얼굴 인식 기반
엘리베이터 TV 타겟팅 광고

Focus on Me [도시영, 이성렬, 장희연, 조현재, 최예슬]

목차

1. 프로젝트 배경 및 목적

2. 팀 구성 및 역할

3. 시스템 설계 및 구축

4. 모델링 설계 및 구축

5. 데이터베이스 활용 및 시각화

6. 수행 결과 시연

7. Self Evaluation

프로젝트 배경

1. 옥외 광고란?

- 가정 밖에서 '이동 중에' 경험하는 모든 광고

2. 옥외 광고의 성장과 한계

2018년 옥외광고 시장은 1조342억 원으로 소폭 성장한 가운데 디지털 옥외광고는 옥외광고 자유표시구역의 화제성 및 프로그래매틱 광고 등 디지털 기술의 발전에 따라 지속적으로 성장할 것으로 전망된다.

그러나 이 같은 성장 전망에도 불구하고 효과 측정의 어려움으로 실제 광고 집행은 정체되어 있는 실정이다. 이를 해결하기 위해 옥외광고 및 디지털 옥외광고 효과 측정에 관한 많은 연구가 있지만 아직은 개념적인 측면에 치우쳐 있고 영향 변수를 측정하기 위한 별도의 조사가 필요한 한계점을 가지고 있다.

2019.10
디지털 옥외광고 매체 효과 측정 모델 개발에 관한 연구: 옥상형과 벽면형을 중심으로
Kim, Shin Youp; Shim , Sung Wook

프로젝트 배경

3. 광고 매체 별 기획의 특징

	TV	디지털	옥외
매체 기획 방법 및 광고 효율성 지표	과거 노출데이터를 기반으로 산출된 적정 예산 및 목표 Reach, Frequency 제안	적정 예산, 목표 CPV, CPM, CPC 등 예산에 따른 광고 효율성 지표	적정 예산, 적정 노출 지역
광고 노출 데이터 플랫폼	Nilsen Korea	구글 애드센스 네이버광고	각 매체사 영업직원

프로젝트 배경

3. 광고 매체 별 기획의 특징

	TV	디지털	옥외
매체 기획 방법 및 광고 효율성 지표	과거 노출데이터를 기반으로 산출된 적정 예산 및 목표 Reach, Frequency 제안	적정 예산, 목표 CPV, CPM, CPC 등 예산에 따른 광고 효율성 지표	적정 예산, 적정 노출 지역
광고 노출 데이터 플랫폼	Nilsen Korea	구글 애드센스 네이버광고	각 매체사 영업직원

1. 효율성 지표가 뚜렷하지 않아 항상 경험에 의존한 광고 집행

2. 축적 데이터가 없어 누구에게 얼마나 노출할지 정교하게 목표할 수 없음

프로젝트 목적

문제점

- ✓ 효율성 지표가 뚜렷하지 않아 항상 경험에 의존한 광고 집행
- ✓ 축적 데이터가 없어 누구에게 얼마나 노출할지 정교하게 목표할 수 없음

해결방안

- ✓ 옥외 광고 시청자의 얼굴을 인식하여 타겟팅 광고를 송출하기 위한 연령대 및 성별 예측
- ✓ 세분화된 타겟 별 맞춤형 콘텐츠 송출
- ✓ 안면인식을 통해 남녀, 연령대에 따른 광고 노출 및 DB화
- ✓ 최종적인 광고 효율성 지표 정의 및 추출

서비스 고객

- ✓ 옥외 광고 서비스를 제공하고 있는 매체사　EX) 포커스미디어코리아, KT타운보드

엘리베이터 TV 광고란

참고 사진

한정된 공간, 안면인식 정확도를 고려

옥외광고 매체 중 엘리베이터 TV 광고 선정

전체 시나리오 플로우


광고 집행을 원하는 유저가
웹사이트 접속하여
회원가입 및 로그인
유저가 원하는 타겟, 기간, 예산을
설정하여 광고 등록
엘리베이터에 탑승한 사람의
모델링을 통한 라벨 추출
유저가 목표한
타겟 라벨이 검출 시
해당 광고 노출 및 DB화
누적된 DB를 활용하여
최종 광고 노출 데이터 리포팅
FOM

프로젝트 팀 구성 및 개발 툴

BIG DATA

장희연 최예슬

테스트 데이터 크롤링

Input Image 전처리

중복인물 검출 알고리즘 구축

광고 효율성 지표 정의 및 시각화

AI

이 성 렬

성별 및 나이 구별 모델링

모델링 결과 데이터베이스 연동

광고 송출 모델링

IOT

도 시 영

최종 이미지 선정 알고리즘 구축

센서/디바이스/클라이언트
서버 통신 구축

서버용 데이터 송/수신 구축

광고 송출 기능 구현

CLOUD

조 현 재

서비스 프론트엔드, 백엔드

AWS 서버 관리

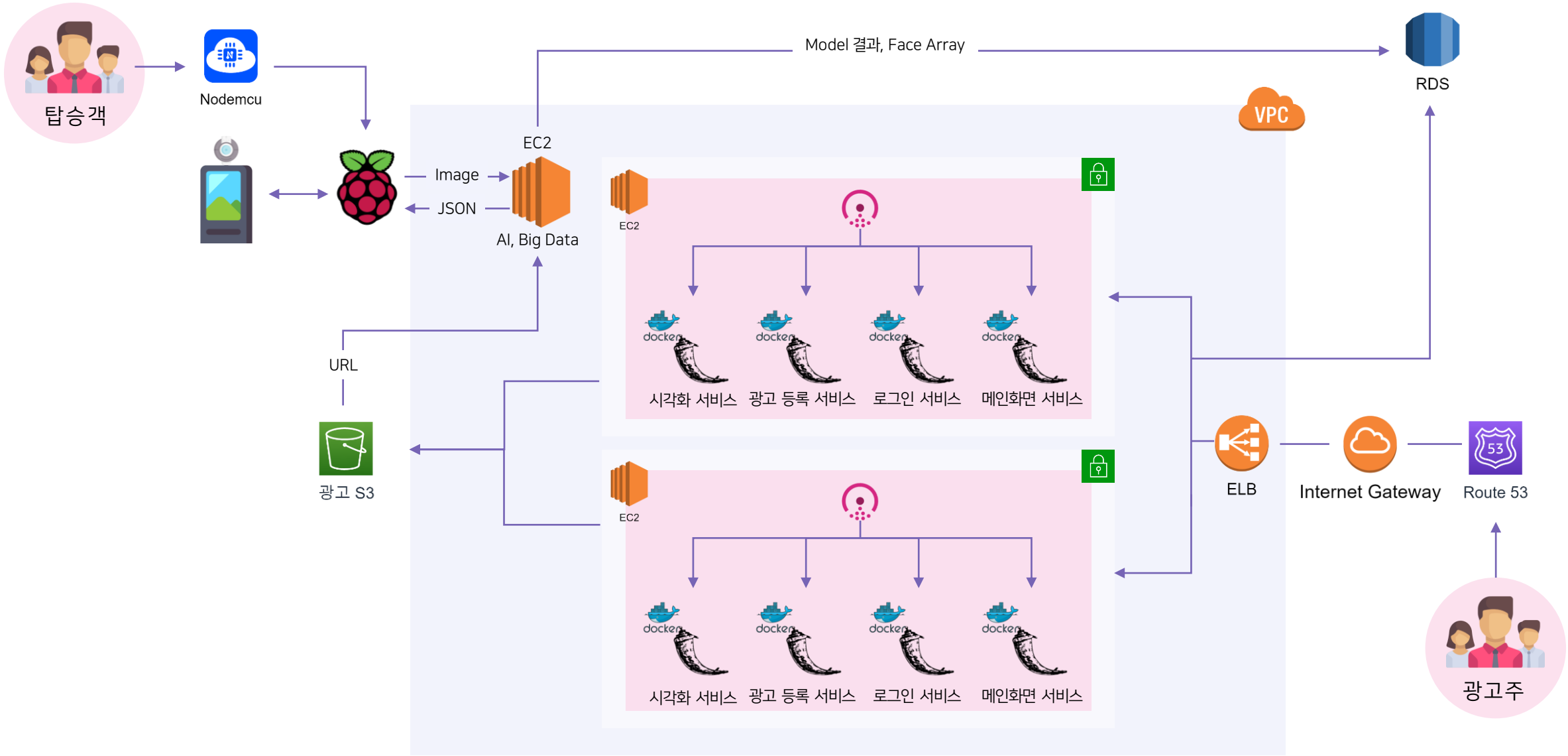
아키텍쳐 및 데이터베이스 설계

프로젝트 프로세스

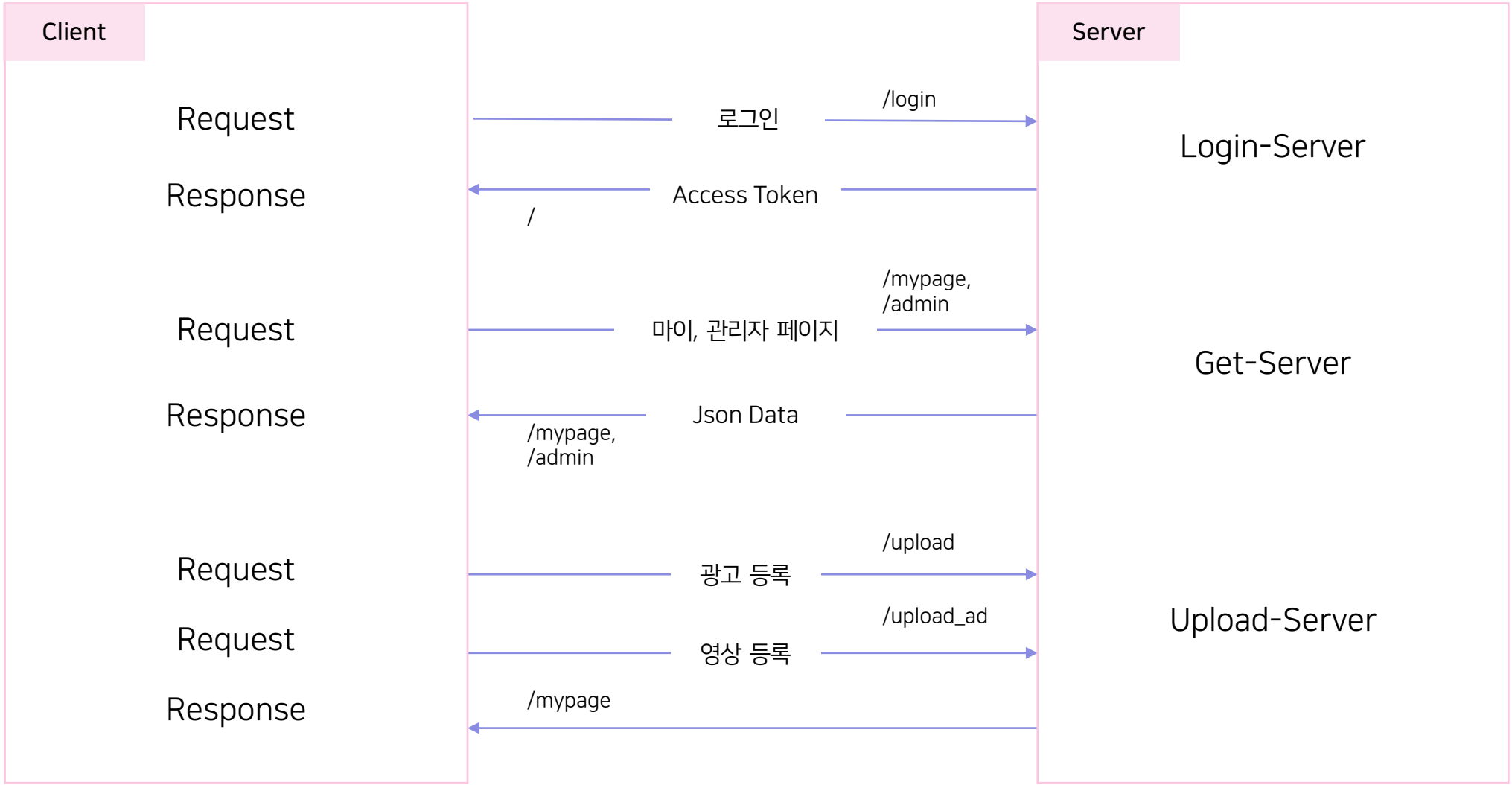
2021. 04						
25	26	27	28	29	30	1
			프로젝트 계획			
2021. 05						
2	3	4	5	6	7	8
프로젝트 설계						
	기본 기능 구현					
9	10	11	12	13	14	15
기본 기능 구현						
16	17	18	19	20	21	22
각 기능별 프로토타입						
					병합 및 디버깅	
23	24	25	26	27	28	29
	최종 구현 및 테스트					
2021. 06						
30	31	1	2	3	4	5
		수행 결과 작성			발표	

시스템 설계 및 구축

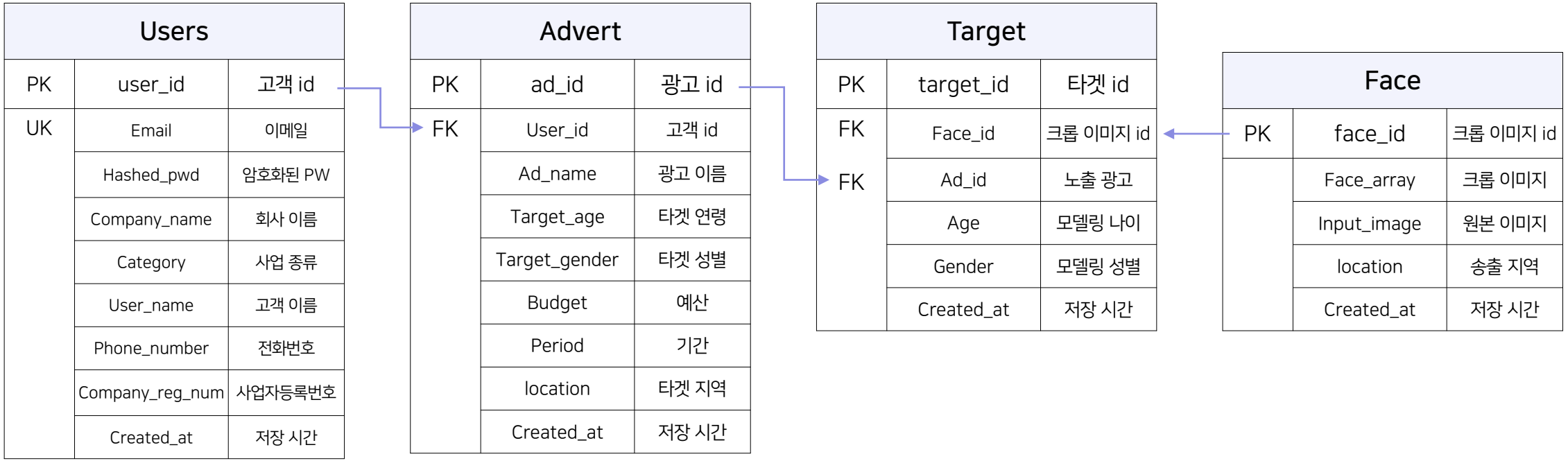
아키텍쳐


탑승객
Nodemcu
Model 결과, Face Array
RDS
VPC
EC2
Image
JSON
AI, Big Data
EC2
시각화 서비스
광고 등록 서비스
로그인 서비스
메인화면 서비스
URL
광고 S3
EC2
시각화 서비스
광고 등록 서비스
로그인 서비스
메인화면 서비스
ELB
Internet Gateway
Route 53
광고주

서버 사이드


Client
Server
Request
로그인
/login
Login-Server
Response
Access Token
/
Request
마이, 관리자 페이지
/mypage,
/admin
Get-Server
Response
Json Data
/mypage,
/admin
Request
광고 등록
/upload
Request
영상 등록
/upload_ad
Upload-Server
Response
/mypage

데이터베이스



회원가입 기준 User DB 축적, User가 광고를 등록할 시 Advert DB 축적
광고 노출 시 Advert의 ad_id 키로 연결된 Target DB 축적, Target 별 Face Array DB 축적

JWT를 이용한 로그인 서비스 문제 해결

Issue_1

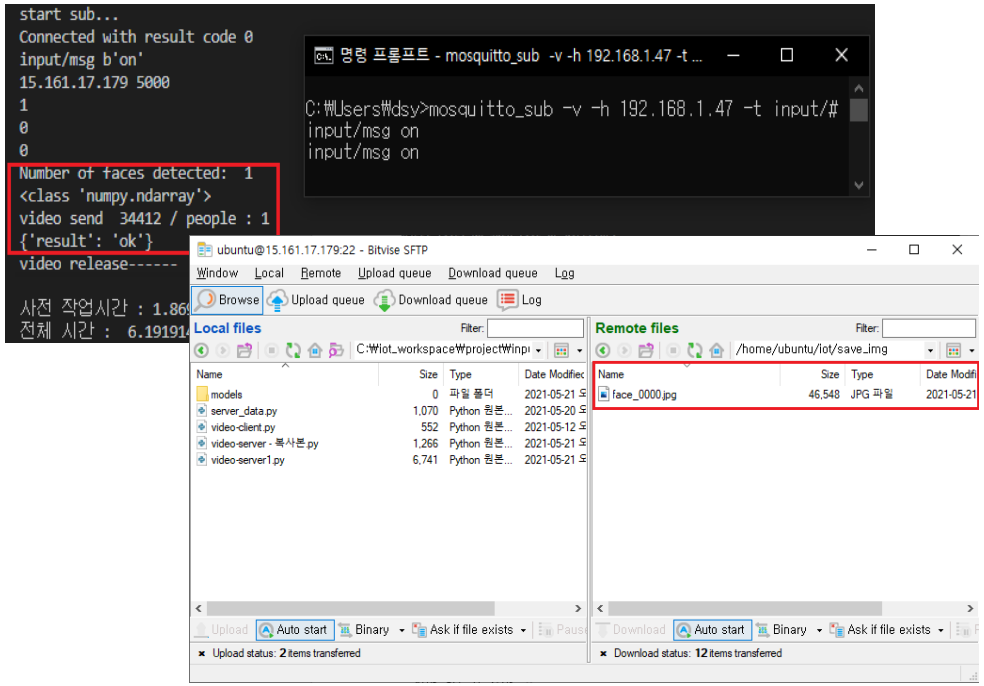
```
response = make_response(render_template('accounts/login.html',
                                         access_token=access_token,
                                         refresh_token=refresh_token,
                                         check=200))
response.set_cookie("access_token_cookie", access_token)

return response
```

```
access_token = request.cookies.get("access_token_cookie")

headers = {"Authorization": "Bearer " + (access_token if access_token else "")}

serviceName = "get-server"
# Consul address, port number
service_address = client.catalog.service(serviceName)[1][0]['ServiceAddress']
service_port = client.catalog.service(serviceName)[1][0]['ServicePort']

url = "http://{}:{}".format(service_address, service_port)

get = requests.post(url + '/mypage', headers=headers)
```

다른 서버의 **Requests header에 Authorization을 보내지 못해**
JWT의 기능 중 권한, 접속제한, 접속한 사람의 ID 추출 등의 기능들을 사용하지 못함
메인 서버에서 **Access Token을 Cookie에 저장하여 다른 URL에서 값을 가져와**
다른 서버의 Requests header에 값을 보내 해결

인식 이미지 선별 및 광고 송출 시스템 문제 해결

Issue_1



카메라를 통해 얼굴 인식 후 검출된 얼굴을
Cropping하여 서버에 전송하려 했으나
Raspberry Pi의 과부하로 인하여
AWS EC2에서 처리하는 것으로 변경

Issue_2

효율적인 기기 및 데이터 관리를 위해
기기의 위치 송출 기능 요청
각 기기 번호와 설치 위치를 AWS EC2에 전송

모델링 설계 및 구축

Input image 전처리

MTCNN

Haar cascade

Open CV 선정 기준 첫 번째, **<Detection Accuracy>**
얼굴 검출율과 **측면 얼굴 검출 정확도**를 고려하여 MTCNN 선정

Input image 전처리

Ver.1

```python
def get_cropimg(filename) :
    detector = MTCNN()
    result_list= detector.detect_faces(cv2.imread(filename))
    res = []
    imgNum = 0
    data = plt.imread(filename)
    for i in range(len(result_list)) :
        result_list2=[]
        for v in result_list[i]['box'] :
            if v >=0:
                result_list2.append(v)
            else :
                result_list2.append(0)
        x1, y1, width, height = result_list2
        x2, y2 = x1 + width, y1 + height
        cropped = cv2.resize(data[y1:y2, x1:x2], (128, 128))
        res.append(cropped)
    return res
```

1.34 Sec

Ver.2

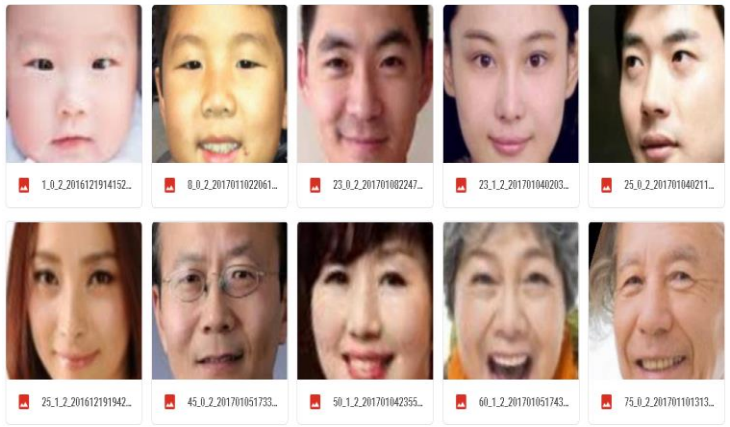
```python
def get_cropimg2(filename) :
    detector = MTCNN()
    result_list= detector.detect_faces(cv2.imread(filename))
    res = []
    imgNum = 0
    data = plt.imread(filename)
    for i in range(len(result_list)) :
        x1, y1, width, height =  result_list[i]['box']
        x1, y1 = abs(x1), abs(y1)
        x2, y2 = x1 + width, y1 + height
        cropped = cv2.resize(data[y1:y2, x1:x2], (128, 128))
        res.append(cropped)
    return res
```

1.38 Sec

Open CV 선정 기준 두 번째, **< Inference Time >**
MTCNN 내의 버그 수정 및 시간 단축을 위해 자체 코드 개발

모델링 데이터 및 Class 선정

모델링 데이터



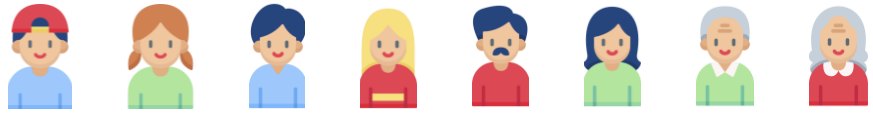
```
DF['age_group'].value_counts()

1.0    11881
2.0     4544
0.0     2896
3.0     2690
Name: age_group, dtype: int64
```

```
ages.value_counts()

3    7789
2    7789
1    7789
0    7789
Name: Ages, dtype: int64
```

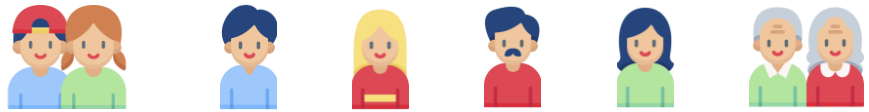
Age와 Gender가 라벨링된 약 24,000장의 UTK Face 데이터
Training(60%) Validation(20%) Test(20%)
Stratify 옵션 사용하여 Class 별 균일하게 분배

Class 선정 개요

8 Class



6 Class



Kids – 2030 남 - 2030 여 – 4050 남 - 4050 여 - Silver

Class가 세분화 될수록 Class 간 경계에서 오분류율이
상승하여 모델 Accuracy에 악영향
마케팅 관점에서 세대별 성향이 뚜렷한 X, Y, Z세대로 구분

Age Model – 모델 선정 이유 및 개요

Model 1 : VGG16

CPU times: user 11.1 s, sys: 363 ms, total: 11.5 s
Wall time: 20.6 s

→ CPU times 11.1 SEC

VGG16
kids_precision : 0.9 , kids_recall : 0.88
2030_precision : 0.67 , 2030_recall : 0.98
4050_precision : 0.32 , 4050_recall : 0.1
silver_precision : 0.92 , silver_recall : 0.09

Model 2 : MobileNet

CPU times: user 4.11 s, sys: 217 ms, total: 4.33 s
Wall time: 5.87 s

→ CPU times 4.11 SEC

MobileNet
kids_precision : 0.97 , kids_recall : 0.61
2030_precision : 0.77 , 2030_recall : 0.92
4050_precision : 0.48 , 4050_recall : 0.42
silver_precision : 0.71 , silver_recall : 0.5

기타 Model

CNN
kids_precision : 0.38 , kids_recall : 0.01
2030_precision : 0.56 , 2030_recall : 0.94
4050_precision : 0.31 , 4050_recall : 0.02
silver_precision : 0.3 , silver_recall : 0.18

focal Loss
kids_precision : 0.95 , kids_recall : 0.53
2030_precision : 0.62 , 2030_recall : 0.99
4050_precision : 0.33 , 4050_recall : 0.06
silver_precision : 0.72 , silver_recall : 0.14

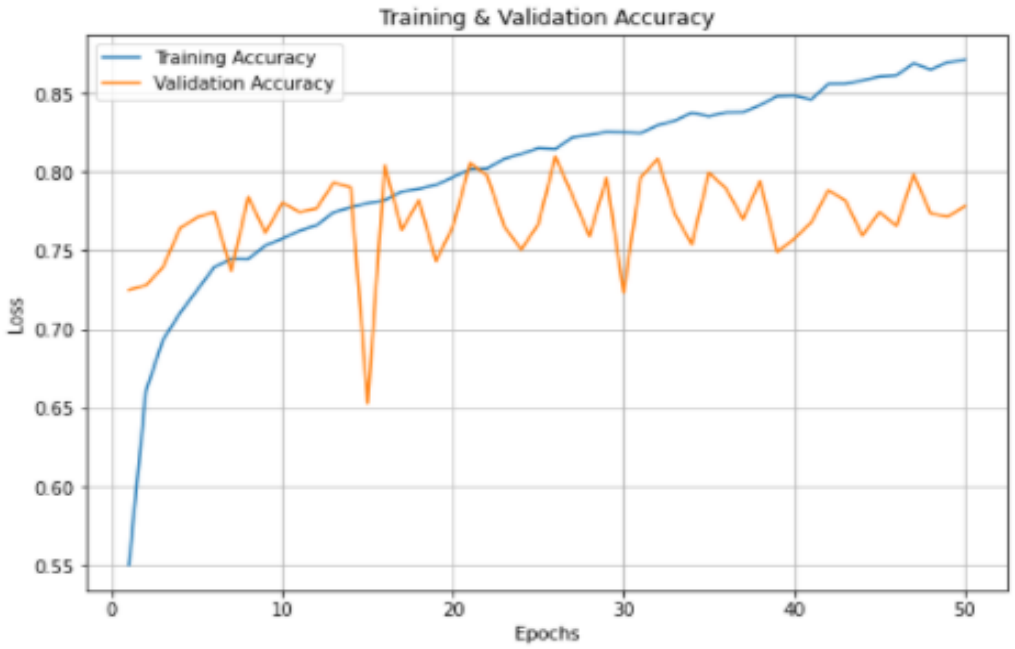
다른 모델들과 비교하여 유독 낮은 Precision이나 Recall을 보이는 Class가 존재하지 않고
이미지 분류 모델에 비해 **경량화** 된(70MB) 모델인 **<MobileNet>** 모델 사용

Age Model – 모델 평가

모델 평가

```
loss, accuracy = model_age_MN.evaluate(test_age,
                                       batch_size = batch_size)

print('Loss = {:.5f}'.format(loss))
print('Accuracy = {:.5f}'.format(accuracy))
```

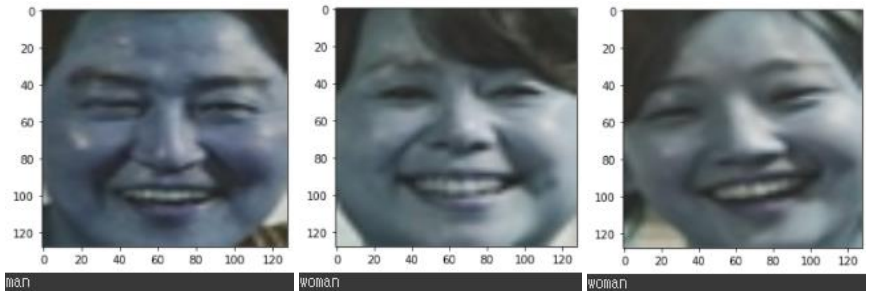
```
69/69 [==============================] - 1s 14ms/step - loss: 0.6549 - accuracy: 0.7844
Loss = 0.65490
Accuracy = 0.78442
```



Test Dataset 기준으로 약 **78%의 Accuracy**

Gender Model – 모델 선정 및 평가

모델 선정 이유 및 개요




```
from keras import backend as K

def focal_loss(total_y_train, total_y_test):
    gamma = 2.0
    alpha = 0.25
    pt_1 = tf.where(tf.equal(total_y_train, 1), total_y_test, tf.ones_like(total_y_test))
    pt_0 = tf.where(tf.equal(total_y_train, 0), total_y_test, tf.zeros_like(total_y_test))
    return -K.sum(alpha * K.pow(1. - pt_1, gamma) * K.log(pt_1))-K.sum((1-alpha) * K.pow( pt_0, gamma) * K.log(1. - pt_0))
```

CNN 모델에 Loss Function으로
Binary Cross Entropy 대신,
분류 어려운 객체에 더 높은 가중치 적용하는
Focal Loss Function 적용

모델 평가

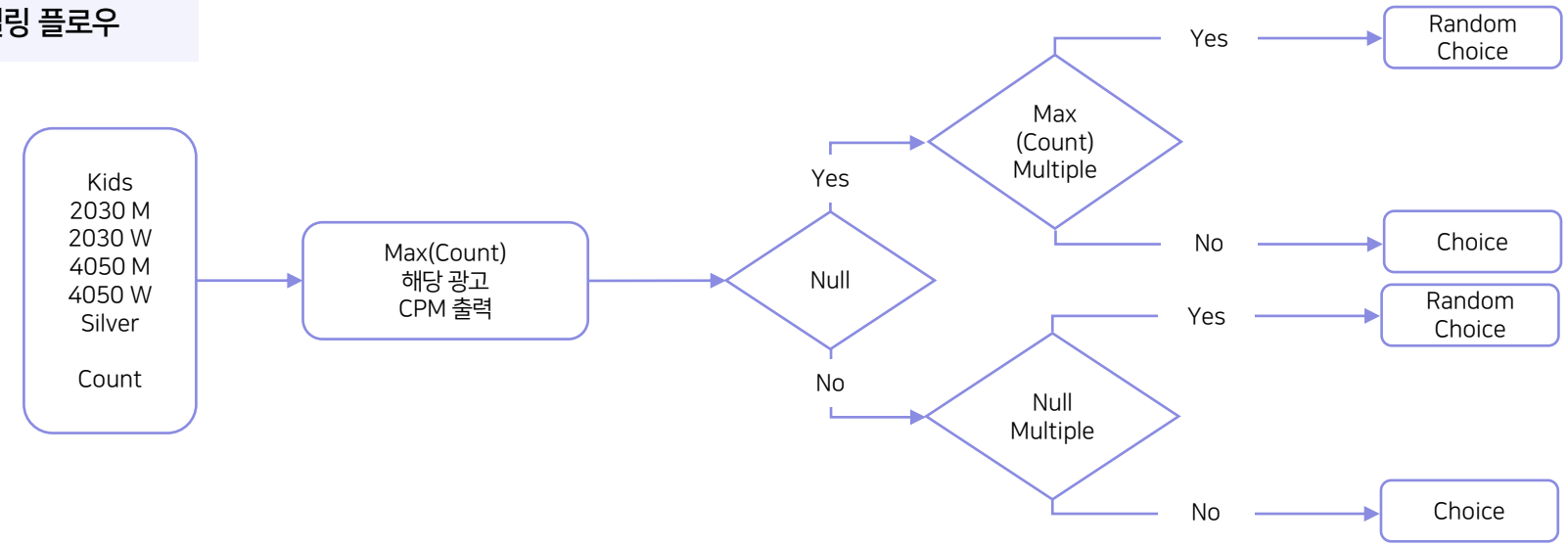
```
loss, accuracy = model_gender_CNN_Loss.evaluate(test_gender,
                                                batch_size = batch_size)

print('Loss = {:.5f}'.format(loss))
print('Accuracy = {:.5f}'.format(accuracy))
```

```
69/69 [==============================] - 1s 10ms/step - loss: 3.1977 - accuracy: 0.8794
Loss = 3.19770
Accuracy = 0.87937
```

Test Dataset을 대상으로 한 Accuracy**는 대부분의 모델이**
약 87% 정도의 높은 Accuracy를 보여서
테스트 환경에서 가장 정확한 모델을 선정

AD_Choice Modeling

모델링 플로우



광고 선택 기준 선정

광고 효율성을 측정하는 대표적인 기준인
CPM을 DB에서 추출하여
송출 광고 선택의 기준으로 사용

```
sql_CPM = '''SELECT ROUND(sub.cost / sub.cnt * 1000, 2) AS CPM
            FROM (SELECT ad.ad_id, ad.budget / ad.period AS cost, COUNT(t.ad_id) AS cnt
                FROM advert ad LEFT JOIN target t
                ON ad.ad_id = t.ad_id
                GROUP BY ad.ad_id, t.ad_id) sub
            WHERE sub.ad_id = %s
            GROUP BY sub.ad_id'''
```

최종 데이터 DB 저장

Iot 출력

```
IoT_ad_url = f'https://yangjae-team08-bucket.s3.eu-south-1.amazonaws.com/{ad_name}.mp4'
IoT_ad_url = IoT_ad_url.replace(' ', '+')
IoT_ad_name = ad_name + '.mp4'
IoT_ad_name = IoT_ad_name.replace(' ', '_')

result = json.dumps({'url': IoT_ad_url, 'ad_file': IoT_ad_name})
net.send(writer, result.encode())
```

Iot 광고 송출이 우선이므로
광고 URL 전송 후 DB 저장 수행

DB Face Table 저장

```
sql_face = '''INSERT INTO face(face_array, input_image, location)
        VALUES(%s, %s, %s)'''
cur.execute(sql_face, [crop, data_date, cam_location])
conn.commit()
```

face_id	face_array	input_image	location	created_at
857	/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGB...	2021_6_1_6_40	영재동-1	2021-06-01 07:06:46
858	/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGB...	2021_6_1_6_40	영재동-1	2021-06-01 07:06:47
859	/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGB...	2021_6_1_6_49	영재동-1	2021-06-01 07:06:58
860	/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGB...	2021_6_1_6_49	영재동-1	2021-06-01 07:06:59
861	/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGB...	2021_6_1_6_49	영재동-1	2021-06-01 07:07:00
862	/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGB...	2021_6_1_6_49	영재동-1	2021-06-01 07:07:00

Crop된 이미지마다
Array를 Binary File로 변환하여,
Input Image에 대한 시각과
IoT 기기 Location과 함께 저장

DB Target Table 저장

```
sql_target = '''INSERT INTO target(face_id, ad_id, age, gender)
        VALUES(%s, %s, %s, %s)'''
cur.execute(sql_target, [face_id, ad_id, age, gender])
conn.commit()
```

target_id	face_id	ad_id	gender	age	created_at
857	857	1	man	2030	2021-06-01 07:06:47
858	858	1	man	2030	2021-06-01 07:06:47
859	859	3	man	2030	2021-06-01 07:06:59
860	860	3	man	4050	2021-06-01 07:06:59
861	861	3	man	2030	2021-06-01 07:07:00
862	862	3	man	4050	2021-06-01 07:07:01

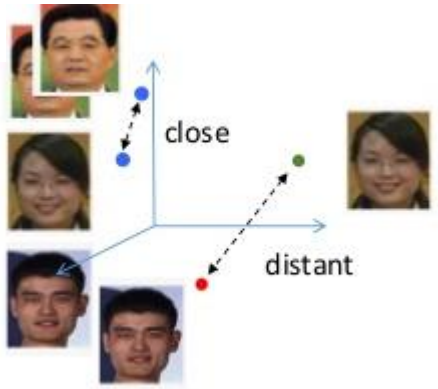
Crop된 이미지마다
모델링 결과인 Age와 Gender 입력
Input Image에 대해
선택된 Ad_id를 함께 저장

데이터베이스 활용 및 시각화

광고 효율성 지표 정의

Embedding 및 Distance 설정

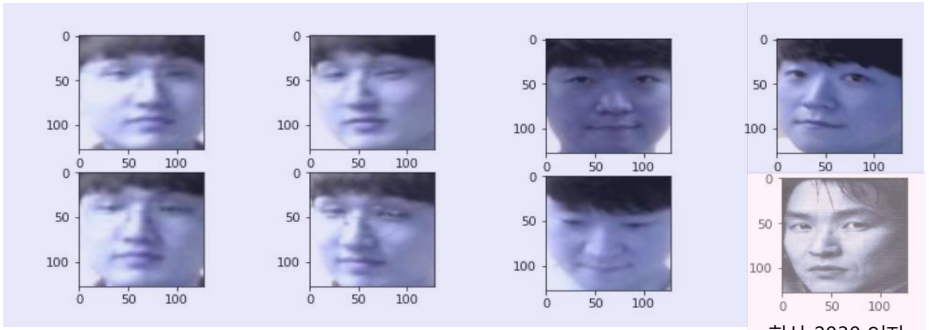
$$f(\text{  }) = \begin{pmatrix} 0.112 \\ 0.067 \\ 0.091 \\ 0.129 \\ 0.002 \\ 0.012 \\ 0.175 \\ \vdots \\ 0.023 \end{pmatrix}$$



Face Recognition Open CV 사용하여 Crop Array별
Embedding 값 도출 및 Distance 산출하여 그룹화

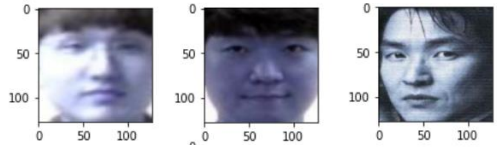
예시

해당 엘리베이터에 탑승하여 DB에 축적된 2030 남자 크롭 어레이

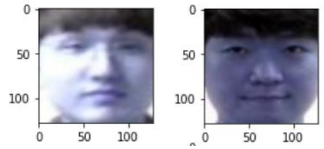


항상 2030 여자와 탑승하여 2030 여성 타깃 광고에만 노출된 사람

중복인물을 제외한 2030 남성 Unique Face Array 추출



그 중 무신사 광고를 시청한 2030 남성



광고 효율성 지표 정의

광고 효율성 지표

Reach	광고 매체에 최소 한번 이상 노출된 타깃 비율 (중복인물 제외)
Frequency	광고 시청 인구가 해당 광고에 노출된 평균 횟수
CPM	1000명 노출 당 비용 (중복인물 포함)

EX) 무신사 광고 (타겟 : 2030 남성)

Reach

(무신사 광고 시청 2030 남) / (전체 2030 남) = 66.7%

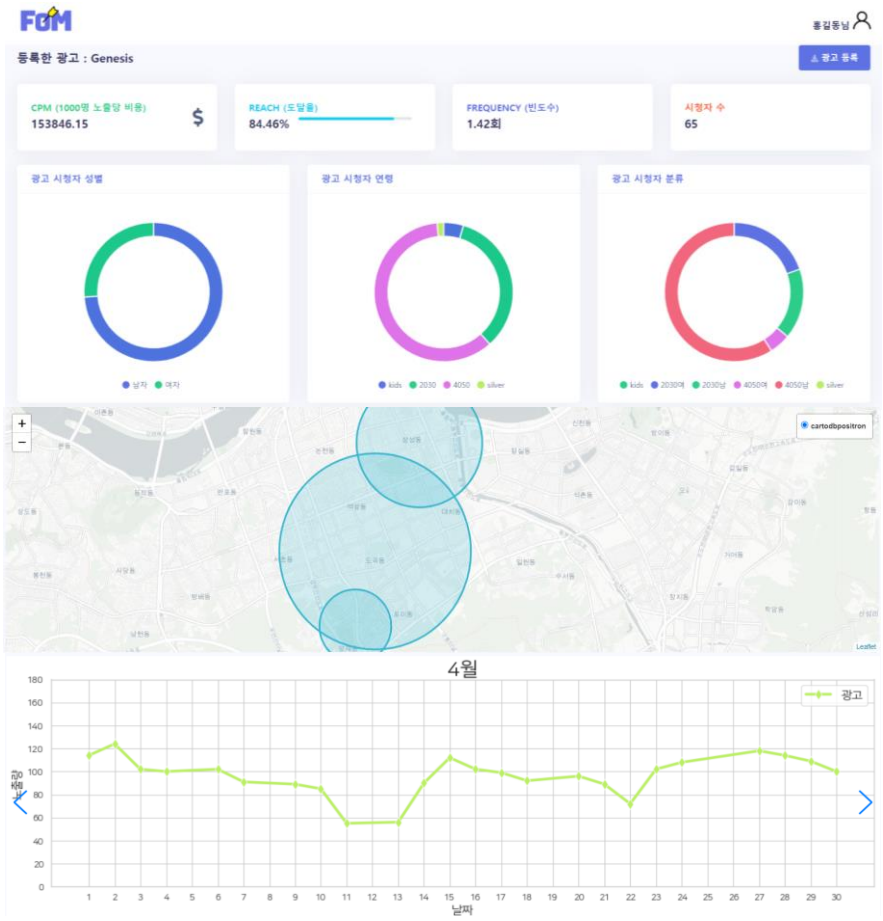
CPM

Budget / (무신사 광고 시청 2030 남) * 1000명

Frequency

(2030 남성에게 노출된 횟수) / (무신사 광고 시청 2030 남) = 3.5회

효율성 시각화



시연 영상

Self Evaluation

프로젝트 아쉬운 점 및 발전 가능성

아쉬운 점

- ✓ 개인정보 문제를 해결하기 위한 데이터 마스킹 기술 혹은 특징값 저장 기술 구현의 필요
- ✓ 서버 및 디바이스 메모리 제한
- ✓ 개인 정보상 동양인 이미지 수집 한계 따른 모델의 정확도 하향
- ✓ 엘리베이터 내 사각지대 이미지 수집 제한

발전 가능성

1. 서비스 측면

- ✓ 안경, 강아지 및 유모차 **등 다양한 객체 인식을 기반**으로 한 타겟팅 광고 발전 가능
- ✓ 개인 정보 문제의 경우 광고 TV 설치 아파트 내의 동의 후 과정 진행
  그 외 **데이터 마스킹 기술 혹은 특징 값만 저장하는 방향**으로 개선 가능
- ✓ 엘리베이터TV 광고 뿐만 아니라 버스나 지하철 외 **다양한 옥외 광고에 적용가능**

2. 기술 측면

- ✓ 동양인 데이터를 추가 확보하여 한국인 대상으로 **모델 정확도 향상 가능**
- ✓ 일정 시간 경과 후 사용자 확인을 거쳐 자동 로그아웃 등 서비스 **Back-end 기능 강화**
- ✓ 플라스크 서버를 람다를 통해 **서버리스로 작동**

팀원 소감 및 총평

전공 과정에서 테스트 데이터로 진행하던 데이터 수집 및 분석이 실무 혹은 서비스에 활용해볼 수 있어서 좋았습니다.

데이터를 분석한 내용을 직접 서비스로 구현하는 전체 플로우를 경험할 수 있어서 좋았습니다. 또한 IT 업계의 다양한 분야에 대한 용어, 기술들에 대한 전반적인 이해를 얻을 수 있었습니다.

디바이스만으로 구현할 수 없는 한계점을 서버 운영과 데이터 분석을 더하여 극복할 수 있었고, 다양한 분야에 이해를 넓히는 계기가 됐습니다.

전공 과정에서는 로컬 환경에서 모델 수립 및 결과 출력만 해보았는데, 융합 프로젝트를 통해 다른 과정들과 연계하여 전체적인 서비스를 구현해 볼 수 있는 기회를 가져서 좋았습니다.

전공 프로젝트에서 클라우드만 작업했을 때 보다 클라우드의 주요 역할인 서비스들을 연결하며 AWS의 여러 기능들을 공부할 수 있었고, 팀원들과 소통하며 원하는 결과물을 만들어서 전체 서비스 과정을 이해했습니다.

자신의 작업 뿐만 아니라 여러 기술들과 협업을 하면서 많은 기술 스택 들을 배울 수 있었고
다른 분야와 협업하는 커뮤니케이션 능력을 키울 수 있었습니다!

THANK YOU!