“웹툰 원작시대”

웹툰의 드라마, 영화 예측

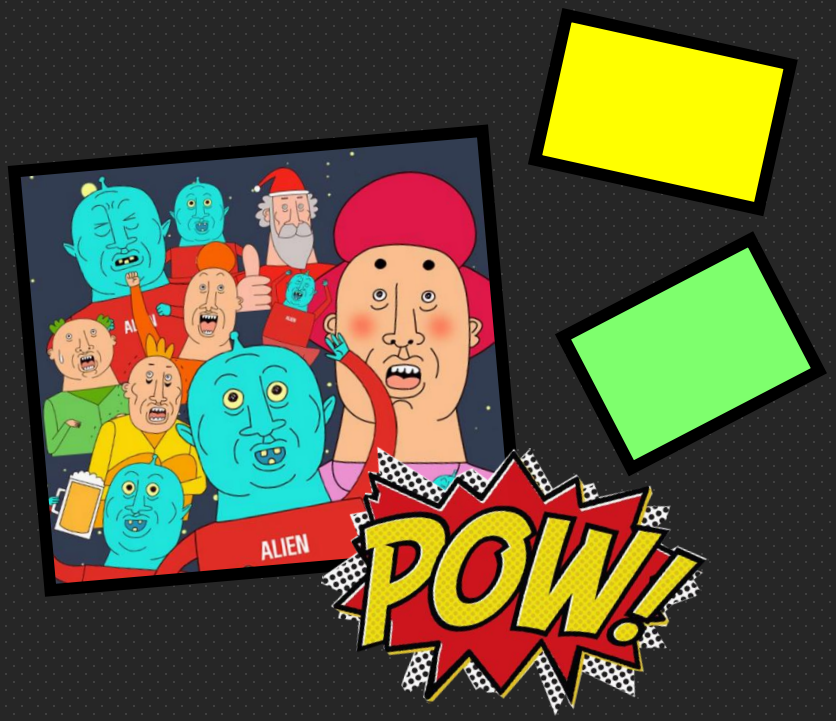
크롤링
활용하기

데이터관리와 지적경영 Final Project

영어영문학과 2012130844 조호신
중어중문학과 2014130751 박혜원
중어중문학과 2015131142 신재원
보건환경융합과학 2015250336 김연주

목차

CONTENTS

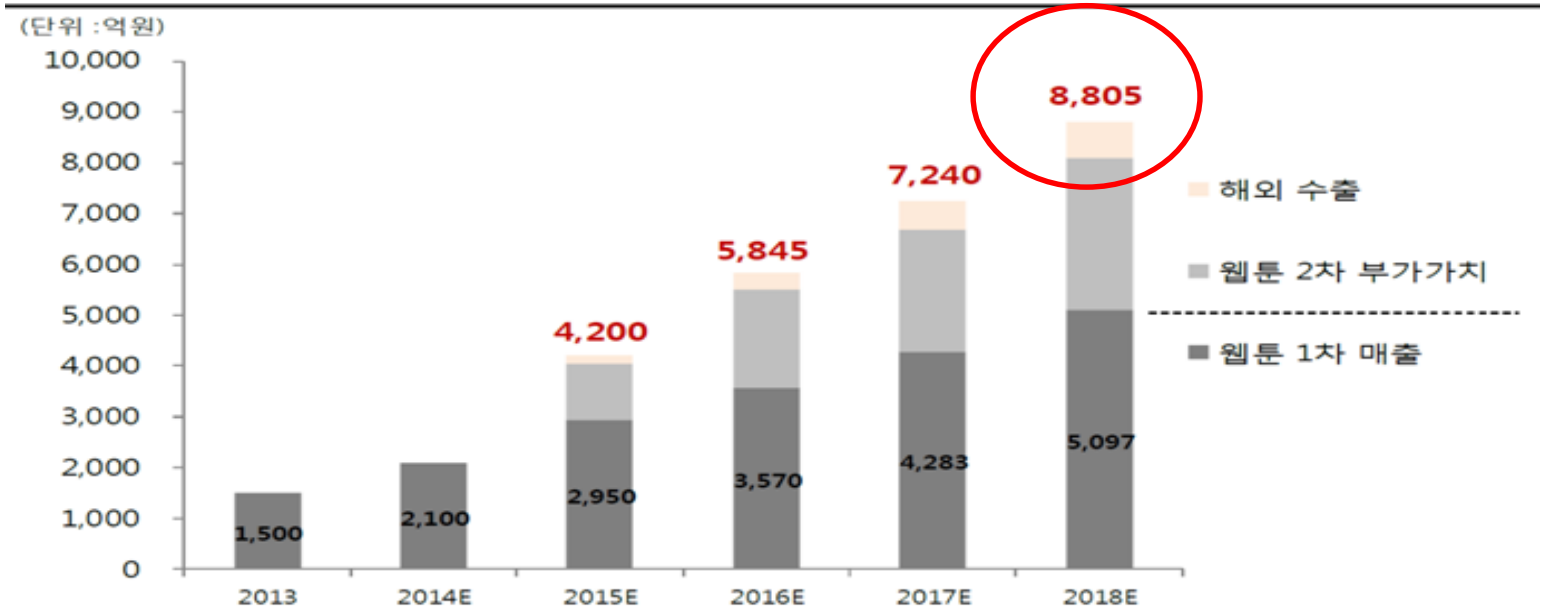


1 주제 선정 배경

웹툰 시장의 성장 및 발전

웹툰 콘텐츠의 상업적 예상 가치 창출 의미 有

" 2018 국내 웹툰 시장 규모 8,805억 원"



출처 : 한국콘텐츠진흥원

웹툰의 파격적인 성장세로 웹툰이 브라운관까지 이어지는 추세

➡ **Then, 웹툰의 영화/드라마화(상업화) 가능성?**

1 주제 선정 배경

웹툰으로 시작해서
드라마, 영화까지!

2003년 강풀의 '순정만화'를 필두로 주목 받기 시작

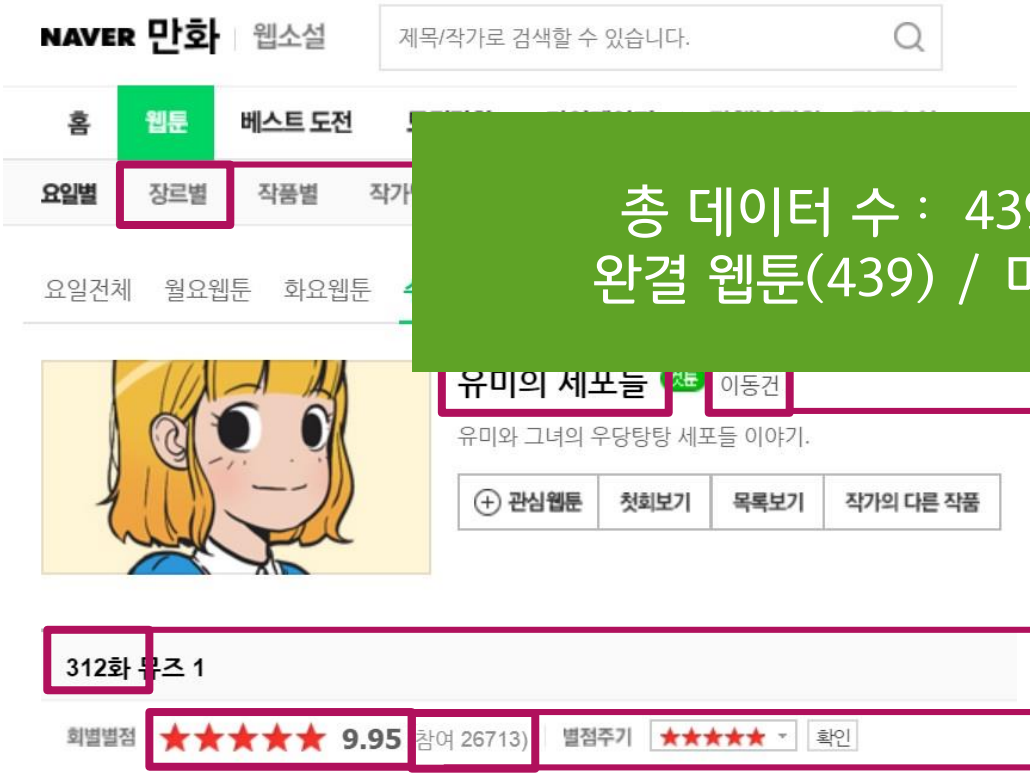
영화 '은밀하게 위대하게'(2013)
tvN드라마 '미생'(2014)

비단 영화나 드라마뿐 아니라
게임, 뮤지컬, 웹드라마로
그 외연도 넓어짐

2 데이터 수집 목록 & 방법

크롤링
[crawling]

무수히 많은 컴퓨터에 분산 저장되어 있는 문서를 수집하여
검색 대상의 색인으로 포함시키는 기술



총 데이터 수 : 439 + 170 = 609
완결 웹툰(439) / 미완결 웹툰(170)

[장르]
daily / comic / fantasy / action / drama / pure
… tical / sports

[작가명]
+ [종이책 출판 여부]
+ [완결 여부] 완결 : 1 / 미완결 : 0

[완결된/현재까지의 최종 회 수]

[별점] 모든 회차별 정보 평균

[참여 인원] 모든 회차별 정보 평균

[종속 변수] : 영화드라마화 有 1/ 無 0 (수작업)

2 데이터 수집 목록 & 방법

크롤링으로 제작한 Dataframe

```
for i in range(1,number):
    url='http://comic.naver.com/webtoon/detail.nhn?titleId=%s&no=%s&weekday=%s'% (link_value["value"][j],i,link_value["weekday"][j])
    print(url)
    req=requests.get(url)
    html=req.text


    soup=BeautifulSoup(html,'html.parser')

    #CSS selector를 사용하여 가져오기 (단, tag와 값이 같이 반환됨 ex)<h3>1141. 코딩</h3>)
    titles=soup.select('div.view > h3')
    ratings=soup.select('div.rating_type4 > span > strong')
    rating_clicks=soup.select('#topTotalStarPoint > span.pointTotalPerson > em')
    bigtitle=soup.select('div.remote_cont > a')
```

문아
문유
므시미르
미결
미라클! 용사님
미래소녀
미션 임파서블
미숙한 친구는 G구인
미스터리 호러 지하...
미쳐 날뛰는 생활툰
미호이야기
바나나걸
바람의 색
바람이 머무는 난
바로잡는 순애보
바이올린처럼.
반월
빵점동맹
뻐뚤빼뚤해도 괜찮아
사노라면
사또(Satto)
사랑in
사랑을 연기하다
사랑의 아쿠아리움
사랑의 외계인
사랑일까
사이드킥
사이드킥2
삶의 발톱
삼국전투기
삼봉이발소
새끼손가락
새벽9시
새와 같이
스튜디오 짭쪼롬
스페이스 차이나드레스
스페이스 킹
스펙트럼 분석기
시간의 섬
시노딕
시큼새큼
시타를 위하여
식스센스
신령
실질객관동화
실질객관영화
심부름센터 K
심심한 마왕
심연의 하늘 시즌 ...
심장이뛰다
싸우자귀신아
언더프린
언터처블
얼룩말
엄마와 딸 x2
에피소드메이비
여중생A
여탕보고서
역전! 야매요리
연 시즌2
연
연애세포
연옥님이 보고계셔
열무가 익어간다
열쇠줍는아이
열아홉스물하나
열일곱
영수의 봄

3 데이터 전처리

(1) 데이터 통합

concat, merge

```
df_final = pd.concat([df1,df2,df3], axis =1)
df_final.reset_index()
df_final.head()
```

```
all_x = pd.merge(all_data,gt_df, on = 'label')
all_x.reset_index()
```

to 변수 데이터 합치기

	label	Unnamed: 0_x	별점	참여	Unnamed: 0_y	value	artist	ratings	episode	omnibus	...	drama	pure	sensibility	thrill	historical	sports	pieces
0	10월 28일	32.0	9.75303	18571.12121	0	702423	천정학	9.54	0	0	...	0	0	0	1	0	0	3
1	10월 28일	32.0	9.75303	18521.75758	0	702423	천정학	9.54	0	0	...	0	0	0	1	0	0	3
2	10월 28일	32.0	9.75303	18521.75758	0	702423	천정학	9.54	0	0	...	0	0	0	1	0	0	3
3	2013 전설의고향	49.0	9.22840	18131.56000	5	574303	웹툰작가	9.20	0	1	...	0	0	0	1	1	0	11

3 데이터 전처리

(2) 데이터 변환

groupby

```
df1 = df.groupby("label")['Unnamed: 0'].idxmax()
df2 = df.groupby("label")['별점'].mean()
df3 = df.groupby("label")['참여'].mean()
```

to '제목'중심으로 마지막 회차수, 총회차마다의 별점평균, 별점 참여수 평균 정리

label	Unnamed: 0	별점	참여
0.0	0	0.000000	0.000000
10월 28일	32	9.753030	18571.121212
17살, 그 여름날...	5	9.920000	5247.833333
2011 미스테리 ...	30	9.260968	17378.838710
2012 지구가 멸...	34	9.280286	37434.028571

3 데이터 전처리

(3) 데이터 축소

del, drop

```
all_df.drop_duplicates(inplace=True)
all_df =all_df.drop(['Unnamed: 0_x','Unnamed: 0_y', 'ratings','value','artist'], axis=1)
all_df.columns
```

to 중복값 없애기 / 필요없는 컬럼 없애기

```
Index(['label', '별점', '참여', 'episode', 'omnibus', 'story', 'daily', 'comic',
       'fantasy', 'action', 'drama', 'pure', 'sensibility', 'thrill',
       'historical', 'sports', 'pieces', '영화', '드라마', '도서', 'N_clicks'],
      dtype='object')
```

(4) 결측치 제거

isnull().sum()/len()

```
all_df.isnull().sum()/len(all_df)
```

```
label           0.0
Unnamed: 0_x    0.0
별점              0.0
참여              0.0
Unnamed: 0_y    0.0
value           0.0
artist          0.0
ratings         0.0
episode         0.0
omnibus         0.0
story           0.0
daily           0.0
comic           0.0
fantasy         0.0
action          0.0
drama           0.0
pure            0.0
sensibility     0.0
thrill          0.0
historical      0.0
sports          0.0
pieces          0.0
영화              0.0
드라마             0.0
도서              0.0
dtype: float64
```

결측치 無
제거 無

3 데이터 전처리

(5) 데이터 정규화

minmaxscaler

```
from sklearn.preprocessing import MinMaxScaler
minmax_scaler = MinMaxScaler().fit(all_df[['ratings']])
minmax_mat= minmax_scaler.transform(all_df[['ratings']])
minmax_mat
```

```
all_df['N_clicks'] = minmax_mat[:,0:1]
all_df.columns
all_df.isnull().sum() /len(all_df)
all_df
```

to 스펙트럼이 넓은 값들을 정규화를 통해 차이를 줄여줌

abel	Unnamed: 0_x	별점	참여	Unnamed: 0_y	value	artist	ratings	episode	omnibus	...	pure	sensibility	thrill	historical	sports	pieces	영화	느라마	도서	N_clicks
28일	32.0	9.753030	18571.121210	0	702423	천정학	9.54	0	0	...	0	0	1	0	0	3	0	0	0	0.927184
28일	32.0	9.753030	18521.757580	0	702423	천정학	9.54	0	0	...	0	0	1	0	0	3	0	0	0	0.927184
28일	32.0	9.753030	18521.757580	0	702423	천정학	9.54	0	0	...	0	0	1	0	0	3	0	0	0	0.927184
전설 고향	49.0	9.228400	18131.560000	5	574303	웹툰작가	9.20	0	1	...	0	0	1	1	0	11	0	0	0	0.872168
사이	19.0	9.868500	38074.450000	7	666516	웹툰작가	9.88	0	1	...	1	0	0	0	0	11	0	0	0	0.982201

3 데이터 전처리

(6) 데이터 이산화

np.where

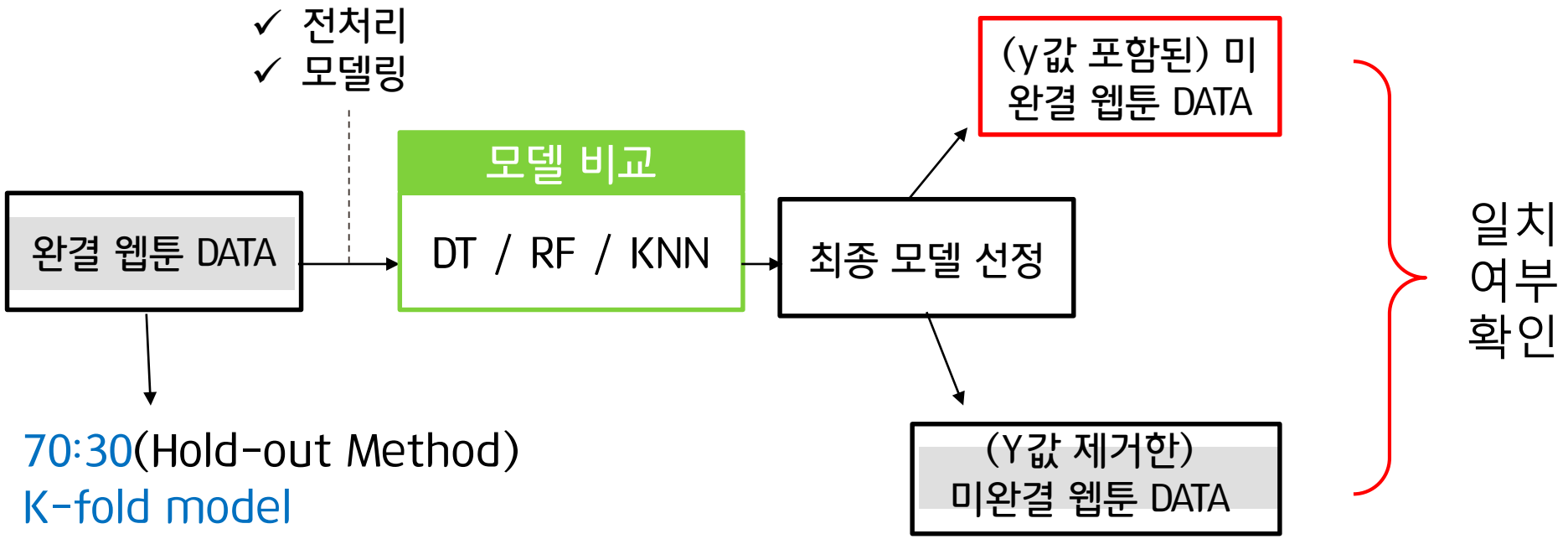
```
df1=train.df["MOVIE"]
df2=train.df["DRAMA"]
df4=df1+df2
train_df["SUM"]=df4
train_df.head()
train_df["OSMU"]=np.where(train_df["SUM"]>0,1,0)
train_df.columns
```

to Dataframe에 column 추가

```
Index(['label', 'ratings', 'clicks', 'N_clicks', 'finish', 'pieces', 'episode',
       'omnibus', 'story', 'daily', 'comic', 'fantasy', 'action', 'drama',
       'pure', 'sensibility', 'thrill', 'historical', 'sports', 'MOVIE',
       'DRAMA', 'BOOK', 'N_ratings', 'N_pieces', 'SUM', 'OSMU'],
      dtype='object')
```

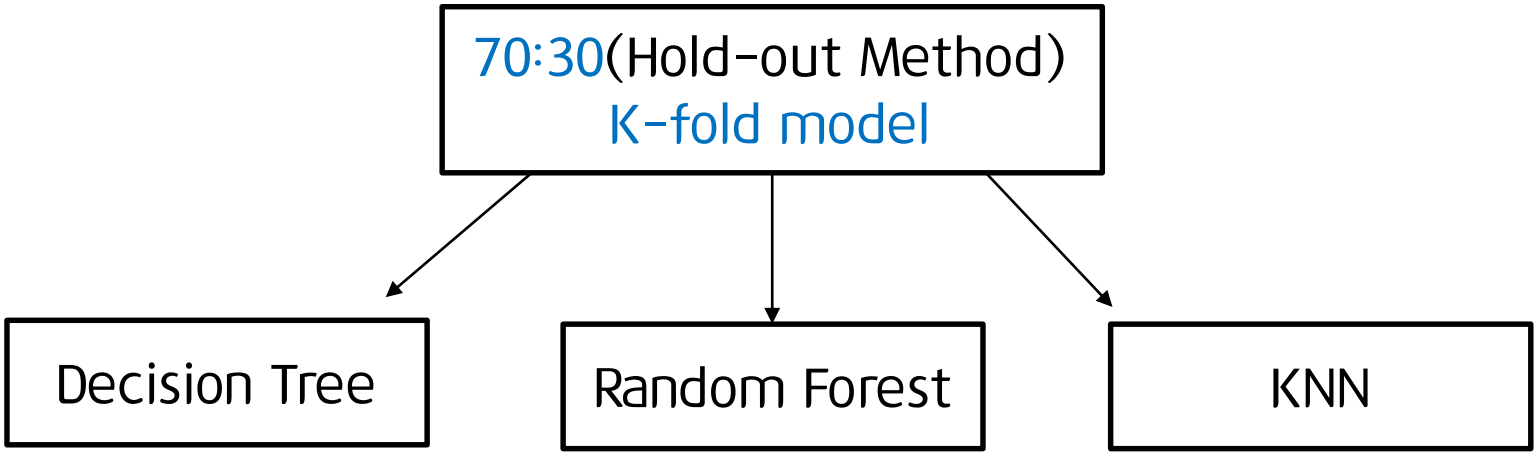
4 분석 방법

적용 알고리즘



4 분석 방법

모델링 평가


70:30(Hold-out Method)
K-fold model
Decision Tree
Random Forest
KNN

4 분석 방법

모델링 평가 (70:30)

Decision Tree

```
훈련 데이터 결과:  1.0
검증 데이터 결과:  0.8636363636363636
훈련 데이터 결과(d=4):  0.9250814332247557
검증 데이터 결과(d=4):  0.9015151515151515
```

	n	training accuracy	test accuracy
0	1	0.889251	0.946970
1	2	0.889251	0.946970
2	3	0.892508	0.924242
3	4	0.925081	0.901515
4	5	0.941368	0.909091
5	6	0.954397	0.878788
6	7	0.973941	0.878788
7	8	0.986971	0.901515
8	9	0.996743	0.863636

DT의 최적 n=4

Accuracy 적용한 데이터 결과
0.925081(Training)
0.901515(Test)

4 분석 방법

모델링 평가 (70:30)

훈련 데이터 결과(e=2): 0.9413680781758957
검증 데이터 결과(e=2): 0.9242424242424242
훈련 데이터 결과(e=5): 0.9674267100977199
검증 데이터 결과(e=5): 0.8787878787878788

Random Forest

	n	training accuracy	test accuracy
0	1	0.957655	0.863636
1	2	0.941368	0.924242
2	3	0.967427	0.886364
3	4	0.960912	0.924242
4	5	0.967427	0.878788
5	6	0.970684	0.924242
6	7	0.977199	0.901515
7	8	0.970684	0.924242
8	9	0.983713	0.924242

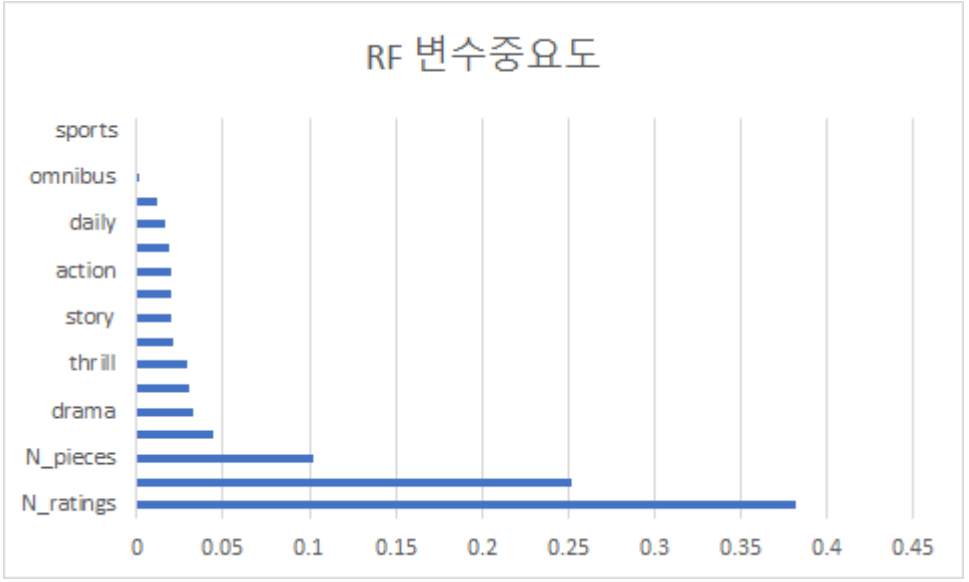
RF의 최적 n=2

Accuracy 적용한 데이터 결과
0.941368(Training)
0.924242(Test)

4 분석 방법

모델링 평가 (70:30)

Random Forest



변수 중요도(영향력)가 높다고 평가된 변수

N_pieces N_ratings N_clicks Story Fantasy
Action Drama Book

4 분석 방법

모델링 평가 (70:30)

KNN

훈련 데이터 결과: 0.925081
검증 데이터 결과: 0.901515

	n	training accuracy	test accuracy
0	1	0.889251	0.946970
1	2	0.889251	0.946970
2	3	0.892508	0.924242
3	4	0.925081	0.901515
4	5	0.941368	0.909091
5	6	0.954397	0.878788
6	7	0.973941	0.878788
7	8	0.986971	0.901515
8	9	0.996743	0.863636

KNN의 최적 n=4

Accuracy 적용한 데이터 결과
0.925081(Training)
0.901515(Test)

4 분석 방법

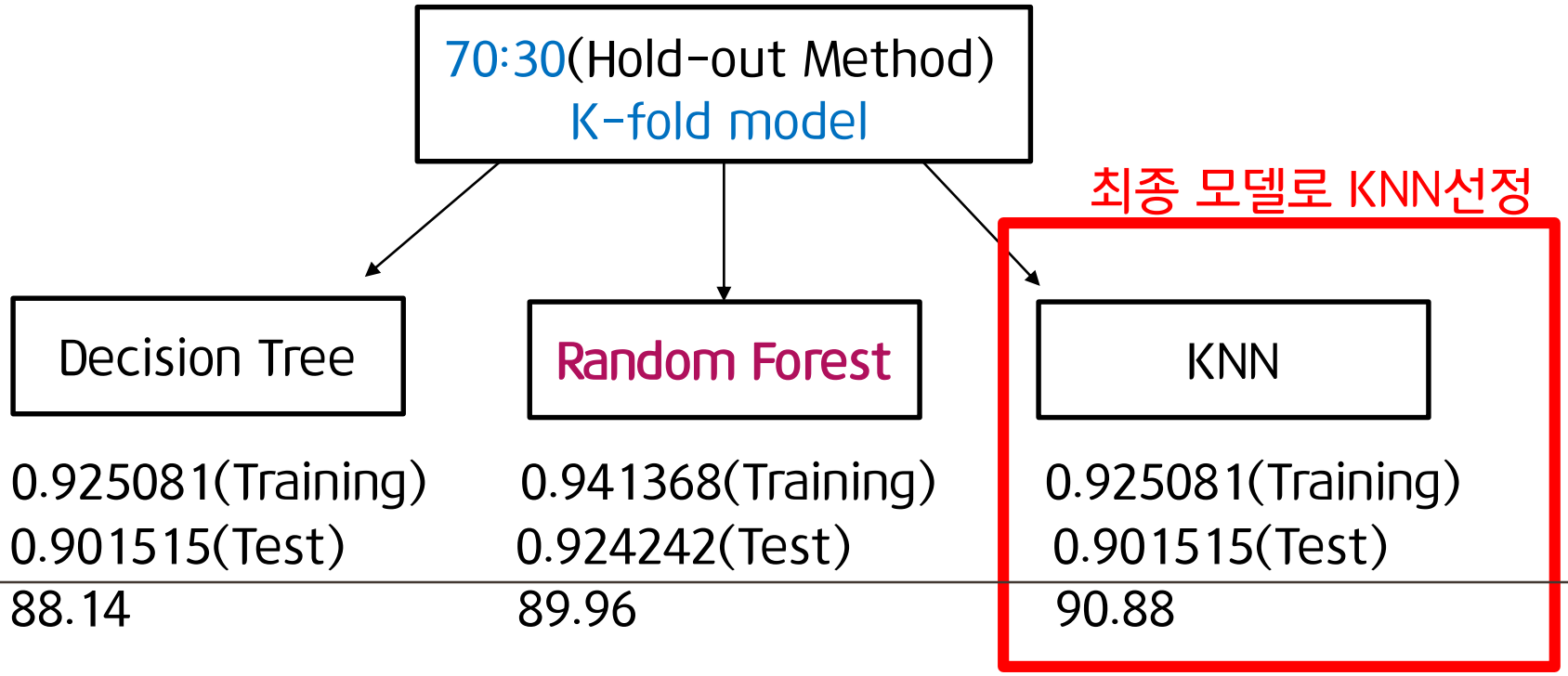
모델링 평가 K-fold

K-fold(교차검증) & DT/RF/KNN 비교

Decision Tree	Random Forest	KNN
88.14	89.96	90.88

4 분석 방법

모델링 평가



예측하면서, KNN사용한 결과값(1)을 1개만 예측(본래 총 4개)
Thus, **예측은 Random Forest 인 forest.predict를 사용**함

4 분석방법

모델링 적용 결과

최종 모델인
Random Forest

나무 개수를 2개로 제한하여 y값을 없앤 test data를 적용해서 예측값 생성

```
#################### predict ####################

y_pred = forest.predict(all_x_test)

from sklearn.metrics import accuracy_score

print("테스트 세트에 대한 예측값  ₩n {}".format(y_pred))
print("테스트 세트의 정확도 : {:.2f}".format(forest.score(all_x_test, all_y_test)))

import collections
print(collections.Counter(y_pred))
# 0: 미완결 ; 1: 완결
```

```
테스트 세트에 대한 예측값
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
테스트 세트의 정확도 : 0.95
Counter({0: 166, 1: 4})
```

KNN은
결과값(1)을 1개로만 예측

RF 결과
결과값(1)을 정확히 **4개**로 예측

4 분석 방법

모델링 적용 결과

최종 모델인
Random Forest

나무 개수를 2개로 제한하여 y값을 없앤 test data를 적용해서 예측값 생성

```
y_index = np.where(y_pred == 1)
print(y_index)

print(train_df['label'][459])
print(train_df['OSMU'][459])

print(train_df['label'][493])
print(train_df['OSMU'][493])

print(train_df['label'][511])
print(train_df['OSMU'][511])

print(train_df['label'][528])
print(train_df['OSMU'][528])
```

```
(array([20, 54, 72, 89], dtype=int64),)
공대생 너무만화
0
바른연애 길잡이
0
스피릿 핑거스
0
야채호빵의 봄방학
0
```

RF가 말하는 1값의 웹툰들

```
all_y_test = np.where(all_y_test ==1)
print(all_y_test)
```

```
(array([27, 37, 48, 74, 77], dtype=int64),)
```

```
print(train_df['label'][466])
print(train_df['OSMU'][466])

print(train_df['label'][476])
print(train_df['OSMU'][476])

print(train_df['label'][513])
print(train_df['OSMU'][513])

print(train_df['label'][516])
print(train_df['OSMU'][516])
```

```
기기괴괴
1
놓지마 정신줄 시즌2
1
신과함께 (재)
1
신암행어사
1
```

실제 1값의 웹툰들

개수는 정확히 예측
But, 그에 맞는 **웹툰의 제목은 상이하게 도출**됨

5 결론

포털사이트 + 웹툰 작가들이 만들어낼 수 있는 수익성은 얼마나 될까?
실제로, 웹툰 작가들의 꿈인 웹툰의 '드라마/영화화'
∵ 큰 돈을 벌어들일 수 있기 때문

세계 가장 빠르게 모방하고 움직이는 중국도 웹툰의 수익화 고려

그래서 알아본 수익성 검증 !

결과 : '공대생 너무만화','바른연애 길잡이','스피릿 핑거스',
'야채호빵의 봄방학' 수익화 모델로 만들어질 가능성이 있다 !

6 Team members' talk

호신's saying

주제를 여러번 바꾸게 되어 시간이 조금 부족했지만 팀원들 모두 열심히 해준 덕분에 프로젝트를 완성할 수 있었습니다. 모두 고생하셨습니다.

재원's saying

프로젝트를 통해서 데이터를 직접 다뤄보고 모델링도 해보고 하면서 수많은 에러들과 싸우느라 너무나 스트레스를 받기도 했지만.. 에러를 해결하고 원하는 결과값이 딱 나왔을 때는 정말 행복했습니다.. 크롤링부터 전처리, 모델링 하나하나 하고 수작업까지 해가면서 밤새가며 고생한 팀원들 모두모두 너무 고생했고 감사합니다!!!ㅜㅜ 그리고 어떻게 보면 도전일 수 있는 이런 강의를 맡고 열어주셔서 정말 감사합니다 어디 가서 이렇게 은 시간 안에 이렇게 많은 내용을 제대로 배울 수 있을까요.. 교수님 덕분에 많이 배우고 갑니다!!

5 Team members' talk

혜원's saying

교수님, 3조 팀원분들 한 학기동안 고생하셨습니다. 수업이나 과제를 통해 데이터사이언티스트들의 고충을 몸소 느낄 수 있었고, 저에게 데이터 관련 직군이 맞을지 진지하게 고민해볼 수 있었습니다. 또한 코드를 돌려보면서 쾌감과 짜증나는 감정의 양 극단을 왔다갔다하며, 좋은 자극을 받을 수 있었지만 동시에 스트레스가 많이 쌓이기도 했네요. 종강하면 당분간 푹 쉬고 이번 프로젝트에서 제일 난항을 겪었던 크롤링에 대해 공부해봐야겠습니다. 한 학기 정말 감사했습니다.

연주's saying

한 학기동안 파이썬과 머신 러닝 이론들을 한꺼번에 배우면서 양도 많았지만 그만큼 얻어가는 것도 많았습니다. 그리고 그것을 적용해보는 팀프로젝트는 의미 있는 과제였다고 생각합니다. 팀플에 정말 열심히 임해주시는 팀원들께도 정말 감동받았습니다! 주제 고민만 해도 엄청났지만 결과적으로 좋은 주제를 얻을 수 있었고 좋은 결과도 냈다고 생각합니다. 한 학기 동안 감사했습니다 교수님 팀원분들!! ☺

감사합니다