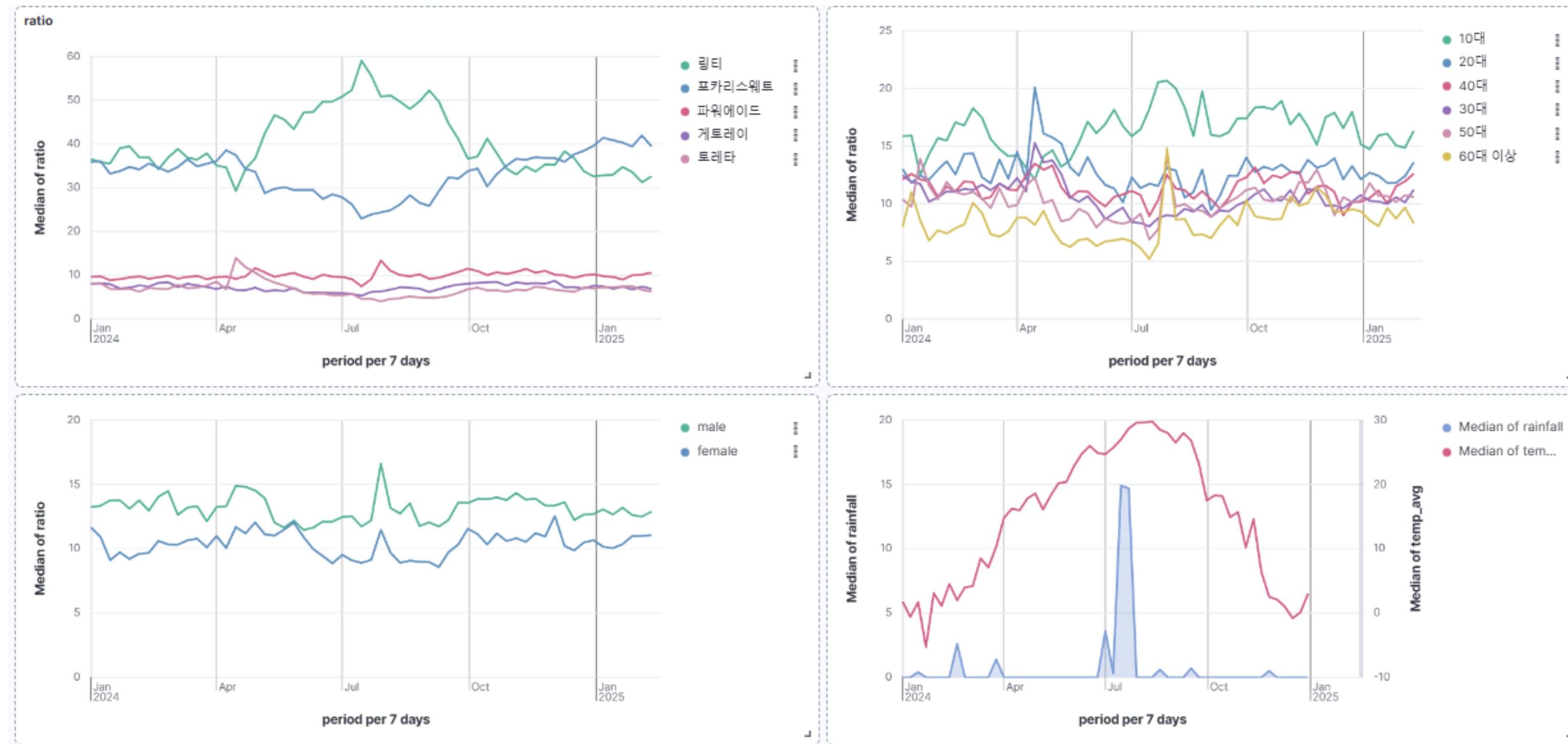
검색어	변화량 (%)	과거 (%)	예측 (%)
[2025-02-24] 포카리스웨트 (여성, 60대 이상)	↑ 13 새 메시지	48.81%	16.67%
[2025-02-25] 게토레이 (남성, 10대)	검색량이 28.21% 변화!	(과거: 42.50%)	예측: 14.29%
[2025-02-25] 토레타 (남성, 10대)	검색량이 23.57% 변화!	(과거: 5.00%)	예측: 28.57%
[2025-02-25] 파워에이드 (남성, 10대)	검색량이 21.07% 변화!	(과거: 7.50%)	예측: 28.57%
[2025-02-25] 링티 (여성, 20대)	검색량이 32.62% 변화!	(과거: 43.73%)	예측: 11.11%
[2025-02-25] 포카리스웨트 (남성, 30대)	검색량이 23.51% 변화!	(과거: 32.60%)	예측: 9.09%
[2025-02-25] 링티 (여성, 30대)	검색량이 49.02% 변화!	(과거: 49.02%)	예측: 0.00%
[2025-02-25] 토레타 (여성, 30대)	검색량이 24.14% 변화!	(과거: 9.19%)	예측: 33.33%
[2025-02-25] 파워에이드 (여성, 30대)	검색량이 23.92% 변화!	(과거: 9.41%)	예측: 33.33%
[2025-02-25] 링티 (남성, 50대)	검색량이 27.34% 변화!	(과거: 27.34%)	예측: 0.00%
[2025-02-25] 파워에이드 (여성, 50대)	검색량이 42.48% 변화!	(과거: 7.52%)	예측: 50.00%
[2025-02-25] 포카리스웨트 (여성, 50대)	검색량이 34.51% 변화!	(과거: 34.51%)	예측: 0.00%
[2025-02-25] 포카리스웨트 (남성, 60대 이상)	검색량이 46.78% 변화!	(과거: 57.89%)	예측: 11.11%
[2025-02-25] 토레타 (여성, 60대 이상)	검색량이 28.71% 변화!	(과거: 4.62%)	예측: 33.33%
[2025-02-25] 포카리스웨트 (여성, 60대 이상)	검색량이 29.48% 변화!	(과거: 46.15%)	예측: 16.67%
[2025-02-26] 게토레이 (남성, 10대)	검색량이 31.54% 변화!	(과거: 45.83%)	예측: 14.29%
[2025-02-26] 토레타 (남성, 10대)	검색량이 25.45% 변화!	(과거: 3.12%)	예측: 28.57%
[2025-02-26] 링티 (여성, 20대)	검색량이 34.39% 변화!	(과거: 45.50%)	예측: 11.11%
[2025-02-26] 포카리스웨트 (남성, 30대)	검색량이 28.07% 변화!	(과거: 37.16%)	예측: 9.09%
[2025-02-26] 링티 (여성, 30대)	검색량이 51.72% 변화!	(과거: 51.72%)	예측: 0.00%

04

# 데이터 시각화

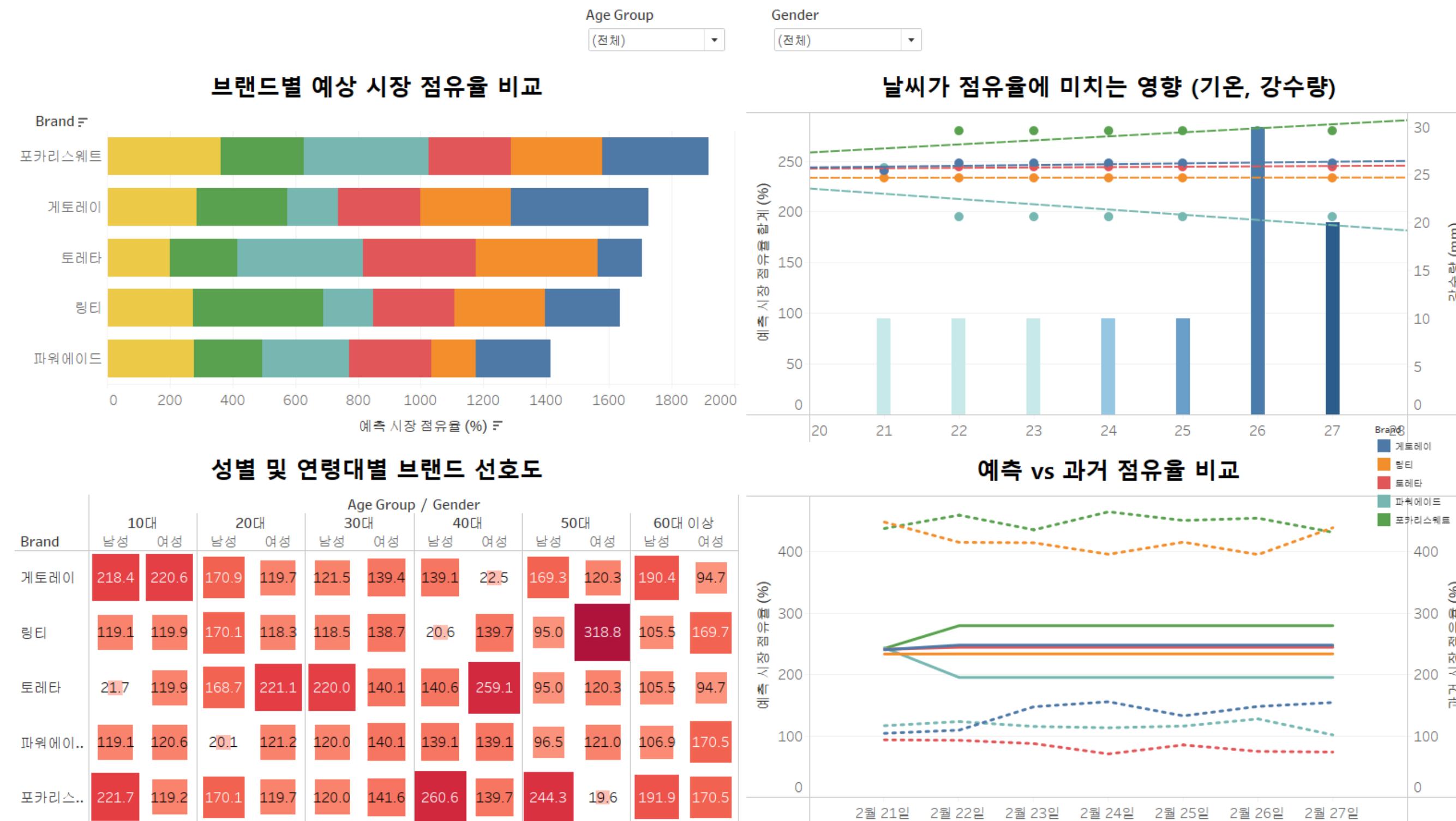
데이터 시각화

# Kibana 대시보드



## 데이터 시각화

## Tableau 대시보드



05

# 결론 및 한계점

## 결론 &amp; 한계

# 결론 및 한계점

## 결론

LSTM 모델을 활용한 브랜드 점유율 예측 시스템 구축

네이버 검색량 기반으로 브랜드별 시장 점유율 변화 분석 가능

예측 결과를 Slack 알림으로 실시간 제공 → 마케팅 및 의사결정 활용

## 한계점

데이터 한계 : 네이버 검색량만 반영하여 SNS, 판매 데이터 부족

예측 모델의 정밀도 : 단기 예측 성능은 우수하나 장기 예측 불확실성 높음

검색량과 실제 점유율 차이 : 검색량이 실제 매출과 일치하지 않을 가능성

06

# Q & A

1 조

감사합니다