

영화 흥행은 어디에서 오는가

감독인가?





김지운 감독

배우인가?

아니면 재밌는 스토리인가?



영화 흥행은 어디에서 오는가

인공신경망, K-NN군집, 의사결정나무 알고리즘을 통한
영화 관객수 예측모델 구축

GROUP : TEAMSIX

이종범, 장재원, 김익환, 박창영, 김도은, 송준영



임무분담



이름	직책	임무
장재원	팀장	임무 총괄 및 기획, 나이트 베이스 알고리즘 연구, SNS 크롤링 분석, 시계열 데이터 분석, 데이터 클렌징, 전문가 인터뷰
이종범	기술팀장, 팀의 수석 테크니션	KNN알고리즘 연구, FNN알고리즘 연구, SNS 크롤링 분석, 데이터 시각화, 데이터 클렌징, 시계열 데이터 분석, 감독/배우 지수 분석
김익환	미필막내, 팀의 인텔리 담당	의사결정 나무 알고리즘 연구, PPT 편집, 영화 예측관련 논문 분석, 데이터 클렌징, 전문가 섭외 및 인터뷰
박창영		ANN 알고리즘 연구, 데이터 클렌징, 영화 손익분기점 분석, 감독/배우 지수 분석
김도은	대장	데이터 클렌징, PPT 편집, 선형회귀 연구, 데이터 클렌징, 전문가 섭외 및 인터뷰
송준영		나이트 베이스 알고리즘 연구, 영화 배급사 현황 조사, 데이터 클렌징

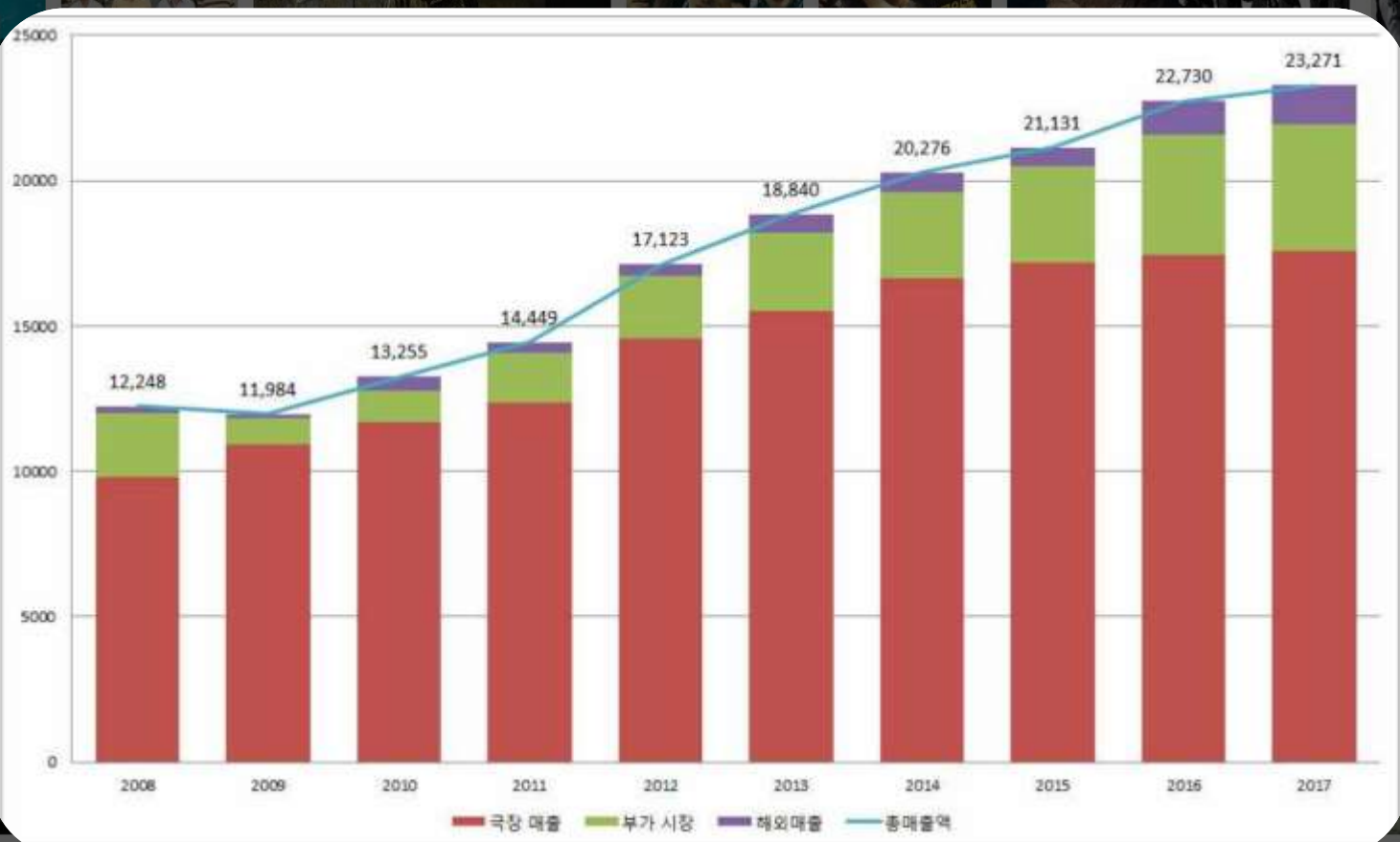


이 연구가 왜 중요한가?

문제 의식 제고

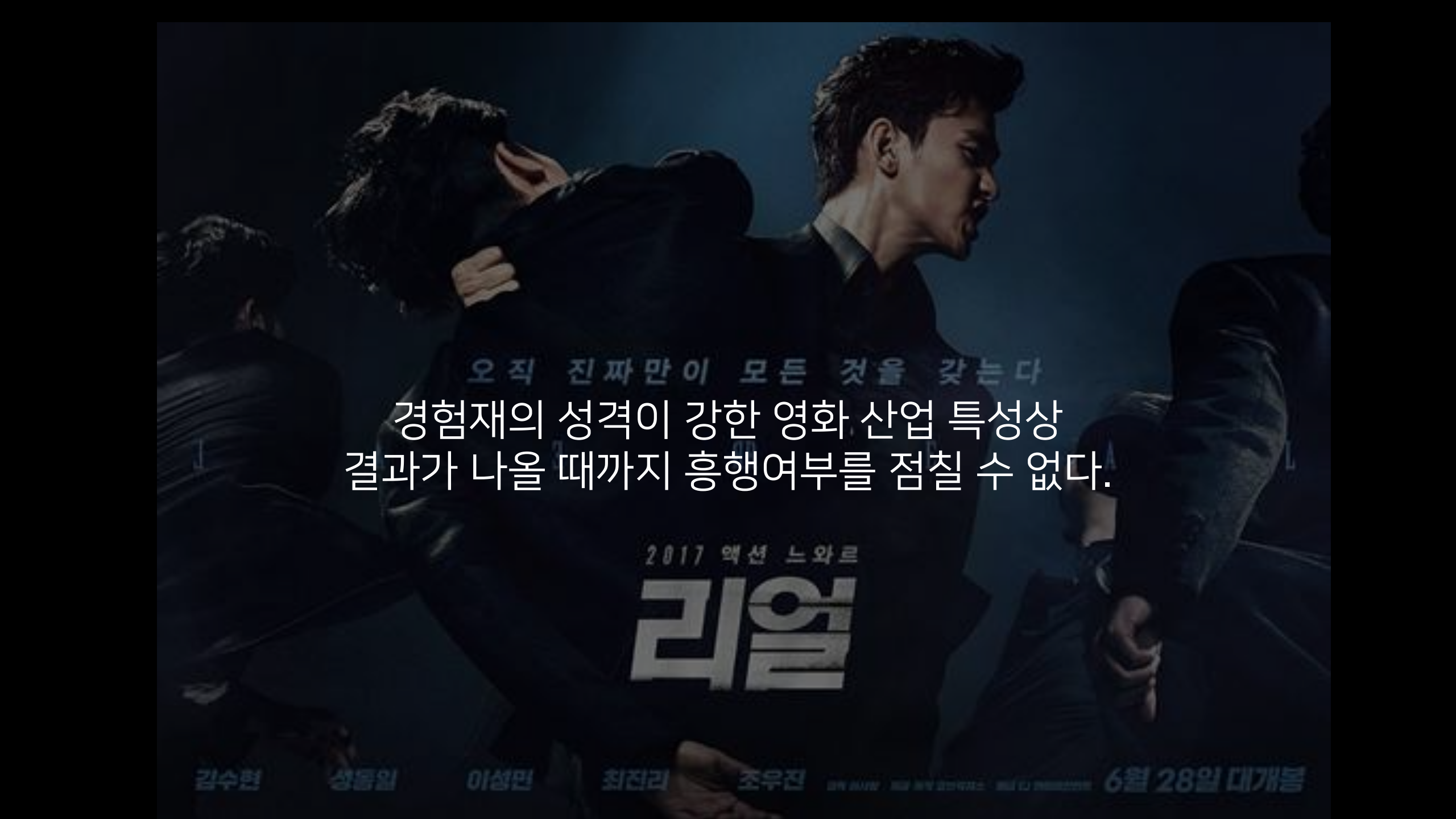


해마다 증가하는 영화산업 규모 연평균 15.9% 증가



2008-2017년 한국 영화산업 매출 추이
(출처 : 영화진흥위원회 2017)





오직 진짜만이 모든 것을 갖는다

경험자의 성격이 강한 영화 산업 특성상
결과가 나올 때까지 흥행여부를 점칠 수 없다.

2017 액션 느와르

리얼

김수현

성동일

이성민

최전리

조우전

감독 김사범 / 제작 박영호, 김민석, 김민준 / 배급 CJ 엔터테인먼트

6월 28일 대개봉

누구도 건드릴 수 없는 놈이 북에서 넘어왔다


흥행 Risk를 크게 느낄수록
해마다 관객수가 늘어가더라도

‘새로움’에 대한 시도와 투자는 점점 없어지고
한국 영화는 안전하게 흥행한 전작을 답습하고 있다.

〈신세계〉 박훈정 감독 작품

장동건 김명민 박희순 이종석

8월 23일 대개봉

A man with dark, wavy hair and a serious expression is pointing both index fingers directly at the camera. He is wearing a light green jacket over a dark shirt. He is holding a large, light-colored paper bag in front of him. The background shows shelves in a shop, with various items like bottles and boxes on the left, and a display of framed photos or posters on the right.

천편일률

흥행이 뭐길래?

어떤 데이터, 어떻게 모았는가

데이터 수집 및 전처리 과정

프로젝트 사용 변수

변수	변수명	출처	비고
종속 변수	전국 관객수	www.kobis.or.kr	
	전국 매출액	www.kobis.or.kr	전국 관객수와 1의 상관관계 (정비례)
독립 변수	상영 스크린 수	www.kobis.or.kr	
	개봉 전, 후 평점	movie.naver.com	기술적 제한
	개봉 전, 후 뉴스빈도	www.bigkinds.or.kr	기술적 제한
	개봉 1주차 관객수	www.kobis.or.kr	상영일과 개봉일의 차가 7일인 데이터만 정제
	제작비 및 마케팅비	영화별 검색	비공개인 영화가 다수이므로 손익분기점 손수 검색
	국적	www.kobis.or.kr	국적 중에서도 대표국적만을 사용
	장르	www.kobis.or.kr	장르 중에서도 대표장르만을 사용
	배우 및 감독 (스타파워)	www.cine21.com	최근 2년 활동한 배우 300명 대상 배우지수를 정규화
	배급사 및 제작사	www.kofic.or.kr	상영 스크린수와 다중공선성문제로 인하여 배제
	SNS빈도(Instagram, facebook, twitter)	Startag.io, fb.com twitter.com	R프로그램 패키지: Rfacebook ::, twitterR ::
	주별 종합 데이터	kofic.co.kr	시계열분석을 위하여 수집

데이터 수집 및 전처리 과정

개봉후 1주일 데이터

영화별 주간 데이터 수집(www.kofic.or.kr)

1. 상영일과 개봉일 차이 비교
2. 엑셀 함수를 사용하여 상영일과 개봉일의 차를 ± 7 로 두고 결과값 출력

▼	영화명	▼	누적관객수	▼	기간	▼	개봉일	▼	상영일	▼	날짜차이
65	팔로우		32567		15.04.09-04.15		2015-04-02		2015-04-09		7
138	파울볼		29232		15.04.09-04.15		2015-04-02		2015-04-09		7
129	윈드랜드		23432		15.04.09-04.15		2015-04-02		2015-04-09		7
29	모스트 바이어런트		9272		15.04.09-04.15		2015-04-02		2015-04-09		7
53	맛있는 택배		6276		15.04.09-04.15		2015-04-02		2015-04-09		7
81	맵 투 더 스타		6199		15.01.01-01.07		2014-12-25		2015-01-01		7
53	화이트 갓		2345		15.04.09-04.15		2015-04-02		2015-04-09		7
162	원령		912		15.08.20-08.26		2015-08-13		2015-08-20		7

데이터 수집 및 전처리 과정

배우 및 감독 (스타파워)

1. 영화별 배우/감독 데이터 크롤링
2. 배우지수 (스타파워)로 환산(Frequency / Performance)
3. 배우지수 데이터 정규화 후 영화별 배우 데이터에 병합

KOFIC 영화관입장권통합전산망 오픈API

시네마



normalization

		V	W	X			
	1987	김윤석	0.227169	하정우	0.510277	유해진	0.76
4	리얼	김수현	0	성동일	0.092595	이성민	0.0493
22	신과함께-	하정우	0.510277	차태현	0.291644	주지훈	0.269048
63	암살	전지현	0	이정재	0.062567	하정우	0.510277
94	국제시장	황정민	0.472483	김윤진	0	오달수	0.49350
	인랑	강동원	0	한효주	0	정우성	0.043
		류승룡	0	심은경	0		
		대포	0	브루클린	0		

데이터 수집 및 전처리 과정

손익분기점 = 제작비용 + 마케팅비용

- 한국영화 손익분기점 : 목표 관객 수
- 외국영화 손익분기점 : 최소목표 매출액(달러)

감독	김봉환
각본	김봉환
원작	주호민
제작	최지선
발행	김병서
조형	신경만
음악	박준혁
출연	최정호, 주지훈, 김광진, 이종석, 김동욱 등
장르	범죄자, 도둑
제작사	[K] 리얼라이즈 픽처스, 에픽스튜디오
배급사	[K] 롯데엔터테인먼트 [USA] Wee Go USA
수입사	[USA] Wee Go USA
제작 기간	2016년 8월 30일 ~ 2017년 3월 30일
제작비	순 제작비 350억 원(전환 할인) 총 제작비 400억 원(전환 할인)
개봉일	[K] 2018년 8월 1일
상영 시간	141분
컬트 박스오피스	\$48,223,338 (2018년 8월 9일 기준)
복합 박스오피스	\$638,884 (2018년 8월 9일 기준)
대한민국 총 관객수	8,859,600명 (2018년 8월 13일 기준)
국내등급	12세 이상 관람가



A	B	S	T
순위	영화명	손익분기점	
19	어벤져스:	6E+08	
37	캡틴 아메리카: 시빌 워		
29	쥬라기 월드	3.4E+08	
1	미션 임파서블	3.56E+08	
22	신과함께-죄	700000000	
16	어벤져스: 에이지 오브 울트론		
91	검사외전		
170	부산행		
5	앤트맨과	3.24E+08	
3	트랜스포머	4.34E+08	
45	배트맨 레		

데이터 수집 및 전처리 과정

주별 종합 시계열 데이터

* 2015년부터 2018년 까지 주단위로 나누어 팀원 6명이 데이터 전처리

데이터 정제

- 재개봉작 제거
- 스크린 수, 총 관객수가 0 또는 1 데이터 제거
- 개봉일과 상영일이 과도하게 차이나는 데이터 제거(예 : 1960년대 이벤트성 재개봉 영화)



순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국매출액	전국관객수
1	명량	김한민	(주)빅스톤픽처스		씨제이이엔엠(주)	2014-07-30	개봉영화	장편	한국	1,587	1.35748E+11	17,613,682
2	신과함께-죄와 벌	김용화	리얼라이즈픽처스(주),(주)텍스터스튜디오		롯데쇼핑(주)롯데엔터테인먼트	2017-12-20	개봉영화	장편	한국	1,912	1.157E+11	14,410,931
3	국제시장	윤제균	(주)제이케이필름,씨제이이엔엠(주)		씨제이이엔엠(주)	2014-12-17	개봉영화	장편	한국	966	1.10828E+11	14,245,998
4	베테랑	류승완	(주)외유내강,(주)필름케이		씨제이이엔엠(주)	2015-08-05	개봉영화	장편	한국	1,064	1.05025E+11	13,395,400

예측 모델 생성 및 실험 결과

본론

1. Artificial Neural Network

1. 데이터 클렌징

→ ANN 알고리즘은 임의의 두 변수 간의 편차가 클 경우 제대로 학습되지 않는 특징이 있기 때문에 변수간 normalization 함 : $\text{function}(x) \{ (x - \min(x)) / (\max(x) - \min(x)) \}$

X	name	dir_score	actor_1	actor_2	ac_sales	ac_attend	screen	show_freq	deadline	attend
1	57 내부자들	0	0.031759	0.14972	0.487931	0.490753	0.589838	0.0002	0.033285	0.054543
2	78 뷰티 인사	0	0.03228	0.303428	0.138793	0.142492	0.337873	8.59E-05	0.029951	0.005822
3	11 대립군	0	0.02997	0.079919	0.054052	0.058074	0.422211	0.000687	0.054953	0.076167
4	33 박열	0.165522	0.046733	0.077888	0.096552	0.101136	0.614458	1	0.024951	0.107568
5	79 봉이 김선	0	0.024849	0.347884	0.139655	0.14233	0.491881	0.000115	0.049952	0.075314
6	50 부산행	0.045808	0.038118	0.247765	0.803448	0.802542	0.935045	5.73E-05	0.054953	0.75038
7	99 미쓰 와이	0	0	0	0.063448	0.068212	0.227344	2.86E-05	0.021618	0.001773
8	22 보안관	0	0.100443	0.175775	0.17931	0.179583	0.569932	0.000229	0.033285	0.114736
9	155 퇴마: 무녀	0	0.000736	0.009574	0.008207	0.008381	0.171818	0.000745	0.016617	0.001644
10	88 해빙	0	0.079986	0.239957	0.085431	0.083589	0.477737	0.000659	0.019951	0.203118
11	63 더 킹	0	0.021399	0.069547	0.375	0.36889	0.684652	0.000458	0.063953	0.15816
12	105 카트	0	0.040747	0.203736	0.051034	0.056478	0.000524	0.000745	0.021618	0
13	161 견기왕	0	0	0	0.006103	0.006598	0.24044	0.000115	0.00745	0.005225
14	106 상의원	0	0.003476	0.029543	0.053448	0.054831	0.136721	0.00063	0.049952	0.054162
15	125 협녀, 칼의	0	0.037025	0.148099	0.028965	0.029923	0.298062	0	0.058286	0.002074
16	98 그놈이다	0	0.061013	0.366079	0.069827	0.072586	0.35516	0.000859	0.019951	0.053335

X	name	dir_score	actor_1	actor_2	ac_sales	ac_attend	screen	show_freq	deadline	attend
1	2 리얼	0.007134	0.044614	0.111534	0.026465	0.029596	0.506548	0.516398	0.116622	0.042013
2	3 신과함께-	1	0.35699	0.35699	1	1	1	0.004153	0.113289	1
3	6 인랑	0	0.010671	0.042682	0.050862	0.052694	0.566789	0.532552	0.099955	0.213014
4	12 곡성	0	0.02027	0.253381	0.481897	0.477418	0.776323	0.000544	0.049952	0.048514
5	16 임금님의	0	0	0	0.108621	0.113417	0.55055	0.000831	0.049952	0.032799
6	18 골든슬럼	0.095505	0.088009	0.176018	0.098276	0.096287	0.434783	0.000659	0.044952	0.27507
7	24 미옥	0.009239	0.021335	0.064006	0.016638	0.016844	0.331063	0.000401	0.033285	0.053975
8	27 석조저택	0	0.082519	0.103148	0.02431	0.024422	0.280775	2.86E-05	0.031618	0.031631
9	29 쾰	0.285194	0.153957	0.153957	0.269828	0.278826	0.686747	0.001031	0.029951	0.466468
10	32 재심	0	0.124598	0.249197	0.166379	0.167983	0.488214	0.000601	0.026618	0.035342
11	41 카이: 겨울	0	0	0	0.00156	0.00169	0.088528	0.001604	0.003283	0.00063
12	43 마리안느	0	0	0	0.001672	0.00192	0.031954	0.000344	0.000783	0.002901
13	44 피의 연대	0	0	0	0.000659	0.000678	0.012048	0.000372	0.000783	0.0001
14	51 검사의전	0	0.019866	0.205287	0.666379	0.673606	0.947617	8.59E-05	0.044952	0.150843
15	52 공조	0	0.161838	0.377622	0.55	0.542482	0.727606	0.000659	0.049952	0.044708
16	53 히말라야	0	0.02953	0.472483	0.518965	0.538458	0.572027	0	0.069953	0.062552
17	59 럭키	0	0.054298	0.705878	0.486207	0.483978	0.64484	0.001919	0.033285	0.037847
18	60 연평해전	0	0	0	0.393103	0.419457	0.529073	8.59E-05	0.039952	0.043716
19	61 덕혜옹주	0	0.000982	0.015215	0.382759	0.388409	0.503405	8.59E-05	0.058286	0.078701

2. 변수 간의 상관관계 분석

```
> Train<-read.csv("Train2.csv",header = TRUE)
> Test<-read.csv("Test2.csv",header = TRUE)

> cor(Train$ac_attend,Train$dir_score) # 누적 판매량과 감독점수
[1] 0.32543548
> cor(Train$ac_attend,Train$actor_2)    # 누적 판매량 과 배우점수
[1] 0.3104214129
> cor(Train$ac_attend,Train$screen)     # 누적 판매량 과 스크린
[1] 0.7097708609
> cor(Train$ac_attend,Train$deadline)   # 누적 판매량 과 손익 분기점
[1] 0.3116334713
> cor(Train$ac_attend,Train$attend)     #누적 판매량과 개봉후 1주일 관람객수
[1] 0.4459931448
```

위는 Train셋의 누적 관객수와의 상관계수

1. screen 2. attend 3. dir_score 4. deadline 5. actor_2 순으로 상관계수가 높음

➔ 독립변수에 대한 insight 획득

3. 예측 모델링(1차 실험)

```

model <- neuralnet(ac_attend ~ screen + attend ,
                    data = Train , hidden=c(5,3))
# 사전에 분류한 Train에서 모델링될 ac_attend(누적 관람객) 를 사용한다.
# 인공신경망에 screen + attend 를 학습시킨다.
# 데이터는 사전에 분류한 Train을 이용하며, 은닉층은 5,3 개 를 사용한다.

model_result <- compute(model,Test[, c("screen","attend")])

# model_result 에는 사전에 학습된 model을 이용해 Test파일에 있는
# screen,attend 변수를 이용해 예측한다.

predict_attend <- model_result$net.result

# Test의 attend 예측값을 predict_attend 변수에 대입.

cor(predict_attend,Test$ac_attend)

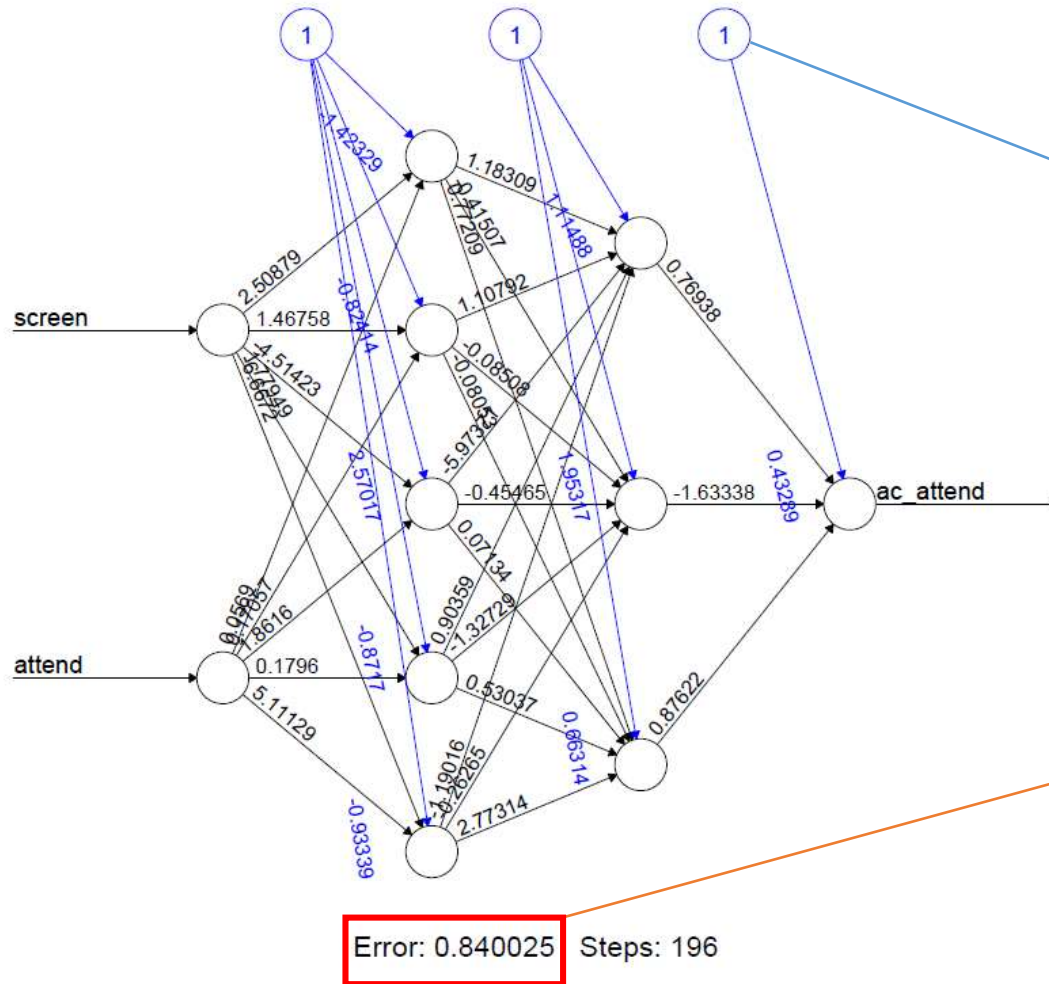
# predict_attend 예측값 과 실제 값 사이의 상관계수

plot(Test$ac_attend,predict_attend,xlab = "실제 값",ylab="예측 값",)
# x축을 예측값, y축을 실제값을 넣어 그래프를 그린다.
abline(a=0, b=1, col="red")
# y=x의 그래프를 그린다.

```

➔ 가장 유의미한 상관관계가 보였던 독립 변수를 ANN 모델링에 넣는 실험을 함.

3. 예측 모델링(1차 실험) : 시각화



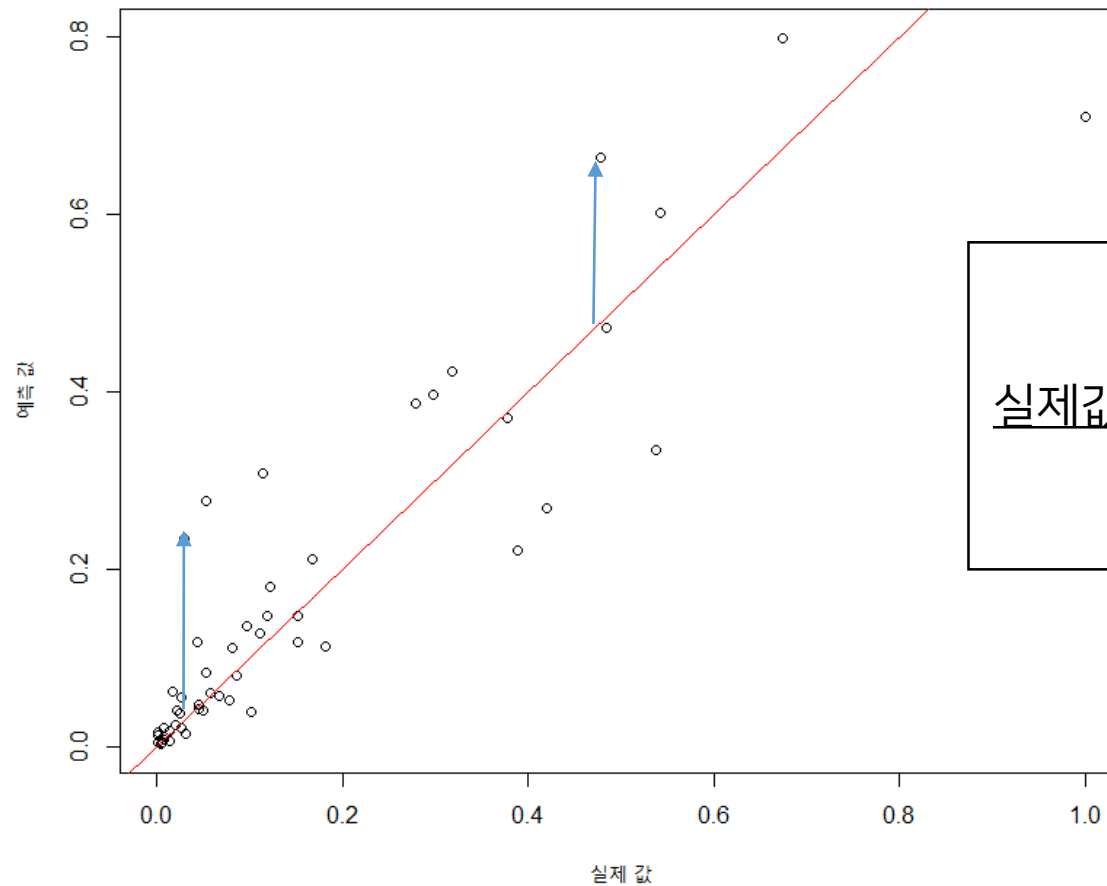
Bias(바이어스) 항목으로.
선형 경계의 절편의 값을 의미.
직선일 경우, y절편을 의미한다.

오차 제곱 합(SSE)

➔ 1차 실험에서 예측값과 실제값이 상당히 큰 차이를 보임

3. 예측 모델링(1차 실험) : 시각화

→ $y=x$ 지점 : 예측값과 실제값이 일치하는 지점



→ 시각화를 해보면,
실제값과 예측값 사이 상당한 격
차를 확인할 수 있음

3. 예측 모델링(2차 실험)

```
model <- neuralnet(ac_attend ~ screen + attend + dir_score ,
  data = Train , hidden=c(5,3))
# 사전에 분류한 Train에서 모델링될 ac_attend(누적 관람객) 를 사용한다.
# 인공신경망에 screen + attend + dir_score를 학습시킨다.
# 데이터는 사전에 분류한 Train을 이용하며, 은닉층은 5,3 개 를 사용한다.

model_result <- compute(model,Test[, c("screen","attend","dir_score")])
# model_result 에는 사전에 학습된 model을 이용해 Test파일에 있는
#screen,attend,dir_score 변수를 이용해 예측한다.

predict_attend <- model_result$net.result
#Test의 attend 예측값을 predict_attend 변수에 대입.

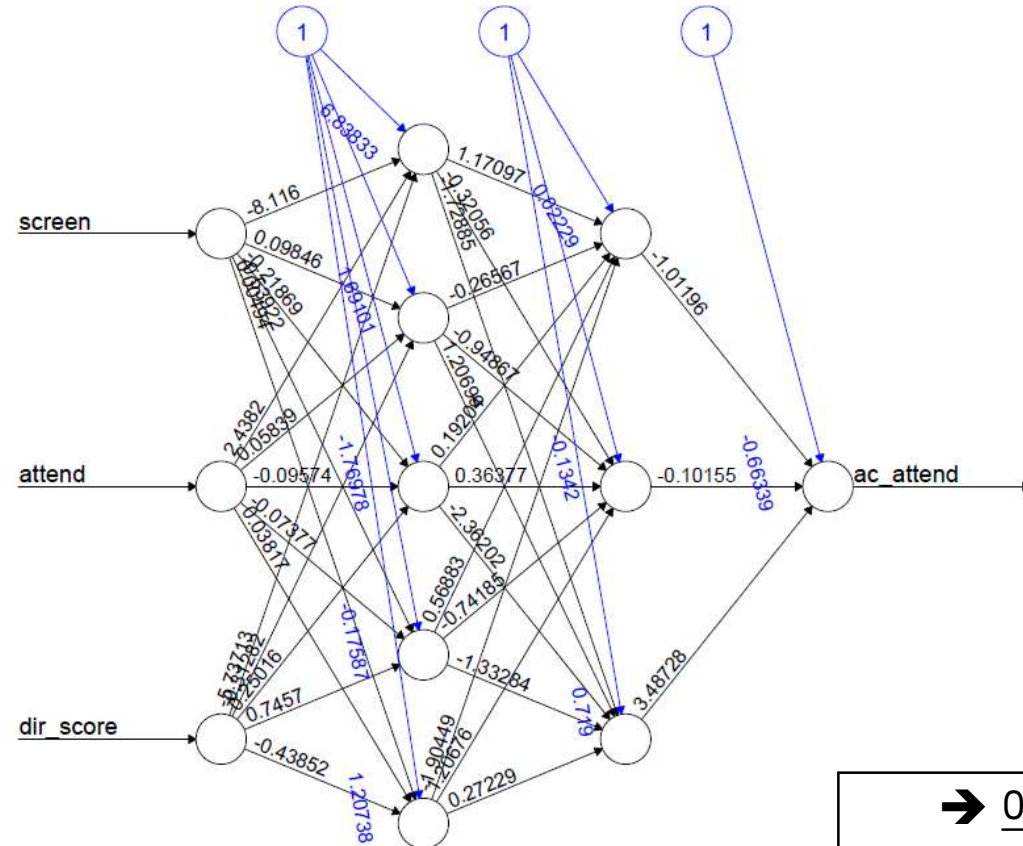
cor(predict_attend,Test$sac_attend)
# predict_attend 예측값 과 실제 값 사이의 상관계수

plot(Test$sac_attend,predict_attend,xlab = "실제 값",ylab="예측 값",)
# x축을 예측값, y축을 실제값을 넣어 그래프를 그린다.

abline(a=0, b=1, col="red")
# y=x의 그래프를 그린다.
```

→ 독립 변수 수정하여
2차 실험 실시

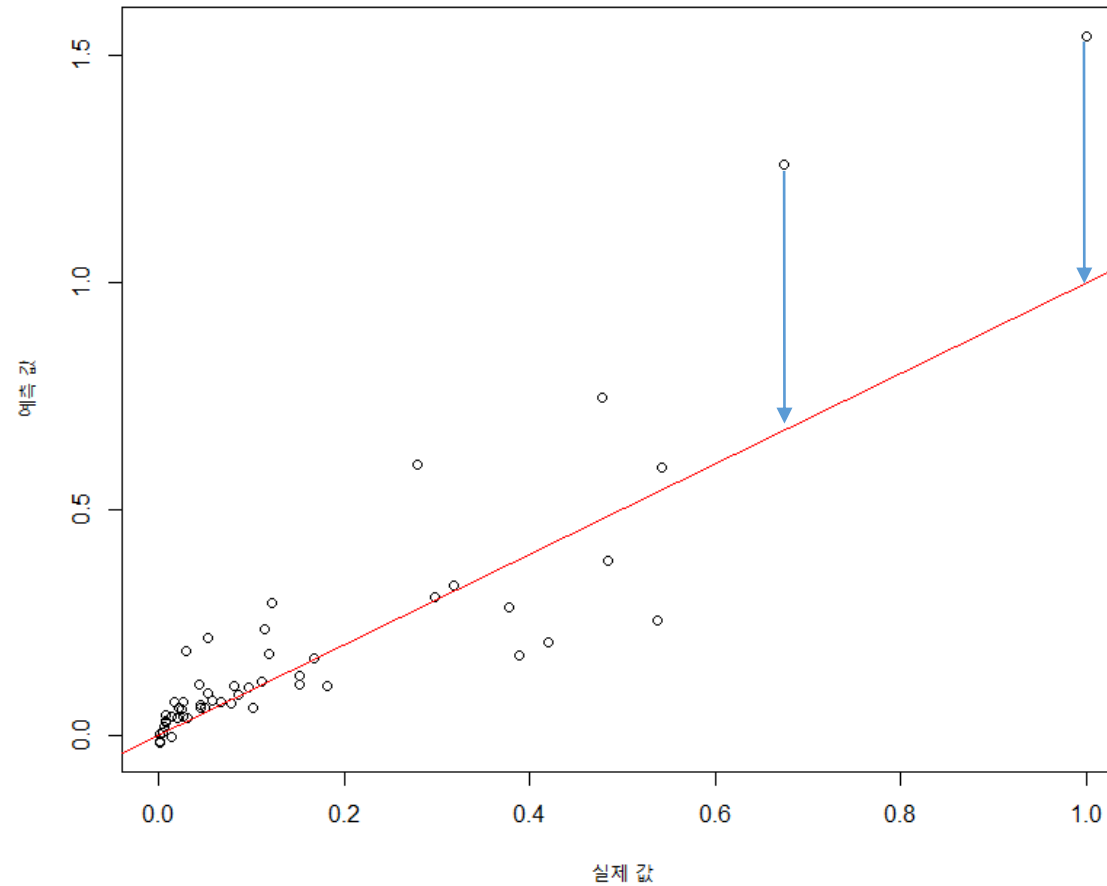
3. 예측 모델링(2차 실험) : 시각화



Error: 0.695987 Steps: 1711

→ 에러값 1차실험 대비
17.1% 감소

3. 예측 모델링(2차 실험) : 시각화



3. 예측 모델링(3차 실험)

```
model <- neuralnet(ac_attend ~ screen + attend + dir_score + deadline ,
  data = Train , hidden=c(5,3))
# 사전에 분류한 Train에서 모델링될 ac_attend(누적 관람객) 를 사용한다.
# 인공신경망에 screen + attend + dir_score + deadline + actor_2 를 학습시킨다.
# 데이터는 사전에 분류한 Train을 이용하며, 은닉층은 5,3 개 를 사용한다.

model_result <- compute(model,Test[, c("screen","attend","dir_score","deadline")])
# model_result 에는 사전에 학습된 model을 이용해 Test파일에 있는
#screen,attend,dir_score,deadline,actor_2 변수를 이용해 예측한다.

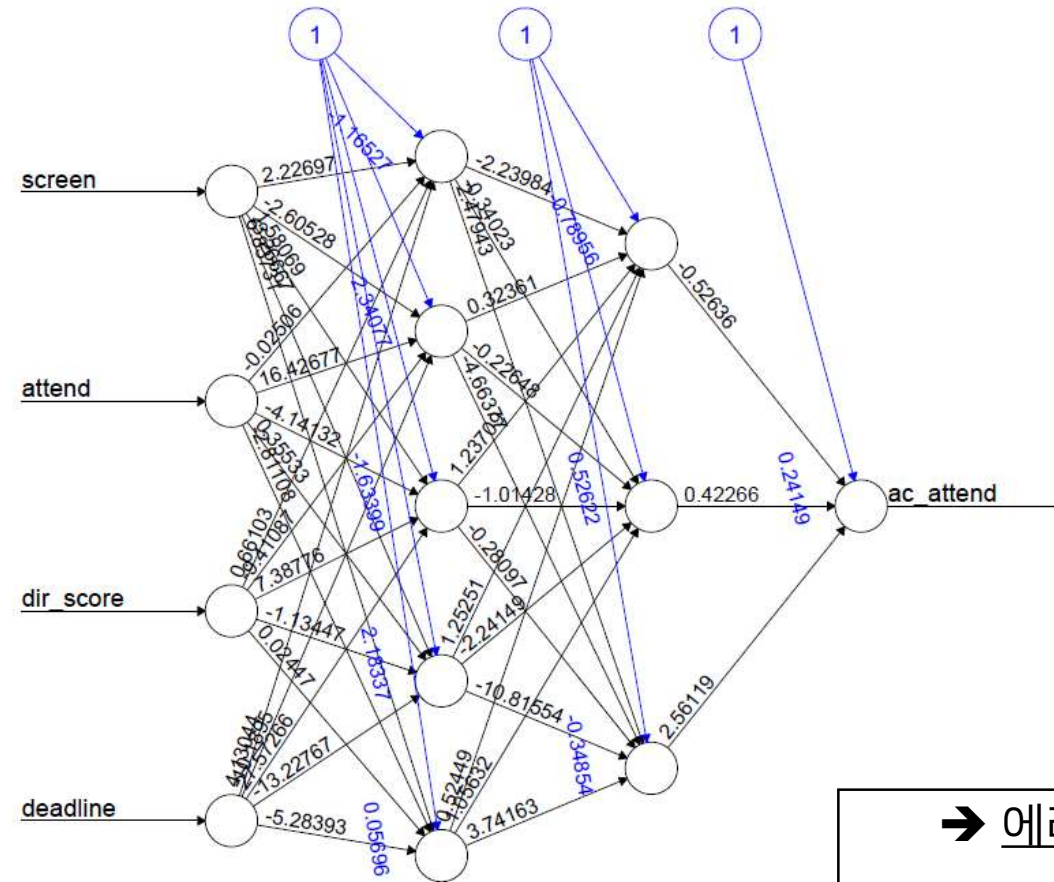
predict_attend <- model_result$net.result
#Test의 attend 예측값을 predict_attend 변수에 대입.

cor(predict_attend,Test$ac_attend)
# predict_attend 예측값 과 실제 값 사이의 상관계수

plot(Test$ac_attend,predict_attend,xlab = "실제 값",ylab="예측 값",)
# x축을 예측값, y축을 실제값을 넣어 그래프를 그린다.
abline(a=0, b=1, col="red")
# y=x의 그래프를 그린다.
```

→ 독립 변수 수정하여
3차 실험 실시

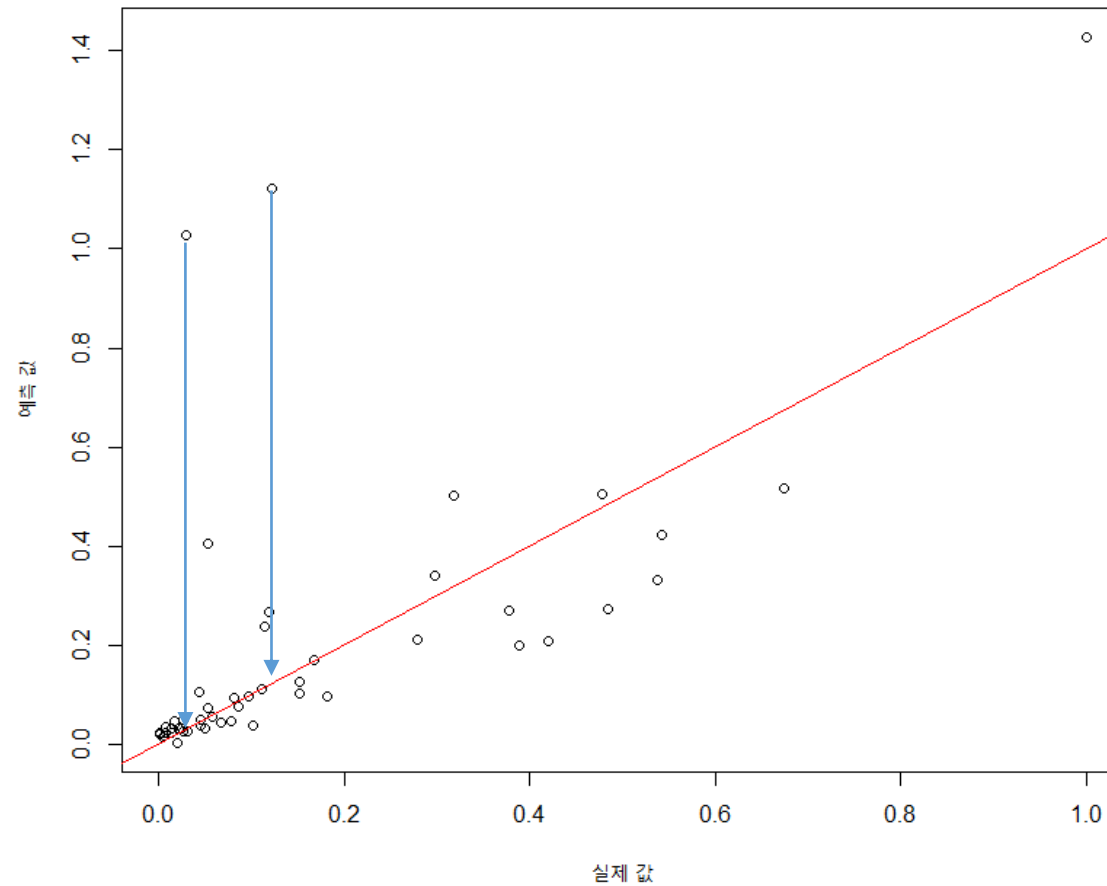
3. 예측 모델링(3차 실험) : 시각화



Error: 0.290197 Steps: 358

→ 에러값 1차실험 대비
58.3% 감소

3. 예측 모델링(3차 실험) : 시각화



→ 에러값은 상당히 감소하였으나, 여전히 Outlier값 존재

3. 예측 모델링(4차 실험)

→ 독립변수 수정하여 4차 실험 실시

```
model <- neuralnet(ac_attend ~ screen + attend + dir_score + deadline + actor_2 ,
  data = Train , hidden=c(5,3))
# 사전에 분류한 Train에서 모델링할 ac_attend(누적 관람객) 를 사용한다.
# 인공신경망에 screen + attend + dir_score + deadline + actor_2 를 학습시킨다.
# 데이터는 사전에 분류한 Train을 이용하며, 은닉층은 5,3 개 를 사용한다.

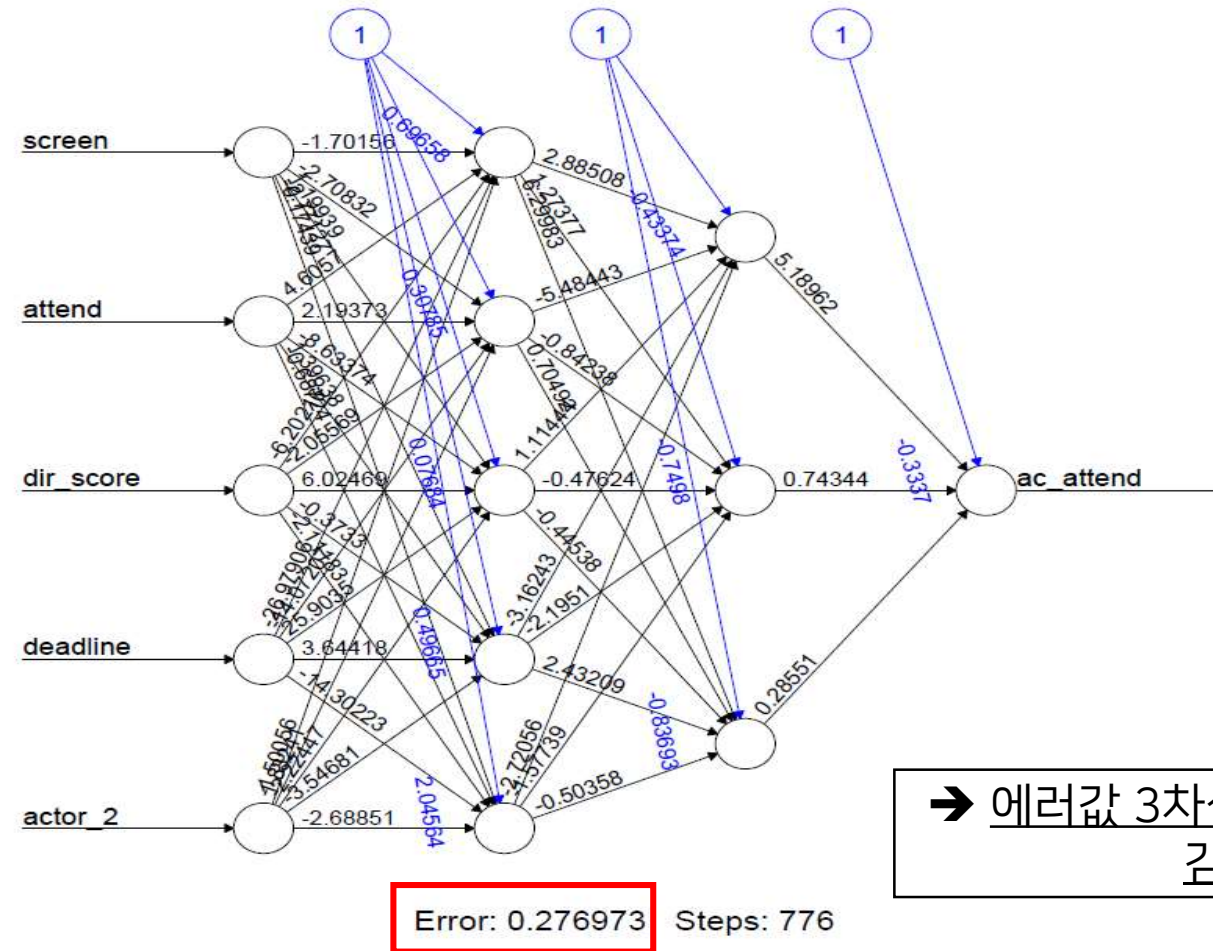
model_result <- compute(model,Test[, c("screen","attend","dir_score","deadline","actor_2")])
# model_result 에는 사전에 학습된 model을 이용해 Test파일에 있는
#screen,attend,dir_score,deadline,actor_2 변수를 이용해 예측한다.

predict_attend <- model_result$net.result
#Test의 attend 예측값을 predict_attend 변수에 대입.

cor(predict_attend,Test$ac_attend)
# predict_attend 예측값 과 실제 값 사이의 상관계수

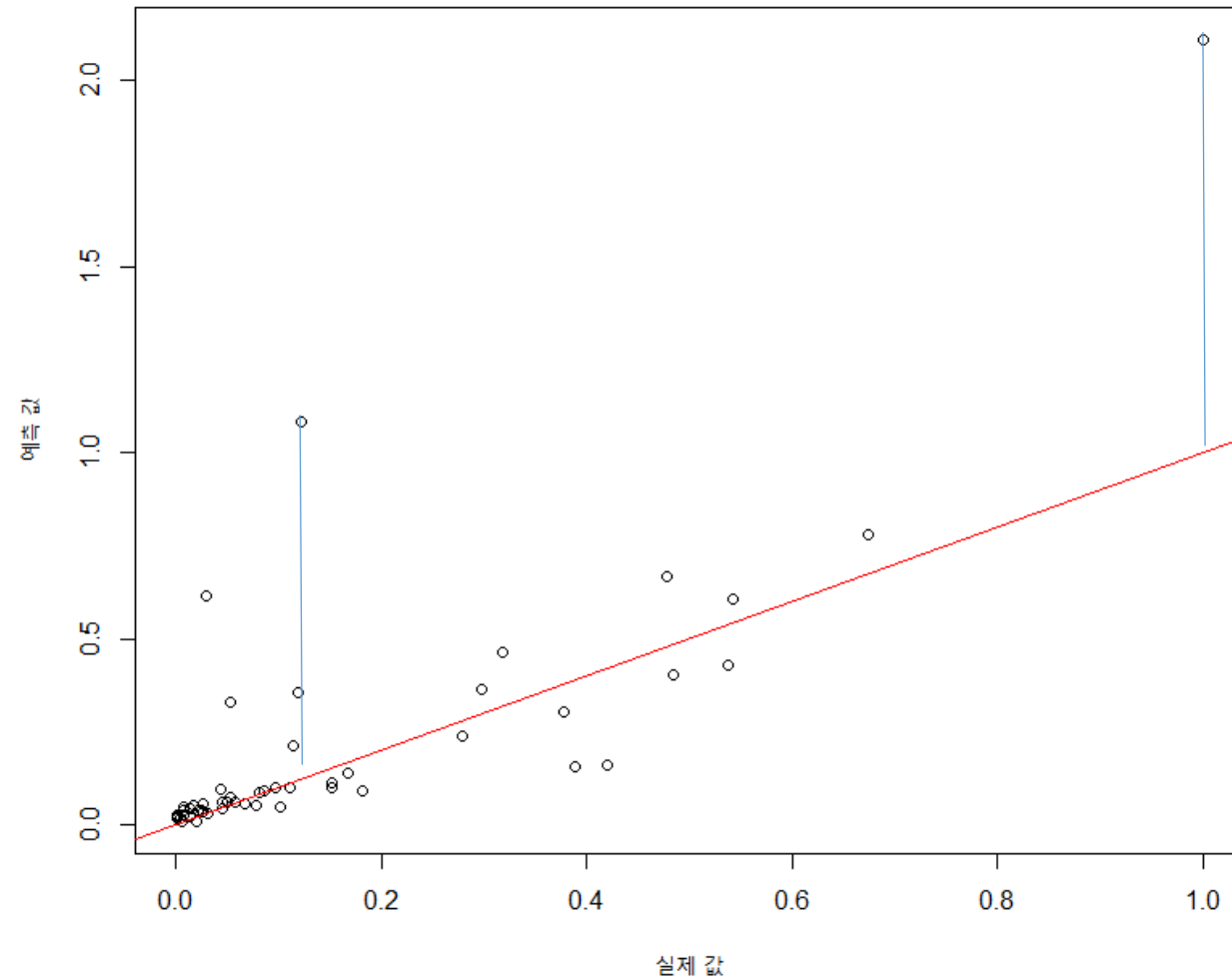
plot(Test$ac_attend,predict_attend,xlab = "실제 값",ylab="예측 값",)
# x축을 예측값, y축을 실제값을 넣어 그래프를 그린다.
abline(a=0, b=1, col="red")
# y=x의 그래프를 그린다.
```

3. 예측 모델링(4차 실험) : 시각화



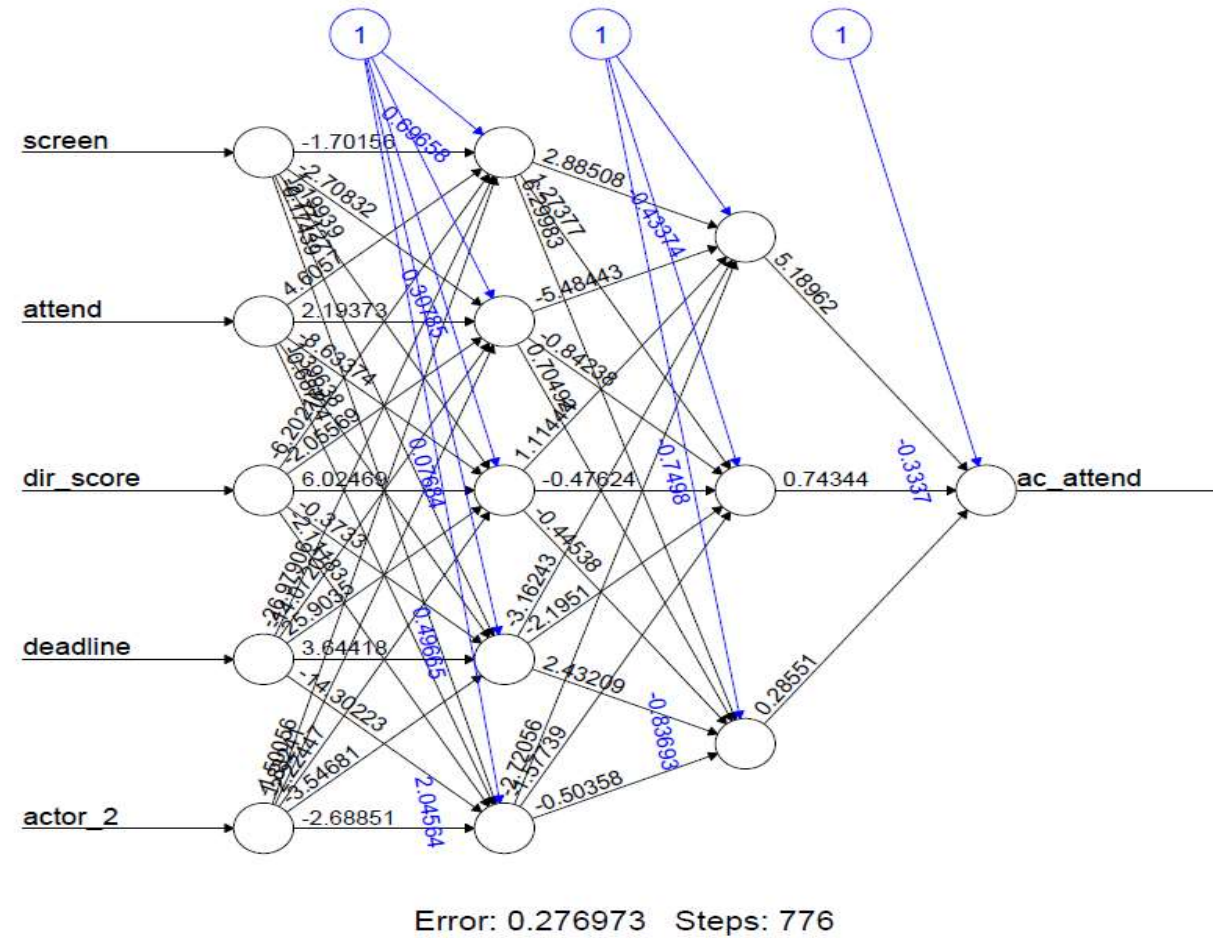
→ 에러값 3차실험 대비 4.5% 감소

3. 예측 모델링(4차 실험) : 시각화



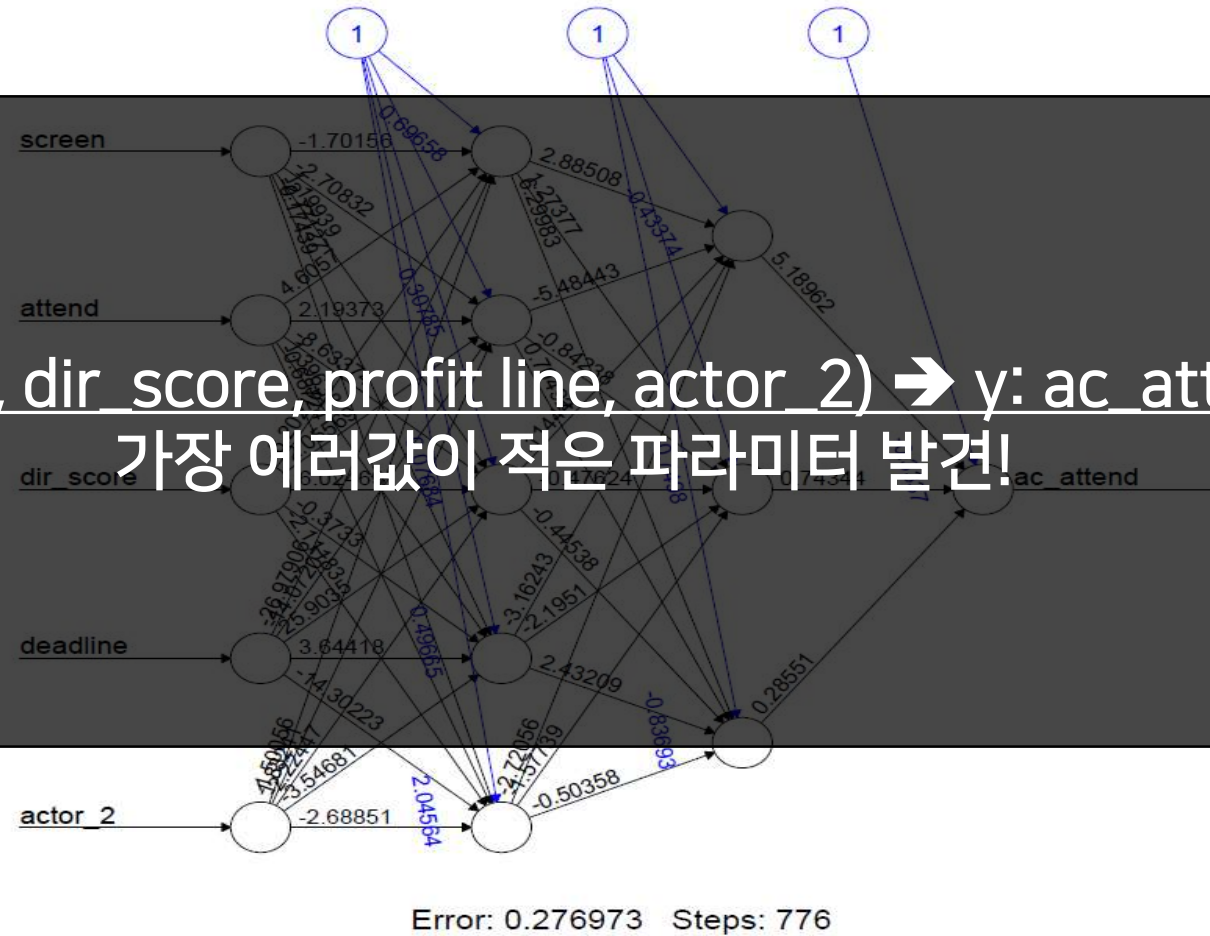
→ 예측값이 실제값에
수렴하는 추세를 보임

4. 결론



4. 결론

$F(x: \text{screen, attend, dir_score, profit line, actor_2}) \rightarrow y: \text{ac_attend}(\text{누적 관객수})$
가장 어려값이 적은 파라미터 발견!



2. Decision Tree / Random Forest

Decision Tree / Random Forest

데이터셋 분리 → 모델 설계 → 최적 독립변수 발견 작업 → 결과 시각화 → RMSE 분석

[분석 모델 정의]

Decision tree (의사결정 나무) 분석은 이상치와 노이즈에 큰 영향을 받지 않으며 모델의 간편한 시각화로 모델을 이해하고 분석하는데 큰 장점이 있는 분석법

[관련 패키지]

Rpart, party 등 여러 패키지를 사용하여 분석 가능

Rpart, party 패키지는 의사결정 나무를 그리는 논리, 시각화 형태에 따른 차이가 존재

1. Party 패키지 : 데이터 클렌징

1. 소스코드 화면

→ 모델링 전 데이터 정리 작업

```
library(caret)
library(randomForest)
library(party)
library(datasets)
library(data.table)
# test, train 데이터
Train<-read.csv("Train_korea.csv")
Test<-read.csv("Test_korea.csv")
Train<-Train[,c(7,8,16,17)]
Test<-Test[,c(7,8,16,17)]
str(Test)
str(Train)

#1. 모델 결측치 처리
Train2 <- Train
Test2 <- Test

Train2$ac_attend[is.na(Train2$ac_attend)] <-0
Train2$attend[is.na(Train2$attend)] <-0
Train2$screen[is.na(Train2$screen)] <-0
Train2$act_point[is.na(Train2$act_point)] <-0
Train2$profit_line[is.na(Train2$profit_line)] <-0

Test2$ac_attend[is.na(Test2$ac_attend)] <-0
Test2$attend[is.na(Test2$attend)] <-0
Test2$screen[is.na(Test2$screen)] <-0
Test2$act_point[is.na(Test2$act_point)] <-0
Test2$profit_line[is.na(Test2$profit_line)] <-0
```

#<< 원하는 변수만 불러온다.

1. Party 패키지 : 모델링

1. 소스코드 화면(전체)

```
#2. formula 생성 : Y변수 연속형
# -----

party_tree <- ctree(ac_attend~screen+attend+act_point,Train2)
plot(party_tree)

#3. 모델 그래프 시각화
party_pred <- predict(party_tree,Test2,type="response")
party_pred

#관객 예측 정확률 : 예측결과/실제결과
party_pred_per <- (1-(abs(party_pred - Testa))/Testa))*100
party_pred_per

#4. 모델링과 RMSE 지수

model<-randomForest(ac_attend~attend+screen,data=Train2,importance=TR
randomforest_pred<-predict(model, Test3)      #x 정의역 변수를 넣으면
RMSE<-sqrt(mean(Test3$ac_attend - randomforest_pred)^2)
RMSE|
```

→ 1-1) 의사결정 나무 모델 생성

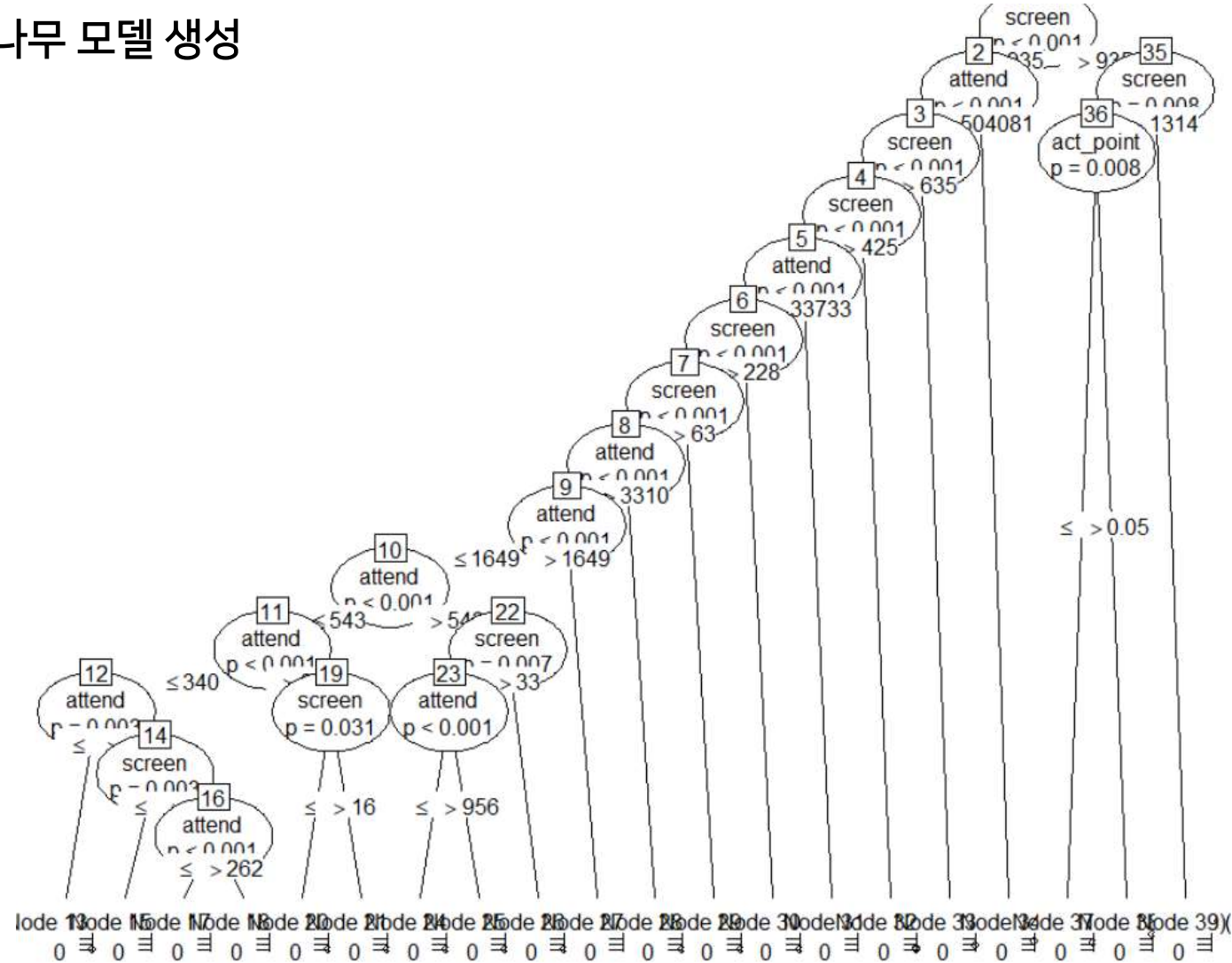
→ 1-2) 관객 예측 정확률 : 예측 결과 / 실제결과

→ 2-1) 랜덤 포레스트 모델 생성

→ 2-2) RMSE 계산

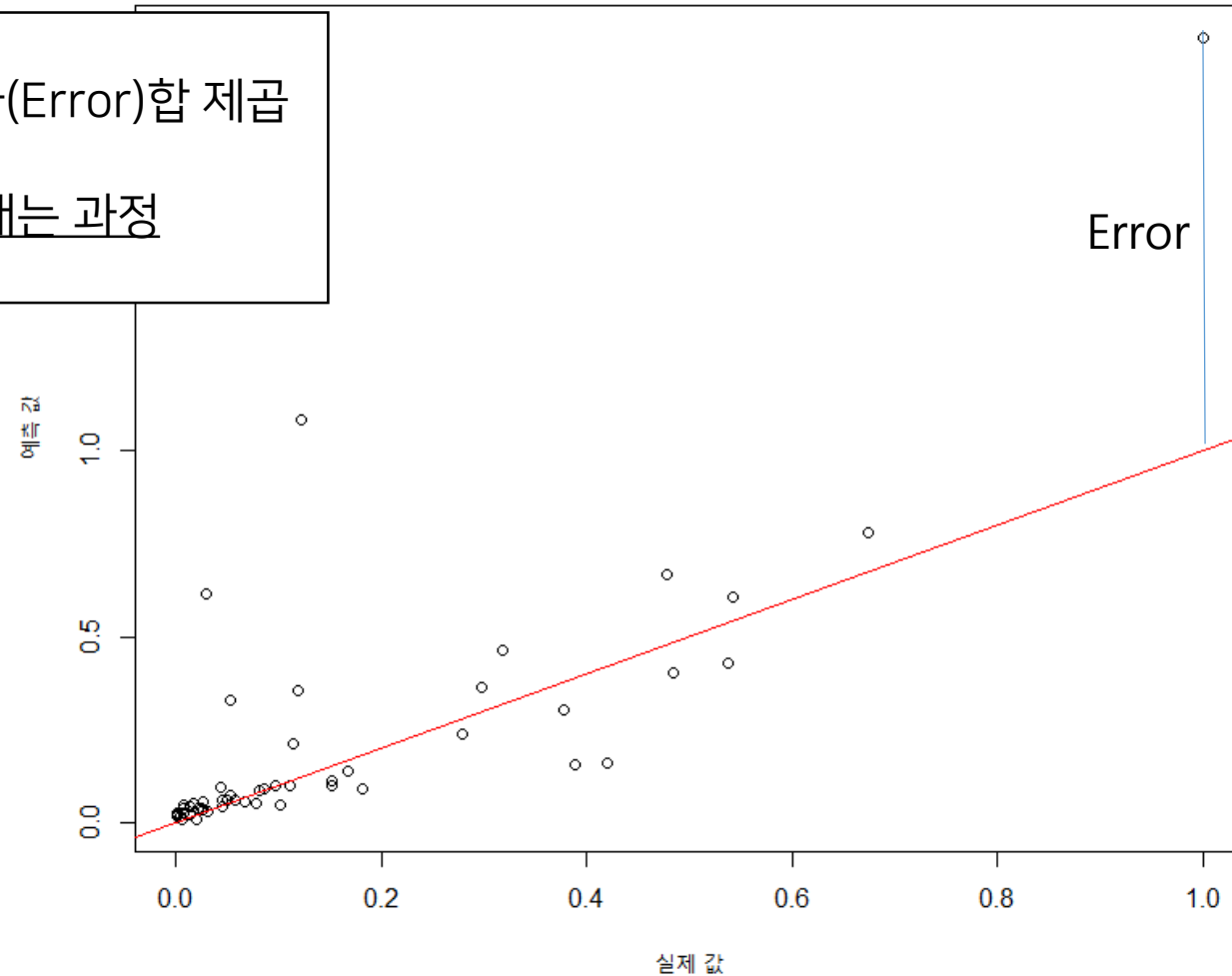
* 시각화

→ 1-1) 의사결정 나무 모델 생성



* RMSE 계산의 의미

실제값과 예측값 오차(Error)합 제곱
의 평균 중
가장 작은 값을 찾아내는 과정



1. Party 패키지 : 결론

→ 2-2) RMSE 계산 결과

[출력 결과]

```
> Test3
      ac_attend screen act_point  attend
[1,]   2675618   1124         0 1299451
> Test3<-as.data.table(Test3)
> randomforest_pred<-predict(model, Test3)
> randomforest_pred
      1
4315304
> RMSE<-sqrt(mean(Test3$ac_attend - randomforest_pred)^2)
> RMSE
[1] 1639686
```

→ Party 패키지에서 RMSE가 16만에서 최소값을 보임

1. Party 패키지 : 결론

➔ 실제 데이터로 예측 시뮬레이션

완성된 모델을 평가하기 위해 **영화_곤지암** 데이터를 대입

[출력 결과]

실제값	<pre>> Test3 ac_attend screen act_point attend [1,] 2675618 1124 0 1299451 > Test3<-as.data.table(Test3) > randomforest_pred<-predict(model, Test3) > randomforest_pred 1 4315304 > RMSE<-sqrt(mean(Test3\$ac_attend - randomforest_pred)^2) > RMSE [1] 1639686</pre>
예측값	

➔ 영화 '곤지암'의 실제 데이터 대입 결과, ac_attend(누적관객수)를 오차 1639686명으로 예측함 이는 실제값의 61.2%의 예측을 보임

충분한 예측률을 보이지 않아
다른 패키지로 2차 실험 진행

2. Rpart 패키지 : 데이터 클렌징

```
# test, train 데이터
Train<-read.csv("Train_korea.csv")
Test<-read.csv("Test_korea.csv")
Train<-Train[,c(7,8,16,17)]      #<< 사용할 변수만 불러온다.
Test<-Test[,c(7,8,16,17)]
str(Test)
str(Train)

library(caret)
library(rpart)
library(rpart.plot)
library(randomForest)

#4. 모델링과 RMSE 지수

#결측치 처리
Train2 <- Train
Test2 <- Test

Train2$ac_attend[is.na(Train2$ac_attend)] <-0
Train2$attend[is.na(Train2$attend)] <-0
Train2$screen[is.na(Train2$screen)] <-0
Train2$act_point[is.na(Train2$act_point)] <-0
Train2$profit_line[is.na(Train2$profit_line)] <-0

Test2$ac_attend[is.na(Test2$ac_attend)] <-0
Test2$attend[is.na(Test2$attend)] <-0
Test2$screen[is.na(Test2$screen)] <-0
Test2$act_point[is.na(Test2$act_point)] <-0
Test2$profit_line[is.na(Test2$profit_line)] <-0
model<-randomForest(ac_attend~.,data=Train2,importance=TRUE)
randomforest_pred<-predict(model, Test2)

RMSE<-sqrt(mean(Test2$ac_attend - randomforest_pred)^2)
RMSE
```

→ 모델링 전 데이터 정리 작업

2. Rpart 패키지 : 모델링

1. 소스코드 화면(계속)

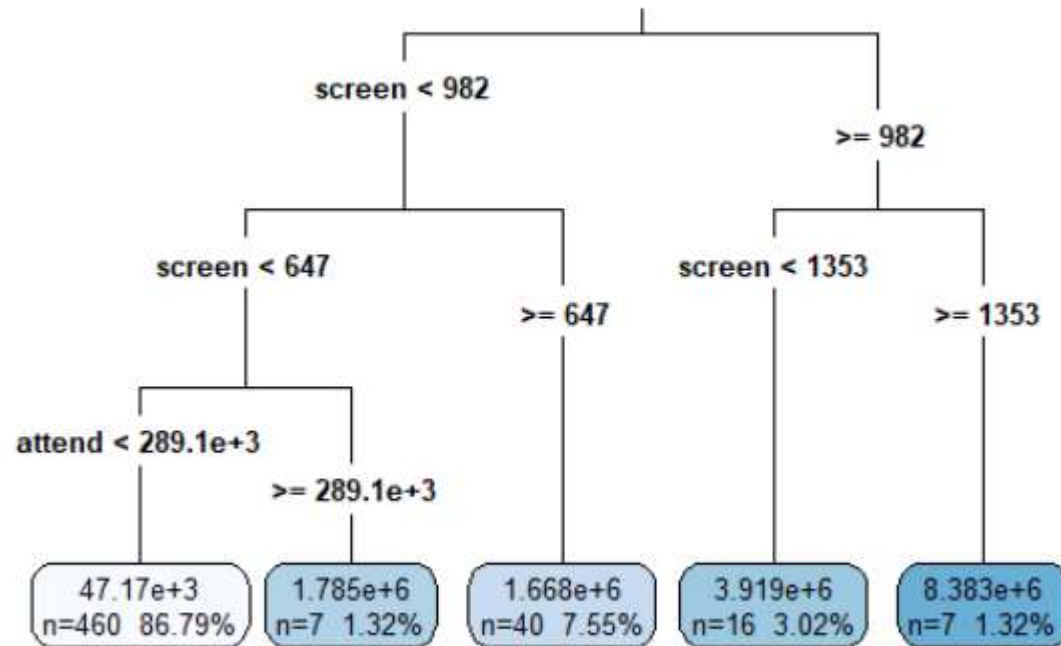
```
# 1. 모델링
m.rpart <- rpart(ac_attend~screen+act_point+attend,data=Train)

# 2. 시각화 model design
rpart.plot(m.rpart, digits=3)

#----digits는 숫자의 자리 수 fallen.leaves는 잎 노드의 정렬, type과 extra는 결정과 노드가 레이블
rpart.plot(m.rpart, digits=4, fallen.leaves = T, type=3, extra =101)
|
#3. 상관 관계 평가
p.rpart <-predict(m.rpart, Test)
summary(p.rpart)
cor(p.rpart, Test$ac_attend)

> cor(p.rpart, Test$ac_attend)
[1] 0.8417104
```


2. Rpart 패키지 : 시각화



➔ Rpart 패키지를 이용한 의사결정 나무에서는 이 불법적 해석으로 데이터를 분류해서 나아간다.

변수로는 `act_point`, `screen`, `attend` 를 사용하였다.

2. Rpart 패키지 : RMSE 계산 모델링

```
#5. 모델 평가 작업 - 정의역 변수 1개 데이터 값 대입 후 실제 누적관객수와 비교
# ex. 실제 영화인 검은사제들의 데이터값을 대입
Test3 <-c(1299451)
Test3 <-cbind(Test3,1124)
Test3 <-cbind(Test3,0)
Test3 <-cbind(Test3,2675618)
colnames(Test3)<-c("ac_attend","screen","act_point","attend")
Test3
Test3<-as.data.table(Test3)

randomforest_pred<-predict(model, Test3)
randomforest_pred

RMSE<-sqrt(mean(Test3$ac_attend - randomforest_pred)^2)
RMSE
```

→ RMSE 계산

2. Rpart 패키지 : 결론

→ RMSE 계산 결과

```
> Test3
  ac_attend screen act_point attend
[1,] 2675618 1124      0 1299451
> Test3<-as.data.table(Test3)
> randomforest_pred<-predict(model, Test3)
> randomforest_pred
      1
3230677
> RMSE<-sqrt(mean(Test3$ac_attend - randomforest_pred)^2)
> RMSE
[1] 555059.1
```

→ RMSE 값이 Rpart패키지에서 55만에서 최소값을 보임

2. Rpart 패키지 : 결론

➔ 실제 데이터로 예측 시뮬레이션

완성된 모델을 평가하기 위해 **영화_곤지암**들의 데이터를 대입

실제값

예측값

```
> Test3
  ac_attend screen act_point attend
[1,] 2675618  1124         0 1299451
> Test3<-as.data.table(Test3)
> randomforest_pred<-predict(model, Test3)
> randomforest_pred
      1
3230677
> RMSE<-sqrt(mean(Test3$ac_attend - randomforest_pred)^2)
> RMSE
[1] 555059.1
```

영화 '곤지암' 의 실제 데이터 대입 결과, ac_attend(누적관객수)를 약 555059명
오차로 예측함 실제값의 **82.8% 예측률을 보임**

전문가 인터뷰

전문가 인터뷰



“82.8% 예측은 준수하나, 알고리즘의 파라미터를 조정하면 더 높은 예측률을 보일 수 있을 것”

“예측모델의 핵심은 사용되는 독립변수”

“알고리즘 별 파라미터 선택에 만전을 기해야함”

박동연 (2017 빅콘테스트 - 영화 흥행 예측 부분 대상 수상자) ,
민다빈(서울시 빅데이터 연구소 SBLI 소속 연구원)

3. K-Nearest Neighbor(FNN)

1. FNN 모델을 쓴 이유

데이터셋 분리 → 모델 설계 → 결과 시각화 → RMSE 분석

[분석 모델 정의]

패턴 인식을 위해, 분류나 회귀에 사용되는 KNN(K- 최근접 이웃 알고리즘)은 지역적으로 근처에 다다르고 모든 계산이 분류될 때 까지 연기되는 간단한 기계 학습 알고리즘에 속한다.

데이터를 분류해, 분류된 클러스터만을 출력하는 기능만 수행하는 KNN에서 확장해, k개의 최근접 이웃이 가진 값들의 평균으로, KNN 회귀에서 모델의 예측값 출력

[관련 패키지]

Knn.reg(FNN) 사용

2. K-Nearest Neighbor(FNN) : 모델 설계

```
> Org<-read.csv("Go.csv")
> Data<-Org[c(1,3,4,5,6)]
> str(Data)
'data.frame': 1781 obs. of 5 variables:
 $ name      : Factor w/ 1780 levels "007 스펙터",
 ..
 $ attend    : int  3548260 3255874 2210275 191391
 $ ac_attend : int  14410537 3848108 6879822 72311
 $ screen    : int  1912 232 728 1299 1113 1314 14
 $ season    : num  1 0.283 0.283 1 0.775 ...
```

→ Fnn 모델은 반드시 인자가 int형이어야함

```
> set.seed(2018) #랜덤 seed 설정
> RandomIndex<-sample(x=1:nrow(Data),size=round(0.7*nrow(Data)),replace=F) # 훈련,테스트 (7:3) 분리
> Train<-Data[RandomIndex,]
Error in `[.data.frame'`(Data, RandomIndex, ) :
  object 'RandomIndex' not found
> Train<-Data[RandomIndex,]
> Test<-Data[-RandomIndex,]
```

```
model<-knn.reg(Train,Test,Train$ac_attend,35)
#모델 생성 knn.reg(훈련,테스트,y,k개수 {일반적으로 훈련 데이터 원소의 제곱근} )
```

→ 모델 생성 (최적의 k 포함)

3. 모델 성능향상을 위한 feature 발견

name	month	attend	ac_attend	screen	season	post	like
신과함께-죄와 벌	12	3548260	14410537	1912	1	8,365	156
남한산성	10	3255874	3848108	232	0.2826543	144790	460
범죄도시	10	2210275	6879822	728	0.2826543	117390	445
1987	12	1913914	7231189	1299	1	666831	965
터미네이터 제니시스	7	1894340	3240370	1113	0.7752099	2351	44
꾼	11	1655893	4018035	1314	0.2537844	71877	203
강철비	12	1643489	4452515	1426	1	46596	1845
부산행	7	1436163	11565056	1788	0.7752099	226486	1397
토르: 라그나로크	10	1356454	4853554	1456	0.2826543	32042	915
테이큰 3	1	1196326	2006561	616	0.3185174	3180	33
23 아이덴티티	2	1178815	1675719	952	0.4771435	13378	143
저스티스 리그	11	1141912	1786383	1308	0.2537844	25310	90

KNN 예측 모델 실험에서는 성능을 향상시키기 위해 2가지 feature를 시도

➔ { post(인스타그램 게시물 수) , like(인기글 평균 좋아요 수) }

➔ Season : 월 별 누적관객수의 총합의 정규화(normalization)

3. 모델 성능향상을 위한 feature 발견 : (1) 인스타그램

```
> for (i in 1:1781) {
+   url<-paste(url_base1,search[i,1],url_base2)
+   movie<-read_html(url)
+   content<-html_nodes(movie,".search_tag")
+
+   content1<-html_nodes(content,".td_col.result_td01")
+   content1<-gsub("<.+?>|\t","",content1)
+   content1<-gsub("\r","",content1)
+   content1<-gsub("\n","",content1)
+   title<-c(title,content1) # 제목 뽑기
+
+   content2<-html_nodes(content,".td_col.result_td02")
+   content2<-gsub("<.+?>|\t","",content2)
+   post<-c(post,content2) # 누적게시물 수 뽑기
+
+   content3<-html_nodes(content,".td_col.result_td03")
+   content3<-gsub("<.+?>|\t","",content3)
+   content3<-gsub("\n","",content3)
+   like<-c(like,content3) # 인기글 평균 좋아요 수 뽑기
+
+   print(i)
+ }
[1] 1
[1] 2
[1] 3
```

→ 스타태그(startag.io)에서
title을 검색하면 나오는 결과 출력
(누적 게시물 수, 인기글 평균 좋아요 수)

title	post	like
#캡틴아메리카시빌워	14,694	505
#신과함께죄와벌	8,365	156
#검사외전	53,337	190
#부산행	226,486	1,397
#트랜스포머최후의기사	3,528	271
#스파이더맨홈커밍	44,259	1,336
#미녀와야수	158,153	1,158
#암살	55,434	298
#닥터스트레인지	62,530	5,375
#마스터	97,359	241
#곡성	146,277	164
#토르라그나로크	32,042	915

2. 모델 성능향상을 위한 feature 발견 : (1) 인스타그램

[한계점 발견]

1. 어느 정도 overground인 영화만 검색
2. 좋아요수 의도적 조작의 흔적 발견 : 영화 시청과 관련 없이 관여 (e.g. 아이돌 팬)

신과함께-인과 연 (Along with the Gods: The Last 49 Days, 2017) 상영중

관람객 ★★★★★ 8.65 (8,320) | 기자-평론가 ★★★★★ 6.31 (13) 평점주기

판타지, 드라마 | 2018.08.01 개봉 | 141분 | 한국 | 12세 관람가
 감독 김용화
 관객수 9,859,650명
 내용 천 년 동안 48명의 망자를 환생시킨 저승 삼차사, 한 명만 더 환생... 더보기

#신과함께_인과연	7,853	1,106
#신과함께2	70,036	295
#신과함께인과연	33,279	245

1000만 영화 <- like 총 1646

여중생A (Student A, 2018)

관람객 ★★★★★ 7.39 (138) | 기자-평론가 ★★★★★ 6.00 (2) 평점주기

드라마 | 2018.06.20 개봉 | 114분 | 한국 | 12세 관람가
 감독 이경섭
 관객수 97,851명
 내용 취미는 게임, 특기는 글쓰기인 여중생 '미래'가 가장 좋아하는 ... 더보기
 관련정보 연판도서, 원작만화, 여중생A-특별편

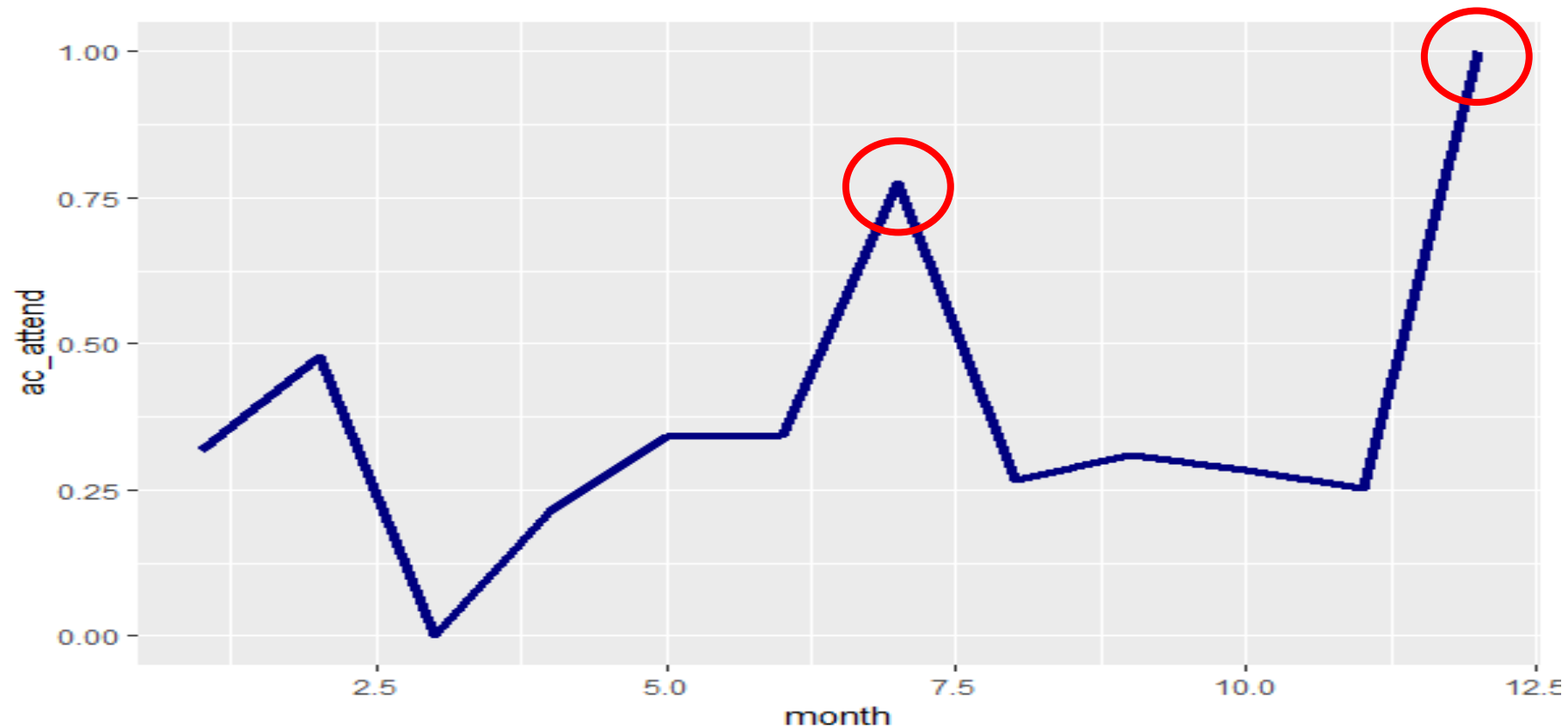
#여중생a	10,113	1,314	1443.32	0.01	0	0.48
-------	--------	-------	---------	------	---	------

10만 영화 <- like 1314 ??

→ 파라미터 최종 기각 결정

3. 모델 성능향상을 위한 feature 발견 : (2) season vs 관람객

→ Season 시각화 : 월 별 누적관객수를 모두 모아 정규화(normalization) 진행

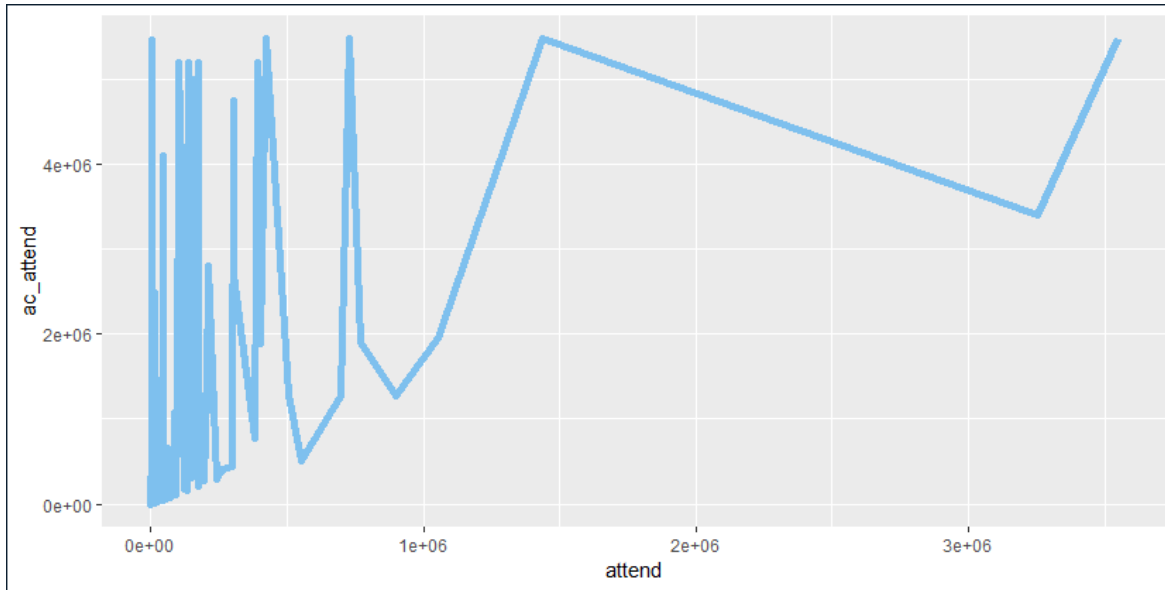


→ 여름방학시즌과 크리스마스 시즌에 상당히 편향되어있음을 발견

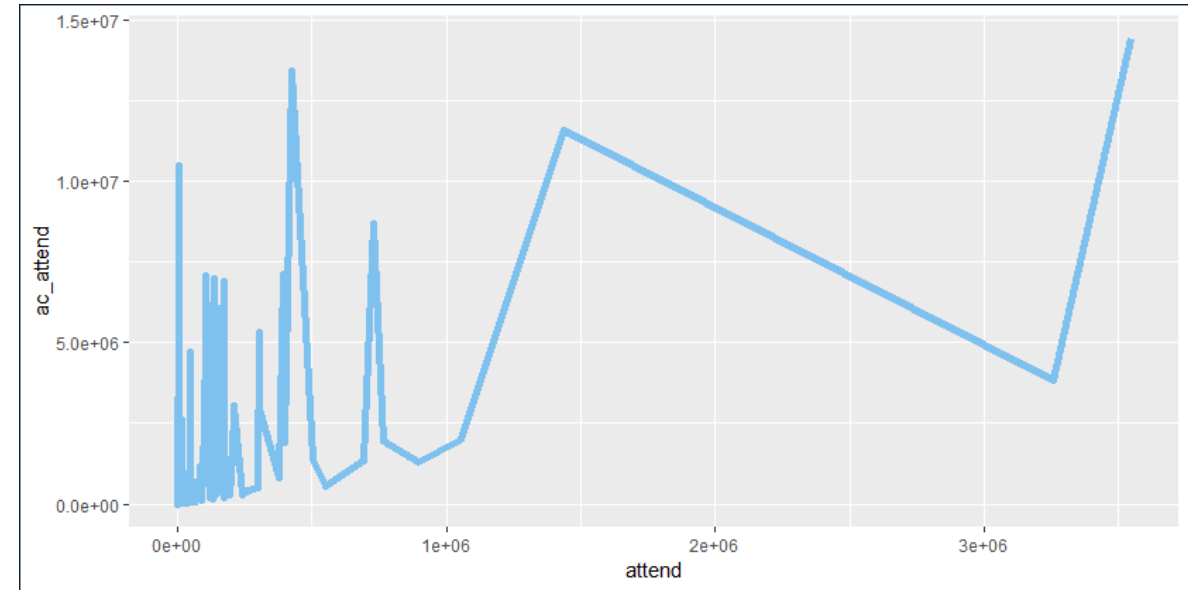
3. 모델 성능향상을 위한 feature 발견 : (3) 개봉 후 1주일 관람객

➔ 예측모델과 실제값과 거의 유사한 형태로 매우 밀접한 feature로 확인

Pred_Model



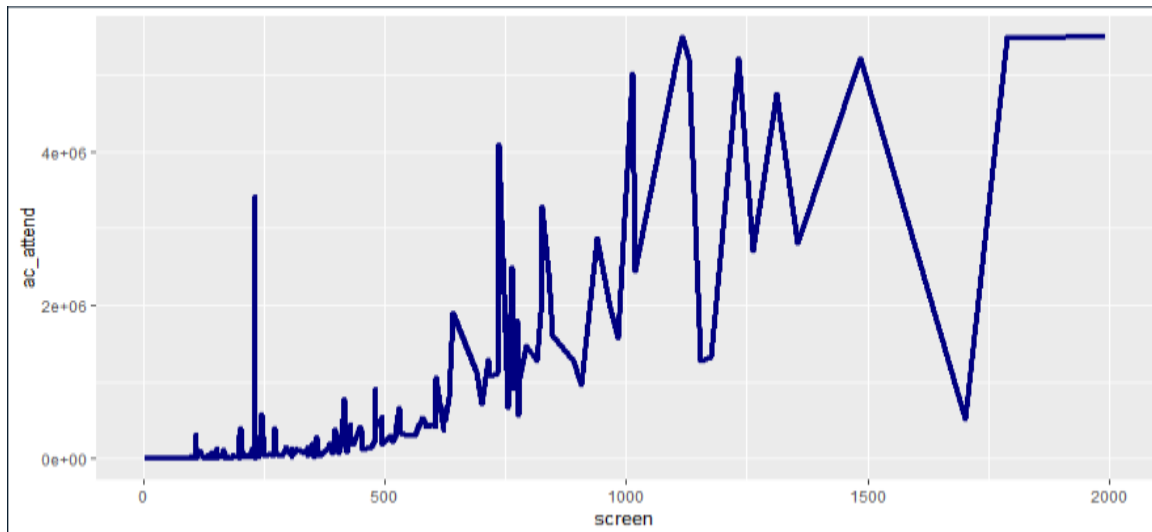
Actual



3. 모델 성능향상을 위한 feature 발견 : (4) screen 수

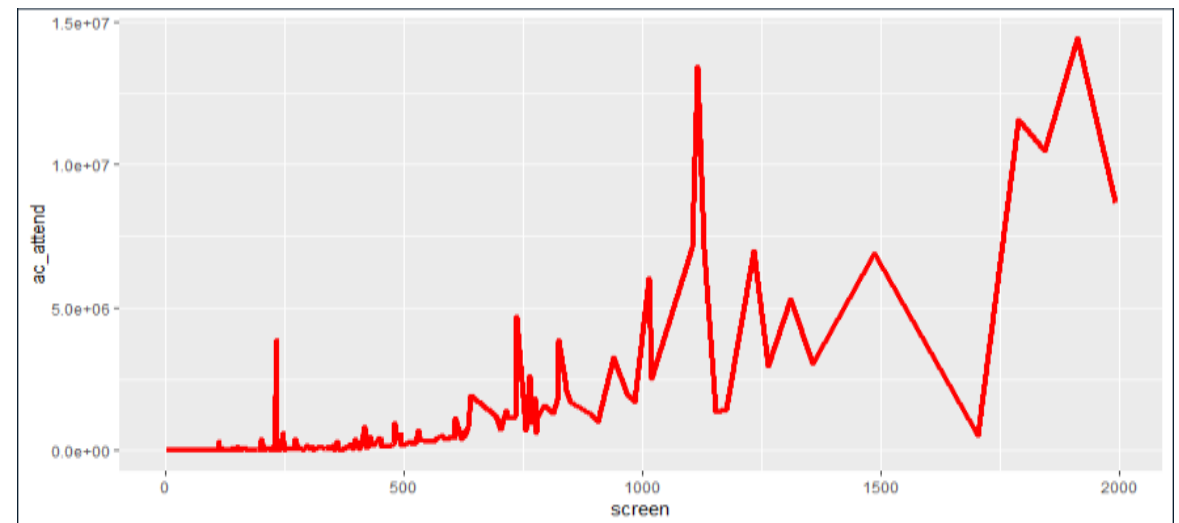
Pred_Model

```
> first<-cbind(Test$screen,model$pred)
> first<-as.data.table(first)
> colnames(first)<-c("screen","ac_attend")
> ggplot(data=first,aes(x=screen,y=ac_attend))+geom_line(size=1.5,color="navy")
```



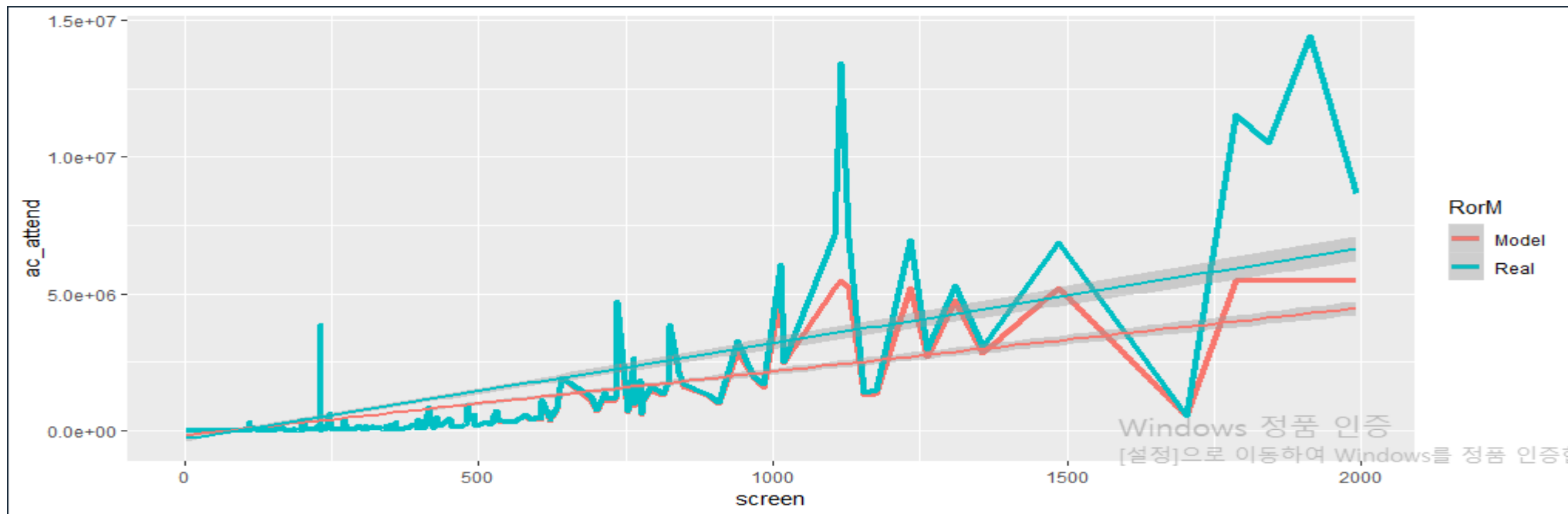
Actual

```
> second<-cbind(Test$screen,Test$ac_attend)
> second<-as.data.table(second)
> colnames(second)<-c("screen","ac_attend")
> ggplot(data=second,aes(x=screen,y=ac_attend))+geom_line(size=1.5,color="red")
```



3. 모델 성능향상을 위한 feature 발견 : (4) screen 수

```
> RorM<-rep("Model",nrow(first))
> str(RorM)
chr [1:534] "Model" "Model" "Model"
> RorM<-rep("Real",nrow(second))
> first<-cbind(first,RorM)      second<-cbind(second,RorM)
> colnames(first)<-c("screen","ac_attend","RorM")
> colnames(second)<-c("screen","ac_attend","RorM")
> ans<-rbind(first,second)
> ggplot(data=ans,aes(x=screen,y=ac_attend,color=RorM,group=RorM))+geom_line(size=1.5)+stat_smooth(method=lm)
```



4. 결론

1) RMSE 계산

모델과 실제 값의 차이를 비교 (공모전 제출 데이터)

```
> RMSE<-sqrt(mean( (Test$ac_attend-model$pred)^2 ))  
> RMSE  
[1] 657310.6
```

→ FNN 알고리즘 RMSE 65만에서 최소값을 보임

4. 결론

2) 성능 측정을 위해 2018년 개봉 영화 곤지암 입력.

```
> predict<-knn.reg (Train,gon,Train$ac_attend,35)
> predict
Prediction:
실제값 [1] 2578148
예측값 > gon$ac_attend
[1] 2675618
> 2675618 - 2578148
[1] 97470
```

96%의 예측률 기록

5. 한계점

```
head(Test$ac_attend, 10)
[1] 14410537 3848108 11565056 2007657 1278871 1928605 8676837 1329664 551878 1382490
head(model$pred, 10)
[1] 5484300.3 3411514.3 5484300.3 1961194.1 1278951.7 1906317.8 5484300.3 1279238.6 515833.6 1290496.1
```

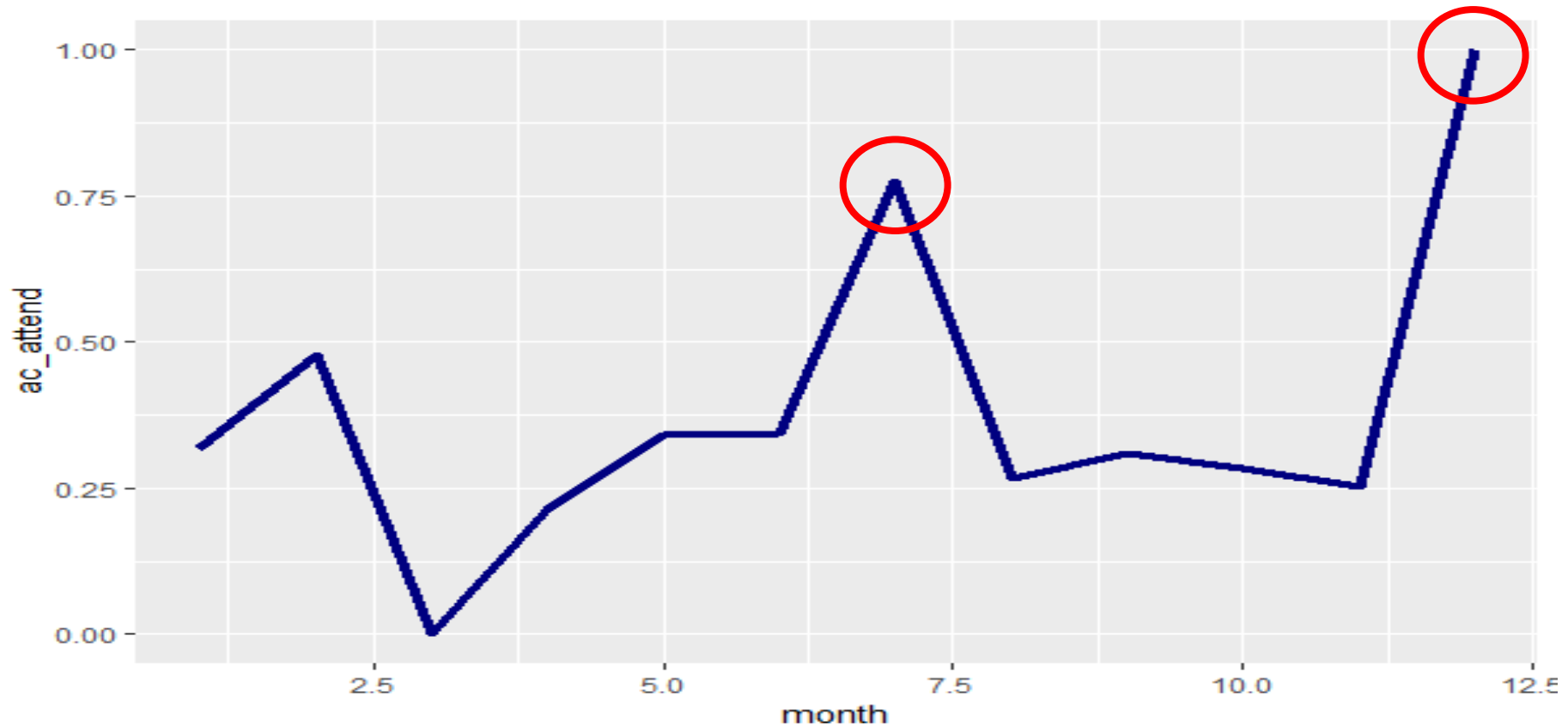
“오차의 평균은 65만 수준으로 상당한 예측력”

0.001%의 1500만 이상 메가 히트작의 경우,
테스트 데이터가 매우 희소하기 때문에
예측률이 다소 떨어지는 한계를 보임.

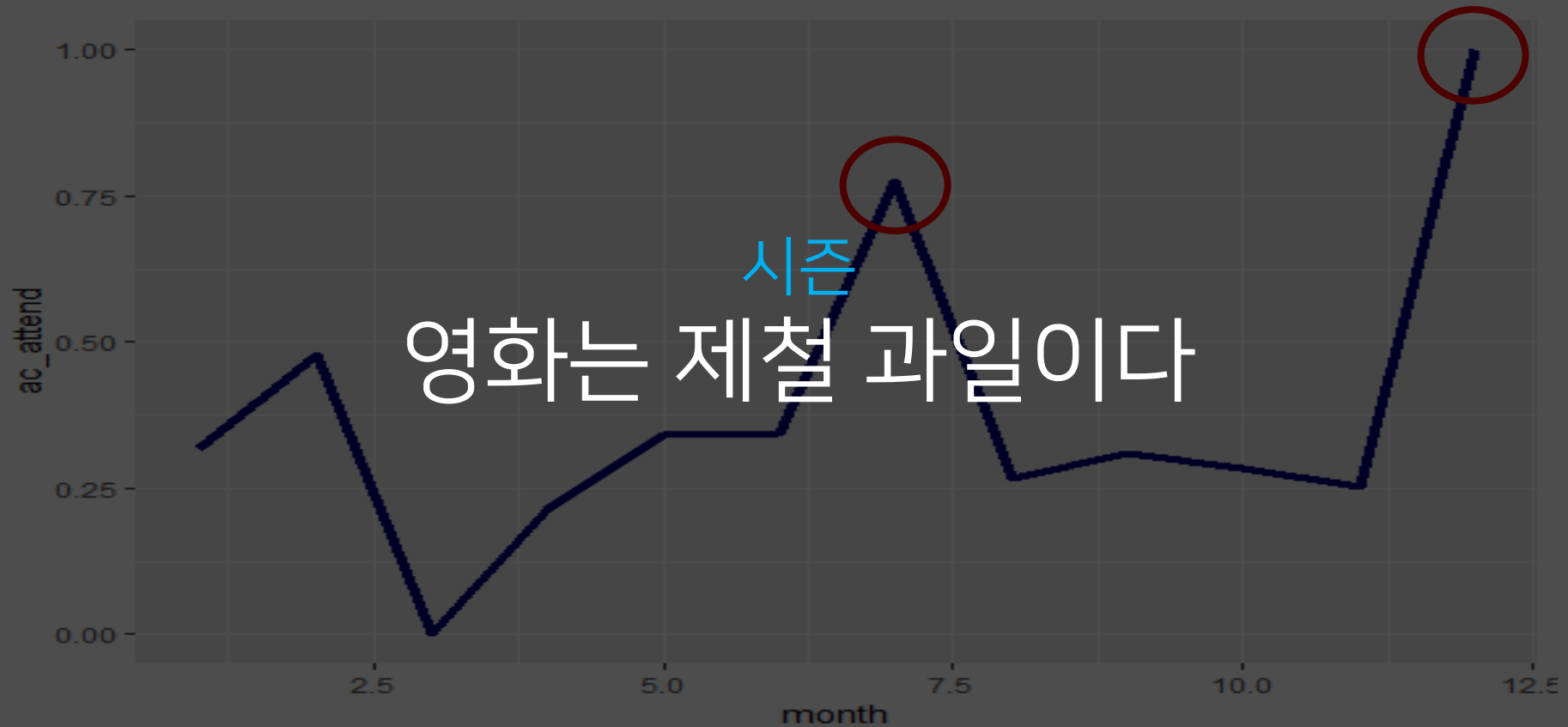
이 결과는 무슨 의미를 가지는가?

KNOWLEDGE

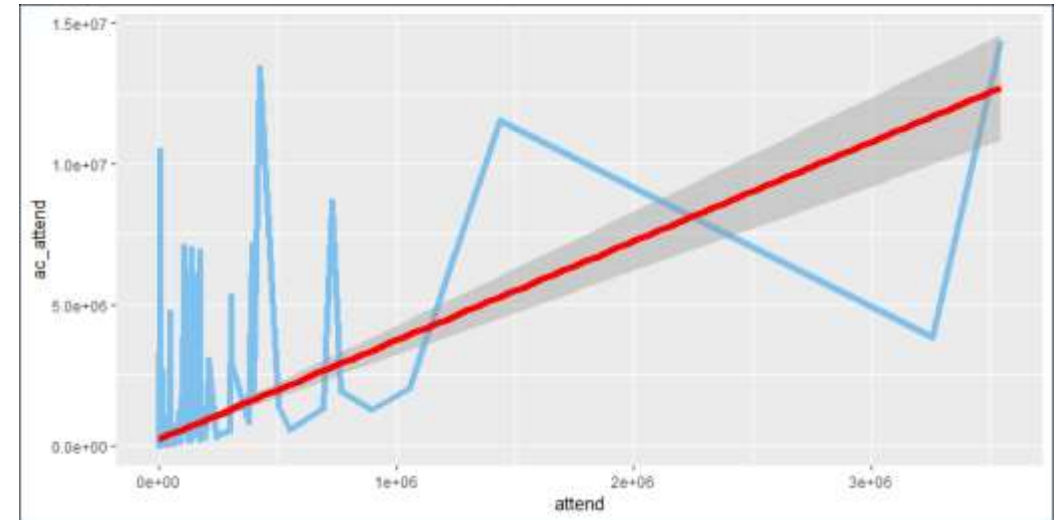
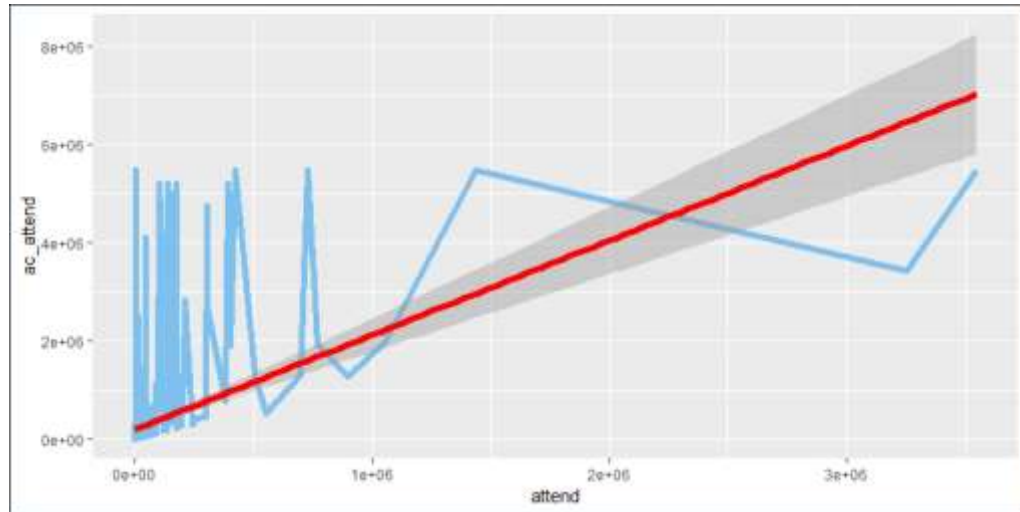
96%의 예측률을 보이게 된 요인 : (1) season



96%의 예측률을 보이게 된 요인 : (1) season



96%의 예측률을 보이게 된 요인 : (2) 개봉 후 첫 주 관람객





개봉 후 1주
사랑도, 영화도,
결국 첫인상

96%의 예측률을 보이게 된 요인 : (3) screen 수

