2021 빅콘테스트 스포츠 테크 분야

: 프로야구 배럴(Barrel) 을 통한 타자 성적 예측

팀명 : 늘보
팀장 : 홍진우 rw2006@naver.com
팀원 : 박종서 qkrwhdtj208@naver.com

<목차>

01 문제정의

sports2i
Information & Interest Provider in Sports

스포츠투아이에서 제공하는 야구데이터(타자 기본정보, 타자 트래킹 데이터 등)를 활용하여

좋은 타구(배럴)에 대하여 정의하고,

타자 성적 예측 모형 개발을 통한 **타자의 OPS(장타율+출루율)** 예측

02 데이터개요

제9회 데이터 분석 경진대회 2021 빅콘테스트

	GYEAR	PCODE	GAMENUM	타석	타수	타율	안타	홈런	루타	장타율	희생플라이	볼넷	삼진	고의4구	사구	병살타
0	2018	60100	70	169	152	0.243	37	3	63	0.414	1	12	36	0	4	5
1	2018	60184	6	10	10	0.200	2	0	3	0.300	0	0	3	0	0	0
2	2018	60288	1	1	1	0.000	0	0	0	0.000	0	0	0	0	0	0
3	2018	60343	83	174	162	0.216	35	8	63	0.389	0	8	49	0	3	3
4	2018	60456	4	8	6	0.500	3	0	3	0.500	0	2	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1093	2021	79290	4	8	8	0.125	1	0	1	0.125	0	0	2	0	0	0
1094	2021	79365	73	261	219	0.265	58	16	117	0.534	1	32	69	0	6	5
1095	2021	79402	70	278	237	0.211	50	1	61	0.257	0	30	26			
1096	2021	79456	62	135	124	0.274	34	0	38	0.306	2	7	11			
1097	2021	79608	55	246	212	0.325	69	5	102	0.481	5	26	32			

1098 rows × 16 columns

2018, 2019, 2020, 2021 시즌

타자 일반 데이터 총 1098개

2018, 2019, 2020, 2021 시즌

타구 트래킹 데이터 총 120,745개

	GYEAR	G_ID	PIT_ID	PCODE	T_ID	INN	타구속도	발사각도	HIT_RESULT	투구구속	STADIUM
0	2018	20180324HHWO0	180324_140436	62797	HH	1	131.50	42.7	플라이	149.59	고척
1	2018	20180324HHWO0	180324_140514	76753	HH	1	135.18	9.9	1루타	148.78	고척
2	2018	20180324HHWO0	180324_140647	71752	HH	1	152.41	2.1	1루타	148.59	고척
3	2018	20180324HHWO0	180324_140911	62700	HH	1	113.72	13.5	1루타	139.13	고척
4	2018	20180324HHWO0	180324_142050	68730	HH	2	54.11	16.4	번트안타	134.07	고척
...	...	...	...	...	...	...	...	...	...	...	...
17959	2021	20210711LTSS0	210711_195321	50458	SS	8	141.23	-4.9	땅볼아웃	137.76	대구
17960	2021	20210711LTSS0	210711_195424	62415	SS	8	101.09	31.8	1루타	138.51	대구
17961	2021	20210711LTSS0	210711_195627	75566	SS	8	146.84	33.4	홈런	134.10	대구
17962	2021	20210711LTSS0	210711_200047	69418	SS	8	146.12	23.9	홈런	139.96	대구
17963	2021	20210711LTSS0	210711_200245	64793	SS	8	140.45	28.2	플라이	137.36	대구

120745 rows × 11 columns

03 분석배경 및 목표 - 분석 배경

다승 순위		평균자책 순위		탈삼진 순위		세이브 순위	
1 요키시 키움	13승	1 미란다 두산	2.33	1 미란다 두산	164	1 오승환 삼성	31
2 루친스키 NC	12승	2 수아레즈 LG	2.46	2 폰트 SSG	139	2 김재윤 KT	25
2 미란다 두산	12승	3 요키시 키움	2.50	3 카펜터 한화	137	3 고우석 LG	24
2 원태인 삼성	12승	4 백정현 삼성	2.63	4 데스파이네 KT	128	3 김원중 롯데	24
5 뷰캐넌 삼성	11승	5 원태인 삼성	2.69	5 루친스키 NC	126	5 정해영 KIA	19

타율 순위		타점 순위		홈런 순위		도루 순위	
1 강백호 KT	0.374	1 강백호 KT	86	1 나성범 NC	28	1 김혜성 키움	33
2 이정후 키움	0.361	1 피렐라 삼성	86	2 최정 SSG	27	1 박해민 삼성	33
3 양의지 NC	0.336	3 양의지 NC	85	3 피렐라 삼성	25	3 최원준 KIA	25
4 박건우 두산	0.331	4 김재환 두산	79	4 양의지 NC	23	3 구자욱 삼성	25
5 홍창기 LG	0.325	5 나성범 NC	78	4 양석환 두산	23	5 김지찬 삼성	19

클래식 스탯 (투수의 경우 낮은 평균 자책점, 승리, 세이브, 타자의 경우 타율, 홈런, 타점 등)**에 대한 평가 위주의 한국프로야구**

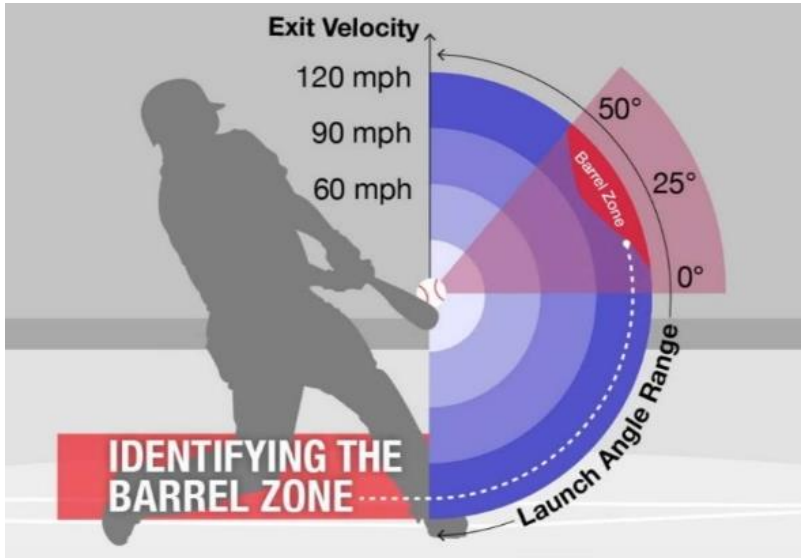
최근 야구팬들의 **세이버 매트릭스**에 대한 관심 증가에 따른 추세 변화

최근 2, 3년 간 구장 전광판에 팀 타자의 타율 대신 **OPS**를 표기하는 구단이 생겨났고, 방송사는 중계에서 **WAR, WRC+ 등의 지표**를 적극적으로 사용 중

03 분석배경 및 목표 - 분석 목표

지난 5-6년 간 강한 타구에 대한 주목도가 높아진 메이저리그

➔ 강하고 좋은 타구를 만들어 내는 타자, 그런 타구를 효과적으로 억제할 수 있는 투수를 훌륭한 선수로 평가



<MLB의 배럴타구 기준>

: 타구 속도와 발사각도의 조합상 평균적으로 타율 .500, 장타율 1.500이상을 생산하는 타구들

➔ 배럴 타구가 되기 위해서는 최소 시속 98마일을 기록해야 하며, 해당 속도에서는 발사각이 26°~30°가 돼야 한다. 시속 98마일을 넘어가는 타구들에 대해서는 그 발사각의 범주가 커짐.

➔ 시속 100마일부터 시속 116마일까지는 시속 1마일이 증가할 때마다 배럴 타구가 되는 발사각의 범위가 2°~3° 증가하며, 시속 116마일짜리 타구에서는 발사각이 8°~50°사이이기만 하면 배럴 타구

즉, 타자의 좋은 타구를 만들어 내는 능력을 타자의 고유 능력으로 상정

배럴 타구 기준 선정

: MLB와 KBO는 평균 투구 속도, 평균 타구 속도에서 차이를 보이므로 한국 프로야구 만의 배럴 타구 기준을 타구 데이터 분석을 통해 선정

➔ **결과 해석** : 공인구의 차이와 같은 외부 환경을 제외한다고 하더라도, KBO의 선수들과 MLB선수들 사이에 기본적인 **신체적 능력의 차이**가 있는 것으로 보인다. 이는 두 리그 내에서 만들어지는 **인플레이 타구의 환경**이 확연히 다르다는 것을 보여준다. 또한 **수비범위와 송구력의 차이**에 따라 같은 타구에 있어서도 타율, 장타율이 상이하게 나타날 수 있다. 따라서 MLB의 배럴타구의 기준을 그대로 가져와서 KBO의 타자들에게 적용시키는 것이 아닌, **KBO만의 배럴타구 기준**을 만들어야 할 것이다.

이 기준을 바탕으로 타자 개인의 능력을 수치화 후,

9/15 ~ 10/8 기간의 105경기에 대한 타자들의 출루율, 장타율 그리고 이 둘을 더한 OPS를 예측

04 데이터 전처리

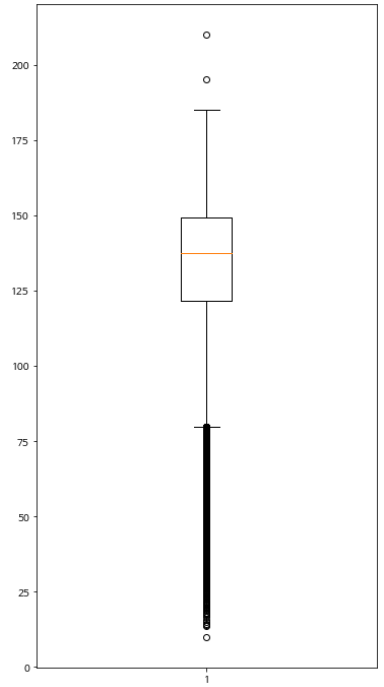
2018, 2019, 2020, 2021 시즌 타자 일반 데이터 총 1098개 정규 시즌 통계 기록이므로 이상치가 없다고 판단

타구 트래킹 데이터에 대해서만 EDA를 통한 이상치 제거

➔ Box Plot을 통해 시각화

➔ 이상치 값을 보이는 것으로 의심되는 데이터의 경우 직접 해당 경기일, 타석의 중계를 통해 **입력 설정 오류값인지, 실제 값인지 판단**

<18,19,20,21 시즌 전체 타구 트래킹 데이터 중 '타구속도' 분포>



Q1 – 1.5 * IQR 값들에 대해서는 통계적인 이상치 기준 적용 X

➔ **야구의 경우, 번트 혹은 특정 파울 타구의 경우 매우 낮은 타구 속도를 충분히 기록할 수 있으므로 이상치로 판단 X**

< 18km/h 이하의 타구속도를 보인 타구 트래킹 데이터 >

	GYEAR	G_ID	PIT_ID	PCODE	T_ID	INN	타구속도	발사각도	HIT_RESULT	투구구속	STADIUM	color	score
6984	2018	20180429HHLT0	180429_165816	72546	LT	7	14.28	57.8	파울플라이	141.47	사직	gainsboro	0
7099	2018	20180429SKWO0	180429_140319	73209	SK	1	17.25	22.6	희생번트	134.10	고척	gainsboro	0
8660	2018	20180508LTLG0	180508_203411	67539	LT	7	14.04	-29.9	희생번트	132.68	잠실	gainsboro	0
14536	2018	20180607HTKT0	180607_201018	60558	KT	6	13.61	82.7	플라이	143.21	수원	gainsboro	0
20659	2018	20180710HTNC0	180710_203633	60566	NC	4	16.59	-3.3	희생번트	140.54	마산	gainsboro	0
23814	2018	20180728HHOB0	180728_202331	62700	HH	4	14.91	-25.7	땅볼아웃	133.02	잠실	gainsboro	0
24215	2018	20180729LTWO0	180729_192918	62332	WO	4	9.88	77.8	땅볼아웃	139.25	고척	gainsboro	0
28574	2018	20180908KTWO0	180908_182840	63450	KT	4	15.64	-16.2	희생번트	139.34	고척	gainsboro	0
34155	2018	20181009HHKT0	181009_152843	78756	HH	4	17.74	-22.5	희생번트	120.03	수원	gainsboro	0

04 데이터 전처리

타구 트래킹 데이터에 대해서만 EDA를 통한 이상치 제거

➔ Box Plot을 통해 시각화

➔ 이상치 값을 보이는 것으로 의심되는 데이터의 경우 직접 해당 경기일, 타석의 중계를 통해 입력 설정 오류값인지, 실제 값인지 판단

< 190km/h 이상의 타구속도를 보인 타구 트래킹 데이터 >

	GYEAR	G_ID	PIT_ID	PCODE	T_ID	INN	타구속도	발사각도	HIT_RESULT	투구구속	STADIUM	color	score
19787	2018	20180705SKWO0	180705_194020	67394	WO	4	195.25	-4.1	1루타	139.72	고척	thistle	1
24374	2018	20180731LGOB0	180731_220412	68103	LG	9	210.13	-13.3	땅볼아웃	144.04	잠실	gainsboro	0

G_ID : 20180705SKWO0

195km/h의 타구 속도를 충분히 가지고 있는 것으로 판단(타구가 내야를 빠져나가는 시간, 좌익수에게 도달하는 시간, 좌익수가 공을 잡는 모션)

G_ID : 20180731LGOB0

평범한 땅볼 타구 210km/h는 기계 오류 등에 의해 잘못 측정된 것으로 판단

➔ 경기 중계영상을 통해 이상치 여부를 직접 눈으로 확인하고, **24374행 제거**

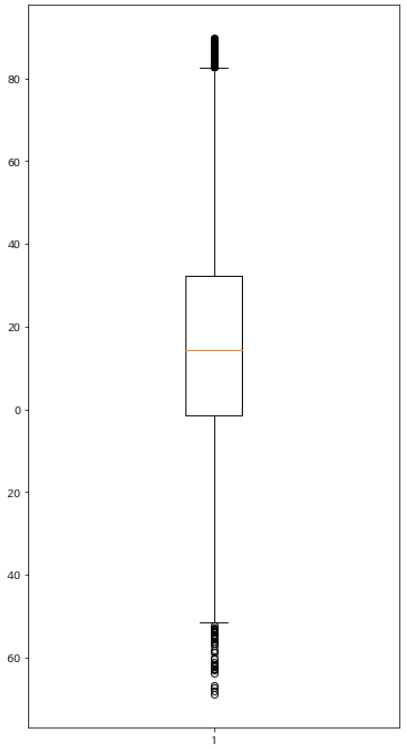
04 데이터 전처리

타구 트래킹 데이터에 대해서만 EDA를 통한 이상치 제거

➔ Box Plot을 통해 시각화

➔ 이상치 값을 보이는 것으로 의심되는 데이터의 경우 직접 해당 경기일, 타석의 중계를 통해 입력 설정 오류값인지, 실제 값인지 판단

<18,19,20,21 시즌 전체 타구 트래킹 데이터 중 '발사각도' 분포>



< 89.5도 이상의 발사각도를 보인 타구 트래킹 데이터 >

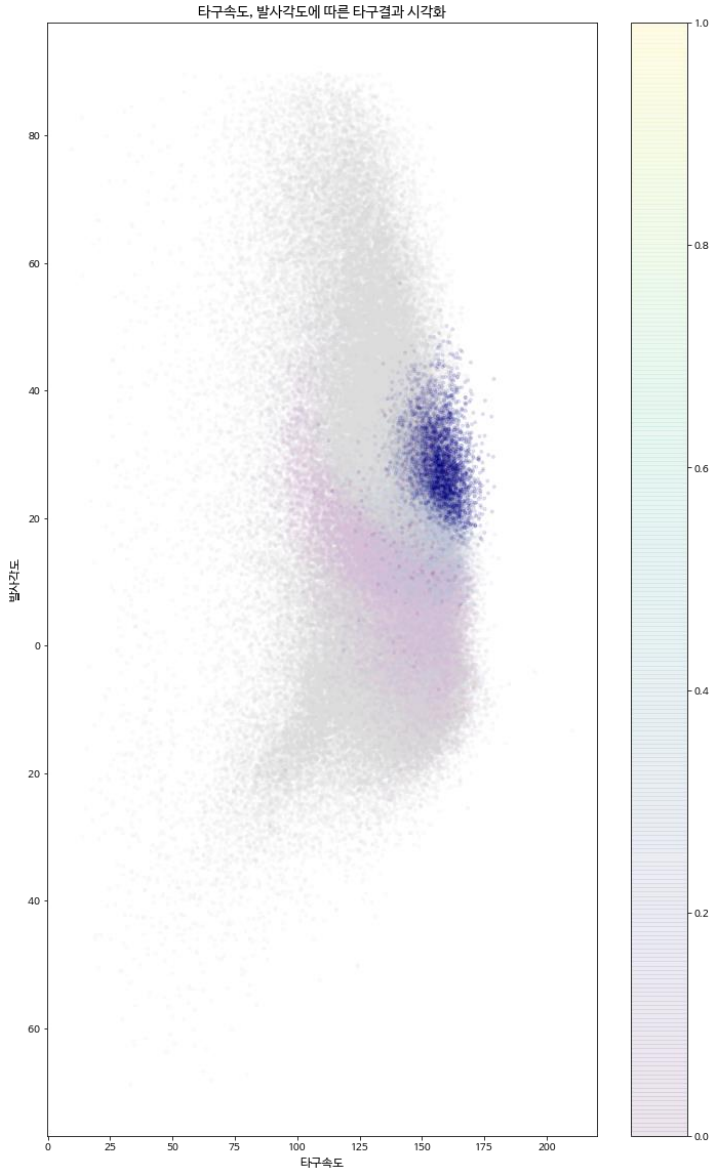
	index	타구속도	발사각도	HIT_RESULT	투구구속	STADIUM	color	score
1919	1919	132.89	89.6	파울플라이	127.35	대구	gainsboro	0
66884	66885	55.00	89.6	파울플라이	138.81	수원	gainsboro	0
70529	70530	109.52	89.7	파울플라이	137.56	대전	gainsboro	0
101700	101701	109.34	89.8	파울플라이	120.41	대전	gainsboro	0

< -65도 이하의 발사각도를 보인 타구 트래킹 데이터 >

	index	타구속도	발사각도	HIT_RESULT	투구구속	STADIUM	color	score
442	442	33.17	-68.9	희생번트	139.04	잠실	gainsboro	0
13670	13670	62.29	-66.8	땅볼아웃	150.12	잠실	gainsboro	0
15273	15273	79.84	-67.2	땅볼아웃	138.30	잠실	gainsboro	0
25897	25898	65.26	-68.1	땅볼아웃	112.96	마산	gainsboro	0

이상치 범위 내의 있는 타구들의 경우, 모두 정상적인 인플레이 상황에서 발생 가능한 타구들로 판단하여, 발사각도 기준으로는 이상치 제거 X

05 배럴타구 정의 - EDA



각 점의 색깔은 각 타구별 결과

파란색 점은 홈런, 분홍색 점은 1루타를 대표

특정 색깔이 뭉쳐 있는 부분이 그래프를 통해 관찰되며, 이는 **특정 타구 결과를 만들어 낼 수 있는 특정한 타구 속도와 타구 각도의 영역이 존재**한다는 것을 보여줌.

좋은 타구 결과를 만들어 낼 수 있는 영역 또한 제한된 구역으로 표시할 수 있음.

05 배럴타구 정의 - EDA

배럴타구 구간 산정을 위해 임의로

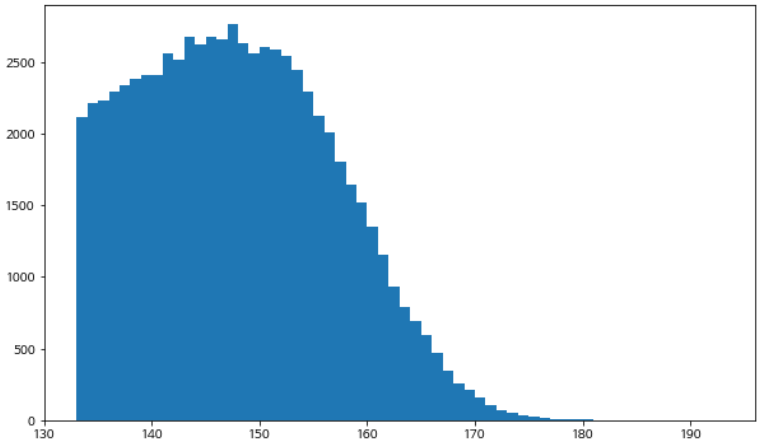
발사각도 : 5도 ~ 65도 범위에서 **1도** 단위,

타구 속도 : 133km/h ~ 193km/h 범위에서 **1km/h** 단위로

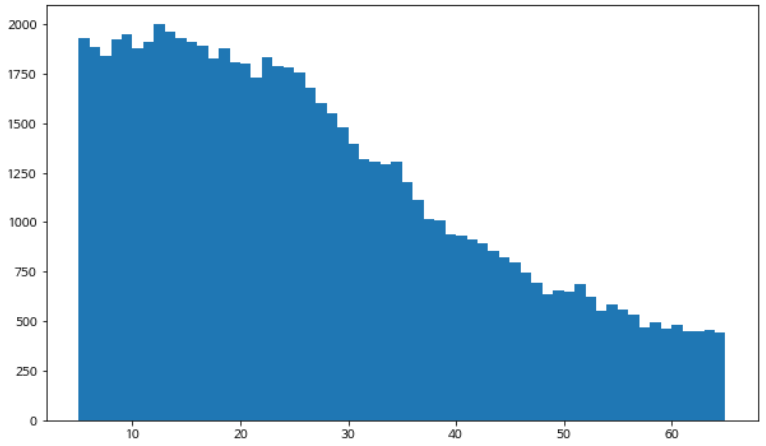
구간 분할

각 구간에 속하는 타구들의 결과를 바탕으로 구간별 장타율, 타율 계산

<산정한 구간의 타구 속도 분포>



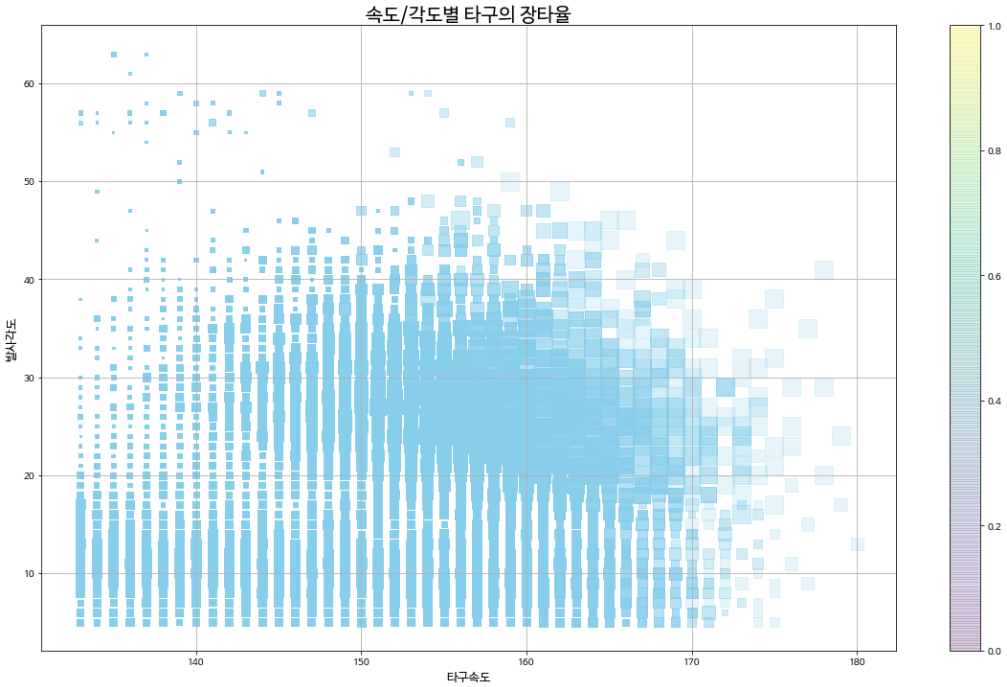
<산정한 구간의 발사 각도 분포>



발사각도_cut2	타구속도_cut2	count	score	hit_count	타율
5	133	34	0.558824	18	0.529412
	134	28	0.500000	14	0.500000
	135	31	0.612903	16	0.516129
	136	41	0.780488	29	0.707317
	137	46	0.565217	26	0.565217
...	...	...	...	...	...
63	151	1	0.000000	0	0.000000
	152	1	0.000000	0	0.000000
	153	1	0.000000	0	0.000000
	154	1	0.000000	0	0.000000
	155	3	0.000000	0	0.000000

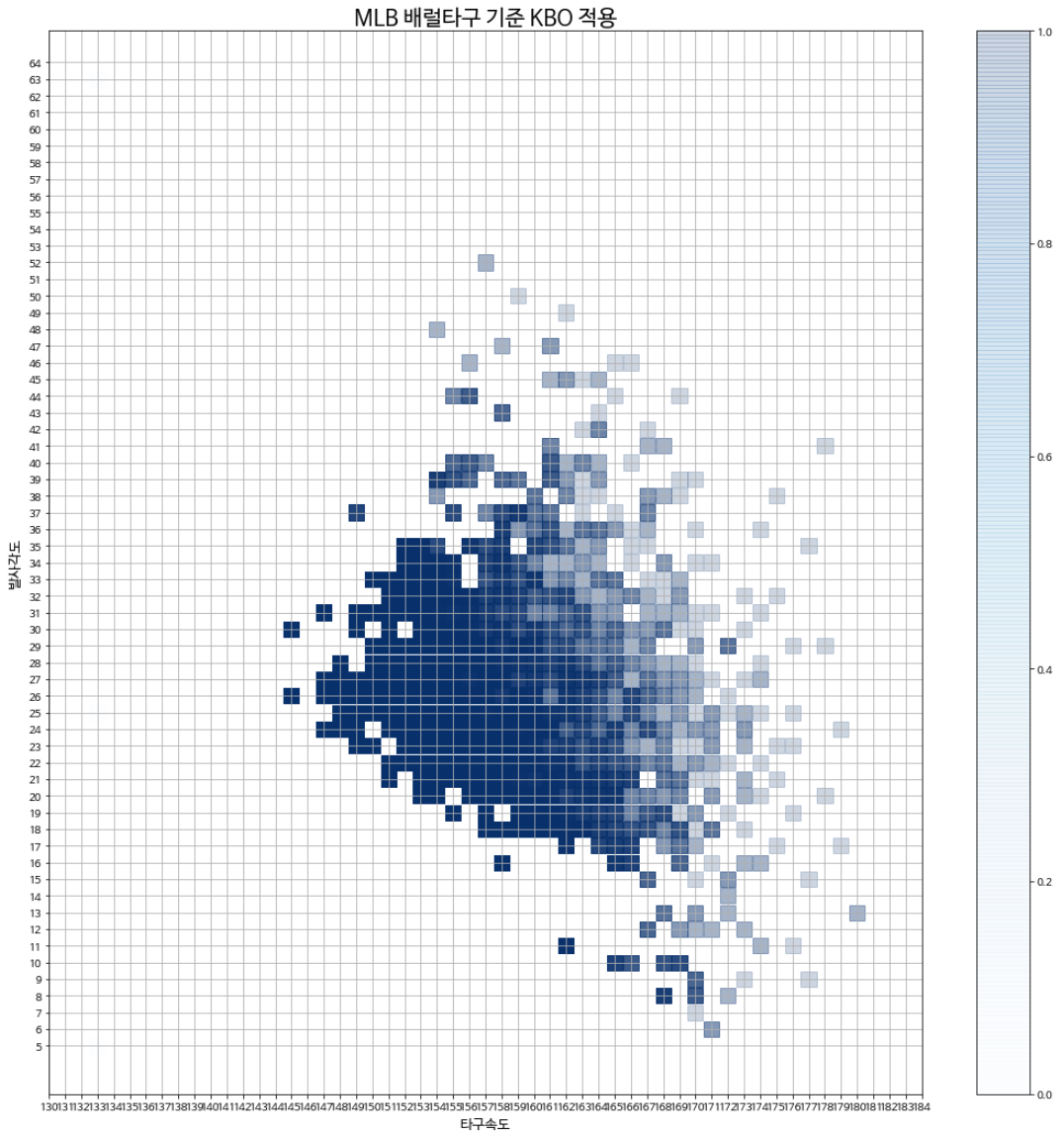
2067 rows × 4 columns

05 배럴타구 정의 - EDA



각 구간의 평균 장타율을 표현한 그래프

: 각 **정사각형의 크기가 클수록** 해당 구간의 평균 장타율이 높음



MLB의 배럴타구 기준

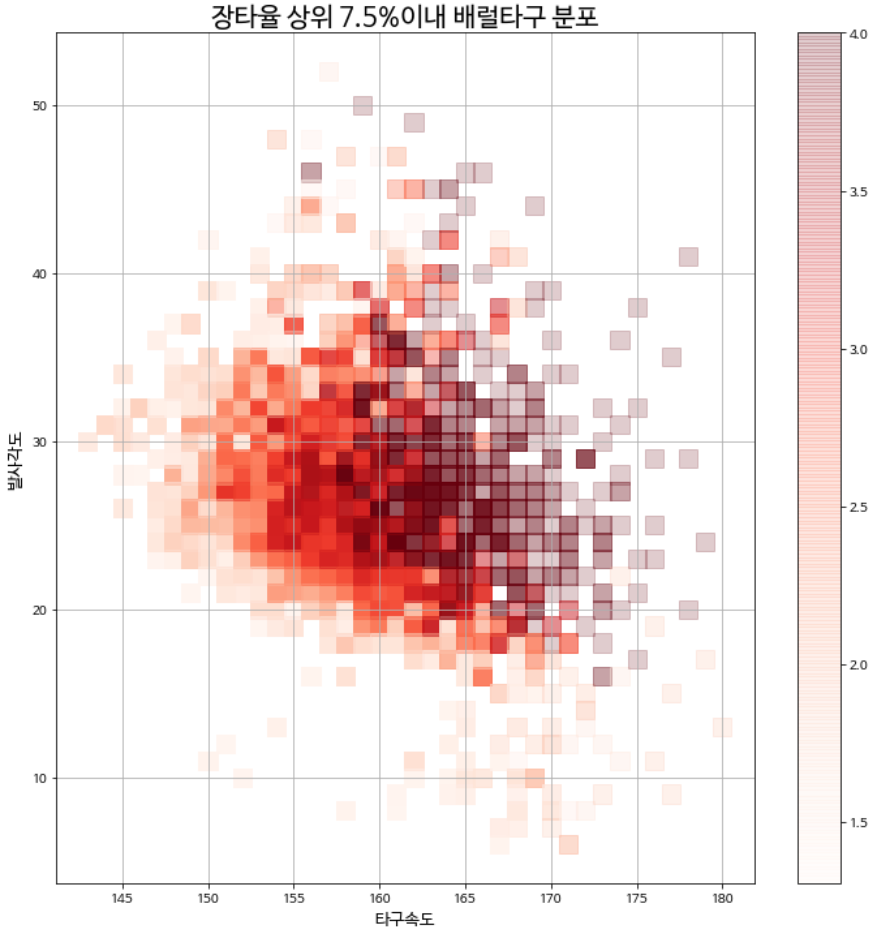
: 타구 속도와 발사각도의 조합상 평균적으로 타율 .500, 장타율 1.500이상을 생산하는 타구

을 **KBO의 타구들에 적용**한 결과

05 배럴타구 정의 - 산정기준

MLB의 배럴타구 기준을 KBO에 그대로 적용?

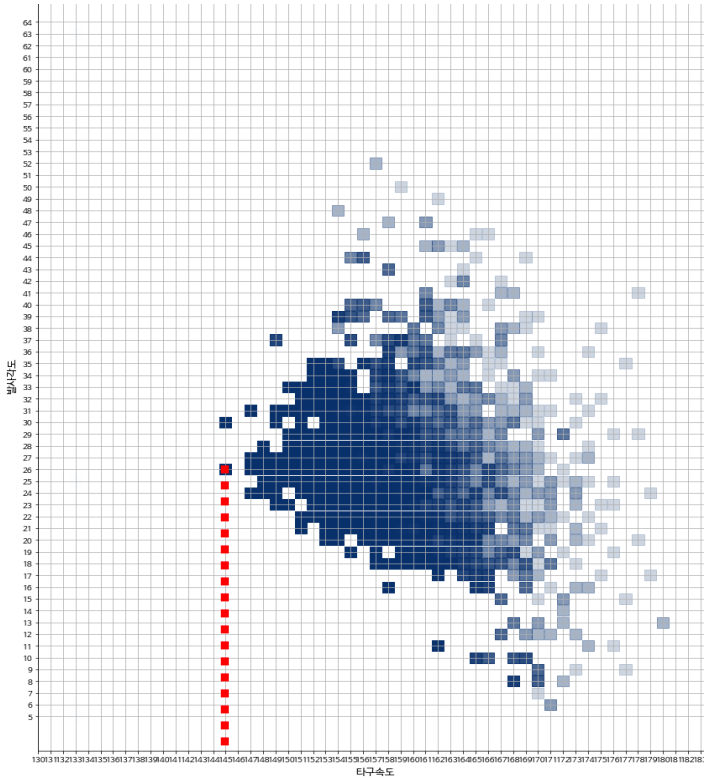
➔ 배럴 타구 기준 산정의 타당성과 MLB의 최근 4년간 리그 평균 장타율, 타율, 평균 타구 속도, 직구 평균 속도, barrel%를 KBO와 비교



지난 4년간 MLB에서 평균 7.5%의 배럴 타구가 생성되었다는 점에서 착안하여서, 전체 타구 개수 중 **장타율 상위 7.5%**의 타구들만 그래프에 포함 될 수 있도록 그래프에 표현

MLB 배럴타구 기준을 KBO에 그대로 적용한 그래프와 유사한 모양과 분포

05 배럴타구 정의 - 산정기준



145

MLB기준 : 타율 .500 이상 / 장타율 : 1.500 이상

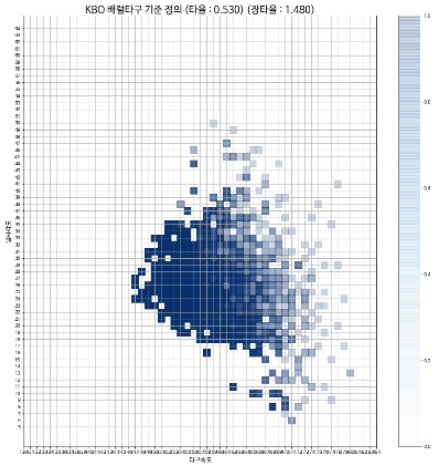
KBO 평균 타구 속도 132km/h → + 11km/h → MLB 평균 타구 속도 143km/h

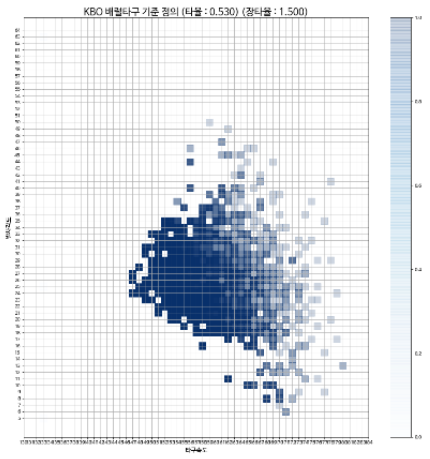
주력/수비력/송구력 등 외부요인에 비해 차이 과다 → 기준 상향 조정

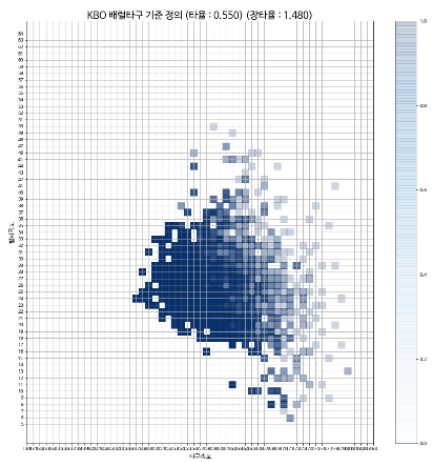
KBO 배럴타구 하한선 145km/h → + 12km/h → MLB 배럴타구 하한선 157km/h

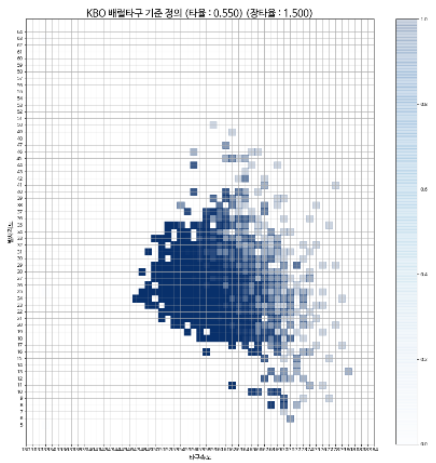
그래프에서 표현된 KBO 배럴타구 기준의 하한선이 두 리그의 평균 타구 속도의 차이를 반영하여,
주력/수비력/송구력 등의 외부요인을 고려하였더라도, 너무 낮다고 판단. **타율과 장타율 기준을 상향 조정할 필요성**

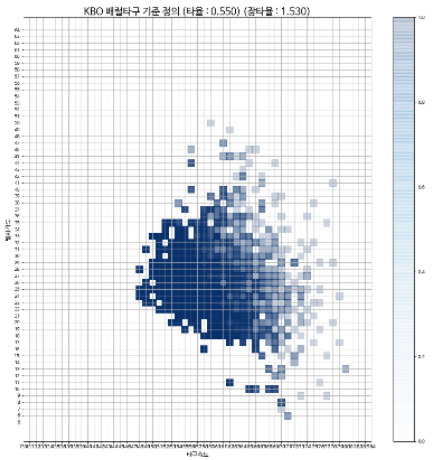
05 배럴타구 정의 - 산정 기준

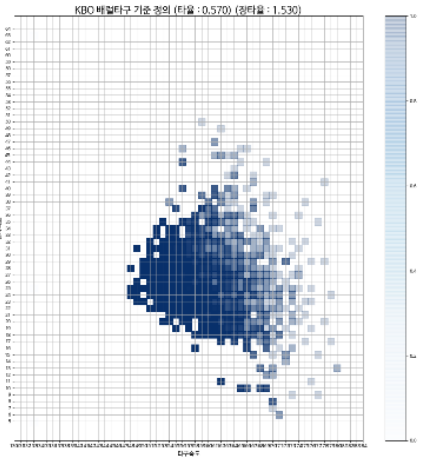


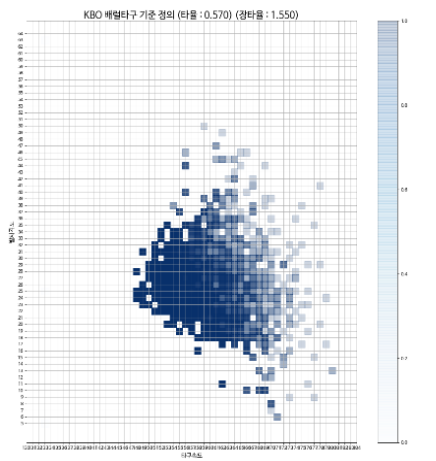


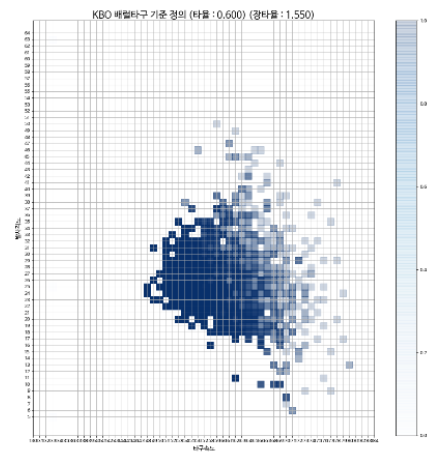






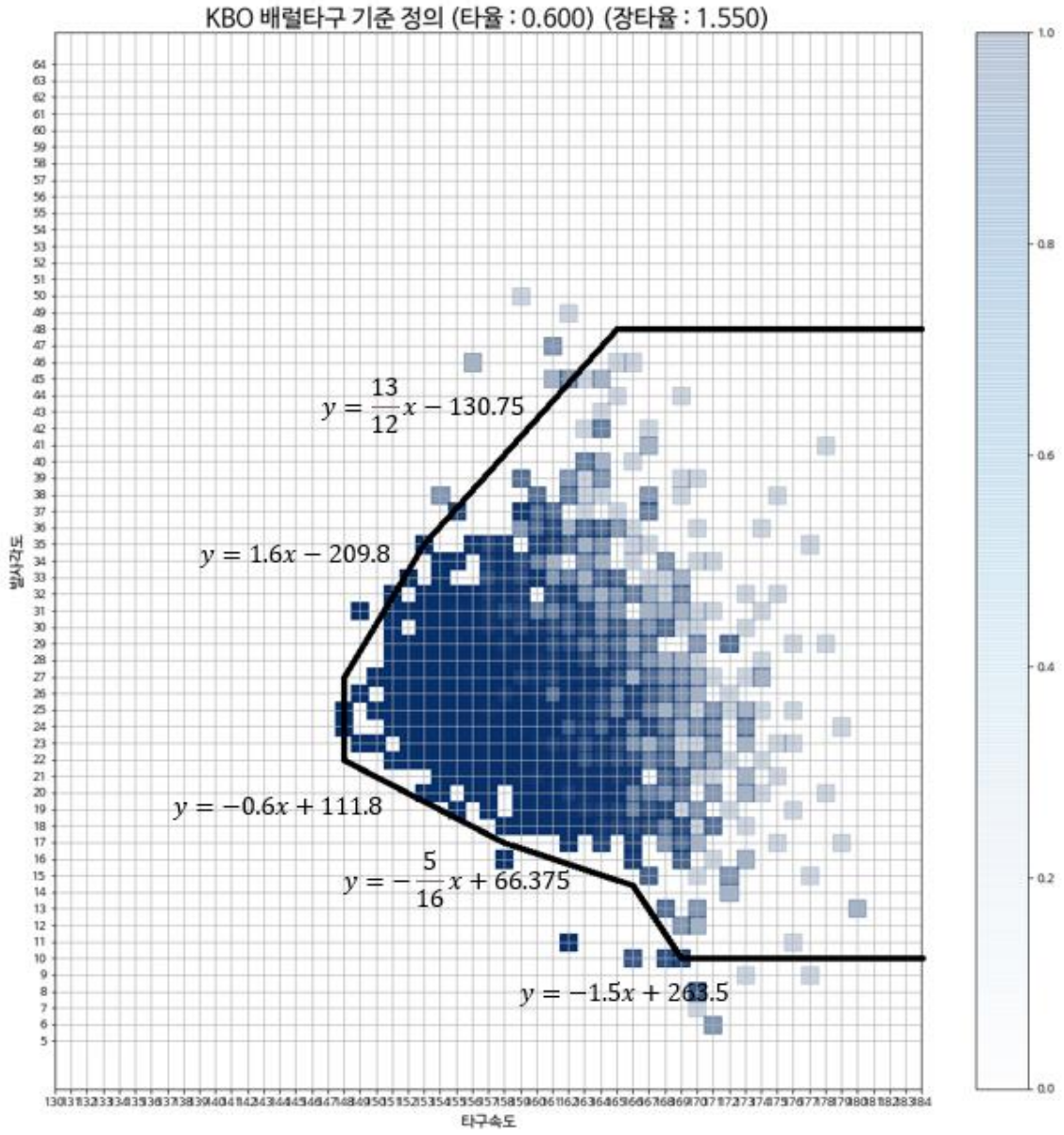






산정한 구간들에 대해, 장타율(1.470 ~ 1.600) – 타율(0.500 ~ 0.600)값을 조정해보며 MLB와 KBO의 차이를 반영한 적절한 기준값 선정

05 배럴타구 정의 - 결과 및 검증



KBO의 최종 배럴 타구 기준 선정

: 타율 0.600, 장타율 1.550 이상을 기록할 수 있다고 판단되는 타구

구간 산정

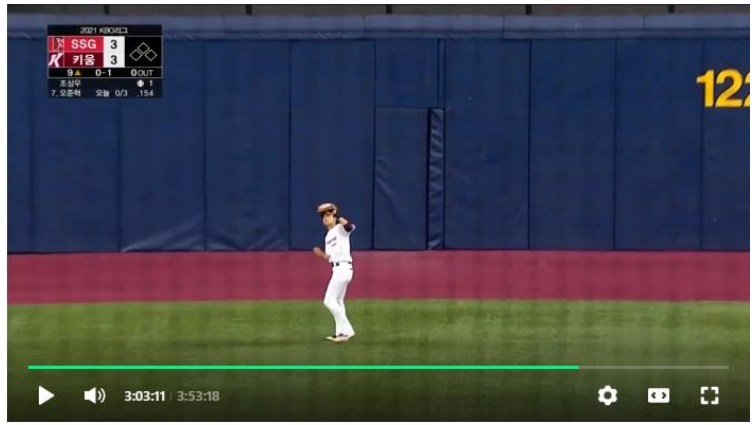
```
#산정된 배럴타구 기준 적용
df['배럴지수'] = 0

for i in range(len(df)):
  a = float(df['타구속도'][i])
  b = float(df['발사각도'][i])
  if (148 <= a) & (a < 153) :
    if (-0.6 * a + 111.8 <= b) & (b < 1.6 * a - 209.8):
      df['배럴지수'][i] = 1
  elif (153 <= a) & (a < 158) :
    if (-0.6 * a + 111.8 <= b) & (b < (13/12) * a - 130.75):
      df['배럴지수'][i] = 1
  elif (158 <= a) & (a < 165) :
    if (-0.3125 * a + 66.375 <= b) & (b < (13/12) * a - 130.75):
      df['배럴지수'][i] = 1
  elif (165 <= a) & (a < 166) :
    if (-0.3125 * a + 66.375 <= b) & (b <= 48):
      df['배럴지수'][i] = 1
  elif (166 <= a) & (a < 169) :
    if (1.5 * a + 263.5 <= b) & (b <= 48):
      df['배럴지수'][i] = 1
```

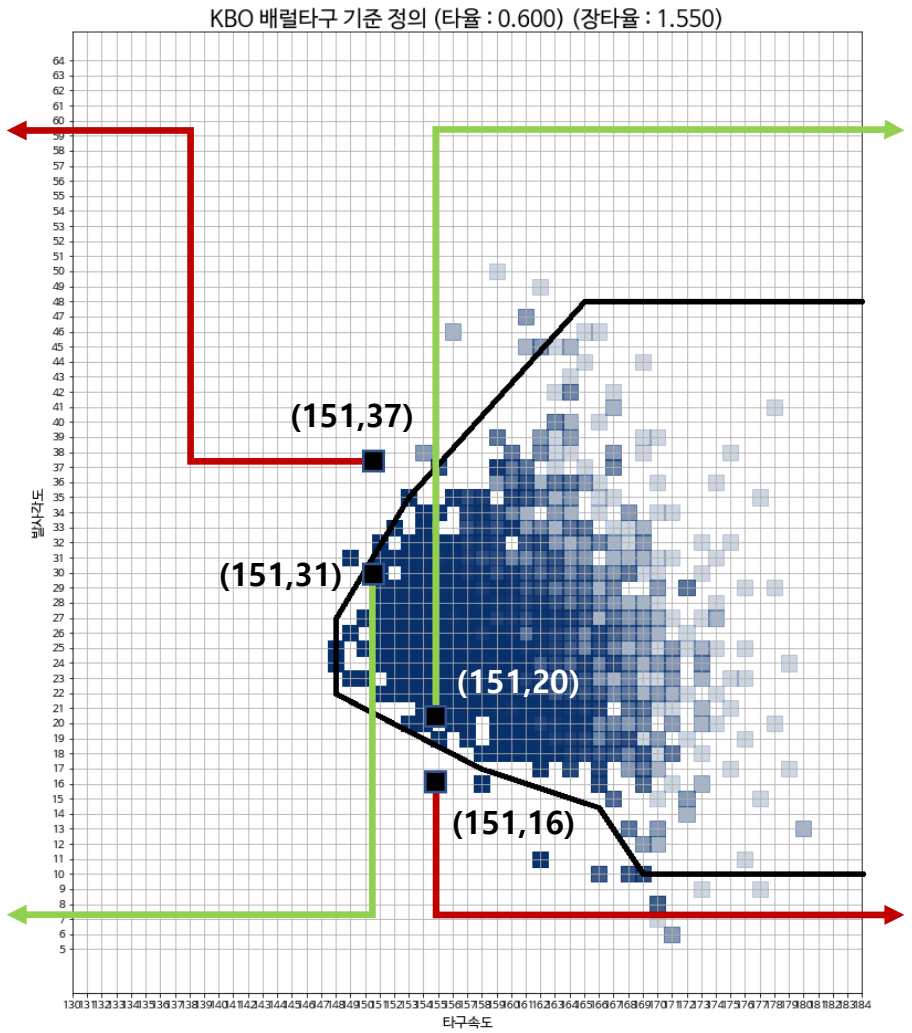
05 배럴타구 정의 - 결과 및 검증

제 9회 데이터 분석 경진대회 2021 빅콘테스트

배럴로 분류되지 않은 타구



배럴로 분류된 타구



배럴로 분류된 타구

배럴로 분류되지 않은 타구

05 배럴타구 정의 - 결과 및 검증

배럴지수 TOP5 (규정타석)

	배럴지수	장타율(순위)
라모스(LG)	0.160584	0.592(4위)
김재환(두산)	0.156550	0.495(17위)
알테어(NC)	0.131410	0.541(9위)
로하스(kt)	0.127551	0.680(1위)
박건우(두산)	0.125000	0.472(25위)

팀 배럴지수 TOP4(2020)

	배럴지수	팀 장타율
LG 트윈스	0.066704	0.428(3위)
NC 다이노스	0.055632	0.462(1위)
두산 베어스	0.055495	0.427(4위)
kt 위즈	0.049343	0.436(2위)

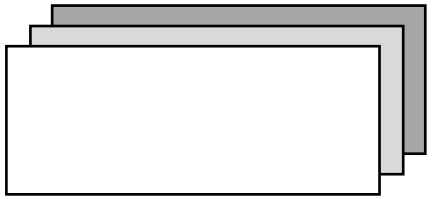
작년 규정 타석을 채운 타자 기준 **가장 높은 배럴지수를 기록한 5명**은 라모스, 김재환, 알테어, 로하스, 박건우.

이들은 장타율 상위권에 위치해 있으며 **리그를 대표하는 강타자**

팀 배럴지수 TOP4는 LG, NC, 두산, kt이며 이는 팀 **장타율 TOP4와 일치**하는 모습

06 OPS 예측 - 개요 및 문제 정의

<전체 시즌 평균 데이터>



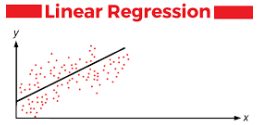
변수 조합을 통해 후보군 3개 선정

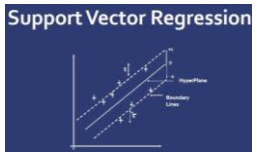
<전체 시즌 가중치 평균 데이터>

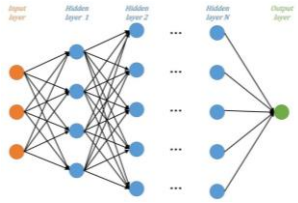
변수 조합을 통해 후보군 3개 선정

<개별 시즌 평균 데이터>

변수 조합을 통해 후보군 3개 선정







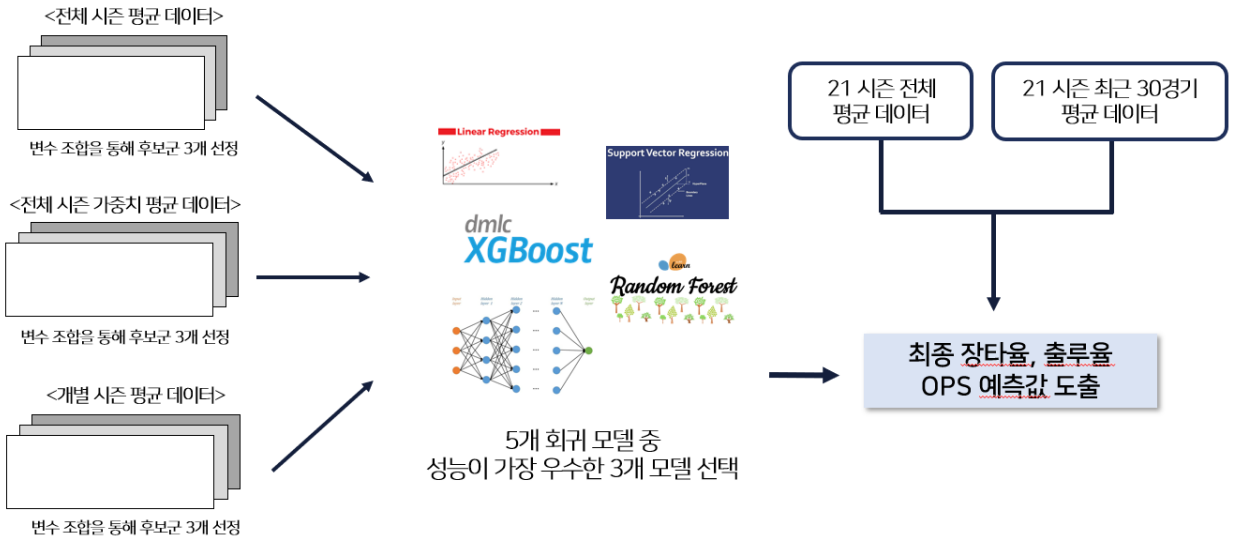
5개 회귀 모델 중
성능이 가장 우수한 3개 모델 선택

21 시즌 전체
평균 데이터

21 시즌 최근 30경기
평균 데이터

최종 장타율, 출루율
OPS 예측값 도출

06 OPS 예측 - 개요 및 문제 정의



선수 10명의 target value가 모두 주어졌으므로, 모델의 과적합 방지와 RMSE를 통해 성능을 평가할 때 의도한 **데이터셋별, 요인별 차이의 효과를 객관적으로 반영**하기 위해

최종 예측 선수 10명은 train, test dataset에서 모두 제외함

데이터셋1), 2), 3) 각각에 대해 변수를 달리하여, 총 3개의 후보군 데이터셋을 추가로 생성하고,

이렇게 생성된 데이터셋 9개에 대해

Linear Regression, SVM Regressor, Random Forest Regressor,

XGBoost Regressor, DNN regression

총 5개의 모델을 사용하여 학습하고,

성능평가 지표로는 **RMSE score**를 비교하여 최종 예측 모델을 선정

데이터셋 1) - 전체 시즌 **평균** 데이터

데이터셋 2) - 전체 시즌 가중치 평균 데이터(타자의 성장 혹은 노쇠화 등의 요인 반영)

18시즌 : **17.5%** / 19시즌 : **22.5%** / 20시즌 : **27.5%** / 21시즌 : **32.5%**의 가중치 부여

데이터셋 3) – 개별 시즌 데이터(스프링 캠프, 부상 등의 외부 요인을 고려해 타자의 **해당 시즌 OPS 예측에는 개별 시즌의 데이터만 주요**하다고 가정) + Train dataset 수 증가의 장점

데이터셋 4) - **21시즌** 전체 평균 데이터

데이터셋 5) - 21시즌 **최근 30일 경기 평균** 데이터(21.08.14 ~ 21.09.14)

06 장타율 예측 - 추가한 변수

인플레이 타구비율, 볼넷삼진비율, 뜬볼-땅골 비율 변수 추가

변수 조합을 달리하여 데이터셋 후보 1, 2, 3 선정

```
df1 = df[['PCODE', '배럴지수', '뜬공땅볼비율', '장타율']]
df2 = df[['PCODE', '배럴지수', '뜬공땅볼비율', '홈런', '장타율']]
df3 = df[['PCODE', '타율', '홈런', '인플레이타구비율', '뜬공땅볼비율', '배럴지수', '장타율']]
df3
```

규정타석 기준의 40% 기준을 적용하여, 해당 기준에 미치지 못하는 타석수를 보인 타자 데이터는 제외

```
out_index1 = h2018['타석'] < 446*0.4
out_index2 = h2019['타석'] < 446*0.4
out_index3 = h2020['타석'] < 446*0.4
out_index4 = h2021['타석'] < 75*3.1*0.4

h2018 = h2018[~out_index1]
h2019 = h2019[~out_index2]
h2020 = h2020[~out_index3]
h2021 = h2021[~out_index4]
```

규정타석의 규정을 그대로 사용하여 데이터의 Quality를 높이는 문제와,
모델 학습을 위한 데이터의 수를 늘리는 **Trade-off 관계**를 고려하여

규정타석 규정의 **40%**로 데이터셋의 규정타석 기준 임의 산정

06 장타율 예측 - 추가한 변수

데이터셋 1) - 전체 시즌 평균 데이터

	PCODE	배럴지수	타석	타수	타율	안타	홈런	루타	장타율	희생플라이	볼넷	삼진	고의4구	사구	병살타	출루율	인플레이타구비율	뜬공땅볼비율	OPS	볼넷삼진비율
3	50165	0.157248	349.50	308.0	0.26050	82.50	23.00	166.50	0.50700	2.00	36.50	89.50	2.50	3.0	4.5	0.353693	0.307494	2.253521	0.860693	0.407821
11	50350	0.062500	271.00	244.0	0.25400	62.00	2.00	82.00	0.33600	3.00	22.00	37.00	1.00	2.0	6.0	0.319853	0.297030	1.897436	0.655853	0.594595
14	50458	0.000000	241.50	213.0	0.24950	52.50	1.00	61.50	0.29300	2.50	18.50	32.50	0.00	1.0	0.5	0.306383	0.290960	1.112360	0.599383	0.569231
15	50468	0.108434	163.00	132.0	0.28000	37.00	6.00	63.00	0.47700	1.00	23.00	41.00	0.00	7.0	3.0	0.411043	0.369048	1.625000	0.888043	0.560976
16	50469	0.078261	217.00	196.0	0.20900	41.00	8.00	72.00	0.36700	3.00	18.00	58.00	0.00	0.0	5.0	0.271889	0.259843	1.629630	0.638889	0.310345
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
428	79402	0.022989	447.75	390.5	0.26225	104.75	5.25	138.25	0.34375	2.25	40.50	59.75	0.50	8.5	12.0	0.348785	0.307811	1.334507	0.692535	0.677824
430	79456	0.004754	239.50	218.0	0.30200	65.75	0.75	76.75	0.35425	1.00	14.00	23.50	1.00	2.5	7.5	0.352008	0.337224	0.790816	0.706258	0.595745
433	79608	0.039496	415.75	370.0	0.31700	117.50	10.25	174.75	0.46725	5.25	34.25	43.25	1.25	4.5	9.5	0.379290	0.344578	1.822034	0.846540	0.791908
435	99606	0.042017	205.00	183.0	0.29500	54.00	4.00	77.00	0.42100	1.00	16.00	49.00	1.00	5.0	5.0	0.368932	0.387597	1.074074	0.789932	0.326531
436	99810	0.036697	312.00	283.0	0.31800	90.00	3.00	118.00	0.41700	3.00	25.00	35.00	1.00	0.0	3.0	0.371795	0.359504	1.142857	0.788795	0.714286

153 rows × 20 columns

데이터셋 2) - 전체 시즌 가중치 평균 데이터

	PCODE	타율	홈런	인플레이타구비율	뜬공땅볼비율	배럴지수	장타율
3	50165	0.26050	23.00	0.307494	2.239521	0.182801	0.50700
11	50350	0.25400	2.00	0.297030	1.897436	0.068750	0.33600
14	50458	0.24950	1.00	0.290960	1.101408	0.000000	0.29300
15	50468	0.28000	6.00	0.369048	1.625000	0.119277	0.47700
16	50469	0.20900	8.00	0.259843	1.629630	0.086087	0.36700
...	...	...	...	...	...	...	...
428	79402	0.26225	5.25	0.307811	1.346857	0.021220	0.34375
430	79456	0.30200	0.75	0.337224	0.826477	0.003962	0.35425
433	79608	0.31700	10.25	0.344578	1.793256	0.035714	0.46725
435	99606	0.29500	4.00	0.387597	1.074074	0.029412	0.42100
436	99810	0.31800	3.00	0.359504	1.142857	0.025688	0.41700

153 rows × 7 columns

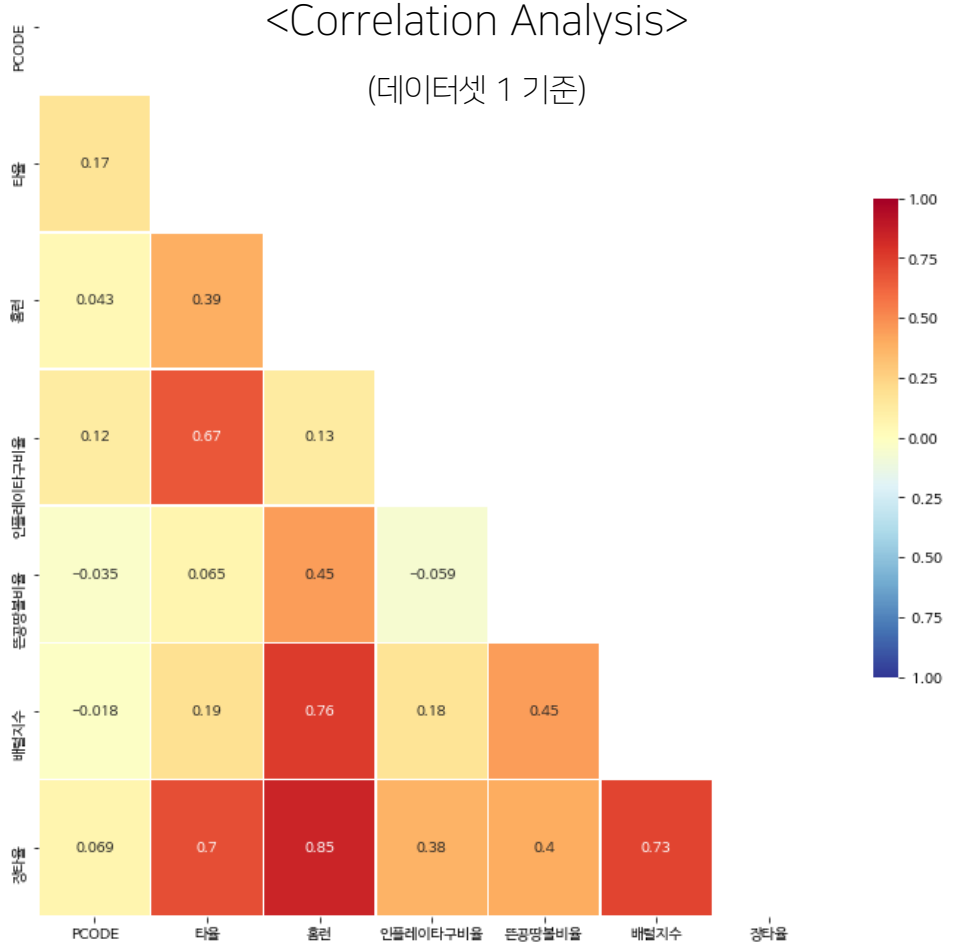
데이터셋 3) – 개별 시즌 데이터

	PCODE_YEAR	타율	홈런	인플레이타구비율	뜬공땅볼비율	배럴지수	장타율
0	50165_2020	0.278	38	0.324111	2.386364	0.160584	0.592
1	50165_2021	0.243	8	0.276119	2.037037	0.150376	0.422
2	50350_2020	0.254	2	0.297030	1.897436	0.062500	0.336
3	50458_2020	0.232	1	0.280193	1.239130	0.000000	0.272
4	50458_2021	0.267	1	0.306122	0.976744	0.000000	0.314
...	...	...	...	...	...	...	...
461	79608_2019	0.315	5	0.346032	1.666667	0.020619	0.412
462	79608_2020	0.286	8	0.309859	2.203390	0.028571	0.413
463	79608_2021	0.325	5	0.376471	1.268293	0.048193	0.481
464	99606_2018	0.295	4	0.387597	1.074074	0.042017	0.421
465	99810_2018	0.318	3	0.359504	1.142857	0.036697	0.417

466 rows × 7 columns

<Correlation Analysis>

(데이터셋 1 기준)



06 장타율 예측 - 모델 구축 및 훈련


```
from sklearn.linear_model import LinearRegression
lr_model = LinearRegression()
lr_model.fit(x_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
y_pred = lr_model.predict(x_test)
```

사용한 모델
Linear Regression

모델 선정 이유
적절한 독립변수 활용과 정규화 과정을 거치면
다른 복잡한 모델에 비해 높은 예측력과 설명력을 동시에 갖기 때문

(데이터의 일반적 인 분포를 고려하여, Standard Scaler 사용)

```
svr_model = SVR(C=1.0, epsilon=0.2, kernel='rbf')
svr_model.fit(x_train, y_train)
```

```
SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.2, gamma='scale',
    kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

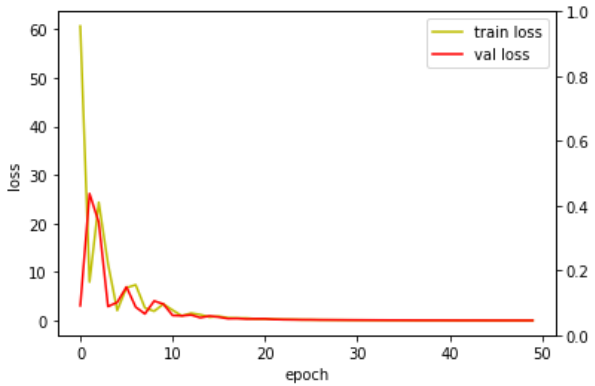
```
y_pred = svr_model.predict(x_test)
```

사용한 모델
Support Vector Machine Regression

모델 선정 이유
데이터의 수가 많지 않기 때문에, 학습속도가 느린 SVR의 단점 X
Overfitting되는 경우가 적음

(데이터의 일반적 인 분포를 고려하여, Standard Scaler 사용)

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                192
_________________________________________________________________
dense_1 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 33
=================================================================
Total params: 2,305
Trainable params: 2,305
Non-trainable params: 0
_________________________________________________________________
```



사용한 모델
Deep Neural Network Regression

모델 선정 이유
2층의 hidden layer, relu activation function 사용
데이터의 비선형적인 특징 고려

06 장타율 예측 - 모델 구축 및 훈련


```
estimator = RandomForestRegressor()
param_grid = {
            'max_depth': [3, 4, 5, 6],
            "n_estimators"      : [100,150,200],
            'min_samples_leaf': [1, 2, 3],
            "min_samples_split" : [2, 3, 4],
            }
grid = GridSearchCV(estimator, param_grid, n_jobs=-1, cv=5)

grid.fit(x_train, y_train)
```

```
GridSearchCV(cv=5, error_score=nan,
             estimator=RandomForestRegressor(bootstrap=True, ccp_alpha=0.0,
                                             criterion='mse', 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=None,
                                             oob_score=False, random_state=None,
                                             verbose=0, warm_start=False),
```

사용한 모델

Random Forest Regressor

Decision Tree기반 Bagging을 활용한 앙상블 알고리즘

모델 선정 이유

예측의 변동성이 줄고, 과적합을 방지함

결측치에 대해 Robust한 학습이 가능하기 때문에, 데이터수가 적은 경우 효과적

Grid Search를 통한 최적의 parameter tuning

```
xgb_model = xgboost.XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=0.75, gamma=0,
             importance_type='gain', learning_rate=0.02, max_delta_step=0,
             max_depth=7, min_child_weight=1, missing=None, n_estimators=1000,
             n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
             reg_alpha=0.03, reg_lambda=1, scale_pos_weight=1, seed=None,
             silent=None, subsample=1, verbosity=1)
xgb_model.fit(x_train,y_train)
```

사용한 모델

XGBoost Regressor

Decision Tree기반 Boosting을 활용한 앙상블 알고리즘

모델 선정 이유

병렬 처리를 이용한 빠른 학습 속도와 우수한 회귀 성능

Greedy 알고리즘을 통한 prunning을 통한 과적합 방지

Grid Search를 통한 최적의 parameter tuning

06 장타율 예측 - 모델 선정 및 장타율 예측

RMSE	Linear Regression	SVM Regression	Random Forest	XGBoost Regressor	DNN Regression
1-df1	0.050434181	0.053617175	0.05026235	0.053986618	0.056097957
1-df2	0.035939917	0.06036132	0.038473453	0.038987665	0.061332755
1-df3	0.020625518	0.080701662	0.021703205	0.021975356	0.032886636
2-df1	0.052219195	0.064047883	0.05781137	0.058628853	0.058746854
2-df2	0.038708033	0.068708343	0.034226742	0.034373811	0.045040512
2-df3	0.015867147	0.093634368	0.026819903	0.020690334	0.028152229
3-df1	0.060608077	0.063210976	0.067153797	0.077771392	0.068625346
3-df2	0.037691405	0.06624536	0.043528447	0.04699012	0.037703187
3-df3	0.024537555	0.101503589	0.025691432	0.025572611	0.02766533

데이터셋 1,2,3번에 대해 모두 공통적으로,

타율, 홈런, 인플레이타구비율, 뜬공땅볼비율, 배럴지수를 통해

장타율을 예측한 데이터셋(df3)에서 가장 높은 성능을 보임

예측 모델로는 **Linear Regression, Random Forest, XGBoost Regressor** 사용

데이터셋 1,2,3 각각에 대한

성능이 우수한 3개 모델의 예측값

+ 21년 기록

+ 최근 30경기의

ensemble을 통해

최종 예측값 도출

```
slg = (slg1 + slg2 + slg3 + slg4 + slg5)/5
slg
```

```
PCODE
76232    0.529828
68050    0.534003
75847    0.541094
67341    0.530886
79192    0.469827
78224    0.494640
78513    0.448415
76290    0.461833
79215    0.444699
67872    0.432151
dtype: float64
```

06 출루율 예측 - 추가한 변수

볼넷삼진비율, P/PA, 초구, 전체, 컨택%, 2S후 커트%, 2S후 선구% 변수 추가

P/PA : 타석 당 투구수

초구 : 초구 배트 적극성, 초구에 배트가 나올 확률

전체 : 전체 배트 적극성, 전체 투구수에서 배트가 나올 확률

컨택 % : 배트를 휘둘렀을 때 공을 맞춘 확률

2S후커트% : 2스트라이크 이후 커트율

2S후선구% : 2스트라이크 이후 선구율

장타율과 같은 기준의 정규타석 기준(QAB * 0.4)으로 이를 충족하지 못하는 타자들을 제거했으며,
변수 추가하는 과정에서, 선수이름과 PCODE를 연결하며 개명한 선수 2명 확인 및 동일 이름으로 처리

```
pf[pf['PCODE']==67539]
```

	index	PCODE	NAME
343	343	67539	나종덕
969	416	67539	나균안

```
pf[pf['PCODE']==62895]
```

	index	PCODE	NAME
113	113	62895	한동민
1071	205	62895	한동민

변수 조합을 달리하여 데이터셋 후보군 1, 2, 3 선정

```
df1 = df[['PCODE', '타율', '홈런', '볼삼비', 'P/PA', '초구', '전체', '컨택%', '2S후커트%', '2S후선구%', '출루율']]
df2 = df[['PCODE', '타율', '홈런', '볼삼비', 'P/PA', '전체', '컨택%', '2S후선구%', '출루율']]
df3 = df[['PCODE', '타율', '홈런', '볼삼비', '전체', '2S후선구%', '출루율']]
```

06 출루율 예측 - 추가한 변수

데이터셋 1) - 전체 시즌 평균 데이터

	PCODE	타율	홈런	볼삼비	P/PA	초구	전체	컨택%	2S후커트%	2S후선구%	출루율
0	74540	0.278500	16.250000	0.451362	3.7400	26.775	45.275	77.725	74.750	36.825	0.345431
1	68050	0.337750	18.750000	0.670487	4.0000	31.225	44.225	78.775	74.625	37.525	0.415301
2	62925	0.263250	4.500000	0.387931	3.7300	14.800	47.050	85.450	81.650	26.400	0.337010
3	61353	0.285000	3.000000	0.215686	3.7250	36.950	52.850	76.050	73.200	30.300	0.327765
4	62404	0.296750	15.250000	0.470930	3.9175	32.025	47.825	78.075	77.050	32.850	0.366544
...	...	...	...	...	...	...	...	...	...	...	...
121	79240	0.316750	6.250000	0.881481	3.5875	15.275	43.375	89.325	88.600	31.200	0.370721
122	68730	0.261333	17.333333	0.451613	3.7100	30.150	47.700	78.350	74.400	35.000	0.352336
123	66108	0.258500	2.000000	1.006623	4.2800	18.950	36.800	84.050	79.200	42.350	0.433815
124	76313	0.301500	18.250000	0.524691	3.9325	25.000	44.175	80.700	76.050	34.225	0.366316
125	51725	0.257000	7.000000	0.275862	3.8800	23.500	46.700	75.700	72.600	27.500	0.305970

126 rows × 11 columns

데이터셋 2) - 전체 시즌 가중치 평균 데이터

	NAME	P/PA	전체	초구	컨택%	2S후커트%	2S후선구%	PCODE	GYEAR	GAMENUM	타석	타수	타율	안타	홈런	루타	장타율	희생플라이	볼넷	삼진	고의4구	사구	병살타	출루율	볼삼비
0	강경학	3.80650	34.2700	20.3150	69.2000	61.0350	33.3400	61700	2019.0	78.65	232.050	191.900	0.222700	48.000	3.950	69.35	0.322550	1.450	32.55	55.850	0.000	3.050	3.55	0.365145	0.582811
1	강로한	3.44700	45.0900	33.2100	57.6900	50.2200	27.4500	65515	2019.0	93.60	275.400	259.200	0.216000	62.100	3.600	97.20	0.337500	0.900	11.70	96.300	0.000	2.700	2.70	0.278689	0.121495
2	강민호	3.72750	44.9275	25.5975	78.3475	75.5250	36.9725	74540	2019.5	103.25	367.725	328.525	0.283950	91.750	15.575	152.85	0.469475	4.025	28.50	58.825	2.850	6.225	10.90	0.349409	0.484488
3	강백호	4.01950	43.8725	30.1375	79.0825	75.2475	37.3975	68050	2019.5	110.10	480.775	416.350	0.345475	140.000	17.575	222.95	0.540850	3.225	58.85	81.475	5.650	2.350	6.40	0.425245	0.722307
4	강승호	4.97900	68.3800	40.8200	96.5900	94.1200	38.0900	63123	2021.0	61.10	217.100	195.000	0.295100	44.200	2.600	61.10	0.406900	0.000	14.30	46.800	1.300	2.600	3.90	0.292683	0.305556
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
162	호잉	2.95100	38.1300	24.2850	62.5150	59.1100	27.8800	68730	2018.5	105.50	442.750	399.350	0.234900	117.450	18.600	204.60	0.407550	4.000	34.95	73.050	5.750	4.000	4.45	0.361902	0.478439
163	홍창기	5.12600	43.9300	22.6450	100.9950	95.1100	50.8250	66108	2020.5	123.00	500.500	399.250	0.373150	121.850	4.700	170.20	0.514700	2.300	85.30	76.450	1.850	12.000	7.00	0.441382	1.115762
164	황대인	4.55000	66.9500	45.7600	98.2800	84.1100	38.0900	65610	2021.0	44.20	163.800	152.100	0.310700	36.400	5.200	57.20	0.488800	3.900	6.50	33.800	0.000	0.000	2.60	0.264000	0.192308
165	황재균	3.94825	44.2675	25.0800	80.6750	76.3500	34.3675	76313	2019.5	105.20	449.250	402.125	0.303650	121.375	16.925	200.30	0.491650	2.750	40.35	75.300	0.625	2.850	8.05	0.368175	0.535857
166	힐리	5.04400	60.7100	30.5500	98.4100	94.3800	35.7500	51725	2021.0	87.10	348.400	323.700	0.334100	83.200	9.100	127.40	0.512200	1.300	20.80	75.400	0.000	2.600	7.80	0.305970	0.275862

167 rows × 25 columns

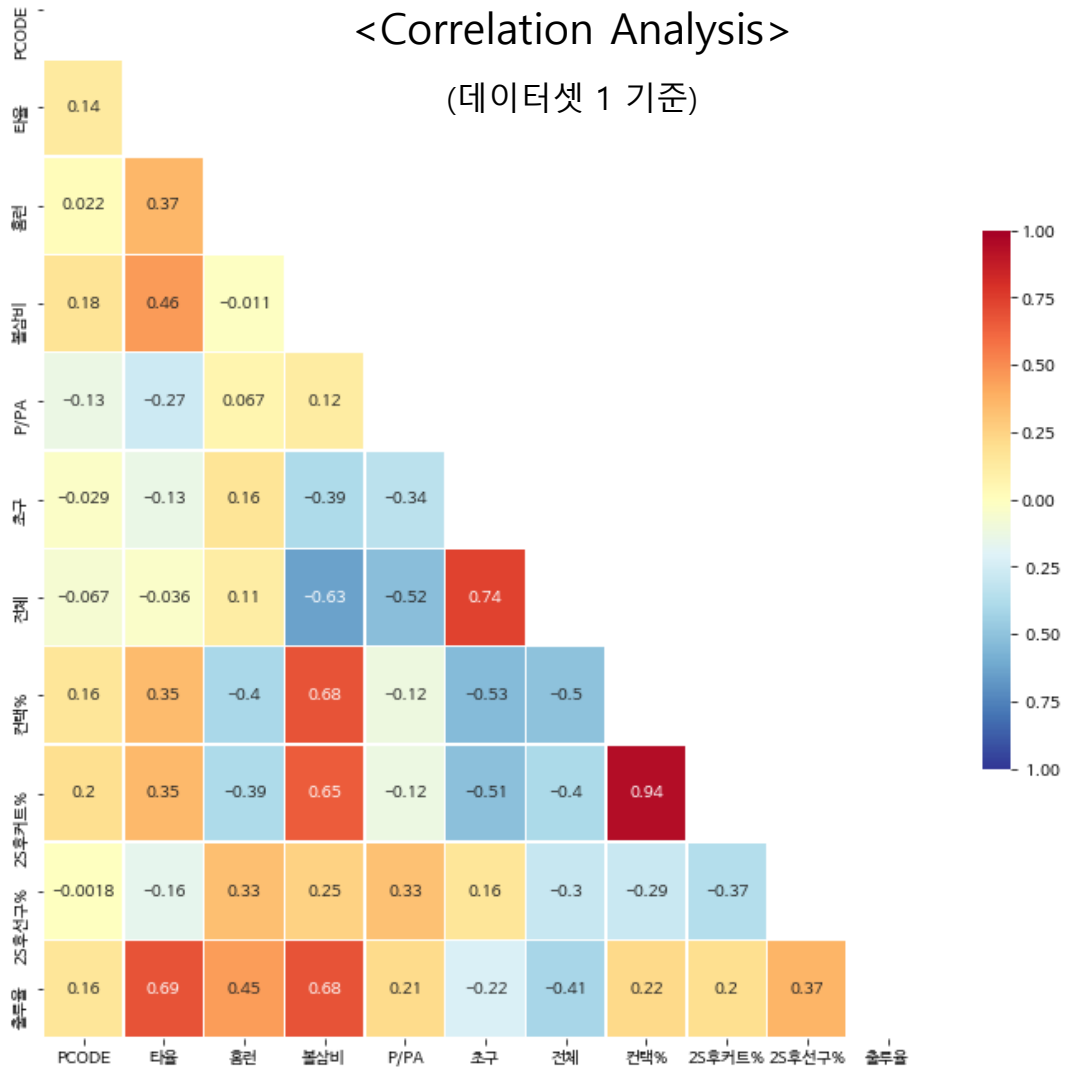
데이터셋 3) – 개별 시즌 데이터

	PCODE_x	이름	P/PA	전체	초구	컨택%	2S후커트%	2S후선구%	GYEAR_x	PCODE_YEAR	PCODE_y	NAME	GYEAR_y	GAMENUM	타석	타수	타율	안타	홈런	루타	장타율	희생플라이	볼넷	삼진	고의4구	사구	병살타	출루율	볼삼비
0	60558	오태곤	3.75	48.2	30.0	71.5	66.6	33.4	2018	60558_2018	60558	오태곤	2018.0	128.0	374.0	342.0	0.254	87.0	12.0	144.0	0.421	1.0	24.0	92.0	0.0	4.0	7.0	0.309973	0.260870
1	61102	유강남	3.76	50.1	28.0	78.3	76.8	31.6	2018	61102_2018	61102	유강남	2018.0	132.0	465.0	425.0	0.296	126.0	19.0	216.0	0.508	1.0	28.0	80.0	1.0	9.0	21.0	0.353448	0.350000
2	61186	이천웅	3.63	46.9	35.3	82.4	80.2	36.7	2018	61186_2018	61186	이천웅	2018.0	112.0	405.0	359.0	0.340	122.0	2.0	156.0	0.435	2.0	39.0	48.0	2.0	3.0	11.0	0.409877	0.812500
3	61208	정진호	3.75	48.4	29.8	84.7	79.8	29.9	2018	61208_2018	61208	정진호	2018.0	111.0	299.0	269.0	0.301	81.0	2.0	101.0	0.375	1.0	19.0	44.0	0.0	1.0	8.0	0.348276	0.431818
4	61353	고종욱	3.81	51.0	39.9	72.6	63.8	34.8	2018	61353_2018	61353	고종욱	2018.0	102.0	351.0	330.0	0.279	92.0	6.0	132.0	0.400	4.0	15.0	96.0	1.0	1.0	2.0	0.310541	0.156250
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
395	51203	안재석	3.87	50.2	33.7	77.0	71.0	30.8	2021	51203_2021	51203	안재석	2021.0	52.0	144.0	131.0	0.275	36.0	2.0	52.0	0.397	3.0	7.0	35.0	0.0	3.0	3.0	0.319444	0.200000
396	51463	피렐라	3.79	44.3	18.5	79.1	77.5	37.8	2021	51463_2021	51463	피렐라	2021.0	80.0	361.0	324.0	0.312	101.0	20.0	177.0	0.546	2.0	30.0	45.0	1.0	5.0	3.0	0.378453	0.666667
397	51725	힐리	3.88	46.7	23.5	75.7	72.6	27.5	2021	51725_2021	51725	힐리	2021.0	67.0	268.0	249.0	0.257	64.0	7.0	98.0	0.394	1.0	16.0	58.0	0.0	2.0	6.0	0.305970	0.275862
398	51817	주신수	4.18	39.2	23.2	74.0	62.9	43.6	2021	51817_2021	51817	주신수	2021.0	75.0	319.0	251.0	0.255	64.0	13.0	114.0	0.454	3.0	56.0	69.0	1.0	9.0	6.0	0.406250	0.811594
399	69102	문보경	4.33	39.0	24.7	81.4	79.8	38.1	2021	69102_2021	69102	문보경	2021.0	46.0	167.0	137.0	0.270	37.0	7.0	67.0	0.489	2.0	27.0	30.0	0.0	0.0	4.0	0.385542	0.900000

400 rows × 29 columns

<Correlation Analysis>

(데이터셋 1 기준)



06 장타율 예측 - 모델 구축 및 훈련


```
from sklearn.linear_model import LinearRegression
lr_model = LinearRegression()
lr_model.fit(x_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
y_pred = lr_model.predict(x_test)
```

사용한 모델
Linear Regression

모델 선정 이유
적절한 독립변수 활용과 정규화 과정을 거치면
다른 복잡한 모델에 비해 높은 예측력과 설명력을 동시에 갖기 때문

(데이터의 일반적 인 분포를 고려하여, Standard Scaler 사용)

```
svr_model = SVR(C=1.0, epsilon=0.2, kernel='rbf')
svr_model.fit(x_train, y_train)
```

```
SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.2, gamma='scale',
    kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

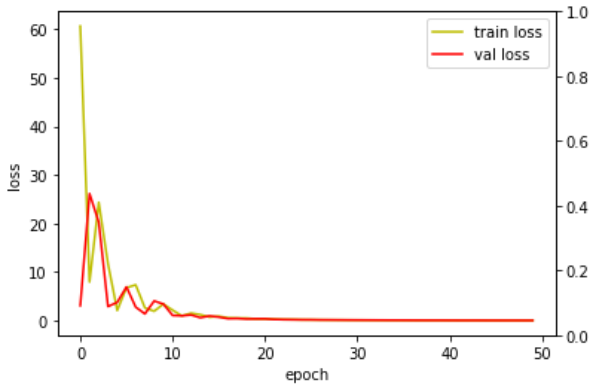
```
y_pred = svr_model.predict(x_test)
```

사용한 모델
Support Vector Machine Regression

모델 선정 이유
데이터의 수가 많지 않기 때문에, 학습속도가 느린 SVR의 단점 X
Overfitting되는 경우가 적음

(데이터의 일반적 인 분포를 고려하여, Standard Scaler 사용)

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                192
_________________________________________________________________
dense_1 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 33
=================================================================
Total params: 2,305
Trainable params: 2,305
Non-trainable params: 0
_________________________________________________________________
```



사용한 모델
Deep Neural Network Regression

모델 선정 이유
2층의 hidden layer, relu activation function 사용
데이터의 비선형적인 특징 고려

06 장타율 예측 - 모델 구축 및 훈련


```
estimator = RandomForestRegressor()
param_grid = {
            'max_depth': [3, 4, 5, 6],
            "n_estimators"      : [100,150,200],
            'min_samples_leaf': [1, 2, 3],
            "min_samples_split" : [2, 3, 4],
            }
grid = GridSearchCV(estimator, param_grid, n_jobs=-1, cv=5)

grid.fit(x_train, y_train)
```

```
GridSearchCV(cv=5, error_score=nan,
             estimator=RandomForestRegressor(bootstrap=True, ccp_alpha=0.0,
                                             criterion='mse', 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=None,
                                             oob_score=False, random_state=None,
                                             verbose=0, warm_start=False),
```

사용한 모델

Random Forest Regressor

Decision Tree기반 Bagging을 활용한 앙상블 알고리즘

모델 선정 이유

예측의 변동성이 줄고, 과적합을 방지함

결측치에 대해 Robust한 학습이 가능하기 때문에, 데이터수가 적은 경우 효과적

Grid Search를 통한 최적의 parameter tuning

```
xgb_model = xgboost.XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=0.75, gamma=0,
             importance_type='gain', learning_rate=0.02, max_delta_step=0,
             max_depth=7, min_child_weight=1, missing=None, n_estimators=1000,
             n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
             reg_alpha=0.03, reg_lambda=1, scale_pos_weight=1, seed=None,
             silent=None, subsample=1, verbosity=1)
xgb_model.fit(x_train,y_train)
```

사용한 모델

XGBoost Regressor

Decision Tree기반 Boosting을 활용한 앙상블 알고리즘

모델 선정 이유

병렬 처리를 이용한 빠른 학습 속도와 우수한 회귀 성능

Greedy 알고리즘을 통한 prunning을 통한 과적합 방지

Grid Search를 통한 최적의 parameter tuning

06 출루율 예측 - 모델 선정 및 장타율 예측

RMSE	Linear Regression	SVM Regression	Random Forest	XGBoost Regressor	DNN Regression
1-df1	0.011179908	0.034197882	0.016365512	0.022348527	0.59161625
1-df2	0.010432962	0.031445665	0.016626596	0.01597909	0.118699831
1-df3	0.017801209	0.032180909	0.021511706	0.025755987	0.211677245
2-df1	0.010476095	0.038380662	0.029513569	0.017557252	0.24527692
2-df2	0.013337261	0.029608601	0.022173497	0.02413501	0.102014505
2-df3	0.016742267	0.042136354	0.022958638	0.018426288	0.082582563
3-df1	0.009785504	0.034604691	0.014482869	0.013204167	0.46162966
3-df2	0.012448365	0.03798404	0.017769258	0.014105301	0.056389797
3-df3	0.014435321	0.041499461	0.015701606	0.015025819	0.062156958

데이터셋 1은 후보군 2번

데이터셋 2,3번에는 후보군 1번이

가장 높은 성능

예측 모델로는

Linear Regression, Random Forest, XGBoost Regressor 사용

데이터셋 1,2,3 각각에 대한

성능이 우수한 3개 모델의 예측값

+ 21년 기록

+ 최근 30경기의

ensemble을 통해

최종 예측값 도출

```
obp = (obp1 + obp2 + obp3 + obp4 + obp5) / 5
obp
```

```
PCODE
76232    0.388486
68050    0.405424
75847    0.375414
67341    0.451350
79192    0.330667
78224    0.369018
78513    0.355794
76290    0.373556
79215    0.384098
67872    0.339398
dtype: float64
```

06 OPS 예측

```
ops = slg + obp
ops
```

```
PCODE
76232    0.918314
68050    0.939427
75847    0.916508
67341    0.982236
79192    0.800494
78224    0.863658
78513    0.804209
76290    0.835389
79215    0.828798
67872    0.771549
dtype: float64
```

출루율과 장타율의 합으로

OPS 예측

```
test['PCODE'] = ops.index
test['OPS'] = ops.values
test['장타율'] = slg.values
test['출루율'] = obp.values
test
```

	NO.	PCODE	OPS	장타율	출루율
1	1	76232	0.918314	0.529828	0.388486
2	2	68050	0.939427	0.534003	0.405424
3	3	75847	0.916508	0.541094	0.375414
4	4	67341	0.982236	0.530886	0.451350
5	5	79192	0.800494	0.469827	0.330667
6	6	78224	0.863658	0.494640	0.369018
7	7	78513	0.804209	0.448415	0.355794
8	8	76290	0.835389	0.461833	0.373556
9	9	79215	0.828798	0.444699	0.384098
10	10	67872	0.771549	0.432151	0.339398

최종 예측결과

팬, 미디어

과거 타자의 **타점, 홈런 수, 타율**을 통해 타자를 평가했다면, 최근 장타율과 출루율을 더한 **OPS** 정도는 일반 야구를 가볍게 시청하는 시청자들에게 익숙해졌으며 이에 더해 타자의 **WAR, WRC+**를 찾아보는 팬층도 증가하는 추세

타구 데이터에 대한 축적과 분석, 새로운 기준을 제공함으로써, 야구라는 스포츠를 즐기는데 있어서 **더 풍부한 경험**을 소비자들에게 제공

구단

한국 프로야구 만의 **배럴타구 기준의 선정**을 통해 더 다양한 방면에서 타자의 능력을 측정 가능

선수의 자유계약, 트레이드, 신인드래프트를 진행할 때 양질의 데이터 분석을 통해 **더 좋은 선수** 혹은 **저평가된 숨겨진 좋은 선수** 영입

타석 당 투구수, 배트 적극성%, 컨택%, 2S후커트%, 2S후선구%

STATIZ www.statiz.co.kr

savant Baseballsavant.mlb.com

www.fangraphs.com

Exit Velo Average, Pitch Velo Average, Barrels%