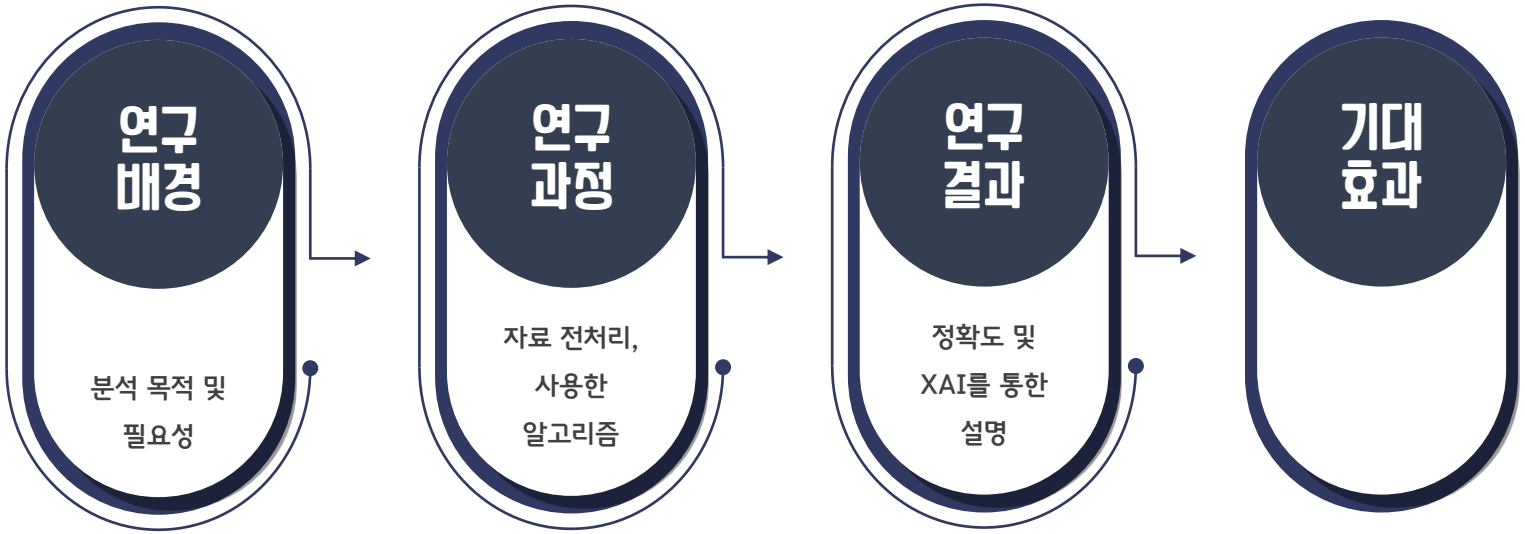
KOPIS 빅데이터 분석 공모전 딥솜(이지현, 김단은, 박초은)

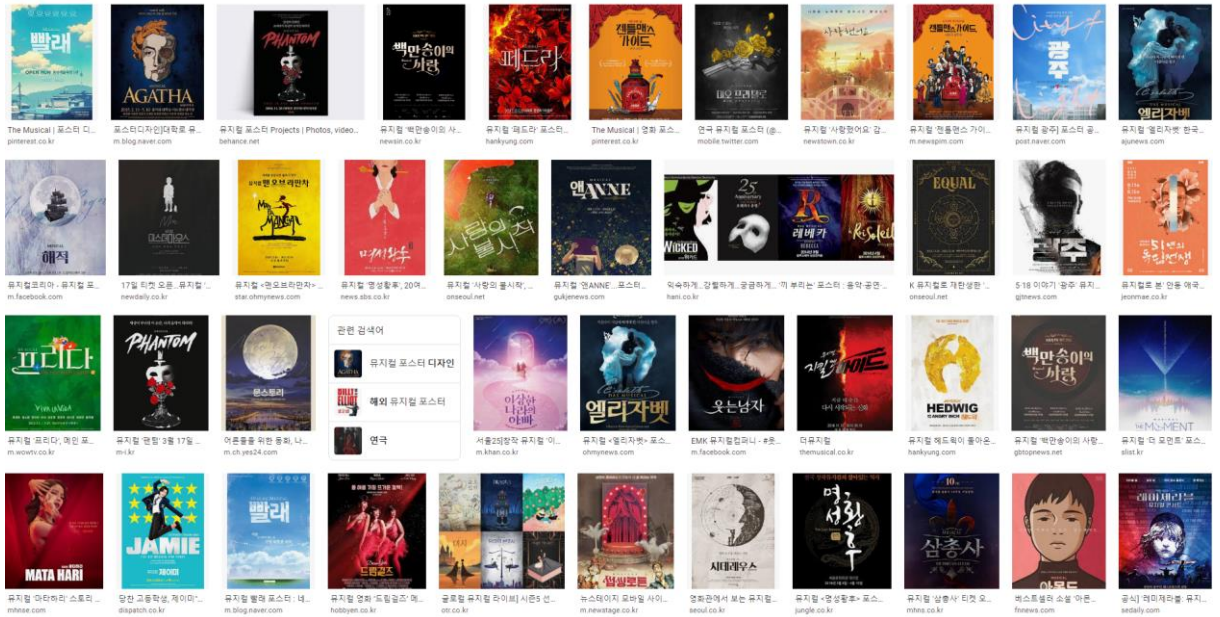
뮤지컬 정형데이터와
이미지 데이터를 활용한
흥행 예측

01

CONTENTS



01 연구 배경



- 뮤지컬은 개막되기 전에는 흥행 여부를 알 수 없어 손실 발생
- 포스터를 이용한 예측을 통해 손실 상황 대비 및 전략 계획 수립 가능
- 좋은 포스터의 기준을 세워 포스터 제작할 때 참고 자료로 활용 가능
- 디자인 비용 절감 효과

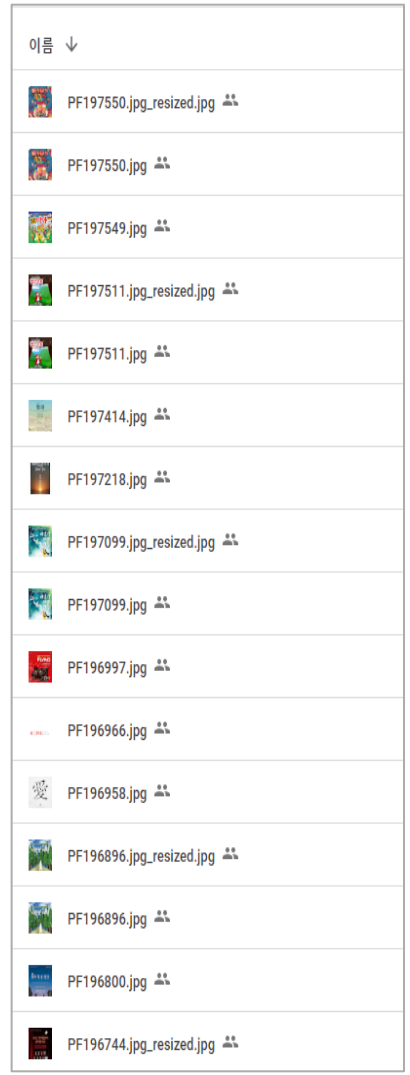
02

연구 과정

- KOPIS OPEN API를 통해 데이터 수집
- 2020.01.20~2022.09.06에 상영한 뮤지컬 크롤링
- OPEN API에서 가져온 정보들 중 공연 아이디, 이름, 포스터 주소만 추출
- 포스터가 없는 뮤지컬 제거
- 포스터가 같은 중복 뮤지컬은 한 개만 남기고 제거(포스터는 같으나 공연 지역이 다른 경우)

필드명	설명	샘플데이터
prfnm	공연명	아동
cate	장르	연극
mt20id	공연ID	PF131558
fcltynm	공연시설명	대학로아트원씨어터
entrpsnm	기획/제작사	(주)티앤비컴퍼니
prfpdfrom	공연시작일	2016.06.07
prfpdto	공연종료일	2016.07.24
prfdtcnt	상연횟수	31

필드명	설명	샘플데이터
mt20id	공연ID	PF132236
mt10id	공연시설ID	FC001431
prfnm	공연명	우리연애할까
prfpdfrom	공연시작일	2016.05.12
prfpdto	공연종료일	2016.07.31
fcltynm	공연시설명(공연장명)	피가로아트홀(구 롯아트홀) (피가로아트홀)
prfcast	공연출연진	김우연, 임정군, 최수영
prfcrew	공연제작진	천창민
prfruntime	공연 런타임	1시간 30분
prfage	공연 관람 연령	만 12세 이상
entrpsnm	제작사	극단 피에로
pcseguidance	티켓가격	전석 30,000원
poster	포스터이미지경로	http://www.kopis.or.kr/upload/pfmPoster/PF_PF132236_160704_142630.gif
sty	줄거리	
genrenm	장르	연극
prfstate	공연상태	공연중
openrun	오픈런	Y
styurls	소개이미지목록	
styurl	소개이미지1	http://www.kopis.or.kr/upload/pfmIntroImage/PF_PF132236_160704_0226303.jpg
styurl	소개이미지2	http://www.kopis.or.kr/upload/pfmIntroImage/PF_PF132236_160704_0226302.jpg
styurl	소개이미지3	http://www.kopis.or.kr/upload/pfmIntroImage/PF_PF132236_160704_0226301.jpg
styurl	소개이미지4	http://www.kopis.or.kr/upload/pfmIntroImage/PF_PF132236_160704_0226300.jpg
dtguidance	공연시간	화요일 ~ 금요일(20:00), 토 요일(16:00,19:00), 일요일(15:00,18:00)



02

연구 과정

PLAY DB

- Selenimum 라이브러리를 이용해서 크롤링
- PLAYDB에서 제작사, 제작진, 뮤지컬 배우 전체 크롤링
- 크롤링 후 데이터프레임으로 저장

02 연구 과정

	공연코드	좌석수	무대시설_무대넓이	공연일시	공연시작일자	공연종료일자	출연진내용	제작진내용	기획제작사명	레이블	포스터경로
2	PF193402	352	NaN	2022-06-22 20:00	2022-06-22	2022-06-25	김종구, 백기범, 김찬종, 정민, 홍승안, 조풍래	이현재, 성재준, 홍정의 등	(주)더 웨이브(제작사)	0.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
3	PF193398	352	NaN	2022-06-22 11:00	2022-06-22	2022-06-26	정욱진, 배나라, 백동현, 송원근, 고훈정 등	송은도, 임병우, 한경숙 등	(주)아떼오드(주최)	0.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
4	PF193364	638	NaN	2022-06-25 11:00	2022-06-25	2022-06-25	NaN	NaN	지니아트랩(제작사), 딴(기획사)	0.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
6	PF193177	638	NaN	2022-06-25 11:00	2022-06-15	2022-06-25	NaN	NaN	지니아트랩(제작사), 딴(기획사)	0.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
8	PF193148	288	NaN	2022-06-22 15:00	2022-06-22	2022-06-22	이정수, 이상엽, 이종호	이정수	놀이기획(주최), 옴니아트홀(주최), 옴니아트홀(기획사)	1.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
...	...	...	...	...	...	...	...	...	...	...	...
4849	PF146195	609	13500X7500	2020-01-10 19:30	2020-01-10	2020-02-02	김신기, 최나라, 이지연, 오재성, 김주희, 김수지, 이강민 등	김광보, 오세혁, 신재훈 등	서울시극단(주관), (재)세종문화회관(주최)	0.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
4857	PF142172	118	NaN	2019-07-06 10:30	2018-03-10	2021-06-30	NaN	소미경, 조성중, 김윤미 등	(주)팀플레이예술기획(제작사)	0.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
4869	PF136032	659	603	2019-07-01 17:00	2012-04-01	2022-12-31	NaN	NaN	(주)피엠씨프러덕션(PMC Production)(제작사)	1.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
4873	PF128014	323	NaN	2019-07-01 17:00	2015-10-01	2021-01-03	이한범, 김곤호, 설호열, 유승수, 고장환, 이동원, 김태완 등	NaN	(주)피엠씨프러덕션(PMC Production)(제작사)	0.0	/content/drive/MyDrive/KOPIS/posters_all/poste...
4877	PF122017	386	148	2019-07-01 17:00	2013-07-01	2020-04-30	손석배, 유승수, 황요한, 김문수, 김대호, 조기철, 이한범 등	송승환	(주)피엠씨프러덕션(PMC Production)(제작사)	0.0	/content/drive/MyDrive/KOPIS/posters_all/poste...

2308 rows × 11 columns

- Kopis api에서 추출한 '공연시작일자', '공연종료일자', '소요시간', '관람연령' 등을 토대로 식별화된 제공 데이터들의 실제 공연코드 결합.
- Kopis의 좌석규모 기준으로 중소극장에 해당하는 1000석 이하의 데이터 활용.
- 총 3093개의 데이터 중에서 중복 공연코드는 삭제 후 2307개의 데이터 활용.
- 레이블은 점유율을 고려한 판매 좌석수/총 좌석 수로 하며, 50%를 넘은 경우 흥행 성공:1, 흥행 실패:0으로 레이블

02 연구 과정

- 총 이미지 수 2308개를 활용
- 정형 데이터와 shape을 맞추기 위해 test_size = 0.08로 맞춤
- Train set의 크기는 2121, Test set의 크기는 185
- 모든 이미지의 크기는 224,224,3으로 통일
- 이미지를 Numpy 배열로 변환하여 학습
- VGG16 정확도: 0.7622

```
#이미지 데이터 전처리 후 배열로 변환
image_w = 224
image_h = 224
pixels = image_h * image_w * 3
X = {'train':[], 'test':[]}
Y = {'train':[], 'test':[]}
categories = ['0', '1']

x_list_str = ['train',  'test']

destination = '/content/drive/MyDrive/KOPIS/posters_all/'

for list_str in x_list_str:
  for idx, cls in enumerate(categories):
    label = [0 for i in range(len(categories))]
    label[idx] = 1

    image_dir = destination + list_str +'/' + cls
    files = glob.glob(image_dir+'/*.jpg')
    print(list_str, cls, " 파일 길이 : ", len(files))

    for n, k in enumerate(files):
      files[n] = k.split('/')[-1]
    files = sorted(files)

    for i, f in enumerate(files):
      f = destination + list_str + '/' + cls + '/' + str(f)
      img = Image.open(f)
      img = img.convert("RGB")
      img = img.resize((image_w, image_h))
      data = np.asarray(img)

      X[list_str].append(data)
      Y[list_str].append(label)

  print('ok', len(Y[list_str]))
X['train'] = np.array(X['train'])
Y['train'] = np.array(Y['train'])

X['test'] = np.array(X['test'])
Y['test'] = np.array(Y['test'])
```

```
train 0  파일 길이 :  1618
train 1  파일 길이 :  503
ok 2121
test 0  파일 길이 :  141
test 1  파일 길이 :  44
ok 185
```

```
Model: "vgg16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_3 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

=================================================================
Total params: 14,714,688
Trainable params: 0
Non-trainable params: 14,714,688
_________________________________________________________________
```

```
#이미지데이터 모델 성능 평가
print(model.evaluate(X['test'], Y['test']))
```

```
6/6 [==============================] - 6s 965ms/step - loss: 0.4919 - accuracy: 0.7622
[0.4919221699237823, 0.7621621489524841]
```

02

연구 과정

- 제공데이터와 중소규모 데이터를 결합하여 사용
- playDB에서 추출한 목록과 출연진내용, 제작진내용, 기획제작사명을 비교하여 포함될 경우 +1
- 출연진, 제작진, 제작사 수를 카운트하여 column으로 추가
- 사용한 항목은 오른쪽 하단과 같음
- 총 35개의 항목 이용

```
DF = DF[['공연코드', '시설특성', '편의시설_레스토랑 여부', '편의시설_카페 여부', '편의시설_편의점 여부', '편의시설_놀이방 여부',
       '편의시설_수유실 여부', '장애인시설_주차장 여부', '장애인시설_화장실 여부', '장애인시설_경사로 여부',
       '장애인시설_전용엘리베이터 여부', '주차시설_자체 여부', '주차시설_공영 여부', '좌석수', '장애인석',
       '공연일시', '공연시작일자', '공연종료일자', '소요시간', '공연지역명', 'director', 'actor', 'company',
       '관람연령', '아동공연 여부', '축제 여부', '내한공연 여부', '오픈런 여부',
       '출연진내용', '제작진내용', '기획제작사명', '레이블']]
DF
```

	공연코드	시설특성	편의시설_레스토랑 여부	편의시설_카페 여부	편의시설_편의점 여부	편의시설_놀이방 여부	편의시설_수유실 여부	장애인시설_주차장 여부	장애인시설_화장실 여부	장애인시설_경사로 여부	...	company	관람연령	아동공연 여부	축제 여부	내한공연 여부	오픈런 여부	출연진내용	제작진내용	기획제작사명	레이블
0	PF193402	민간(대학로)	N	Y	N	N	N	N	N	N	...	0	만 13세 이상	N	N	N	N	김종구, 백기범, 김잔중, 정민, 홍승안, 조풍래	이현재, 성재준, 홍정의 등	(주)더 웨이브(제작사)	0.0
1	PF193398	민간(대학로)	N	Y	N	N	N	N	N	N	...	0	만 13세 이상	N	N	N	N	정욱진, 배나라, 박동현, 송원근, 고훈정 등	송은도, 임병우, 한경숙 등	(주)아떼오드(주최)	0.0
2	PF193364	공공(문예회관)	N	N	N	N	N	Y	N	N	...	0	24개월 이상	Y	N	N	N	0	0	지니아트랩(제작사), 딴(기획사)	0.0
3	PF193177	공공(문예회관)	N	N	N	N	N	Y	N	N	...	0	24개월 이상	Y	N	N	N	0	0	지니아트랩(제작사), 딴(기획사)	0.0
4	PF193148	공공(기타)	N	Y	N	N	N	Y	Y	Y	...	0	24개월 이상	Y	N	N	N	이정수, 이상엽, 이종호	이정수	놀이기획(주최), 옴니아트홀(주최), 옴니아트홀(기획사)	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4357	PF146195	공공(문예회관)	N	N	N	N	N	Y	Y	N	...	0	48개월 이상	Y	N	N	N	김신기, 최나라, 이지연, 오재성, 김주희, 김수지, 이강민 등	김광보, 오세혁, 신재훈 등	서울시극단(주관), (재)세종문화회관(주최)	0.0
4359	PF142172	민간(대학로)	N	N	N	N	N	N	N	N	...	0	24개월 이상	Y	N	N	Y	0	소미경, 조성중, 김윤미 등	(주)팀플레이예술기획(제작사)	0.0
4361	PF136032	민간(대학로 외)	Y	Y	Y	N	N	Y	Y	Y	...	0	12개월 이상	N	N	N	Y	0	0	(주)피엠씨프러덕션(PMC Production)(제작사)	1.0
4363	PF128014	민간(대학로 외)	N	Y	Y	N	N	N	N	N	...	0	12개월 이상	N	N	N	Y	이한범, 김곤호, 설호열, 유승수, 고창환, 이동원, 김태완 등	0	(주)피엠씨프러덕션(PMC Production)(제작사)	0.0
4365	PF122017	민간(대학로 외)	N	N	N	N	N	N	N	N	...	0	12개월 이상	N	N	N	Y	손석배, 유승수, 황요한, 김문수, 김대호, 조기철, 이한범 등	송승환	(주)피엠씨프러덕션(PMC Production)(제작사)	0.0

2307 rows × 35 columns

```
Index(['공연코드', '시설특성', '편의시설_레스토랑 여부', '편의시설_카페 여부', '편의시설_편의점 여부',
       '편의시설_놀이방 여부', '편의시설_수유실 여부', '장애인시설_주차장 여부', '장애인시설_화장실 여부',
       '장애인시설_경사로 여부', '장애인시설_전용엘리베이터 여부', '주차시설_자체 여부', '주차시설_공영 여부', '좌석수',
       '장애인석', '무대시설_오케스트라피트 여부', '무대시설_연습실 여부', '무대시설_분장실 여부', '공연일시',
       '공연시작일자', '공연종료일자', '소요시간', '공연지역명', 'director', 'actor', 'company',
       '관람연령', '아동공연 여부', '축제 여부', '내한공연 여부', '오픈런 여부', '출연진내용', '제작진내용',
       '기획제작사명', '레이블', '제작진수', '출연진수', '제작사수'],
      dtype='object')
```

02 연구 과정

- 제공 데이터의 대부분의 항목들은 범주형이므로 숫자로 변경
- 시설 특성은 6가지로 나눔
- 시설 여부 및 축제, 내한공연, 오픈런 여부는 Y 혹은 N으로 나눠지므로 1, 2로 변경
- 소요시간은 21가지로 나눠짐
- 공연 지역은 서울, 충청도, 경기도, 경상도, 전라도, 강원도로 6으로 나눔
- 관람연령은 1~3까지 어린이, 청소년, 성인 나이 기준으로 나눔
- 출연진내용, 제작진내용, 기획제작사명은 인물 수 카운트와 playDB 목록과 비교하여 숫자화
- 이미지 데이터를 0.08 비율로 나눴으므로 이미지 인덱스와 맞춰 train test로 분리

```
import datetime

def preprocessing(df):

  df.fillna(0, inplace=True)

  df['시설특성'] = df['시설특성'].map({'민간(대학로 외)' : 1, '민간(대학로)' : 2, '공공(기타)' : 3,
                                  '국립' : 4, '공공(문예회관)' : 5, '기타(비공연장)' : 6})

  df['편의시설_레스토랑 여부'] = df['편의시설_레스토랑 여부'].map({'Y' : 1, 'N' : 2})
  df['편의시설_카페 여부'] = df['편의시설_카페 여부'].map({'Y' : 1, 'N' : 2})
  df['편의시설_편의점 여부'] = df['편의시설_편의점 여부'].map({'Y' : 1, 'N' : 2})
  df['편의시설_놀이방 여부'] = df['편의시설_놀이방 여부'].map({'Y' : 1, 'N' : 2})
  df['편의시설_수유실 여부'] = df['편의시설_수유실 여부'].map({'Y' : 1, 'N' : 2})
  df['장애인시설_주차장 여부'] = df['장애인시설_주차장 여부'].map({'Y' : 1, 'N' : 2})
  df['장애인시설_화장실 여부'] = df['장애인시설_화장실 여부'].map({'Y' : 1, 'N' : 2})
  df['장애인시설_경사로 여부'] = df['장애인시설_경사로 여부'].map({'Y' : 1, 'N' : 2})
  df['장애인시설_전용엘리베이터 여부'] = df['장애인시설_전용엘리베이터 여부'].map({'Y' : 1, 'N' : 2})
  df['주차시설_자체 여부'] = df['주차시설_자체 여부'].map({'Y' : 1, 'N' : 2})
  df['주차시설_공영 여부'] = df['주차시설_공영 여부'].map({'Y' : 1, 'N' : 2})
  df['장애인석'] = df['장애인석'].map({'Y' : 1, 'N' : 2})
  df['무대시설_오케스트라피트 여부'] = df['무대시설_오케스트라피트 여부'].map({'Y' : 1, 'N' : 2})
  df['무대시설_연습실 여부'] = df['무대시설_연습실 여부'].map({'Y' : 1, 'N' : 2})
  df['무대시설_분장실 여부'] = df['무대시설_분장실 여부'].map({'Y' : 1, 'N' : 2})
  df['아동공연 여부'] = df['아동공연 여부'].map({'Y' : 1, 'N' : 2})
  df['축제 여부'] = df['축제 여부'].map({'Y' : 1, 'N' : 2})
  df['내한공연 여부'] =  df['내한공연 여부'].map({'Y' : 1, 'N' : 2})
  df['오픈런 여부'] = df['오픈런 여부'].map({'Y' : 1, 'N' : 2})

  #df['극작가명'] = df['극작가명'].map({'손님(각색)' : 1})

  df['소요시간'] = df['소요시간'].map({'2시간 20분' : 1, '1시간 30분' : 2, '1시간 50분' : 3, '1시간' : 4, '1시간 25분' : 5, '1시간 40분' : 6, '2시간' : 7,
                                  '1시간 35분' : 8, '2시간 30분' : 9, '1시간 10분' : 10, '1시간 20분' : 11, '1시간 45분' : 12, '55분' : 13,
                                  '2시간 10분' : 14, '2시간 5분' : 15, '50분' : 16, '1시간 15분' : 17, '2시간 15분' : 18, '1시간 5분' : 19,
                                  '2시간 35분' : 20, '2시간 40분' : 21})

  df['공연지역명'] = df['공연지역명'].map({'서울' : 1, '충청도' : 2, '경기도' :3, '경상도' : 4, '전라도' : 5, '강원도' : 6})

  df['관람연령'] = df['관람연령'].map({'전체 관람가' : 1, '만 13세 이상' : 2, '만 14세 이상' : 2, '만 15세 이상' :2, '만 17세 이상' : 2,
                                  '만 16세 이상' : 2, '만 12세 드림아트센터이상' : 2, '만 8세 이상' : 3, '만 7세 이상': 3, '만 5세 이상': 3, '만 10세 이상': 3,
                                  '만 11세 이상': 3, '36개월 이상': 3, '만 9세 이상': 3, '만 6세 이상': 3, '만 4세 이상': 3, '24개월 이상': 3, '48개월 이상': 3, '만 3세 이상': 3})
```

	공연코드	시설특성	편의시설_레스토랑 여부	편의시설_카페 여부	편의시설_편의점 여부	편의시설_놀이방 여부	편의시설_수유실 여부	장애인시설_주차장 여부	장애인시설_화장실 여부	장애인시설_경사로 여부	...	축제 여부	내한공연 여부	오픈런 여부	레이블	제작진수	출연진수	제작사수	시작-종료 일수	공연일시_요일	공연일시_시간
0	PF193402	2	2	1	2	2	2	2	2	2	...	2	2	2	0.0	3.0	6.0	1.0	3	2	200000
1	PF193398	2	2	1	2	2	2	2	2	2	...	2	2	2	0.0	3.0	5.0	1.0	4	2	110000
2	PF193364	5	2	2	2	2	2	1	2	2	...	2	2	2	0.0	0.0	0.0	2.0	0	5	110000
3	PF193177	5	2	2	2	2	2	1	2	2	...	2	2	2	0.0	0.0	0.0	2.0	10	5	110000
4	PF193148	3	2	1	2	2	2	1	1	1	...	2	2	2	1.0	1.0	3.0	3.0	0	2	150000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4357	PF146195	5	2	2	2	2	2	1	1	2	...	2	2	2	0.0	3.0	7.0	2.0	23	4	193000
4359	PF142172	2	2	2	2	2	2	2	2	2	...	2	2	1	0.0	3.0	0.0	1.0	1208	5	103000
4361	PF136032	1	1	1	1	2	2	1	1	1	...	2	2	1	1.0	0.0	0.0	1.0	3926	0	170000
4363	PF128014	1	2	1	1	2	2	2	2	2	...	2	2	1	0.0	0.0	7.0	1.0	1921	0	170000
4365	PF122017	1	2	2	2	2	2	2	2	2	...	2	2	1	0.0	1.0	7.0	1.0	2495	0	170000

2307 rows × 35 columns

02 연구 과정

- 정형데이터의 경우 XGBOOST 활용
- XGBoost(Extreme Gradient Boosting)은 이름에서 볼 수 있듯 앙상블의 Boosting 기법을 이용하여 병렬학습을 구현하여 Validation 기준 logloss는 초반 0.7에서 학습완료까지 0.5으로 수렴하여 적합한 학습 수행
- Xgboost 정확도: 0.8270

```
#정형데이터 불러오기

train = pd.read_csv('/content/drive/MyDrive/KOPIS/정형데이터/train_nums2.csv', index_col=0)
test = pd.read_csv('/content/drive/MyDrive/KOPIS/정형데이터/test_nums2.csv', index_col=0)

train = train.dropna(subset=['공연코드'])

y_train = train['레이블']
y_test = test['레이블']
x_train = train.drop(['레이블', '공연코드'], axis=1)
x_test = test.drop(['레이블', '공연코드'], axis=1)
```

```
#정형데이터 모델 불러와서 xgboost 학습
from xgboost import XGBClassifier

model = XGBClassifier(n_estimators=2000,
                      seed=1234,
                      max_depth=10,
                      learning_rate=0.0015,
                      num_class = 1)

xgb_model = model.fit(x_train, y_train, early_stopping_rounds=100,
                      eval_metric='logloss', eval_set=[(x_test, y_test)])
```

```
xgboost_pred = model.predict(x_test)
print('XGBOOST 정확도 :',accuracy_score(y_test, xgboost_pred))
```

```
XGBOOST 정확도 : 0.8270270270270270
```

연구 과정

- XGBOOST와 VGG16을 합치기 위해 stacking 앙상블 기법 사용.
- 수월한 모델 학습을 위해 처리한 이미지 데이터를 정규화.
- Xgboost의 predict 수치와 vgg16의 predict 수치를 결합
- Xgboost + VGG16 앙상블 정확도: 83.24%

```
#이미지 데이터 크기 정규화 후 vgg16 모델 로딩
X['train'] = X['train'].astype(float) / 255
X['test'] = X['test'].astype(float) / 255

from tensorflow.python.keras.models import load_model
from sklearn.metrics import accuracy_score
import joblib

vgg = load_model('/content/drive/MyDrive/KOPIS/정형데이터/중소2vgg.h5')

vgg_pred = vgg.predict(X['test'])

vgg_pred = np.argmax(vgg_pred, axis=1)
Y['test'] = np.argmax(Y['test'], axis=1)
```

```
from sklearn.metrics import accuracy_score

pred1 = xgb_model.predict_proba(x_test)
pred2 = vgg.predict(X['test'])

print(pred1)
print(' ')
print(pred2)
```

```
[[0.9704728  0.02952721]
 [0.9480162  0.0519838 ]
 [0.79223305 0.20776697]
 [0.82024443 0.17975557]
 [0.93733394 0.06266604]
 [0.79213804 0.20786195]
 [0.815159   0.18484099]
 [0.6397983  0.3602017 ]
 [0.702687   0.29731297]
 [0.8791465  0.12085351]
 [0.95938146 0.04061855]
 [0.7110537  0.28894624]
```

```
arr = []
for n in range(len(pred1)):
  rows = []
  for k in range(2):
    m = (pred1[n][k] + pred2[n][k]) / 2
    rows.append(m)
  arr.append(rows)

pred = []
for pr in arr:
  k = np.argmax(pr)
  pred.append(k)

print(arr)
print(' ')

accuracy = accuracy_score(y_test, pred)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
```

```
[[0.7871100306510925, 0.30123162269592285], [0.77

Accuracy: 83.24%
```

연구 결과

	공연코드	시설특성	편의시설_레스토랑 여부	편의시설_카페 여부	편의시설_편의점 여부	편의시설_놀이방 여부	편의시설_수유실 여부	장애인시설_주차장 여부	장애인시설_화장실 여부	장애인시설_경사로 여부	...	축제 여부	내한공연 여부	오픈런 여부	레이블	제작 진수	출연 진수	제작 사수	시작-종료 일수	공연일시_요일	공연일시_시간
0	PF193402	2	2	1	2	2	2	2	2	2	...	2	2	2	0.0	3.0	6.0	1.0	3	2	200000
1	PF193398	2	2	1	2	2	2	2	2	2	...	2	2	2	0.0	3.0	5.0	1.0	4	2	110000
2	PF193364	5	2	2	2	2	2	1	2	2	...	2	2	2	0.0	0.0	0.0	2.0	0	5	110000
3	PF193177	5	2	2	2	2	2	1	2	2	...	2	2	2	0.0	0.0	0.0	2.0	10	5	110000
4	PF193148	3	2	1	2	2	2	1	1	1	...	2	2	2	1.0	1.0	3.0	3.0	0	2	150000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4357	PF146195	5	2	2	2	2	2	1	1	2	...	2	2	2	0.0	3.0	7.0	2.0	23	4	193000
4359	PF142172	2	2	2	2	2	2	2	2	2	...	2	2	1	0.0	3.0	0.0	1.0	1208	5	103000
4361	PF136032	1	1	1	1	2	2	1	1	1	...	2	2	1	1.0	0.0	0.0	1.0	3926	0	170000
4363	PF128014	1	2	1	1	2	2	2	2	2	...	2	2	1	0.0	0.0	7.0	1.0	1921	0	170000
4365	PF122017	1	2	2	2	2	2	2	2	2	...	2	2	1	0.0	1.0	7.0	1.0	2495	0	170000

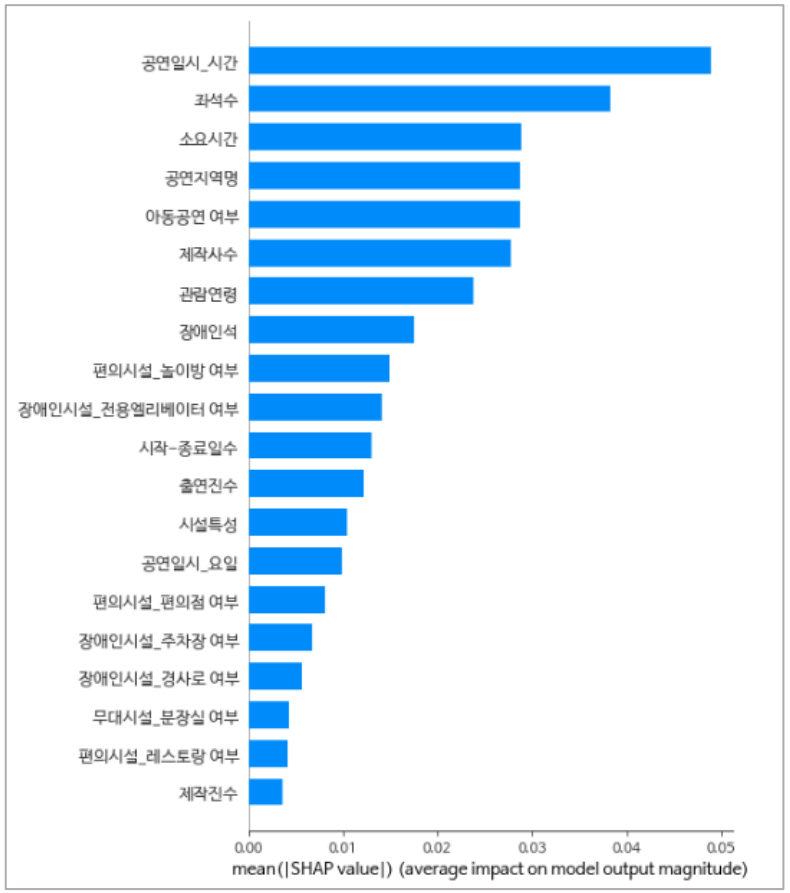
2307 rows × 35 columns

- target data인 '흥행수치'의 개별 예측 값에 대한 각 변수들의 영향력을 계산하기위해 XAI 방법 중 하나인 SHAP를 사용.
- 모델을 설명하기위해 Shapley Value라는 값을 계산.
- Shapely Value란 게임이론을 통해 개별 플레이어들의 기여도를 수치화한 값이다. Shapley Value를 통해 여러 변수 중 한 가지의 변수가 결과에 미치는 중요도 판단 가능.
- 정형데이터에 사용했던 모든 항목을 똑같이 사용하여 총 35개의 Feature을 사용.

연구 결과

- 해당 그래프에서는 각 Feature의 Shapely Value 절대값의 평균을 Plot하여, 각 Feature가 예측 값에 미치는 평균적인 영향력을 확인.
- 큰 영향력을 보일수록, target과 관계성이 크다는 것을 나타냄.
- 위 그래프를 통해 공연일시_시간, 좌석수 순으로 예측 값에 영향을 주는 것을 확인.

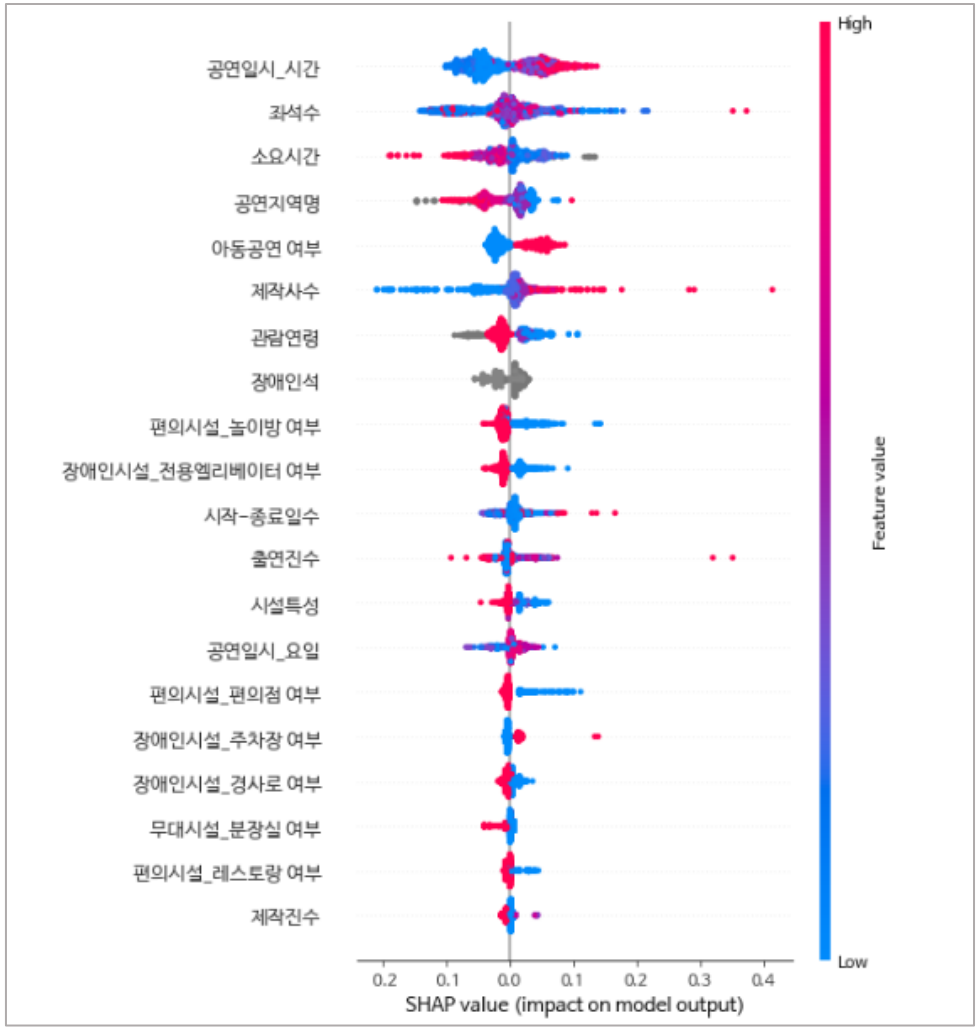
SHAP Feature Importance Plot



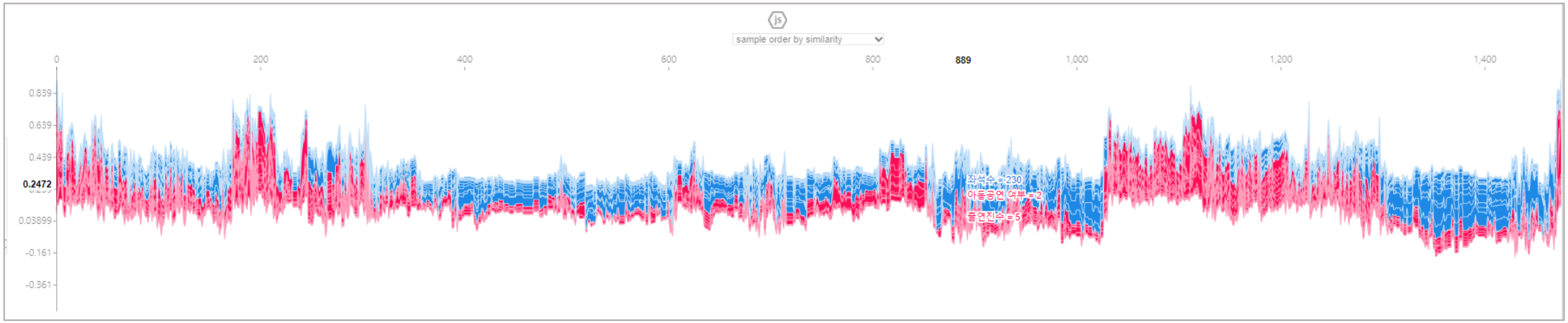
연구 결과

- 위 그래프는 모든 변수들의 shap value를 요약한 것.
- x축은 Shapley value에 의해 결정되고 y축은 특성에 의해 결정.
- 해당 변수가 빨간색을 띄면 target에 대해 양의 영향력이 존재하는 것이고, 파란색을 띄면 음의 영향력이 존재하는 것.
- 겹치는 점이 y축 방향으로 내포됨에 따라 특성 당 Shapley value의 분포를 알 수 있다.
- 또한, 특성은 중요도에 따라 정렬이 된다. 가령 위 그래프에서 actor, 즉 'rank에 해당하는 출연진 수' 변수를 봤을 때, 변수의 값이 높을수록 흥행에 성공할 높은 경향성이 있다고 볼 수 있다.

SHAP summary plot



03 연구 결과



- SHAP 패키지에서 시각화를 통해 전체 데이터 해석.
- 전체 데이터 해석시에는 각 열에 대한 Shapley Value를 누적하여 시각화.
- shapley value를 사용하여 데이터를 군집화.
- 위 그래프에서 x축의 각 위치는 관측치를 나타낸다.
- 빨간색 Shapley Value 예측을 증가시키고, 파란색 Shapley Value은 예측을 감소.
- 그래프를 보면 왼쪽 그룹의 경우 흥행에 성공할 확률이 높은 그룹이 있음을 확인.
- 해당 그래프를 통해 각 관측치의 shapely value에서 군집화하여 클러스터링.

04 기대 효과

- 해당 모델의 결과를 통해 흥행 예측에 영향을 주는 변수 해석을 통해 다양한 전략을 수립하는데 근거가 될 수 있음.
- 뮤지컬 분야 뿐만이 아니라 다른 공연예술분야에서도 활용이 가능할 것.
- 영화, 홍보, 광고 등 포스터를 기반으로 하는 광고, 홍보 영역에서 적용 가능

04

참고 논문 및 출처

- 조유정, 강경표 and 권오병. (2021). 공연예술에서 광고포스터의 이미지 특성을 활용한 딥러닝 기반 관객예측. 한국전자거래학회지, 26(2), 19-43.
- 김유리, 『[기획-1] 뮤지컬 포스터-제작 과장 [No.97]. 더뮤지컬. 2011-10-21』
- 공지은. "국내 뮤지컬포스터 디자인의 현황과 조형요소 분석." 국내석사학위논문 성신여자대학교, 2006. 서울
- 배준혁. "국내 창작뮤지컬 포스터의 디자인 연구." 국내석사학위논문 嶺南大學校, 2012. 경상북도