

빅 데이터 분석 최종 보고서

보고서 및 논문 윤리 서약

1. 나는 보고서 및 논문의 내용을 조작하지 않겠습니다.
2. 나는 다른 사람의 보고서 및 논문의 내용을 내 것처럼 무단으로 복사하지 않겠습니다.
3. 나는 다른 사람의 보고서 및 논문의 내용을 참고하거나 인용할 시 참고 및 인용 형식을 갖추고 출처를 반드시 밝히겠습니다.
4. 나는 보고서 및 논문을 대신하여 작성하도록 청탁하지도 청탁받지도 않겠습니다.

나는 보고서 및 논문 작성 시 위법 행위를 하지 않고, 명지인으로서 또한 공학인으로서 나의 양심과 명예를 지킬 것을 약속합니다.



학 과 : 융합소프트웨어학부 데이터테크놀로지 전공

과 목 : 빅데이터프로그래밍

담당교수 : 권동섭 교수님

강좌 번호: 5951

학 번 : 60160223

이 름 : 남종호 (서명) 남종호

문제 정의

영화 제작은 큰 규모의 사업이기 때문에 몇 가지 요인을 통해 이 영화가 수익일지, 범작일지, 졸작일지 혹은 평균 이상의 작품일지, 평균 미만의 작품일지 등을 예측할 수 있는 모델을 만든다면 소위 말하는 흥행 공식을 제시하는 것이며 이를 통해 제작사들의 수익 극대화에 과학적인 도움을 줄 수 있게 된다. 또한, 소비자들에게 위 모델은 졸작은 거르고 명작을 선택할 수 있게 하는 합리적인 기준이 될 수 있다.

따라서 필자는 web crawler를 이용하여 영화 데이터를 모으고, 이 데이터를 전처리하고 가공하여 classification 알고리즘으로 학습시켜 영화의 작품성을 분류하는 모델을 만들고자 한다. 하지만, 영화는 예술의 영역이기도 하기 때문에 하나의 공식에 따른 획일화는 문화적으로 바람직하지 않지만 이 프로젝트에서는 어떠한 사회·문화적인 파급도 고려하지 않는 걸로 한다.

시스템 아키텍처

우선, 자동화된 데이터 수집을 위해 가장 먼저 떠오른 library는 python의 beautifulsoup이었습니다. 하지만, beautifulsoup는 js가 동적으로 rendering 한 객체에는 접근할 수 없고 한정적인 html요소, 키보드 입력 같은 이벤트 불가 등의 한계점이 많았습니다. 그러다 위에서 언급한 모든 한계를 극복한 selenium이라는 library를 교수님이 첨부해 준 자료를 통해 알게 됐고 그 사용법에 대해서 학습했습니다.

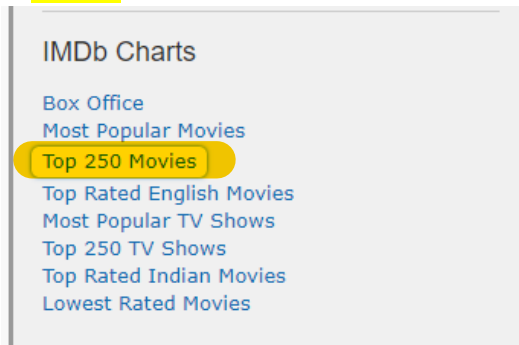
학습한 내용을 토대로 영화 데이터 크롤링이라는 컨셉에 맞게 저만의 crawler를 만들어 자료를 수집해 csv로 저장하였습니다. 수집된 Movies_data는 title, genre, running_time, director, screening_rat(상영 등급), 감독이 참여한 모든 영화의 개수, 매인 주연의 출연작 개수, rating 이렇게 8개의 column으로 이루어진 schema를 갖게 했습니다.

그 뒤, pyspark.sql, pyspark.ml 패키지를 이용하여 Random Forest classifier를 통해 classification했습니다. 제가 Random Forest를 선택한 이유는 일단 영화의 작품성 예측이라는 주제는 여러 개의 features를 가지고 movies를 몇 개의 class로 분류하는 작업 즉, supervised learning && classification입니다. 이에 해당하는 알고리즘은 logistic regression with multiclass, naïve bayes, decision tree, svm, k-nearest Neighbor, dnn, ensemble Learning 등이 있지만, 저는 솔직히 딥러닝에 대한 지식이 거의 없고 svm, k-nearest Neighbor 같은 알고리즘은 학습한 적이 없었기 때문에 혼자서 제한된 시간 안에

부정확한 정보를 보며 새로운 알고리즘을 학습하는 것보다 지난 학기 알고리즘 시간에 학습한 강력한 알고리즘인 ensemble 중 Random Forest를 사용하는 것이 지금 제가 할 수 있는 최선이며 가장 좋은 output을 낼 수 있다고 생각했습니다. 차후에 관련 code가 git에 올라가고 인공지능에 대한 학습을 계속해 나간다면, 위에 언급한 알고리즘들도 시도해 나온 결과들을 기록해 볼 예정입니다.

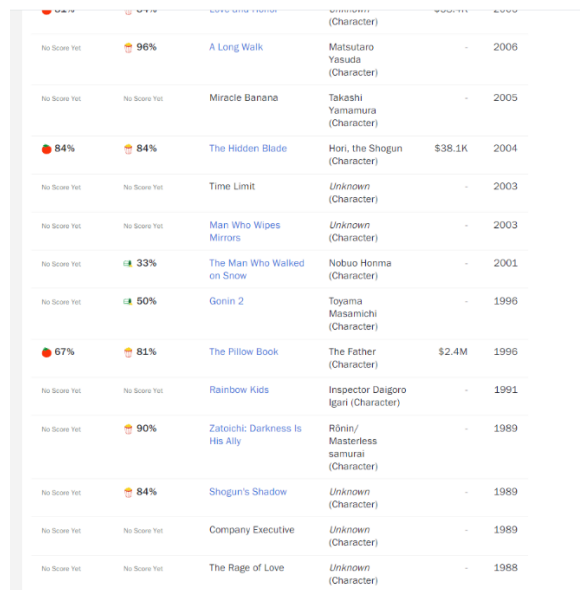
데이터 수집 방법

우선 과목에 '빅 데이터'가 들어가는 만큼 최대한 데이터를 모으고 싶었습니다. 또한, 애초에 제가 모으고자 하는 schema에 맞는 data도 인터넷에서는 구하기 어려웠겠지만 교수님께서 데이터 수집의 자동화를 강조하셨기 때문에 크롤러를 만들기 전에 어디서 movies_data를 긁어야 할 지 찾아야 했습니다. 그렇게 찾다가 찾은 movieDB 사이트가 naver 영화, IMDB, rotten tomatoes였습니다. 하지만, **IMDB**는 크롤링을 위한 영화 목록이



많아야 250개밖에 되지 않는 검색 위주의 사이트


였고, **rotten tomatoes**는 19000천개 정도 영화 목록에 접근이 가능했으나 감독의 연출작

A screenshot of a table from the Rotten Tomatoes website. The table lists movies with their scores, titles, directors, and years. The first row is 'A Long Walk' with a score of 96%, directed by Matsutaro Yasuda, from 2006. The second row is 'Miracle Banana' with a score of 84%, directed by Takashi Yamamura, from 2005. The third row is 'The Hidden Blade' with a score of 84%, directed by Hori, the Shogun, from 2004. The fourth row is 'Time Limit' with a score of 84%, directed by Unknown, from 2003. The fifth row is 'Man Who Wipes Mirrors' with a score of 84%, directed by Unknown, from 2003. The sixth row is 'The Man Who Walked on Snow' with a score of 33%, directed by Nobuo Honma, from 2001. The seventh row is 'Gonin 2' with a score of 50%, directed by Toyama Masamichi, from 1996. The eighth row is 'The Pillow Book' with a score of 67%, directed by The Father, from 1996. The ninth row is 'Rainbow Kids' with a score of 81%, directed by Inspector Daigoro Igari, from 1991. The tenth row is 'Zatoichi: Darkness Is His Ally' with a score of 90%, directed by Rōnin/Masterless samurai, from 1989. The eleventh row is 'Shogun's Shadow' with a score of 84%, directed by Unknown, from 1989. The twelfth row is 'Company Executive' with a score of 84%, directed by Unknown, from 1989. The thirteenth row is 'The Rage of Love' with a score of 84%, directed by Unknown, from 1988.

개수와 주연의 출연작 개수를 count할 때,

위와 같은 table의 row를 죄다 count했어야 돼서, 크롤링 속도가 안 그래도 느린데 이런 작업까지 영화 하나마다 해야 하는 것은 효율적이지 않다고 생각했습니다. 하지만 **naver**

필모그래피 총 15건

영화 디렉토리는  이처럼 value 하나 긁어오는 작업으로 감독의 연출작 개수나 주연의 출연작 개수를 알 수 있습니다. 따라서, 저는 naver 영화 평점순 2

1991	인사이드 르윈	★★★★★ 8.16	평점주기	↓ 1
1992	콰이어트 플레이스 2	★★★★★ 8.16	평점주기	↓ 1
1993	킹메이커	★★★★★ 8.15	평점주기	↓ 1
1994	어디선가 누군가에 무슨 일이 생기면 틀림없이 나타난...	★★★★★ 8.15	평점주기	↓ 1
1995	내 첫사랑을 너에게 바친다	★★★★★ 8.15	평점주기	↓ 1
1996	원라인	★★★★★ 8.15	평점주기	↓ 1
1997	프란시스 하	★★★★★ 8.15	평점주기	↓ 1
1998	홀드 가드	★★★★★ 8.15	평점주기	↓ 1
1999	차일드 44	★★★★★ 8.15	평점주기	↓ 1
2000	귀 없는 토끼	★★★★★ 8.15	평점주기	↓ 1

천개의 영화

◀ 이전 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |

와 네이버

디렉토리에 존재하는 미국에서 만든 영화 약 5만개 미국 (52296)에 접근하여 크롤링 할 계획을 세웠습니다

먼저, 저는 네이버 영화에서 제공하는 평점순 2천개의 영화를 crawling했습니다.

순위	영화명	평점	변동폭
1	밥정	★★★★★ 9.61	평점주기 - 0
2	그린 북	★★★★★ 9.60	평점주기 - 0
3	가버나움	★★★★★ 9.59	평점주기 - 0
4	디지몬 어드벤처 라스트 에볼루션 : 인연	★★★★★ 9.55	평점주기 - 0
5	원더	★★★★★ 9.53	평점주기 - 0
6	베일리 어게인	★★★★★ 9.53	평점주기 - 0
7	아일라	★★★★★ 9.52	평점주기 - 0
8	극장판 바이올렛 에버가든	★★★★★ 9.51	평점주기 - 0
9	먼 훗날 우리	★★★★★ 9.50	평점주기 - 0
10	당갈	★★★★★ 9.49	평점주기 - 0
11	포드 V 페라리	★★★★★ 9.48	평점주기 - 0
12	그대, 고맙소 : 김호중 생애 첫 팬미팅 무비	★★★★★ 9.48	평점주기 - 0
13	주전장	★★★★★ 9.47	평점주기 - 0
14	소생크 탈출	★★★★★ 9.45	평점주기 - 0
15	터미네이터 2:오로라지널	★★★★★ 9.44	평점주기 - 0
16	라이언 일병 구하기	★★★★★ 9.43	평점주기 - 0
17	클래식	★★★★★ 9.43	평점주기 - 0
18	덕구	★★★★★ 9.43	평점주기 - 0
19	나 홀로 집에	★★★★★ 9.43	평점주기 - 0
20	월-E	★★★★★ 9.42	평점주기 - 0

이와 같은 movie 목록에 접근하여 모든 row를 돌며 50 * 40 pages 총 2000개의 데이터를 가져옵니다. 이 중에 네이버 로그인을 해야 접근이 가능한 영화 데이터는 받을 수 없으므로 대략 1960개 정도의 데이터를 받을 수 있었습니다. 소스코드와 코드의 동작원리, 예외 처리 등등은 git 혹은 과제 첨부파일로 첨부하겠습니다.

다음으로, 네이버 디렉토리에 있는 미국에서 만든 5만개의 영화에 접근해야 하는데, 이

0011 나폴레옹 솔로 - 구사일생 To Trap A Spy , 1964

기대지수 [?]

보고싶어요 3 1 기록 글베요

미국 | 92분 | [해 외] NR [?]

감독 돈 메드포드 출연 로버트 본, 루치아나 파루치 더보기 >

♡ 3

디렉토리에 있는 데이터들은

위처럼 평점이 없고 '보고싶어요 글썬요 지수'밖에 없는 것들이 많았습니다. 따라서 저는 영화의 감독이나 상영 시간 등의 정보는 네이버에서 얻고 평점 정보만 IMDB에서 검색하여 얻는 것으로 정했습니다. 하지만, 아무래도 2가지 tab을 왔다 갔다 하며 검색 기능까지 넣다 보니 한 데이터를 긁어 오는데도 상당한 시간이 걸렸습니다.

또한, 감독 데이터, 감독의 연출작 개수 데이터, 영어 제목이 없는 경우, 네이버 로그인을 해야 데이터 접근이 가능한 경우, 네이버에는 있었으나 IMDB에는 영화가 없는 경우, IMDB에도 rating 정보 없음 등등 변수가 많아 긁어 오지 못하고 skip하는 데이터가 생각보다 많았습니다.

Crawling을 2주 정도 진행했는데 만 2천개 정도의 데이터밖에 모으지 못했습니다. 물론 2주 24시간 내내 모은 것이 아니라 중간에 멈추고 수정하고 하는 과정이 포함되어서 그런 것 같습니다. 크롤링에 관한 소스코드와 자세한 코드 동작 원리, 예외 처리 등등은 git 혹은 과제 첨부파일로 첨부하겠습니다.

데이터 분석 방법

	A	B	C	D	E	F	G	H	I
1		title	genre	running_ti	screening_director	dir_movies	main_role	rating	
2	0	감독	장르	시청률	감독	감독	감독	감독	
3	1	감독	장르	시청률	감독	감독	감독	감독	
4	2	감독	장르	시청률	감독	감독	감독	감독	
5	3	감독	장르	시청률	감독	감독	감독	감독	
6	4	감독	장르	시청률	감독	감독	감독	감독	
7	5	감독	장르	시청률	감독	감독	감독	감독	
8	6	감독	장르	시청률	감독	감독	감독	감독	
9	7	감독	장르	시청률	감독	감독	감독	감독	
10	8	감독	장르	시청률	감독	감독	감독	감독	
11	9	감독	장르	시청률	감독	감독	감독	감독	
12	10	감독	장르	시청률	감독	감독	감독	감독	
13	11	감독	장르	시청률	감독	감독	감독	감독	
14	12	감독	장르	시청률	감독	감독	감독	감독	
15	13	감독	장르	시청률	감독	감독	감독	감독	
16	14	감독	장르	시청률	감독	감독	감독	감독	
17	15	감독	장르	시청률	감독	감독	감독	감독	
18	16	감독	장르	시청률	감독	감독	감독	감독	
19	17	감독	장르	시청률	감독	감독	감독	감독	
20	18	감독	장르	시청률	감독	감독	감독	감독	
21	19	감독	장르	시청률	감독	감독	감독	감독	
22	20	감독	장르	시청률	감독	감독	감독	감독	
23	21	감독	장르	시청률	감독	감독	감독	감독	
24	22	감독	장르	시청률	감독	감독	감독	감독	
25	23	감독	장르	시청률	감독	감독	감독	감독	
26	24	감독	장르	시청률	감독	감독	감독	감독	
27	25	감독	장르	시청률	감독	감독	감독	감독	
28	26	감독	장르	시청률	감독	감독	감독	감독	
29	27	감독	장르	시청률	감독	감독	감독	감독	
30	28	감독	장르	시청률	감독	감독	감독	감독	
31	29	감독	장르	시청률	감독	감독	감독	감독	
32	30	감독	장르	시청률	감독	감독	감독	감독	
33	31	감독	장르	시청률	감독	감독	감독	감독	

크롤링 해 온 데이터를 처음 열면 이와 같이 encoding이 깨져 보입니다. 따라서, 해당 파일을 연결 프로그램 메모장으로 열고 다른 이름으로 저장을 누른 후, 파일 제목 뒤에 .csv 확장자 명을 붙이고 ANSI로 인코딩 type

title	genre	running_time	screening_director	dir_movies	main_role	rating	
밥정	다큐멘터리	82분	전체 관람	박혜령	1개	3개	9.61
그린 북	드라마	130분	12세 관람	피터 패럴	22개	51개	9.6
가버나움	드라마	126분	15세 관람	나딘 라바	11개	1개	9.59
디지몬 어	애니메이션	114분	12세 관람	타구치 토	2개	22개	9.54
베일리 어	모험	100분	전체 관람	라세 할스	25개	33개	9.53
원더	드라마	113분	전체 관람	스티븐 크	9개	12개	9.53
아일라	드라마	123분	15세 관람	잔 올카이	2개	2개	9.52
극장판 바	애니메이션	140분	전체 관람	이시타테	7개	5개	9.51
먼 훗날 우	드라마	120분	상영등급	유약영	24개	19개	9.5
당갈	드라마	161분	12세 관람	니테쉬 티	6개	30개	9.49
포드 V 페	액션	152분	12세 관람	제임스 맨	19개	82개	9.48
그대, 고맙	공연상황	80분	전체 관람	오윤동	22개	1개	9.48
주전장	다큐멘터리	121분	상영등급	미키 데자	1개	연기자 없	9.47
쇼생크 탈	드라마	142분	15세 관람	프랭크 다	19개	66개	9.45
터미네이	SF	137분	15세 관람	제임스 카	43개	65개	9.44
클래식	멜로/로맨	132분	12세 관람	곽재용	28개	31개	9.43
라이언 일	전쟁	170분	15세 관람	스티븐 스	145개	98개	9.43
덕구	드라마	91분	전체 관람	방수인	7개	220개	9.43
나 홀로 집	모험	105분	전체 관람	크리스 콜	38개	19개	9.43
퀵-E	애니메이션	104분	전체 관람	앤드류 스	20개	11개	9.42
보헤미안	드라마	134분	12세 관람	브라이언	25개	23개	9.42
백 투 더	SF	120분	12세 관람	로버트 저	46개	49개	9.41
사운드 오	멜로/로맨	172분	전체 관람	로버트 와	51개	38개	9.41
포레스트	드라마	142분	12세 관람	로버트 저	46개	98개	9.41
잭 스나이	액션	242분	12세 관람	잭 스나이	19개	70개	9.41
위대한 쇼	드라마	104분	12세 관람	마이클 그	2개	46개	9.41
글래디에	액션	154분	15세 관람	리들리 스	76개	49개	9.41
타이타닉	멜로/로맨	194분	15세 관람	제임스 카	43개	58개	9.41
인생은 아	드라마	116분	전체 관람	로베르토	20개	20개	9.4
헬프	드라마	146분	전체 관람	테이트 테	10개	28개	9.4
살인의 추	범죄	132분	15세 관람	봉준호	33개	41개	9.4
매트릭스	SF	136분	12세 관람	킬리 워쇼	14개	83개	9.4

을 바꿔주어야 합니다. 그러면 코딩 이슈가 없는 데이터가 나옵니다. 그리고

100분	15세 관람가	감독	3인	33인	0.1
93분	PG	하워드 프	8개	19개	5.6
88분	PG	로렌스 조	21개	2개	8.2
102분	상영등급	티파니 바	4개	연기자 없	7.9
미국	PG-13	조지 C. 울	11개	21개	7.8
68분	상영등급	조지 큐커	61개	16개	5.7
78분	R	제프 크록	8개	26개	4.4
83분	PG	티르프 알	7개	7개	8.1
91분	NR	버트 케네	26개	60개	6.9
101분	R	스콧 토머	4개	5개	4.2
115분	12세 관람	켄 마리노	30개	19개	6
104분	상영등급	머빈 르로	71개	34개	5.5

보시는 것처럼 미국

영화에 대한 데이터가 많기 때문에 상영 등급인 G, PG, PG-13 등이 존재하는데 데이터 분석 시 model을 fit할 때, feature의 요소 중 하나인 상영 등급의 index label이 너무 많으면 좋은 모델을 만들기 어려우므로 G와 PG는 전체 관람가, PG-13은 12세 관람가, R은 15세 관람가, NC-17은 청소년 관람불가로 매핑합니다. 이로써 상영 등급이 가질 수 있는 값은 [상영 시간 정보 없음, 전체 관람가, 12세 관람가, 15세 관람가, 청소년 관람불가, NR]로 축소됩니다. 또한, 영화 평점을 기준으로 영화를 classification하는 게 목적이므로 8점 이상이라면 명작, 8점 미만 5점 이상이라면 범작, 5점 미만이라면 졸작으로 매핑합니다.

상영등급	단 로하우	1개	명작
15세 관람	통 디칠로	137개	범작
상영등급	케렘 산가	4개	범작
상영등급	팀 고르스	1개	범작
12세 관람	데이비드	4개	범작
청소년 관	짐 필드 스	34개	범작
15세 관람	크레이그	30개	범작
12세 관람	샤리 스프	60개	범작
전체 관람	커크 존스	65개	범작
상영등급	브래드포드	1개	범작
상영등급	안드리야	2개	범작
전체 관람	빌 팩스톤	40개	범작
청소년 관	제프 카뉴	30개	졸작
상영등급	소니아 케	연기자 없	범작
12세 관람	데니스 듀	29개	범작
상영등급	베네딕트	연기자 없	명작
12세 관람	존 터틀타	108개	범작
12세 관람	존 터틀타	108개	범작
청소년 관	구스 트리	1개	범작
상영등급	버나드 L	21개	범작
15세 관람	세바스찬	56개	범작
상영등급	존 데릭	21개	졸작
상영등급	로버트 엘	1개	범작

다. 이를 통해 이런 결과를 얻으며 1차 preprocessing이 완성됩니다.

이후, 2차 preprocessing을 해야 되는데, random forest model을 fit할 때, decision tree 특징 상 features로 들어가는 genre, running_time, screening_rat, 감독의 연출작 개수, 주인공의 출연작 개수는 모두 discrete이며 범주형일 필요가 있습니다. Genre와 screening_rat은 이미 범주형이기 때문에 따로, mapping 할 필요가 없지만 running_time, 감독의 연출

작 개수, 주인공의 출연작 개수는 연속형 데이터이므로 평균을 기준으로 평균 이상, 평균 미만으로 범주화해 줍니다. 그러면

덕 어택	애니메이션	러닝 타임	상영등급	척 존스	연출작 개	연기자 없	명작
덕 패밀리	코미디	상영 시간	상영등급	피터 볼드	연출작 개	주연의 출	범작
던 오브 더 액션	러닝 타임	12세 관람	앤 K. 블랙	연출작 개	주연의 출	출작	
던 오브 더 드라마	러닝 타임	상영등급	커크 루던	연출작 개	주연의 출	범작	
던 오브 아 드라마	상영 시간	상영등급	마이클 W.	연출작 개	주연의 출	범작	
던 오브 언 공포	러닝 타임	상영등급	데이먼 패	연출작 개	연기자 없	범작	
던 월	다큐멘터리	러닝 타임	상영등급	조쉬 로웰	연출작 개	연기자 없	명작
던디 소형	서부	러닝 타임	NR	샘 페킨파	연출작 개	주연의 출	범작
던랜드	다큐멘터리	러닝 타임	상영등급	아담 마조	연출작 개	연기자 없	범작
던스모어	드라마	러닝 타임	15세 관람	피터 스피	연출작 개	주연의 출	범작
던위치 호	공포	러닝 타임	상영등급	다니엘 할	연출작 개	주연의 출	범작
던전 드래	판타지	러닝 타임	NR	게리 리블	연출작 개	주연의 출	출작
덤 앤 더머	코미디	상영 시간	12세 관람	트로이 말	연출작 개	주연의 출	출작
덤 앤 더머	장르 정보	상영 시간	12세 관람	크리스토프	연출작 개	주연의 출	범작
덤 앤 더머	코미디	러닝 타임	15세 관람	바비 패럴	연출작 개	주연의 출	범작

이렇게 분석의 대상이 될 수

있는 data set이 완성됩니다.

마지막으로, 위에서 언급한 모든 csv 파일을 Hadoop directory에서 접근했을 시,

Excel 97 - 2003 통합 문서 (*.xls)

CSV UTF-8(심표로 분리)(*.csv)

XML 데이터 (*.xml)

encoding이 깨져 보이므로 csv를 엑셀로 열어

꼭 csv

utf-8 형태로 저장하여 upload해야 csv 파일이 깨짐 없이 잘 보입니다.

또한, 보통 네이버 영화 사이트에 영화 감독을 클릭하면 바로 밑에 필모그래피의 개수가 나오는데 '샘 와이즈만' 이 감독 혼자만 page 상 다른 곳에 필모그래피가 있습니다. 전처리 과정에 오류가 나지 않도록 엑셀 파일에 들어가서 해당 감독의 연출작 개수를 '개'가 아니라 '17개'로 바꿔 주어야 합니다. 전처리에 관한 소스코드 및 소스코드 주석은 git 혹은 과제 첨부파일로 첨부하겠습니다.

이제 이 데이터를 불러와서 Hadoop 상에서 약간의 추가 처리 이후 분석을 진행하면 됩니다. 우선 putty를 통해 Hadoop 상에 upload한 csv를 DataFrame으로 load합니다. 그리고 지금 이 data set의 features가 될 genre, running_time, screening_rat, 감독의 연출작 개수, 주인공의 출연작 개수는 모두 String type이기 때문에 classification의 feature가 될 수 없습니다. 따라서, 위의 columns의 값들을 모두 one-hot encoding해서 실수와 mapping합니다.

title	director	genre_one_hot_encoding	running_time_one_hot_encoding	screening_rat_one_hot_encoding	dir_movies_one_hot_encoding	main_role_movies_one_hot_encoding	rating
반전	박해령	2.0	1.0	4.0	0.0	0.0	명작
그린 북	피터 패럴리	0.0	0.0	2.0	1.0	0.0	명작
가바나움	나딘 라바키	0.0	0.0	1.0	0.0	0.0	명작
디지털 어드벤처 라스트 메탈루션...	타구치 토모히사	5.0	0.0	2.0	0.0	0.0	명작
베일리 여제인	라세 할스트롬	8.0	0.0	4.0	1.0	0.0	명작
원더	스티븐 크로보스키	0.0	0.0	4.0	0.0	0.0	명작
아일랜드	잔 톨카이	0.0	0.0	1.0	0.0	0.0	명작
극장판 바이올렛 에반게론	이시타테 리미치	5.0	0.0	4.0	0.0	0.0	명작
먼 훗날 우리	유안젤	0.0	0.0	1.0	0.0	0.0	명작
포드 V 페라리	니레쉬 리와인	0.0	0.0	2.0	0.0	0.0	명작
그대, 고만소 : 김홍중 생애 ...	제임스 매크드	4.0	0.0	2.0	1.0	0.0	명작
주전장	미키 데자키	2.0	0.0	4.0	1.0	0.0	명작
소셜크 랄홀 프랭크 디라몬트		0.0	0.0	1.0	1.0	0.0	명작
터미네이터 2:오리지널	제임스 카메론	12.0	0.0	1.0	1.0	0.0	명작
클래식	프랭클	11.0	0.0	2.0	1.0	0.0	명작
라이언 일병 쿠하기	스티븐 스필버그	18.0	0.0	1.0	0.0	0.0	명작
단구	방수인	0.0	0.0	4.0	0.0	0.0	명작
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	명작

21/12/05 09:13:08 INFO BlockManagerInfo: Removed broadcast_10_piece0 on sandbox-hdp.hortonworks.com:33343 in memory (size: 33.3 KB, free: 366.3 MB)

title	director	genre_one_hot_encoding	running_time_one_hot_encoding	screening_rat_one_hot_encoding	dir_movies_one_hot_encoding	main_role_movies_one_hot_encoding	rating
반전	박해령	2.0	1.0	4.0	0.0	0.0	명작
그린 북	피터 패럴리	0.0	0.0	2.0	1.0	0.0	명작
가바나움	나딘 라바키	0.0	0.0	1.0	0.0	0.0	명작
디지털 어드벤처 라스트 메탈루션...	타구치 토모히사	5.0	0.0	2.0	0.0	0.0	명작
베일리 여제인	라세 할스트롬	8.0	0.0	4.0	1.0	0.0	명작
원더	스티븐 크로보스키	0.0	0.0	4.0	0.0	0.0	명작
아일랜드	잔 톨카이	0.0	0.0	1.0	0.0	0.0	명작
극장판 바이올렛 에반게론	이시타테 리미치	5.0	0.0	4.0	0.0	0.0	명작
먼 훗날 우리	유안젤	0.0	0.0	1.0	0.0	0.0	명작
포드 V 페라리	니레쉬 리와인	0.0	0.0	2.0	0.0	0.0	명작
그대, 고만소 : 김홍중 생애 ...	제임스 매크드	4.0	0.0	2.0	1.0	0.0	명작
주전장	미키 데자키	2.0	0.0	4.0	1.0	0.0	명작
소셜크 랄홀 프랭크 디라몬트		0.0	0.0	1.0	1.0	0.0	명작
터미네이터 2:오리지널	제임스 카메론	12.0	0.0	1.0	1.0	0.0	명작
클래식	프랭클	11.0	0.0	2.0	1.0	0.0	명작
라이언 일병 쿠하기	스티븐 스필버그	18.0	0.0	1.0	0.0	0.0	명작
단구	방수인	0.0	0.0	4.0	0.0	0.0	명작
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	명작
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	명작

(위 그림은 local, 아래 그림은 hadoop)

그러면 위와 같은 dataframe이 생성됩니다. 그 뒤, vectorassembler module을 이용하여 model 학습의 features로 쓰일 columns를 지정해 줍니다. 그 결과는 다음과 같습니다.

title	director	genre_one_hot_encoding	running_time_one_hot_encoding	screening_rat_one_hot_encoding	dir_movies_one_hot_encoding	main_role_movies_one_hot_encoding	rating	features
반전	박해령	2.0	1.0	4.0	0.0	0.0	0.0	명작 [[2.0,1.0,4.0,0.0,...]]
그린 북	피터 패럴리	0.0	0.0	2.0	1.0	0.0	1.0	명작 [[0.0,0.0,2.0,1.0,...]]
가바나움	나딘 라바키	0.0	0.0	1.0	0.0	0.0	0.0	명작 [[5.2],[1.0]]
디지털 어드벤처 라스트 메탈루션...	타구치 토모히사	5.0	0.0	2.0	0.0	0.0	0.0	명작 [[5.0,2],[5.0,2.0]]
베일리 여제인	라세 할스트롬	8.0	0.0	4.0	1.0	0.0	0.0	명작 [[8.0,0.0,4.0,1.0,...]]
원더	스티븐 크로보스키	0.0	0.0	4.0	0.0	0.0	0.0	명작 [[5.2],[4.0]]
아일랜드	잔 톨카이	0.0	0.0	1.0	0.0	0.0	0.0	명작 [[5.2],[1.0]]
극장판 바이올렛 에반게론	이시타테 리미치	5.0	0.0	4.0	0.0	0.0	0.0	명작 [[5.0,2],[5.0,4.0]]
먼 훗날 우리	유안젤	0.0	0.0	1.0	0.0	0.0	0.0	명작 [[5.3],[1.0]]
포드 V 페라리	니레쉬 리와인	0.0	0.0	2.0	0.0	0.0	0.0	명작 [[5.2],[2.0]]
그대, 고만소 : 김홍중 생애 ...	제임스 매크드	4.0	0.0	2.0	1.0	0.0	0.0	명작 [[4.0,0.0,2.0,1.0,...]]
주전장	미키 데자키	2.0	0.0	4.0	1.0	0.0	0.0	명작 [[20.0,1.0,4.0,1.0,...]]
소셜크 랄홀 프랭크 디라몬트		0.0	0.0	1.0	1.0	0.0	0.0	명작 [[5.0,4],[2.0,2.0]]
터미네이터 2:오리지널	제임스 카메론	12.0	0.0	1.0	1.0	0.0	0.0	명작 [[0.0,0.0,1.0,1.0,...]]
클래식	프랭클	11.0	0.0	2.0	1.0	0.0	0.0	명작 [[12.0,0.0,1.0,1.0,...]]
라이언 일병 쿠하기	스티븐 스필버그	18.0	0.0	1.0	0.0	0.0	0.0	명작 [[11.0,0.0,2.0,1.0,...]]
단구	방수인	0.0	0.0	4.0	0.0	0.0	0.0	명작 [[18.0,0.0,1.0,1.0,...]]
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	0.0	명작 [[5.2,4],[4.0,1.0]]
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	0.0	명작 [[8.0,0.0,4.0,1.0,...]]
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	0.0	명작 [[5.0,0.0,4.0,1.0,...]]

title	director	genre_one_hot_encoding	running_time_one_hot_encoding	screening_rat_one_hot_encoding	dir_movies_one_hot_encoding	main_role_movies_one_hot_encoding	rating	features
반전	박해령	2.0	1.0	4.0	0.0	0.0	0.0	명작 [[2.0,1.0,4.0,0.0,...]]
그린 북	피터 패럴리	0.0	0.0	2.0	1.0	0.0	1.0	명작 [[0.0,0.0,2.0,1.0,...]]
가바나움	나딘 라바키	0.0	0.0	1.0	0.0	0.0	0.0	명작 [[5.2],[1.0]]
디지털 어드벤처 라스트 메탈루션...	타구치 토모히사	5.0	0.0	2.0	0.0	0.0	0.0	명작 [[5.0,2],[5.0,2.0]]
베일리 여제인	라세 할스트롬	8.0	0.0	4.0	1.0	0.0	0.0	명작 [[8.0,0.0,4.0,1.0,...]]
원더	스티븐 크로보스키	0.0	0.0	4.0	0.0	0.0	0.0	명작 [[5.2],[4.0]]
아일랜드	잔 톨카이	0.0	0.0	1.0	0.0	0.0	0.0	명작 [[5.2],[1.0]]
극장판 바이올렛 에반게론	이시타테 리미치	5.0	0.0	4.0	0.0	0.0	0.0	명작 [[5.0,2],[5.0,4.0]]
먼 훗날 우리	유안젤	0.0	0.0	1.0	0.0	0.0	0.0	명작 [[5.3],[1.0]]
포드 V 페라리	니레쉬 리와인	0.0	0.0	2.0	0.0	0.0	0.0	명작 [[5.2],[2.0]]
그대, 고만소 : 김홍중 생애 ...	제임스 매크드	4.0	0.0	2.0	1.0	0.0	0.0	명작 [[4.0,0.0,2.0,1.0,...]]
주전장	미키 데자키	2.0	0.0	4.0	1.0	0.0	0.0	명작 [[20.0,1.0,4.0,1.0,...]]
소셜크 랄홀 프랭크 디라몬트		0.0	0.0	1.0	1.0	0.0	0.0	명작 [[5.0,4],[2.0,2.0]]
터미네이터 2:오리지널	제임스 카메론	12.0	0.0	1.0	1.0	0.0	0.0	명작 [[0.0,0.0,1.0,1.0,...]]
클래식	프랭클	11.0	0.0	2.0	1.0	0.0	0.0	명작 [[12.0,0.0,1.0,1.0,...]]
라이언 일병 쿠하기	스티븐 스필버그	18.0	0.0	1.0	0.0	0.0	0.0	명작 [[11.0,0.0,2.0,1.0,...]]
단구	방수인	0.0	0.0	4.0	0.0	0.0	0.0	명작 [[18.0,0.0,1.0,1.0,...]]
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	0.0	명작 [[5.2,4],[4.0,1.0]]
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	0.0	명작 [[8.0,0.0,4.0,1.0,...]]
나 홀로 집에	크리스 콜럼버스	8.0	0.0	4.0	1.0	0.0	0.0	명작 [[5.0,0.0,4.0,1.0,...]]

(위 그림은 local, 아래 그림은 hadoop)

마지막으로, 분류의 기준이 되는 rating에 labelIndex를 붙임으로써, 데이터 분석을 위한 모든 준비가 끝나게 됩니다.

	title	director	genre_one_hot_encoding	running_time_one_hot_encoding	screening_rat_one_hot_encoding	dir_movies_one_hot_encoding	main_role_movies_one_hot_encoding	rating	features	labelIndex
	발칙!	박혁경	2.0	1.0	4.0	0.0	0.0	평점	[2.0,1.0,4.0,0.0,...]	1.0
	그림자	피터 패들리	0.0	0.0	2.0	1.0	1.0	평점	[0.0,0.0,2.0,1.0,...]	1.0
	가버나뎀	니콜라 코폴라	0.0	0.0	1.0	0.0	0.0	평점	(5,[2],[1.0])	1.0
	라스트 벵골루션...	타구치 토모히사	0.0	0.0	2.0	0.0	0.0	평점	(5,[0.2],[5.0,2.0])	1.0
	베일리 어게인	라세 할스트롬	0.0	0.0	4.0	1.0	0.0	평점	[8.0,0.0,4.0,1.0,...]	1.0
	원더 스타트	크보스키	0.0	0.0	4.0	0.0	0.0	평점	(5,[2],[4.0])	1.0
	아일라	잔 줄라이	0.0	0.0	1.0	0.0	0.0	평점	(5,[2],[1.0])	1.0
	바이올렛 에버가든	이시타다 다이치	0.0	0.0	4.0	0.0	0.0	평점	(5,[0.2],[5.0,4.0])	1.0
	만 쏘발	주리	0.0	0.0	0.0	1.0	0.0	평점	(5,[3],[1.0])	1.0
	노란	니테워 탄완리	0.0	0.0	2.0	0.0	0.0	평점	(5,[2],[2.0])	1.0
	포드 V 페리	제임스 맨콧	4.0	0.0	2.0	1.0	1.0	평점	[4.0,0.0,2.0,1.0,...]	1.0
	김호중 생애...	오종동	20.0	1.0	4.0	1.0	0.0	평점	[20.0,1.0,4.0,1.0,...]	1.0
	주진실	미키 데자키	2.0	0.0	0.0	0.0	2.0	평점	(5,[0.4],[2.0,2.0])	1.0
	소생의 물결	프랑크 다라멘트	0.0	0.0	1.0	1.0	1.0	평점	[0.0,0.0,1.0,1.0,...]	1.0
	미네이타 2:오리자	제임스 카메론	12.0	0.0	1.0	1.0	1.0	평점	[12.0,0.0,1.0,1.0,...]	1.0
	올랜트	관개환	11.0	0.0	2.0	1.0	0.0	평점	[11.0,0.0,2.0,1.0,...]	1.0
	라이언 월브	구하기 스티븐 스탈버그	10.0	0.0	1.0	1.0	1.0	평점	[10.0,0.0,1.0,1.0,...]	1.0
	덕구	방수인	0.0	0.0	4.0	0.0	0.0	평점	(5,[2.4],[4.0,1.0])	1.0
	나 홀로 집에	크리스 콜럼버스	0.0	0.0	4.0	1.0	0.0	평점	[8.0,0.0,4.0,1.0,...]	1.0
	원더	앤드루 스타튼	0.0	0.0	4.0	1.0	0.0	평점	[5.0,0.0,4.0,1.0,...]	1.0

	title	director	genre_one_hot_encoding	running_time_one_hot_encoding	screening_rat_one_hot_encoding	dir_movies_one_hot_encoding	main_role_movies_one_hot_encoding	rating	features	labelIndex
	발칙!	박혁경	2.0	1.0	4.0	0.0	0.0	평점	[2.0,1.0,4.0,0.0,...]	1.0
	그림자	피터 패들리	0.0	0.0	2.0	1.0	1.0	평점	[0.0,0.0,2.0,1.0,...]	1.0
	가버나뎀	니콜라 코폴라	0.0	0.0	1.0	0.0	0.0	평점	(5,[2],[1.0])	1.0
	라스트 벵골루션...	타구치 토모히사	0.0	0.0	2.0	0.0	0.0	평점	(5,[0.2],[5.0,2.0])	1.0
	베일리 어게인	라세 할스트롬	0.0	0.0	4.0	1.0	0.0	평점	[8.0,0.0,4.0,1.0,...]	1.0
	원더 스타트	크보스키	0.0	0.0	4.0	0.0	0.0	평점	(5,[2],[4.0])	1.0
	아일라	잔 줄라이	0.0	0.0	1.0	0.0	0.0	평점	(5,[2],[1.0])	1.0
	바이올렛 에버가든	이시타다 다이치	0.0	0.0	4.0	0.0	0.0	평점	(5,[0.2],[5.0,4.0])	1.0
	만 쏘발	주리	0.0	0.0	0.0	1.0	0.0	평점	(5,[3],[1.0])	1.0
	노란	니테워 탄완리	0.0	0.0	2.0	0.0	0.0	평점	(5,[2],[2.0])	1.0
	포드 V 페리	제임스 맨콧	4.0	0.0	2.0	1.0	1.0	평점	[4.0,0.0,2.0,1.0,...]	1.0
	김호중 생애...	오종동	20.0	1.0	4.0	1.0	0.0	평점	[20.0,1.0,4.0,1.0,...]	1.0
	주진실	미키 데자키	2.0	0.0	0.0	0.0	2.0	평점	(5,[0.4],[2.0,2.0])	1.0
	소생의 물결	프랑크 다라멘트	0.0	0.0	1.0	1.0	1.0	평점	[0.0,0.0,1.0,1.0,...]	1.0
	미네이타 2:오리자	제임스 카메론	12.0	0.0	1.0	1.0	1.0	평점	[12.0,0.0,1.0,1.0,...]	1.0
	올랜트	관개환	11.0	0.0	2.0	1.0	0.0	평점	[11.0,0.0,2.0,1.0,...]	1.0
	라이언 월브	구하기 스티븐 스탈버그	10.0	0.0	1.0	1.0	1.0	평점	[10.0,0.0,1.0,1.0,...]	1.0
	덕구	방수인	0.0	0.0	4.0	0.0	0.0	평점	(5,[2.4],[4.0,1.0])	1.0
	나 홀로 집에	크리스 콜럼버스	0.0	0.0	4.0	1.0	0.0	평점	[8.0,0.0,4.0,1.0,...]	1.0
	원더	앤드루 스타튼	0.0	0.0	4.0	1.0	0.0	평점	[5.0,0.0,4.0,1.0,...]	1.0

(위 그림은 local, 아래 그림은 hadoop)

이제 데이터를 (0.8 : 0.2), (0.7 : 0.3), (0.75 : 0.25) 등으로 random Split하고 decision tree의 개수는 1000으로 고정시키고 모델을 학습시킨 뒤, test set을 input으로 하여 학습시킨 모델의 정답율과 오답율을 알아 봅니다. (데이터 분석에 관한 소스코드와 코드 설명은 git 혹은 과제 첨부파일에 첨부하겠습니다.)

데이터 분석 결과

Rating을 명작, 범작, 졸작으로 mapping하고, running_time, 감독의 연출작 개수, 주인공의 출연작 개수를 평균 이상과 미만으로 mapping하였더니

```
21/12/05 09:13:43 INFO TaskSchedulerImpl: Adding task set 36.0
21/12/05 09:13:43 INFO TaskSetManager: Starting task 0.0 in st
21/12/05 09:13:43 INFO Executor: Running task 0.0 in stage 36.
21/12/05 09:13:43 INFO ShuffleBlockFetcherIterator: Getting 1
21/12/05 09:13:43 INFO ShuffleBlockFetcherIterator: Started 0
21/12/05 09:13:43 INFO Executor: Finished task 0.0 in stage 36
21/12/05 09:13:43 INFO TaskSetManager: Finished task 0.0 in st
21/12/05 09:13:43 INFO TaskSchedulerImpl: Removed TaskSet 36.0
21/12/05 09:13:43 INFO DAGScheduler: ResultStage 36 (collectAs
21/12/05 09:13:43 INFO DAGScheduler: Job 23 finished: collectA
Accuracy = 0.621
Test Error = 0.379
21/12/05 09:13:43 INFO SparkContext: Invoking stop() from shut
21/12/05 09:13:43 INFO AbstractConnector: Stopped Spark@17bb8e
21/12/05 09:13:43 INFO SparkUI: Stopped Spark web UI at http:/
21/12/05 09:13:43 INFO MapOutputTrackerMasterEndpoint: MapOutp
21/12/05 09:13:43 INFO MemoryStore: MemoryStore cleared
```

이와 같은 정답율과 오답율이 나왔습니다. 그래서 저는 어떻게 하면 정답율을 올릴 수 있을까 생각하던 도중에 features를 구성하는 column들을 labelling 한 기준에 따라, 정답율을 올릴 수 있다는 생각이 들었습니다.

우선, rating(labelIndex)의 범주가 3개인 것에 제일 먼저 주목했습니다. 아무래도 2개로 classification 하는 것이 3개보다는 더 decision tree를 형성하기 명확하다고 생각했기 때문입니다. 따라서, 영화 데이터의 평균 평점을 구하여 평균 평점 이상의 데이터와 평균 평점 미만의 데이터 2가지로 분류하여 분석을 해봤습니다. 결과는

```
21/12/05 10:48:10 INFO DAGScheduler: Resu
21/12/05 10:48:10 INFO DAGScheduler: Job 2
Accuracy = 0.66
Test Error = 0.34
21/12/05 10:48:10 INFO SparkContext: Invo
21/12/05 10:48:10 INFO AbstractConnector:
21/12/05 10:48:10 INFO SparkUI: Stopped S
21/12/05 10:48:10 INFO MapOutputTrackerMa
```

이처럼 정답율이 약간 상승한 것을 볼 수 있습니다.

저는 여기서 영화 rating을 3가지 범주(명작, 범작, 졸작)로 나누는 경우, 평균을 기준으로 2가지 범주로 나누는 경우, 중앙값을 기준으로 2가지 범주로 나누는 경우와 running_time, 감독의 연출작 개수, 주인공의 출연작 개수를 평균을 기준으로 나누는 것과 중앙값을 기준으로 나누는 것 총 $3 * 2 = 6$ 개의 경우의 수로 나누어 정답율을 비교해 보면 최고의 정답율을 가지는 조합을 찾을 수 있을 거라는 생각이 들었습니다.

그래서, 위에서 이미 다룬 2가지 조합을 제외한 4가지 조합의 정답율을 보면

Rating을 3가지 범주로 나누고 running_time, 감독의 연출작 개수, 주인공의 출연작 개수

```
in 5 ms on localhost (executor driver) (1/1)
21/12/05 10:59:30 INFO TaskSchedulerImpl: Rem
all completed, from pool
21/12/05 10:59:30 INFO DAGScheduler: ResultSt
etrics.scala:53) finished in 0.008 s
21/12/05 10:59:30 INFO DAGScheduler: Job 23 f
Metrics.scala:53, took 2.325927 s
Accuracy = 0.637
Test Error = 0.363
21/12/05 10:59:30 INFO BlockManagerInfo: Remo
hdp.hortonworks.com:36411 in memory (size: 19
21/12/05 10:59:30 INFO SparkContext: Invoking
21/12/05 10:59:30 INFO AbstractConnector: Sto
1.1]}{0.0.0.0:4040}
21/12/05 10:59:30 INFO SparkUI: Stopped Spark
nworks.com:4040
```

를 중앙값을 기준으로 나누는 것:

Rating을 평균을 기준으로 2가지 범주로 나누고 running_time, 감독의 연출작 개수, 주인공의 출연작 개수를 중앙값을 기준으로 나누는 것:

```
21/12/05 11:03:13 INFO Executor: Finishe
21/12/05 11:03:13 INFO TaskSetManager: F
21/12/05 11:03:13 INFO TaskSchedulerImpl
21/12/05 11:03:13 INFO DAGScheduler: Res
21/12/05 11:03:13 INFO DAGScheduler: Job
Accuracy = 0.693
Test Error = 0.307
21/12/05 11:03:13 INFO SparkContext: Inv
21/12/05 11:03:13 INFO AbstractConnector
21/12/05 11:03:13 INFO SparkUI: Stopped
21/12/05 11:03:13 INFO MapOutputTrackerM
21/12/05 11:03:13 INFO MemoryStore: Memo
```

Rating을 중앙값을 기준으로 2가지 범주로 나누고 running_time, 감독의 연출작 개수, 주

```
21/12/05 11:07:00 INFO Exec
21/12/05 11:07:00 INFO Task
21/12/05 11:07:00 INFO Task
21/12/05 11:07:00 INFO DAGS
21/12/05 11:07:00 INFO DAGS
Accuracy = 0.66
Test Error = 0.34
21/12/05 11:07:00 INFO Spar
21/12/05 11:07:00 INFO Abst
21/12/05 11:07:00 INFO Spar
21/12/05 11:07:00 INFO MapO
21/12/05 11:07:00 INFO Memo
```

인공의 출연작 개수를 평균을 기준으로 나누는 것:

Rating을 중앙값을 기준으로 2가지 범주로 나누고 running_time, 감독의 연출작 개수, 주인공의 출연작 개수를 중앙값을 기준으로 나누는 것:

```
21/12/05 11:08:31 INFO ShuffleB
21/12/05 11:08:31 INFO ShuffleB
21/12/05 11:08:31 INFO Executor
21/12/05 11:08:31 INFO TaskSetM
21/12/05 11:08:31 INFO TaskSched
21/12/05 11:08:31 INFO DAGSchedt
21/12/05 11:08:31 INFO DAGSchedt
Accuracy = 0.686
Test Error = 0.314
21/12/05 11:08:31 INFO SparkCont
21/12/05 11:08:31 INFO Abstract
21/12/05 11:08:31 INFO SparkUI:
21/12/05 11:08:34 INFO MapOutput
21/12/05 11:08:34 INFO MemorySt
```

예상대로 rating의 범주를 3개에서 2개로 줄이니 정답율이 올라간 것을 확인할 수 있었습니다. 특히, Rating을 평균을 기준으로 2가지 범주로 나누고 running_time, 감독의 연출작 개수, 주인공의 출연작 개수를 중앙값을 기준으로 나누는 것과 Rating을 중앙값을 기준으로 2가지 범주로 나누고 running_time, 감독의 연출작 개수, 주인공의 출연작 개수를 중앙값을 기준으로 나누는 것, 이 2개의 조합은 정답율이 거의 70%에 육박했습니다. 하지만, 조합마다 한 개의 표본만으로 어떤 조합이 최고의 정답률을 갖는다고 하기 어려우므로, 조합마다 10번씩 실행하여 평균을 내어 비교해 보았습니다.

결과적으로, Rating을 중앙값을 기준으로 2가지 범주로 나누고 running_time, 감독의 연출작 개수, 주인공의 출연작 개수를 중앙값을 기준으로 나누는 것이 Rating을 평균을 기준으로 2가지 범주로 나누고 running_time, 감독의 연출작 개수, 주인공의 출연작 개수를 중앙값을 기준으로 나누는 것보다 근소하게 좋았으며 연속적인 값을 2개의 범주로 나누어야 한다면 평균보다는 중앙값을 기준으로 하는 것이 좋으며 약 70%의 정답율로 [genre, running_time, screening_rat, 감독의 연출작 개수, 주인공의 출연작 개수] 이 5개의 features가 영화의 작품성에 관여한다고 결론 내릴 수 있습니다.

추가적인 확장 가능성

위에서 언급한 것처럼 logistic regression with multiclass, naïve bayes, decision tree, svm, k-nearest Neighbor, dnn, ensemble Learning 등의 알고리즘을 적용하거나 연속형 자료를 범주형 자료로 분류할 때 조금 더 과학적인 방법을 쓰거나 features에 의미 있는 column들이 추가된다면 더욱 더 정확도 높은 모델이 나올 수도 있을 것이다. 하지만, 예를 들어 features에 rating을 넣어 영화의 흥행 여부 혹은 box office 수익을 예측하는 것은 바람직하지 않다. Rating과 (흥행 여부, 수익)과는 너무 강력한 상관 관계가 존재하기 때문에 rating이 높다면 당연히 흥행할 가능성이 높고 rating이 낮다면 영화가 망할 확률이 크기에 이와 같은 데이터 분석은 큰 의미가 없다고 생각한다. 따라서, features에는 너무 상관 관계가 약한 column이 와도 안 되고 너무 강한 column이 와도 안 되며 적절한 features(ex - 연출자의 수상 경력, 주연의 수상 경력, 배급사, staff의 총 수 등등)를 추가해서 model에 먹인다면 더 좋은 정답율을 보여줄 수도 있을 것이다.

또한, 이 프로젝트의 뼈대만을 활용하여 소재는 무궁무진하게 변화 가능한데 예를 들어, 중고차의 여러 features(차량 생산 연도, 차량 주행 거리, 사고 횟수, 사고 경중 etc..)를 model에 먹여 classification한다면 해당 중고차가 상급인지 중급인지 하급인지에 대한 자동 분류가 가능하다. 물론 이 모델이 어플에 담겨 시장에 나오려면 features에 대해 허위 기재할 수 없게 하는 제도적, 기술적 뒷받침이 필요할 것이다.