

---

2025 Personal Project

# ‘카카오톡 대규모 업데이트’는 정말 실패했을까?

개인 프로젝트(25.11.19 – 25.12.13)

차윤미

---

# 목차

---

01.

## 결론

가설 검증 요약 및 최종 결론

02.

## 배경 및 문제 정의

분석 시작 : “카카오 업데이트는 정말 실패했을까?”

03.

## 분석 요약 및 핵심 발견

핵심 발견 : 데이터가 말해주는 것들

04.

## 상세 분석

심층 분석 결과

05.

## 제언

“그래서 앞으로 어떻게 하면 좋을 것인가?”

06.

## 부록(Appendix)

데이터 출처 및 분석 방법

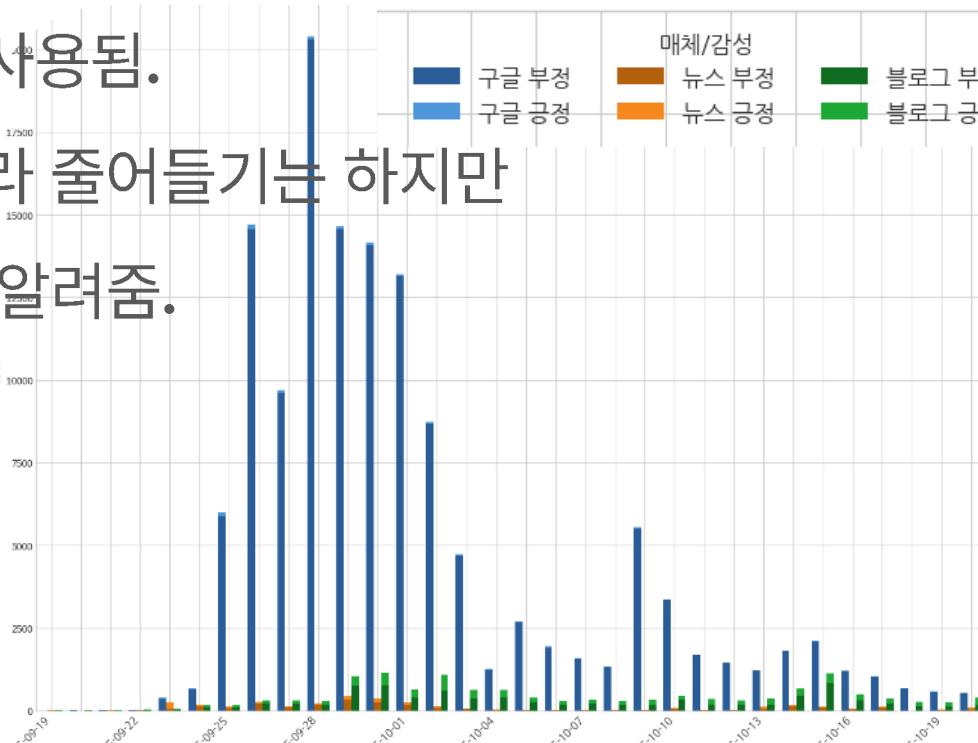
# 가설 검증 요약 및 최종 결론

## 가설 1

이번 카카오톡 대규모 업데이트는 부정적인 사용자 피드백이 강할 것이다.

## 결론 : 검증 성공

긍정 감성보다 부정 감성으로 정의된 구글스토어, 뉴스, 블로그에서 사용되는 단어들이 집중화되고 강력한 단어들을 많이 사용됨.  
이는 부정적인 사용자의 피드백이 시간에 따라 줄어들기는 하지만 부정적인 반응은 크게 변하지 않는다는 것을 알려줌.



## 가설 2

이번 카카오톡 대규모 업데이트에서 가장 부정적인 피드백이 강한 변경 사항은 '친구' 탭의 개선에 대한 항목일 것이다.

## 결론 : 가설 기각 및 새로운 인사이트 발견

불만에서 자주 나타나는 키워드 '최악', '진짜', '개편', '친구', '이용자', '방법', '기능' 등에서 '친구'와 같이 높은 빈도로 사용된 단어는 **'개편'(뉴스 제목 1위, 본문 3위, 블로그 제목 3위, 본문 10위)**으로 이는 '개편'이라는 형태 자체에 대한 불만이 가장 높다는 것을 보여줌.  
다만, 그에 대한 해결책으로 제시된 단어들 중에서 '친구', '메신저'와 같은 단어들이 사용된 것을 보았을 때 **'친구 탭에 대한 롤백'**과 함께 **'메신저 기능에 대한 충실함'**이 사용자가 가장 원하는 복구 방법인 것으로 의견 제시 가능.

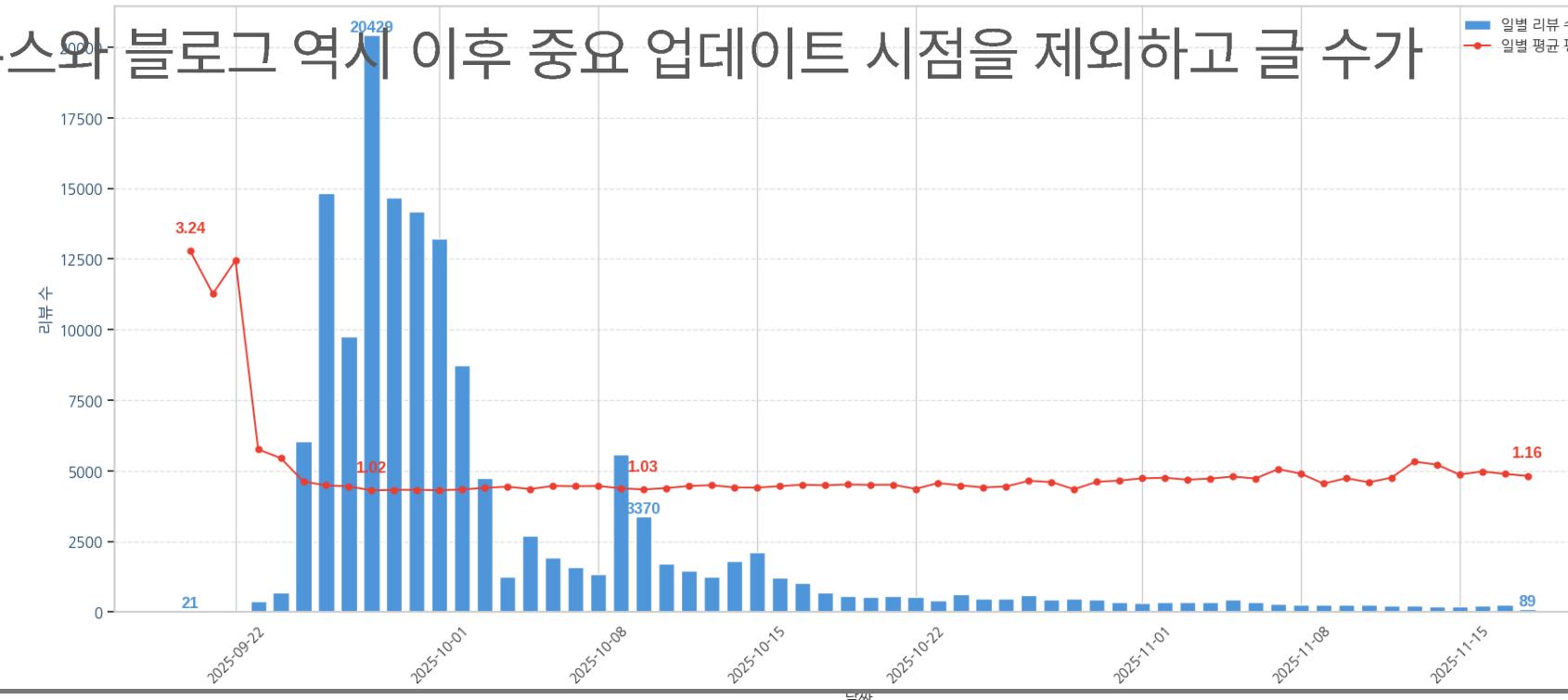
# 가설 검증 요약 및 최종 결론

## 가설 3

카카오톡 대규모 업데이트에 대한 반응을 추석 연휴를 지나면서 일부 소강 상태(리뷰 수가 점차 줄어들 것)로 변경될 것이다.

## 결론 : 가설 검증 성공

실제로 추석 주간을 기점으로 구글 스토어의 리뷰 수는 **83.5%** 가량 줄어든 상태이고, 뉴스와 블로그 역시 이후 중요 업데이트 시점을 제외하고 글 수가 줄어듦.



## 가설 4

카카오톡 대규모 업데이트 후 사용자의 부정적인 반응이 있더라도 단기간(한달) 내 친구 탭을 기존 형태로 되돌리지는 않을 것이다.  
(최소 25년 12월, 최대 26년 1분기에 진행될 것)

## 결론 : 가설 검증 성공

9월 23일 진행되었던 업데이트에 대한 롤백을 언급한 것은 9월 말. 이후 자잘한 버그 및 기능 업데이트는 진행되었으나 실질적인 '친구' 탭 롤백에 대해서는 현재(12월 10일)까지 진행되지 않았음. 다만, 9-10일 뉴스 기사를 보았을 때 '빠르면 15일 전후로 업데이트 될 것이다'라는 내용이 공지됨.

# 가설 검증 요약 및 최종 결론

## 가설 5

카카오톡 대규모 업데이트에 대한 롤백 작업은 완벽한 롤백이 아닌 일부 내용만 수정 혹은 반영한 상태로 반영될 것이다.

## 결론 : 미확인

실제 롤백 작업이 아직 진행되지 않았기 때문에 아직 알 수 없음.  
(친구 탭을 되돌린다는 기사가 나왔으나, 실제 어떤 형태가 될지 알 수 없으며 함께 문제가 되었던 솟恫은 계속 유지될 것으로 보임)

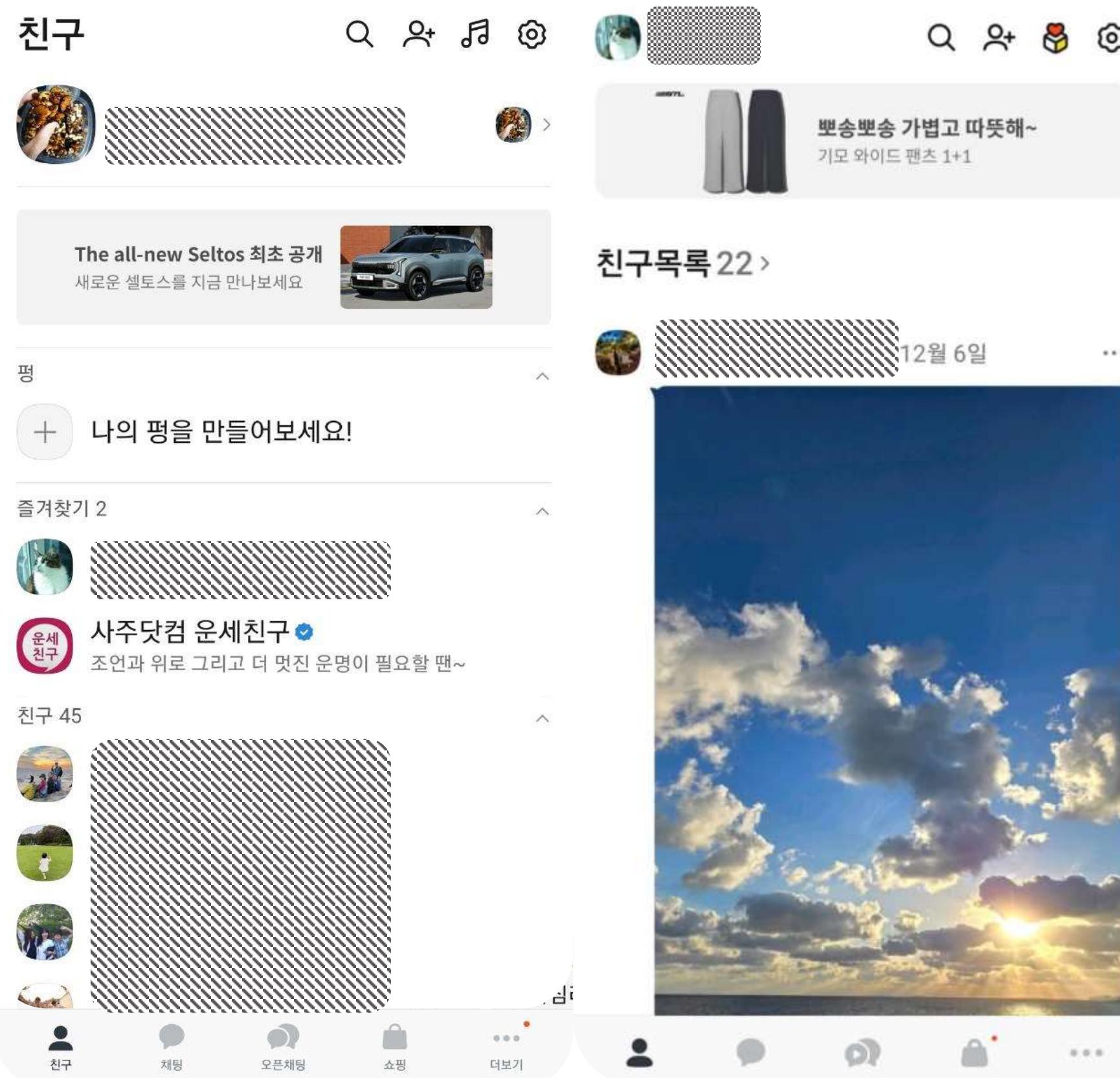
## 가설 6

카카오톡 대규모 업데이트 중 '친구' 탭에 대한 수정은 바로 가능하지만(플래그 형태로 처리되어 있을 것) 반영하지 않을 것이며 롤백이 불가능하다고 시간을 끌 것이다.

## 결론 : 가설 검증 성공

실제 카카오톡 업데이트에 대한 롤백이 불가능하다고 10월 중순 언급하였으나, 일부 사용자들이 버전 롤백과 config를 이용해 기존 형태의 친구 탭을 복구한 내용을 공유함.  
**(!) 실제, 해당 업데이트를 받지 않고 현재(12/14)까지 카카오톡 사용 가능.**  
진짜 롤백이 불가능한 경우 패킷이 모두 변경된 상태이기 때문에 업데이트를 받지 않으면 카카오톡 접속이 되지 않아야 함.

# “카카오 업데이트는 정말 실패했을까?”



## 배경 요인 1

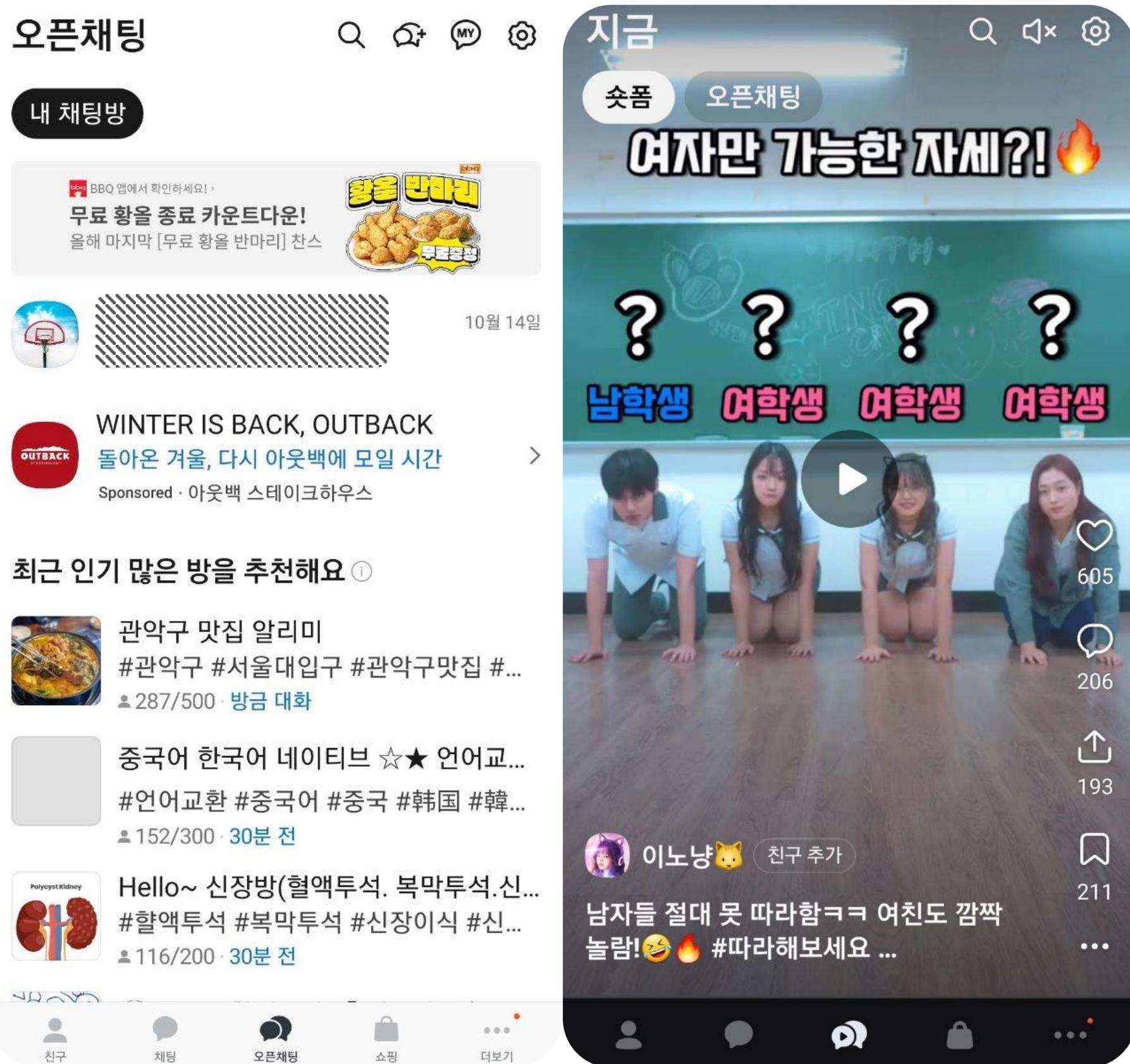
9월 23일 진행된 v25.8.0 카카오톡 업데이트를 진행한 후

친구 탭에서

- **스크롤을 이용해 친구를 확인**하던 방법이
- 친구목록을 눌러 새로운 창에서 친구를 확인하고,  
**스크롤을 이용해 친구리스트에 등록된 타인의 프로필 정보**를 확인  
하게 변경

> 내 개인 정보가 타인의 화면에 노출되고, 관심없는 타인의 정보를  
친구 탭을 통해 확인해야 하는 점에 대한 극심한 스트레스 호소

# “카카오 업데이트는 정말 실패했을까?”



## 배경 요인 2

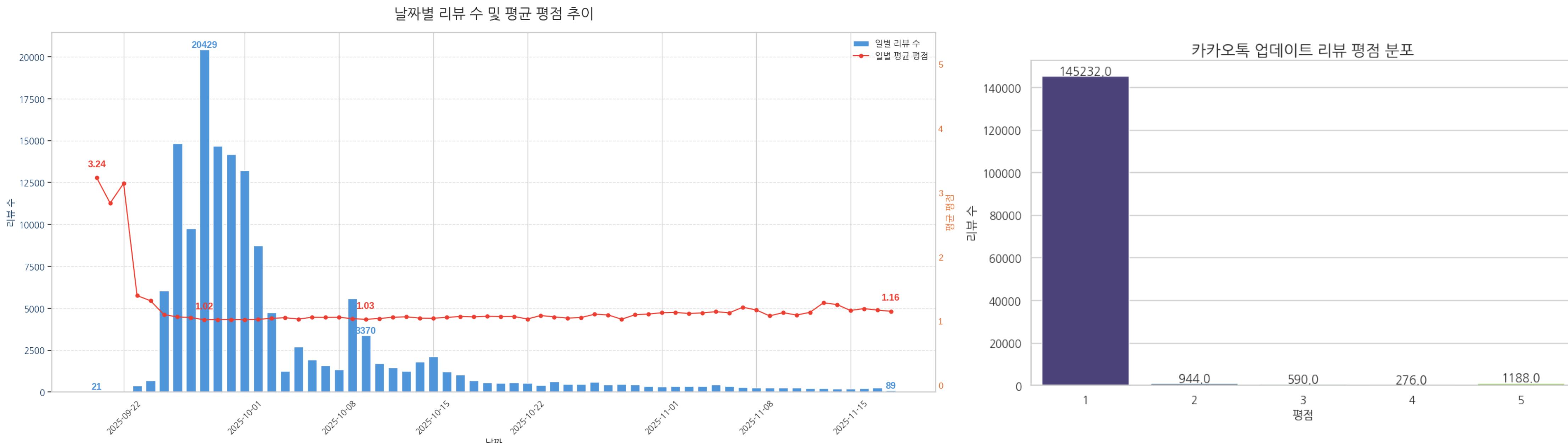
9월 23일 진행된 v25.8.0 카카오톡 업데이트를 진행한 후

오픈채팅 탭에서

- 오픈 채팅만 제공되다가
- 화면 입장시 솟폼이 추가되고 **숏폼이 제일 먼저 노출되도록 설정**,
- 연령대 상관없이 솟폼이 노출되고 **이를 제어할 기능 없음**

> 의사와 상관없는 솟폼에 강제로 노출되고, 사용 연령대를 고려하지 않은 기능 추가에 대한 급격한 불만 증가

# “카카오 업데이트는 정말 실패했을까?”



9월 19일부터 11월 20일까지 총 63일간 등록된 구글 스토어 리뷰를 확인한 결과 총 148230건의 리뷰가 등록되었고, 그 중 리뷰 1점이 97.97%에 달해 명백한 실패 신호로 해석됩니다.

그러나, 이 '실패'가 단순한 사용자들의 감정적 반발 때문인지, 아니면 더 깊은 구조적 원인이 있는지, 그리고 이러한 사용자 반응에 대해 언론과 시장은 어떻게 다르게 이야기했는지를 입체적으로 파악하기 위해, 뉴스 및 블로그 데이터를 포함한 심층 분석을 진행했습니다.

# “카카오 업데이트는 정말 실패했을까?”

---

'배경 1'과 '배경 2'와 같은 **급격한 사용자 부정 반응**을 보고 “카카오톡 업데이트는 정말 실패했을까?”라는 질문과 그에 대한 답을 하기 위해 다음과 같은 **6가지 가설**을 세우고 분석을 진행하기로 하였습니다.

**가설 1.** 이번 카카오톡 대규모 업데이트는 부정적인 사용자 피드백이 강할 것이다.

**가설 2.** 이번 카카오톡 대규모 업데이트에서 가장 부정적인 피드백이 강한 변경 사항은 '친구' 탭의 개선에 대한 항목일 것이다.

**가설 3.** 카카오톡 대규모 업데이트에 대한 반응을 추석 연휴를 지나면서 일부 소강 상태(리뷰 수가 점차 줄어들 것)로 변경될 것이다.

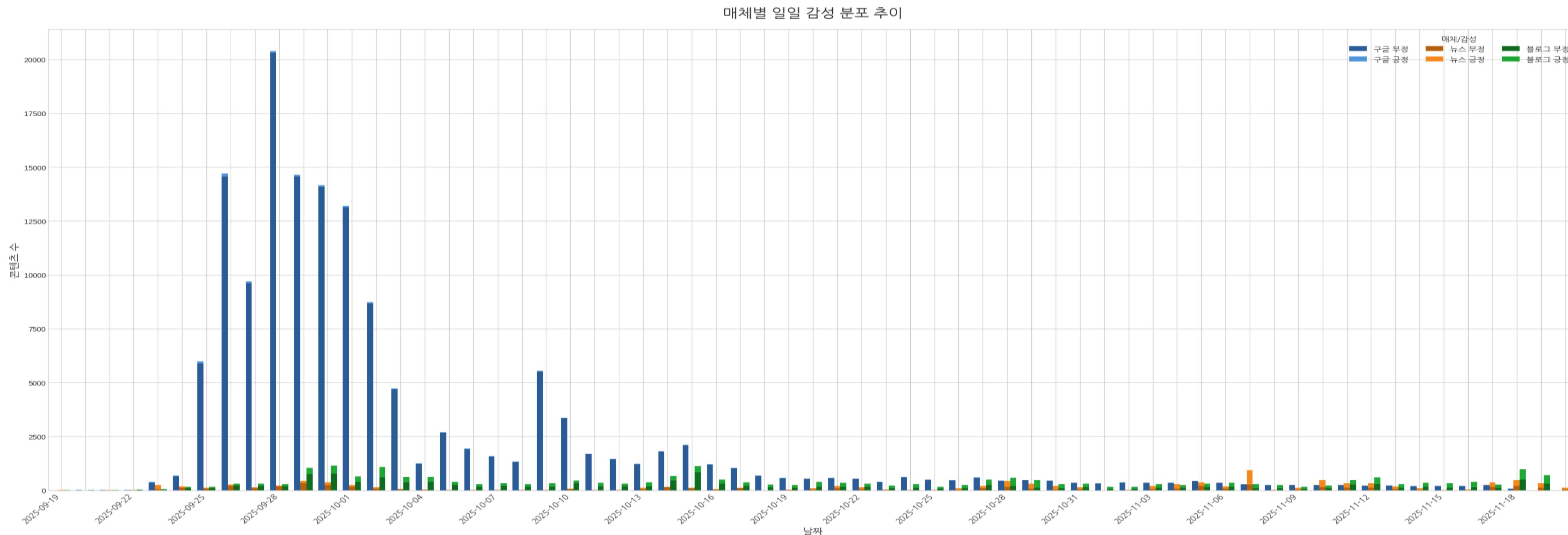
**가설 4.** 카카오톡 대규모 업데이트 후 사용자의 부정적인 반응이 있더라도 단기간(한달) 내 친구 탭을 기존 형태로 되돌리지는 않을 것이다. (최소 25년 12월, 최대 26년 1분기에 진행될 것)

**가설 5.** 카카오톡 대규모 업데이트에 대한 롤백 작업은 완벽한 롤백이 아닌 일부 내용만 수정 혹은 반영한 상태로 반영될 것이다.

**가설 6.** 카카오톡 대규모 업데이트 중 '친구' 탭에 대한 수정은 바로 가능하지만(플래그 형태로 처리되어 있을 것) 반영하지 않을 것이며 롤백이 불가능하다고 시간을 끌 것이다.

# 핵심 발견 : 데이터가 말하는 것!

**Finding 1** 시간이 지남에 따라 분노의 양은 줄었지만, **질은 변하지 않았다.**

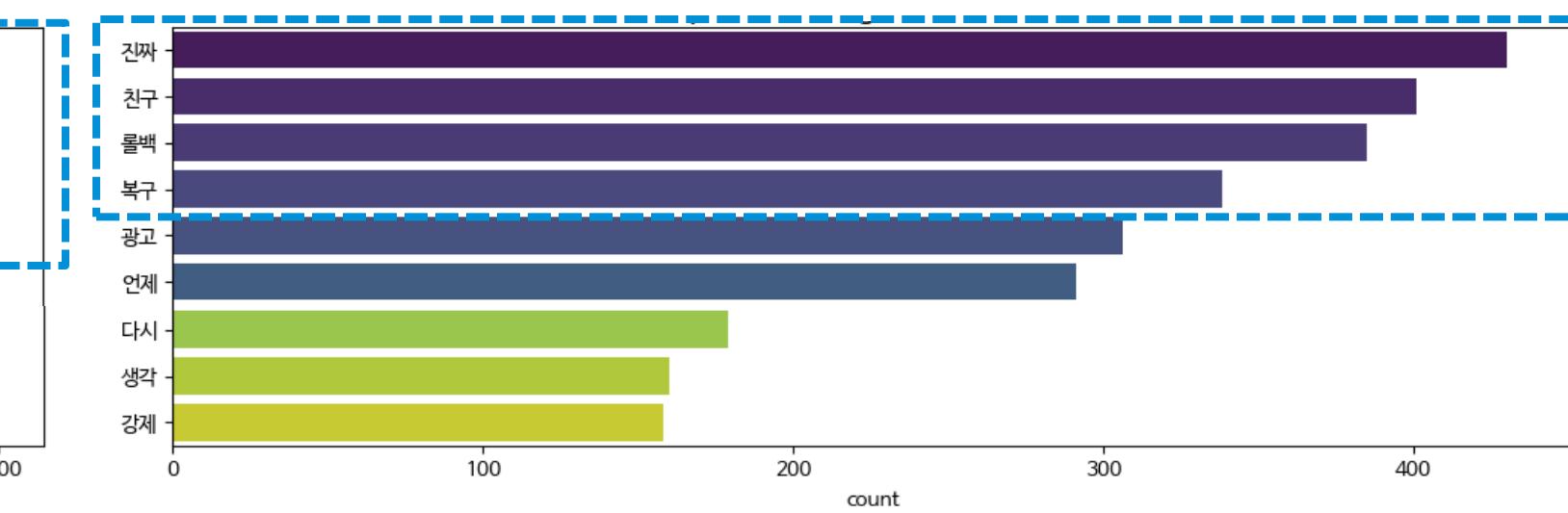
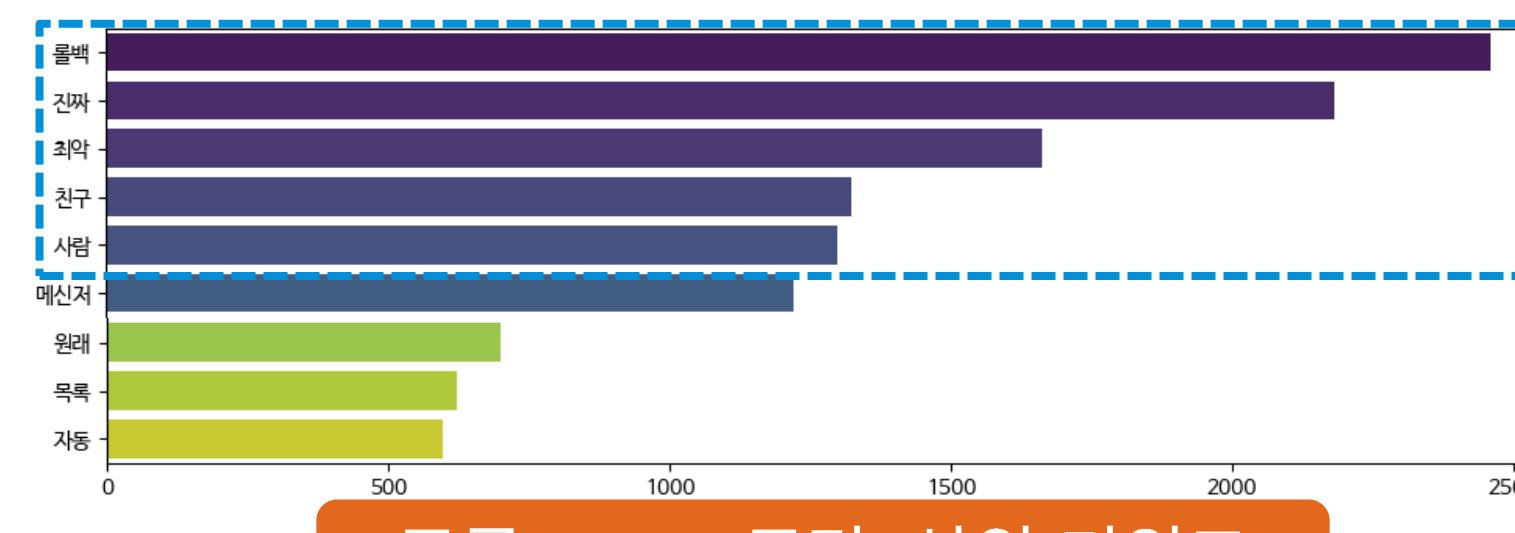
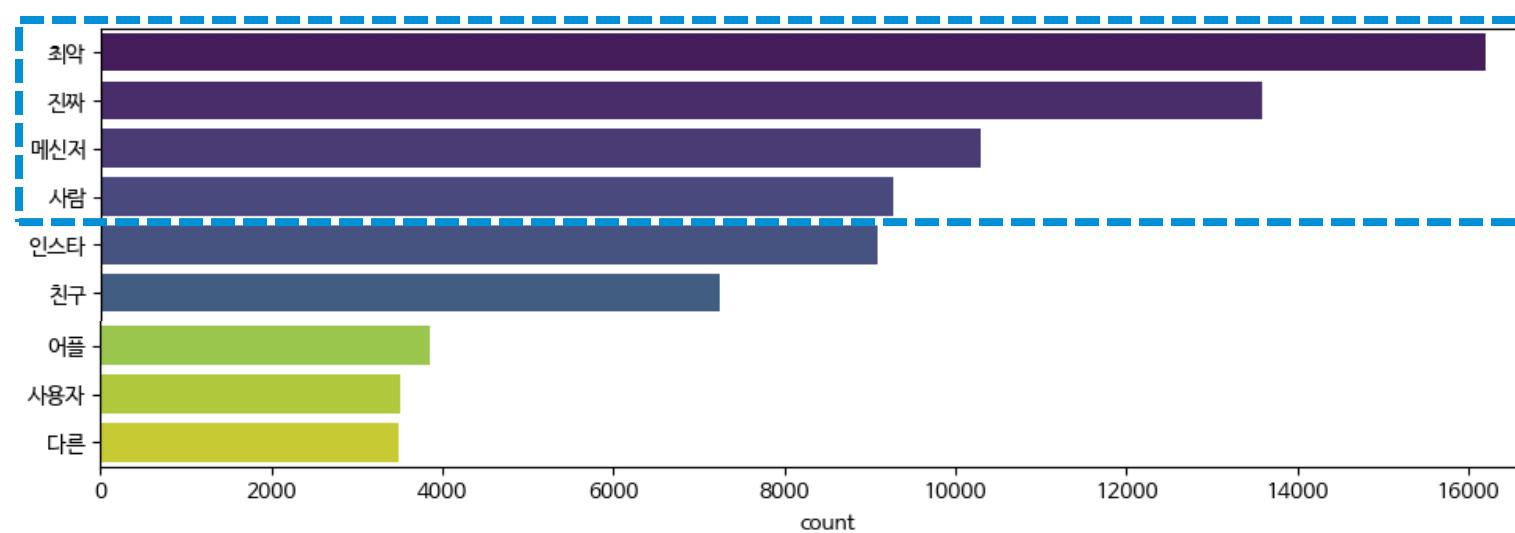


모든 매체(구글 스토어 리뷰, 뉴스, 블로그)에서 부정 감성의 리뷰가 시간에 따라 줄어드는 것을 확인할 수 있습니다.

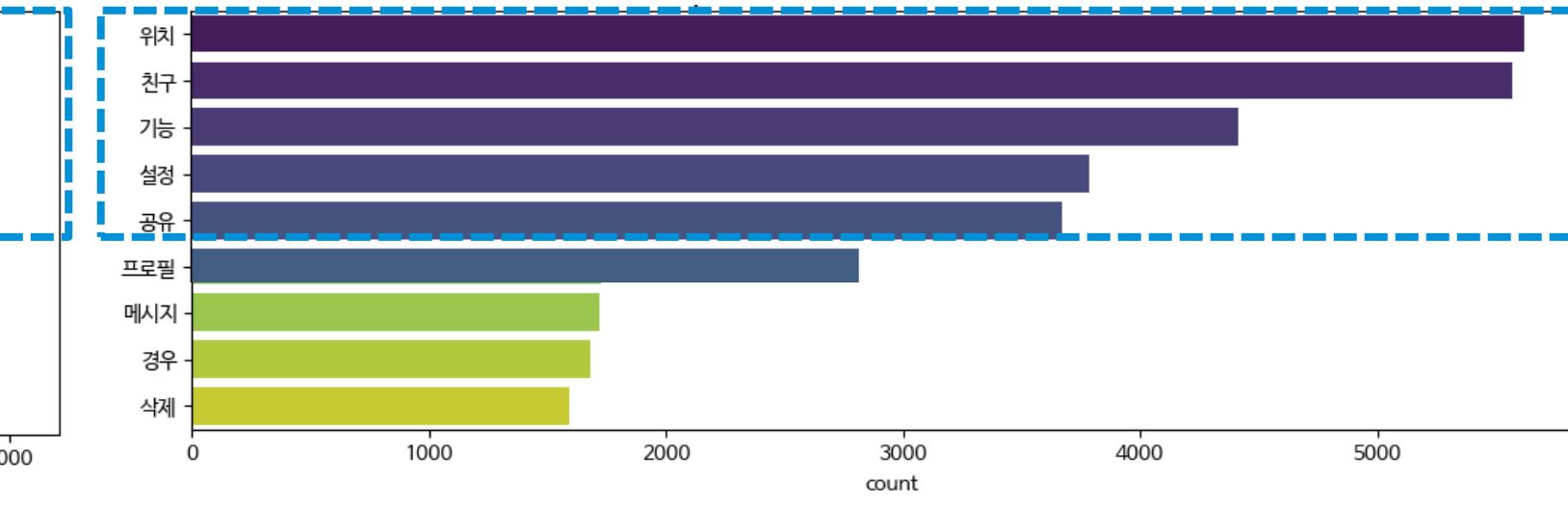
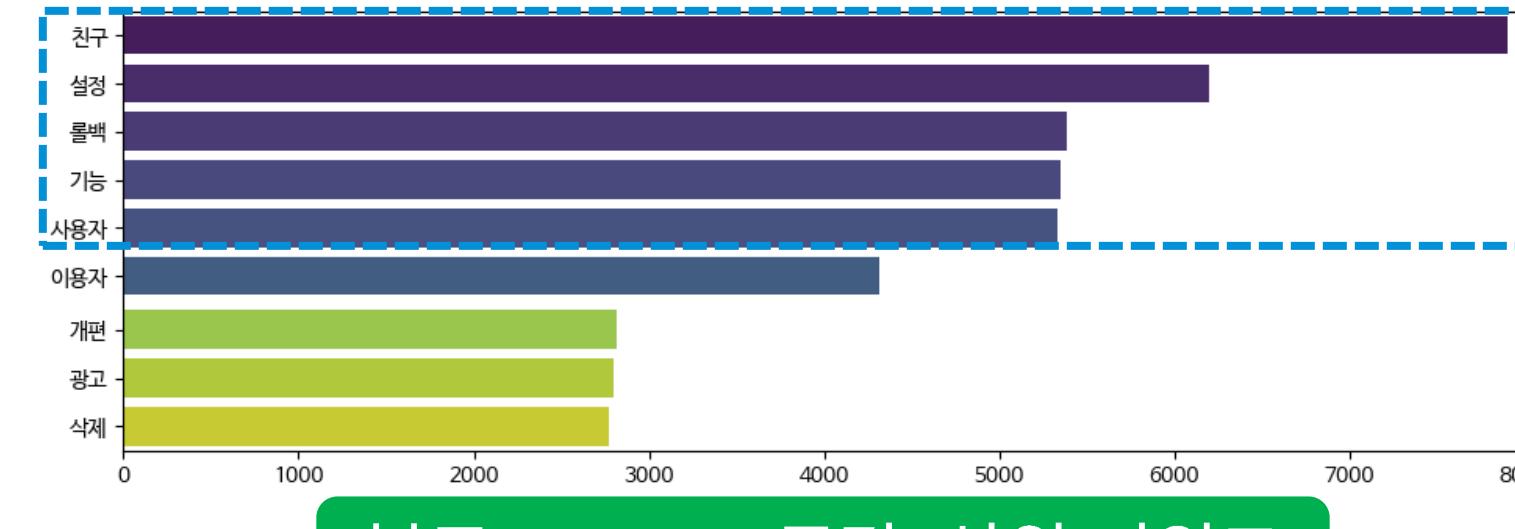
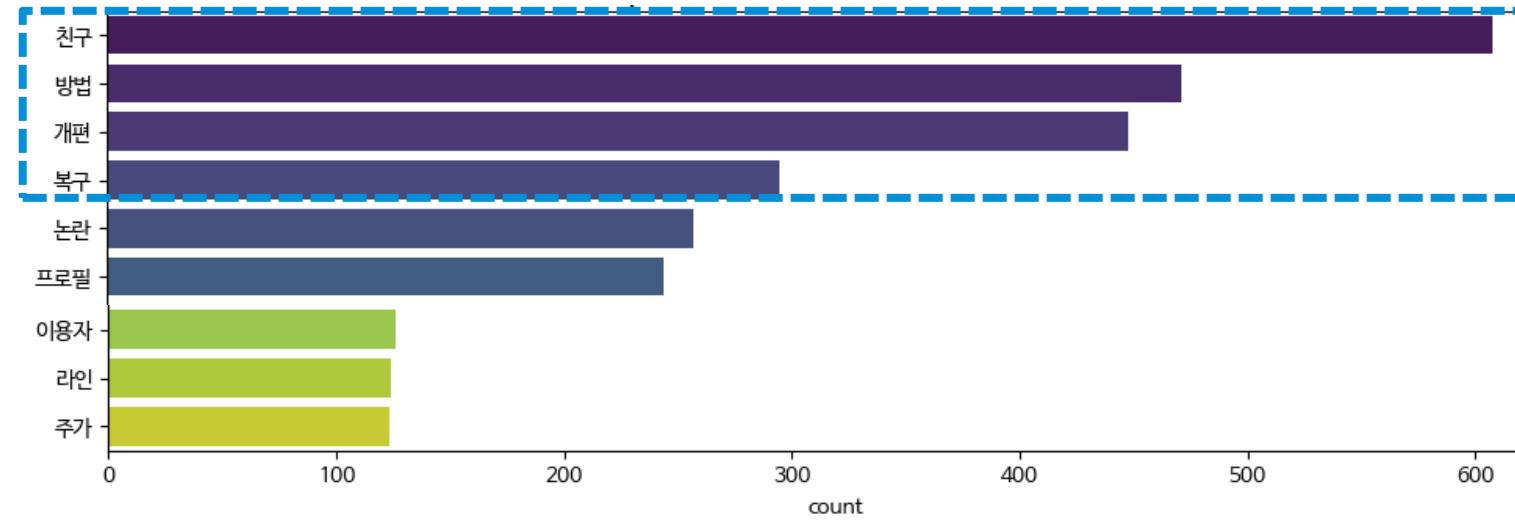
# 핵심 발견 : 데이터가 말하는 것!

## Finding 1

시간이 지남에 따라 분노의 양은 줄었지만, 질은 변하지 않았다.



구글 1, 2, 3구간 상위 키워드



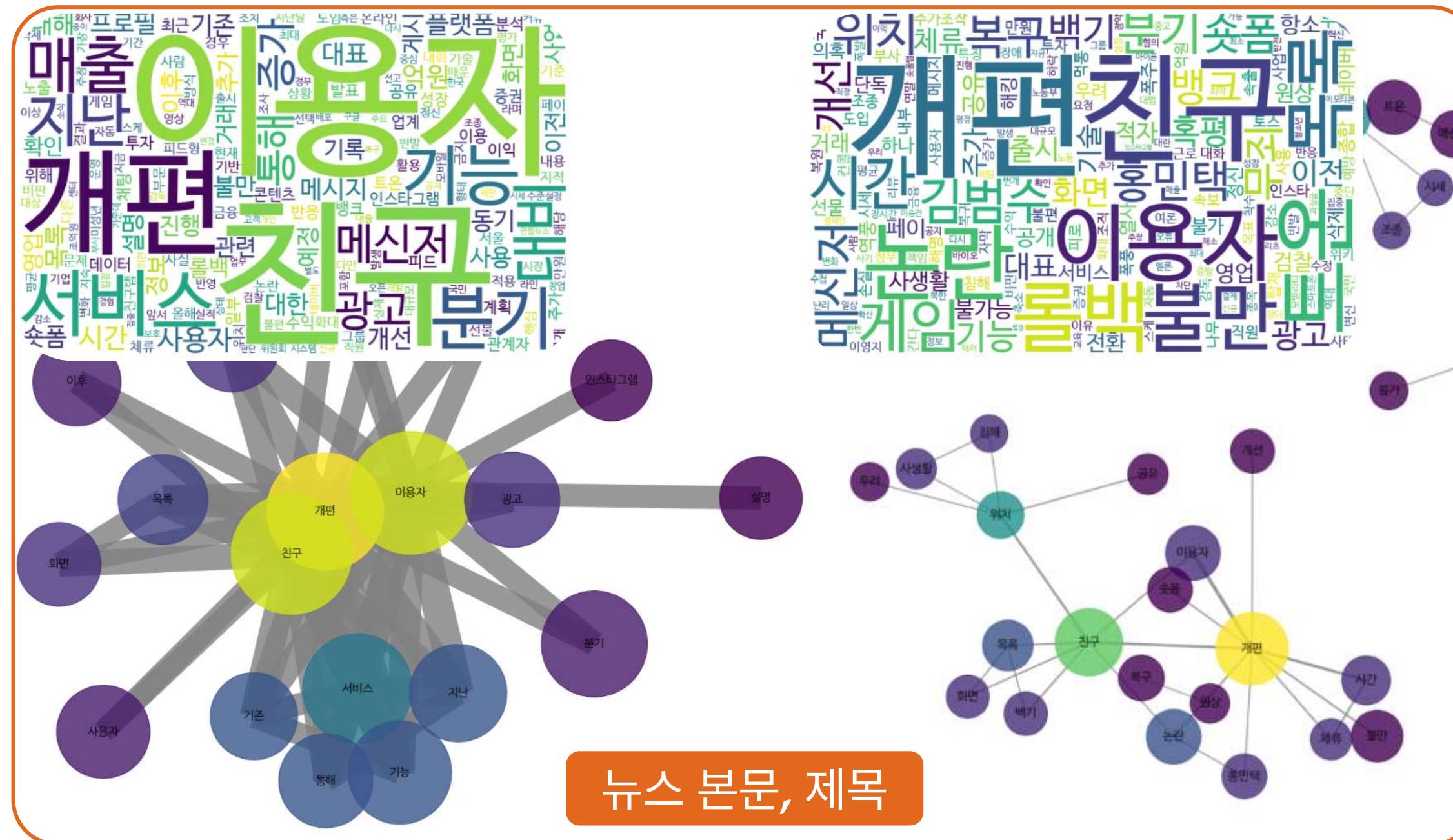
블로그 1, 2, 3구간 상위 키워드

또한 주요 이벤트와 데이터 변곡점을 기준으로 나눈 구간별 부정 감성 리뷰의 상위 키워드를 확인하였을 때 특정 단어들이 유지되는 것을 통해 문제가 해결되지 않고 계속 유지되고 있는 것을 확인할 수 있었습니다.

# 핵심 발견 : 데이터가 말하는 것!

## Finding 2

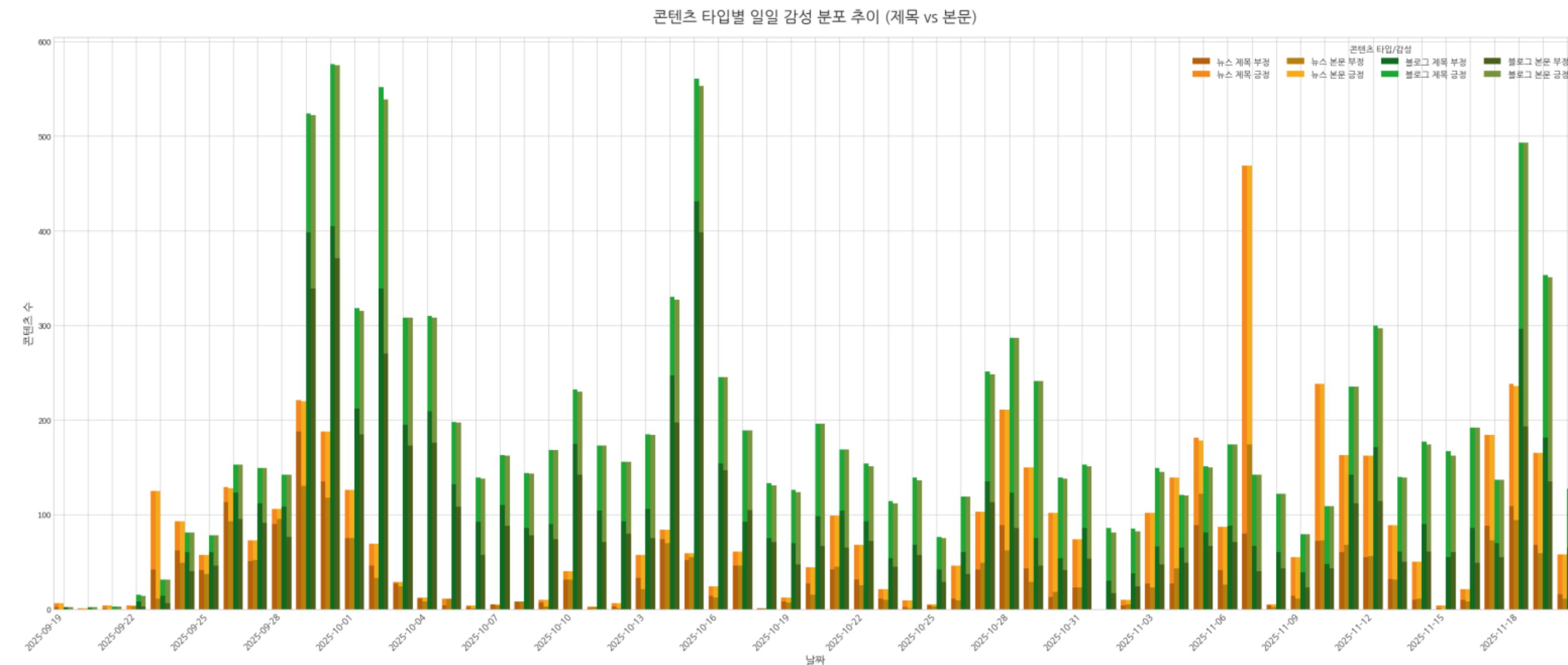
사용자 불만의 중심에는 ‘친구 탭’과 함께 ‘개편 행위’도 있었다.



‘친구’, ‘이용자’와 함께 ‘개편’이라는 키워드가 찾은 빈도로 사용되었습니다. 이는 기능적으로는 친구 탭이 불편하지만, 이번 개편 자체에 대한 반감도 불만의 중심에 함께 있다라는 것을 의미합니다.

# 핵심 발견 : 데이터가 말하는 것!

**Finding 3** 뉴스와 블로그는 제목과 본문의 감성이 불일치하는 경우가 있었다.



뉴스와 블로그에서는 제목과 내용의 감성과 주요 키워드가 일치하지 않는 경우가 확인되었고, 특히 블로그보다 뉴스에서 보다 많은 케이스가 발견되었습니다.

# 핵심 발견 : 데이터가 말하는 것!

**Finding 4** 각 매체(구글 스토어 리뷰, 네이버 기사, 네이버 블로그)는 각기 다른 시각으로 상황을 대표한다.



구글 스토어는 사용자의 직접적인 감정을, 네이버 기사는 카카오톡에 대한 피드백보다 카카오라는 기업에 대한 상황 분석 및 실적 확보, 네이버 블로그는 사용자의 직접적인 감정과 함께 해당 문제에 대한 해결&대안을 제시하는 단어들이 키워드의 상위권에 위치하였습니다.

# 심층 분석 결과 – 구간 설정

[!] **심층 분석은 시간에 따른 사용자 반응 변화를 추적하기 위해**, 전체 분석 기간을 **주요 이벤트와 데이터 변곡점을 기준**으로 다음과 같은 3개 구간으로 나누어 분석 결과를 비교하였습니다.

## 1구간: 초기 분노 폭발기 (9/22 ~ 10/09)

특징 : 업데이트 직후, 사용자들의 즉각적이고 폭발적인 부정 반응이 최고조에 달했던 시기.

주요 이벤트 : v25.8.0 업데이트, 추석 연휴.

분석 목표 : 사용자의 원초적인 분노의 핵심 원인 파악. 패턴 분석.

수집 데이터 : 구글스토어 리뷰(122,170건), 블로그(4,049건), 뉴스기사(1,270건)

## 2구간: 2차 파동과 조정기 (10/10 ~ 11/02)

특징 : 연휴 이후 2차 불만이 발생하고, '카나나' 등 신규 기능이 추가되며 여론이 복합적으로 변하는 시기.

주요 이벤트: 카나나(AI) 업데이트.

분석 목표 : 감정적 반응이 가라앉은 뒤, 구체적인 사용성 문제와 새로운 기능에 대한 평가 분석.

수집 데이터 : 구글스토어 리뷰(21,835건), 블로그(4,539건), 뉴스기사(1,289건)

## 3구간: 새로운 국면과 안정기 (11/03 ~ 11/20)

특징: '위치 공유' 등 새로운 논란이 발생하고, 실적 발표 등 기업 관점의 이슈가 부상하는 시기.

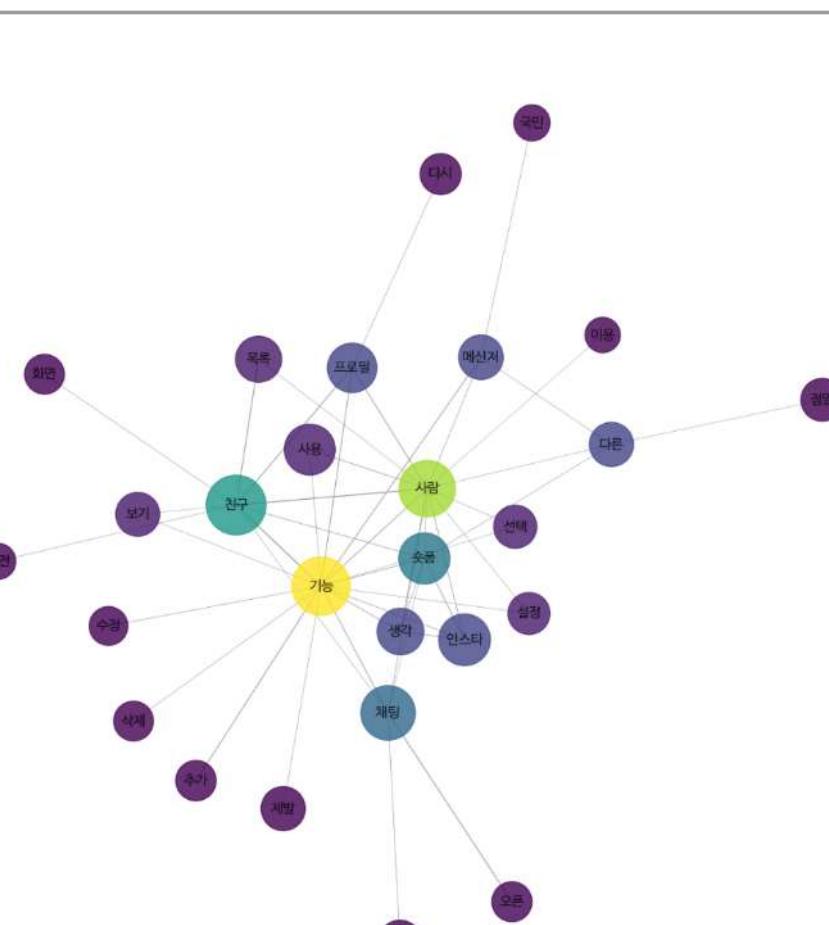
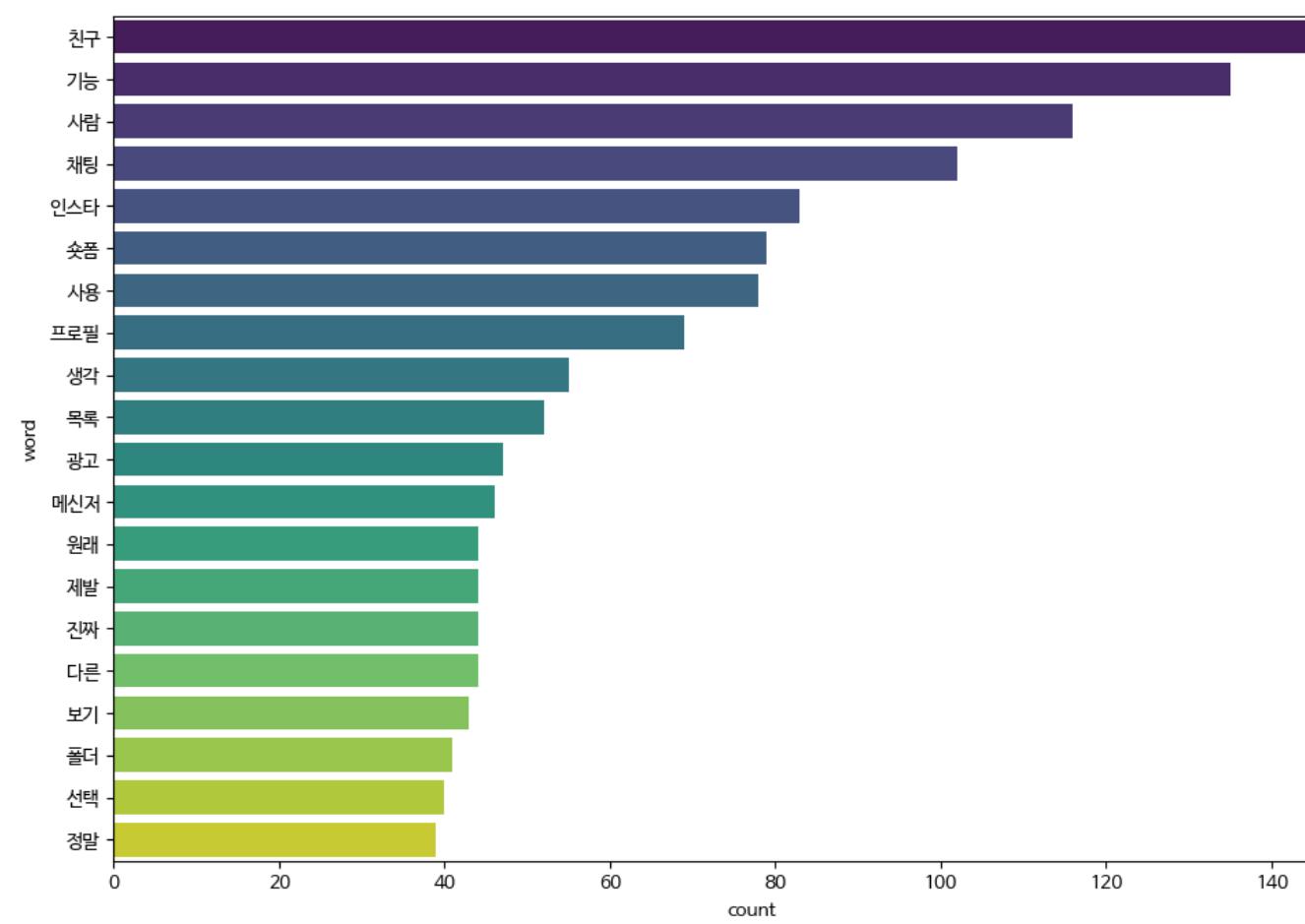
주요 이벤트: 실적 발표, 위치 공유 업데이트.

분석 목표: 기존 불만의 '일상화'와 '신규 이슈'에 대한 반응 패턴 분석.

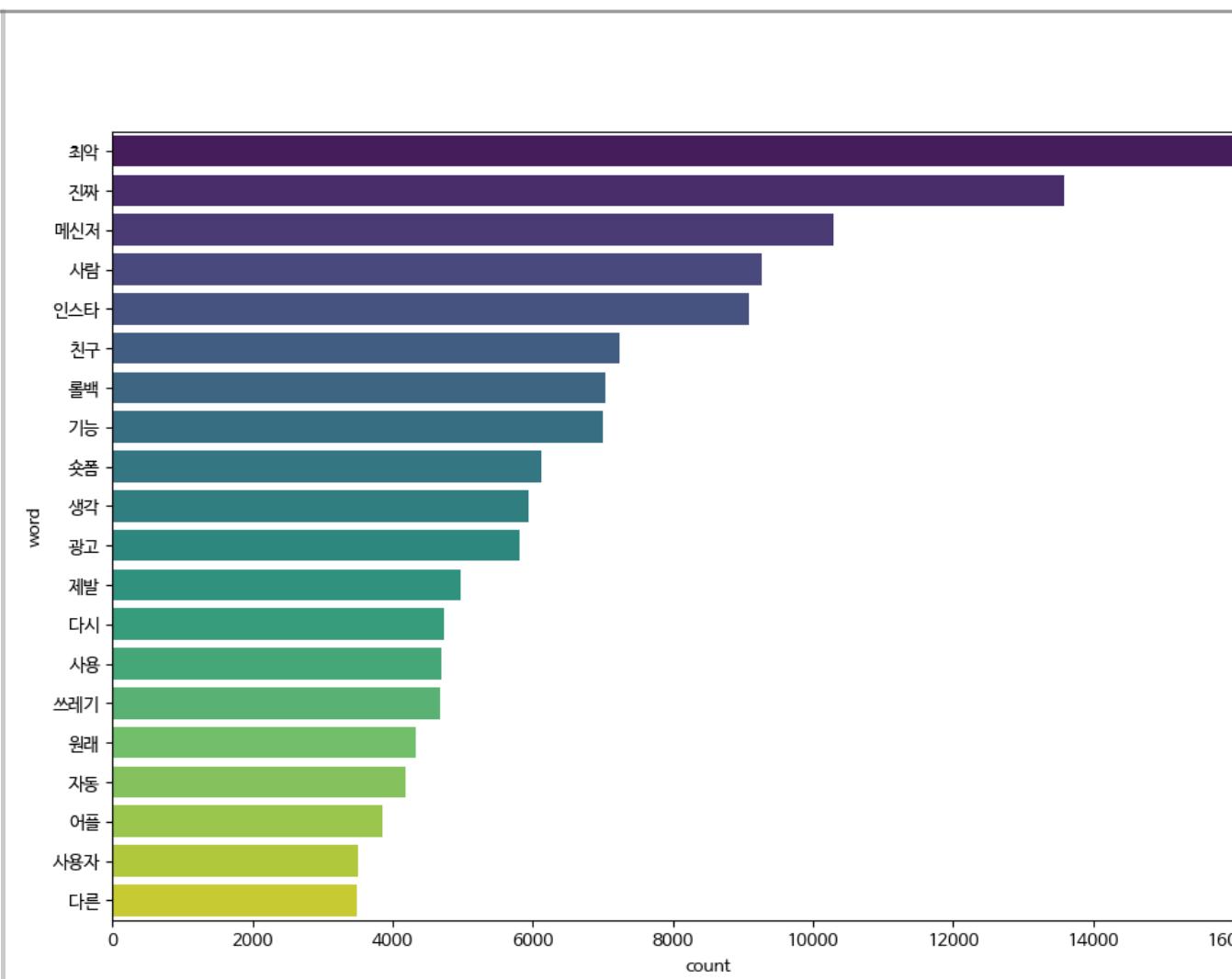
수집 데이터 : 구글스토어 리뷰(4,185건), 블로그(3,368건), 뉴스기사(2,410건)

# 심층 분석 결과 1. 사용자의 목소리는 어떻게 진화하는가?

## 01 1구간: 초기 분노 폭발기 (9/22 ~ 10/09)



구글 리뷰 긍정 - 1구간



구글 리뷰 부정 - 1구간

# 심층 분석 결과 1. 사용자의 목소리는 어떻게 진화하는가?

# 01 1구간: 초기 분노 폭발기 (9/22 ~ 10/09)



# 구글 리뷰 긍정 - 1구간



구글 리뷰 부정 - 1구간

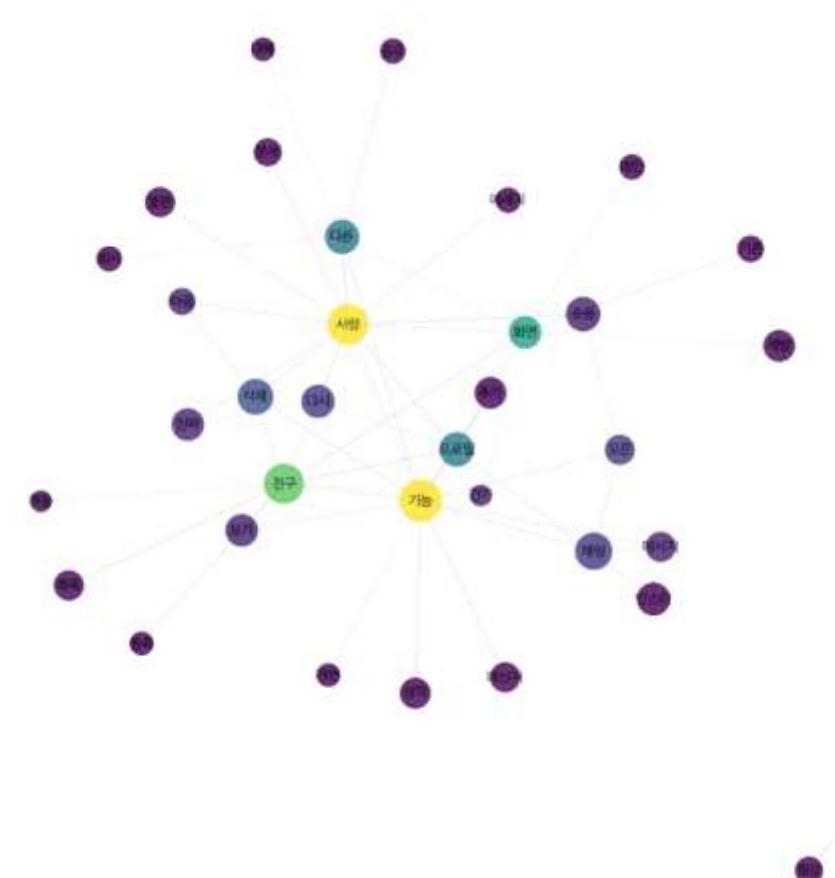
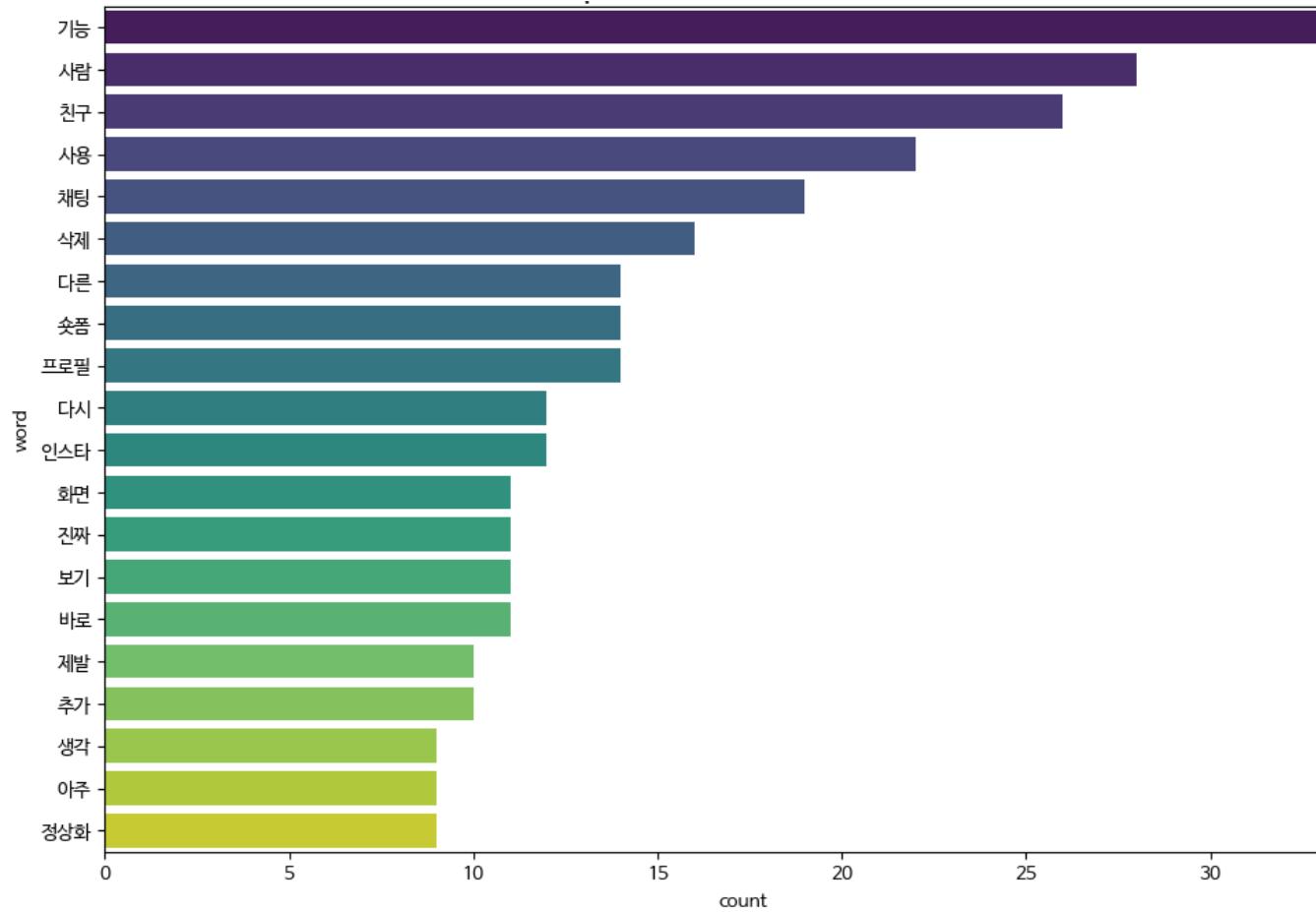
긍정 1구간의 상위 키워드로는 '친구, 기능, 사람'이 각 110~140건 사이 사용된 것에 반해,

부정 1구간의 상위 키워드인 '진짜, 최악'은  
140000~16000건 사이로 사용되었습니다.

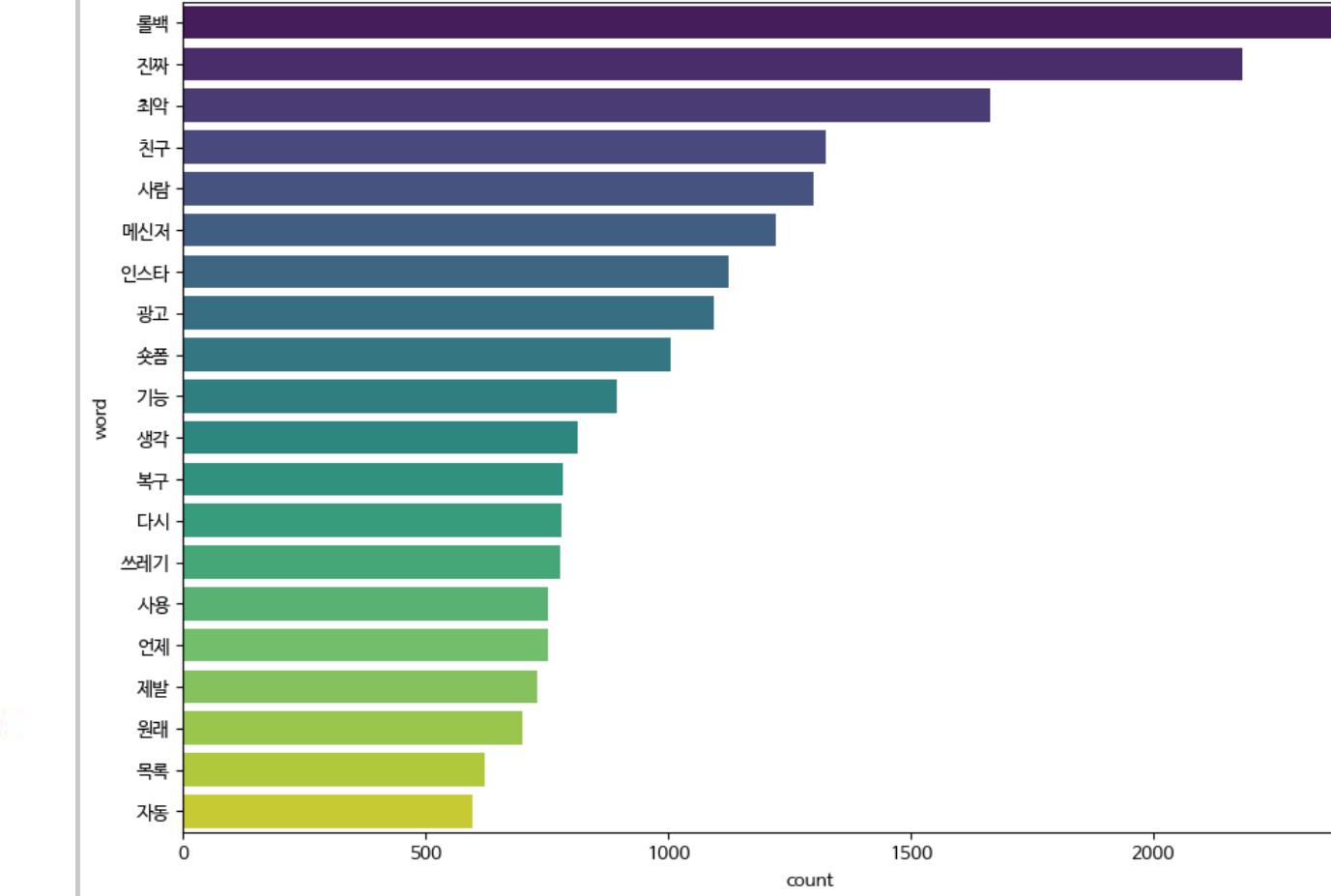
긍정적인 키워드에 비해 부정적인 키워드의 사용  
빈도가 1000배 이상 많은 것을 통해  
1구간은 분노 폭발기인 것을 확인할 수 있었습니다.

# 심층 분석 결과 1. 사용자의 목소리는 어떻게 진화하는가?

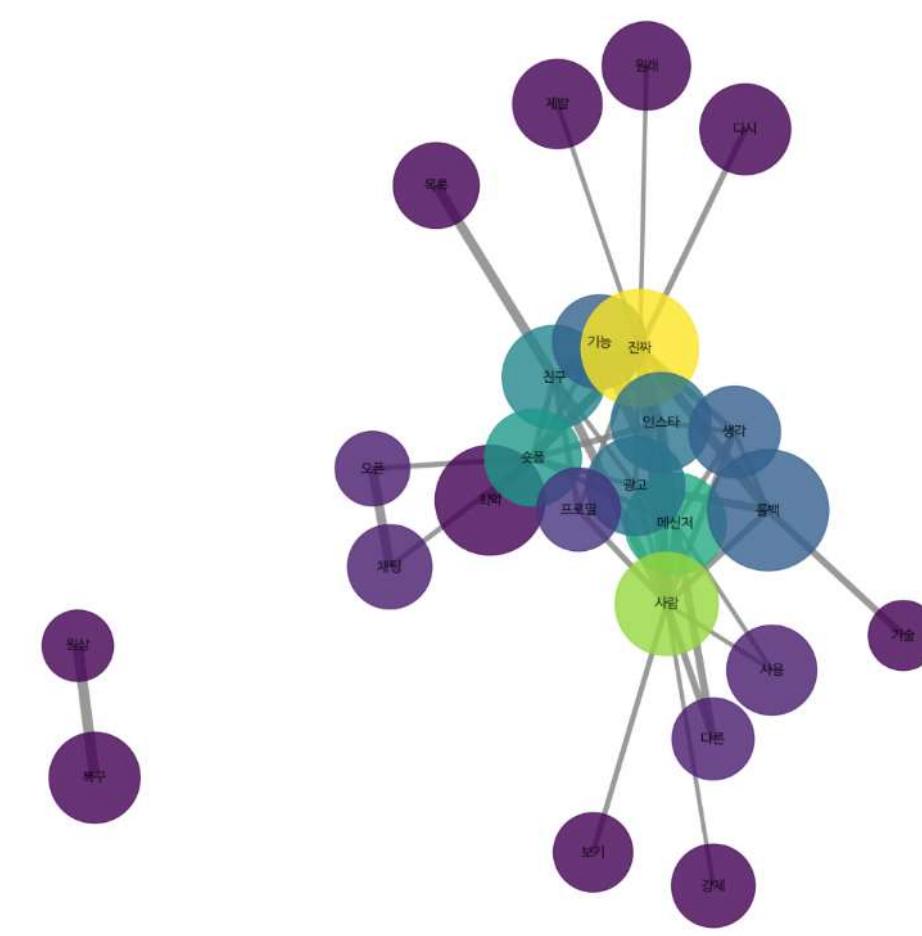
02 2구간: 2차 파동과 조정기 (10/10 ~ 11/02)



구글 리뷰 긍정 - 2구간



구글 리뷰 부정 - 2구간



# 심층 분석 결과 1. 사용자의 목소리는 어떻게 진화하는가?

## 02 2구간: 2차 파동과 조정기 (10/10 ~ 11/02)



## 구글 리뷰 긍정 - 2구간



## 구글 리뷰 부정 - 2구간

2구간에 들어서며 리뷰의 '양'은 1구간 대비 약 80%  
급감하였습니다.

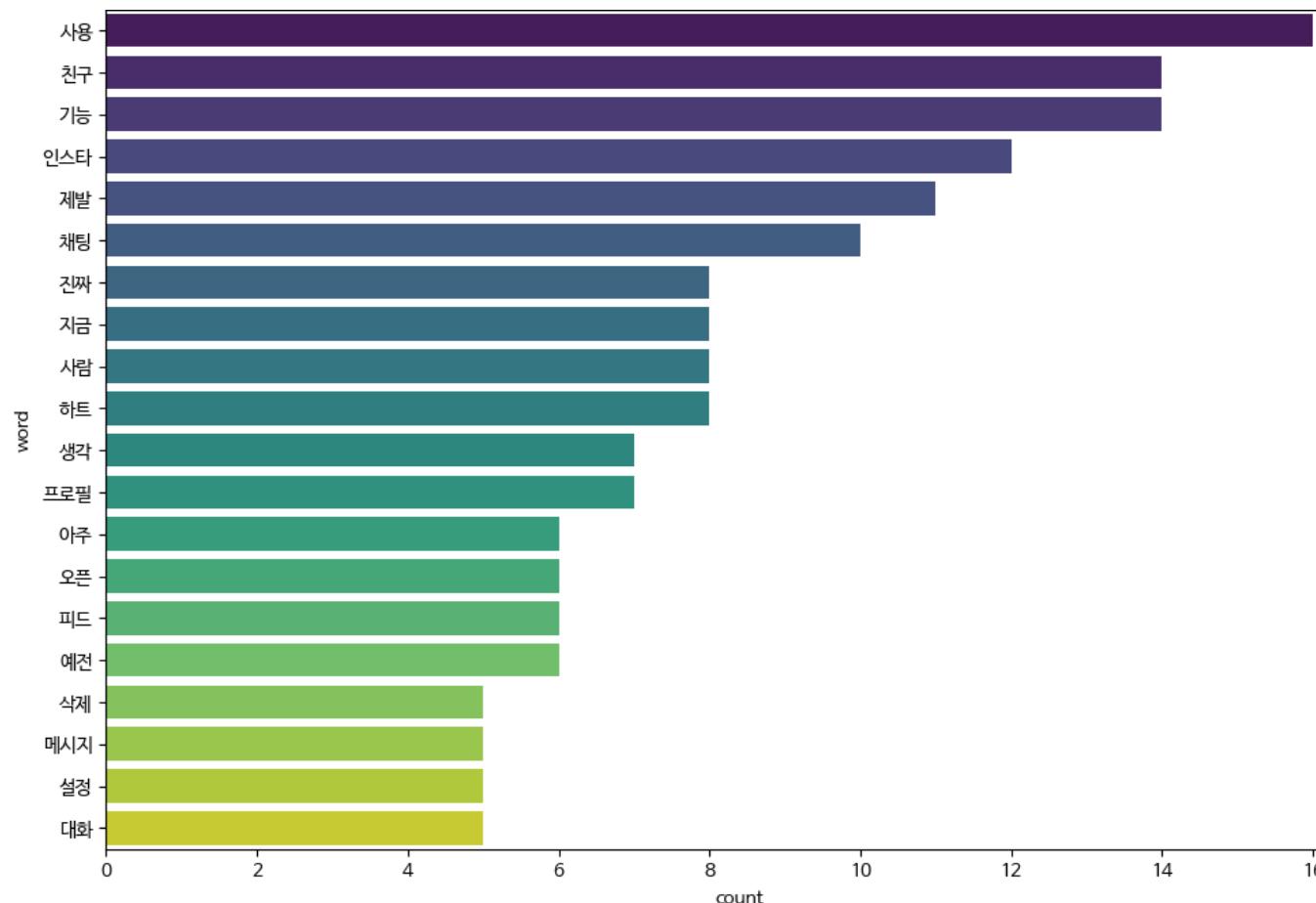
하지만 여전히 부정 리뷰가 긍정 리뷰보다 많이 등록되고 있으며,

여전히 '롤백, 진짜, 최악, 친구, 사람'과 같은  
1구간과 동일한 키워드가 상위 키워드로 등록된  
것을 보았을 때 **여전히 문제가 해결되지 않았음**을  
이야기하고 있습니다.

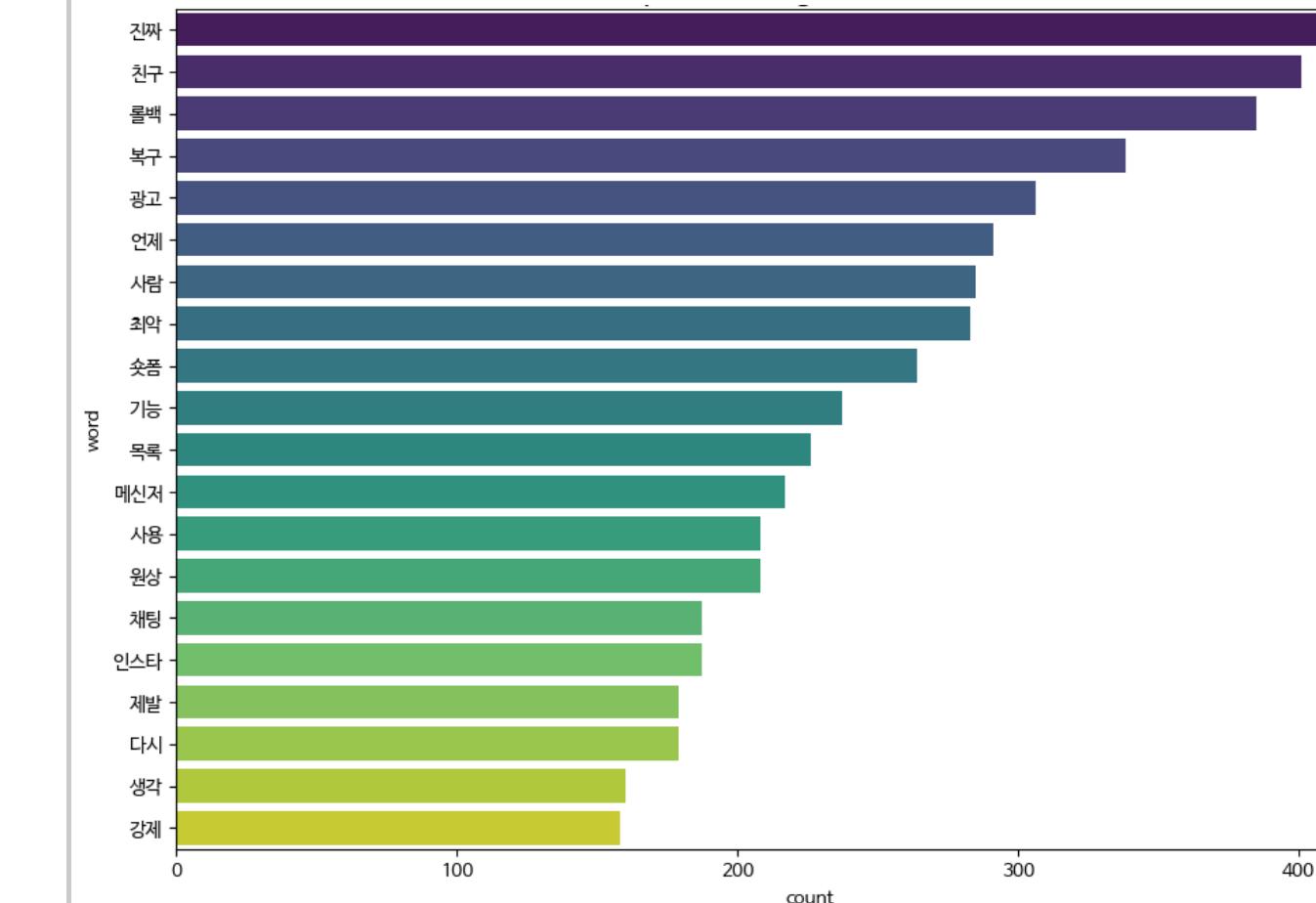
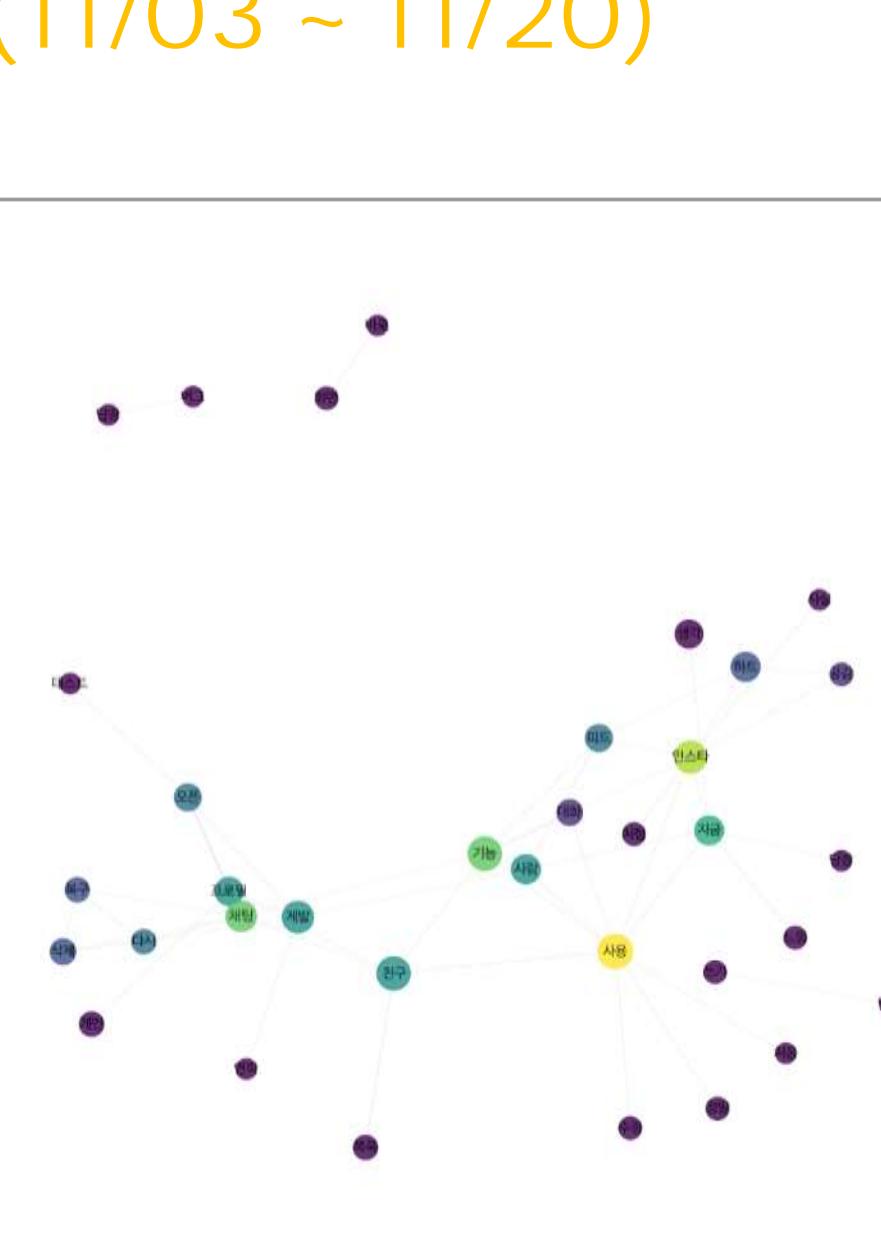
# 심층 분석 결과 1. 사용자의 목소리는 어떻게 진화하는가?

03

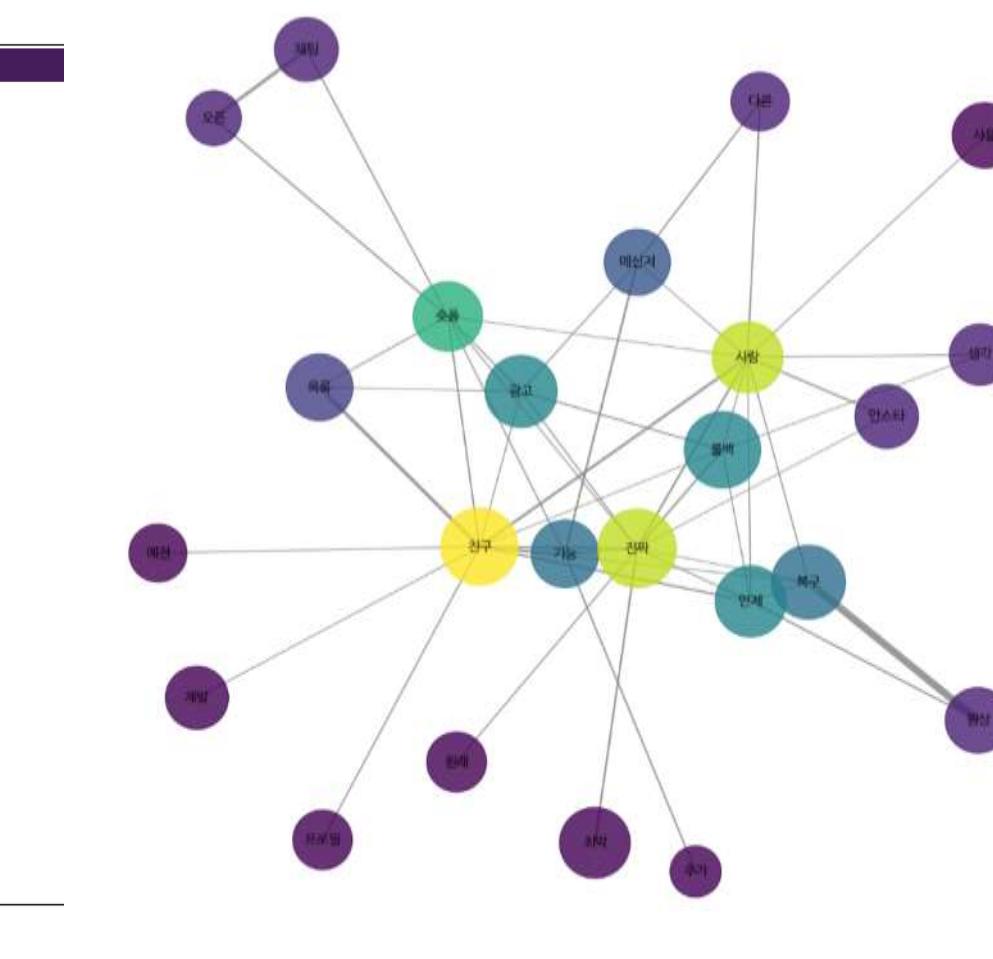
3구간: 새로운 국면과 안정기 (11/03 ~ 11/20)



구글 리뷰 긍정 - 3구간



구글 리뷰 부정 - 3구간



# 심층 분석 결과 1. 사용자의 목소리는 어떻게 진화하는가?

## 03 3구간: 새로운 국면과 안정기 (11/03 ~ 11/20)



## 구글 리뷰 긍정 - 3구간



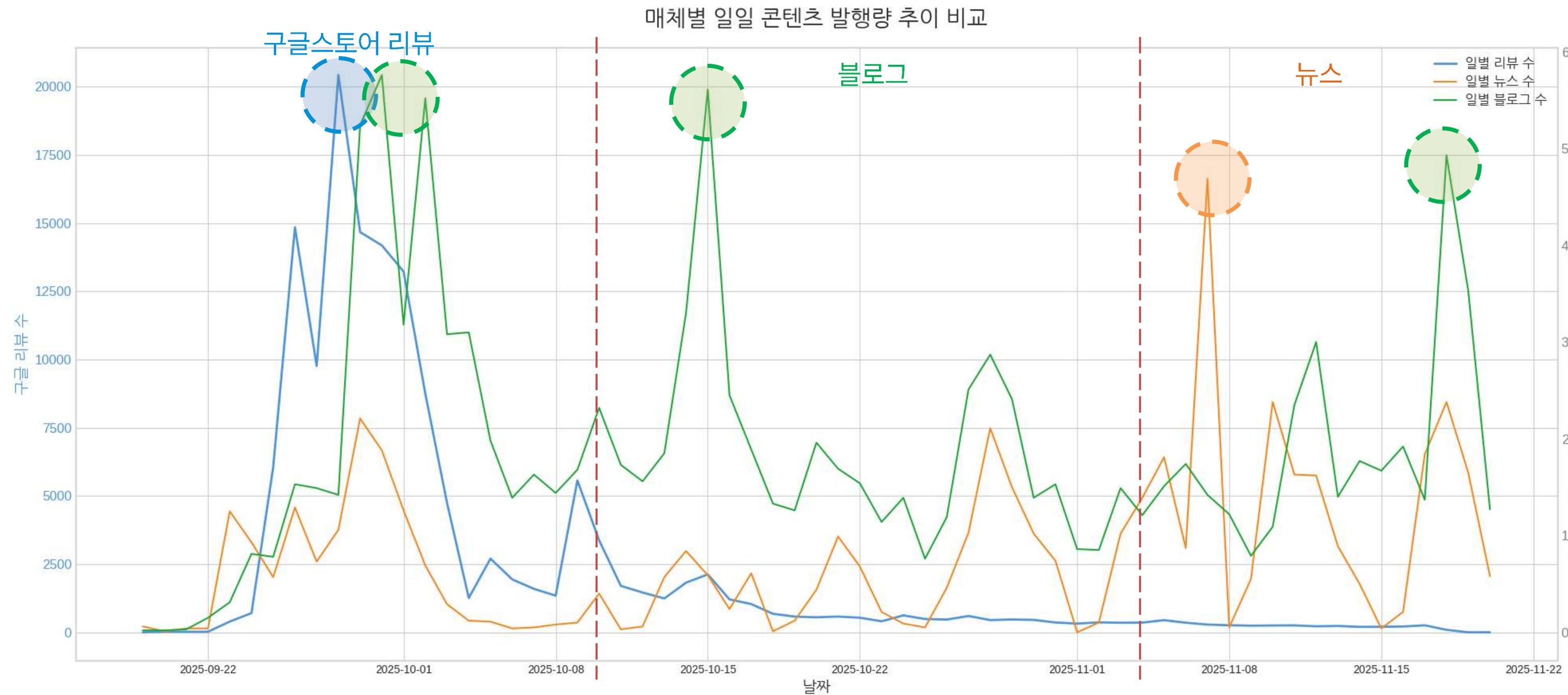
## 구글 리뷰 부정 - 3구간

3구간에서 리뷰는 4천여 건으로 초기 폭발량에  
비해 96% 이상 감소하였고,  
**'위치 공유'라는 사용자의 반응이 있을 법한  
업데이트가 진행되었음에도 유의미한 패턴을 찾기  
어려울 정도의 '침묵' 상태에 접어들었습니다.**

이는 가장 강한 불만을 가졌던 사용자들은 이미 의견 표출을 마쳤거나, 변화에 대한 기대를 접고 소극적인 태도로 돌아섰을 가능성,  
그리고 피드백을 남겨도 바뀌지 않는다는 부정적인 경험이 학습되어, 어차피 말해도 소용없다며 적극적인 개선 요구를 포기했을 가능성을 보여주는 것으로 보여집니다.

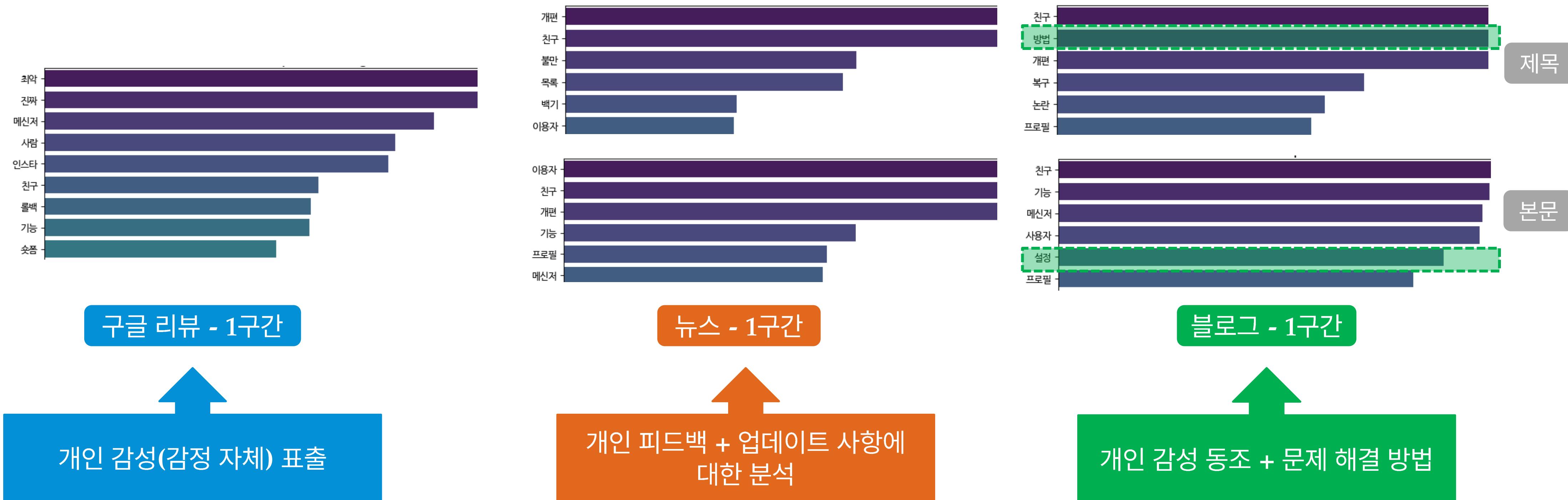
# 심층 분석 결과 2. 각 매체는 무엇을 대변하는가?

## 01 반응 속도와 양상의 차이



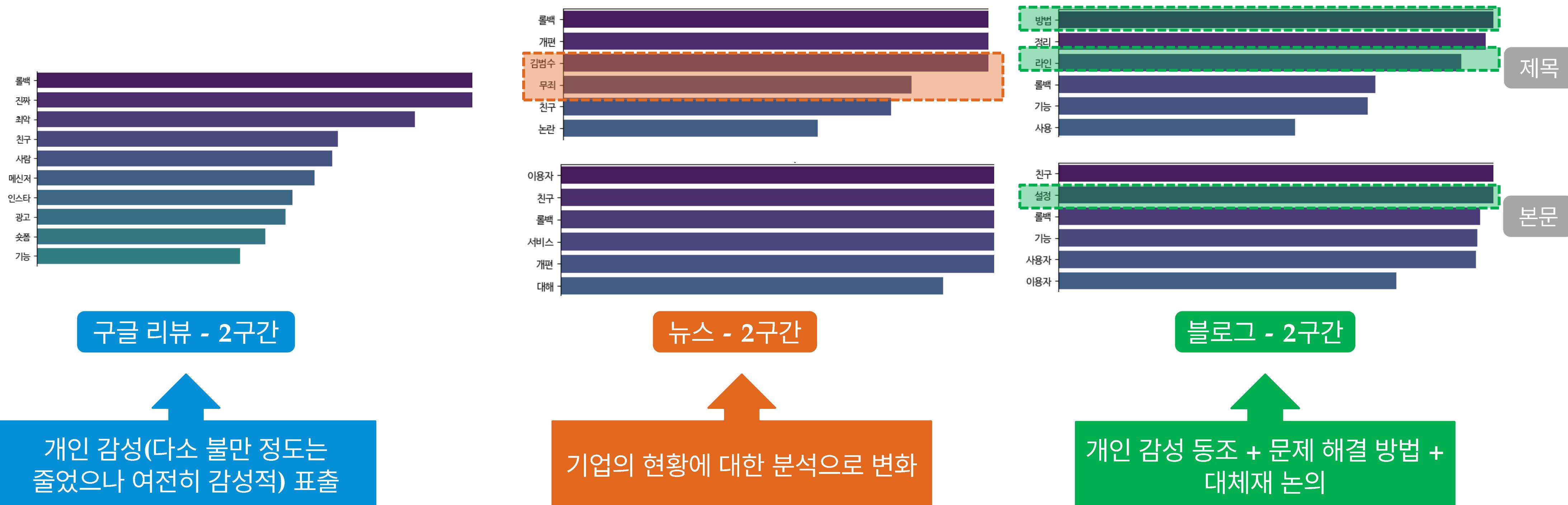
# 심층 분석 결과 2. 각 매체는 무엇을 대변하는가?

## 02 구간별 '부정' 여론의 관점 차이



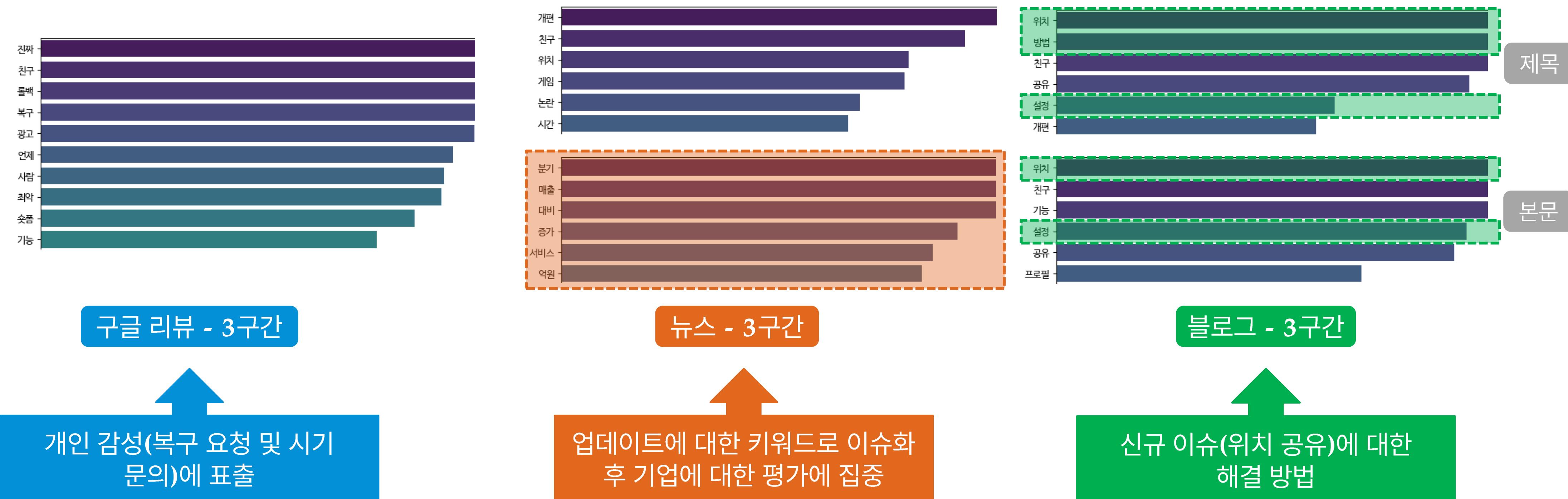
# 심층 분석 결과 2. 각 매체는 무엇을 대변하는가?

## 02 구간별 '부정' 여론의 관점 차이



# 심층 분석 결과 2. 각 매체는 무엇을 대변하는가?

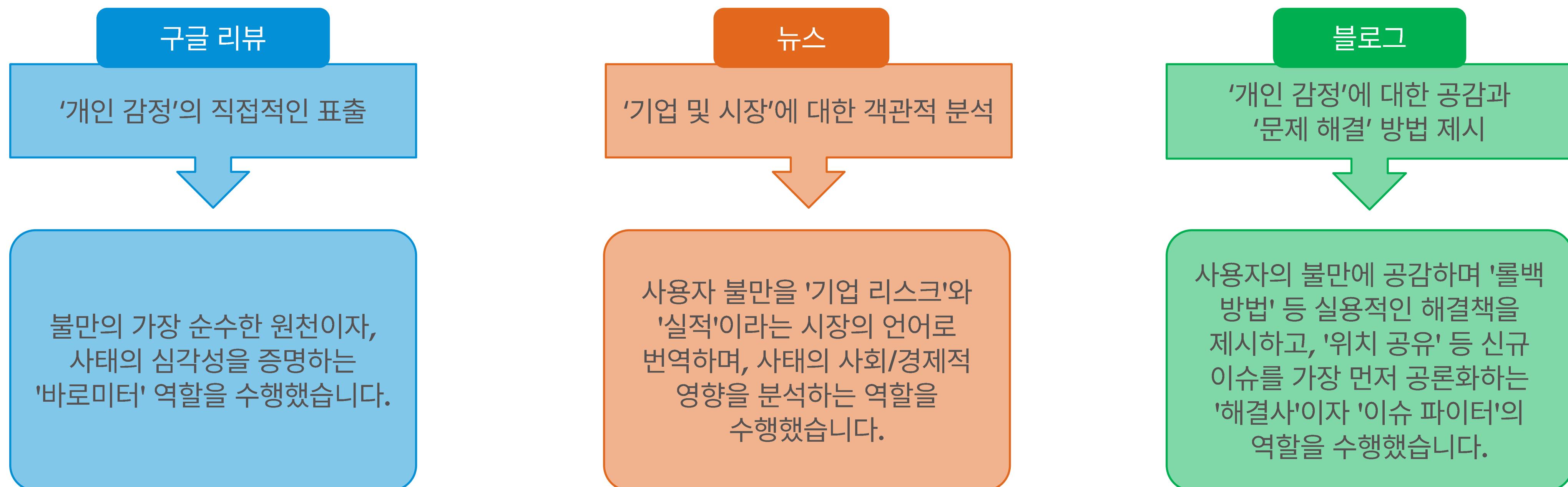
## 02 구간별 '부정' 여론의 관점 차이



# 심층 분석 결과 2. 각 매체는 무엇을 대변하는가?

## 02 구간별 '부정' 여론의 관점 차이

---



# 심층 분석 결과 3. 언론에서 발견한 특이점

## 01 뉴스 본문 부정 감성 기사와 본문 긍정 감성 기사의 네트워크 분석 차이



구간별 뉴스 부정 기사의  
네트워크 분석 결과에서는  
큰 차이가 없었던 반면,

구간별 뉴스 긍정 기사의  
네트워크 분석 결과에서는  
**1구간에서 다양한 목소리를 보여주던**  
**형태**에서  
시간이 지남에 따라 점차 '서비스'라는  
단어를 중심으로 집중화되는 형태로  
변화하는 모습을 확인할 수 있었습니다.

# 심층 분석 결과 3. 언론에서 발견한 특이점

## 02 뉴스 기사의 제목과 본문 네트워크 분석 차이

---



또한, 구간별 뉴스 기사의  
**본문 네트워크는**  
큰 흐름을 중심으로 연결된 망이 그려지는  
반면,  
**제목 네트워크는**  
2-3개의 키워드를 중심으로 개별적으로  
망이 그려진 형태로 구성되고 있습니다.

이는 뉴스를 이용하는 고객들이 보다 많은  
기사를 읽게 하기 위해 다양한 키워드로  
제목을 구성하고 있다는 것을 보여줍니다.

# 심층 분석 결과 3. 언론에서 발견한 특이점

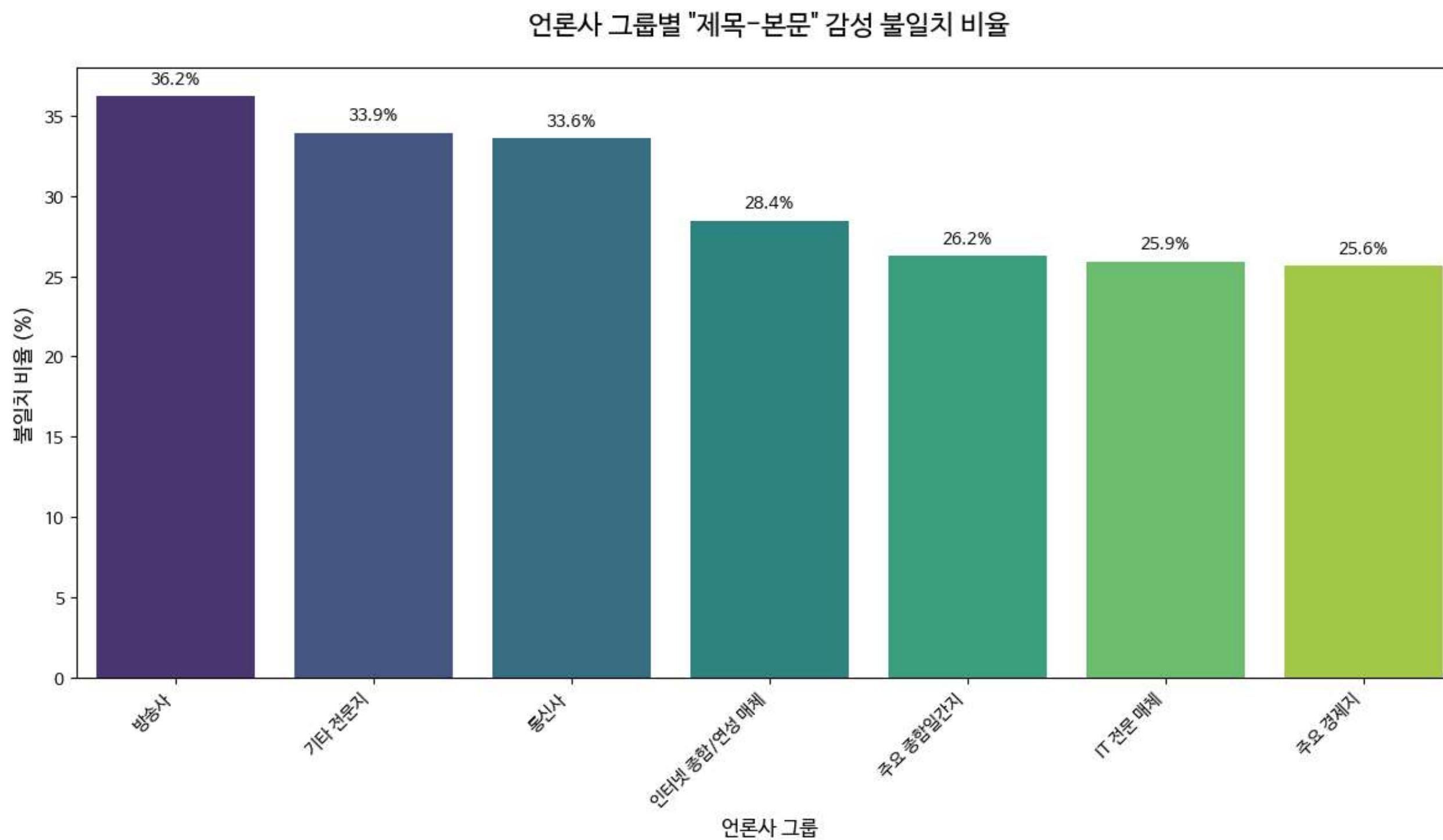
## 02 뉴스 기사의 제목과 본문 네트워크 분석 차이



또한  
동일 감성 / 동일 구간의  
본문과 제목 키워드 및 네트워크 분석  
결과를 확인했을 때  
**제목과 본문에서**  
주로 다뤄지는 **키워드 및 단어 구성이**  
**일치하지 않는 현상**을 확인할 수  
있었습니다.

# 심층 분석 결과 3. 언론에서 발견한 특이점

## 03 '클릭 베이트'와 '양면적 보도'의 증거



이러한 감성 불일치 현상은,

긍정적인 제목으로 클릭을 유도한 뒤 본문에서 비판적인 내용을 덧붙이거나(클릭베이트),  
하나의 사안에 대해 긍정/부정의 양면을 모두 다루려는(양면적 보도) 언론의 보도 행태에서 비롯된 것으로 분석됩니다.

이는 언론 그룹별 감성 불일치 비율을 확인하였을 때 특히 '방송사' 그룹에서 발생 비율이 높은 것을 통해 뒷받침된다고 볼 수 있습니다.

# 심층 분석 결과

	항목의 세부 내용
사용자 목소리의 진화	<p><b>하나의 거대한 분노 목소리 &gt; 여전한 불만, 그러나 열어지는 연결 &gt; 핵심만 남은 불만</b></p> <p>시간이 흐르며 복합적이던 불만이 "친구 탭의 복구"라는 가장 중요한 요소(해결 방안)로 집중된 것을 확인 할 수 있었습니다.</p>
매체별 관점 차이	<p>구글 스토어 리뷰 : 개인(감성 표출) / 뉴스 : 기업에 대한 평가 및 분석 / 블로그 : 개인(얼리 어답터, 해결사)</p> <p>구간별 사용된 핵심 키워드가 구글 스토어 리뷰, 뉴스, 블로그 간 차이가 있는 것을 통해 각 매체별 관점 및 역할의 차이가 있음을 알게 되었습니다.</p>
언론 보도의 이면	<p><b>부정 기사는 일관된 문제 제기를, 긍정 기사는 시간이 지나며 '기업 성장'으로 논조가 집중화되는 차이를 보임.</b></p> <p>또한, 언론보도에서 '제목'과 '본문'의 감성과 내용이 불일치하는 경향이 발견되어 언론 보도를 비판적으로 해석할 필요도 있음을 알게 되었습니다.</p>

# “그래서 앞으로 어떻게 하면 좋을 것인가?”

## 카카오톡 v25.8.0 업데이트 최종 평가

### 카카오톡 관점 : 업데이트 실패

#### 1. 무의식적 사용에서 의식적 고민으로 전환

기존 무비판적으로 사용하던 국민 앱에 대한 의문 제기  
대체 메신저(네이트온, 슬랙 등) 탐색 시작 → 장기적 리스크 발생

#### 2. 핵심 업데이트 묻힌 홍보 실패

친구 탭 · 솟폼 논란에 가려진 긍정(카나나, 읽지 않은 대화 필터 등)적인  
기능들의 평가 제외

#### 3. 하향식(Top-down) 의사결정으로 인한 내부 소통 부재

개발 진행 중 발생된 내부 의견의 소통 불가로 인해 개발 참가자들의 불만  
및 이탈 가능성 상승

### 기업 관점 : 일부 성공

#### 1. 견고한 시장 지배력 입증

큰 불만에도 실제 이탈자 수 극소 → 대체 불가능한 포지션 확인  
카카오 기업 가치 안정성 증명

#### 2. 실적 개선 여지 확보

업데이트 후 이용자 수·실적 발표로 독점적 지위 재확인  
향후 대응에 따라 4Q·26년 1Q 실적 개선 가능성 확보

# “그래서 앞으로 어떻게 하면 좋을 것인가?”

## 분석 진행을 통해 찾은 근본 원인 : 사용자 주권 침해

분석을 통해 확인된 이번 업데이트의 근본적인 실패 원인은 단순한 '친구 탭'에 대한 UI 변경 문제가 아니었습니다.

그보다는 사용자의 '**익숙한 사용 습관**'을 존중하지 않았고,  
**'개인 정보에 대한 통제권'**을 침해했으며,  
이 모든 과정을 '**사전 소통 없이 강제**'했다는,  
**'사용자 주권(User Sovereignty)'**의 문제였다고 생각합니다.

따라서, 앞으로 중요한 것은 **앞으로 어떤 대응을 할 것인지**를 결정하는 것입니다.

# “그래서 앞으로 어떻게 하면 좋을 것인가?”

**단기 : 신뢰 회복을 위한 즉각적인 액션**

## Action 1. '친구 탭'에 대한 완전한 선택권 제공 :

최근 발표된 '친구/소식 탭 분리' 안을 12월 내에 반드시 실행하고, 나아가 '소식' 탭 자체를 설정에서 숨길 수 있는 옵션(Opt-out)까지 제공하여, 사용자에게 완전한 통제권을 돌려주어야 합니다.

측정 지표:

업데이트 이후 '친구 탭 숨기기' 옵션 설정률  
부정 키워드(롤백, 복구) 언급량 감소 추이

## Action 2. 진정성 있는 '사과' 및 '소통' 채널 개설 :

단순한 공지가 아닌 CPO 또는 담당 관리자 명의의 공식적인 사과문을 게시하고, 향후 주요 업데이트 방향성을 미리 공유하고 피드백을 받는 '카카오 NOW'와 같은 공식 소통 채널(구글스토어 리뷰 외에 양방향의 의견을 주고 받을 수 있는)을 명확하게 제공해야 합니다.

측정 지표:

관련 뉴스/블로그의 긍정/부정 감성 비율 변화  
공식 소통 채널의 사용 비율  
구글스토어 리뷰의 급격한 변화 감소

# “그래서 앞으로 어떻게 하면 좋을 것인가?”

## 중장기 : 데이터 기반 의사결정 프로세스 개선

### Action 3. '사용자 경험 지표'의 KPI 강화 :

매출, DAU와 같은 비즈니스 지표 외에,  
'부정 리뷰 비율',  
'핵심 불만 키워드 언급량',  
'특정 기능에 대한 감성 점수'  
등을 프로덕트의 공식적인 핵심 성과 지표(KPI)로 추가하고,  
이를 모니터링하도록 내부 대시보드를 갱신해야 합니다.

### Action 4. 점진적 배포 및 A/B 테스트 의무화 :

모든 사용자에게 일괄 적용하는 '빅뱅' 방식의 업데이트를 자양하고,  
'카나리아 배포(Canary Release)'나 'A/B 테스트'를 통해  
일부 사용자에게 먼저 새로운 기능을 선보인 뒤,  
데이터를 기반으로 확산 여부를 결정하는 프로세스를 의무화해야 합니다.

# “그래서 앞으로 어떻게 하면 좋을 것인가?”

근본적인 체질 개선 : 'Dogfooding' 문화 정착

## Action 5. '사내용 카카오톡' 도입 검토 :

보안 문제를 해결한 '업무용 카카오톡(for Business)' 버전을 개발하여,  
전 직원이 자신들이 만드는 제품을 직접 사용하며 문제점을 가장 먼저 발견하는 'Dogfooding' 문화를 정착시켜야 합니다.  
(예: 네이버 – 네이버웍스, 슬랙 – 업무용 채널 등)

'우리가 우리의 사용자가 되기'를 통해 업무와 일상 모두에서 카카오톡을 이용함으로써  
보다 빠르고 정확한 내부 고객의 피드백을 받을 수 있고,  
'나의 관여가 국민 앱인 카카오톡을 보다 성장시키고 있다'라는 자긍심을 가질 수 있기 때문입니다.

# 데이터 출처 및 분석 방법

분석 대상 기간 : 25.09.22 – 25.11.18 / 데이터 수집 및 분석 : 25.11.19 – 25.12.10

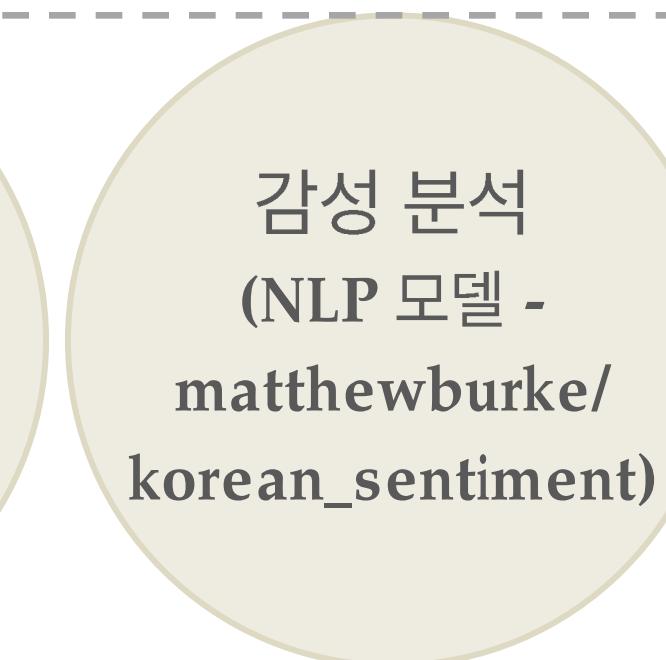
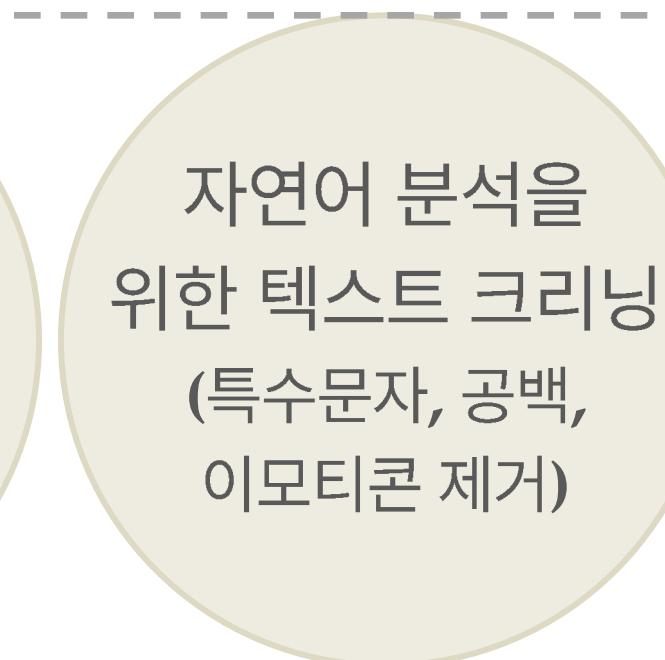
	항목의 세부 내용	비고
구글 스토어 리뷰	초반 일일 리뷰가 너무 빠르게 증가하여 크롤링 방식으로 수집 불가 > google-play-scraping 방식으로 변경하여 수집	-
네이버 뉴스	검색 키워드 (검색 방법 : 관련도순) : 카카오톡 개편, 카카오톡 업데이트, 카톡, 네이트온, 카카오	-
네이버 블로그	검색 키워드 (검색 방법 : 최신순) : 카카오톡 업데이트, 카카오톡 개편, 카톡 업데이트, 카톡 개편, 네이트온, 라인, 카카오톡, 카톡	11월 9일부터 키워드 추가 - '카카오톡, 카톡' 11월 28일부터 검색 방법 변경 - '관련도순'

# 데이터 출처 및 분석 방법

## 01. 뉴스&블로그 수집



## 02. 데이터 수집 및 정제



# 데이터 출처 및 분석 방법

03. 분석 및 시각화

04. 결론 도출



# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 1. 대규모 리뷰 데이터 수집 실패 및 방법론 변경

### 상황 (Problem)

프로젝트 초기, Selenium을 이용한 웹 크롤링 방식으로 구글 스토어 리뷰 수집을 시도했습니다. 그러나 업데이트 직후 폭증한 리뷰로 인해 페이지 스크롤이 길어지면서, 목표 날짜까지 도달하기 전에 메모리 초과(Out of Memory) 오류가 발생하여 데이터 수집이 반복적으로 실패해 데이터 확보에 어려움을 겪었습니다.

### 해결책 (Solution)

직접 크롤링 방식의 한계를 인지한 뒤 해결 방안을 찾던 중 구글 플레이 스토어 데이터를 API 형태로 제공하는 [google-play-scrapers](#) 라이브러리를 알게되어 수집 방법을 전면 변경했습니다. 방법 변경 후 UI 렌더링을 하지 않고 데이터에 직접 접근하여 수집하는 것으로 메모리 초과 문제를 해결했습니다.

### 결과 (Outcome)

새로운 방식을 통해 목표했던 기간(9/22~11/18)의 리뷰 약 15만 건을 안정적이고 신속하게 수집할 수 있었습니다. 이 경험을 통해, 문제 상황에 맞는 최적의 도구를 선택하는 것의 중요성을 학습했습니다.

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 2. 초기 텍스트 정제 규칙의 정보 소실 문제

### 상황 (Problem)

초기 텍스트 정제 시,  
일반적인 NLP 전처리 방식을 적용하여 '한글'을 제외한 숫자, 영어, 특수  
문자를 모두 제거하였습니다.  
그러나 데이터 샘플링 검토 과정에서  
'0점이 없어서 1점', 'cpo님'와 같은 리뷰가  
'점이 없어서 점', '님'으로 정제되면서,  
분석에 중요한 핵심 정보(0, 1, cpo)가 소실되는 문제를 발견했습니다.

### 해결책 (Solution)

분석에서 중요한 정보가 소실되지 않도록 정제 규칙을 재설계했습니다.  
**한글, 영어, 숫자는 보존하되,**  
**분석에 불필요한 이모티콘과 기타 특수 문자만을 선별적으로 제거하는**  
**emoji 라이브러리 기반의 맞춤형 정제 함수를 구현하고 재적용하여 정보**  
**소실 문제를 해결했습니다.**

### 결과 (Outcome)

핵심 정보를 보존함으로써 분석의 정확성과 깊이를 향상시킬 수 있었습니다.  
숫자와 영문을 보존함으로써 사용자의 감정을 더 정확히 파악하고, 비판의 대상을 명확히 특정하는 데 기여했습니다.

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 3. 감성 분석 모델의 정확도 이슈 및 교체

### 상황 (Problem)

구글스토어 리뷰 3점 리뷰(1,2점 – 부정 / 4,5점 – 긍정)의 긍정/부정을 판단하기 위해 범용 한국어 감성 분석 모델(monologg/koelectra-base-v3-discriminator)을 사용했으나, 테스트 결과 일부 리뷰의 맥락을 정확하게 잡아내지 못하는 '리뷰'라는 특정 도메인에서의 성능 한계가 관찰되었습니다.

### 해결책 (Solution)

보다 정확한 감성 분석을 위해 모델 변경에 대한 고민을 하고 리뷰 특성을 분석하는데 적합한 [matthewburke/korean\\_sentiment 모델](#)(이커머스 리뷰를 이용해 학습된 모델)로 교체하여 분석을 재진행하여 결과를 얻었습니다.

### 결과 (Outcome)

도메인에 특화된 모델을 사용함으로써 3점 리뷰에 대한 긍/부정 분류의 신뢰도를 높였고, 이를 통해 전체 감성 분석 결과의 정확성을 향상시킬 수 있었습니다.

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 4. 다중 데이터 소스의 날짜 포맷 불일치 문제

### 상황 (Problem)

총 데이터: 5301건	날짜 변환 실패(NaT): 5074건
--- [ 날짜 변환 실패 상위 5개 샘플 ] ---	
언론사	기사작성일
1 이데일리	2025-10-03 10:00:00
2 SBS	2025.09.29 16:16
3 뉴스1	2025-09-29T16:28:30+09:00
5 헤럴드경제	2025-09-29 16:11:41
6 한국경제	2025.10.02 13:00

초기 수집된 리뷰 데이터와 추가 수집된 리뷰 데이터, 그리고 뉴스 데이터 간에 날짜('at' 컬럼)가 기록된 문자열 포맷이 미세하게 달라, **to\_datetime** 함수 적용 시 **NaT(Not a Time)** 오류가 발생하며 시계열 분석이 불가능했습니다.

### 해결책 (Solution)

데이터 통합 전, 각 데이터 소스별로 **.astype(str)**을 통해 타입을 명시적으로 문자열로 통일하고, **.str.strip()**으로 보이지 않는 공백을 제거한 뒤 **to\_datetime**을 적용하는 **데이터 표준화(Standardization) 단계**를 추가했습니다.

### 결과 (Outcome)

모든 데이터의 날짜 포맷을 성공적으로 통일하여, 정확한 시계열 분석을 수행할 수 있는 기반을 마련했습니다.

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 4. 다중 데이터 소스의 날짜 포맷 불일치 문제

# 현황 파악 : 날짜 변환 실패(NaT)	# 날짜 정제 함수 만들기																																																																										
<pre>total_df['datetime_temp'] = pd.to_datetime(total_df['기사작성일'], errors='coerce')  # NaT 찾기 nat_df = total_df[total_df['datetime_temp'].isnull()]  print(f"총 데이터: {len(total_df)}건") print(f"날짜 변환 실패(NaT): {len(nat_df)}건")  if not nat_df.empty:     print("----")     # 문제 형식 :     display(nat_df)</pre>	<pre># 날짜 정제 함수 만들기  def standardize_date(date_string):     # 'YYYY-MM-DD HH:MM:SS' 표준 형식으로 변환하기      text = str(date_string)      try:         # 1순위: ISO 8601 형식 (예: 2025-09-29T16:28:30+09:00)         if 'T' in text and '+' in text:             return pd.to_datetime(text).strftime('%Y-%m-%d %H:%M:%S')          # 2순위: YTN 형식 등 '오전/오후'가 포함된 모든 케이스 처리         # 예: '2025.11.18 오후 09:22', '2025-09-29, 오전 08:23'         if '오전' in text or '오후' in text:             # 일관된 처리를 위해 모든 구문자를 공백으로 변경             processed_text = text.replace('.', ' ').replace('오전', 'AM').replace('오후', 'PM')             # 여러 개의 공백을 하나로 합침             processed_text = re.sub(r'\s+', ' ', processed_text).strip()          # 가능한 format들을 순서대로 시도         for fmt in ('%Y-%m-%d %p %I:%M', '%Y %m %d %p %I:%M'):             try:                 return pd.to_datetime(processed_text, format=fmt).strftime('%Y-%m-%d %H:%M:%S')             except ValueError:                 continue          # 3순위: 그 외 숫자와 구분자만 있는 형식 (정규표현식)         match = re.search(r'(\d{4})[-./\s]+(\d{1,2})[-./\s]+(\d{1,2})[.\s]+(\d{1,2}):(\d{2})(:(\d{2}))? [날짜 정제 최종 결과 샘플] ---', date_string)         if match:             groups = match.groups()             year, month, day, hour, minute = [int(g) if g else 0 for g in groups[:5]]             second = int(groups[6]) if groups[6] else 0             return f'{year:04d}-{month:02d}-{day:02d} {hour:02d}:{minute:02d}:{second:02d}'          # 위 모든 경우에 해당하지 않으면 원래 텍스트 반환         return text     except Exception:         return text</pre>																																																																										
<table border="1"> <thead> <tr> <th>언론사</th> <th>기사작성일</th> <th>기사작성일_정제</th> </tr> </thead> <tbody> <tr> <td>148 조선일보</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>403 조선일보</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>551 조선일보</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>2820 MBN</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>3169 매일경제</td><td>오류: 'NoneType' object has no attribute 'find'</td><td>오류: 'NoneType' object has no attribute 'find'</td></tr> <tr> <td>3249 MBN</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>3287 MBN</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>3456 서울신문</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>3763 서울경제</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>3777 연합뉴스</td><td>오류: HTTPConnectionPool(host='app.yonhapnews.co... 오류: HTTPConnectionPool(host='app.yonhapnews.co...</td><td>날짜 수집 실패</td></tr> <tr> <td>3787 IT조선</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>3812 연합뉴스</td><td>날짜 수집 실패</td><td>날짜 수집 실패</td></tr> <tr> <td>3942 매일경제</td><td>오류: 'NoneType' object has no attribute 'find'</td><td>오류: 'NoneType' object has no attribute 'find'</td></tr> <tr> <td>3943 매일경제</td><td>오류: 'NoneType' object has no attribute 'find'</td><td>오류: 'NoneType' object has no attribute 'find'</td></tr> <tr> <td>4159 매일경제</td><td>오류: 'NoneType' object has no attribute 'find'</td><td>오류: 'NoneType' object has no attribute 'find'</td></tr> <tr> <td>4222 매일경제</td><td>오류: 'NoneType' object has no attribute 'find'</td><td>오류: 'NoneType' object has no attribute 'find'</td></tr> <tr> <td>4389 매일경제</td><td>오류: 'NoneType' object has no attribute 'find'</td><td>오류: 'NoneType' object has no attribute 'find'</td></tr> <tr> <td>4469 매일경제</td><td>오류: 'NoneType' object has no attribute 'find'</td><td>오류: 'NoneType' object has no attribute 'find'</td></tr> </tbody> </table>	언론사	기사작성일	기사작성일_정제	148 조선일보	날짜 수집 실패	날짜 수집 실패	403 조선일보	날짜 수집 실패	날짜 수집 실패	551 조선일보	날짜 수집 실패	날짜 수집 실패	2820 MBN	날짜 수집 실패	날짜 수집 실패	3169 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'	3249 MBN	날짜 수집 실패	날짜 수집 실패	3287 MBN	날짜 수집 실패	날짜 수집 실패	3456 서울신문	날짜 수집 실패	날짜 수집 실패	3763 서울경제	날짜 수집 실패	날짜 수집 실패	3777 연합뉴스	오류: HTTPConnectionPool(host='app.yonhapnews.co... 오류: HTTPConnectionPool(host='app.yonhapnews.co...	날짜 수집 실패	3787 IT조선	날짜 수집 실패	날짜 수집 실패	3812 연합뉴스	날짜 수집 실패	날짜 수집 실패	3942 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'	3943 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'	4159 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'	4222 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'	4389 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'	4469 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'	<table border="1"> <thead> <tr> <th>기사작성일</th> <th>기사작성일_datetime</th> </tr> </thead> <tbody> <tr> <td>0</td><td>2025.10.04 08:00:00</td><td>2025-10-04 08:00:00</td></tr> <tr> <td>1</td><td>2025-10-03 10:00:00</td><td>2025-10-03 10:00:00</td></tr> <tr> <td>2</td><td>2025.09.29 16:16</td><td>2025-09-29 16:16:00</td></tr> <tr> <td>3</td><td>2025-09-29T16:28:30+09:00</td><td>2025-09-29 16:28:30</td></tr> <tr> <td>4</td><td>2025.09.29 16:13:41</td><td>2025-09-29 16:13:41</td></tr> </tbody> </table>	기사작성일	기사작성일_datetime	0	2025.10.04 08:00:00	2025-10-04 08:00:00	1	2025-10-03 10:00:00	2025-10-03 10:00:00	2	2025.09.29 16:16	2025-09-29 16:16:00	3	2025-09-29T16:28:30+09:00	2025-09-29 16:28:30	4	2025.09.29 16:13:41	2025-09-29 16:13:41
언론사	기사작성일	기사작성일_정제																																																																									
148 조선일보	날짜 수집 실패	날짜 수집 실패																																																																									
403 조선일보	날짜 수집 실패	날짜 수집 실패																																																																									
551 조선일보	날짜 수집 실패	날짜 수집 실패																																																																									
2820 MBN	날짜 수집 실패	날짜 수집 실패																																																																									
3169 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'																																																																									
3249 MBN	날짜 수집 실패	날짜 수집 실패																																																																									
3287 MBN	날짜 수집 실패	날짜 수집 실패																																																																									
3456 서울신문	날짜 수집 실패	날짜 수집 실패																																																																									
3763 서울경제	날짜 수집 실패	날짜 수집 실패																																																																									
3777 연합뉴스	오류: HTTPConnectionPool(host='app.yonhapnews.co... 오류: HTTPConnectionPool(host='app.yonhapnews.co...	날짜 수집 실패																																																																									
3787 IT조선	날짜 수집 실패	날짜 수집 실패																																																																									
3812 연합뉴스	날짜 수집 실패	날짜 수집 실패																																																																									
3942 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'																																																																									
3943 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'																																																																									
4159 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'																																																																									
4222 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'																																																																									
4389 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'																																																																									
4469 매일경제	오류: 'NoneType' object has no attribute 'find'	오류: 'NoneType' object has no attribute 'find'																																																																									
기사작성일	기사작성일_datetime																																																																										
0	2025.10.04 08:00:00	2025-10-04 08:00:00																																																																									
1	2025-10-03 10:00:00	2025-10-03 10:00:00																																																																									
2	2025.09.29 16:16	2025-09-29 16:16:00																																																																									
3	2025-09-29T16:28:30+09:00	2025-09-29 16:28:30																																																																									
4	2025.09.29 16:13:41	2025-09-29 16:13:41																																																																									

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 5. 다수의 언론사 그룹핑 및 분석 단위 설정

### 상황 (Problem)

수집된 뉴스 데이터의 '언론사' 종류가 수십 개에 달해,  
개별 언론사 단위로 분석할 경우 유의미한 패턴을 발견하기 어렵고  
결과가 지나치게 분산되는 문제가 있습니다.

모든 정제된 뉴스 파일 통합 완료!	
-> 총 기사 수: 11148 개	
[ 언론사별 기사 수 ]	
언론사	
뉴스1	203
뉴스2	192
연합뉴스	175
한국경제	167
이데일리	149

총 482개 언론사 중, 기사 수가 50개 이상인 주요 언론사는 총 61개입니다.	
[ 주요 언론사 목록 (기사 수 50개 이상) ]	
언론사	
뉴스1	227
연합뉴스	210
한국경제	209
뉴스2	203
이데일리	201
... 서울와이어	51
비즈니스포스트	50
CEO스코어데일리	50
마이데일리	50
파이낸셜포스트	50
Name: count, Length: 61, dtype: int64	

주요 언론사의 총 기사 수: 6146 개

### 해결책 (Solution)

개별 언론사의 성향을 고려하여  
방송사, IT 전문 매체, 주요 경제지 등 7개의 의미 있는 '언론사 그룹'으로  
재분류(Grouping)하고, 이 그룹을 분석의 기본 단위로 설정하여 비교  
분석을 진행했습니다.

### 결과 (Outcome)

분석의 단위를 적절히 설정함으로써, 개별 언론사의 노이즈를 줄이고 각 그룹의 뚜렷한 논조와 경향성 차이를 성공적으로 발견하고 비교할 수 있었습니다.

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 5. 다수의 언론사 그룹핑 및 분석 단위 설정

```
# 언론사 명에 "\n"이 추가된 경우 제거해서 이상 분리 처리를 수정하기
# combined_df는 이전에 모든 뉴스를 통한 데이터프레임입니다
combined_df['언론사'] = combined_df['언론사'].str.strip()

# 이제 다시 value_counts()를 해보면 '\n'이 사라지고 카운트가
press_counts = combined_df['언론사'].value_counts()
major_press = press_counts[press_counts >= 50] # 50개 이상

# 결정 수준 고민 중

# 기사 수가 50개 이상인 언론사는 몇 개일까?
print(f"50개 이상: {len(press_counts[press_counts >= 50])} 개")

# 기사 수가 20개 이상인 언론사는 몇 개일까?
print(f"20개 이상: {len(press_counts[press_counts >= 20])} 개")

# 기사 수가 10개 이상인 언론사는 몇 개일까?
print(f"10개 이상: {len(press_counts[press_counts >= 10])} 개")

50개 이상: 61 개
20개 이상: 154 개
10개 이상: 245 개
```

언론사	카운트
뉴시스	227
연합뉴스	210
한국경제	209
뉴스1	203
이데일리	201
파이낸셜뉴스	194
서울경제	183
머니투데이	179
매일경제	171
뉴스핌	155
아시아경제	150
데일리안	143
이투데이	141
서울신문	141
헤럴드경제	132
전자신문	130
디지털데일리	129
조선비즈	125
YTN	121
아이뉴스24	121
SBS Biz	113
노컷뉴스	107
디지털투데이	100
아주경제	98
지디넷코리아	97
디지털타임스	93
테크M	91
EBN	91
인사이트	87
연합뉴스TV	85
조선일보	85
톱스타뉴스	85
중앙일보	80
SBS	80
MBN	78
세계일보	76

### [ 그룹 1: 통신사 (News Agencies) ]

- 특징: 가장 기본적인 1차 뉴스 생산자. 웹사이트 구조가 비교적 단순하고 텍스트 위주인 경우가 많습니다. (가장 먼저 공략하기 좋은 그룹)
- 목록: 뉴스1, 뉴시스, 연합뉴스

### [ 그룹 2: 주요 경제지 (Major Economic Dailies) ]

- 특징: 경제/산업 분야에 집중. 비슷한 독자층을 대상으로 하므로 웹 구조가 유사할 가능성이 높습니다.
- 목록: 한국경제, 이데일리, 파이낸셜뉴스, 서울경제, 머니투데이, 매일경제, 헤럴드경제, EBN, 머니S, 이뉴스투데이, 이코노믹리뷰, 브릿지경제, 글로벌이코노믹, 비즈니스포스트, CEO스코어데일리, 파이낸셜포스트

### [ 그룹 3: IT 전문 매체 (IT/Tech Media) ]

- 특징: IT, 테크, 스타트업 전문. 이 그룹 역시 비슷한 CMS를 사용할 확률이 높아 크롤러 재사용성이 높습니다.
- 목록: 뉴스핌, 전자신문, 디지털데일리, 조선비즈, 아이뉴스24, 디지털투데이, 지디넷코리아, 디지털타임스, 테크M, IT조선

### [ 그룹 4: 주요 종합일간지 (Major General Dailies) ]

- 특징: 전통적인 주요 신문사. 웹사이트가 크고 복잡하며 광고가 많을 수 있지만, 한 번 템플릿을 파악하면 다른 곳에 적용하기 좋습니다.
- 목록: 조선일보, 중앙일보, 세계일보, 국민일보, 동아일보, 서울신문, 메트로신문, 매일일보

### [ 그룹 5: 방송사 (Broadcasting Companies) ]

- 특징: TV 방송을 기반으로 한 언론사. 동영상 콘텐츠가 많고, 텍스트 본문 구조가 다른 그룹과 다를 수 있습니다.
- 목록: YTN, SBS Biz, 연합뉴스TV, SBS, MBN, KBS, 한국경제TV, 서울경제TV

### [ 그룹 6: 인터넷 종합/연예 매체 (Internet General/Entertainment Media) ]

- 특징: 온라인을 기반으로 한 종합/연예 뉴스. 클릭을 유도하는 자극적인 제목이나 이미지 위주의 레이아웃이 많을 수 있습니다.
- 목록: 데일리안, 이투데이, 아시아경제, 노컷뉴스, 아주경제, 인사이트, 톱스타뉴스, 뉴스웨이, 뉴데일리, 포인트데일리, 천지일보, 핀포인트뉴스, 데일리한국, 서울와이어, 마이데일리

### [ 그룹 7: 기타 전문지 (Others) ]

- 특징: 위 그룹에 속하지 않는 기타 전문 분야 매체.
- 목록: 중앙이코노미뉴스

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 6. 파일명 내 특수문자로 인한 저장 오류

### 상황 (Problem)

분석 결과 그래프를 그룹별로 자동 저장하는 과정에서, '인터넷 종합/연성 매체'와 같이 슬래시(/)가 포함된 그룹 이름이 파일명으로 사용되면서, 시스템이 이를 폴더 경로로 오인하여 `FileNotFoundException`가 발생했습니다.

### 해결책 (Solution)

파일 이름을 생성하기 전  
파일명으로 사용할 수 없는 특수문자(/, \, \* 등)를 안전한 문자(예: \_)로  
자동 변환해주는 `sanitize_filename` 헬퍼 함수를 구현하여 파일 저장  
로직에 적용했습니다.

### 결과 (Outcome)

파일 이름을 생성하기 전에, 파일명으로 사용할 수 없는 특수문자(/, \, \* 등)를 안전한 문자(예: \_)로 자동 변환해주는 `sanitize_filename` 헬퍼 함수를 구현하여 파일 저장 로직에 적용했습니다.

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 7. 대량 텍스트 데이터 처리 중 메모리 초과 발생

### 상황 (Problem)

블로그 데이터에 대한 '네트워크 분석' 실행 시, 모든 단어의 연결 관계(약 수십만~수백만 쌍)를 한 번에 메모리에 생성하는 과정에서 구글 코랩(Colab)의 사용 RAM을 초과하여 런타임이 반복적으로 중단되는 문제가 발생했습니다.

### 해결책 (Solution)

전체 데이터를 한 번에 처리하는 대신, 10,000건 단위의 작은 덩어리(chunk)로 나누어 순차적으로 처리하고, 각 덩어리의 결과(Counter)를 점진적으로 업데이트하는 방식으로 코드를 수정했습니다. 또한, 각 루프가 끝날 때마다 파이썬의 가비지 컬렉터(`gc.collect()`)를 강제 호출하여 불필요한 메모리를 즉시 해제하도록 했습니다.

### 결과 (Outcome)

메모리 사용량을 안정적인 수준으로 유지하며, 제한된 리소스 환경에서도 대용량 텍스트 데이터에 대한 복잡한 네트워크 분석을 성공적으로 완료할 수 있었습니다. 이를 통해 대용량 데이터 처리 시 메모리 최적화(Memory Optimization)의 중요성과 기법을 학습했습니다.

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 7. 대량 텍스트 데이터 처리 중 메모리 초과 발생

```
# --- [5단계] 네트워크 분석 (데이터 계산 및 시각화) ---
def step5_analyze_and_visualize_network(df_processed, sentiment_column, word_freq_results, top_n=20, save_prefix=""):
    print("\n[STEP 5] 네트워크 분석 및 시각화를 시작합니다...")
    for sentiment in sorted(df_processed[sentiment_column].unique()):
        documents = df_processed[df_processed[sentiment_column] == sentiment][['keywords']].tolist()
        documents = [doc for doc in documents if doc]
        if not documents: continue

        pairs = [combination for keywords in documents if len(keywords) >= 2 for combination in combinations(set(keywords), 2)]
        if not pairs: continue

        pair_counts = Counter(pairs)
        df_edges = pd.DataFrame(pair_counts.most_common(top_n), columns=['pair', 'count'])
        df_edges[['source', 'target']] = pd.DataFrame(df_edges['pair'].tolist(), index=df_edges.index)
        G = nx.from_pandas_edgelist(df_edges, source='source', target='target', edge_attr='count')

        print(f" -> '{sentiment}' 그룹 네트워크 시각화...")
        plt.figure(figsize=(18, 18))
        pos = nx.spring_layout(G, k=0.8, iterations=100, seed=42)
        node_sizes = [np.sqrt(word_freq_results[sentiment].get(node, 1)) * 300 for node in G.nodes()]
        edge_widths = [d['count'] * 0.05 for (u, v, d) in G.edges(data=True)]

        nx.draw_networkx(G, pos,
                         font_family='NanumBarunGothic', font_size=14,
                         node_size=node_sizes, node_color=list(dict(G.degree()).values()),
                         cmap=plt.cm.viridis, alpha=0.8,
                         width=edge_widths, edge_color='grey')

        plt.title(f'{save_prefix}{sentiment} 키워드 네트워크 분석', fontsize=25)
        plt.axis('off')
        plt.show()

# --- [5단계] 네트워크 분석 (메모리 최적화 버전) ---
def step5_analyze_and_visualize_network_optimized(df_processed, sentiment_column, word_freq_results, top_n=40, save_prefix="", chunk_size=10000):
    print("\n[STEP 5] 네트워크 분석 및 시각화 (메모리 최적화 버전)를 시작합니다...")
    for sentiment in sorted(df_processed[sentiment_column].unique()):
        print(f" -> '{sentiment}' 그룹 네트워크 데이터 계산 중...")
        documents = df_processed[df_processed[sentiment_column] == sentiment][['keywords']].tolist()
        documents = [doc for doc in documents if doc]
        if not documents: continue

        # 🔴 전체 pairs를 한 번에 만들지 않고, Counter를 점진적으로 업데이트
        pair_counts = Counter()
        for i in tqdm(range(0, len(documents), chunk_size), desc=" - 단어 쌍 계산 진행률"):
            chunk_documents = documents[i:i+chunk_size]
            pairs_chunk = [
                combination for keywords in chunk_documents
                if len(keywords) >= 2
                for combination in combinations(set(keywords), 2)
            ]
            pair_counts.update(pairs_chunk)
            del pairs_chunk # 메모리에서 즉시 해제
            gc.collect() # 가비지 컬렉션 강제 실행

        if not pair_counts: continue

        # (미하 시각화 코드는 미첨과 동일)
        df_edges = pd.DataFrame(pair_counts.most_common(top_n), columns=['pair', 'count'])
        df_edges[['source', 'target']] = pd.DataFrame(df_edges['pair'].tolist(), index=df_edges.index)
        G = nx.from_pandas_edgelist(df_edges, source='source', target='target', edge_attr='count')

        print(f" -> '{sentiment}' 그룹 네트워크 시각화...")
        plt.figure(figsize=(18, 18))
        pos = nx.spring_layout(G, k=0.8, iterations=100, seed=42)
        node_sizes = [np.sqrt(word_freq_results[sentiment].get(node, 1)) * 300 for node in G.nodes()]
        edge_widths = [d['count'] * 0.05 for (u, v, d) in G.edges(data=True)]

        nx.draw_networkx(G, pos,
                         font_family='NanumBarunGothic', font_size=14,
                         node_size=node_sizes, node_color=list(dict(G.degree()).values()),
                         cmap=plt.cm.viridis, alpha=0.8,
                         width=edge_widths, edge_color='grey')

        plt.title(f'{save_prefix}{sentiment} 키워드 네트워크 분석', fontsize=25)
        plt.axis('off')
        plt.show()
```

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 8. 분석 프로세스 비일관성 및 파이프라인 재설계

### 상황 (Problem)

프로젝트 초기 구글스토어 리뷰 수집 후 분석, 뉴스 기사 수집 후 분석, 블로그 수집 후 분석과 같이 **순차 분석으로 인해 각 데이터 소스별로 작성된 개별 분석 스크립트는** 코드의 중복을 야기하고, 소스 간 분석 결과의 일관성 있는 비교를 어렵게 만드는 **문제가** 있었습니다.

### 해결책 (Solution)

분석의 일관성과 재사용성을 높이기 위해, 기존 코드를 **모듈화된 '범용 분석 파이프라인'**으로 재설계(**Refactoring**)했습니다. 키워드 추출, 빈도 분석, **n-gram** 분석 등 각 단계를 독립적인 함수로 정의하고, 데이터셋과 상관없이 동일한 프로세스로 분석할 수 있도록 구조를 **개선했습니다.**

### 결과 (Outcome)

이 '범용 분석 프레임워크' 구축을 통해 모든 **비교 분석을 일관된 기준 하에 신속하게 수행**할 수 있었으며, 분석의 신뢰도와 효율성을 크게 높일 수 있었습니다. 이 경험을 통해 '잘 설계된 분석 프로세스'의 중요성을 체감했습니다.

# 데이터 출처 및 분석 방법

## 분석 과정 중 주요 문제 해결 사례 8. 분석 프로세스 비일관성 및 파이프라인 재설계

```

# 뉴스/블로그에서 제목과 본문을 나눠서 분석하는 것에 대한 처리
def reshape_news_blog_data(df, title_col, title_sentiment_col, body_col, body_sentiment_col):
    """
    뉴스/블로그 데이터를 분석하기 쉬운 Long Format으로 변환합니다.
    제목과 본문을 별도의 행으로 분리합니다.
    """
    print("뉴스/블로그 데이터를 분석용 구조로 변환합니다...")

    # 1. 제목 데이터프레임 생성
    df_title = df[[title_col, title_sentiment_col]].copy()
    df_title.columns = ['text', 'sentiment']
    df_title['source_type'] = '제목'

    # 2. 본문 데이터프레임 생성
    df_body = df[[body_col, body_sentiment_col]].copy()
    df_body.columns = ['text', 'sentiment']
    df_body['source_type'] = '본문'

    # 3. 두 데이터프레임을 위아래로 합치기
    df_reshaped = pd.concat([df_title, df_body], ignore_index=True)
    # --- 1. 분석 환경 설정 ---
    # 분석할 데이터와 결과 파일 경로 등을 여기에 지정합니다.

    # 4. 분석 그룹을 명확히 하기 위해 새로운 컬럼 생성 (예: '제목_긍정')
    df_reshaped['analysis_group'] = df_reshaped['source_type'] + '_' + df_reshaped['sentiment']

    # 5. 비어있는 텍스트 행은 분석에서 제외
    df_reshaped = df_reshaped.dropna(subset=['text'])
    df_reshaped = df_reshaped[df_reshaped['text'].str.strip() != '']

    print("✓ 데이터 구조 변환 완료!")
    return df_reshaped

# [네이버 뉴스, 블로그] blog, news
ANALYSIS_TARGET = 'news'
SOURCE_DATA_PATH = 'final_news_with_sentiment.pkl'
PROCESSED_DATA_PATH = '251201_뉴스_keywords.pkl'
SAVE_PREFIX = '뉴스_'
# 아래 컬럼 이름들은 실제 데이터에 맞게 수정 필요
TITLE_COL = '제목_cleaned'
TITLE_SENTIMENT_COL = '제목_감성'
BODY_COL = '본문_cleaned'
BODY_SENTIMENT_COL = '본문_감성'

# 분석에 사용될 공통 불용어 리스트
common_stopwords = [
    '등', '수', '이', '것', '저', '그', '때', '및', '의', '를', '에', '가', '를', '좀', '잘', '더', '그냥',
    '카카오', '카톡', '업데이트', '업뎃', '이번', '버전', '기자', '뉴스', '사진', '제공', '무단', '전재'
]

```

```

# --- 2. 분석 파이프라인 실행 ---
print(f"--- '{SAVE_PREFIX}' 전체 분석을 시작합니다 ---")
try:
    # 개선된 함수를 호출합니다.
    check_and_set_korean_font(stop_on_fail=True)
except SystemExit as e:
    # 폰트 문제로 중단되었을 경우 메시지를 출력하고 종료합니다.
    print(e)
else:
    # 폰트가 정상적일 경우에만 아래의 모든 분석을 실행합니다.

    # [STEP 1] 키워드 추출
    if not os.path.exists(PROCESSED_DATA_PATH):
        print(f"\n[STEP 1] '{PROCESSED_DATA_PATH}' 파일이 없어 키워드 추출을 시작합니다.")
        df_raw = pd.read_pickle(SOURCE_DATA_PATH)
        # 분석 대상에 따라 데이터 구조 결정
        if ANALYSIS_TARGET in ['blog', 'news']:
            df_to_process = reshape_news_blog_data(df_raw, TITLE_COL, TITLE_SENTIMENT_COL, BODY_COL, BODY_SENTIMENT_COL)
        # 키워드 추출 함수에는 변환된 헤더 이름을 전달
        df_processed = step1_extract_and_save_keywords(df_to_process, ['text'], common_stopwords, PROCESSED_DATA_PATH)
        # 분석 그룹(的感情) 컬럼을 새로 만든 'analysis_group' 으로 지정
        SENTIMENT_COLUMN = 'analysis_group'
    else:
        # 구글 리뷰 또는 기타 단일 텍스트 데이터
        df_processed = step1_extract_and_save_keywords(df_raw, TEXT_COLUMNS, common_stopwords, PROCESSED_DATA_PATH)

    else:
        print(f"\n[SKIP] 이미 '{PROCESSED_DATA_PATH}' 파일이 존재하여 1단계를 건너뛰고 파일을 불러옵니다.")
        df_processed = pd.read_pickle(PROCESSED_DATA_PATH)
        # 저장된 파일을 불러올 때도 분석 대상에 따라 SENTIMENT_COLUMN을 재정의 해줘야 함
        if ANALYSIS_TARGET in ['blog', 'news']:
            SENTIMENT_COLUMN = 'analysis_group'

    # [STEP 2] 키워드 빈도 분석 및 시각화
    word_freq_results = step2_calculate_frequency(df_processed, SENTIMENT_COLUMN)
    for sentiment, counts in word_freq_results.items():
        if counts:
            visualize_frequency(sentiment, counts, save_prefix=SAVE_PREFIX)

    # [STEP 3] N-gram 분석 및 시각화
    step3_analyze_and_visualize_ngrams(df_processed, SENTIMENT_COLUMN, save_prefix=SAVE_PREFIX)

    # [STEP 4] LDA 토큰 모델링
    step4_run_lda_modeling(df_processed, SENTIMENT_COLUMN, num_topics=4, save_prefix=SAVE_PREFIX)

    # [STEP 5] 네트워크 분석 및 시각화
    # 2단계에서 계산한 단어 빈도 결과를 노드 크기 조절에 사용합니다.
    step5_analyze_and_visualize_network(df_processed, SENTIMENT_COLUMN, word_freq_results, save_prefix=SAVE_PREFIX)

print("\n 모든 분석이 성공적으로 완료되었습니다. ")

```