데이터마이닝 프로젝트

게임데이터분석(스타크래프트2)

동국대학교 통계학과

2018110479 김 은 지
2017110527 김 태 현
2017110501 성 주 혁
2017110518 윤 상 우

2 0 2 2

목 차

제1장 서 론

제1절 주제 선정 및 프로젝트의 목적

본 조는 'StarCraft2ReplayAnalysis' 데이터 셋을 통해 게임 내 랭크에 영향을 요인들에 대하여 탐구하고 통계적 분석을 실시하였다. 'StarCraft2ReplayAnalysis' 데이터는 21개의 변수와 3,395개의 관측값을 가진다. 데이터 셋은 Thompson JJ, Blair MR, Chen L, Henrey AJ (2013)의 'Video Game Telemetry as a Critical Tool in the Study of Complex Skill Learning.'라는 연구 주제로 선정되었던 논문에서 참조되었다. 논문과 본 조의 데이터의 차이는 LeagueIndex 8(Professional) 55개 행의 추가이다.

분석의 방향은 다음과 같이 이루어진다. 게이머의 객관적 지표 개발 -> 랭크 예측 -> 고객의 유형과 특성 파악, 유지·보수 및 업데이트, 새로운 유저 창출을 위한 마케팅 활동 스타크래프트와 같은 RTS 장르는 '더 다양한 전략'과 '더 강한 유닛'을 뽑고 이를 컨트롤하는 '실력'이 게임을 플레이하는 주된 방법이다. 따라서 여러 작업을 동시에 수행할 수 있는 능력이 있어야 한다. 스타크래프트는 맵이 제한되어 있고, 자원이 한정적이다. 이러한 이유와 데이터의 논문에서 게임에 큰 영향을 미치는 요소는 인지(지각, 행동, 주기)라는 것을 통해, 본 조의 주된 분석 방향은 각 플레이어의 인지능력에 중점을 두고 있다. 신체적으로 힘이나, 속도가 있어야 하는 운동선수만큼은 아니지만, 게임에서도 중하위권은 전략 연습 등을 통해 도달할 수 있으나 최상위권을 차지하려면 화면에 대한 빠른 인지 능력과 수행 능력이 동반되어야 한다고 가정하였다. 이를, 분석에 적용 가능하게 하는 것은 데이터 셋에서 PAC라는 특이 변수가 존재하기 때문이다. PAC는 'Perceptio Action Cycle'을 의미하며 이는, 사람의 인지 능력을 지각과 행동의 시간을 주기로 나타낸 것이다. PAC는 Perception Action Cycle이라는 뜻으로 인지를 시작하고 행동하기까지의 동작이다. 사용자의 인지 부하 능력을 나타낼 수 있는 변수로 생각하였으며, 이를 데이터셋 내의 다른 행동 척도를 나타내는 변수들과 함께 분석에 주로 사용할 예정이다.

제2장 본론

제1절 데이터 소개 및 변수 소개

SkillCraft2 데이터는 서로 다른 분위에서 플레이된 스타크래프트2 게임 데이터셋이다. 게임 내에는 1:1부터 다대다 등의 여러 가지 모드가 있지만, 현 데이터에는 1:1 모드에 대한 데이터만을 기록하였고, 게임을 진행하는 맵들 또한 동일한 사이즈로 진행되었다. 온라인 설문조사를 통해 초보자부터 전문가에 이르는 8개로 나누어진 게임 내의 리그에서 3,395명의 각기 다른 분위에 속한 플레이어들에 대한 데이터를 모집했다.

변수 중에 일부 데이터는 우리가 일반적으로 사용하는 시간과는 다른 타임스탬프라는 단위를 사용하였는데, 1초가 88.5 타임스탬프로 정의된다. 또한 PAC란 Hotkey는 단축키를 의미한다. 전체적인 데이터들이 단위 시간 또는 기준(타임스탬프) 당 횟수로 기록되어 있어서 연속형 변수들이 대부분이고, 데이터의 abstract는 다음과 같다.

Data Set Characteristics:	Multivariate	Number of Instances:	3395	Area:	Game
Attribute Characteristics:	Integer, Real	Number of Attributes:	20	Date Donated	2013-10-22
Associated Tasks:	Regression	Missing Values?	Yes	Number of Web Hits:	9320

[표 0] 데이터셋 Abstract

번호	변수명	변수 설명	데이터타입
1	GameID	각 게임마다 고유한 ID 숫자	integer
2	LeagueIndex	Bronze, Silver, Gold, Platinum, Diamond, Master, GrandMaster, Professional leagues (1-8)	Ordinal
3	Age	플레이어의 나이	integer
4	HoursPerWeek	일주일동안 플레이한 시간	integer
5	TotalHours	게임에 접속한 총 시간	integer
6	APM	1분당 컴퓨터에 입력된 명령어의 횟수	continuous
7	SelectByHotkeys	타임스탬프당 사용된 고정 단축키의 횟수	continuous
8	AssignToHotkeys	타임스탬프당 사용된 지정 단축키의 횟수	continuous
9	UniqueHotkeys	타임스탬프당 사용된 특수고정 단축키의 사용횟수	continuous
10	MinimapAttacks	타임스탬프당 미니맵을 통한 공격명령의 횟수	continuous
11	MinimapRightClicks	타임스탬프당 미니맵을 통한 이동명령의 횟수	continuous
12	NumberOfPACs	타임스탬프당 시행된 PAC의 총 횟수	continuous
13	GapBetweenPACs	PAC간 평균 지속 시간(millisecond)	continuous
14	ActionLatency	PAC 시작부터 첫 번째 액션까지 평균 대기 시간(millisecond)	continuous
15	ActionsInPAC	하나의 PAC내의 액션 수	continuous
16	TotalMapExplored	타임스탬프당 플레이어의 시야에 들어온 24x24 사이즈맵의 좌표 그리드 수	continuous
17	WorkersMade	타임스탬프당 자원을 채취할 수 있는 유닛들의 생산 개수	continuous
18	UniqueUnitsMade	타임스탬프당 생성한 특정 유닛들의 수	continuous
19	ComplexUnitsMade	타임스탬프당 훈련된 ghost, infestor, high templar수	continuous
20	ComplexAbilitiesUsed	타임스탬프당 특정 유닛의 능력사용수	continuous
21	MaxTimeStamp	게임이 종료될 때까지 총 타임스탬프	integer

[표 1] 변수 설명

제2절 데이터 탐색 및 전처리

1. 데이터 탐색

번호	변수명	Mean	Std	Min	1Q	Median	3Q	Max
2	LeagueIndex	4.184	1.517	1	3	4	5	8
3	Age	21.647	4.206	16	19	21	24	44
4	HoursPerWeek	15.91	11.963	0	8	12	29	168
5	TotalHours	960.421	17318.13	3	300	500	800	1,000,000
6	APM	117.046	51.945	22.059	79.9	108.01	142.79	389.831
7	SelectByHotkeys	0.004299	0.005284	0	0.001258	0.0025	0.005133	0.0430
8	AssignToHotkeys	0.000374	0.000225	0	0.000204	0.000353	0.000499	0.00175
9	UniqueHotkeys	0.000059	0.000038	0	0.000033	0.000053	0.000079	0.000338
10	MinimapAttacks	0.000098	0.000166	0	0	0.00004	0.000119	0.003019
11	MinimapRightClick	0.000387	0.00037	0	0.00014	0.000281	0.000514	0.004041
12	NumberOfPACs	0.003463	0.000992	0.00067	0.002754	0.003395	0.004027	0.007971
13	GapBetweenPACs	40.3615	17.15357	6.6667	28.95775	36.7235	48.2905	237.1429
14	ActionLatency	63.7394	19.2388	24.0936	50.4466	60.9318	73.6813	176.3721
15	ActionsInPAC	5.272988	1.4948	2.0389	50.4466	5.0955	6.0336	18.5581
16	TotalMapExplored	0.000283	0.000087	0.00009	0.000224	0.00027	0.000325	0.000832
17	WorkersMade	0.001032	0.000519	0.00007	0.000683	0.000905	0.001259	0.005149
18	UniqueUnitsMade	0.000085	0.000025	0.00002	0.000068	0.000082	0.000099	0.000202
19	ComplexUnitsMade	0.000059	0.000111	0	0	0	0.000086	0.000902
20	ComplexAbilities Used	0.000142	0.000265	0	0	0.000181	0.003084	0.003084
21	MaxTimeStamp	83598.22	3497.668	25224	60090	81012	102074	388032

[표 2] 변수별 기초통계량

[그림 1] 변수별 히스토그램

- 반응속도와 명령어 입력 관련 변수들 (APM, NumberOfPACs, GapBetweenPACs, ActionsInPAC)

[그림 1]의 히스토그램을 보면 정규분포에 가깝게 분포되어 있으며, LeagueIndex와 높거나 낮은 상관관계를 가지고 있는 것으로 보인다. RTS라는 게임의 장르 특성은 전략과 전술이 중요하다. 이러한 이유로 반응속도와 멀티태스킹이 중요하며 여러 가지 일을 처리해야 한다. 이 변수들은 그러한 특성을 나타내는 변수이기 때문에 상관관계가 나타나는 것으로 보인다.

-Hotkeys 관련 변수들 (SelectByHotkeys, AssignToHotkeys, UniqueHotkeys)
AssignToHotkeys와 UniqueHotkeys는 어느 정도 정규분포의 형태를 띠고 있으나 left skewed 되어 있는 것으로 보일 수 있다.

-Map 관련 변수 (MinimapAttacks, MinimapRightClicks, TotalMapExplored)
MinimapAttacks와 MinimapRightClicks는 대부분의 값이 0에 매우 가깝게 분포한 것으로 보인다. 그에 반해 TotalMapExplored는 정규분포에 가까운 분포를 보인다.

-그 외 변수들
Unit관련 변수들은 대체로 정규분포에 가까우면서도 left skewed 되어있는 형태를 가지고 있다. 게임플레이 시간을 표시한 변수들(HoursPerWeek, TotalHours)은 max 값이 비정상적으로 큰 값인 것으로 보인다.

2. 이상치 제거

먼저 각 변수의 boxplot을 확인하였다 [부록 참고].
전체 변수에 대해 한꺼번에 boxplot을 그려봤을 때, MaxTimeStamp(가장 우측)의 값이 범위가 가장 넓음을 확인할 수 있었다. 그리고, TotalHours에서 매우 큰 이상치가 존재함을 확인하였다. 각 변수의 왜도와 첨도를 확인해보았다. West et al(1995)의 정규분포 기준에 의하여 왜도의 절댓값이 3 미만이거나 첨도의 절댓값이 8 미만인 변수들을 정규분포라고 가정하여, 이를 벗어나는 값을 가지는 변수를 정규분포가 아니라고 판단하였다. 그 결과 TotalHours, MinimapAttacks, ComplexAbilityUsed가 왜도의 절댓값이 3 이상으로 왜도가 큼을 확인하였고, TotalHours, MinimapAttacks, ComplexAbilityUsed, HoursPerWeek, MiniapRightClicks, SelectByHotkeys, GapBetweenPACs가 첨도의 절댓값이 8 이상으로 첨도가 큼을 확인하였다. 그 중 TotalHours의 왜도, 첨도가 다른 변수들에 비해 눈에 띄게 높음을 알 수 있었다. 이상치를 판단하는 것이 목적이므로, 첨도가 큰 변수들이 이상치를 가지고 있을 가능성이 크다고 판단하여, 첨도가 큰 7개의 변수들을 하나씩 boxplot을 그려보며 의미와 더불어 이상치를 판단하였다.

	왜도	첨도
TotalHours	57.5653	3321.6885
MinimapAttacks	4.8192	45.5416
ComplexAbilityUsed	3.7789	21.5422
SelectByHotkeys	2.9653	11.3255
HoursPerWeek	2.6731	16.7369
MinimapRightClicks	2.5638	11.5824
ComplexUnitsMade	2.3014	6.1091
GapBetweenPACs	1.9083	9.3152
WorkersMade	1.6614	4.6244
ActionsInPAC	1.5990	7.3297
UniqueHotkeys	1.2244	3.5437
APM	1.2045	2.2529
TotalMapExplored	1.1847	2.6638
MaxTimeStamp	1.1645	4.1929
Age	1.1524	2.0073
ActionLatency	1.1517	2.6355
AssignToHotkeys	1.1413	2.8905
UniqueUnitsMade	0.7007	1.1637
NumberOfPACs	0.5504	0.6203

[표 3] 전체 변수의 왜도와 첨도

1) TotalHours - 첨도 : 3321.688498 [부록 참고]

TotalHours의 boxplot을 그려봤을 때, 다른 값들의 분포와 아주 동떨어진 이상치가 존재함을 확인할 수 있었다. describe()함수를 이용해 값의 통계치들을 확인해본 결과, 평균이 960.4218, Q1, Median, Q3가 각각 300, 500, 800인 것에 비해, max값이 1,000,000으로 매우 차이가 크게 남을 알 수 있었다.

TotalHours의 75%, 80%, 90%, 95%, 99% 값을 확인해본 결과, 99%의 값도 2520이므로, max값이 이상치임을 확인할 수 있었다. 따라서 우리는 데이터의 양이 적기 때문에 우선적으로 max값만 이상치 처리하여 제거하였다.

TotalHours의 max값을 제거한 후 boxplot을 그려본 결과 이전보다 비교적 4분위수 범위가 잘 보임을 확인했으나 여전히 분포가 치우쳐져 있으므로 로그 변환을 진행하여 범위를 좁히기로 하였다.

2) MinimapAttacks-첨도 : 45.541630 [부록 참고]

MinimapAttacks의 boxplot을 그려봤을 때, 다른 값들의 분포와 조금 동떨어진 이상치가 존재함을 확인할 수 있었다. describe()함수를 이용해 값의 통계치들을 확인해본 결과, 평균이 0.000098, Q1, Median, Q3가 각각 0.000166, 0.00, 0.000040, 0.000119인 것과 비교했을 때, max값이 0.003019로 차이는 나지만 애초에 범위 값이 적었다.

MinimapAttacks의 75%, 80%, 90%, 95%, 99% 값을 확인해본 결과, 99%의 값이 0.000769로 max값이 큰 이상치가 아님을 확인하였다. 따라서 이상치 제거를 진행하지 않고, 추후에 모델링을 통해 정규성을 만족하지 않을 경우 이상치 제거를 추가로

진행할 것이다.

ComplexAbilityUsed, MinimapRightClicks,SelectByHotkeys, GapBetweenPACs변수들도 MinimapAttacks의 경우와 비슷하므로 이상치 제거를 진행하지 않고 추후에 모델링 과정에서 필요하면 이상치 제거를 추가로 진행할 것이다.

3) HoursPerWeek- 첨도 : 16.736863 [부록 참고]

HoursPerWeek의 boxplot을 그려봤을 때, 0인 값들이 존재함을 확인할 수 있었다. HoursPerWeek는 일주일에 게임에 쓴 시간으로 0인 값이 존재할 수 없기 때문에, 이상치로 판단하였다. describe()함수를 이용해 HoursPerWeek 값들의 통계치들을 확인해본 결과, 평균이 15.910752, Q1, Median, Q3가 각각 8, 12, 20인 것과 비교했을 때, max값이 168로 차이가 크게 남을 확인할 수 있었다.

HoursPerWeek의 75%, 80%, 90%, 95%, 99% 값을 확인해본 결과, 99%의 값이 56이므로, max값 168이 이상치임을 확인할 수 있었다. 따라서, TotalHours와 마찬가지로 데이터의 양이 적기 때문에 우선적으로 max값과 0인 값만 이상치 처리하여 제거하였다.

HoursPerWeek의 max값을 제거한 후 boxplot을 그려본 결과 범위가 많이 좁혀졌고, 왜도도 2.22정도로 작아졌기 때문에 HoursPerWeek는 max값과 0만 제거하는 걸로 이상치 처리를 진행했다.

그 외에 0인 값들이 존재하는 변수들을 확인해봤다.

변수명	값이 0인 행의 개수
HoursPerWeek	1
SelectByHotkeys	16
AssignToHotkeys	6
UniqueHotkeys	189
MinimapAttacks	877
MinimapRightClicks	72
ComplexUnitsMade	2281

[표 4] 0인 값들이 존재하는 변수 확인

HoursPerWeek은 값이 0인 행이 1개뿐이었다. 하지만, 그 외에 0의 개수가 많은 변수들이 존재했기 때문에, 각 변수들의 의미를 파악하며 0이 이상치인지 아닌지 확인해봤다.

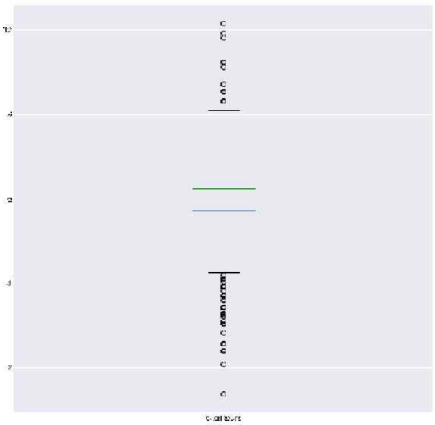
SelectByHotkeys와 AssignToHotkeys는 각각 타임스탬프당 사용된 고정/지정 단축키의 수를 의미한다. 게임 유저가 단축키를 쓰지 않을 수 있으므로 0인 값이 존재할 수 있다. 따라서 이 경우 0은 이상치가 되지 않는다. 또한 UniqueHotkeys는 타임스탬프당 사용된 특수유닛 단축키의 수는 단축키를 쓰지 않은 경우도 있으므로 0인 값이 존재할 수 있다.

MinimapAttacks는 타임스탬프당 미니맵 공격명령의 수로 미니맵 안에서 공격을 안할 수 있으므로 0인 값이 존재할 수 있다. 그리고 MinimapRightClicks는 타임스탬프

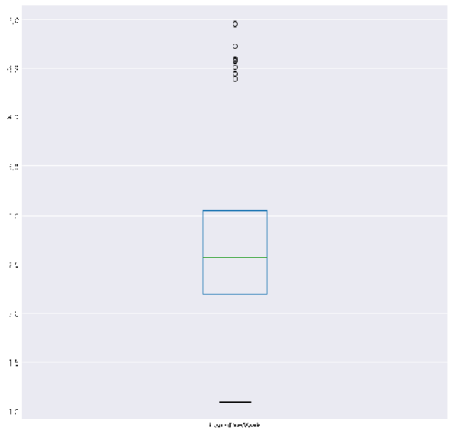
당 미니맵을 통한 이동명령의 수를 의미하므로 이동명령의 수가 0일 수 있기 때문에 0인 값이 존재할 수 있다. ComplexUnitsMade는 타임스탬프당 훈련된 ghost, infestor, high templar수를 의미하므로 이 역시 0인 값이 존재할 수 있다.
따라서 이 변수들이 0인 행들은 지우지 않았다.

3. 로그 변환

시간을 나타내는 TotalHours, HoursPerWeek 는 범위가 0~1인 다른 변수들보다 범위가 크기 때문에 범위를 축소할 필요가 있다. 따라서 TotalHours, HoursPerWeek 은 로그 변환을 진행했다. 다음은 로그 변환한 TotalHours, HoursPerWeek 의 그래프이다.

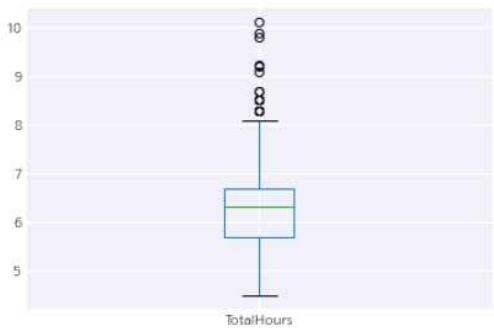


[그림 2] 로그변환 후 TotalHours 박스플랏



[그림 3]로그변환 후 HoursPerWeek 박스플랏

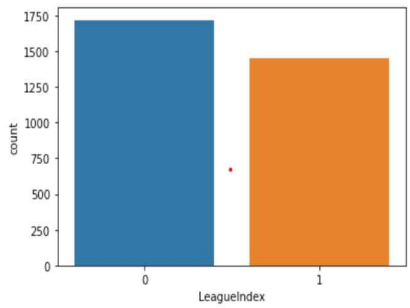
로그 변환한 TotalHours, HoursPerWeek 의 박스플랏을 확인한 결과 범위가 축소된 것을 확인했다.
하지만 TotalHours의 Q1 아래의 이상치가 존재함을 확인하여, 0.05 quantile 값 미만의 값을 제거하였다. 따라서 최종 TotalHours의 boxplot은 다음과 같다.



[그림 4] 로그 변환 후 + 이상치 제거 TotalHours의 boxplot

4. LeagueIndex 값 수정

LeagueIndex의 값은 1부터 8까지의 값으로 매핑된 티어(Bronze, Silver, Gold, Platinum, Diamond, Master, GrandMaster, Professional leagues)이다. 데이터 설명 논문 설명에선 LeagueIndex의 종류에 GrandMaster가 없고, 행의 수도 3360개로, 본 데이터의 행의 수 3395보다 35개 적다. LeagueIndex가 7인 행의 개수는 35개로 LeagueIndex가 7인 행이 추가된 것을 확인할 수 있었다. 따라서 LeagueIndex가 8인 행을 모두 7로 간주하여 LeagueIndex를 1~7값으로 변경했다. 이후, 로지스틱 회귀를 진행하기 위해, 약 50대 50의 비율로 변환하기 위해 분위값을 기준으로 판단하였다. 그 결과로, LeagueIndex 1~4값을 0 5~7값을 1로 변환하였다. 변환 후 LeagueIndex 0의 개수는 1718개 1의 개수는 1449개로 나뉘었다.



[그림 5] LeagueIndex 이분화 count plot

5. 결측치 대체

결측치 개수를 확인한 표는 다음과 같다.

	결측치 개수	결측치 비율
Age	55	0.016200
HoursPerWeek	56	0.016495
TotalHours	57	0.016789

[표 5] 결측치 개수와 비율 확인

결측치가 존재하는 변수들은 Age, HoursPerWeek, TotalHours 세 개다. 이들의 결측치 비율은 약 16%이다. 이 변수들은 모두 원 데이터의 LeagueIndex가 8인 행들로, 원 데이터의 LeagueIndex가 8인 행은 모두 이 변수들이 결측값이었다.
LeagueIndex값이 8인 행을 7로 수정하였기 때문에 이 결측값들을 LeagueIndex가 7인 행의 Age, HoursPerWeek, TotalHours의 평균으로 각각 대체하였다.

6. 중복 데이터 제거

GameID를 제외하고 모든 변수의 값이 같은 행이 존재했다.

```
data[data.duplicated(subset = sub_col,keep=False)]
```

	GameID	LeagueIndex	Age	HoursPerWeek	TotalHours	APM	SelectByHotkeys	AssignToHotkeys	UniqueHotkeys	MinimapAttacks	...	NumberOfPA
1401	4064	5	25.0	20.0	700.0	95.5704	0.167685	0.029345	0.005239	0.040873	...	0.2850
1409	4075	5	25.0	20.0	700.0	95.5704	0.167685	0.029345	0.005239	0.040873	...	0.2850

2 rows × 21 columns

[그림 6] 중복 데이터 확인

이 데이터는 데이터수집과정에서 온라인 설문조사 창의 오류 등의 이유로 중복 수집되었다고 판단하여 하나만 남기고 제거하였다.

7. 변수 값 수정 및 생성

1) 변수 시간 단위 수정 및 APS 변수 생성

타임스탬프당 변수들과 분당 변수들이 존재했다. 따라서 변수들의 시간 단위를 통일하기 위해 변수들의 시간 단위를 1초로 바꾸었다. 88.5 타임스탬프는 1초를 의미하므로, 시간 단위가 타임스탬프인 변수는 88.5를 곱하고, 시간 단위가 millisecond인 변수들은 1000을 나누었다. 시간 단위가 분(minute)인 APM을 60으로 나누어 시간 단위가 초(second)인 APS를 만들었다.

2) 변수 생성(HotkeyRate, MinimapkeyRate, PACtime, Unitmades)

[표 1]을 참고하여 단축키와 관련된 변수 3개(SelectByHotkeys, AssignToHotkeys, UniqueHotkeys), 미니맵과 관련된 변수 2개(MinimapAttacks, MinimapRightClicks), PAC와 관련된 변수 4개(NumberOfPACs, GapBetweenPACs, ActionLatency, ActionsInPAC), 유닛과 관련된 변수 3개(WorkersMade, UniqueUnitsMade, ComplexUnitsMade)를 각각 축소할 방법을 생각했다.

2-1) HotkeyRate, MinimapkeyRate

게임에는 여러 액션이 존재하며 이는 게임에 영향을 미친다. 어떠한 액션이 더 게임에 영향을 미치는지 비교를 하기 위해선 '총 액션 대비 개별 액션 비율'의 필요성을 느꼈고, 액션에 관련된 변수들을 하나로 합쳐 변수 축소를 진행하였다. 먼저, 단축키와 관련된 변수 중 SelectByHotkeys와 AssignToHotkeys는 UniqueHotkeys와 달리 데이터값이 현저히 낮은 것으로 보아 특수한 상황에서만 사용되는 단축키 사용임을 알 수 있다. 그래서 기본 상황에서의 단축키 사용 빈도를 알아보기 위해 UniqueHotkeys를 제외한 두 변수를 합하고, APS로 나누어 '단위 초당 총 액션 대비 단축키 액션'인 'HotkeyRate'를 생성하였다. 이와 같은 논리로 미니맵 사용 빈도를 알아보기 위해 관련 변수인 MinimapAttacks와 MinimapRightClicks 변수를 합하고 APS로 나눠 '단위 초당 총 액션 대비 미니맵 액션'을 생성하였다.

2-2) PACtime

PAC 관련 변수 중 시간 값을 가지는 GapBetweenPACs, ActionLatency에 초점을 맞추었다. PAC 정의에 따르면 위의 두 변수를 합친 결과 '인지를 시작하고 행동하기까지의 총시간'이 된다. 이는 게임을 진행하며 얼마나 빠르게 인지를 하고 행동으로까지 이어지는지를 한눈에 볼 수 있는 변수로 의미를 부여할 수 있음을 뜻한다.

2-3) UnitMade

Unit은 액션 결과의 의미도 있지만, 게임에 영향을 끼치는 자원으로서의 의미도 볼 수 있다. 특수한 상황이 아닌, 기본적인 상황에서의 유닛 생성을 보기 위해 0 값이 2281개나 존재하는 ComplexUnitsMade를 제외한 WorkersMade와 UniqueUnitsMade를 합쳐 'UnitsMade'를 생성하였다.

3) 변수 삭제

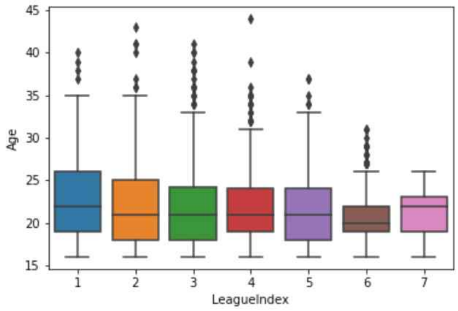
Full Model에 들어갈 설명변수를 지정하기 전, 변수 의미 해석 및 데이터 탐색 단계에서 근거를 발견한 변수들을 삭제하였다.

3-1) GameID

'게임마다 고유한 ID 숫자'를 의미하는 변수로, 모든 row에서 unique한 값을 가지는 Index 역할을 하기 때문에, LeagueIndex를 판별하는 목적에 부합하지 않다고 판단하여 삭제하였다.

3-2) Age

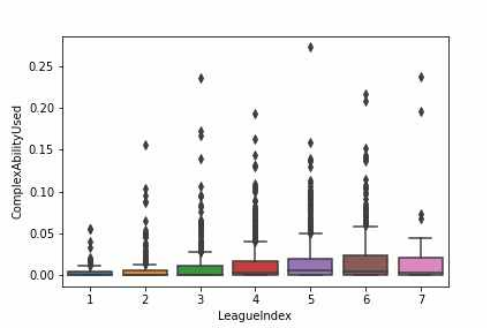
변수 간 상관관계를 보면 'LeagueIndex'와 -0.13으로 매우 낮은 값을 가지며, LeagueIndex별 boxplot을 보면 값별로 비슷한 분포를 띄는 것을 확인할 수 있다. 이와 같은 근거로 LeagueIndex를 판별하는 목적에 부합하지 않다고 판단하여 삭제하였다.



[그림 7] LeagueIndex-Age Boxplot

3-3) ComplexAbilityUsed

변수 의미상 ComplexAbilityUsed는 ComplexUnit의 사용하는 Ability의 사용횟수이기 때문에, ComplexUnit이 생성되어야만 값을 가질 수 있다. 두 변수의 상관관계 또한 0.62로 유의미한 양의 상관관계를 보였다. LeagueIndex별 boxplot을 보면 값별로 비슷한 분포를 띄는 것을 확인할 수 있다. ComplexUnitsMade의 0 값이 총 데이터의 2/3를 차지하는 2281개나 존재하고, 위와 같은 근거들이 뒷받침되어 변수삭제를 진행하였다.



[그림 8] LeagueIndex-ComplexAbilityUsed Boxplot

제3절 모델링

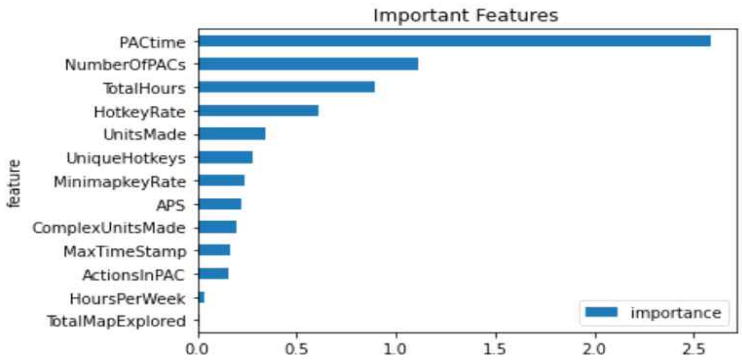
1. 로지스틱 회귀모델

1) Full Model

전처리 단계에서 생성, 삭제된 변수들이 반영된 Full Model은 1개의 반응변수(LeagueIndex)와 13개의 설명변수로 이루어져 있다.

설명변수 = ['HoursPerWeek', 'TotalHours', 'UniqueHotkeys', 'NumberOfPACs', 'ActionsInPAC', 'TotalMapExplored', 'ComplexUnitsMade', 'MaxTimeStamp', 'APS', 'HotkeyRate', 'MinimapkeyRate', 'PACtime', 'UnitsMade']

train과 test는 7:3 비율로 나누었으며, Scaling을 통해 regularization을 진행하였다. 그 이유는 로지스틱 회귀가 기본적으로 L2-norm loss function을 사용하기 때문이다. L2-norm이 문제가 되는 이유는 LSE(최소제곱 오차)를 사용하기 때문에 값이 큰 변수와의 거리를 측정할 때 과적합 문제가 일어날 수 있다. 이러한 이유로 본 조는 과적합 문제를 미연에 방지하기 위해 StandardScaling을 사용하였다. 정확도는 0.800이며 최적 Threshold=0.558을 찾아 다시 fitting한 결과 정확도는 0.807이 측정되었다. model의 변수별 coefficient 값을 logit 변환하여 변수 중요도를 생성한 결과 아래와 같은 결과를 보였다.



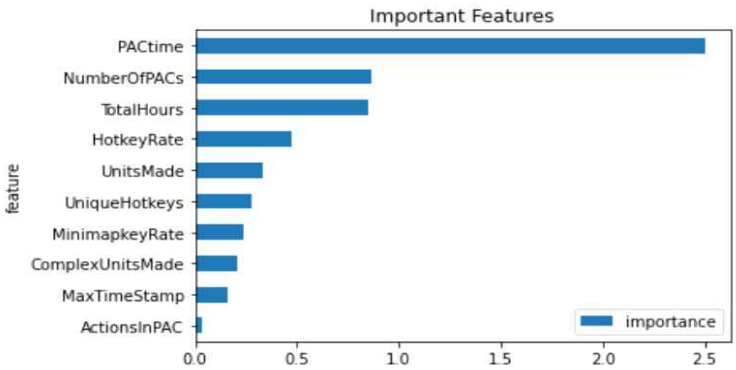
[그림 9] Full Model 변수 중요도

본 조는 Full Model에서 TotalMapExplored, HoursPerWeek, APS 총 3개의 변수

를 삭제하였다. TotalMapExplored의 경우, 매우 낮은 변수 중요도와 더불어 변수 의미상 게임별 Map, 게임 시간 등 다양한 요소에 달라질 수 있기 때문에 삭제를 진행하였다. HoursPerWeek의 경우, 매우 낮은 변수 중요도와 더불어 게임 시간을 의미하는 TotalHours가 있기에 삭제하였다. 마지막으로 APS는 새로 생성한 HotkeyRate, MinimapkeyRate가 APS로 나누었기에 다중공선성의 가능성이 농후하며, 변수 중요도 역시 이 둘보다 낮기에 삭제하였다.

2) 1차 모델

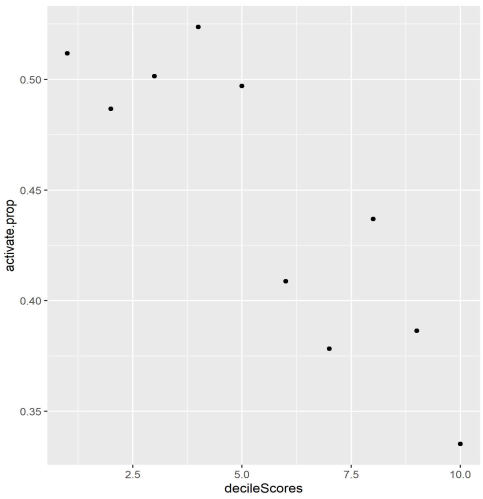
1차 수정 모델은 1개의 반응변수(LeagueIndex)와 10개의 설명변수로 이루어져 있다.
설명변수 = ['TotalHours', 'UniqueHotkeys', 'NumberOfPACs', 'ActionsInPAC', 'ComplexUnitsMade', 'MaxTimeStamp', 'HotkeyRate', 'MinimapkeyRate', 'PACtime', 'UnitsMade']
모델링 과정은 앞 과정과 동일하며 정확도는 0.803, 최적 Threshold=0.503를 찾아 다시 fitting한 결과 정확도는 0.804가 측정되었다. Full Model보다 변수가 3개 줄어들었음에도 불구하고 정확도가 상승한 점을 볼 수 있다. 모델의 변수 중요도를 생성한 결과 아래와 같은 결과를 보였다.



[그림 10] 1차모형 변수 중요도

본 조는 1차 모델에서 ActionsInPAC 변수를 삭제하였다. Action을 대표하는 HotkeyRate, MinimapkeyRate가 이미 존재하며, 변수 의미 상 PAC 내에서의 Action은 통일된 PACtime에서 결정된 값이 아니기 때문에 변수중요도가 낮게 나왔다고 판단하여 삭제하였다.

다음으로, 1차 모델 변수 선택 과정 이후 독립변수 전체에 대한 십분위 분석을 진행하였다. 십분위분석 결과 ComplexUnitsMade가 monotonous하게 감소하지 않고 두 가지 정도의 범주로 분류할 수 있다고 판단하였다. 위의 표를 통해 십분위 1~5까지를 1로, 6~10까지를 0으로 변환하여 변수를 세분화하였다.



[그림 11] ComplexUnitsMade의 십분위 plot

3) 2차 모델

2차 수정 모델은 1개의 반응변수(LeagueIndex)와 9개의 설명변수로 이루어져 있다.

설명변수 = ['TotalHours', 'UniqueHotkeys', 'NumberOfPACs', 'ComplexUnitsMade', 'MaxTimeStamp', 'HotkeyRate', 'MinimapkeyRate', 'PACtime', 'UnitsMade']

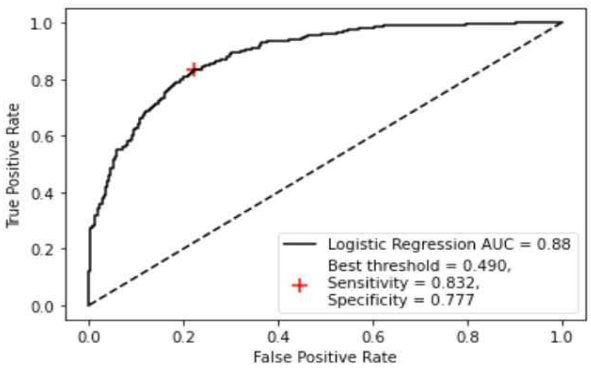
모델링 과정은 앞 과정과 동일하며 정확도는 0.801, 최적 Threshold=0.490를 찾아 다시 fitting한 결과 정확도는 0.802가 측정되었다.

decileScores	n	n_LeagueIndex_1	activate.prop
1	318	249	0.783
2	360	264	0.733
3	307	188	0.612
4	205	116	0.566
5	466	240	0.515
6	338	133	0.393
7	348	121	0.348
8	302	93	0.308
9	404	83	0.205
10	345	28	0.0812

[표 6] ComplexUnitsMade의 십분위 표

변수	범주	관측치수	비율(%)
ComplexUnitsMade	0	2279	32.83
	1	1114	67.16

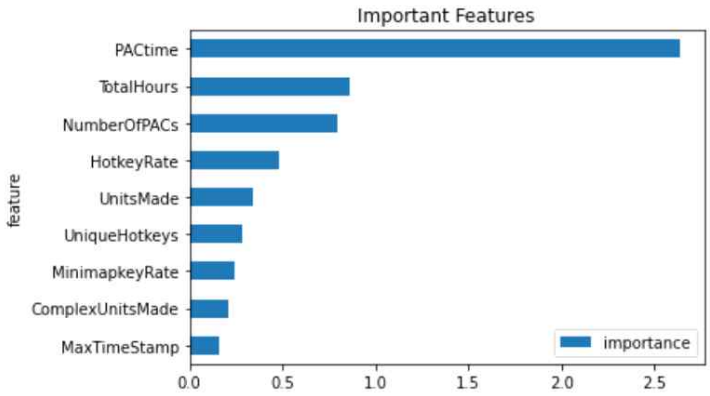
[표 7] ComplexUnitsMade의 십분위 표



[그림 12] 2차 모델 ROC curve

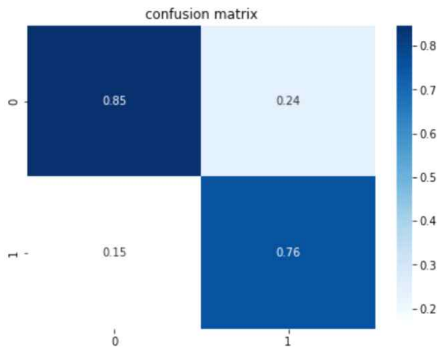
Full Model보다 변수가 줄어들었음에도 불구하고 정확도가 비슷하였다. 모델의 변수

중요도를 생성한 결과 아래와 같은 결과를 보였다.



[그림 13] 2차 모델 변수 중요도 plot

아래 그림을 참고하면, 0을 정확히 분류할 확률은 0.85, 1을 정확히 분류할 확률은 0.76을 보인다.



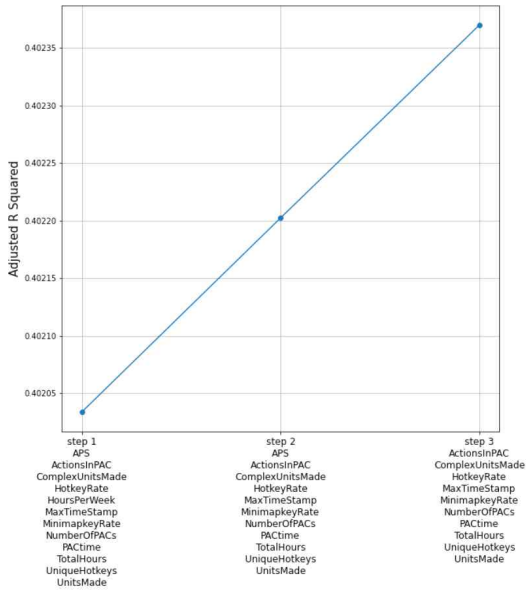
[그림 14] 2차 모델 confusion-matrix

4) 비교

본 조의 최종모델 변수 선택 과정과 비교를 하기 위해 Full Model에서의 후진 선택법 변수 선택을 진행하였다. 그 결과 'TotalMapExplored', 'HoursPerWeek', 'APS' 순으로 변수 제거를 하였으며, adjusted R squared 값 역시 증가하고 있는 양상을 보였다. 따라서 본 조의 변수 선택법과 매우 유사한 결과를 보여주었다.

2. autoML

본 데이터는 정형데이터이므로, 정형데이터의 모델 적합에 유용하게 쓰이는 파이썬의 pycaret.classification 패키지를 이용하여 머신러닝 모델링을 확인해봤다.
반응 변수를 LeagueIndex로 두고 setup한 결과는 다음과 같다.



[그림 15] 후진 선택법 adjusted R squred plot

	Description	Value
0	session_id	8206
1	Target	LeagueIndex
2	Target Type	Binary
3	Label Encoded	0: 0, 1: 1
4	Original Data	(3167, 10)
5	Missing Values	False
6	Numeric Features	7
7	Categorical Features	2
8	Ordinal Features	False
9	High Cardinality Features	False
10	High Cardinality Method	None
11	Transformed Train Set	(2216, 2198)
12	Transformed Test Set	(951, 2198)
13	Shuffle Train-Test	True
14	Stratify Train-Test	False
15	Fold Generator	StratifiedKFold

[그림 16] autoML setup

10 Fold를 진행하여 분류모델을 돌린 결과 Accuracy가 높은 상위 3개의 모델은 Ridge Classifier, Logistic Regression, Extra Trees Classifier이었다.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
ridge	Ridge Classifier	0.7974	0.0000	0.7838	0.7771	0.7798	0.5922	0.5930	0.287
lr	Logistic Regression	0.7951	0.8809	0.7769	0.7766	0.7761	0.5873	0.5882	0.398
et	Extra Trees Classifier	0.7951	0.8731	0.7562	0.7895	0.7716	0.5860	0.5875	2.703

[그림 17] Accuracy 상위 3개의 모델

이 세 개의 모델의 하이퍼 파라미터는 각각 다음과 같다.

- Ridge Classifier의 하이퍼 파라미터

```
RidgeClassifier(alpha=1.0, class_weight=None, copy_X=True,
                fit_intercept=True,max_iter=None,
                normalize=False, random_state=8206,solver='auto', tol=0.001)
```

- Logistic Regression의 하이퍼 파라미터

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=1000,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=8206, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

- RandomForest Classifier의 하이퍼 파라미터

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=-1, oob_score=False, random_state=8206,
                      verbose=0, warm_start=False)
```

이 세개의 모델을 블렌딩한 모델은 다음과 같다.

```
blended = blend_models(estimator_list = best_3, fold = 5)
```

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7995	0.0	0.7389	0.8065	0.7712	0.5935	0.5953
1	0.7810	0.0	0.7833	0.7500	0.7663	0.5605	0.5610
2	0.8217	0.0	0.8276	0.7925	0.8096	0.6421	0.6426
3	0.7698	0.0	0.7475	0.7475	0.7475	0.5359	0.5359
4	0.8014	0.0	0.7723	0.7879	0.7800	0.5990	0.5991
Mean	0.7947	0.0	0.7739	0.7769	0.7749	0.5862	0.5868
Std	0.0179	0.0	0.0313	0.0238	0.0203	0.0361	0.0363

[그림 18] Accuracy 상위 3개 모델을 블렌딩한 모델

로지스틱 모델의 Accuracy는 0.802로 블렌딩한 모델의 평균 Accuracy인 0.7947보다 높은 것을 확인할 수 있었다.

제3장 결론

제1절 분석결과 정리

1) 로지스틱 회귀모델 해석

feature	coef	odds ratio
TotalHours	0.619327	1.857677
NumberOfPACs	0.583666	1.792598
HotkeyRate	0.390594	1.477859
UnitsMade	0.296532	1.345185
UniqueHotkeys	0.246443	1.279467
MinimapkeyRate	0.213711	1.238265
ComplexUnitsMade	0.186826	1.205417
MaxTimeStamp	-0.144834	0.865166
PACtime	-1.290275	0.275195

[표 8] 최종 모형 feature coefficient와 odsds ratio

위의 결과를 바탕으로 변수 중요도 상위 n개 변수의 오즈비에 대해 해석하면 다음과 같다.

1. PACtime: PACtime이 1단위씩 증가할 때마다 상위 랭크일 확률은 약 0.275배 증가한다.
2. TotalHours: TotalHours 1단위씩 증가할 때마다 상위 랭크일 확률은 약 1.858배 증가한다.
3. NumberOfPACs: NumberOfPACs 1단위씩 증가할 때마다 상위 랭크일 확률은 약 1.793배 증가한다.
4. HotkeyRate: HotkeyRate 1단위씩 증가할 때마다 상위 랭크일 확률은 약 1.478배 증가한다.
5. UnitsMade: UnitsMade 1단위씩 증가할 때마다 상위 랭크일 확률은 약 1.345배 증가한다.

2) 모형 평가

1. train data

Decile	N	Cum % of	Predicted	Precent Rank High	Cum % of Rank High	# of Rank High	% of Total Rank High	Cum # of Rank High	Cum % of Total Rank High	Lift(%)	Cum Lift(%)
1	222	10	96.07838	95.04504505	95.04504505	211	20.81	211	20.81	207.7119	207.7119
2	221	20	87.89521	89.14027149	92.0993228	197	19.43	408	40.24	194.8075	201.2743
3	222	30	77.69161	75.67567568	86.61654135	168	16.57	576	56.81	165.3819	189.2922
4	221	40	65.25135	69.23076923	82.27990971	153	15.09	729	71.9	151.2972	179.8149
5	222	50	50.73381	49.0990991	75.63176895	109	10.75	838	82.65	107.3014	165.286
6	221	60	36.52006	36.19909502	69.0744921	80	7.89	918	90.54	79.10966	150.9557
7	222	70	23.58694	26.57657658	62.99161831	59	5.82	977	96.36	58.08057	137.6622
8	221	80	12.95061	10.40723982	56.43340858	23	2.27	1000	98.63	22.74403	123.3298
9	222	90	5.795234	5.405405405	50.75225677	12	1.18	1012	99.81	11.813	110.9142
10	222	100	1.166077	0.900900901	45.75812274	2	0.2	1014	100.01	1.968833	100

[그림 19] train data 10분위 분석표

모형평가를 위해 train data set을 이용하여 십분위 분석표를 작성하였다. N은 각 분위에 해당하는 관측치의 개수, Predicted probability는 y(target variable)=1일 예측확률을 나타낸다. 모든 분위에서 단조롭게 Actual 값이 줄어들고 있으므로 모형은 train data set에 대해 양호하게 작동한다고 판단할 수 있다. <부록 참고>

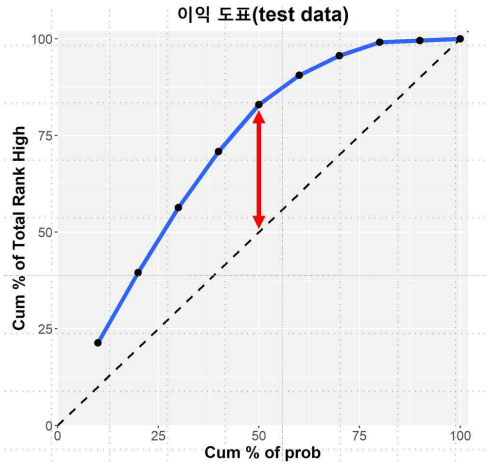
2. test data

Decile	N	Cum % of prob	Predicted prob	Precent Rank High	Cum % of Rank High	# of Rank High	% of Total Rank High	Cum # of Rank High	Cum % of Total Rank High	Lift(%)	Cum Lift(%)
1	95	10	96.2438964	97.89473684	97.89473684	93	21.38	93	21.38	214.0181488	214.0181488
2	95	20	88.81119692	83.15789474	90.52631579	79	18.16	172	39.54	181.800363	197.9092559
3	95	30	79.07944243	76.84210526	85.96491228	73	16.78	245	56.32	167.9927405	187.9370841
4	95	40	67.64235561	66.31578947	81.05263158	63	14.48	308	70.8	144.9800363	177.1978221
5	95	50	55.57103576	55.78947368	76	53	12.18	361	82.98	121.9673321	166.1517241
6	95	60	42.43081568	34.73684211	69.12280702	33	7.59	394	90.57	75.94192377	151.1167574
7	95	70	29.56999206	23.15789474	62.55639098	22	5.06	416	95.63	50.62794918	136.7612134
8	95	80	17.30722755	15.78947368	56.71052632	15	3.45	431	99.08	34.51905626	123.9809437
9	95	90	8.151597687	2.105263158	50.64327485	2	0.46	433	99.54	4.602540835	110.7166767
10	96	100	1.872032166	2.083333333	45.74132492	2	0.46	435	100	4.554597701	100

[그림 20] test data 10분위 분석표

이어서 test data set에 대해서도 똑같이 십분위 분석표를 작성하였다. 각 변수가 의미하는 바는 training data set의 십분위 분석표에서와 같다.
모든 분위에서 단조롭게 Actual 값이 줄어들고 있으므로 모형은 test data set에 대해서도 양호하게 작동한다고 판단할 수 있다. <부록 참고>

3. 이익 도표 해석 (2의 test data 10분위 분석표와 연결)



[그림 21] test data 이익도표

이익 도표를 통해 평균적인 모형의 적합성에 비해 구축된 모형이 얼마나 잘 적용이 되는지 알아보았다. 다음은 test data set 십분위 분석표를 통해 얻은 이익 도표이다. 1그룹에서는 평균보다 y=1인 관측값을 2.14배 더 포함하고 있고, 2그룹은 y=1인 관측값을 1.81배 더 포함하고 있다. 5그룹까지는 구축된 모형이 평균

모형보다 우수함을 확인할 수 있다. 또한, 전체 데이터의 50%만으로 y=1인 값들 중 약 83%를 찾을 수 있으며, 이는 모형을 구축하지 않았을 때보다 약 61% 증가한 것이다. 즉, 전체 데이터의 50%만을 사용하여 구축된 모형으로 평균 모형보다 83% 더 많은 상위 랭크 게임 플레이어를 찾을 수 있다. 이를 통해, 로지스틱 모델과 십분위 분석을 통하여 모델의 성능과 적합성이 적절함을 검증하였다.

제2절 한계점과 기대효과

1. 한계점

데이터 적 측면

1)

각 게임당 맵에 대한 정보를 알았다면, 맵의 위치 정보를 통해 새로운 특성과 전략을 알아낼 수 있었을 것이다.

생산한 유닛이나 건물에 대한 정보 등 게임 내 외적으로 데이터가 정보를 적게 담고 있다고 할 수 있다.

게임 플레이어의 종족을 알 수 있었다면, 종족별로의 상성 등의 정보를 통해 게임의 승패 예측에도 기여할 수 있었을 것이다.

게임 플레이어의 한 게임 정보가 아닌, 다수의 플레이 정보가 있다면, 게임 플레이어의 능력치를 환산하여 분석에 기여할 수 있었을 것이다.

분석적 측면

2)

로지스틱뿐만 아니라 따로 진행하였던, AutoML을 사용한 분류 모델에서도

종속 변수를 세분화하지 않고 0과 1의 이분법적인 측면에서만 바라보았다는 것이다.

원 데이터의 LeagueIndex 1부터 8까지가 아니더라도 플레이어의 랭크를 세부적으로 나눌 수 있는 범주값을 늘려 분류를 진행하지 못한 점이다.

2. 기대효과

본 조는 스타크래프트2의 실력을 향상 시킬 수 있는 요인들을 알아보기 위해 분석을 실시한 결과 다음과 같은 사실을 파악할 수 있었다. LeagueIndex를 향상시키기 위해서는 먼저 PACtimes라는 것은 인지과정 시간을 줄여야 한다. 전략 시뮬레이션 게임이기 때문에 여러 전략을 세우기도 하고, 여러 가지 상황들을 당면하는데, 예를 들면 상대의 진영을 찾기 위해 정찰을 나갔는데 입구에서 상대의 유닛이 보인다면 많은 가능성을 생각해 볼 수 있다. 크게 두 가지 정도의 가능성을 보자면 '상대도 정찰을 위해 유닛을 밖으로 내보냈다' 또는 '상대가 초반에 승부를 보기 위해 공격을 하러온다' 등의 판단을 할 수 있고 이러한 판단을 빠르고 정확하게 할수록 순위상승에 긍정적인 영향을 미친다.

두 번째는 TotalHours이다. 이 게임에 접속한 시간을 나타내는 데이터인데, 여러 전략과 단축키와 같은 기능들이 숙련되었을수록 빠른 자원채취와 유닛 생산이 가능하기 때문에 중요한 변수로 알 수 있다. 많은 시간을 들여 게임을 진행할수록 실력이 향상된

다.

세 번째로 NumberOfPAC 인데 인지하고 행동을 하는데 게임 시간에 비례해서 많으면 많을수록 순위가 높아진다. 라고 하신다면 유닛생산, 건물을 건축하여 인프라 구축, 전투 수행 모두 인지 행동에 포함되기 때문에 의미 있는 변수이다. 상대보다 더 많은 정보를 수집하고, 상대방보다 많은 유닛을 생산했다면 상대편보다 많은 PAC를 거친것이고 승리에 가까워질 것이다.

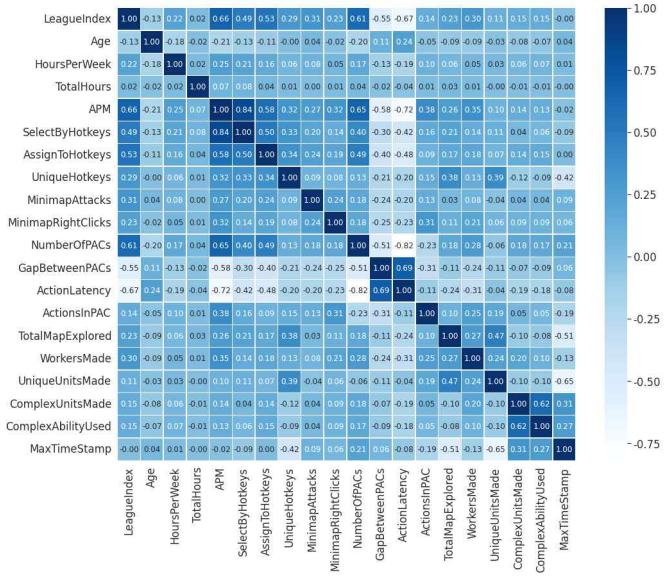
HotkeyRate란 단축키를 사용한 비율이며 컴퓨터에는 키보드와 마우스로 입력을 하는데 플레이어의 편의를 위한 단축키 기능이 있다. 간단한 명령은 이동 또는 공격 명령이 있다. 하지만 간단한 명령인 만큼 일회성이 이다. 단축키의 한 기능 중에는 유닛들을 선택하여 P버튼을 활용하여 맵의 두 지점을 정하면 이유닛은 두지점을 왕복하며 공격범위내에 들어온 상대방의 유닛을 공격한다. 아무리 멀티 태스킹 능력이 뛰어 나더라도 게임 내에서 발생하는 모든 사건에 관여할 수는 없으므로 단축키를 이용하면 좀 더 중요한 지점에 정신을 집중하는 것이 가능하다. 그러므로 같은 역량의 상대를 만났을 때 단축키의 사용여부는 승부를 결정짓는 데에 중요한 분기점이 될 수 있다.

UnitsMade는 유닛의 생산량이라고 할 수 있는데 일부 건물을 제외하고는 건물들은 공격할 수 없고, 공격이 가능하더라도 고정되어있기 때문에 결국 승리를 위해서는 유닛들을 최대한 많이 생산해야 한다. 시간당 유닛의 생산량이 많으면 많을수록 상대방의 병력과의 전투에서 우위를 정할 수 있다.

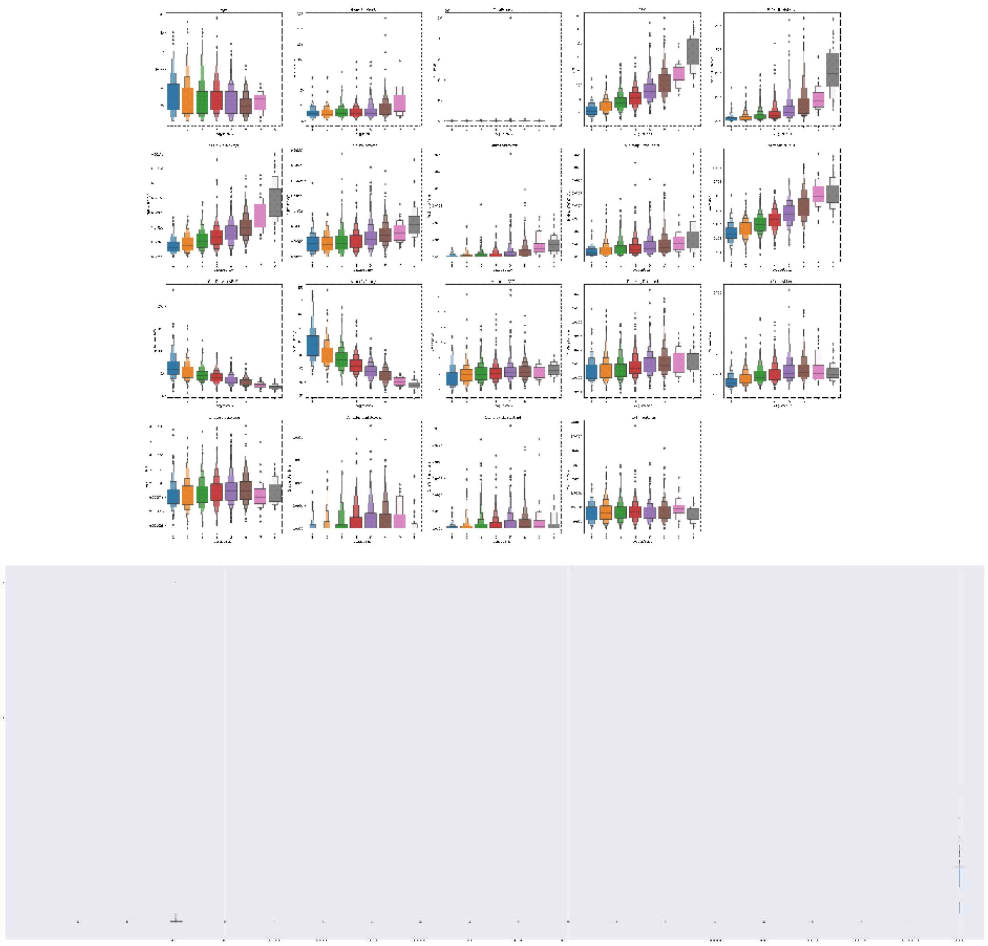
적용 방안)

위 변수들이 순위 향상에 영향을 미침을 볼 때 TotalHours와 HotkeyRate의 상승은 순위와 함께 나머지 변수의 개선도 기대해 볼 수 있다. 따라서 플레이어들의 플레이타임 상승과 실력을 개선시키기 위한 일환으로 여러 가지 상황을 유도할 수 있는 싱글 플레이 모드를 제안하는 바이다. 게임 내에서 단축키를 적극적으로 활용하도록 유도하거나, 게임 중 일어날 수 있는 상황을 설정한 후에 목표를 완료한다면 난이도에 따른 적절한 보상을 제시하는 것이다. 그리고 게이머들의 흥미를 유발하는 스토리라인이 있다면, 플레이 시간은 늘어나고, 단축키에 대한 숙련도도 향상되어 더욱 치열하고, 흥미진진한 리그가 될 것이다.

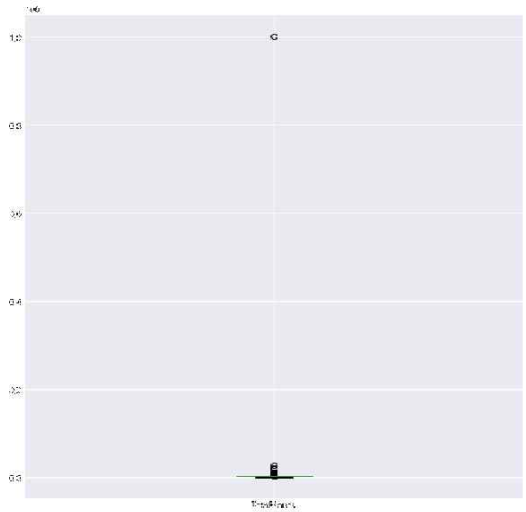
부록



[참고1] 변수간 상관계수 확인 히트맵



[참고2] 전체 변수에 대한 boxplot



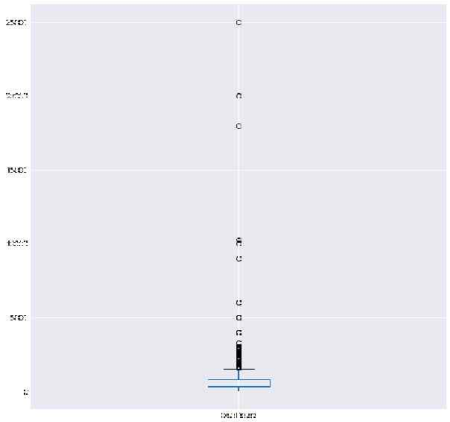
[참고 3] TotalHours Boxplot

	TotalHours
count	3338.000000
mean	960.421809
std	17318.133922
min	3.000000
25%	300.000000
50%	500.000000
75%	800.000000
max	1000000.000000

[참고 4] TotalHours의 통계치들

	TotalHours
0.75	800.0
0.80	900.0
0.90	1200.0
0.95	1566.0
0.99	2520.0

[참고 5] TotalHours의 75%, 80%, 90%, 95%, 99% 값



```
df_cut_max['TotalHours'].kurt()
```

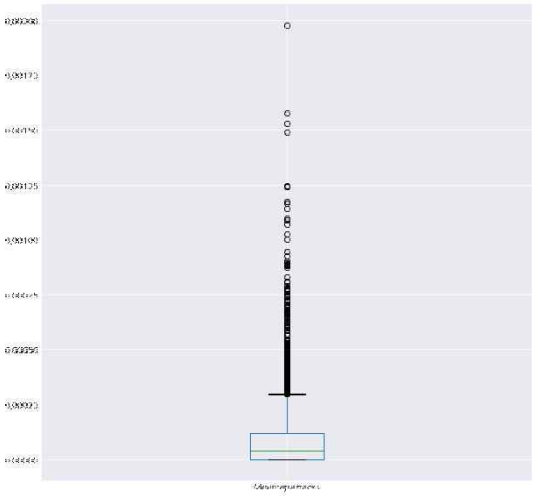
```
3320.693402774889
```

```
df_cut_max['TotalHours'].skew()
```

```
57.55672625232161
```

[참고 6] max값 제거 후 TotalHours의 boxplot

[참고 7] max값 제거 후 TotalHours의 첨도, 왜도



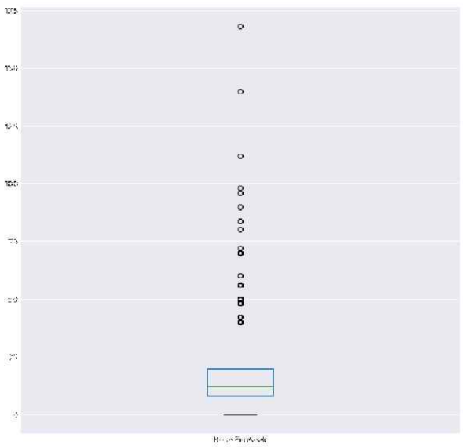
[참고 8] MinimapAttacks Boxplot

	MinimapAttacks
count	3395.000000
mean	0.000098
std	0.000166
min	0.000000
25%	0.000000
50%	0.000040
75%	0.000119
max	0.003019

[참고 9] MinimapAttacks의 통계치들

	MinimapAttacks
0.75	0.000119
0.80	0.000154
0.90	0.000261
0.95	0.000395
0.99	0.000769

[참고 10] MinimapAttacks의 75%, 80%, 90%, 95%, 99% 값



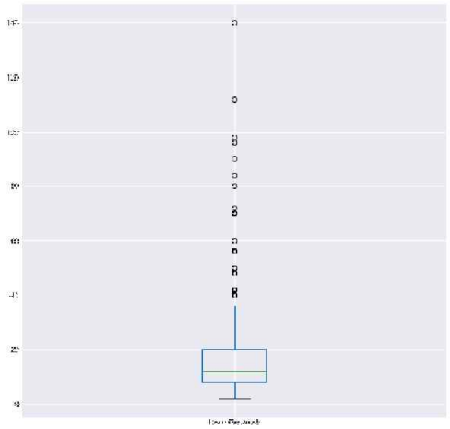
[참고 11] HoursPerWeek Boxplot

	HoursPerWeek
count	3339.000000
mean	15.910752
std	11.962912
min	0.000000
25%	8.000000
50%	12.000000
75%	20.000000
max	168.000000

[참고 12] HoursPerWeek 의 통계치들

	HoursPerWeek
0.75	20.0
0.80	24.0
0.90	28.0
0.95	40.0
0.99	56.0

[참고 13] HoursPerWeek의 75%, 80%, 90%, 95%, 99% 값



```
In [435]: df_cut_max['HoursPerWeek'].kurt()
Out[435]: 10.175408894671383

In [436]: df_cut_max['HoursPerWeek'].skew()
Out[436]: 2.2286386471170245
```

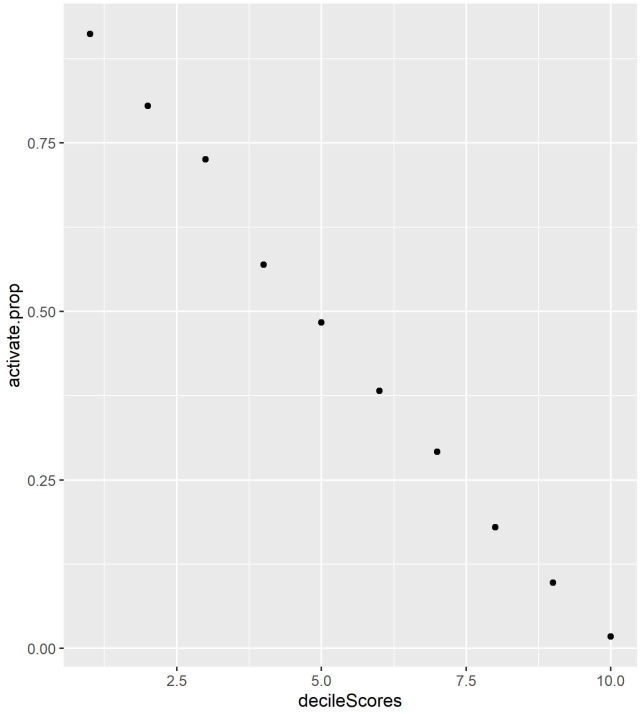
[참고 14] max, 0값 제거 후 HoursPerWeek의 boxplot

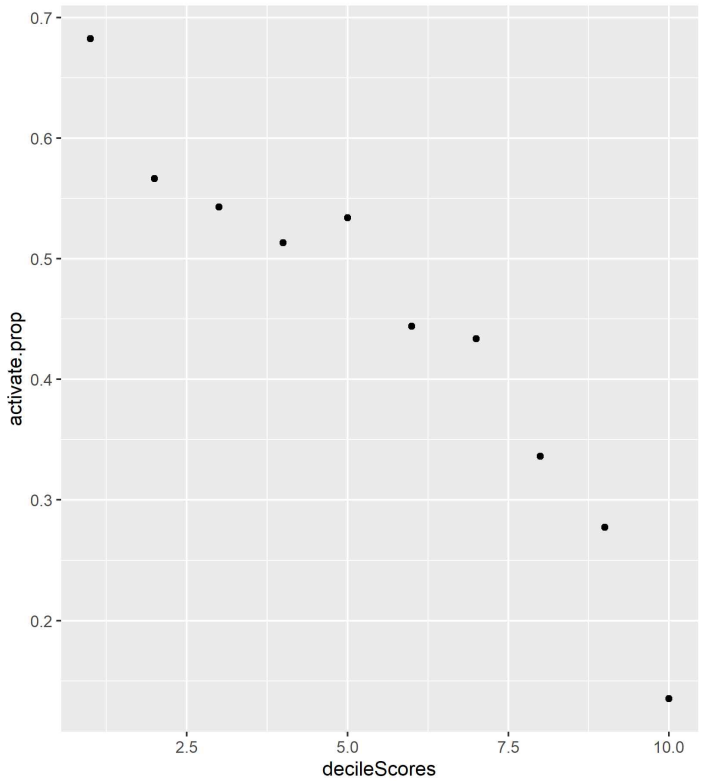
[참고 15] max, 0값 제거 후 HoursPerWeek의 첨도,왜도

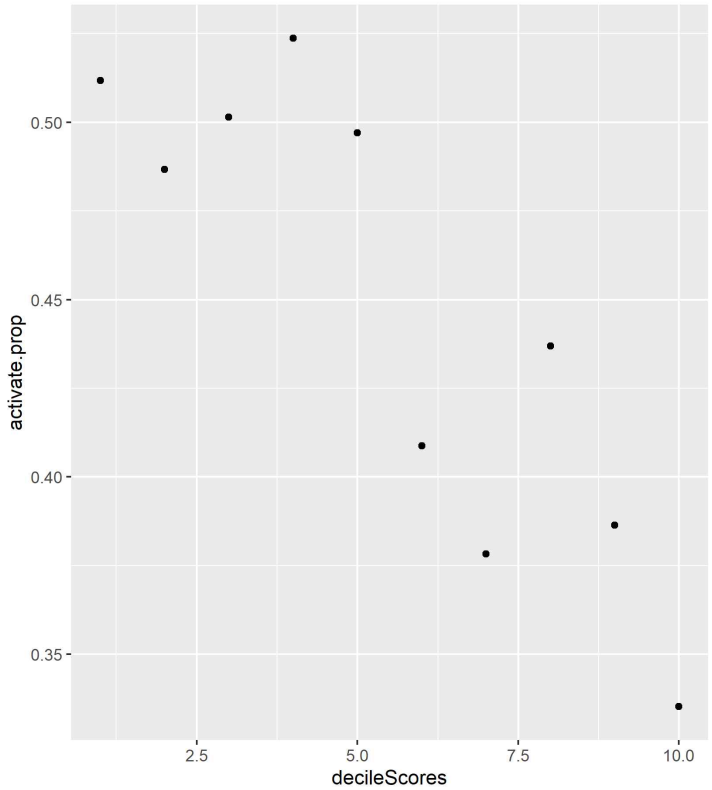
	Age	HoursPerWeek	TotalHours
3340	NaN	NaN	NaN
3341	NaN	NaN	NaN
3342	NaN	NaN	NaN
3343	NaN	NaN	NaN
3344	NaN	NaN	NaN
3345	NaN	NaN	NaN
3346	NaN	NaN	NaN
3347	NaN	NaN	NaN
3348	NaN	NaN	NaN
3349	NaN	NaN	NaN
3350	NaN	NaN	NaN
3351	NaN	NaN	NaN
3352	NaN	NaN	NaN
3353	NaN	NaN	NaN

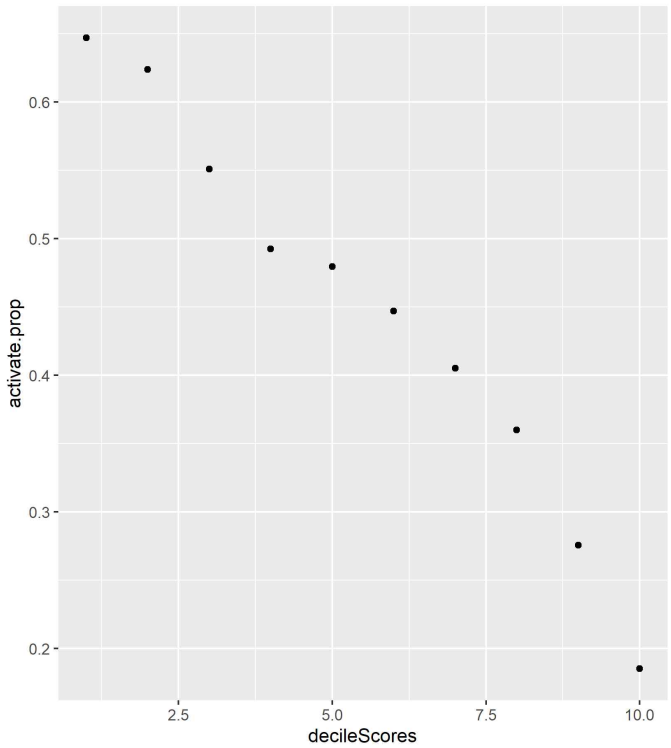
[참고 16] 원 데이터의 LeagueIndex 8인 행 확인

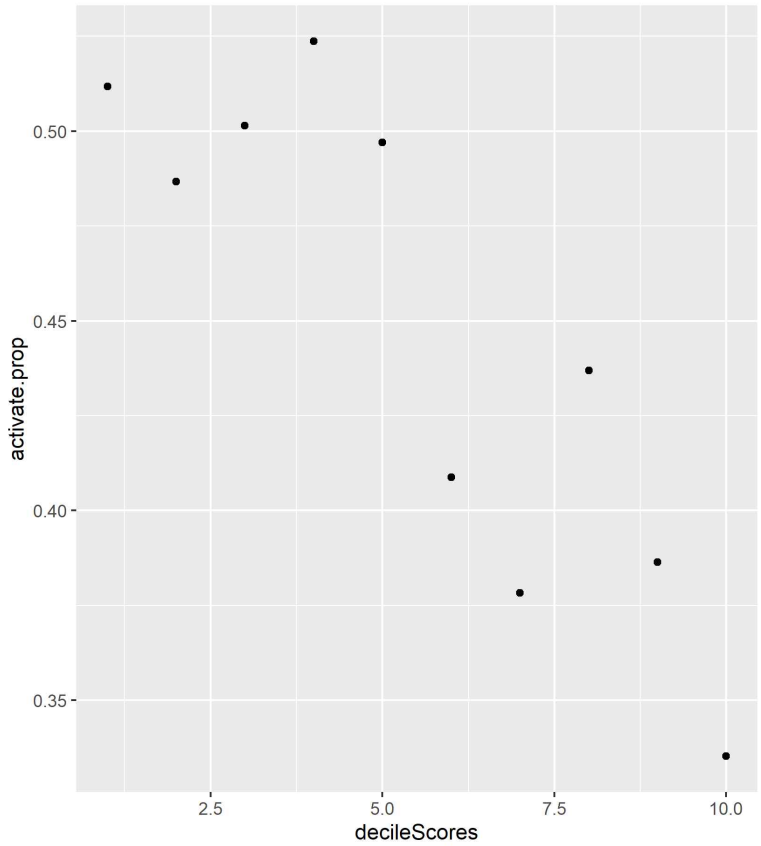
[참고 17] 변수별 십분위 분석 활성화 확률plot

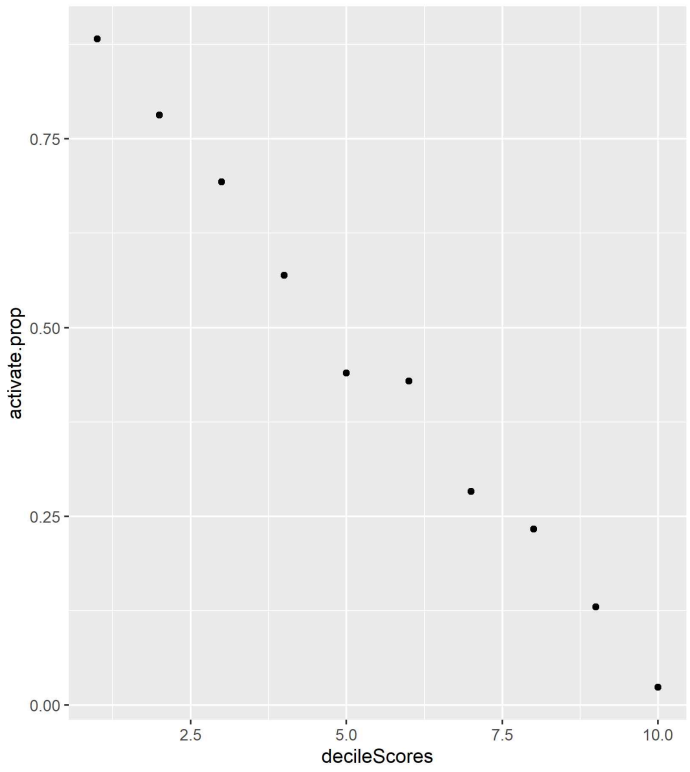


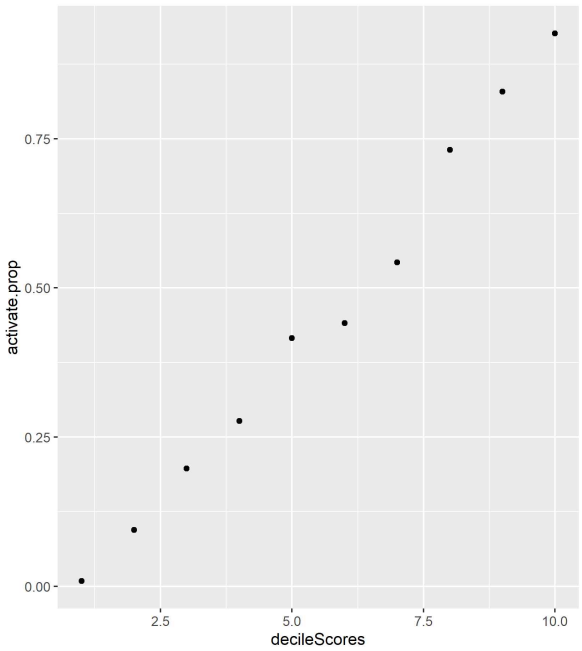


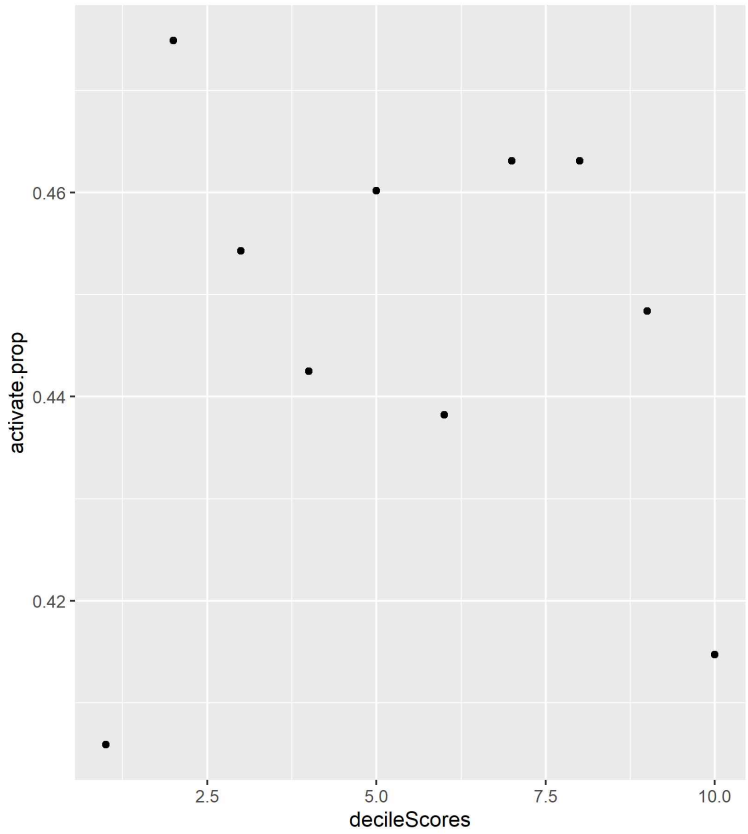

0.50
0.45
0.40
0.35
activate.prop
2.5
5.0
7.5
10.0
decileScores

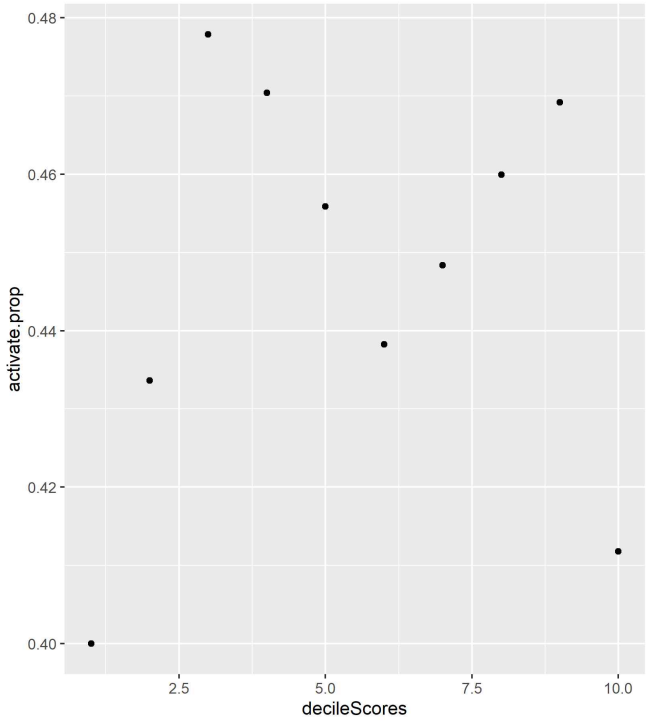

0.6
0.5
0.4
0.3
0.2
activate.prop
2.5
5.0
7.5
10.0
decileScores

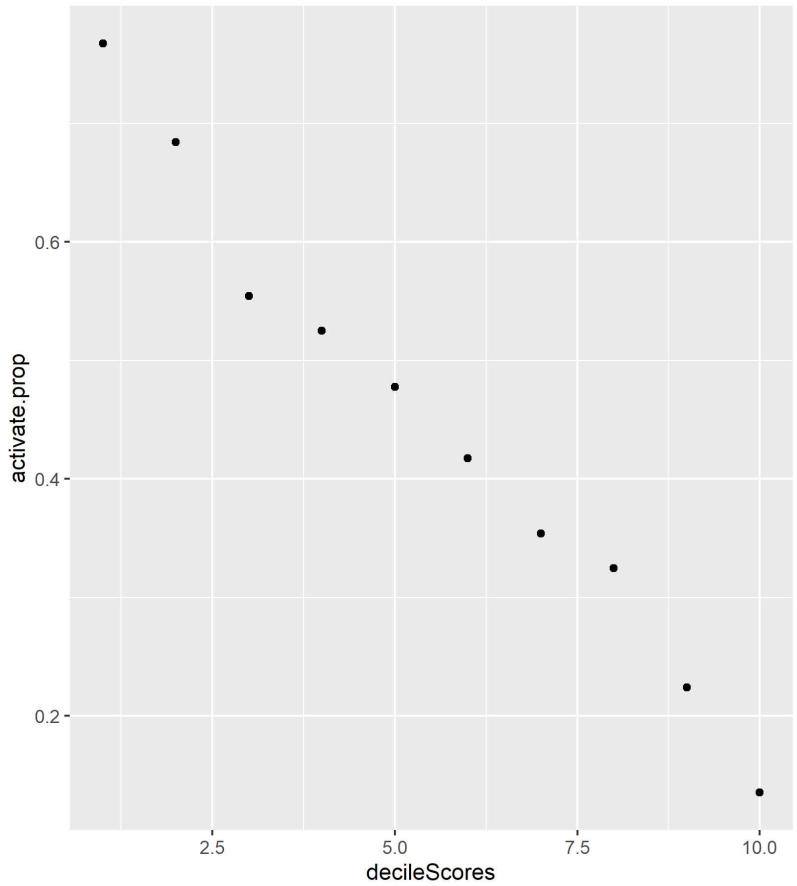

0.50
0.45
0.40
0.35
activate.prop
2.5
5.0
7.5
10.0
decileScores


0.75
0.50
0.25
0.00
activate.prop
2.5
5.0
7.5
10.0
decileScores

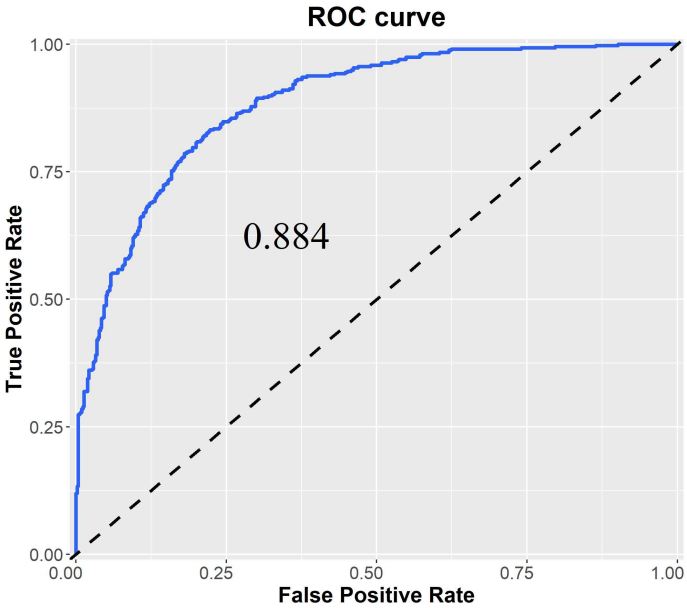

0.75
0.50
0.25
0.00
activate.prop
2.5
5.0
7.5
10.0
decileScores


0.46
0.44
0.42
activate.prop
2.5
5.0
7.5
10.0
decileScores

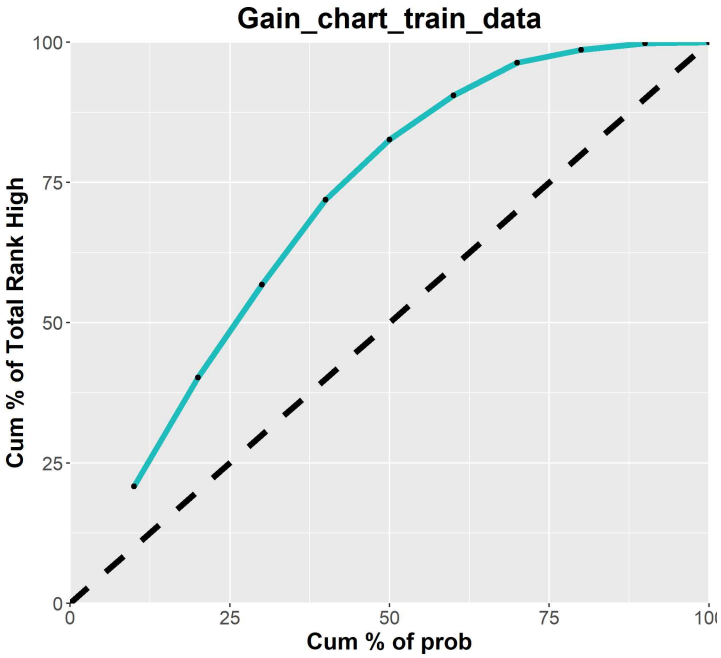

0.48
0.46
0.44
0.42
0.40
activate.prop
2.5
5.0
7.5
10.0
decileScores


0.6
0.4
0.2
activate.prop
2.5
5.0
7.5
10.0
decileScores

[참고 18] 최종 모델 roc curve



[참고 19] train 데이터 이익 도표



[참고 20] 원 데이터의 shapiro 정규성 검정 결과

```
import scipy.stats as stats
for i in range(21):
    test_stat,p_val=stats.shapiro(St[St.columns[i]])
    print("Test-statisctics: {},p-value:{}".format(test_stat,p_val))
```

```
Test-statisctics: 0.9612607955932617,p-value:3.1937545804496627e-29
Test-statisctics: 0.9468255043029785,p-value:2.2371001967811884e-33
Test-statisctics: nan,p-value:1.0
Test-statisctics: nan,p-value:1.0
Test-statisctics: nan,p-value:1.0
Test-statisctics: 0.9291172027587891,p-value:2.1260231251430974e-37
Test-statisctics: 0.6779911518096924,p-value:0.0
Test-statisctics: 0.9379782676696777,p-value:1.682439347474009e-35
Test-statisctics: 0.934127151966095,p-value:2.3767770775432375e-36
Test-statisctics: 0.5863298177719116,p-value:0.0
Test-statisctics: 0.7851278781890869,p-value:0.0
Test-statisctics: 0.9836727380752563,p-value:2.306079769136452e-19
Test-statisctics: 0.8869502544403076,p-value:1.961817850054744e-44
Test-statisctics: 0.9405912160873413,p-value:6.711138890993705e-35
Test-statisctics: 0.9168087840080261,p-value:9.623571346581757e-40
Test-statisctics: 0.9372852444648743,p-value:1.1747542141204392e-35
Test-statisctics: 0.8829360604286194,p-value:5.605193857299268e-45
Test-statisctics: 0.975207507610321,p-value:7.345996600580182e-24
Test-statisctics: 0.6129510402679443,p-value:0.0
Test-statisctics: 0.581068754196167,p-value:0.0
Test-statisctics: 0.9438270330429077,p-value:3.986465471380643e-34
```