x gigs

2024 디지털하나로 미니프로젝트 [T사이 F조]

고객 군집분석 및 후기데이터 감성분석을 통한 아웃소싱 플랫폼 개선 프로젝트

2024.03.29

| 김수정 송재은 안성재 이우재 전예진 한재원

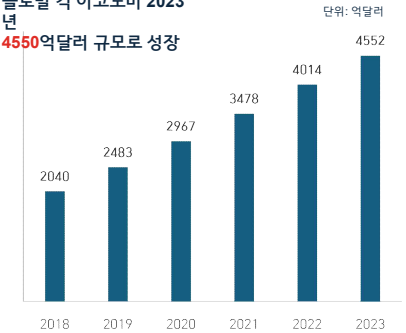

INDEX
01 추진배경
02 현황
03 목표 설정
04 데이터 분석 계획
05 분석 결과 및 인사이트 도출
06 개선안
07 프로젝트 소감


I
II
III
IV
V
VI
추진배경

"긱 워커"

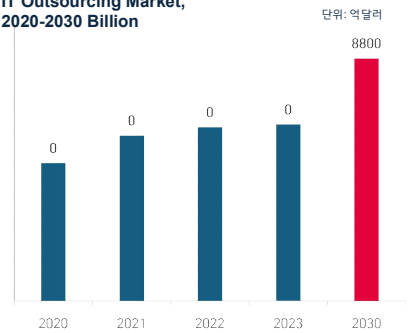
: 디지털 플랫폼 등을 통해 단기로 계약을 맺고 일회성 일을 맡는 등 초단기 노동을 제공하는 근로자를 이르는 말

글로벌 긱 이코노미 2023년
4550억달러 규모로 성장



* 마스터카드 앤드 카이저어소시에이츠

IT Outsourcing Market, 2020-2030 Billion



예측기간 동안 연평균 성장률 585.60% 성장예정

IT 아웃소싱 플랫폼의 포화상태 속 차별화된 전략 필요


I
II
III
IV
V
VI
현황

	크몽	위시캣	원티드	숨고
장점	▪ 단순 채팅시 과금X ▪ 다양한 도메인 ▪ 매칭률 높은 편	▪ 프로젝트 매니저가 존재해 본업에만 충실할 수 있는 구조	▪ 높은 등급의 프리랜서의 경우 수수료 7%인하 ▪ 착수금 지급 보장, 명절 선물 증정 등 개선된 처우 제공	▪ 간단한 서비스 ▪ 가격 비교 가능 ▪ 빠른 서비스 제공 속도
단점	▪ 높은 수수료 ▪ 부분 출금 불가 ▪ 불합리한 타이머제도	▪ IT분야에만 특화 -> 도메인 한정적	▪ 8-90% 개발 직군 프로젝트 -> 도메인 한정적	▪ 체계적인 고수 관리 시스템 부재 ▪ 과도한 중개비
수수료	▪ 판매자 총 거래금액에서 서비스 이용료 15% + 결제 수수료 3.3%	▪ 10%	▪ [프리랜서] 전체 계약금의 10% 매칭비 ▪ [클라이언트] 프리랜서 매칭 성사 시 5% 기본 수수료 부과	▪ 견적서 및 채팅 당 수수료 부과 ▪ 마켓 등록 상품의 경우 총 판매금액의 13%

체계적인 수수료율 기준
간단한 서비스나 매칭률 등 서비스 편의성이 장점

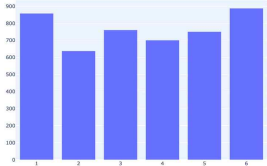
[문제점 ①] 적정 수수료의 부재 및 거래 사유화로 사이트 내에서 거래가 지속되지 않고 있음.
[문제점 ②] 같은 분야의 서비스끼리 적정가격을 책정하지 못해, 사용하는 고객들의 불만이 높아지고 있다.

크몽에서는 얼마에 해주던데요?

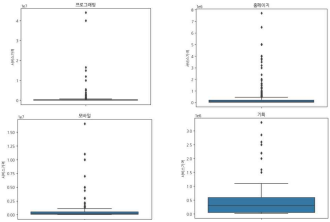
내부에서 수준에 따른 단가에 대한 하한선이 지정되지 않아
전문가들 간의 단가가 터무니없이 꺾이며,
전문가들이 전혀 보호받지 못해 이탈하는 상황

[출처: 경기청년유니온 설문조사]

▲ 월별 판매자 수, 월별 서비스 개수가 지속적 감소 중



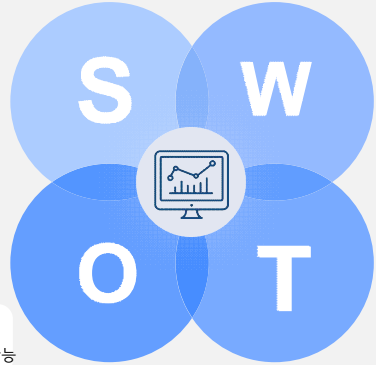
▲ 월별 거래취소 수의 증가



▲ 대부분의 품목이 일정하지 못한 가격폭을 지님

Strengths

국내 최대 IT 아웃소싱 플랫폼
IT 관련 전문 기술 **수요 증가**
다양한 전문가 풀 확보



Weakness

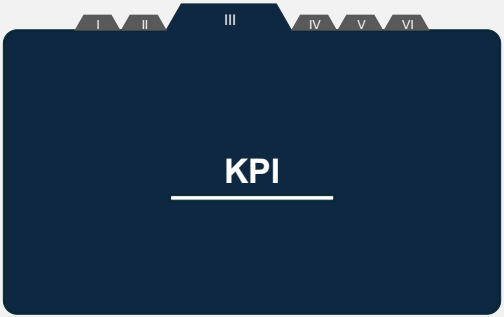
적정가격 책정 미흡에 대한
고객의 불만과 전문가의 이탈 증가
외주요청의 **늦은 확인**
높은 추가금액으로 인한 거래 중단

Opportunities

재구매율의 지속적인 **향상**
기존 경쟁사에서 미시행중인 서비스 발굴 가능

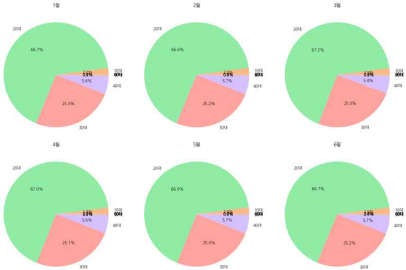
Threats

동종 업체의 공격적인 마케팅
전문가 및 고객의 이탈
거래 사유화


I
II
III
IV
V
VI
KPI

연령대, 성별 등 대부분 컬럼의 월별 분포 차이 無

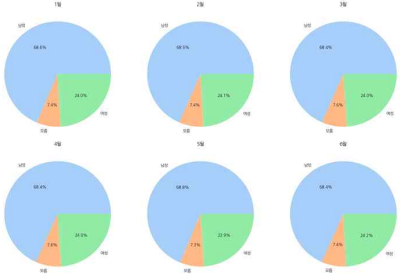
연령대 분포



월별 연령대 분포 카이제곱 독립성 검정 결과
: 월별 연령대 분포에 차이가 없다.

Statistic	P-value
41.52081	0.20774

성별 분포

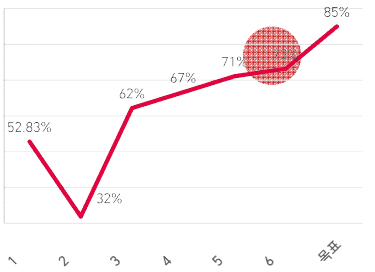


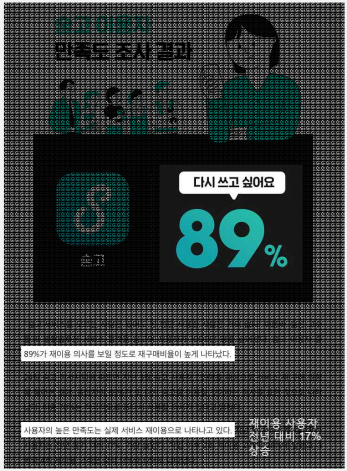
월별 성별 분포 카이제곱 독립성 검정 결과
: 월별 성별 분포에 차이가 없다.

Statistic	P-value
9.51976	0.20774

[KPI] 고객 만족도를 높여 재구매율 85%달성

출처: 스마트투데이 이민하 기자



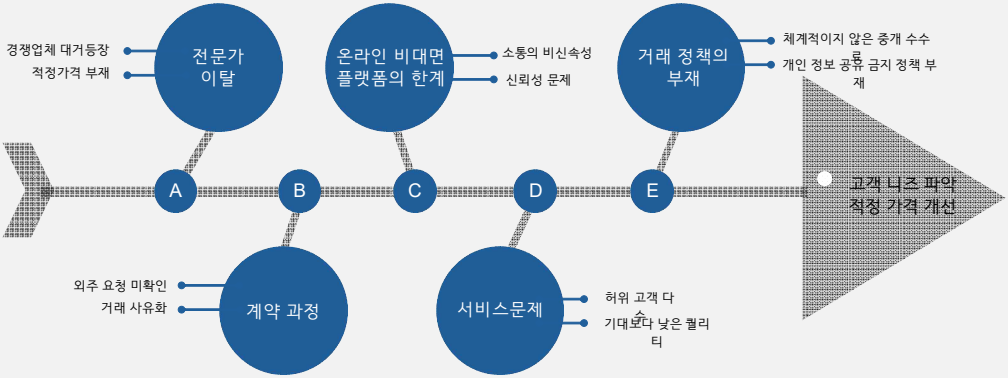


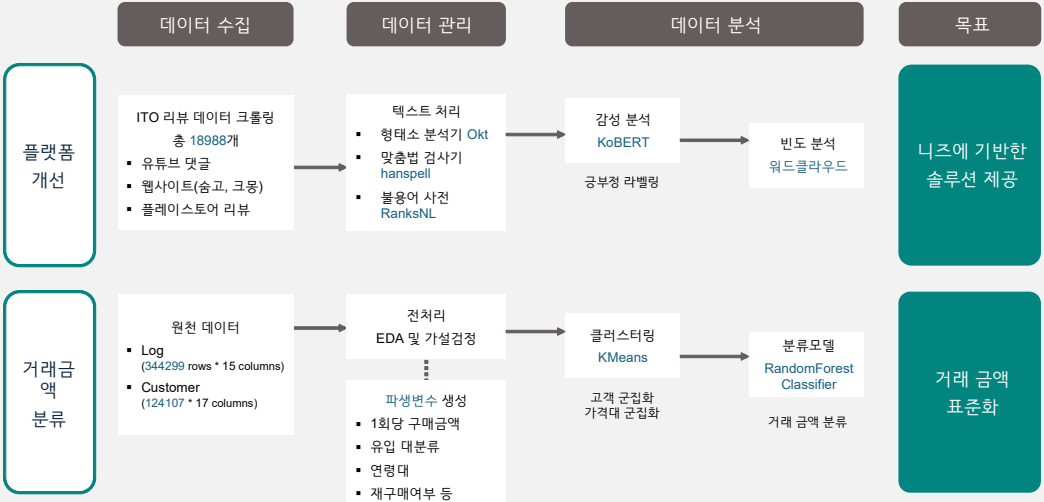
Log 데이터에서 고객별 구매 기록 검사 후 재구매여부 분류 진행	
첫 구매	빈 리스트에 고객 ID를 추가한 후 재구매 여부를 0으로
재구매	고객ID가 리스트에 이미 존재하는 경우 재구매로 간주하여 1로

I II III IV V VI

분석 계획

1. 변수요인도
2. 분석계획서


경쟁업체 대거등장
적정가격 부재
전문가 이탈
온라인 비대면 플랫폼의 한계
소통의 비신속성
신뢰성 문제
거래 정책의 부재
체계적이지 않은 중개 수수
개인 정보 공유 금지 정책 부재
A
B
C
D
E
고객 니즈 파악
적정 가격 개산
외주 요청 미확인
거래 사유화
계약 과정
서비스문제
허위 고객 다
기대보다 낮은 퀄리티


데이터 수집
데이터 관리
데이터 분석
목표
플랫폼 개선
ITO 리뷰 데이터 크롤링
총 18988개
유튜브 댓글
웹사이트(숨고, 크몽)
플레이스토어 리뷰
텍스트 처리
형태소 분석기 Okt
맞춤법 검사기 hanspell
불용어 사전 RanksNL
감성 분석
KoBERT
긍부정 라벨링
빈도 분석
워드클라우드
니즈에 기반한 솔루션 제공
거래금액 분류
원천 데이터
Log
(344299 rows * 15 columns)
Customer
(124107 * 17 columns)
전처리
EDA 및 가설검정
파생변수 생성
1회당 구매금액
유입 대분류
연령대
재구매여부 등
클러스터링
KMeans
고객 군집화
가격대 군집화
분류모델
RandomForest Classifier
거래 금액 분류
거래 금액 표준화

I II III IV V VI

분석① 플랫폼 개선

후기데이터 기반고객 니즈 파악

고객 니즈 분석

고객 후기 데이터 크롤링

분석 목표

- 경쟁사인 숨고와 크몽의 후기 데이터 크롤링
- 긱스의 주 상품인 IT 외주 후기데이터 크롤링

→ 고객 니즈 분석

분석 방법

- **Selenium**

리뷰 감정평가 모델 구축

분석 목표

- 영화 후기 긍·부정 라벨링된 데이터를 **Bert** 기반으로 **토큰화** 및 **딥러닝** 학습
- 감정평가 모델 구축하여 크롤링한 리뷰 데이터의 긍정·부정을 구별

분석 방법

- **KoBERT**, **Sentimental Analysis**

검증 방법

- **Accuracy**

빈도 분석

분석 목표

- 후기데이터 긍정확률을 기반으로 **부정(-1), 중립(0), 긍정(1)**로 구분
- 긍정 리뷰와 부정 리뷰를 빈도분석하여 고객의 니즈를 분석 및 개선하고자 함

분석 방법

- **Okt**, WordCloud

Sentimental Model

ACC: 90.7%

긍정

유튜브

크몽사이트

플레이스토어

숨고사이트

부정

유튜브

크몽사이트

플레이스토어

숨고사이트

	긍정리뷰	부정리뷰
상위 빈도 단어	▪ 잘, 감사합니다, 정말, 작업, 님, 고수, 도움, 설명, 요청, 부분, 친절하게, 좋은, 시간	▪ 견적, 돈, 수수료, 캐시, 비용, 이용, 결제 , 가격, 서비스, 연락, 의뢰, 환불, 업체
비율	▪ 13868 문장 중 7357 문장 53.05%	▪ 5120 문장 중 2592 문장 50.625%
분석 내용	▪ 약속시간 잘 지켜짐 ▪ 자세하고 친절한 설명 ▪ 빠른 매칭 ▪ 합리적인 견적	▪ 원치 않는 다량의 견적서 수신 ▪ 늦어지는 작업물 ▪ 수신한 견적서의 넓은 가격 분포 ▪ 자동견적 시 맞지 않는 업체 추천

주요 단어 기반 고객 니즈 개선 및 리뷰에 기반한 전문가 평가 지표 제작

I II III IV V VI

플랫폼 개선

개선안① 후기데이터 기반고객 니즈 파악

주요 단어 기반 고객 니즈 개선

목적

- 주요 단어를 통한 문제점 식별
- 긍정적 요인 강화, 부정적 요인 보완

개선방안

- 주요 단어를 통한 고객 니즈 기반 서비스 제안

기대효과

- 고객 니즈에 맞는 의뢰 환경 조성

솔큐러시 기반 수수료율 재산정

목적

- 솔큐러시 (like 당근 매너온도) 를 통한 전문가의 등급 선정
- 등급에 맞는 수수료 할인율 제공

개선방안

- 긍정일 확률 * 별점 = 거래 당 올라가는 솔큐러시
- 별점은 5점인데 평가는 안 좋음
  → 긍정확률이 0에 수렴해서 온도는 0만 올라감

기대효과

- 고객의 의뢰 전문가 선택시 직관적인 판단 도움
- 성과지표의 가시화로 타 플랫폼으로의 전문가 유출 방지

긍정 리뷰	#약속시간 잘 지킴 #빠른 매칭 #자세하고 친절한 설명 #합리적인 견적

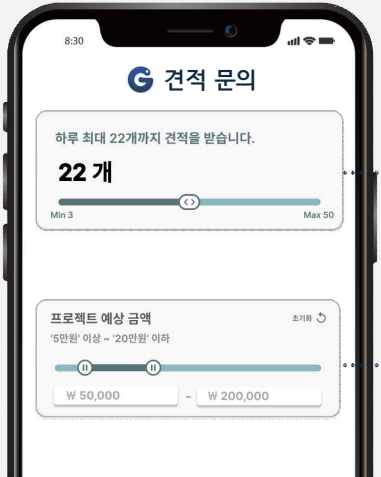
POINT

- 고객이 긍정으로 느끼는 포인트를 리뷰 태그로 제작
- 필터링을 통해 원하는 고수를 찾을 수 있도록 설정

EFFECT

- 조건이 맞는 전문가에게 먼저 고객 요청을 보내 니즈에 맞는 전문가와 매칭될 확률 UP

부정 리뷰	#원치않는 다량의 견적서 수신 #수신한 견적서의 넓은 가격 분포



POINT

- 고객이 받아볼 견적서 개수를 사전 설정하도록 변경
- EFFECT) 고객의 편의성 증진

- 고객이 외주 요청시 견적서의 가격대를 설정하도록 변경
- 현재는 가장 낮은 판매가를 기준으로 전문가가 검색되지만,
  받는 견적서의 금액대는 고객 요청에 따라 매우 다양
- EFFECT, 원하는 가격대의 견적서만 받아 빠른 계약 결정 가능

솔큐러시 = Solve + Accuracy

리뷰 데이터

- 긍정 확률 도출
- 평점 Scaling
  1~5 -> -2~2

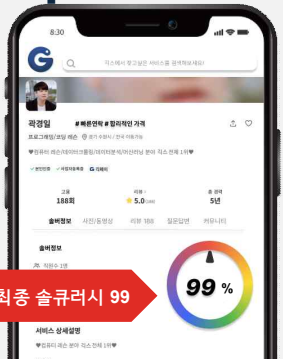
기본 솔큐러시 30 설정

- 긍정 확률 기반 솔큐러시 도출
- 기본 30 솔큐러시 + 전문가 전체 Quantile Transformer값 * 70

곽경일 강사님 리뷰 기반 **솔큐러시** 계산 구조

리뷰	평점	긍정확률	솔큐러시
아주셨어요ㅠㅠ...	2	0.99	1.98
주 의뢰드릴게요!	2	0.96	1.92
품성이 좋으셔요~	2	0.96	1.92
아주셨어요ㅠㅠ...	2	0.99	1.98
주 의뢰드릴게요!	2	0.96	1.92

매 거래마다 솔큐러시가 누적되며,
99%을 최대 솔큐러시로 설정



최종 솔큐러시 99

벤치마킹 기업 선정 – “*지그재그*”

kakaostyle zigzag

유사한 디지털 마켓플레이스 구조를 지닌 기업 중 시행하고자 하는 정책을 시행중인 기업 선정

중개자 역할

- **지그재그**와 **긱스** 모두 판매자와 구매자 사이의 **중개 플랫폼** 역할
- 플랫폼을 통해 제품이나 서비스를 판매하고 구매할 수 있는 편리한 방법 제공

플랫폼 기반의 BM

- 지그재그와 긱스 **모두 플랫폼 기반의 비즈니스 모델** 채택하여 운영 중

리뷰 평점

- 사용자들이 제품이나 서비스에 대한 **리뷰와 평점**을 남길 수 있게 하여 타 유저들의 구매 결정에 도움을 줌

▪ 벤치마킹 이유

지그재그의 수수료범위가 긱스와 비슷함 (5.5 ~ 9.5% 범위 이용)

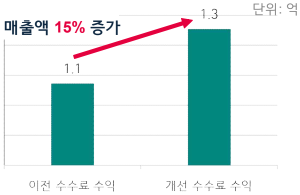
[지그재그 판매 수수료]

파트너별 구매전환율, 재구매율, 고객 혜택, CS, 주문 취소율,
배송, 리뷰 등을 통해 결정되며 3개월마다 갱신되는 구조

▪ 솔큐러시 기반 재산정된 긱스 수수료율

수수료율	5.5	6.5	7.5	8.5	9.5
솔큐러시	90이상	80이상	65이상	50이상	50이하

▪ 크몽 후기 데이터 기반 예상 수수료율 도출



분석② 거래 금액 개선

파생변수를 통한 거래금액 예측 모델

고객 데이터 파생변수 생성

고객 Clustering - KMeans

분석 목표

- 고객의 **사용금액**, **수정 횟수** 등의 변수를 활용하여 고객의 최적의 군집을 찾는다

분석 방법

- **KMeans Clustering**

검증 방법

- **엘보우** 분석, **실루엣 계수** 확인

기타 파생 변수 생성

고객 특성 발굴을 위한 파생변수 생성

변수 설명

- [1회당 구매 금액]
  소비자가 한번 거래에 지불하는 평균 금액
- [유입 대분류]
  분류를 '유튜브, 검색엔진, SNS, 카페/블로그'로
  크게 나누어 주 유입 경로를 확인
- [연령대]
  주 고객층 확인을 위한 연령대 파생변수 생성

적정 거래금액 예측

분류 모델

분석 목표

- 적정 거래 금액 및 추가 결제금액의 기준 無
  소비자들과 판매자들의 불만 야기
- 분류 모델을 통한 **적정 거래금액** 도출

분석 방법

- **RandomForest Classifier**

검증 방법

- **F1-Score**

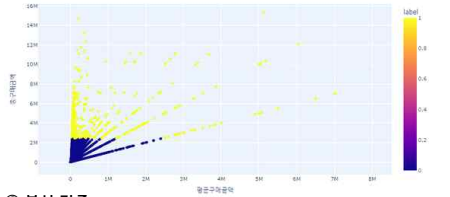
X : 총 구매금액,
1회당 평균 구매금액, 연령대

K-Means Clustering

RandomForest Classifier

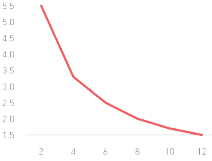
X : 서비스명, 판매자,
대분류, 고객 등급(label)

① 분석 결과



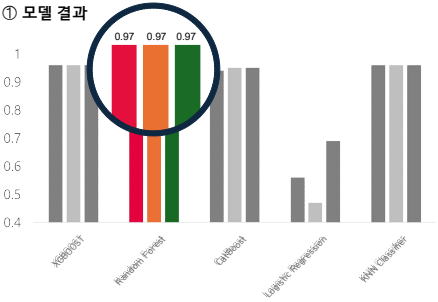
② 분석 검증

②-① Elbow Method



②-② Silhouette Score

① 모델 결과



② 결과 분석

- 이용고객층, 서비스분류 등이 적정거래 가격에 영향을 미침
- 이를 통해

→ 서비스별 적정 가격 제시 기능 제공

I II III IV V VI

플랫폼 개선

개선안② 생성형 AI기반 서비스 기획

: 비정형 데이터를 정형 데이터로 변환하는데
텍스트 처리 기반 생성형 AI 사용

생성형 AI 기반 Fraud Detection

목적

플랫폼 외부에서 개인적으로 연락을 취해 거래가 지속되지 않고 전문가들이 이탈하는 현상 방지
(비즈니스 시나리오 상 문제점)

개선방안 ChatGPT

- 생성형 Open AI API를 적용한 챗봇 활용으로 개인연락 우회 탐지 및 차단

기대효과

- 수수료 수익 위협요소 제거(사업 목적)

+ RandomForest Classifier 모델 기반 견적예상 챗봇

목적

적정가격 추정의 불편함에 대한 VOC 해결

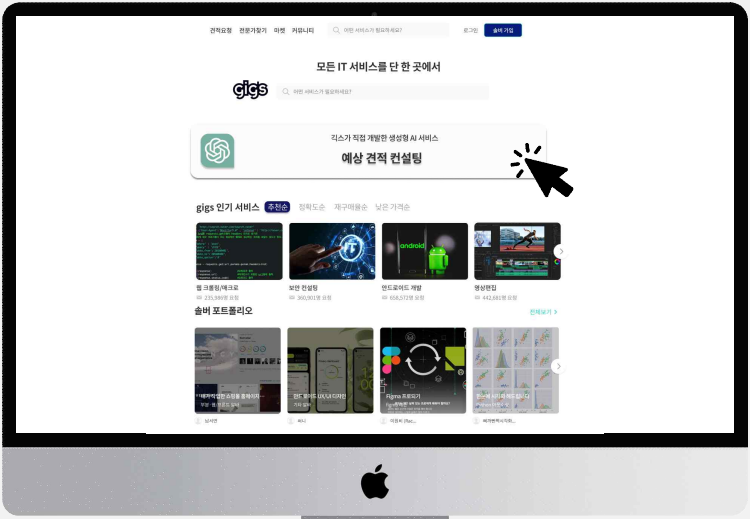
개선방안

- 고객이 원하는 서비스명, 판매자, 대분류, 개인 혹은 기업 여부를 작성하면 생성한 분류 모델에 기반한 최적화된 가격을 제시

기대효과

- 맞춤 견적 예상으로 무분별한 견적서 수신 방지
- 대기시간 단축
  - 프리미엄 서비스를 구독한 판매자의 견적서가 빠르게 제공됨
  - 고객이 원하는 판매자의 견적서 확인 가능

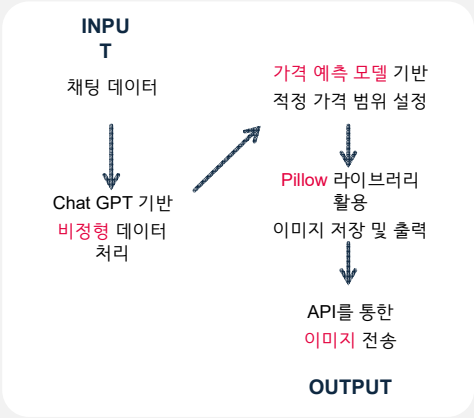

I
II
III
IV
V
VI
개선된 gigs

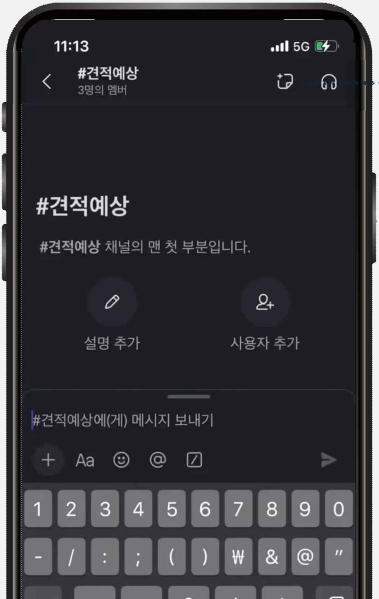

견적요청
전문가찾기
마켓
커뮤니티
로그인
솔버 가입
모든 IT 서비스를 단 한 곳에서
gigs
긱스가 직접 개발한 생성형 AI 서비스
예상 견적 컨설팅
gigs 인기 서비스
추천순
정확도순
재구매율순
낮은 가격순
웹 크롤링/제크로
보안 컨설팅
안드로이드 개발
영상편집
솔버 포트폴리오
전체보기

POINT

Open AI API를 통한 텍스트·이미지 처리 및 견적서 제공

Technology





POINT

Open AI API를 통한 텍스트 이미지 처리 및 견적서 제공

Technology

INPUT

채팅 데이터

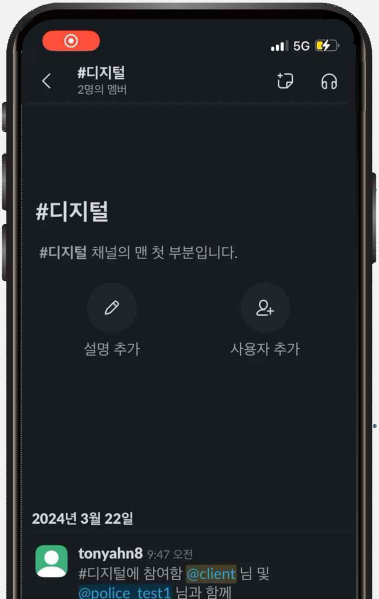
Chat GPT 기반
비정형 데이터
처리

가격 예측 모델 기반
적정 가격 범위 설정

Pillow 라이브러리
활용
이미지 저장 및 출력

API를 통한
이미지 전송

OUTPUT



POINT

[거래사유회 방지]
Open AI API를 통한 연락처 전달 우회 감지 및 차단

차단 Case

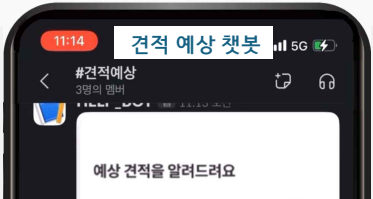
- 295E 삼사5육 으로 깨톡 주세요~
- hanjw일이 3 sa @ 네입어 로 보내주세요~
- jenna 골뱅이 지m네일로 빠르고 저렴하게 진행 가능합니다


이상 탐지


솔큐러시

gigs


태그 기반 리뷰
5.0 | 23.12.23 11:30
매번 신속하고 정확하게 작업해주셔서 감사합니다
#시간 엄수
#자세한 설명
#빠른 연락
#합리적 가격


11:14
견적 예상 챗봇
#견적예상
3명의 멤버
예상 견적을 알려드려요

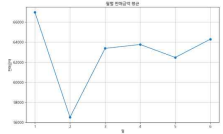
귀무가설: 재구매율과 평균 추가금액은 상관성이 없을 것이다
대립가설: 재구매율과 평균 추가금액은 상관성을 띌 것이다.

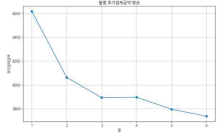
스피어만 상관분석 결과:

Statistic	P-value
-0.95	0.004

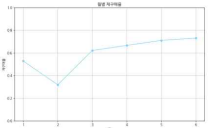
재구매율과 평균 추가금액은 역의 상관성을 가지고 있음.

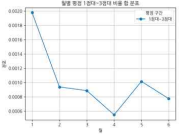
1 1월의 과다 추가금액으로 2월 판매금액이 크게 떨어진 것을 확인





2 추가 결제금액과 역의 상관을 가지는 재방문율, 평점



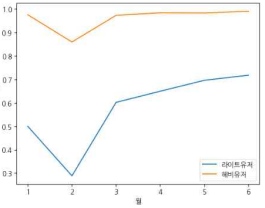


- 1월과 2월은 재방문율이 최저, 평균 추가결제 금액이 최대
- 추가결제금액이 고객의 재방문에 중요한 영향을 미치고 있음

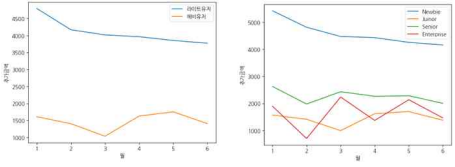
=>따라서 고객의 재방문율을 높이기 위해 추가결제금액이 감소될 수 있는 서비스

AS-IS

2월 신규가입자 저조



라이트 유저 평균 추가 결제금액 과다



TO-BE

1 2월 간단 부업 마케팅

black Gigs = Gigs의 2월달 프로모션

- 2월 Gigs에 처음 가입하고 이용한 고객에게 2회 할인

클래스101+ 구독자 140% 증가, 해외 신규 구독 3배 ↑

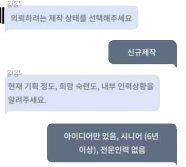
온라인 클래스 플랫폼 클래스101이 블랙프라이데이 글로벌 프로모션을 통해 구독 서비스 클래스101+ 관련 역대급 기록을 달성했다.

클래스101은 지난 24일부터 열주일 간 블랙프라이데이 기념 스페셜 프로모션을 진행. 새롭게 선보인 구독 서비스 클래스101+의 생애 최초 구독 시작 시 서비스 이용 첫달을 1,000원에 만나볼 수 있는 파격 프로모션으로 해당 기간 동안 국내 구독자가 전주 대비 140% 증가하며 구독자 확보를 본격적으로 가속화했다.

* 프로모션을 통한 매출 증대 사례

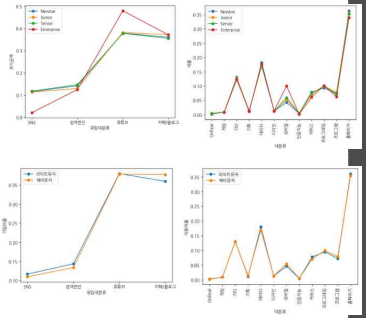
2 챗봇을 활용한 세부 가이드 제

고객의 보다 간편한 의뢰
문의를 위해 세부 가이드를
제공하여
추가 결제금액의 소요 조정



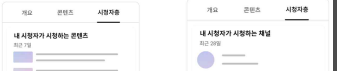
* 긱스 챗봇 예시

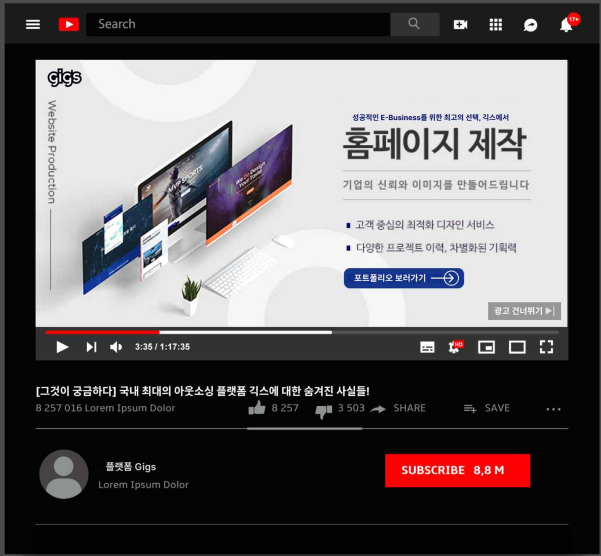
유튜브 유입, 홈페이지 의뢰 고객이 많은 긱스



유튜브 광고로 유입 극대화

광고를 통해 유입되는 고객 데이터 확인





만족도 향상을 통한 재구매율 85% 달성

프로젝트에 있어서 가장 중요한 KPI수립에 많은 고뇌를
했는데 팀원들 덕분에 해결할 수 있었습니다.

어려움에 부딪힐 때마다 팀을 구원해주신
"곽경일 강사님" 정말 감사드립니다.

이우재 김수정 한재원 전예진 송재은 안성재

다양한 모델링과 개발을 직접 해보면서
실력이 많이 성장했던 시간이었습니다.

프로젝트 내내 팀원들이 잘 따라와주고
열심히 임해줘서 즐거운 시간이었습니다.

QnA

벤치마킹 서비스

- 수수료는 어떻게 책정되나요?
  - 동대문 패션

    - PG 및 플랫폼 이용 수수료 4.0% + 판매 수수료 1.5% 부터~ (부가세 별도/개별적용)

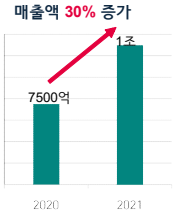
    - 판매 수수료는 종합 지표(파트너 별 구매전환율, 재구매율, 고객 혜택, 고객 클레임(CS), 주문 취소율, 배송, 리뷰 등)를 통하여 결정되며 3개월마다 갱신됩니다.

▪ 도입 목적

고객 구매 경험 개선을 위해 노력하는 파트너사에게 혜택을 제공해
고객에게 더욱 좋은 경험을 선사하고, 선순환 구조를 만들기 위함

▪ 지그재그의 수수료 정책 변화 후 성과

	~2020	2021~
수수료	• Z결제로 입점 시 5.5% • Z결제와 미연동 시 0%	• 파트너별 구매 전환율, 재구매율, 고객혜택, CS, 리뷰 등 종합지표 기반으로 변동 • 3개월마다 재책정



* Appendix | 감성분석 모델 설계

```
import tensorflow as tf
import numpy as np
import pandas as pd
#from transformers import *
import json
import numpy as np
import pandas as pd
from tqdm import tqdm
import os
import sentencepiece as spm
```

```
import os
from google.colab import drive
drive.mount('/content/gdrive/')
```

```
Mounted at /content/gdrive/
```

```
!git clone https://github.com/e9t/nsmc.git
```

```
Cloning into 'nsmc'...
remote: Enumerating objects: 14763, done.
remote: Counting objects: 100% (14762/14762), done.
remote: Compressing objects: 100% (13012/13012), done.
remote: Total 14763 (delta 1748), reused 14762 (delta 1748)
Receiving objects: 100% (14763/14763), 56.19 MiB | 18.12 Mi
Resolving deltas: 100% (1748/1748), done.
Updating files: 100% (14737/14737), done.
```

```
os.listdir('nsmc')
```

```
['ratings.txt',
 'README.md',
 '.git',
 'ratings_test.txt',
 'raw',
 'code',
 'synopses.json',
 'ratings_train.txt']
```

```
train = pd.read_table('nsmc/'+'ratings_train.txt')
test = pd.read_table('nsmc/'+'ratings_test.txt')
```

```
train
```

	id	
0	9976970	아 더빙.. 진
1	3819312	흠...포스터보고 초딩영화줄...오버연
2	10265843	너무재밌었다그
3	9045019	교도소 이야기구먼 ..솔직히
4	6483659	사이몬페그의 익살스런 연기가 돋보였던 영화!스파이더맨에서 늙어
...	...	
149995	6222902	인간이 문
149996	8549745	
149997	9311800	이게 뭐요? 한국인은 거들먹거리고
149998	2378369	청춘 영화의 최고봉 방황과 우
149999	9619869	한국 영화 최초로 수간

150000 rows × 3 columns

```
import logging
import os
import unicodedata
from shutil import copyfile

from transformers import PreTrainedTokenizer


logger = logging.getLogger(__name__)

VOCAB_FILES_NAMES = {"vocab_file": "tokenizer_78b3253a26.model",
                     "vocab_txt": "vocab.txt"}
```

```
def convert_data(data_df):
    global tokenizer

    SEQ_LEN = 64 #SEQ_LEN : 버트에 들어갈 인풋의 길이

    tokens, masks, segments, targets = [], [], [], []

    for i in tqdm(range(len(data_df))):
        # token : 문장을 토큰화함
        token = tokenizer.encode(data_df[DATA_COLUMN].iloc[i], truncation=True, padding='max_length', max_le

        # 마스크는 토큰화한 문장에서 패딩이 아닌 부분은 1, 패딩인 부분은 0으로 통일
        num_zeros = token.count(0)
        mask = [1]*(SEQ_LEN-num_zeros) + [0]*num_zeros

        # 문장의 전후관계를 구분해주는 세그먼트는 문장이 1개밖에 없으므로 모두 0
        segment = [0]*SEQ_LEN

        # 버트 인풋으로 들어가는 token, mask, segment를 tokens, segments에 각각 저장
        tokens.append(token)
        masks.append(mask)
        segments.append(segment)

        # 정답(긍정 : 1 부정 0)를 targets 변수에 저장해 줌
        targets.append(data_df[LABEL_COLUMN].iloc[i])

    # tokens, masks, segments, 정답 변수 targets를 numpy array로 저장
    tokens = np.array(tokens)
    masks = np.array(masks)
    segments = np.array(segments)
    targets = np.array(targets)

    return [tokens, masks, segments], targets

# 위에 정의한 convert_data 함수를 불러오는 함수를 정의
def load_data(pandas_dataframe):
    data_df = pandas_dataframe
    data_df[DATA_COLUMN] = data_df[DATA_COLUMN].astype(str)
    data_df[LABEL_COLUMN] = data_df[LABEL_COLUMN].astype(int)
    data_x, data_y = convert_data(data_df)
    return data_x, data_y

SEQ_LEN = 64
BATCH_SIZE = 32
# 감성분석 문장을 포함하고 있는 컬럼
DATA_COLUMN = "document"
# 긍정인지 부정인지를 (1=긍정,0=부정) 포함하고 있는 컬럼
LABEL_COLUMN = "label"
```

* Appendix | 감성분석 모델 설계

```
# train 데이터를 버트 인풋에 맞게 변환
train_x, train_y = load_data(train)
```

```
100%|██████████| 150000/150000 [01:34<00:00, 1587.66it/s]
```

```
test_x, test_y = load_data(test)
```

```
100%|██████████| 50000/50000 [00:24<00:00, 2073.80it/s]
```

```
from transformers import TFBertModel

model = TFBertModel.from_pretrained("monologg/kobert", from_pt=True)
# 토큰 인풋, 마스크 인풋, 세그먼트 인풋 정의
token_inputs = tf.keras.layers.Input((SEQ_LEN,), dtype=tf.int32, name='input_word_ids')
mask_inputs = tf.keras.layers.Input((SEQ_LEN,), dtype=tf.int32, name='input_masks')
segment_inputs = tf.keras.layers.Input((SEQ_LEN,), dtype=tf.int32, name='input_segment')
# 인풋이 [토큰, 마스크, 세그먼트]인 모델 정의
bert_outputs = model([token_inputs, mask_inputs, segment_inputs])
```

```
Downloading: 100% ██████████ 352M/352M [00:07<00:00, 49.8MB/s]
All PyTorch model weights were used when initializing TFBertModel.

All the weights of TFBertModel were initialized from the PyTorch model.
If your task is similar to the task the model of the checkpoint was trained on, you can already use TFBertM
```

```
# Rectified Adam 옵티마이저 사용
!pip install tensorflow_addons
import tensorflow_addons as tfa
# 총 batch size = 4 epoch = 2344 = 4
opt = tfa.optimizers.RectifiedAdam(lr=5.0e-5, total_steps = 2344*2, warmup_proportion=0.1, min_lr=1e-5, epsilon=1e-08, clipnorm=1.0)
```

```
Collecting tensorflow_addons
  Downloading tensorflow_addons-0.23.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (611 kB)
                                              611.8/611.8 kB 4.2 MB/s eta 0:00:00
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow_addons) (24.0)
Collecting typeguard<3.0.0,>=2.7 (from tensorflow_addons)
  Downloading typeguard-2.13.3-py3-none-any.whl (17 kB)
Installing collected packages: typeguard, tensorflow_addons
Successfully installed tensorflow_addons-0.23.0 typeguard-2.13.3
/usr/local/lib/python3.10/dist-packages/tensorflow_addons/utils/tfa_eol_msg.py:23: UserWarning:

TensorFlow Addons (TFA) has ended development and introduction of new features.
TFA has entered a minimal maintenance and release mode until a planned end of life in May 2024.
Please modify downstream libraries to take dependencies from other repositories in our TensorFlow community (e.g. Keras, Keras-CV, and K

For more information see: https://github.com/tensorflow/addons/issues/2807

  warnings.warn(
/usr/local/lib/python3.10/dist-packages/tensorflow_addons/optimizers/rectified_adam.py:121: UserWarning: The `lr` argument is deprecated
  super().__init__(name, **kwargs)
```

```
sentiment_drop = tf.keras.layers.Dropout(0.5)(bert_outputs)
sentiment_first = tf.keras.layers.Dense(1, activation='sigmoid', kernel_initializer=tf.keras.initializers.TruncatedNormal(stddev=0.02))(
sentiment_model = tf.keras.Model([token_inputs, mask_inputs, segment_inputs], sentiment_first)
sentiment_model.compile(optimizer=opt, loss=tf.keras.losses.BinaryCrossentropy(), metrics = ['accuracy'])
```

```
sentiment_model.fit(train_x, train_y, epoche=2, shuffle=True, batch_size=64, validation_data=(test_x, test_y))
```

```
Epoch 1/2
2344/2344 [==============================] - 2628s 1s/step - loss: 0.3698 - accuracy: 0.8210 - val_loss: 0.2755 - val_accuracy: 0.8842
Epoch 2/2
2344/2344 [==============================] - 2565s 1s/step - loss: 0.2309 - accuracy: 0.9072 - val_loss: 0.2564 - val_accuracy: 0.8958
<keras.src.callbacks.History at 0x7e4bfb891c30>
```

```
def predict_convert_data(data_df):
    global tokenizer
    tokens, masks, segments = [], [], []

    for i in trange(range(len(data_df))):

        token = tokenizer.encode(data_df[DATA_COLUMN].iloc[i], max_length=SEQ_LEN, truncation=True, padding='max_length')
        num_zeros = token.count(0)
        mask = [1]*(SEQ_LEN-num_zeros) + [0]*num_zeros
        segment = [0]*SEQ_LEN

        tokens.append(token)
        segments.append(segment)
        masks.append(mask)

    tokens = np.array(tokens)
    masks = np.array(masks)
    segments = np.array(segments)
    return [tokens, masks, segments]

# 위에 정의한 convert_data 함수를 불러오는 함수를 정의
def predict_load_data(pandas_dataframe):
    data_df = pandas_dataframe
    data_df[DATA_COLUMN] = data_df[DATA_COLUMN].astype(str)
    data_x = predict_convert_data(data_df)
    return data_x
```

```
SEQ_LEN = 64
BATCH_SIZE = 32
# 긍부정 문장을 포함하고 있는 칼럼
DATA_COLUMN = "리뷰"

predict = predict_load_data(kmong)
preds = sentiment_model.predict(predict)
kmong['긍정확률'] = preds

kmong.to_csv('Kmong_expert_sentiment.txt', sep = '₩t')
```

```
100%|██████████| 87007/87007 [00:43<00:00, 1989.83it/s]
1596/2719 [================>.............] - ETA: 4:43
```

API 설정

```
messages = []
channel_police = "C06RFUVA62U"
channel_cost = 'C06R415EA2E'
content = """ 지금부터 채팅 내용에 ''견적'과 '판매자,서비스 명,대분류,기업/개인을 입력받으면, ['견적계산','판매자', '서비스 명', '대분류', 'label']을 입력 : 「견적 판매자:KD), 서비스명: GPT 강의해드립니다., 대분류:홈페이지, 기업/개인:개인
예시: " KD), GPT 강의해드립니다., 홈페이지, 개인" """
messages.append({"role": "user", "content": content})

completion = openai.ChatCompletion.create(
    model="gpt-3.5-turbo",
    messages=messages
)

bot_token = 'xoxb-6836556715910-6843166868903-M3lzx4b82xZhodYAlNB0MX12'
bot_user_token = 'xoxp-6836556715910-6843082869731-6866062997472-d67101a4c5cc5c8f0434bc5fac7d1d29'
token = bot_token
channel = "C06R415EA2E"
target_channel = 'C06R415EA2E'
url = "https://slack.com/api/conversations.history"
headers = {"Authorization": "Bearer " + token}
params = {"channel": channel}
```

저장모델 불러오기

```
import pickle
import pandas as pd
import joblib
f = open("encoder.dat", 'rb')
label_encoder_dict = pickle.load(f)
f.close()

# 모델 불러오기
random_forest_model = joblib.load('random_forest_model.pkl')
```

가격 예측

```
input_df = pd.DataFrame([split_list2], columns=['판매자', '서비스명', '대분류', 'label'])
input_df_r=input_df.copy()
for i in input_df.columns:

    encoder = label_encoder_dict[i]
    labels = encoder.transform(input_df[i])

    input_df[i] = labels
print(input_df)
prediction = random_forest_model.predict(input_df)

# 결과 출력
print("Predicted cluster label:", prediction[0])
df_cluster_minmax = pd.read_csv('cluster_minmax.csv')

# 'cluster_label_transformed2' 열에서 유사값과 일치하는 행 선택
matched_rows = df_cluster_minmax[df_cluster_minmax['cluster_label_transformed2'] ==prediction[0]]
```

이미지 생성

```
# 예측된 클러스터 레이블을 다시 원래의 문자열로 디코딩
# 견적 문서 이미지 열기
image = Image.open('숨고견적양식_jpg.jpg')
display(image)

# 출력 이미지에 텍스트 추가하기
draw = ImageDraw.Draw(image)
font_path = 'BMDOHYEON_ttf.ttf'  # 폰트 파일 경로
font_size = 20

# 월급 추가금액 텍스트 추가
seller_text = f'{seller_t[0]} 전문가의'
draw.text(xy=(50, 170), text=seller_text, fill='black', font=ImageFont.truetype(font=font_path, size=font_size))

# 월급 추가금액 텍스트 추가
avg_text = f'평균 추가금액: {avg_t}'
draw.text(xy=(50, 210), text=avg_text, fill='black', font=ImageFont.truetype(font=font_path, size=font_size))

# 최대금액 텍스트 추가
max_text = f'최대금액: {max_t}'
draw.text(xy=(50, 300), text=max_text, fill='black', font=ImageFont.truetype(font=font_path, size=font_size))

# 최소금액 텍스트 추가
min_text = f'최소금액: {min_t}'
draw.text(xy=(50, 360), text=min_text, fill='black', font=ImageFont.truetype(font=font_path, size=font_size))

# 이미지 저장
image.save('숨고견적결과2.jpg')

# 결과 이미지 보여주기
result_img = Image.open('숨고견적결과2.jpg')
```

견적 결과 전송

```
#filepath = '숨고견적결과.jpg'
# 견적결과 이미지 전송
filepath = '숨고견적결과2.jpg'
try:
    response = client.files_upload(file=filepath, channels=target_channel, title="price screenshot")
    file_id = response.data['file']['id']
    client.chat_postMessage(channel=target_channel, text="", files=[file_id])
except Exception as e:
    print(f"Error uploading file: {e}")
```