IT파이썬. 딥러닝

딥러닝

강사 이수현

어떤 영화가 흥행할까?

앙상블 모델의 학습 방법을 설명할 수 있다.
대표적인 랜덤 포레스트 모델을 생성하고 데이터를 분류할 수 있다.

어떤 영화가 흥행할까?

이번 활동에서는 **랜덤 포레스트**를 이용하여 흥행할 영화를 분류해 봅니다.

100만 명의 관객을 동원한 영화의 흥행 요소들을 파악하면 다른 영화의 흥행 여부를 미리 점쳐 볼 수 있을까요?

문제 정의하기	흥행할 영화를 미리 알아낼 수 있을까요?
데이터 불러오기	영화진흥위원회 자료를 이용한 DACON 영화 데이터셋 불러오기
데이터 처리하기	• 데이터셋 확인하기 • 데이터 결측치를 중간값으로 채우기
모델 학습하기	랜덤 포레스트로 학습하기
모델 테스트 및 평가하기	테스트 데이터로 평가하기

랜덤 포레스트의 이해

랜덤 포레스트란?

의사결정트리(Decision Tree)

- 데이터셋을 기반으로 예측을 위한 단일 트리 구조 생성
- 정보 이득(Information Gain) 또는 엔트로피(Entropy)를 이용한 질문 선택

의사결정 트리의 한계

- 단일 트리만을 사용하여 예측하는 불안감 존재

랜덤 포레스트의 도입

- 트리 여러 개를 결합하여 예측 성능 향상
- 여러 트리의 활용을 통한 정확도와 안정성 제고
- 앙상블 학습의 대표적인 모델
- 레오 브레이먼에 의해 개념 정리

랜덤 포레스트의 이해

앙상블 학습

앙상블 학습(Ensemble Learning) 소개

- 여러 모델의 예측 결과를 종합하여 더 정확한 예측 도출
- 약한 모델 여러 개를 조합하는 방식 >
- '집단 지성'의 힘: 여러 명의 지성 한 명의 전문가

랜덤 포레스트의 이해

앙상블 학습

랜덤 포레스트

- 앙상블 기법 중 대표적인 방법
- 단일 트리(Decision Tree) 대신 여러 트리의 '숲' 활용
- 여러 나무를 통한 더욱 강력한 예측 성능 제공

◀ 트리와 숲(포레스트)

랜덤 포레스트의 이해

앙상블 학습

랜덤 포레스트

- **핵심 기술** : **배깅**(Bagging), **랜덤 노드 최적화**(RNO : Randomized Node Optimization)
- **배깅과 랜덤 노드 최적화라는 방법을 이용하여 하나의 데이터셋을 이용해 여러 개의 트리를 만드는 방법**

"트리는 정보 이득(Information Gain)이라는 방법을 이용해 정해진 규칙에 따라 만들게 되는데, 어떻게 여러 개를 만들지?"

"하나의 데이터로부터 엔트로피(Entropy), 지니 불순도(Gini Impurity) 등의 정보 이론을 위한 식을 이용하면 규칙이 정해져 있으니 하나의 트리만 만들어지지 않을까?"

랜덤 포레스트의 이해

부트스트랩

부트스트랩(Bootstrap)

원본 데이터셋에서 무작위로 중복 추출을 허용하여 동일한 사이즈의 데이터셋을 여러 개 만드는 샘플링 방법

예시

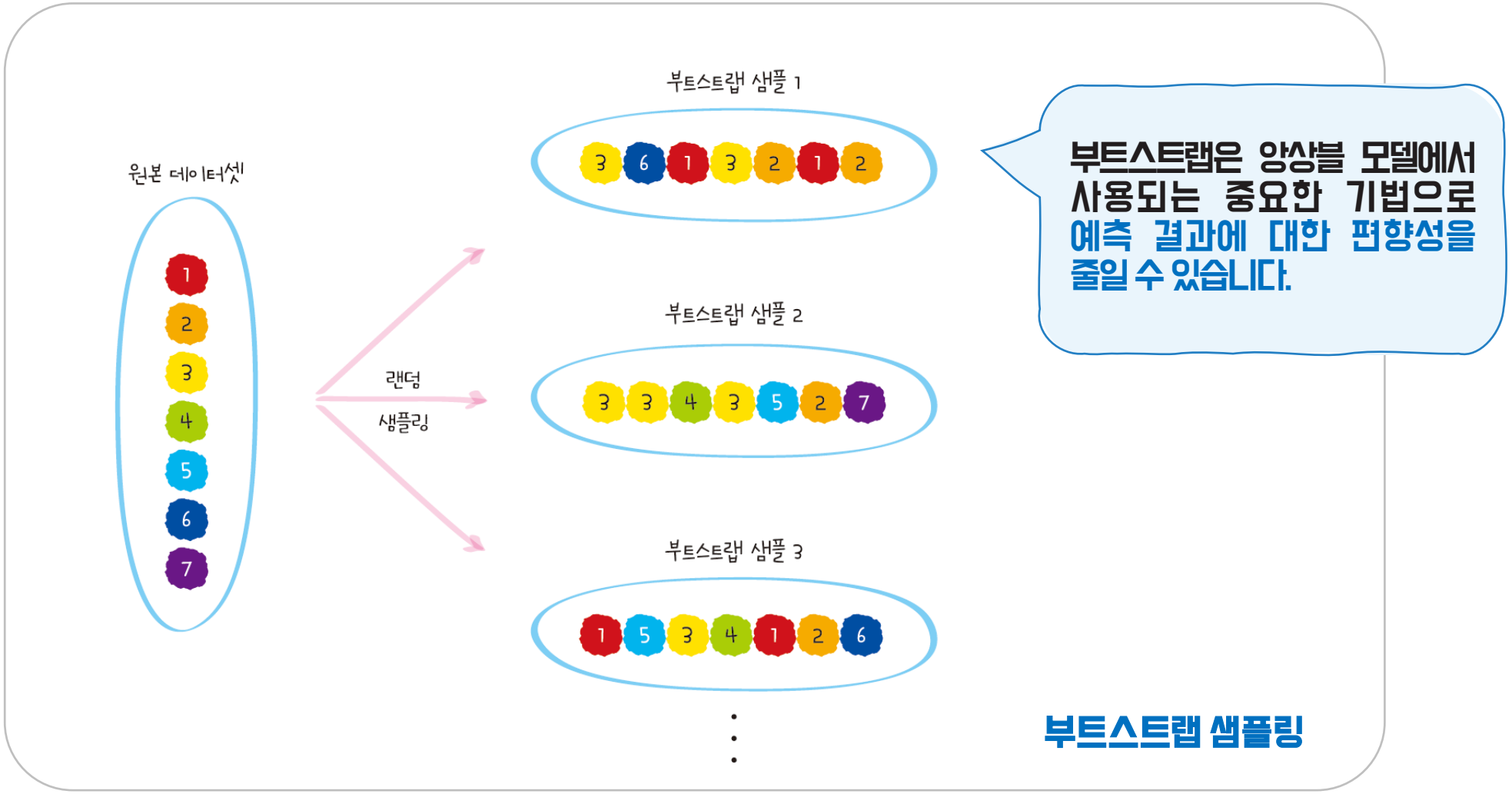
- 원본 데이터셋: 1,000개의 샘플
- 중복을 허용하여 1,000개 샘플 뽑기
- 이런 작업을 100번 반복 → 100개의 새로운 데이터셋 생성
- 각 데이터셋은 비슷하지만 약간 다름

➔ 작은 차이가 성능 개선에 큰 영향

비유 연금 복권: 10개의 숫자로 번호 생성, 중복 허용

랜덤 포레스트의 이해

부트스트랩

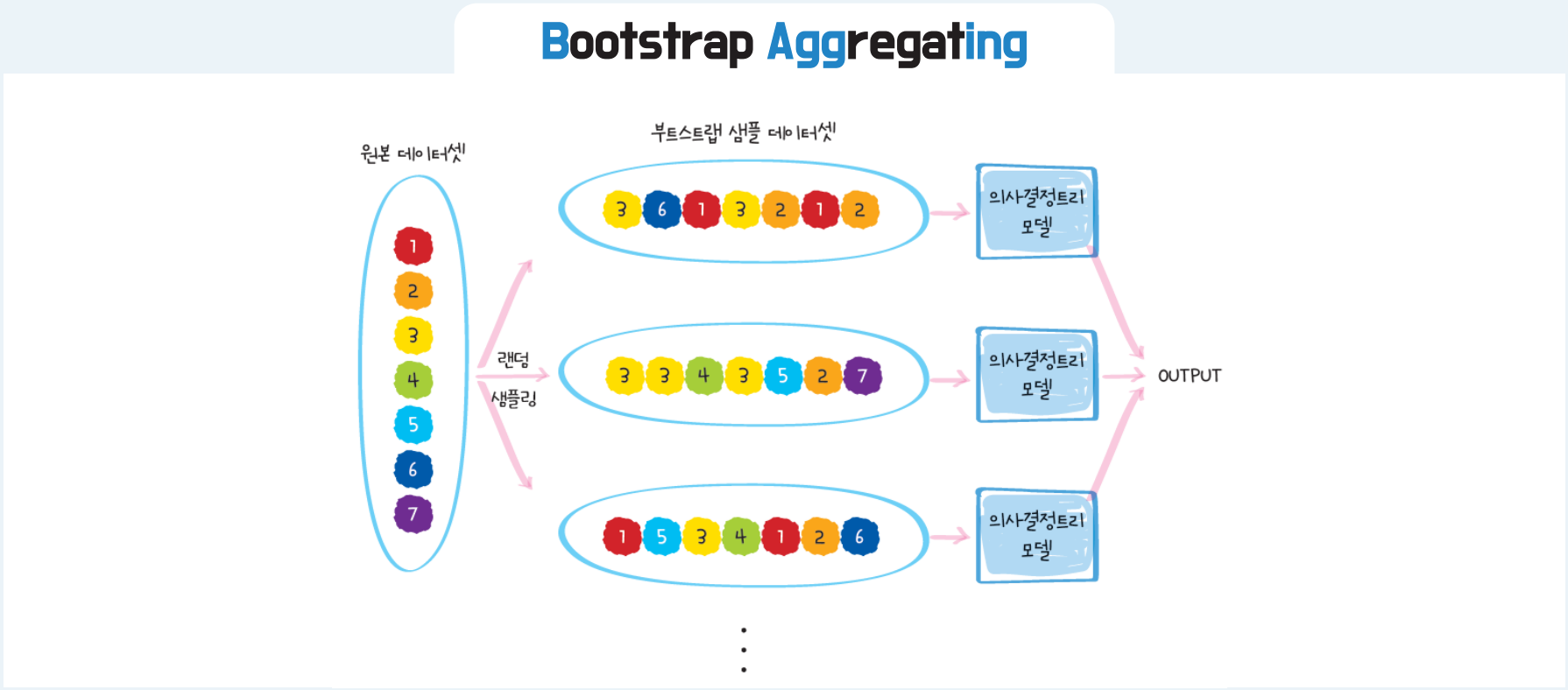


랜덤 포레스트의 이해

배깅

배깅(Bagging)

부트스트랩 샘플링을 이용하여 각각의 의사결정트리를 만들고 그 트리의 예측 결과를 종합하여 결론을 내는 방식



랜덤 포레스트의 이해

랜덤 노드 최적화

랜덤 노드 최적화(randomized node optimization, RNO)

랜덤 포레스트에서 생성된 숲의 트리 결과 평균을 사용하는 방식으로, 트리의 노드를 최적화하는 기법

1. 트리의 특성 선택
   - 일반 트리: 데이터의 모든 특성 사용
   - 랜덤 포레스트: 특성의 일부만 랜덤 선택
2. 특성 선택 규칙
   - 선택할 특성 수 = 전체 특성 수의 제곱근
   - 예: 9개의 특성 → 3개의 특성만 랜덤 선택
3. 부트스트랩과 트리 구성
   - 원본 샘플로부터 100개의 부트스트랩 데이터셋 생성
   - 각 데이터셋으로 100개의 다양한 트리 구성
4. 결과 평가
   - 랜덤 포레스트에서 생성된 숲의 트리 결과 평균 사용

모든 특성을 사용하면 각 트리가 서로 비슷해져 다양성이 부족해지고, 일반화되기 어려운 과대적합(Overfitting) 문제가 발생하기때문입니다.

문제 정의하기

문제 상황 이해하기

흥행할 영화를 미리 알아낼 수 있을까?

C사에서 영화 홍보를 담당하는 김 대리는 영화 개봉 초기에 흥행 여부를 미리 알아내고 싶습니다. 왜냐구요? 그걸 알면 흥행하지 않을 것 같은 영화는 더 이상 마케팅 비용을 쏟아붓지 않아도 되니까요.

문제 정의하기

문제 해결에 필요한 정보 살펴보기

문제 해결 과정에서 필요한 정보를 미리 살펴봅시다.

1 이 활동에 필요한 데이터셋은 무엇이고, 이 데이터셋은 어디에서 수집할 수 있나요?

우리나라 영화 관련 통계를 관장하고 있는 영화진흥위원회(Kofic)의 영화관 입장권 통합 전산망을 이용하면 데이터셋을 구할 수 있습니다. 하지만 가져온 데이터를 전처리해야 하는 번거로움이 있으므로 여기서는 인공지능 경진대회 플랫폼인 DACON에서 제공하는 영화 관객수 데이터셋을 사용합니다.

문제 정의하기

문제 해결에 필요한 정보 살펴보기

2 모델 학습에 사용할 알고리즘은 무엇인가요?

랜덤 포레스트(Random Forest) 알고리즘을 사용합니다. 의사결정 트리 모델에서 성능을 높이기 위해 한 개가 아닌 여러 개의 트리를 만들어 그 결과를 종합하여 결론을 이끌어 냅니다. 집단 지성을 이용하는 방법이죠.

3 모델 학습을 위해 어떤 처리를 해야 할까요?

데이터에서 특징(feature)이 무엇인지 학습하고 정형 데이터에서 특징을 살펴봅니다. 더불어 이번 활동에서는 영화 흥행을 예측하는 것이므로 원래 가지고 있는 데이터를 전처리해 흥행과 비흥행을 가를 수 있는 속성을 추가해야 합니다.

데이터 불러오기

데이터셋 소개하기

영화 데이터셋 살펴보기

- **영화진흥위원회 등에서 데이터를 수집하여 DACON에서 제공하는 데이터**

속성명	의미	비고
title	영화 제목	
distributor	배급사	
genre	장르	
release_time	개봉일	
time	상영 시간(분)	
screening_rat	상영 등급	
director	감독 이름	
dir_prev_bfnum	해당 감독이 이 영화를 만들기 전 제작에 참여한 영화에서의 평균 관객 수	관객 수가 알려지지 않은 영화 제외
dir_prev_num	해당 감독이 이 영화를 만들기 전 제작에 참여한 영화의 개수	
num_staff	스태프 수	
num_actor	주연 배우 수	
box_off_num	관객 수	

줄여서 '감독의 전작 평균 관객 수'라고 정합니다.

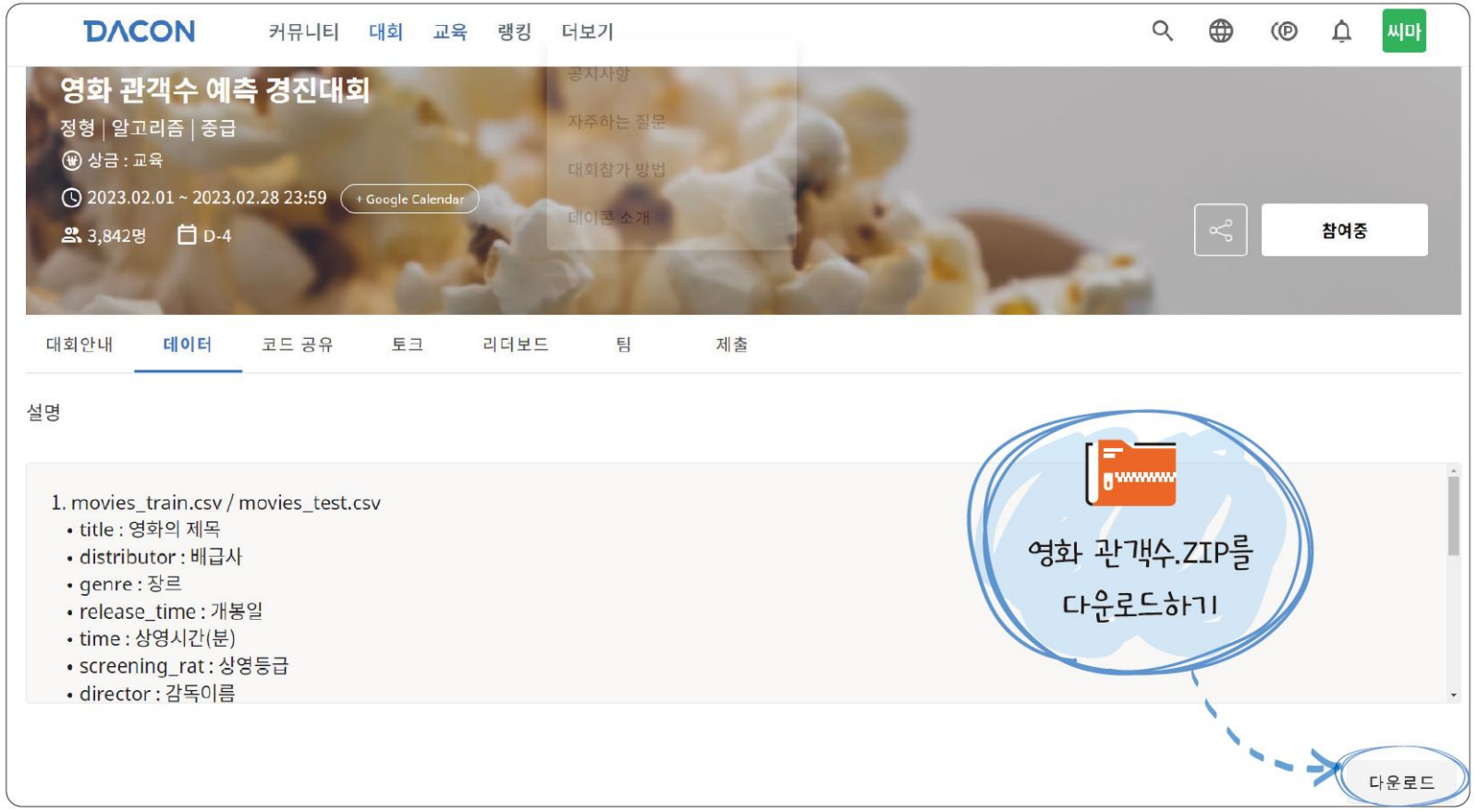
줄여서 '감독의 전작 작품 수'라고 정합니다.

100만명을 기준으로 흥행/비흥행 속성 추가 예정

데이터 불러오기

영화 흥행 데이터셋 다운로드하기

DACON에 접속하여 로그인한 후, 다운로드 버튼을 클릭해서 데이터를 다운로드하기



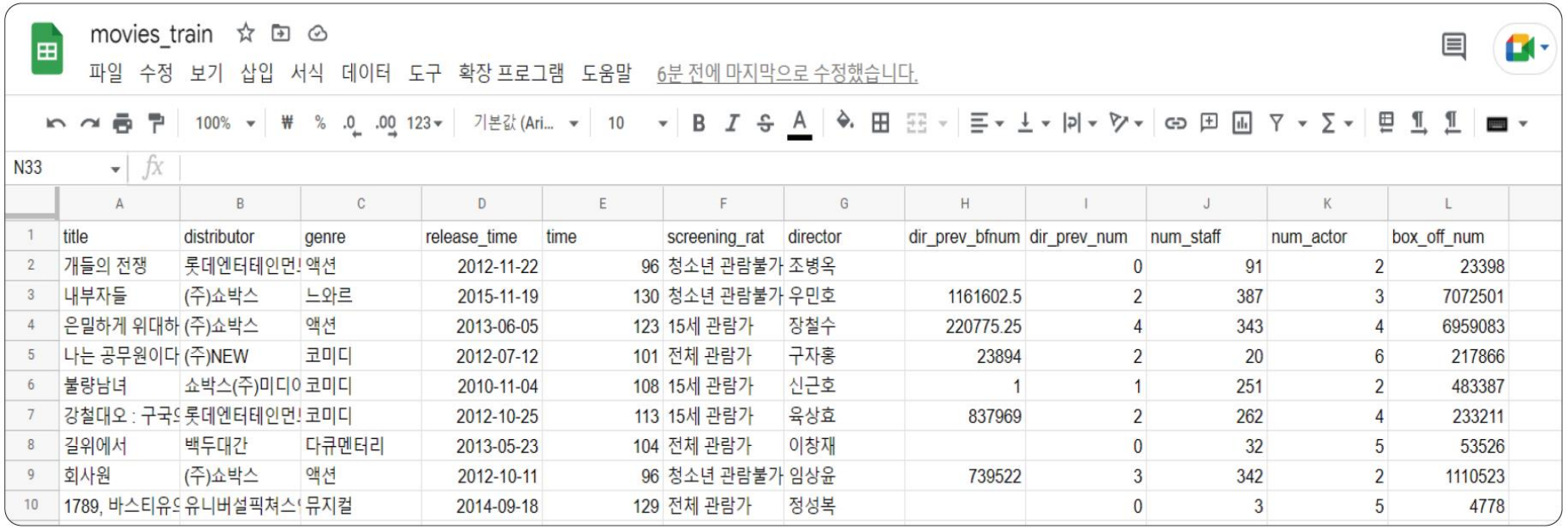
데이터를 다운로드하기 위해서는 대회에 참여 신청을 하고 간단한 레벨 테스트를 마쳐야 합니다.

https://dacon.io/competitions/open/235536/data

데이터 불러오기

데이터셋 불러오기

- 압축 파일 안의 3개의 데이터 중 movies_train.csv 파일 이용



movies_train

파일 수정 보기 삽입 서식 데이터 도구 확장 프로그램 도움말 6분 전에 마지막으로 수정했습니다.

N33

	A	B	C	D	E	F	G	H	I	J	K	L
1	title	distributor	genre	release_time	time	screening_rat	director	dir_prev_bfnum	dir_prev_num	num_staff	num_actor	box_off_num
2	개들의 전쟁	롯데엔터테인먼	액션	2012-11-22	96	청소년 관람불가	조병옥		0	91	2	23398
3	내부자들	(주)쇼박스	느와르	2015-11-19	130	청소년 관람불가	우민호	1161602.5	2	387	3	7072501
4	은밀하게 위대하	(주)쇼박스	액션	2013-06-05	123	15세 관람가	장철수	220775.25	4	343	4	6959083
5	나는 공무원이다	(주)NEW	코미디	2012-07-12	101	전체 관람가	구자홍	23894	2	20	6	217866
6	불량남녀	쇼박스(주)미디어	코미디	2010-11-04	108	15세 관람가	신근호	1	1	251	2	483387
7	강철대오 : 구국의	롯데엔터테인먼	코미디	2012-10-25	113	15세 관람가	육상효	837969	2	262	4	233211
8	길위에서	백두대간	다큐멘터리	2013-05-23	104	전체 관람가	이창재		0	32	5	53526
9	회사원	(주)쇼박스	액션	2012-10-11	96	청소년 관람불가	임상윤	739522	3	342	2	1110523
10	1789, 바스티유의	유니버설픽쳐스	뮤지컬	2014-09-18	129	전체 관람가	정성복		0	3	5	4778

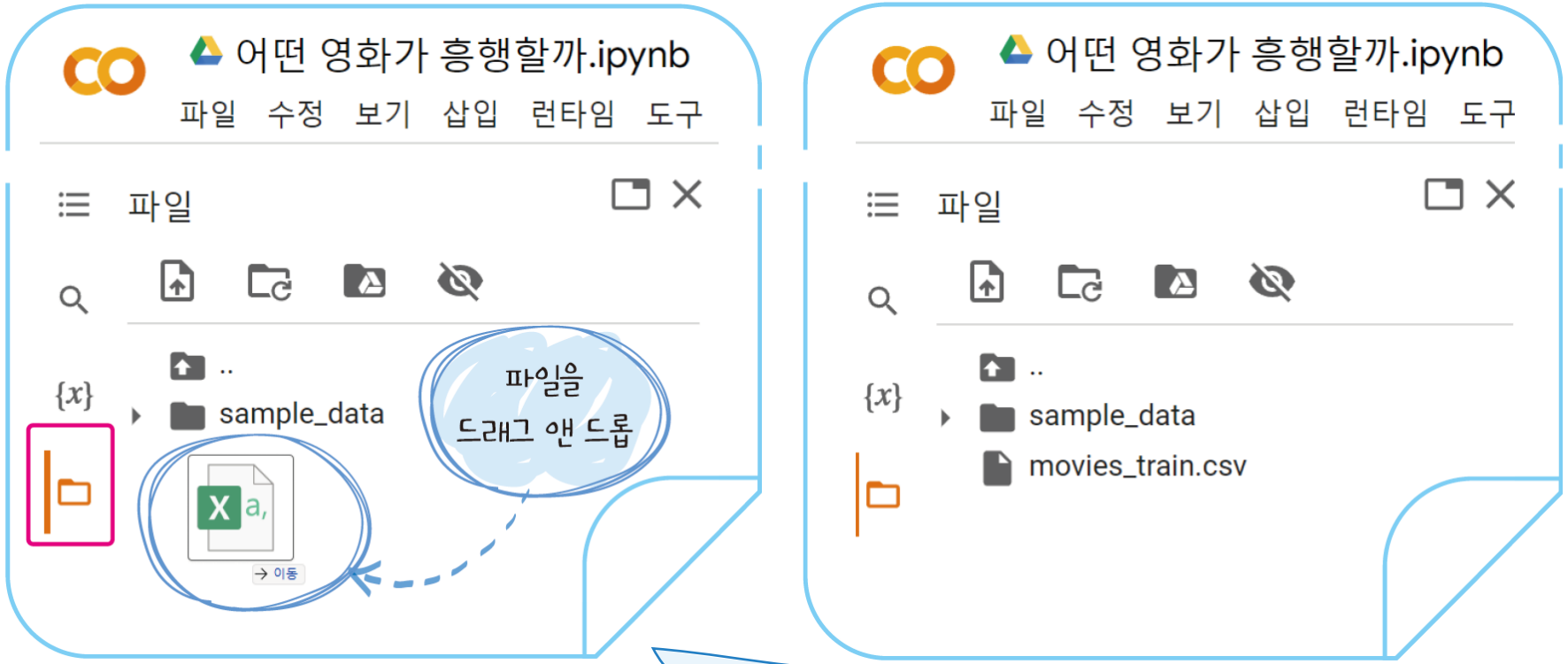
▲ 영화 흥행 데이터 (movies_train.csv)

- 영화의 제목(title)
- 배급사(distributor)
- 장르(genre),
- 개봉일(release_time)
- 상영 시간(time)
- 상영 등급(screening_rat
- 감독(director)
- 감독의 전작 평균 관객 수(dir_prev_bfnum)
- 감독의 전작 작품 수(dir_prev_num)
- 이 영화에 참여한 스태프 수(num_staff)
- 주연 배우 수(num_actor)
- 관객 수(box_ off_num)

데이터 불러오기

파일 업로드하기

- movies_train.csv 파일을 코랩의 작업 디렉토리로 드래그 앤 드롭하여 업로드하기



성능 측정시에 movies_test.csv파일은 필요한 종속변수인 관객 수(box_off_num)가 없기 때문에 사용하지 않습니다.

데이터 불러오기

파일 읽어 들이기

- 판다스 라이브러리 read_csv()를 이용하여 파일을 데이터프레임으로 읽어 들이기

```
1  import pandas as pd
2  movies = pd.read_csv('movies_train.csv')
3  movies
```

	title	distributor	genre	release_time	time	screening_rat	director	dir_prev_bfnum	dir_prev_num	num_staff	num_actor	box_off_num
0	개들의 전쟁	롯데엔터테인먼트	액션	2012-11-22	96	청소년 관람불가	조병옥	NaN	0	91	2	23398
1	내부자들	(주)쇼박스	느와르	2015-11-19	130	청소년 관람불가	우민호	1161602.50	2	387	3	7072501
2	은밀하게 위대하게	(주)쇼박스	액션	2013-06-05	123	15세 관람가	장철수	220775.25	4	343	4	6959083
3	나는 공무원이다	(주)NEW	코미디	2012-07-12	101	전체 관람가	구자홍	23894.00	2	20	6	217866
4	불량남녀	쇼박스(주)미디어플렉스	코미디	2010-11-04	108	15세 관람가	신근호	1.00	1	251	2	483387
...	...	...	...	...	...	...	...	...	...	...	...	...
595	해무	(주)NEW	드라마	2014-08-13	111	청소년 관람불가	심성보	3833.00	1	510	7	1475091
596	파파로티	(주)쇼박스	드라마	2013-03-14	127	15세 관람가	윤종찬	496061.00	1	286	6	1716438
597	살인의 강	(주)마운틴픽쳐스	공포	2010-09-30	99	청소년 관람불가	김대현	NaN	0	123	4	2475
598	악의 연대기	CJ 엔터테인먼트	느와르	2015-05-14	102	15세 관람가	백운학	NaN	0	431	4	2192525
599	베를린	CJ 엔터테인먼트	액션	2013-01-30	120	15세 관람가	류승완	NaN	0	363	5	7166532

600 rows × 12 columns

해석

코랩의 작업 디렉토리로 업로드한 파일을 판다스 데이터프레임 형태로 읽어옵니다. movies 변수를 출력해 보면 총 12개의 속성(columns), 600개의 행(rows, 샘플)을 가진 데이터를 확인할 수 있습니다.

데이터 불러오기

Q 엑셀에서 영화 데이터 CSV 파일을 열었더니 글자가 깨져서 나옵니다.

A 엑셀 실행 후 '데이터' 메뉴 → '텍스트/CSV'→ CSV 파일 선택 → 파일 원본의 인코딩 방식을 '유니코드 (UTF-8)'로 변경하면 됩니다.

데이터 처리하기

데이터 살펴보기

데이터 기초 정보 확인하기

- 판다스 라이브러리의 info( ) 메소드를 사용하여 데이터의 기초 정보를 확인하기

```
1  movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 600 entries, 0 to 599
Data columns (total 12 columns):
 #    Column          Non-Null Count  Dtype
---   ------          --------------  -----
 0    title           600 non-null    object
 1    distributor     600 non-null    object
 2    genre           600 non-null    object
 3    release_time    600 non-null    object
 4    time            600 non-null    int64
 5    screening_rat   600 non-null    object
 6    director        600 non-null    object
 7    dir_prev_bfnum  270 non-null    float64
 8    dir_prev_num    600 non-null    int64
 9    num_staff       600 non-null    int64
 10   num_actor       600 non-null    int64
 11   box_off_num     600 non-null    int64
dtypes: float64(1), int64(5), object(6)
memory usage: 56.4+ KB
```

해석

이 데이터셋은 총 600개의 데이터로 구성되어 있고 속성은 12개입니다. 각 속성별 값의 개수가 대부분 600개인데 감독의 전작 평균 관객 수(dir_prev_bfnum)가 270개인 것을 보니 해당 속성은 결측치가 꽤 많습니다. 속성별 데이터 유형은 정수형(int64), 실수형(float64), 그리고 문자열인 데이터 유형은 object로 표시되어 있습니다.

여기서 이제 결측치를 어떻게 해야 할지 결정해야 합니다. 일반적으로 값이 없는 null은 해당 행들을 삭제하는 것이 가장 좋은 방법이지만, 만약 삭제하는 행이 많다면 데이터가 많이 사라지게 되므로 중간값이나 평균값을 이용하기도 합니다.

데이터 처리하기

데이터 기초 정보 확인하기

- 결측치 중간값을 사용하여 채우기

```
1  data = movies.fillna(movies.median())
2  data.info()
```

```
                                   ⋮
 7    dir_prev_bfnum       600 non-null     float64
                                   ⋮
dtypes: float64(1), int64(5), object(6)
```

해석

원본 데이터에 null 값들, 즉 비어 있는 값이 있으면 해당 칼럼을 fillna( ) 함수를 사용해서 지정한 값으로 채웁니다. 여기서는 중간값으로 채우기 위해 fillna( )의 매개변수로 median( ) 함수를 사용했습니다. 그 결과 dir_prev_bfnum 속성이 270에서 600으로 변한 것을 확인할 수 있습니다.

데이터 처리하기

데이터 통계량 살펴보기

- data라는 이름의 데이터프레임에 있는 수치 데이터들의 기술(descriptive) 통계 정보를 describe( ) 함수를 사용하여 알아보기

```
1  data.describe()
```

해석

관객 수(box_off_num) 속성을 확인해 보니 최소 관객 수가 1명인 영화도 있고, 최대 입장관객수가 1426만명이넘는 것도 확인할수있습니다. 따라서 관객 수(box_off_num) 속성은 표준편차가 매우 크다는 것도 알 수 있습니다. 감독의 전작 평균 관객 수(dir_prev_bfnum)의 경우도 유사한 것을 확인할 수 있습니다.

	time	dir_prev_bfnum	dir_prev_num	num_staff	num_actor	box_off_num
count	600.000000	6.000000e+02	600.000000	600.000000	600.000000	6.000000e+02
mean	100.863333	7.358323e+05	0.876667	151.118333	3.706667	7.081818e+05
std	18.097528	1.233810e+06	1.183409	165.654671	2.446889	1.828006e+06
min	45.000000	1.000000e+00	0.000000	0.000000	0.000000	1.000000e+00
25%	89.000000	4.784236e+05	0.000000	17.000000	2.000000	1.297250e+03
50%	100.000000	4.784236e+05	0.000000	82.500000	3.000000	1.259100e+04
75%	114.000000	4.784236e+05	2.000000	264.000000	4.000000	4.798868e+05
max	180.000000	1.761531e+07	5.000000	869.000000	25.000000	1.426277e+07

데이터 처리하기

범주형 데이터 확인하기

문자열로 되어 있는 범주형 데이터인 배급사와 장르, 상영 등급에 대하여 살펴보겠습니다. 먼저 배급사(distributor) 속성의 고윳값과 고윳값 개수를 알아보겠습니다.

```
1  print(data['distributor'].unique()) # 데이터 고윳값 확인
2  print(data['distributor'].value_counts()) # 데이터 고윳값의 개수 확인
```

```
['롯데엔터테인먼트' '(주)쇼박스' '(주)NEW' '쇼박스(주)미디어플렉스' '백두대간'
'유니버설픽쳐스인터내셔널코리아' … '영화사 廊' '크리에이티브컴즈(주)' 'ysfilm' '이달투' '퍼스트런']
CJ 엔터테인먼트          54
롯데엔터테인먼트         52
(주)NEW                  30
(주)마운틴픽쳐스         29
(주)쇼박스               26
                         ..
OAL(올)                   1
(주)에이원 엔터테인먼트   1
(주)콘텐츠 윙             1
위더스필름                1
퍼스트런                  1
Name: distributor, Length: 169, dtype: int64
```

해석

제일 많은 영화를 배급한 회사는 CJ 엔터테인먼트이고, 상당히 많은 회사가 1개 영화를 배급한 것으로 파악됩니다.

데이터 처리하기

보충

배급사별 비율 알아보기

value_counts( )는 기본적으로 normalize 옵션은 False가 기본값이지만, True로 설정하면 전체 합을 1인 상태 에서 모든 값을 비율로 계산해서 반환해 줍니다.

```
1  print(data['distributor'].value_counts(normalize = True))
```

```
CJ 엔터테인먼트          0.090000
롯데엔터테인먼트         0.086667
(주)NEW                  0.050000
(주)마운틴픽쳐스         0.048333
(주)쇼박스               0.043333
                           ...
OAL(올)                  0.001667
(주)에이원 엔터테인먼트  0.001667
(주)콘텐츠 윙            0.001667
위더스필름               0.001667
퍼스트런                 0.001667
Name: distributor, Length: 169, dtype: float64
```

데이터 처리하기

범주형 데이터 확인하기

- 장르(genre)속성의 고윳값과 고윳값 개수 알아보기

```
1  print(data['genre'].unique())
2  print(data['genre'].value_counts())
```

['액션' '느와르' '코미디' '다큐멘터리' '뮤지컬' '드라마' '멜로/로맨스' '공포' '서스펜스' '애니메이션'
'미스터리' 'SF']

드라마	221
다큐멘터리	93
멜로/로맨스	78
코미디	53
공포	42
액션	28
느와르	27
애니메이션	21
미스터리	17
SF	13
뮤지컬	5
서스펜스	2

Name: genre, dtype: int64

해석

장르는 12개 장르가 있네요. 훈련 데이터셋에 가장 많은 장르는 드라마입니다. 그리고 가장 적은 장르는 서스펜스입니다. 이 데이터셋 안에는 두 개의 영화만 서스펜스 장르입니다.

데이터 처리하기

데이터 시각화하기

통계 기반 분석, 전통적 머신러닝

- 사람이 예측값과 관련 있는 속성을 직접 설정

빅데이터 시대

- 데이터 마이닝, 딥러닝 등에서 기계가 스스로 데이터의 관계성 탐색
- 사람이 미리 설정하지 않음

관객 수가 중요한 속성이므로 **관객 수와 나머지 변수들 간의 관계를 알아보는 것이 중요함.**

pairplot 그리기

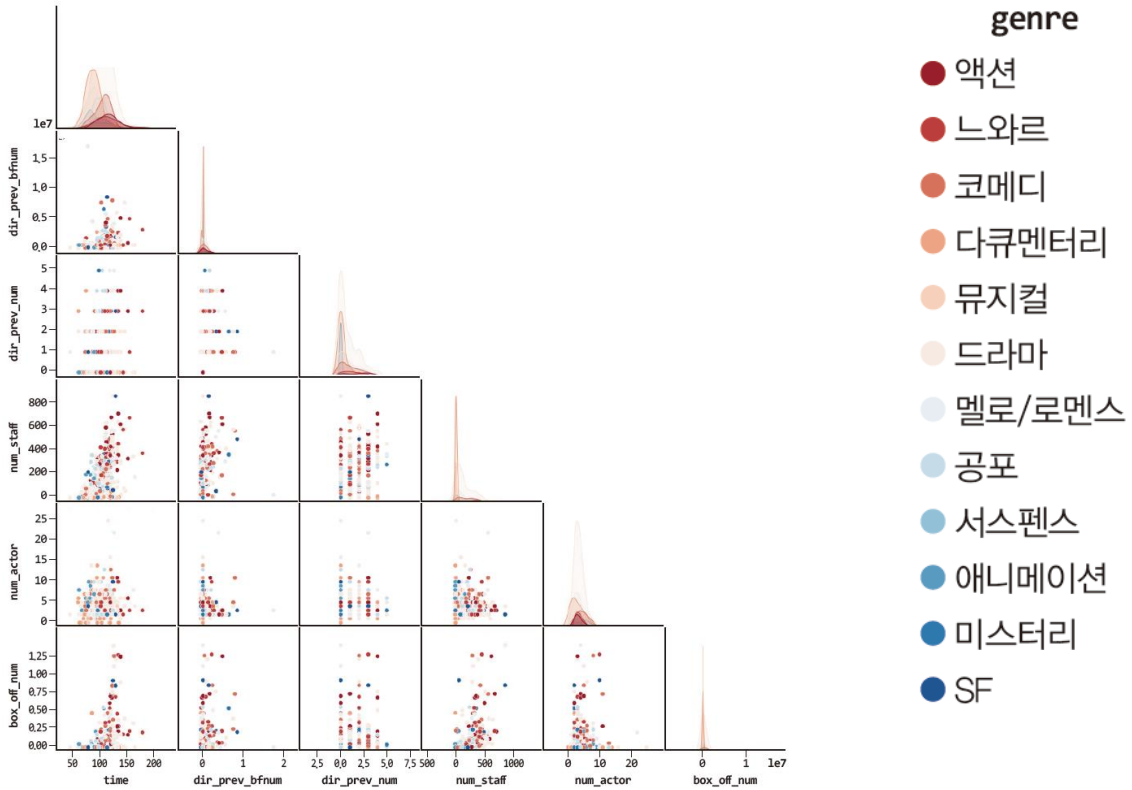
데이터 시각화 도구

- **시본 라이브러리의** 'pairplot()' **사용**
- **히스토그램 및 분포도로 다양한 속성의 관계 그리기**
- **장르**('genre')**를 색으로 구분**, 'RdBu' **팔레트 사용**

데이터 처리하기

pairplot 그리기

```
1  import matplotlib.pyplot as plt
2  plt.rc('font', family = 'NanumBarunGothic') # 한글 폰트 설정
3  import seaborn as sns
4  sns.pairplot(data, hue = 'genre', palette = 'RdBu', corner = True)
```



데이터 처리하기

상관계수 그리기

- corr() 메소드를 사용한 상관관계 분석을 이용해 좀 더 면밀하게 속성들 간의 상관계수 알아보기

```
1  corr = data.corr().round(2)
2  corr
```

	time	dir_prev_bfnum	dir_prev_num	num_staff	num_actor	box_off_num
time	1.00	0.23	0.31	0.62	0.11	0.44
dir_prev_bfnum	0.23	1.00	0.26	0.32	0.05	0.27
dir_prev_num	0.31	0.26	1.00	0.45	0.01	0.26
num_staff	0.62	0.32	0.45	1.00	0.08	0.54
num_actor	0.11	0.05	0.01	0.08	1.00	0.11
box_off_num	0.44	0.27	0.26	0.54	0.11	1.00

실행 결과를 보면 의외의 결과를 볼 수 있는데 상영 시간(time)과 영화 제작의 스태프 수(num_staff)가 상관관계가 굉장히 높습니다. 더욱 재미있는 것은 관객 수(box_off_num)와 스태프의 수(num_staff)가 어느 정도 상관이 있는 것으로 나타났습니다. 그 다음은 상영 시간(time)인 것을 알 수 있고 주연 배우 수(num_actor)는 관객 수(box_off_num)와 가장 관련 없는 속성으로 보입니다.

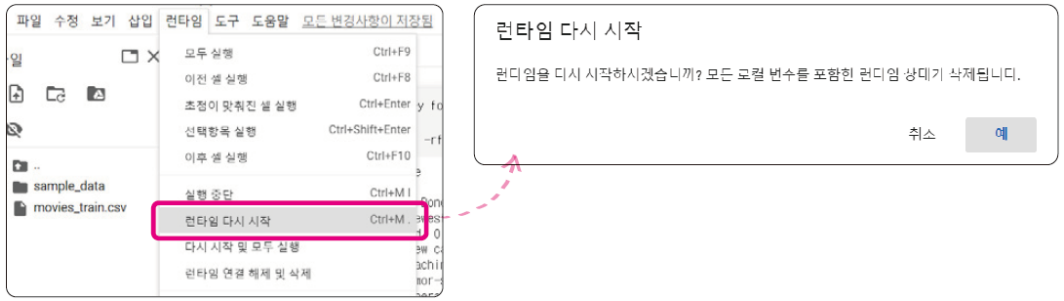
데이터 처리하기

#코랩에서 맷플롯립이나 시본으로 시각화할 때 한글 폰트 깨짐 현상이 나타납니다.

1.한글나눔폰트를설치하는코드를작성하고실행합니다.

```
1 !sudo apt-get install -y fonts-nanum
2 !sudo fc-cache -fv
3 !rm ~/.cache/matplotlib -rf
```

2. 메뉴의 런타임(Runtime) → 런타임 다시 시작 (Restart runtime)을 선택합니다.



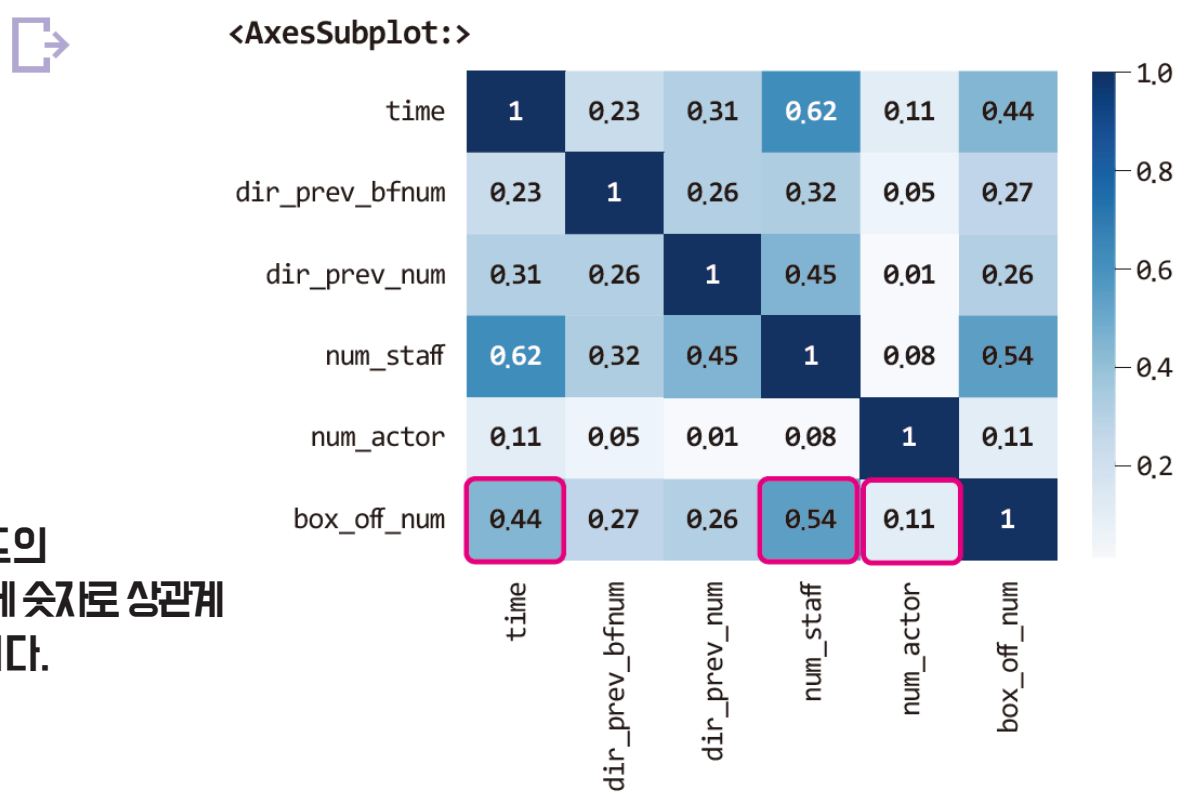
3.한글폰트설정코드를작성하고실행합니다.

```
1 import matplotlib.pyplot as plt
2 plt.rc('font', family = 'NanumBarunGothic')
```

데이터 처리하기

히트맵 그리기

```
1 sns.heatmap(corr, annot = True, cmap
  = 'Blues')
```



heatmap() 메소드의
매개 변수 annot는 히트맵 위에 숫자로 상관계
수 값을 표현해 줍니다.

해석

왼쪽 히트맵에서는 상관계수가 높을수록 진한 색으로 표시하고 있습니다. 관객 수(box_off_ num)는 스태프 수(num_staff)와 가장 상관관계가 높고, 그 다음 상영 시간(time) 순입니다. 주연 배우 수(num_actor)와는 가장 상관관계가 낮음을 확인할 수 있습니다.

데이터 처리하기

데이터 전처리하기

흥행 속성 추가하기

영화 흥행 여부 분류 기준 만들기

배경	▪ 데이터셋에는 영화의 관객 수(box_off_num) 정보가 있음. ▪ 하지만, box_off_num은 연속적인 값으로 분류 작업에 직접적으로 사용하기 어려움.
흥행 여부 기준 설정	▪ 100만 명의 관객을 기준으로 흥행 여부 결정. ◦ 100만 명 이상: 흥행 ◦ 100만 명 미만: 비흥행
데이터 프레임 변경	▪ data 데이터프레임에 새로운 속성 hit 추가. ◦ 흥행: hit = 1 ◦ 비흥행: hit = 0

데이터 처리하기

흥행 속성 추가하기

넘파이 객체.where(조건식,

조건식이 참일 때 반환값, 조건식이 거짓일 때 반환값)

조건식의 결과에 따라 다른 값을 반환

```
1  import numpy as np
2  data['hit'] = np.where(data['box_off_num'] >= 1000000, 1, 0)
3  data.tail()
```

추가한 흥행 여부에 대한 hit 속성이 종속변수입니다.

데이터 처리하기

흥행 속성 추가하기

	title	distributor	genre	release_time	time	screening_rat	director	dir_prev_bfnum	dir_prev_num	num_staff	num_actor	box_off_num	hit
595	해무	(주)NEW	드라마	2014-08-13	111	청소년 관람불가	심성보	3833.000	1	510	7	1475091	1
596	파파로티	(주)쇼박스					윤종찬	496061.000	1	286	6	1716438	1
597	살인의 강	(주)마운틴픽쳐스					김대현	478423.625	0	123	4	2475	0
598	악의 연대기	CJ 엔터테인먼트					백운학	478423.625	0	431	4	2192525	1
599	베를린	CJ 엔터테인먼트					류승완	478423.625	0	363	5	7166532	1

데이터프레임에 hit라는 속성이 1 또는 0값으로 추가되어 있는 것을 확인할 수 있습니다.

데이터 처리하기

흥행 속성 추가하기

- hit 속성이 1인 데이터. 즉 흥행 영화가 총 몇 개인지 알아보기

```
1  print(data['hit'].sum())
```

```
109
```

600개의 영화 중에 100만 명 이상의 관객을 동원한 것이 109개인 것을 확인할 수 있습니다.

데이터 처리하기

독립변수와 종속변수 정하기

#	Column	Non-Null Count	Dtype
0	title	600 non-null	object
1	distributor	600 non-null	object
2	genre	600 non-null	object
3	release_time	600 non-null	object
4	time	600 non-null	int64
5	screening_rat	600 non-null	object
6	director	600 non-null	object
7	dir_prev_bfnum	600 non-null	float64
8	dir_prev_num	600 non-null	int64
9	num_staff	600 non-null	int64
10	num_actor	600 non-null	int64
11	box_off_num	600 non-null	int64
12	hit	600 non-null	int64

독립변수로 사용

종속변수

- 배급사(distributor)
- 장르(genre)
- 상영 등급(screening_rat)
- 감독의 전작 평균 관객 수 (dir_prev_bfnum)
- 감독의 전작 작품 수 (dir_prev_num)
- 스태프 수(num_staff)
- 주연 배우 수(num_actor)

- 흥행 여부(hit)

```
1  X = data.iloc[:, [1, 2, 5, 7, 8, 9, 10]]
2  y = data['hit']
```

데이터 처리하기

독립변수
종속변수
출력하기

```
1 display(X)
2 display(y)
```

	distributor	genre	screening_rat	dir_prev_bfnum	dir_prev_num	num_staff	num_actor
0	롯데엔터테인먼트	액션	청소년 관람불가	478423.625	0	91	2
1	(주)쇼박스	느와르	청소년 관람불가	1161602.500	2	387	3
2	(주)쇼박스	액션	15세 관람가	220775.250	4	343	4
3	(주)NEW	코미디	전체 관람가	23894.000	2	20	6
4	쇼박스(주)미디어플렉스	코미디	15세 관람가	1.000	1	251	2
...	...	...	...	...	...	...	...
595	(주)NEW	드라마	청소년 관람불가	3833.000	1	510	7
596	(주)쇼박스	드라마	15세 관람가	496061.000	1	286	6
597	(주)마운틴픽쳐스	공포	청소년 관람불가	478423.625	0	123	4
598	CJ 엔터테인먼트	느와르	15세 관람가	478423.625	0	431	4
599	CJ 엔터테인먼트	액션	15세 관람가	478423.625	0	363	5

600 rows × 7 columns

```
0      0
1      1
2      1
3      0
4      0
      ..
595    1
596    1
597    0
598    1
599    1
Name: hit, Length: 600, dtype: int64
```

연속형 수치가 아닌 범주형 속성
범주형 속성을 원-핫 인코딩을 사용하여 수치형 데이터로 변환하기

데이터 처리하기

원-핫 인코딩하기

- get_dummies( ) 메소드를 이용하여 범주형 데이터를 수치형 데이터로 바꾸기

판다스 객체.get_dummies

(데이터프레임 객체, columns = ['인코딩할 속성명'])

columns에 인코딩할 속성을 지정해 주지 않으면,

데이터프레임의 모든 범주형 속성을 자동으로 더미 변환함.

데이터 처리하기

원-핫 인코딩하기

```
1  X = pd.get_dummies(X, columns = ['distributor', 'genre', 'screening_rat'])
2  X
```

	dir_prev_bfnum	dir_prev_num	num_staff	num_actor	distributor_(주)브릿지웍스	distributor_(주)액티버스엔터테인먼트	genre_코미디	screening_rat_12세 관람가	screening_rat_15세 관람가	screening_rat_전체 관람가	screening_rat_청소년 관람불가
0	478423.625	0	91	2	0	0	0	0	0	0	1
1	1161602.500	2	387	3	0	0	0	0	0	0	1
2	220775.250	4	343	4	0	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...
598	478423.625	0	91	2	0	0	0	0	1	0	0
599	478423.625	0	91	2	0	0	0	0	1	0	0

600 rows × 189 columns

해석

실행 결과를 보면 distributor, genre, screening_rat 속성을 원-핫 인코딩하여 범주의 개수만큼 새롭게 생성되었습니다. 표시된 것처럼 상영 등급(screening_rat) 속성이 원-핫 인코딩을 하여 범주(유형)의 개수만큼 4개 속성이 새롭게 생성되었음을 확인할 수 있습니다.

데이터 처리하기

훈련 데이터와 테스트 데이터 나누기

```
1  from sklearn.model_selection import train_test_split
2  X_train, X_test, y_train, y_test = \
3      train_test_split(X, y, test_size = 0.2, random_state = 42)
```

훈련 데이터가 80%, 테스트 데이터가 20%가 되도록 나누었습니다. 전체가 600개의 데이터이므로 훈련 데이터의 수는 480개, 테스트 데이터 수는 120개가 되겠군요. 실제 데이터가 이렇게 작을 때는 훈련 데이터를 더 많이 구성하기 위해 9:1의 비율로 나누어 주기도 합니다.

모델 학습하기

모델 생성하기

- RandomForestClassifier( )함수를 호출하여 랜덤 포레스트 객체 생성
- 생성된 객체의 매개 변수로 데이터를 입력하여 훈련시키기

랜덤 포레스트 모델 학습하기

```
1  from sklearn.ensemble import RandomForestClassifier
2  rf = RandomForestClassifier(random_state = 42)
3  rf.fit(X_train, y_train)
```

```
RandomForestClassifier(random_state = 42)
```

사이킷런 라이브러리에서 제공하는 랜덤 포레스트 분류기(classifier) 함수를 이용하여 랜덤 포레스트 모델 객체를 rf라는 이름으로 생성하였습니다.
fit( ) 함수를 이용하여 훈련용 독립변수(train_input)와 종속변수(train_target)로 훈련하였습니다.

모델 학습하기

훈련 데이터로 학습한 결과 확인하기

```
1  print(rf.score(X_train, y_train))
```

```
1.0
```

훈련에 사용한 독립변수(X_train)와 종속변수(y_train)를 이용한 성능 측정은 1.0, 즉 100%로 나왔습니다.

모델 학습하기

훈련 데이터로 학습한 결과 확인하기

의사결정 트리와 랜덤 포레스트의 과대적합 문제

의사결정 트리의 단점	▪ 과대적합(Overfitting)에 빠지기 쉬움. ▪ 아무런 옵션 없이 모델을 학습시키면 리프 노드에서 완벽하게 분류됨. ▪ 하지만 이 완벽함은 훈련 데이터에 한정되어 있음.
왜문제인가?	▪ 완벽하게 분류된 리프 노드의 결과는 훈련 데이터에 대해서만 완벽. ▪ 실제 신규 데이터나 테스트 데이터에 대한 예측 성능이 떨어질 수 있음.
랜덤 포레스트의 해결 방안	▪ 부트스트랩(Bootstrap)을 사용해 데이터의 다양성을 확보. ▪ 여러 개의 트리를 학습시켜 과대적합 문제를 줄여주어 성능을 향상시킴.

모델 테스트 및 평가하기

테스트하기

- 테스트할 때는 학습에 사용하지 않은 테스트용 데이터 사용
- 모델이 얼마나 정확하게 분류했는지 한눈에 비교할 수 있도록 테스트용 데이터와 분류한 결괏값을 데이터프레임으로 만들어 확인하기

```
1  y_pred = rf.predict(X_test)
2  result = pd.DataFrame({'Actual Value':y_test,
3                         'model prediction': y_pred})
4  result.head()
```

	Actual Value	model prediction
110	1	0
419	0	0
565	0	0
77	0	0
181	1	0

테스트용 독립변수에 대해 실젯값과 모델이 분류한 결괏값을 일부만 출력해 보았습니다. 실행 결과에서 비흥행으로 분류한 데이터와 흥행으로 분류한 데이터를 확인할 수 있습니다.

모델 테스트 및 평가하기

평가하기

- 테스트용 데이터로 예측한 결괏값은 성능을 평가할 때 테스트용 종속변수와 비교하여 분류의 정확도를 살펴보는 데 활용

```
1  print(rf.score(X_test, y_test))
```

```
0.8083333333333333
```

결과를 해석해 보면 배급사, 장르, 상영 등급, 감독의 전작 평균 관객 수, 감독의 전작 작품 수, 참여한 스태프의 수, 그리고 주연 배우 수를 알면 그 영화가 100만 관객을 넘겨 흥행할지 그렇지 않을지를 80%의 확률로 예측할 수 있다는 것을 보여 준 것입니다.
당연히 훈련용 데이터를 입력한 것에 비해서는 떨어지지만 비교적 좋은 성능을 나타내고 있습니다.

모델 테스트 및 평가하기

classification_report 출력하기

- 정확도 외에 다양한 분류 모델의 성능 평가 지표(정밀도, 재현율, f1 score)를 확인하기 위해 classification report 출력하기

```
1  from sklearn.metrics import classification_report
2  print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.82	0.96	0.88	92
1	0.69	0.32	0.44	28
accuracy			0.81	120
macro avg	0.76	0.64	0.66	120
weighted avg	0.79	0.81	0.78	120

해석

support는 각 클래스의 샘플 개수를 의미합니다. 실행 결과에서 타깃값이 0인 데이터는 92개, 1인 데이터는 28개임을 알 수 있습니다. 앞에서 확인했듯이 accuracy(정확도)는 약 0.81% 정도이고 precision(정밀도), recall(재현율), f1-score(조화 평균)와 macro avg(매크로 평균), weighted avg(가중 평균)을 확인할 수 있습니다.

모델 테스트 및 평가하기

평가 지표 계산식

	precision	recall	f1-score	support
0	88/(88 + 19) = 0.82	88/92 = 0.96	2×(0.82×0.96)/ (0.82 + 0.96) = 0.88	92
1	9/(9 + 4) = 0.69	9/28 = 0.32	2×(0.69×0.32)/ (0.69 + 0.32) = 0.44	28
accuracy			(88 + 9)/120 = 0.81	120
macro avg	(0.82 + 0.69)/2 = 0.76	(0.96 + 0.32)/2 = 0.64	(0.88 + 0.44)/2 = 0.66	120
weighted avg	(0.82×92 + 0.69×28) /120 = 0.79	(0.96×92 + 0.32×28) /120 = 0.81	(0.88×92 + 0.44×28) /120 = 0.78	120

모델 테스트 및 평가하기

혼동 행렬 출력하기

- 혼동행렬은 분류 모델의 성능을 평가하는 데 사용되며 모델이 분류한 결과와 실젯값을 비교하여 만들어지는 행렬

예측

실제값

	1 (흥행)	0 (비흥행)
1 (흥행)	**TP** 실제 흥행을 흥행이라고 맞게 분류 (True Positive)	**FN** 실제 흥행을 비흥행이라고 잘못 분류 (False Negative)
0 (비흥행)	**FP** 실제 비흥행을 흥행이라고 잘못 분류 (False Positive)	**TN** 실제 비흥행을 비흥행이라고 맞게 분류 (True Negative)

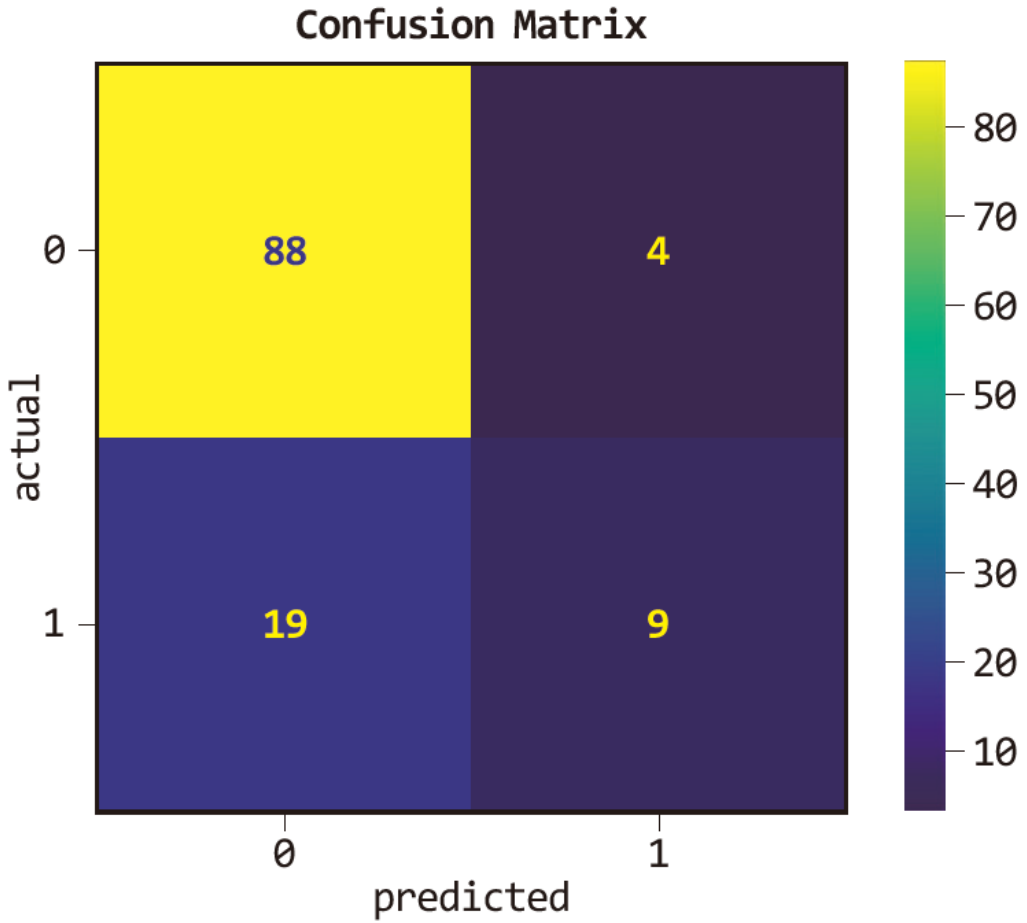
모델 테스트 및 평가하기

혼동 행렬 출력하기

```
1  import matplotlib.pyplot as plt
2  from sklearn import metrics
3
4  confusion_matrix = metrics.confusion_matrix(y_true = y_test,
5                                               y_pred = y_pred)
6  cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix =
7                                               confusion_matrix)
8
9  cm_display.plot()
10 plt.title('Confusion Matrix')
11 plt.xlabel('predicted')
12 plt.ylabel('actual')
13 plt.show()
```

모델 테스트 및 평가하기

혼동 행렬 출력하기



해석

실젯값이 비흥행인 데이터 92개 중에서 88개는 비흥행으로 맞게 분류(True Negative)했고, 4개는 흥행으로 잘못 분류(Fale Positive)하였습니다. 실젯값이 흥행인 데이터 28개 중에서 19개는 비흥행으로 잘못 분류(False Negative)했고, 9개는 흥행으로 맞게 분류(True Positive)했습니다.

모델 테스트 및 평가하기

평가 지표 확인하기

- 정확도 : 모델이 분류한 전체 건수 중에서 흥행인지, 비흥행인지를 맞게 분류한 건수의 비율

accuracy (정확도)

$$\text{accuracy(정확도)} = \frac{\text{분류 결과가 실젯값과 동일한 데이터 건수}}{\text{전체 분류 데이터 건수}} = \frac{TP + TN}{TP + TN + FP + FN}$$

precision (정밀도)

- 정밀도 : 모델이 흥행이라고 분류한 데이터 영화 중에 실제 흥행 영화로 맞게 분류한 데이터 건수가 얼마나 되는지를 나타내는 비율

$$\text{Precision(정밀도)} = \frac{\text{'양성'으로 분류하여 맞춘 데이터 건수}}{\text{'양성'으로 분류한 데이터 건수}} = \frac{TP}{TP + FP}$$

정밀도는 음성을 양성으로 잘못 판단하면 문제가 되는 경우에 사용합니다.

모델 테스트 및 평가하기

recall(재현율)

- 재현율 : 전체 데이터 중에서 실제 흥행 영화를 얼마나 많이 예측하였는지를 나타냄.

$$\text{recall(재현율)} = \frac{\text{'양성'으로 분류하여 맞춘 데이터 건수}}{\text{실젯값이 '양성'인 데이터 건수}} = \frac{TP}{TP + FN}$$

재현율은 양성을 음성으로 잘못 판단하면 문제가 되는 경우에 사용합니다.

정밀도 ↔ 재현율

- 정밀도와 재현율은 상호 보완적인 평가 지표이며 서로 반대되는 성격을 가지고 있어 트레이드오프
- 어느 한쪽의 평가 지표를 강제로 높이면 한쪽의 평가 지표는 떨어 지기 쉬움

f1-score (조화 평균)

- f1-score : 정밀도와 재현율을 조합한 평가 지표
- 데이터의 클래스가 불균형 구조일 때, 정밀도와 재현율은 어느 한쪽으로도 치우지지 않고 모델의 성능 평가 가능

$$\text{f1-score(조화 평균)} = 2 \times \frac{precision \times recall}{precision + recall}$$

모델 테스트 및 평가하기

의사결정 트리와 성능 비교하기

```
1  from sklearn.tree import DecisionTreeClassifier
2  dt = DecisionTreeClassifier(random_state = 42)
3  dt.fit(X_train, y_train)
4  print(dt.score(X_train, y_train))
5  print(dt.score(X_test, y_test))
```

```
1.0
0.7583333333333333
```

성능 약 76%

모델 테스트 및 평가하기

의사결정 트리와 성능 비교하기

- 의사결정 트리에서 **과대적합을 줄이는 옵션** 적용하기
- 과대적합 : 모델이 훈련 데이터에 과도하 게 학습되어 훈련 데이터에서는 좋은 성능을 보이지만, 다른 데이터가 들어오게 되면 정확도가 떨어지는 현상

'max_depth'는 과대적합(overfitting) 문제를 방지하기 위해 사용됩니다. 트리의 깊이가 깊어질수록 복잡한 모델이 되므로, 'max_depth'를 작게 설정하면 모델의 복잡도가 줄어들어일반화성능이향상됩니다.

```
1  from sklearn.tree import DecisionTreeClassifier
2  dt = DecisionTreeClassifier( max_depth = 3, random_state = 42)
3  dt.fit(X_train, y_train)
4  print(dt.score(X_train, y_train))
5  print(dt.score(X_test, y_test))
```

의사결정트리의 최대 깊이를 설정하는 옵션

```
0.9104166666666667
0.7833333333333333
```

테스트 성능 78%로 더 좋아짐

의사결정 트리와 4% 정도 차이밖에 없다고 할 수 있으나옵션을조정하면차이가더줄어들긴합니다)

데이터가 많아지고 속성이 더 복잡하면 랜덤 포레스트의 힘은 더 강력해집니다.

Random Forest 문제 해결 과정

문제 정의하기	어떤 영화가 흥행할까?
↓	
데이터 불러오기	영화진흥위원회 자료를 이용한 영화 DACON 데이터 불러오기
↓	
데이터 처리하기	데이터셋 확인하기 데이터 결측치를 중간값으로 채우기
↓	
모델 학습하기	랜덤 포레스트로 학습하기
↓	
모델 테스트 및 평가하기	테스트 데이터로 평가하기

1. 이번 활동에서 해결할 문제는 무엇이었나요?

영화가 흥행인지 아닌지 분류하는 문제입니다.

2. 이 문제 해결에 필요한 데이터셋은 무엇이었나요?

인공지능 경진대회 플랫폼인 DACON에서 제공한 '영화 관객수'의 'movies_train.csv' 파일을 사용합니다.

3. 모델 학습에 사용된 알고리즘은 무엇이었나요?

랜덤 포레스트 알고리즘을 사용했습니다.

4. 모델 학습을 위해 어떤 처리를 했나요?

영화 관객 수를 기준으로 '흥행'과 '비흥행'을 구분하여 'hit'라는 종속변수를 생성했고 배급사, 장르, 상영 등급, 감독의 전작 작품 수, 스태프 수, 주연 배우 수를 독립변수로 설정하였으며 독립변수 중 범주형 데이터는 원-핫 인코딩했습니다.

5. 활동을 마치며 새롭게 알게 된 용어를 정리해 보세요.

부트샘플링	
보팅	
배깅	
앙상블	
부스팅	

원본 데이터셋에서 무작위로 중복 추출을 허용하여 동일한 사이즈의 데이터셋을 여러 개 만드는 부트스트랩 샘플링 방식을 이용한 앙상블 모델 중 랜덤 포레스트를 사용하여 영화의 흥행 여부를 분류해 보았습니다.
종속 변수는 영화의 흥행 여부 속성으로 설정하였고 범주형 데이터는 원-핫 인코딩하여 전처리했습니다.
영화에 흥행 요소를 알아보기 위해 pairplot과 히트맵을 그려본 결과, 영화 관객 수와 상관관계가 높은 속성은 스태프 수이었으며 주연 배우 수는 상관관계가 낮음을 확인하였습니다.
전체 데이터 중에서 흥행한 영화를 얼마나 정확하게 예측했는지 알아보려면 정확도를 평가 지표로 설정할 수 있습니다.
위험을 감수하더라도 전체 흥행 영화 중에서 흥행 영화를 예측해 큰 수익을 얻고자 한다면 재현율이 높아야 하고, 큰 수익을 놓여도 큰 손해는 피하고 싶다면 정밀도가 높아야 합니다.
해결하고자 하는 문제에 따라 모델의 성능을 확인하는 데 적절한 평가 지표를 적용하는 것이 중요합니다.