

고려대학교 빅데이터 연구회

KU-BIG SPORT TEAM 최종발표

최홍석 정석원 임형준 박수희 김나연 고유경



1루

주제 선정과 데이터 수집

- 주제선정 배경
- 데이터 수집 과정

2루

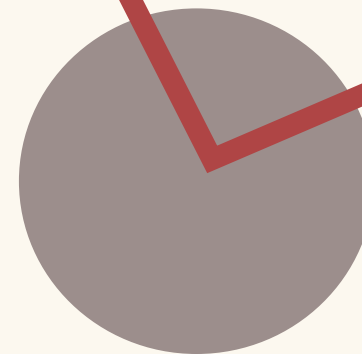
알고리즘과 분석 결과

- 마르코프체인을 통한 기대득점 산출
- 시뮬레이션을 통한 승패 예측

3루

한계 및 개선방안

- 무엇이 부족했는가
- 어떻게 개선할 수 있을까



1루 주제선정과 데이터수집
우리는 어떤 질문에 답하고자 하는가

“ 2019 한국 프로야구
경기결과(승패) 예측 ”



1루 주제선정과 데이터수집 우리는 어떤 데이터를 이용했는가



2018 시즌 팀별 데이터

3 김선빈
KIA, 유격수

종합 연도별 그래프 날짜별 상황별 상대별 Playlog 상세분석 연봉 스쿼트

타자 투수 2019 2018 2017 2016 2014 전체

상대: 검색 코마(,)로 구분 가능 날짜: 검색 (YYYY-MM-DD, 기간 입력시 시작-끝)

WPA 타석 주루 안타 2루타 3루타 홈런 타점 삼진 볼넷 사구 실책출루 희생타 병살 땅볼 뜬공

날짜	상대	이닝	투수	타자	P	결과	이전상황	이후상황	LEV	REa	WPs	WPe	WPa
2018-03-24	KT	2말	피어벤드	9 김선빈	0-1	유격수 땅볼	무사 0:2	1사 0:2	0.62	-0.258	75.0	73.4	-0.016
2018-03-24	KT	4말	피어벤드	9 김선빈	0-0	3루수 내야안타	무사 1루 1:2	무사 1,2루 1:2	1.33	0.620	72.0	76.9	0.048
2018-03-24	KT	6말	심재민	9 김선빈	1-3	볼넷	무사 1루 4:2	무사 1,2루 4:2	2.36	0.620	31.4	40.6	0.093
2018-03-24	KT	8말	이상화	9 김선빈	1-2	우익수 뜬공	무사 5:4	1사 5:4	2.48	-0.258	30.4	24.0	-0.063
2018-03-25	KT	1말	주권	9 김선빈	2-2	김선빈/10008: 삼진 아웃	2사 0:6	이닝종료 0:6	0.08	-0.117	93.3	93.0	-0.002
2018-03-25	KT	3말	주권	9 김선빈	2-1	2루수 뜬공	1사 1루 0:7	2사 1루 0:7	0.10	-0.322	97.6	97.4	-0.002
2018-03-25	KT	5말	류희운	9 김선빈	1-1	3루수 땅볼	2사 1루 1:7	이닝종료 1:7	0.07	-0.251	97.7	97.5	-0.002
2018-03-25	KT	7말	배우열	9 김선빈	1-3	볼넷	2사 1:10	2사 1루 1:10	0.00	0.134	99.9	99.9	0.000
2018-03-27	삼성	3말	보니아	9 김선빈	1-3	2루수 땅볼	무사 0:0	1사 0:0	0.99	-0.258	55.5	53.0	-0.025
2018-03-27	삼성	4말	보니아	9 김선빈	0-0	우익수 안타	1사 1루 0:4	1사 1,2루 0:4	0.35	0.398	91.5	92.5	0.010
2018-03-27	삼성	5말	김기태	9 김선빈	1-1	중견수 뜬공	무사 0:12	1사 0:12	0.01	-0.258	99.9	99.9	0.000
2018-03-27	삼성	7말	황수범	9 김선빈	2-3	볼넷	무사 0:14	무사 1루 0:14	0.00	0.398	100.0	100.0	0.000

야구 데이터 사이트 www.statiz.co.kr

2루 알고리즘과 데이터 분석결과
마르코프체인을 통한 기대득점 산출

마르코프체인

Markov Chain

확률변수의 상태변화에 대한 모형

: 어떤 상태(state)에 도달할 확률이

오직 바로 이전 시점의 상태(state)에 달려 있는 경우



2루 알고리즘과 데이터 분석결과

S_n : 광고가 나가고 n 주 후의 상태 벡터

$$S_0 = [0.2 \ 0.8]$$

$$S_1 = [0.2 \ 0.8] \begin{bmatrix} 0.9 & 0.1 \\ 0.7 & 0.3 \end{bmatrix} = [0.74 \ 0.26]$$

$$S_2 = [0.74 \ 0.26] \begin{bmatrix} 0.9 & 0.1 \\ 0.7 & 0.3 \end{bmatrix} = [0.848 \ 0.152]$$

$$S_3 = [0.848 \ 0.152] \begin{bmatrix} 0.9 & 0.1 \\ 0.7 & 0.3 \end{bmatrix} = [0.869 \ 0.131]$$

$$S_4 = [0.869 \ 0.131] \begin{bmatrix} 0.9 & 0.1 \\ 0.7 & 0.3 \end{bmatrix} = [0.874 \ 0.126]$$

...

$$S_{10} = S_9 P = [0.875 \ 0.125]$$

마르코프체인

Markov Chain

무려 20% -> 74%!!

확률변수의 상태변화에 대한 모형

74% -> 84.8%
: 어떤 상태(state)에 도달할 확률이

오직 바로 이전 시점의 상태(state)에 달려 있는 경우

즉, A의 시장점유율은 불변상태에 도달한다

= 마르코프 연쇄에 정상상태가 존재한다

큰 변화가 없다!

2루 알고리즘과 데이터 분석결과 마르코프체인을 통한 기대득점 산출

마르코프체인 Markov Chain

확률변수의 상태변화에 대한 모형

: 어떤 상태(state)에 도달할 확률이
오직 바로 이전 시점의 상태(state)에 달려 있는 경우

흡수상태

: 한번 들어가면 빠져 나오지 못하는 상태

흡수 마르코프 체인

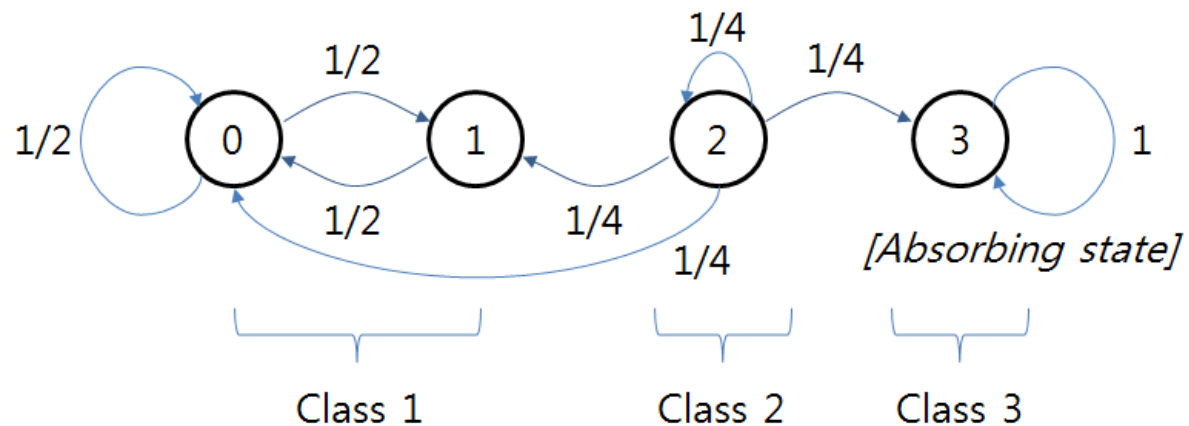
: 마르코프 체인이 하나 이상의 흡수 상태를 포함하고,
유한한 단계를 거쳐 비흡수 상태에서 흡수 상태로 갈 수 있는 것

2루 알고리즘과 데이터 분석결과

마르코프체인을 통한 기대득점 산출

 $P :=$ 상태(i)에서 상태(j)로 갈 확률 행렬, ($i, j = 0, 1, 2, 3$)

$$P = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



흡수상태

: 한번 들어가면 빠져 나오지 못하는 상태

➡ 상태3에 도달하는 순간, 빠져나올 수 없게 된다!

상태3 : **흡수상태**

흡수 마르코프 체인

: 마르코프 체인이 하나 이상의 흡수 상태를 포함하고,
유한한 단계를 거쳐 비흡수 상태에서 흡수 상태로 갈 수 있는 것

2루 알고리즘과 데이터 분석결과

마르코프체이은 토하 기대드전 사츠

Table 2.1. States of X_n

	000	100	010	001	110	101	011	111
no outs	1	2	3	4	5	6	7	8
one out	9	10	11	12	13	14	15	16
two outs	17	18	19	20	21	22	23	24
three outs	25							

상태공간 : $S = \{1, 2, \dots, 25\}$

오직 바로 이전 시점의 상태(state)에 달려 있는 경우

$\Rightarrow X = \{X_n, n = 1, 2, \dots\}$ ^{흡수상태}은 상태공간이 S 인 마르코프 체인
: 한 번 들면 그 상태에 더 이상 오지 못하는 상태

3-out이 되면 이닝이 종료하므로, **상태25는 흡수상태!**

흡수 마르코프 체인

: 마르코프 체인이 하나 이상의 흡수 상태를 포함하고,
유한한 단계를 거쳐 비흡수 상태에서 흡수 상태로 갈 수 있는 것

2루 알고리즘과 데이터 분석결과 마르코프체인을 통한 기대득점 산출

마르코프체인 Markov Chain

● 전이 행렬: 진루 행렬

$$P_{ij} = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,25} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ P_{25,1} & \cdots & P_{25,25} \end{bmatrix}$$

P_{ij} : 출루 상황이 i에서 j로 바뀔 확률

● 기대득점 행렬

$$E_{ij} = \begin{bmatrix} E_{1,1} & E_{1,2} & \cdots & E_{1,25} \\ E_{2,1} & E_{2,2} & \cdots & E_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ E_{25,1} & \cdots & E_{25,25} \end{bmatrix}$$

E_{ij} : 출루 상황이 i에서 j로 바뀔 때 기대 득점 평균

2루 알고리즘과 데이터 분석결과 마르코프체인을 통한 기대득점 산출

마르코프체인 Markov Chain

전이 행렬: 진루 행렬

$$P_{ij} = \frac{\begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,25} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ P_{25,1} & P_{25,2} & \cdots & P_{25,25} \end{bmatrix}}{\text{각 } i \text{의 전체 빈도수}}$$

[진루행렬의 각 확률]
i->j에 해당하는 모든 빈도수

P_{ij} : 출루 상황이 i에서 j로 바뀔 확률

기대득점 행렬

[기대득점행렬 예시]

진루상황변화: 001->000

가능한 득점	예상 가능 시나리오
0점	둘 다 아웃
1점	타자 아웃 & 3루 주자 홈으로
2점	홈런

E_{ij} : 출루

2루 알고리즘과 데이터 분석결과 마르코프체인을 통한 기대득점 산출

마르코프체인 Markov Chain

● 전이 행렬: 진루 행렬

$$P_{ij} = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,25} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ P_{25,1} & \cdots & P_{25,25} \end{bmatrix}$$

P_{ij} : 출루 상황이 i에서 j로 바뀔 확률

i->j에 해당하는 모든 빈도수/각 i의 전체 빈도수

● 기대득점 행렬

$$E_{ij} = \begin{bmatrix} E_{1,1} & E_{1,2} & \cdots & E_{1,25} \\ E_{2,1} & E_{2,2} & \cdots & E_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ E_{25,1} & \cdots & E_{25,25} \end{bmatrix}$$

E_{ij} : 출루 상황이 i에서 j로 바뀔 때 기대 득점 평균

$$E_{I.} = \sum_{j=1}^{25} E_{Ij} P_{Ij} : I \text{ 상태에서 가능한 기대 득점 평균 (後 모든 상태 고려)}$$

2루 알고리즘과 데이터 분석결과 마르코프체인을 통한 기대득점 산출

마르코프체인 Markov Chain

> State = $(\pi_1, \pi_2, \pi_3, \dots, \pi_{25})$

> $E_i = (E_{i,1}, E_{i,2}, E_{i,3}, \dots, E_{i,25})$

$$> E = \sum_{i=1}^{25} E_i \pi_i$$

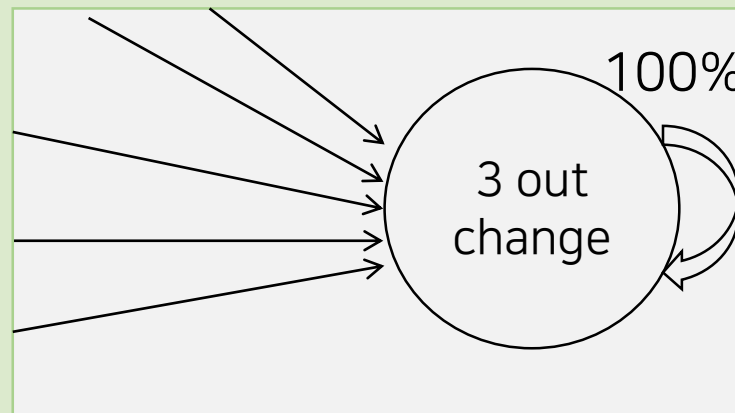
> 모든 타자 들의 E 합 = 해당 경기의 총 기대 득점

* 한 이닝에 대한 정의

흡수상태 π_{25}

cutoff value 임의 지정 가능

$\pi_{25} > \text{cutoff}$: 이닝 종료



2루 알고리즘과 데이터 분석결과 마르코프체인을 통한 기대득점 산출

기아 타이거즈

#타순입력

lineup=c("김선빈","버나디나","김주찬","최형우","이범호","안치홍","나지완","이명기","김민식")

```
> lineup_matrix
, , 1
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]	[,14]	[,15]	[,16]	[,17]	[,18]	[,19]	[,20]	[,21]	[,22]	[,23]	[,24]	[,25]
[1,]	0	0.3017	0.0345	0	0.0000	0.000	0	0.0000	0.6638	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[2,]	0	0.0000	0.0000	0	0.3250	0.075	0	0.0000	0.0000	0.3500	0.1500	0.0000	0.0000	0.0000	0.0000	0.0000	0.1000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[3,]	0	0.0000	0.0000	0	0.1000	0.400	0	0.0000	0.0000	0.0000	0.3000	0.2000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[4,]	0	0.1667	0.0000	0	0.0000	0.500	0	0.0000	0.1667	0.0000	0.0000	0.1667	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[5,]	0	0.0000	0.0000	0	0.1429	0.000	0	0.2857	0.0000	0.0000	0.0000	0.0000	0.1429	0.0000	0.2857	0.0000	0.0000	0.0000	0.0000	0.1429	0.0000	0.0000	0.0000	0.0000	0.0000
[6,]	0	0.0000	0.0000	0	1.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[7,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[8,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[9,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0109	0.2717	0.0978	0.0109	0.0000	0.0000	0.0000	0.6087	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[10,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.1111	0.0000	0.0000	0.0000	0.3333	0.0556	0.0000	0.0000	0.1667	0.2222	0.0556	0.0000	0.0000	0.0000	0.0000	0.0000	0.0556
[11,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0556	0.0000	0.0000	0.1667	0.1667	0.0000	0.0000	0.0000	0.0000	0.2778	0.2222	0.0000	0.0000	0.0000	0.0000	0.1111
[12,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.2500	0.0000	0.0000	0.0000	0.0000	0.0000	0.5000	0.0000	0.0000	0.2500	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
[13,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0714	0.0000	0.0000	0.0000	0.0000	0.0000	0.2143	0.0000	0.0000	0.0000	0.0000	0.3571	0.0714	0.0714	0.0000	0.0000	0.2143
[14,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.2000	0.0000	0.0000	0.2000	0.0000	0.2000	0.0000	0.0000	0.0000	0.2000	0.0000	0.0000	0.2000
[15,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2000	0.0000	0.0000	0.2000	0.0000	0.0000	0.4000	0.2000	0.0000	0.0000
[16,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2000	0.0000	0.0000	0.0000	0.2000	0.2000	0.0000	0.0000	0.0000	0.4000
[17,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.3833	0.0167	0.0000	0.0000	0.0000	0.0000	0.0000	0.6000
[18,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.3333	0.0833	0.0000	0.0000	0.0000	0.5833
[19,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0714	0.1429	0.0000	0.0714	0.0000	0.0000	0.0000	0.7143
[20,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1000	0.0000	0.0000	0.0000	0.2000	0.0000	0.0000	0.7000
[21,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0625	0.0000	0.0000	0.0000	0.1250	0.0000	0.8125
[22,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0769	0.4615	0.0000	0.0000	0.0000	0.4615
[23,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5000
[24,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5000
[25,]	0	0.0000	0.0000	0	0.0000	0.000	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5000

진루행렬(25*25*9) - 9명의 타자에 대한 3차원 매트릭스

```
> lineup_score_matrix
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]	[,14]	[,15]	[,16]	[,17]	[,18]	[,19]	[,20]	[,21]	[,22]	[,23]	[,24]	[,25]
[1,]	0.0000	0.0000	0.0000	0.3333	0.1429	1.0000	0.0000	0.0000	0.0109	0.2222	0.0556	0.7500	0.2143	0.4000	0.2000	0.6000	0.0000	0.0000	0.2857	0.1000	0.1250	0.6154	0.5000	1.0000	0
[2,]	0.0538	0.1000	0.2500	0.0000	0.2000	1.0000	1.0000	0.5000	0.0411	0.0612	0.2222	0.7273	0.3529	0.7143	1.0000	1.5000	0.0370	0.1212	0.2692	0.0000	0.6500	0.7143	1.4286	0.5455	0
[3,]	0.0309	0.1379	0.1250	0.0000	0.1250	0.5556	0.0000	0.7500	0.0256	0.0417	0.2500	0.3333	0.1111	1.2000	0.9167	1.1429	0.0500	0.1250	0.4737	0.6667	0.5000	0.2500	0.6667	1.2000	0
[4,]	0.0465	0.1000	0.3333	0.0000	0.1667	0.7500	0.7500	2.0000	0.0857	0.0625	0.2500	0.5000	0.3529	0.5882	0.3750	0.9000	0.0471	0.0278	0.3548	0.0667	0.5714	0.3333	0.8333	2.3333	0
[5,]	0.0667	0.2222	0.0000	0.0000	0.8000	0.0000	0.0000	2.0000	0.0328	0.2800	0.0000	0.2000	0.3077	1.5000	1.0000	1.2000	0.0222	0.0690	0.2500	0.5714	0.2727	0.2222	0.0000	1.0000	0
[6,]	0.0455	0.2069	0.0000	1.5000	0.7692	1.0000	1.5000	1.2857	0.0405	0.0714	0.2000	0.3333	0.5185	0.7143	1.5000	1.2500	0.0000	0.0513	0.2381	0.5714	0.5500	0.7143	0.5000	0.1667	0
[7,]	0.0548	0.2143	0.1818	0.0000	0.2500	0.6667	0.0000	1.0000	0.0317	0.2258	0.2222	0.7500	0.3529	0.7143	1.0000	1.5000	0.0370	0.1212	0.2692	0.0000	0.6500	0.7143	1.4286	0.5455	0
[8,]	0.0000	0.0256	0.1429	0.0000	0.1111	0.0000	0.3333	0.0000	0.0270	0.1071	0.1500	0.7500	0.3529	0.7143	1.0000	1.5000	0.0370	0.1212	0.2692	0.0000	0.6500	0.7143	1.4286	0.5455	0
[9,]	0.0426	0.0000	0.0000	0.0000	0.0000	1.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.7500	0.3529	0.7143	1.0000	1.5000	0.0370	0.1212	0.2692	0.0000	0.6500	0.7143	1.4286	0.5455	0

득점행렬(9*25) - 9명의 타자에 대한 2차원 매트릭스

2루 알고리즘과 데이터 분석결과 마르코프체인을 통한 기대득점 산출

```
#1이닝 시작
score_for_inning(lineup_matrix,lineup_score_matrix,1)
n=find_next_batter(lineup_matrix,lineup_score_matrix,1)
total_score=score_for_inning(lineup_matrix,lineup_score_matrix,1)

#2이닝 시작
score_for_inning(lineup_matrix,lineup_score_matrix,n)
n=find_next_batter(lineup_matrix,lineup_score_matrix,n)
total_score=total_score+score_for_inning(lineup_matrix,lineup_score_matrix,n)

#3이닝 시작
score_for_inning(lineup_matrix,lineup_score_matrix,n)
n=find_next_batter(lineup_matrix,lineup_score_matrix,n)
total_score=total_score+score_for_inning(lineup_matrix,lineup_score_matrix,n)

#4이닝 시작
score_for_inning(lineup_matrix,lineup_score_matrix,1)
n=find_next_batter(lineup_matrix,lineup_score_matrix,1)
total_score=total_score+score_for_inning(lineup_matrix,lineup_score_matrix,n)

#5이닝 시작
score_for_inning(lineup_matrix,lineup_score_matrix,n)
n=find_next_batter(lineup_matrix,lineup_score_matrix,n)
total_score=total_score+score_for_inning(lineup_matrix,lineup_score_matrix,n)

#6이닝 시작
score_for_inning(lineup_matrix,lineup_score_matrix,n)
n=find_next_batter(lineup_matrix,lineup_score_matrix,n)
total_score=total_score+score_for_inning(lineup_matrix,lineup_score_matrix,n)
```



기아 타이거즈
시즌 평균 기대득점
5.904733

```
> #최종점수
> total_score
      [,1]
[1,] 5.904733
```


“

매 경기결과를
예측할 방법은?

최o석(통계학과, 26세)

”

“

투수의 영향을
반영할 방법은 없을까?

임o준(통계학과, 26세)

”



2루 알고리즘과 데이터 분석결과 시뮬레이션을 통한 승패 예측

타자의 진루행렬과 기대득점 행렬이 있다면

● 전이 행렬: 진루 행렬

$$P_{ij} = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,25} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ P_{25,1} & \cdots & P_{25,25} \end{bmatrix}$$

● 기대득점 행렬

$$E_{ij} = \begin{bmatrix} E_{1,1} & E_{1,2} & \cdots & E_{1,25} \\ E_{2,1} & E_{2,2} & \cdots & E_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ E_{25,1} & \cdots & E_{25,25} \end{bmatrix}$$

상대팀 투수 의 진루행렬과
기대실점 행렬도 존재!

● 전이 행렬: 진루 행렬

$$P_{ij} = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,25} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ P_{25,1} & \cdots & P_{25,25} \end{bmatrix}$$

● 기대실점 행렬

$$E_{ij} = \begin{bmatrix} E_{1,1} & E_{1,2} & \cdots & E_{1,25} \\ E_{2,1} & E_{2,2} & \cdots & E_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ E_{25,1} & \cdots & E_{25,25} \end{bmatrix}$$

2루 알고리즘과 데이터 분석결과 시뮬레이션을 통한 승패 예측

타자의 진루행렬과 기대득점 행렬이 있다면

$$P_{ij} = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,25} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ P_{25,1} & \cdots & P_{25,25} \end{bmatrix}$$

타자 진루행렬과 투수 진루행렬의 평균

상대팀 투수 의 진루행렬과
기대실점 행렬도 존재!

$$E_{ij} = \begin{bmatrix} E_{1,1} & E_{1,2} & \cdots & E_{1,25} \\ E_{2,1} & E_{2,2} & \cdots & E_{2,25} \\ \vdots & \vdots & \ddots & \vdots \\ E_{25,1} & \cdots & E_{25,25} \end{bmatrix}$$

$$E.i = (E_{.1}, E_{.2}, E_{.3}, \cdots, E_{.25})$$

타자 기대득점행렬과 투수 기대실점 행렬의 평균

2루 알고리즘과 데이터 분석결과 시뮬레이션을 통한 승패 예측



선발투수: 박정배



선발투수: 진해수

2루 알고리즘과 데이터 분석결과 시뮬레이션을 통한 승패 예측



- lg 선발타자 9명의 진루행렬과 sk 선발투수 박정배 선수의 진루행렬
- lg 선발타자 9명의 기대득점 행렬과 sk 선발투수 박정배 선수의 기대실점 행렬

새로운 진루행렬

새로운 득점행렬

```

batter.matrix<-lg.bat2[[1]] #선수 진루 행렬
batter.score.matrix<-lg.bat2[[2]] #선수 득점 행렬
pitcher.matrix=sk.pit[[1]] #투수 진루 행렬
pitcher.score.matrix=sk.pit[[2]] #투수 실점 행렬
i=1
bat.pit.matrix=array(0,c(25,25,9))
for(i in 1:9){
  bat.pit.matrix[,i]=(batter.matrix[,i]+pitcher.matrix[,1])/2
  for(j in 1:25){
    if(mean(batter.matrix[j,,i])==0) {bat.pit.matrix[j,,i]<-pitcher.matrix[j,,1]}
    if(mean(pitcher.matrix[j,,1])==0) {bat.pit.matrix[j,,i]<-batter.matrix[j,,i]}
  }
}
bat.pit.score.matrix=matrix(0,9,25) #투수의 실점력과 타자의 득점율을 평균낸 것
for(i in 1:9){
  bat.pit.score.matrix[i,]=(batter.score.matrix[i,]+pitcher.score.matrix[1,])/2
}
  
```

2루 알고리즘과 데이터 분석결과 시뮬레이션을 통한 승패 예측



```
> bat.pit.matrix
, , 1
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]	[,14]	[,15]	[,16]	[,17]	[,18]	[,19]	[,20]	[,21]	[,22]	[,23]	[,24]	[,25]
[1,]	0.0540	0.21965	0.09385	0	0.00000	0.00000	0.00000	0.00000	0.63250	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[2,]	0.0000	0.00000	0.07405	0	0.16665	0.03705	0.03705	0.00000	0.00000	0.37035	0.05555	0.00000	0.00000	0.00000	0.00000	0.00000	0.25925	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[3,]	0.3750	0.06250	0.00000	0	0.18750	0.00000	0.00000	0.00000	0.00000	0.00000	0.18750	0.18750	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[4,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.50000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[5,]	0.0294	0.00000	0.05880	0	0.02940	0.02940	0.00000	0.16665	0.00000	0.00000	0.00000	0.00000	0.56865	0.05880	0.00000	0.00000	0.00000	0.00000	0.00000	0.05880	0.00000	0.00000	0.00000	0.00000	0.00000
[6,]	0.0000	0.00000	0.00000	0	0.07145	0.07145	0.00000	0.07145	0.00000	0.00000	0.00000	0.00000	0.64285	0.00000	0.00000	0.00000	0.14285	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[7,]	0.5000	0.12500	0.00000	0	0.00000	0.00000	0.00000	0.12500	0.00000	0.00000	0.00000	0.25000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[8,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.75000	0.00000	0.00000	0.00000	0.00000	0.00000	0.25000	0.00000	0.00000	0.00000
[9,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.02065	0.23950	0.05320	0.00000	0.00000	0.00000	0.00000	0.68665	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[10,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.02000	0.04165	0.20645	0.06155	0.03845	0.00000	0.39615	0.17115	0.01540	0.00000	0.00000	0.00000	0.00000	0.05385	0.00000
[11,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.02000	0.00000	0.20500	0.06000	0.00000	0.00000	0.00000	0.43000	0.28500	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[12,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.25000	0.00000	0.00000	0.75000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[13,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.02630	0.00000	0.02630	0.00000	0.02630	0.13600	0.08335	0.08335	0.00000	0.00000	0.00000	0.16665	0.29390	0.16665	0.02630	0.00000	0.13160
[14,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.25000	0.11110	0.25000	0.05555	0.00000	0.16665	0.05555	0.00000	0.11110	0.00000	0.00000	0.00000	0.00000
[15,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.41665	0.00000	0.00000	0.00000	0.25000	0.00000	0.00000	0.00000	0.16665	0.00000	0.16665	0.00000	0.00000	0.00000
[16,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.16665	0.00000	0.00000	0.00000	0.00000	0.10000	0.20000	0.00000	0.00000	0.00000	0.10000	0.16665	0.16665	0.10000	0.00000	0.00000	0.00000
[17,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.02365	0.23480	0.06040	0.00440	0.00000	0.00000	0.00000	0.00000	0.67680	0.00000
[18,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.05555	0.00000	0.01725	0.00000	0.09005	0.03450	0.00000	0.00000	0.80270	0.00000
[19,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.14090	0.07045	0.00000	0.14545	0.00000	0.00000	0.00000	0.64320	0.00000
[20,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.26665	0.08335	0.00000	0.08335	0.00000	0.00000	0.00000	0.56665	0.00000
[21,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.11110	0.07145	0.12700	0.69050	0.00000	0.00000
[22,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.05555	0.00000	0.00000	0.05555	0.55555	0.00000	0.05555	0.00000	0.27780	0.00000
[23,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.41665	0.00000	0.16665	0.16665	0.00000	0.41665	0.00000	0.41665	0.00000
[24,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
[25,]	0.0000	0.00000	0.00000	0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

진루행렬(25*25*9) - 투수와 9명의 타자에 대한 3차원 매트릭스

```
> bat.pit.score.matrix
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]	[,14]	[,15]	[,16]	[,17]	[,18]	[,19]	[,20]	[,21]	[,22]	[,23]	[,24]	[,25]
[1,]	0.05400	0.07405	0.81250	0.25000	0.26470	0.28570	2.00000	0.00000	0.02065	0.07240	0.02000	0.25000	0.37720	0.83335	1.00000	1.500	0.02365	0.12835	0.21135	0.35000	0.18255	0.77780	1.00000	0.41665	0
[2,]	0.04960	0.07830	0.68750	0.25000	0.00000	0.66665	1.50000	0.00000	0.02145	0.13255	0.07690	0.25000	0.34520	0.66665	0.50000	1.000	0.05335	0.11110	0.21325	0.22500	0.10480	0.50000	0.50000	0.58335	0
[3,]	0.04705	0.09965	0.61110	0.50000	0.07145	0.50000	1.83335	0.75000	0.04035	0.09720	0.13160	0.37500	0.30950	0.87500	0.58335	1.375	0.03900	0.15990	0.29020	0.32220	0.48810	0.92855	0.75000	1.75000	0
[4,]	0.04035	0.16195	0.54165	0.33335	0.10000	0.50000	2.00000	0.50000	0.05220	0.08450	0.10715	0.37500	0.54165	0.91665	1.00000	0.900	0.03260	0.18085	0.32865	0.21110	0.27515	0.58825	0.95455	0.62500	0
[5,]	0.04395	0.10100	0.60000	0.00000	0.00000	0.33335	1.50000	0.00000	0.02395	0.04165	0.00000	0.00000	0.41665	0.50000	0.83335	1.000	0.01925	0.11110	0.18180	0.43335	0.23810	0.50000	0.50000	0.58335	0
[6,]	0.04555	0.07220	0.60000	0.00000	0.08335	0.25000	1.87500	0.33335	0.04055	0.04165	0.16665	0.25000	0.37120	0.79165	1.00000	1.000	0.02850	0.14050	0.26795	0.18335	0.21035	0.57145	0.75000	0.91665	0
[7,]	0.03190	0.05555	0.60525	0.50000	0.00000	0.33335	2.00000	0.00000	0.02155	0.05300	0.12500	0.50000	0.61665	1.00000	1.00000	1.000	0.03465	0.11110	0.38635	0.21110	0.32145	0.60000	0.70000	0.37500	0
[8,]	0.05340	0.06985	0.57145	0.00000	0.05555	0.33335	1.50000	1.50000	0.03585	0.07195	0.11110	0.35715	0.30950	0.50000	1.00000	0.750	0.03565	0.20410	0.38635	0.32220	0.09525	0.62500	0.50000	0.75000	0
[9,]	0.05320	0.05555	0.66665	0.00000	0.00000	0.33335	1.50000	0.00000	0.01470	0.04165	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0

득점행렬(9*25) - 투수와 9명의 타자에 대한 2차원 매트릭스

2루 알고리즘과 데이터 분석결과 시뮬레이션을 통한 승패 예측



```
> sim[[1]] #해당 이닝
[1] 1
> sim[[2]] #해당 회에 낼 점수
[1] 0.1713
> sim[[3]] #누적 점수
[1] 0.1713
> sim[[4]] #다음 타자 순서
[1] 4
```

 1이닝

```
> #3회 시작
> sim=simulation(bat.pit.matrix,bat.pit.score.matrix,n,last_score,inning)
> n=sim[[4]] ; last_score=sim[[3]] ; inning=sim[[1]]
> sim[[1]] #해당 이닝
[1] 3
> sim[[2]] #해당 회에 낼 점수
[1] 0.89215
> sim[[3]] #누적 점수
[1] 1.15625
> sim[[4]] #다음 타자 순서
[1] 3
```

 3이닝

```
> #2회 시작
> sim=simulation(bat.pit.matrix,bat.pit.score.matrix,n,last_score,inning)
> n=sim[[4]] ; last_score=sim[[3]] ; inning=sim[[1]]
> sim[[1]] #해당 이닝
[1] 2
> sim[[2]] #해당 회에 낼 점수
[1] 0.0928
> sim[[3]] #누적 점수
[1] 0.2641
> sim[[4]] #다음 타자 순서
[1] 7
```

 2이닝

```
> #4회 시작
> sim=simulation(bat.pit.matrix,bat.pit.score.matrix,n,last_score,inning)
> n=sim[[4]] ; last_score=sim[[3]] ; inning=sim[[1]]
> sim[[1]] #해당 이닝
[1] 4
> sim[[2]] #해당 회에 낼 점수
[1] 0.259
> sim[[3]] #누적 점수
[1] 1.41525
> sim[[4]] #다음 타자 순서
[1] 7
```

 4이닝

2루 알고리즘과 데이터 분석결과 시뮬레이션을 통한 승패 예측



```
> score_sk
[1] 6.03730 5.84580 3.35425 6.69795 4.80505 4.58025 5.89210 5.26635 2.97040 4.96510 2.58775 2.99640 12.80335 3.35015 8.02465 3.28635
> score_lg
[1] 2.48115 8.61175 4.82640 5.30635 7.11955 7.08090 7.10265 8.84730 8.44015 7.28490 9.52715 5.26290 2.50515 6.69480 2.72500 4.84415
> sum(score_sk < score_lg)
[1] 12
> sum(score_sk > score_lg)
[1] 4
```

1경기를 16번 반복측정,
이긴 횟수 많은 팀이 해당경기 이길 것이라 예측
lg트윈스(12번 승리) vs sk와이번스(4번승리)

승패 예측:  승

2루 알고리즘과 데이터 분석결과 시뮬레이션을 통한 승패 예측



```
> score_sk
[1] 6.03730 5.84580 3.35425 6.69795 4.80505 4.58025 5.89210 5.26635 2.97040 4.96510 2.58775 2.99640 12.80335 3.35015 8.02465 3.28635
> score_lg
[1] 2.48115 8.61175 4.82640 5.30635 7.11955 7.08090 7.10265 8.84730 8.44015 7.28490 9.52715 5.26290 2.50515 6.69480 2.72500 4.84415
> sum(score_sk<score_lg)
[1] 12
> sum(score_sk>score_lg)
[1] 4
```



LG 트윈스
패 진해수

1

경기종료

2

SK 와이번스
승 박정배



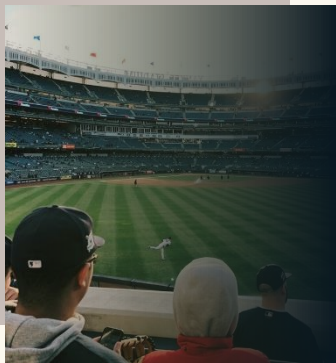
경기기록

하이라이트

2019.03.27.(수) 18:30 | 문학 | SPOTV2 | 네이버스포츠

승패 예측:  승

3루 한계 및 개선방안 무엇이 부족했는가



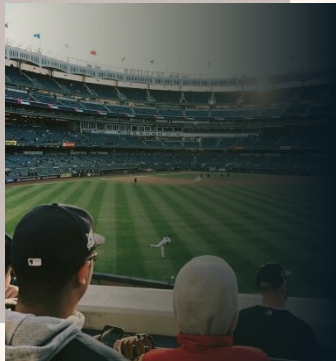
01 2019 시즌 데이터를 활용하지 못했어요.

2018시즌 데이터를 활용하여 19년도의 경기결과를 예측하고자 했으나, 본디 야구라는 것은 전년도에 잘해도 다음해에 충분히 못할 가능성이 농후한 스포츠이기에 이것만으로는 정확한 예측이 불가능해요.

02 총 득점의 결과가 소수점으로 떨어져요.

득점은 언제나 정수꼴로 나타나야함에도 불구하고 분석을 하다보니 소수꼴의 기대득점이 쌓이고 쌓여 결국 누적득점(경기 총 기대득점)도 소수점으로 떨어졌어요. 기대득점이 평균으로 회귀하는 느낌이 들어요.

3루 한계 및 개선방안 어떻게 개선할 수 있을까



01 2018과 2019 시즌 데이터를 합칠 방안을 찾자

2019년도 데이터가 부족해서 이것만으로는 원하는 진루행렬과 득점행렬을 만들지 못해요. 따라서 2019년 데이터를 토대로 행렬을 만들기에 부족한 데이터는 2018년 데이터를 가져와서 쓰면 될거 같아요.

02 시뮬레이션 + 득점별 샘플링

각 진루상황 $i \rightarrow j$ 에 대해 득점분포를 가지고 샘플링을 하는게 좋을거 같아요.

3루 한계 및 개선방안 어떻게 개선할 수 있을까

ex) $i=5$ $j=10$ 이라고 하면

0득점 = 100번

1득점 = 20번

2득점 = 10번

3득점 = 0번

4득점 = 0번

01 2018년 2019시즌 데이터를 합칠 방안을 찾자

2019년도 데이터가 부족해서 이것만으로는 원하는 진루행렬과 득점행렬을 만들지 못해요. 따라서 2019년 데이터를 토대로 행렬을 만들기에 부족한 데이터는 2018년 데이터를 가져와서 쓰는 게 좋을 거 같아요.

기존에 쓴 방법:

$$(0 \cdot 100 + 1 \cdot 20 + 2 \cdot 10) / 130 = 0.3077 \text{ 득점}$$

100/130의 확률로 0득점을 하고, 20/130의 확률로 1득점,
10/130의 확률로 2득점을 하는 sampling을 하자.


즉, $i \rightarrow j$ 로 넘어갈 때 기대득점은 0.3077이 아니라
가장 큰 상한 $i > 10$ 에 대해 득점분포를 가지고 샘플링을 하는 게 좋을 거 같아요.

0, 1, 2 중 하나의 득점을 하는 것.

이 때 0, 1, 2는

`sampel(c(0,1,2,3,4), 1, c(100/130, 20/130, 10/130, 0, 0))`

함수를 통해 sample 한다.

A large crowd at a baseball stadium, with many hands raised in the foreground, suggesting a celebratory or enthusiastic atmosphere. The stadium is filled with spectators, and the field is visible in the background.

Q&A

Thank you for listening