



# < 20/21 프리미어리그 경기 데이터를 활용한 축구선수 시장 가치 평가 >

빅파이 팀

2018606059 정수연

## 1. 주제 선정 배경

- ① 세계 축구 시장에 막대한 자본이 투입되면서 일부 선수들의 시장 가치 적합성에 대한 '거품' 논란 존재
- ② 축구선수의 본질인 '축구 실력'이 '시장 가치'에 얼마나 잘 반영되고 있는지 적합성을 평가하고자 함

## 2. 프로젝트 내용

- ① 2020/21시즌 프리미어리그 공격수 73명의 경기 기록과 시장 가치의 상관성 분석
- ② 경기 기록 변수( $x$ )와 시장 가치 변수( $y$ )의 적합도를 설명하는 회귀 예측 그래프 7가지 Case 표현
- ③ 'R-Square' (독립 변수가 종속 변수를 설명하는 비율)값이 가장 높은 회귀 예측 모형을 통해 선수들의 시장 가치 적합성 평가 (고평가, 저평가, 적정 평가)

### 3. 활용 데이터

- ① 프리미어리그 공식 홈페이지 - 2020/21 Player Stats  
(<https://www.premierleague.com/stats>)



- ② Transfer Market(트랜스퍼마켓) - Market Values  
(<https://www.transfermarkt.com/premier-league/marktwerte/wettbewerb/>)



- ③ WhoScored.com (후스코어드닷컴) - Statistics  
(<https://1xbet.whoscored.com/Statistics>)



- ④ kaggle  
- 'football\_epl\_20\_21.xlsx'  
- 'FPL 20 21.xlsx'



## 4. 분석 과정 및 코드 설명

# "FPL 20 21.xlsx" 파일(첫 번째 데이터)에서 필요한 열 가져오기

```
df1 = pd.read_excel("FPL 20 21.xlsx", engine = 'openpyxl', usecols = "A,B,C,E,F,H,K,L,M,P,T,U")
```

	First_Name	Second_Name	market_value	Goals_Scored	Assists	Minutes	Creativity	Influence	Threat	ICT_Index	Position	Rating
0	Bruno	Fernandes	NaN	18	14	3101	1414.9	1292.6	1253	396.2	MID	NaN
1	Harry	Kane	108.0	23	14	3083	659.1	1318.2	1585	355.9	FWD	7.79
2	Mohamed	Salah	90.0	22	6	3077	825.7	1056.0	1980	385.8	FWD	7.08
3	Heung-Min	Son	76.5	17	11	3119	1049.9	1052.2	1046	315.2	FWD	7.27
4	Patrick	Bamford	19.8	17	11	3052	371.0	867.2	1512	274.6	FWD	7.03
...	...	...	...	...	...	...	...	...	...	...	...	...
708	Harry	Wilson	NaN	0	0	0	0.0	0.0	0	0.0	MID	NaN
709	Ben	Woodburn	NaN	0	0	0	0.0	0.0	0	0.0	MID	NaN
710	Neil	Taylor	NaN	0	0	15	0.7	0.8	0	0.2	DEF	NaN
711	Alexandre	Jankewitz	NaN	0	0	2	0.3	4.4	0	0.5	MID	NaN
712	Kayne	Ramsay	NaN	0	0	90	0.0	13.2	0	1.3	DEF	NaN

# Position 열에서 공격수만 뽑기


```
df1=df1[df1['Position']=='FWD']
```

	First_Name	Second_Name	market_value	Goals_Scored	Assists	Minutes	Creativity	Influence	Threat	ICT_Index	Position	Rating
1	Harry	Kane	108.0	23	14	3083	659.1	1318.2	1585	355.9	FWD	7.79
2	Mohamed	Salah	90.0	22	6	3077	825.7	1056.0	1980	385.8	FWD	7.08
3	Heung-Min	Son	76.5	17	11	3119	1049.9	1052.2	1046	315.2	FWD	7.27
4	Patrick	Bamford	19.8	17	11	3052	371.0	867.2	1512	274.6	FWD	7.03
5	Jamie	Vardy	7.0	15	14	2838	356.6	819.2	1306	248.3	FWD	7.11
...	...	...	...	...	...	...	...	...	...	...	...	...
593	Max	Thompson	NaN	0	0	0	0.0	0.0	0	0.0	FWD	NaN
594	Indiana	Vassilev	NaN	0	0	0	0.0	0.0	0	0.0	FWD	NaN
595	Connor	Wickham	NaN	0	0	0	0.0	0.0	0	0.0	FWD	NaN
596	Kenneth	Zohore	NaN	0	0	0	0.0	0.0	0	0.0	FWD	NaN
655	Harvey	Elliott	13.5	0	0	0	0.0	0.0	0	0.0	FWD	NaN

## 4. 분석 과정 및 코드 설명

# Rating에서 NaN값있는 행 없애기

df1=df1.dropna()

Rating		First_Name	Second_Name	market_value	Goals_Scored	Assists	Minutes	Creativity	Influence	Threat	ICT_Index	Position	Rating	
7.79		1	Harry	Kane	108.0	23	14	3083	659.1	1318.2	1585	355.9	FWD	7.79
7.08		2	Mohamed	Salah	90.0	22	6	3077	825.7	1056.0	1980	385.8	FWD	7.08
7.27		3	Heung-Min	Son	76.5	17	11	3119	1049.9	1052.2	1046	315.2	FWD	7.27
7.03		4	Patrick	Bamford	19.8	17	11	3052	371.0	867.2	1512	274.6	FWD	7.03
7.11		5	Jamie	Vardy	7.0	15	14	2838	356.6	819.2	1306	248.3	FWD	7.11
...		...	...	...	...	...	...	...	...	...	...	...	...	...
NaN		323	Edward	Nketiah	10.8	2	1	411	97.3	116.6	211	42.1	FWD	6.72
NaN		324	Gabriel Teodoro	Martinelli Silva	19.8	2	1	582	133.8	142.6	300	57.5	FWD	7.36
NaN		419	Divock	Origi	10.8	0	0	180	33.2	17.8	60	11.1	FWD	6.69
NaN		432	Amad	Diallo	16.2	0	1	165	31.2	38.0	9	7.8	FWD	7.17
NaN	508	Demarai	Gray	13.2	0	0	18	1.3	0.2	3	0.5	FWD	6.96	

# market\_value 내림차순으로 정렬

df1 = df1.sort\_values(by = ['market\_value'],ascending=False)

	First_Name	Second_Name	market_value	Goals_Scored	Assists	Minutes	Creativity	Influence	Threat	ICT_Index	Position	Rating
1	Harry	Kane	108.0	23	14	3083	659.1	1318.2	1585	355.9	FWD	7.79
42	Jack	Grealish	90.0	6	12	2183	1029.6	707.4	917	265.3	FWD	7.56
2	Mohamed	Salah	90.0	22	6	3077	825.7	1056.0	1980	385.8	FWD	7.08
19	Raheem	Sterling	81.0	10	9	2534	584.1	663.4	1366	260.3	FWD	7.37
3	Heung-Min	Son	76.5	17	11	3119	1049.9	1052.2	1046	315.2	FWD	7.27
...	...	...	...	...	...	...	...	...	...	...	...	...
275	Olivier	Giroud	4.0	4	0	740	112.0	161.4	403	67.2	FWD	6.52
255	Ashley	Barnes	3.5	3	0	1324	243.6	153.8	548	93.8	FWD	6.47
240	Mbaye	Diagne	3.0	3	5	1188	130.8	180.0	346	65.2	FWD	6.68
225	Matej	Vydra	3.0	3	2	1359	194.0	197.4	491	87.3	FWD	6.36
117	David	McGoldrick	1.2	8	1	2391	320.2	464.4	860	164.6	FWD	6.72



## 4. 분석 과정 및 코드 설명

# index 고치기

```
df1 = df1.reset_index(drop=True)
```

	First_Name	Second_Name	market_value	Goals_Scored	Assists	Minutes	Creativity	Influence	Threat	ICT_Index	Position	Rating
0	Harry	Kane	108.0	23	14	3083	659.1	1318.2	1585	355.9	FWD	7.79
1	Jack	Grealish	90.0	6	12	2183	1029.6	707.4	917	265.3	FWD	7.56
2	Mohamed	Salah	90.0	22	6	3077	825.7	1056.0	1980	385.8	FWD	7.08
3	Raheem	Sterling	81.0	10	9	2534	584.1	663.4	1366	260.3	FWD	7.37
4	Heung-Min	Son	76.5	17	11	3119	1049.9	1052.2	1046	315.2	FWD	7.27
...	...	...	...	...	...	...	...	...	...	...	...	...
73	Olivier	Giroud	4.0	4	0	740	112.0	161.4	403	67.2	FWD	6.52
74	Ashley	Barnes	3.5	3	0	1324	243.6	153.8	548	93.8	FWD	6.47
75	Mbaye	Diagne	3.0	3	5	1188	130.8	180.0	346	65.2	FWD	6.68
76	Matej	Vydra	3.0	3	2	1359	194.0	197.4	491	87.3	FWD	6.36
77	David	McGoldrick	1.2	8	1	2391	320.2	464.4	860	164.6	FWD	6.72

# 성, 이름 합치기 : 'Name' = 'First\_Name' + 'Second\_Name'

```
df1.insert(0, 'Name', df1['First_Name'].map(str)+' '+df1['Second_Name'])
```

	Name	First_Name	Second_Name	market_value	Goals_Scored	Assists	Minutes	Creativity	Influence	Threat	ICT_Index	Position	Rating
0	Harry Kane	Harry	Kane	108.0	23	14	3083	659.1	1318.2	1585	355.9	FWD	7.79
1	Jack Grealish	Jack	Grealish	90.0	6	12	2183	1029.6	707.4	917	265.3	FWD	7.56
2	Mohamed Salah	Mohamed	Salah	90.0	22	6	3077	825.7	1056.0	1980	385.8	FWD	7.08
3	Raheem Sterling	Raheem	Sterling	81.0	10	9	2534	584.1	663.4	1366	260.3	FWD	7.37
4	Heung-Min Son	Heung-Min	Son	76.5	17	11	3119	1049.9	1052.2	1046	315.2	FWD	7.27
...	...	...	...	...	...	...	...	...	...	...	...	...	...
73	Olivier Giroud	Olivier	Giroud	4.0	4	0	740	112.0	161.4	403	67.2	FWD	6.52
74	Ashley Barnes	Ashley	Barnes	3.5	3	0	1324	243.6	153.8	548	93.8	FWD	6.47
75	Mbaye Diagne	Mbaye	Diagne	3.0	3	5	1188	130.8	180.0	346	65.2	FWD	6.68
76	Matej Vydra	Matej	Vydra	3.0	3	2	1359	194.0	197.4	491	87.3	FWD	6.36
77	David McGoldrick	David	McGoldrick	1.2	8	1	2391	320.2	464.4	860	164.6	FWD	6.72



## 4. 분석 과정 및 코드 설명

# 'football\_epl\_20\_21'(두 번째 데이터)에서 필요한 열 가져오기

```
df2 = pd.read_excel("football_epl_20_21.xlsx", engine = 'openpyxl', usecols = "A,E,K,L,O,P")
```

	Name	Age	Passes_Attempted	Perc_Passes_Completed	xG	xA
0	Pedro Lomba Neto	20	1212	78.8	0.17	0.22
1	Fabio Silva	18	305	74.4	0.40	0.04
2	Raul Jimenez	29	263	78.7	0.26	0.08
3	Adama Traore	24	879	65.9	0.08	0.18
4	Willian Jos채	28	306	81.4	0.15	0.05
...	...	...	...	...	...	...
527	Bukayo Saka	18	1155	74.9	0.24	0.17
528	Gabriel Teodoro Martinelli Silva	19	159	79.2	0.53	0.32
529	Edward Nketiah	21	89	82.0	0.52	0.18
530	Willian	31	787	79.3	0.13	0.16
531	Reiss Nelson	20	30	56.7	0.06	0.00

# Name으로 첫 번째, 두 번째 데이터 inner join

```
join1 = pd.merge(left = df1, right = df2, how = "inner", on = "Name" )
```

	Name	First_Name	Second_Name	market_value	Goals_Scored	Assists	Minutes	Creativity	Influence	Threat	ICT_Index	Position	Rating	Age	Passes_Attempted	Perc_Passes_Completed	xG	xA
0	Harry Kane	Harry	Kane	108.0	23	14	3083	659.1	1318.2	1585	355.9	FWD	7.79	27	937	70.1	0.60	0.22
1	Jack Grealish	Jack	Grealish	90.0	6	12	2183	1029.6	707.4	917	265.3	FWD	7.56	24	1100	78.5	0.18	0.35
2	Mohamed Salah	Mohamed	Salah	90.0	22	6	3077	825.7	1056.0	1980	385.8	FWD	7.08	28	1288	83.2	0.61	0.18
3	Raheem Sterling	Raheem	Sterling	81.0	10	9	2534	584.1	663.4	1366	260.3	FWD	7.37	25	1127	85.4	0.43	0.17
4	Heung-Min Son	Heung-Min	Son	76.5	17	11	3119	1049.9	1052.2	1046	315.2	FWD	7.27	28	1199	76.7	0.30	0.26
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
67	Karlhan Grant	Karlhan	Grant	4.0	1	0	1120	63.8	93.0	259	41.2	FWD	6.15	22	228	73.2	0.17	0.00
68	Olivier Giroud	Olivier	Giroud	4.0	4	0	740	112.0	161.4	403	67.2	FWD	6.52	33	217	74.2	0.58	0.09
69	Ashley Barnes	Ashley	Barnes	3.5	3	0	1324	243.6	153.8	548	93.8	FWD	6.47	30	307	63.5	0.35	0.04
70	Mbaye Diagne	Mbaye	Diagne	3.0	3	5	1188	130.8	180.0	346	65.2	FWD	6.68	28	230	70.4	0.38	0.13
71	David McGoldrick	David	McGoldrick	1.2	8	1	2391	320.2	464.4	860	164.6	FWD	6.72	32	938	74.5	0.26	0.05

## 4. 분석 과정 및 코드 설명

# EPL 공식 홈페이지 크롤링

# shots 페이지로 이동하기

From selenium import webdriver

from webdriver\_manager.chrome import ChromeDriverManager

from time import sleep

from bs4 import BeautifulSoup

import pandas as pd

chrome\_options = webdriver.ChromeOptions

chrome\_options.add\_argument('--start-maximized')

# 전체 화면으로 크롬 브라우저 실행

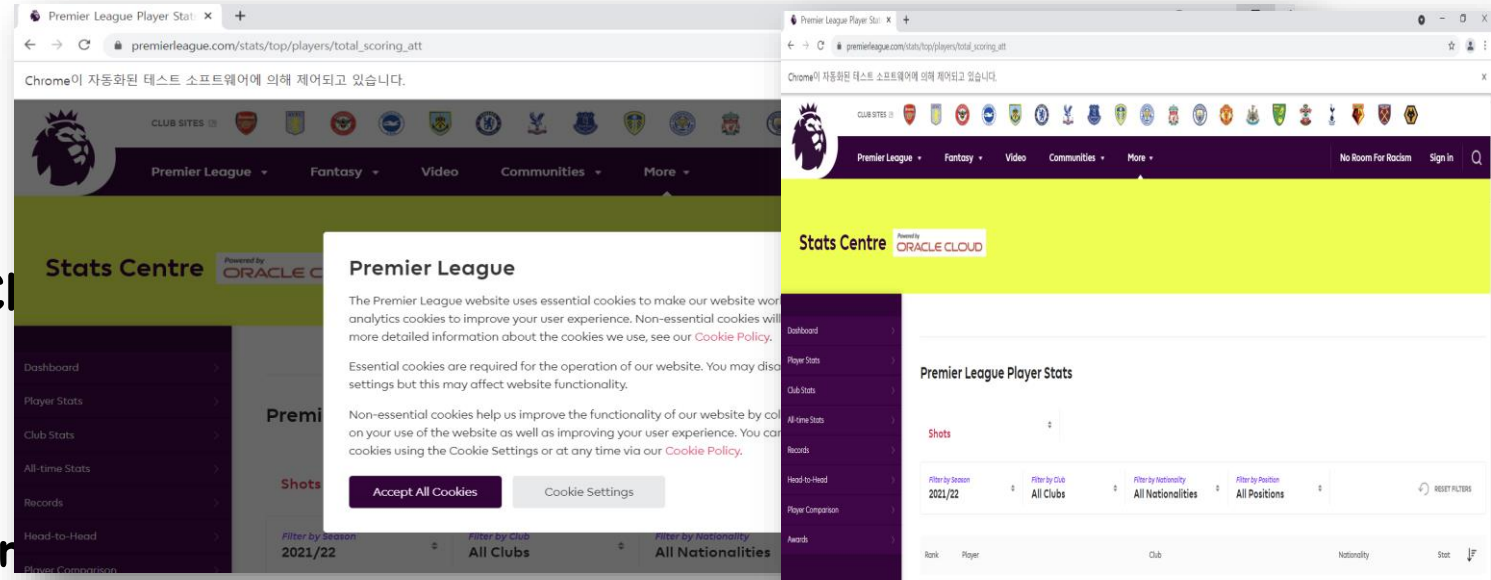
driver = webdriver.Chrome(ChromeDriverManager().install(),chrome\_options=chrome\_options)

driver.get('https://www.premierleague.com/stats/top/players/total\_scoring\_att')

# popup 해결하기

driver.find\_element\_by\_xpath('/html/body/div[1]/div/div[1]/div[5]/button[1]').click()

sleep(5)





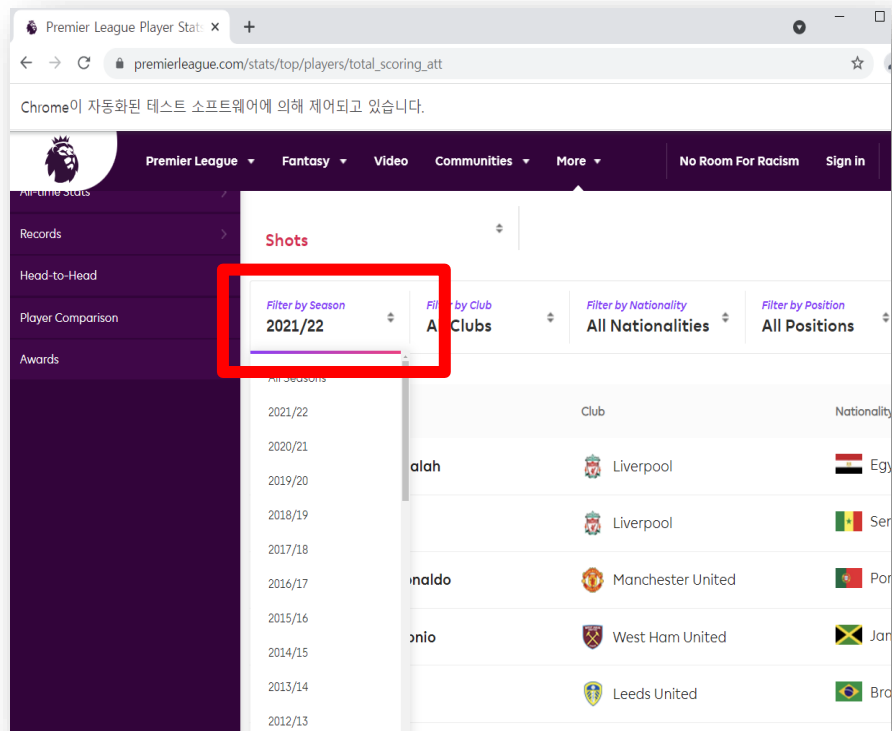
## 4. 분석 과정 및 코드 설명

### # filter by season 클릭하기

```
driver.find_element_by_xpath('//*[@id="mainContent"]/div[2]/div/div[2]/div[1]/section/div[1]/div[2]').click()
sleep(5)
```

### # 2020/21 시즌 클릭하기

```
driver.find_element_by_xpath('//*[@id="mainContent"]/div[2]/div/div[2]/div[1]/section/div[1]/ul/li[3]').click()
sleep(5)
```



The screenshot shows the Premier League Player Stats page. The 'Shots' filter is active, and the 'Filter by Season' dropdown is open, showing the '2020/21' season selected. The dropdown is highlighted with a red box.

Rank	Player
1.	Harry Kane
2.	Mohamed Salah
3.	Bruno Fernandes
4.	Patrick Bamford
5.	Ollie Watkins
6.	Sadio Mané

## 4. 분석 과정 및 코드 설명

### # 다음 페이지로 넘기기

```
result=[]
```

```
for i in range(20):
```

```
    name=driver.find_elements_by_css_selector("tr td:nth-child(1)")
    shoot=driver.find_elements_by_css_selector("tr td:nth-child(2)")
    player_name=name[i].text
    player_shoot=shoot[i].text
    result.append({"Name":player_name,"Shots":player_shoot})
```

```
for j in range(11):
```

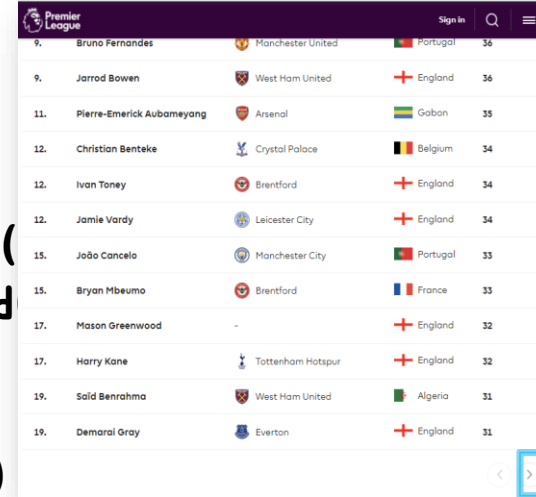
```
    driver.find_element_by_xpath('//*[@id="mainContent"]/div[2]/div/div[2]/div[1]/div[3]/div[2]').click()
```

```
    sleep(2)
```

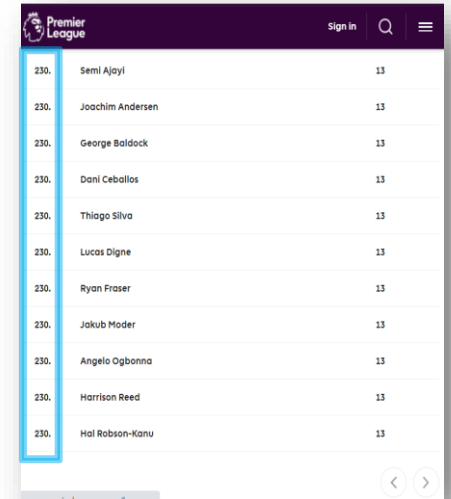
```
    for i in range(20):
```

```
        name=driver.find_elements_by_css_selector("tr td:nth-child(2)")
        shoot=driver.find_elements_by_css_selector("tr td:nth-child(5)")
        player_name=name[i].text
        player_shoot=shoot[i].text
        result.append({"Name":player_name,"Shots":player_shoot})
```

빅데이터 언어\_기말 프로젝트\_빅파이



	Player	Team	Shots
9.	Bruno Fernandes	Manchester United	36
9.	Jarrod Bowen	West Ham United	36
11.	Pierre-Emerick Aubameyang	Arsenal	35
12.	Christian Benteke	Crystal Palace	34
12.	Ivan Toney	Brentford	34
12.	Jamie Vardy	Leicester City	34
15.	João Cancelo	Manchester City	33
15.	Bryan Mbeumo	Brentford	33
17.	Mason Greenwood	-	32
17.	Harry Kane	Tottenham Hotspur	32
19.	Said Benrahma	West Ham United	31
19.	Demarai Gray	Everton	31



230.	Semi Ajayi	13
230.	Joachim Andersen	13
230.	George Baldock	13
230.	Dani Ceballos	13
230.	Thiago Silva	13
230.	Lucas Digne	13
230.	Ryan Fraser	13
230.	Jakub Moder	13
230.	Angelo Ogbonna	13
230.	Harrison Reed	13
230.	Hai Robson-Kanu	13

## 4. 분석 과정 및 코드 설명

# df3에 Shots 데이터 넣기

```
df3=pd.DataFrame(result)
```

```
df3
```

	Name	Shots
0	Harry Kane	137
1	Mohamed Salah	126
2	Bruno Fernandes	121
3	Patrick Bamford	107
4	Ollie Watkins	97
...	...	...
235	Ryan Fraser	13
236	Jakub Moder	13
237	Angelo Ogbonna	13
238	Harrison Reed	13
239	Hal Robson-Kanu	13

# join1에 df3 데이터 merge하기

```
join2=pd.merge(left = join1, right = df3, how = "left", on = "Name" )
```

```
join2
```

	Name	First_Name	Second_Name	market_value	Goals_Scored	Assists	Minutes	Creativity	Influence	Threat	ICT_Index	Position	Rating	Age	Passes_Attempted	Perc_Passes_Completed	xG	xA	Shots
0	Harry Kane	Harry	Kane	108.0	23	14	3083	659.1	1318.2	1585	355.9	FWD	7.79	27	937	70.1	0.60	0.22	137
1	Jack Grealish	Jack	Grealish	90.0	6	12	2183	1029.6	707.4	917	265.3	FWD	7.56	24	1100	78.5	0.18	0.35	50
2	Mohamed Salah	Mohamed	Salah	90.0	22	6	3077	825.7	1056.0	1980	385.8	FWD	7.08	28	1288	83.2	0.61	0.18	126
3	Raheem Sterling	Raheem	Sterling	81.0	10	9	2534	584.1	663.4	1366	260.3	FWD	7.37	25	1127	85.4	0.43	0.17	70
4	Heung-Min Son	Heung-Min	Son	76.5	17	11	3119	1049.9	1052.2	1046	315.2	FWD	7.27	28	1199	76.7	0.30	0.26	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
67	Karlan Grant	Karlan	Grant	4.0	1	0	1120	63.8	93.0	259	41.2	FWD	6.15	22	228	73.2	0.17	0.00	16
68	Olivier Giroud	Olivier	Giroud	4.0	4	0	740	112.0	161.4	403	67.2	FWD	6.52	33	217	74.2	0.58	0.09	29
69	Ashley Barnes	Ashley	Barnes	3.5	3	0	1324	243.6	153.8	548	93.8	FWD	6.47	30	307	63.5	0.35	0.04	38
70	Mbaye Diagne	Mbaye	Diagne	3.0	3	5	1188	130.8	180.0	346	65.2	FWD	6.68	28	230	70.4	0.38	0.13	25
71	David McGoldrick	David	McGoldrick	1.2	8	1	2391	320.2	464.4	860	164.6	FWD	6.72	32	938	74.5	0.26	0.05	71

## 4. 분석 과정 및 코드 설명

### # Shots에 NaN값 채워넣기

```
join2.loc[join2.Name == 'Heung-Min Son', ('Shots')] = 68
join2.loc[join2.Name == 'Sadio Mane', ('Shots')] = 94
join2.loc[join2.Name == 'Gabriel Fernando de Jesus', ('Shots')] = 55
join2.loc[join2.Name == 'Pedro Lomba Neto', ('Shots')] = 61
join2.loc[join2.Name == 'Adama Traore', ('Shots')] = 43
join2.loc[join2.Name == 'Daniel Castelo Podence', ('Shots')] = 32
join2.loc[join2.Name == 'Said Benrahma', ('Shots')] = 35
join2.loc[join2.Name == 'Fabio Silva', ('Shots')] = 48
join2.loc[join2.Name == 'Gabriel Teodoro Martinelli Silva', ('Shots')] = 20
join2.loc[join2.Name == 'Bertrand Traore', ('Shots')] = 64
join2.loc[join2.Name == 'Rodrigo Moreno', ('Shots')] = 46
join2.loc[join2.Name == 'Ayoze Perez', ('Shots')] = 31
join2.loc[join2.Name == 'Amad Diallo', ('Shots')] = 1
join2.loc[join2.Name == 'Joelinton de Lira', ('Shots')] = 44
join2.loc[join2.Name == 'Demarai Gray', ('Shots')] = 1
join2.loc[join2.Name == 'Edward Nketiah', ('Shots')] = 12
join2.loc[join2.Name == 'Divock Origi', ('Shots')] = 7
join2.loc[join2.Name == 'Michy Batshuayi', ('Shots')] = 12
```

### # 데이터 타입 확인

join2.info()

Data columns (total 19 columns):

#	Column	Non-Null	Count	Dtype
0	Name	72	non-null	object
1	First_Name	72	non-null	object
2	Second_Name	72	non-null	object
3	market_value	72	non-null	float64
4	Goals_Scored	72	non-null	int64
5	Assists	72	non-null	int64
6	Minutes	72	non-null	int64
7	Creativity	72	non-null	float64
8	Influence	72	non-null	float64
9	Threat	72	non-null	int64
10	ICT_Index	72	non-null	float64
11	Position	72	non-null	object
12	Rating	72	non-null	float64
13	Age	72	non-null	int64
14	Passes_Attempted	72	non-null	int64
15	Perc_Passes_Completed	72	non-null	float64
16	xG	72	non-null	float64
17	xA	72	non-null	float64
18	Shots	72	non-null	object

dtypes: float64(8), int64(6), object(5)

### # Shots 데이터 int형 타입으로 변환

```
join2["Shots"] = join2['Shots'].astype(int)
```

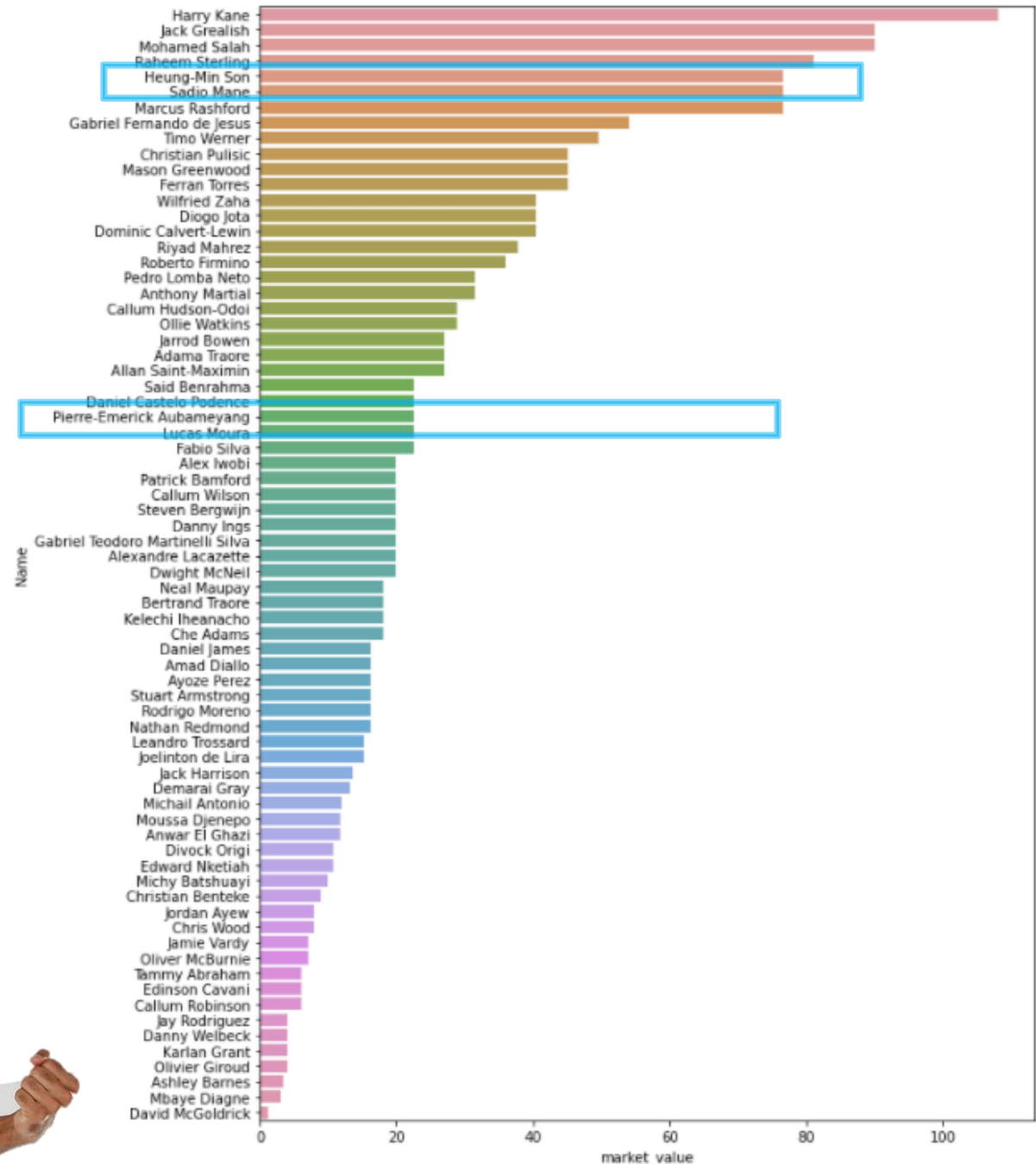
## 4. 분석 과정 및 코드 설명

#market\_value 기준 선수 시장 가치 그래프 표현

```
import statsmodels.api as sm
import statsmodels.formula.api as smf
from sklearn.metrics import r2_score
```

```
plt.figure(figsize=[10,15])
sns.barplot(data=join2, x='market_value', y='Name')
```

빅데이터 언어\_기말 프로젝트\_빅파이



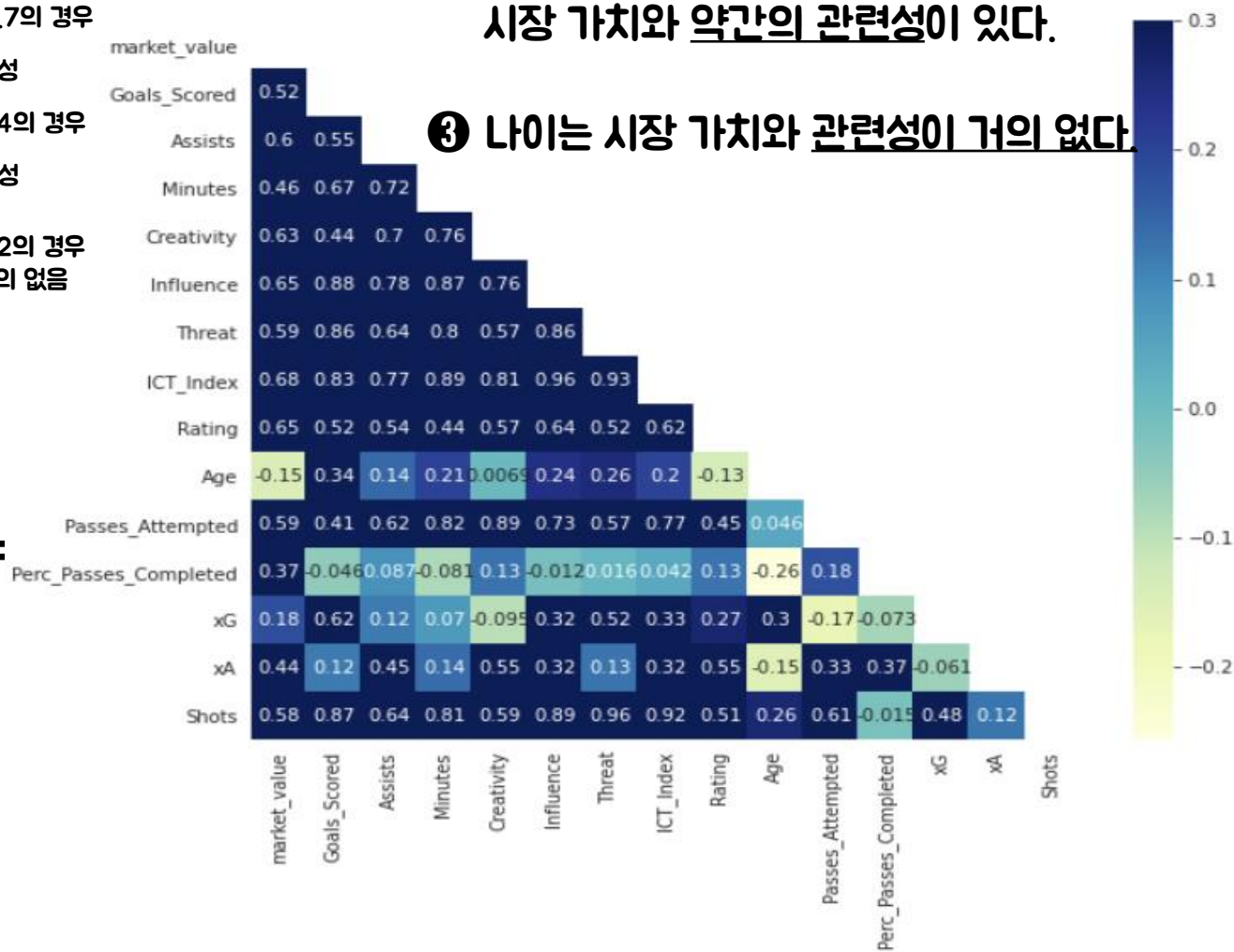


## 빅데이터 언어\_기말 프로젝트\_빅파이

\* 상관계수  $r$   
두 변수 사이의 일차적인 관계가  
얼마나 강한지를 측정해주는 지수

- $r = 0.7 \sim 1.0$ 의 경우
  - : 매우 강한 관련성
- $r = 0.4 \sim 0.7$ 의 경우
  - : 상당한 관련성
- $r = 0.2 \sim 0.4$ 의 경우
  - : 약간의 관련성
- $r = 0.0 \sim 0.2$ 의 경우
  - : 관련성이 거의 없음

- ① **특점, 어시스트, 경기시간, 창의성, 영향력, 위협, ICT지수, 평점, 패스시도, 기대 어시스트는 시장 가치와 상당한 관련성이 있다.**
- ② **패스성공률, 기대 특점은 시장 가치와 약간의 관련성이 있다.**
- ③ **나이는 시장 가치와 관련성이 거의 없다.**



## 4. 분석 과정 및 코드 설명

### #회귀 그래프 그리기

Dep. Variable:	y	<b>1</b>	R-squared:	0.396
Model:	OLS		Adj. R-squared:	0.387
Method:	Least Squares		F-statistic:	45.88
Date:	Wed, 08 Dec 2021		Prob (F-statistic):	3.22e-09
Time:	18:43:53		Log-Likelihood:	-310.14
No. Observations:	72		AIC:	624.3
Df Residuals:	70		BIC:	628.8
Df Model:	1			

Covariance Type:	nonrobust
------------------	-----------

	coef	std err	t	<b>2</b> P> t	[0.025	0.975]
Intercept	4.5966	3.832	1.199	0.234	-3.047	12.240
x	1.8833	0.278	6.774	<b>0.000</b>	1.329	2.438

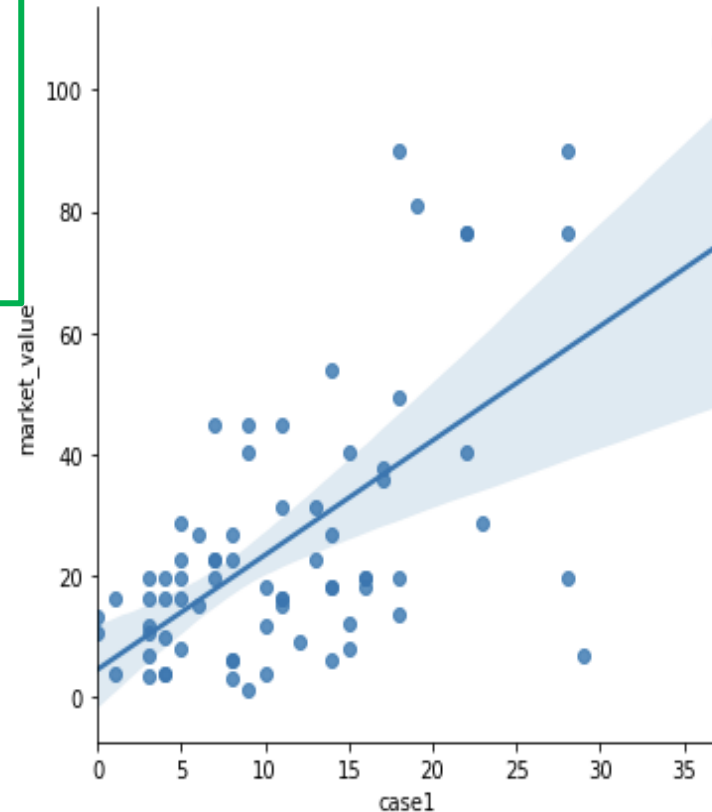
Omnibus:	2.605	Durbin-Watson:	0.834
Prob(Omnibus):	0.272	Jarque-Bera (JB):	1.916
Skew:	0.225	Prob(JB):	0.384
Kurtosis:	3.661	Cond. No.	24.7

#### \*결정계수 (= R-squared)

독립 변수가 종속 변수를 설명하는 비율을 뜻하며, 1에 가까울수록 회귀선이 표본을 설명하는 데 유용함. 실무에서 0.3이상일 경우 '의미가 있다'고 해석 가능.

**① R-squared = 0.396으로 '공격포인트'는 '시장가치'를 39.6%만큼 설명한다.**

**② 유의확률이 0.05보다 작으므로 '공격포인트'가 '시장가치'에 유의하게 영향을 미친다.**



```
R1=result.rsquared
result.summary()
```

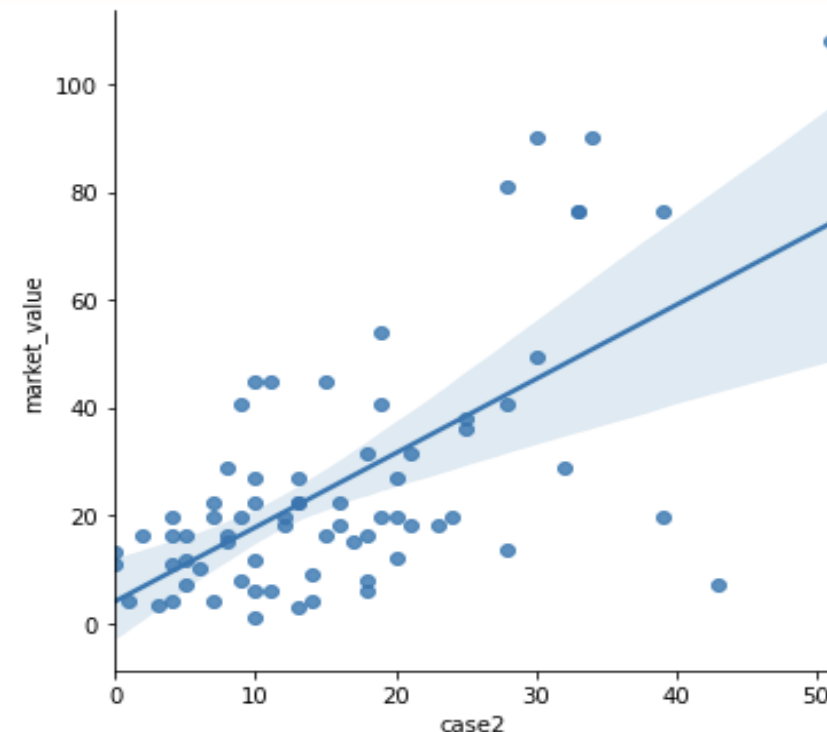
## 4. 분석 과정 및 코드 설명

### OLS Regression Results

Dep. Variable:	y	1	R-squared:	0.417		
Model:	OLS		Adj. R-squared:	0.409		
Method:	Least Squares		F-statistic:	50.04		
Date:	Wed, 08 Dec 2021		Prob (F-statistic):	9.15e-10		
Time:	18:54:38		Log-Likelihood:	-308.87		
No. Observations:	72		AIC:	621.7		
Df Residuals:	70		BIC:	626.3		
Df Model:	1					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
Intercept	4.0124	3.768	1.065	0.291	-3.503	11.528
x	1.3755	0.194	7.074	0.000	0.988	1.763
Omnibus:	2.828	Durbin-Watson:	0.856			
Prob(Omnibus):	0.243	Jarque-Bera (JB):	2.479			
Skew:	0.087	Prob(JB):	0.290			
Kurtosis:	3.892	Cond. No.	34.7			

① R-squared = 0.4170이므로  
'득점+2\*어시스트'는 '시장 가치'를  
41.7%만큼 설명한다.

② 유의확률이 0.05보다 작으므로  
'득점+2\*어시스트'가 '시장가치'에  
유의하게 영향을 미친다.



## 4. 분석 과정 및 코드 설명

### OLS Regression Results

Dep. Variable:	y ①	R-squared:	0.358
Model:	OLS	Adj. R-squared:	0.348
Method:	Least Squares	F-statistic:	38.98
Date:	Wed, 08 Dec 2021	Prob (F-statistic):	2.90e-08
Time:	18:56:05	Log-Likelihood:	-312.35
No. Observations:	72	AIC:	628.7
Df Residuals:	70	BIC:	633.3
Df Model:	1		
Covariance Type:	nonrobust		

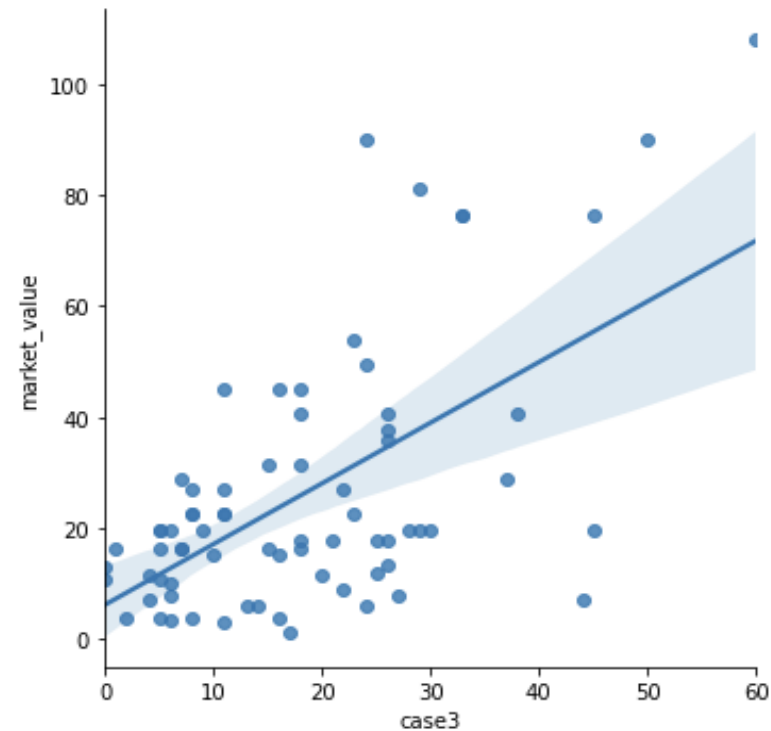
	coef	std err	t	P> t	[0.025	0.975]
Intercept	6.2359	3.876	1.609	0.112	-1.495	13.967
x	1.0916	0.175	6.243	0.000	0.743	1.440

Omnibus:	4.444	Durbin-Watson:	0.776
Prob(Omnibus):	0.108	Jarque-Bera (JB):	3.598
Skew:	0.448	Prob(JB):	0.165
Kurtosis:	3.631	Cond. No.	38.9

join2['Assists']

① R-squared = 0.3580이므로  
'2\*득점+어시스트'는 '시장 가치'를  
35.8%만큼 설명한다.

② 유의확률이 0.05보다 작으므로  
'2\*득점+어시스트'가 '시장가치'에  
유의하게 영향을 미친다.



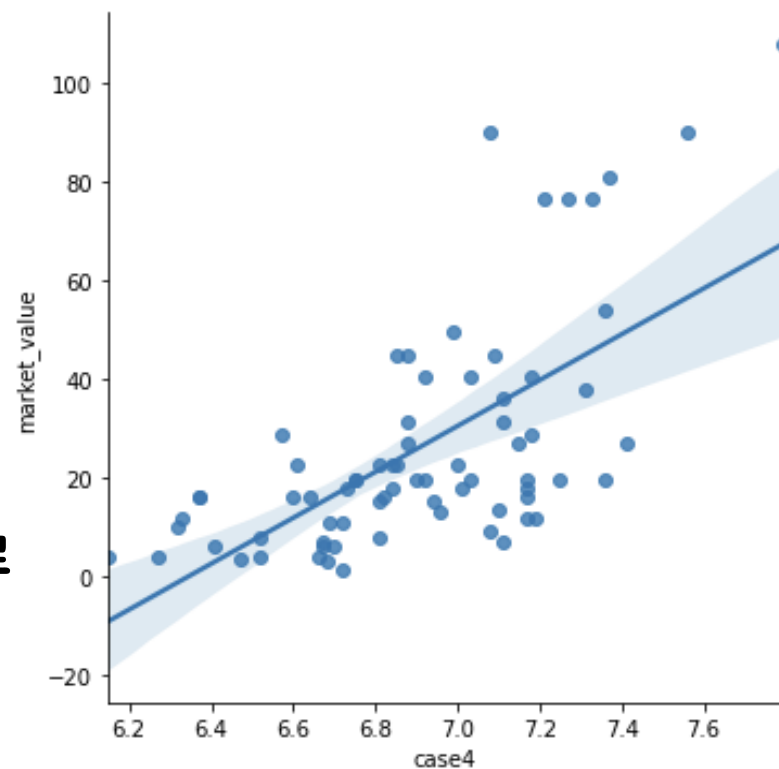
## 4. 분석 과정 및 코드 설명

### OLS Regression Results

Dep. Variable:	y	R-squared:	0.422			
Model:	OLS	Adj. R-squared:	0.414			
Method:	Least Squares	F-statistic:	51.19			
Date:	Wed, 08 Dec 2021	Prob (F-statistic):	6.52e-10			
Time:	18:58:21	Log-Likelihood:	-308.53			
No. Observations:	72	AIC:	621.1			
Df Residuals:	70	BIC:	625.6			
Df Model:	1					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
Intercept	-295.2717	44.966	-6.566	0.000	-384.954	-205.589
x	46.5434	6.505	7.155	0.000	33.569	59.518
Omnibus:	7.964	Durbin-Watson:	0.751			
Prob(Omnibus):	0.019	Jarque-Bera (JB):	7.412			
Skew:	0.747	Prob(JB):	0.0246			
Kurtosis:	3.485	Cond. No.	151.			

① R-squared = 0.4220이므로  
'평점'은 '시장 가치'를  
42.2%만큼 설명한다.

② 유의확률이 0.05보다 작으므로  
'평점'이 '시장가치'에  
유의하게 영향을 미친다.





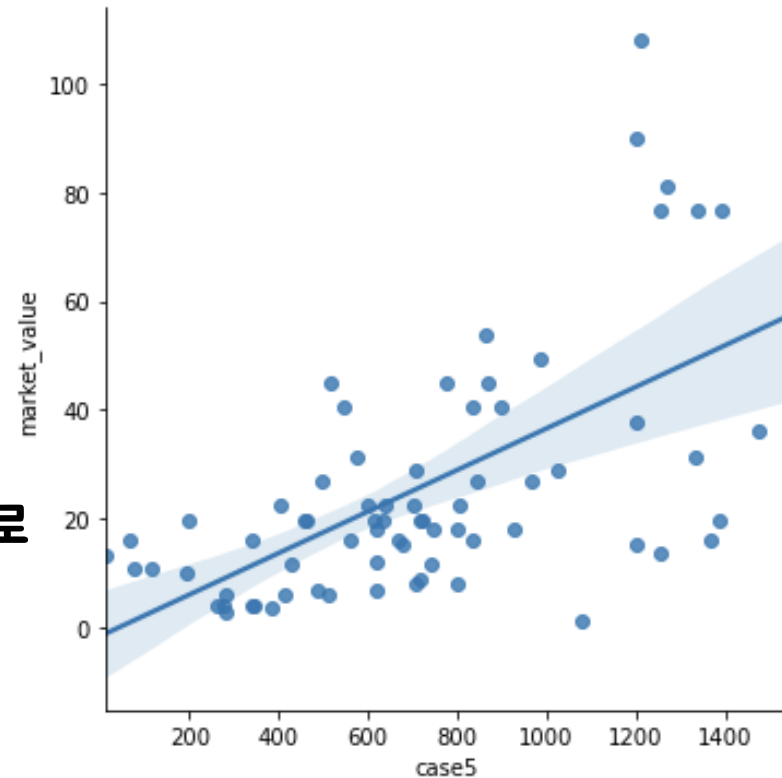
## 4. 분석 과정 및 코드 설명

### OLS Regression Results

Dep. Variable:	y	R-squared:	0.380			
Model:	OLS	Adj. R-squared:	0.371			
Method:	Least Squares	F-statistic:	42.87			
Date:	Wed, 08 Dec 2021	Prob (F-statistic):	8.25e-09			
Time:	18:59:33	Log-Likelihood:	-311.09			
No. Observations:	72	AIC:	626.2			
Df Residuals:	70	BIC:	630.7			
Df Model:	1					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
Intercept	-1.5679	4.753	-0.330	0.742	-11.047	7.911
x	0.0382	0.006	6.548	0.000	0.027	0.050
Omnibus:	9.949	Durbin-Watson:	0.623			
Prob(Omnibus):	0.007	Jarque-Bera (JB):	10.469			
Skew:	0.706	Prob(JB):	0.00533			
Kurtosis:	4.224	Cond. No.	1.78e+03			

① R-squared = 0.3800이므로  
'패스+2\*슈팅'은 '시장 가치'를  
38.0%만큼 설명한다.

② 유의확률이 0.05보다 작으므로  
'패스+2\*슈팅'이 '시장가치'에  
유의하게 영향을 미친다.



## 4. 분석 과정 및 코드 설명

# OLS Regression Results

Dep. Variable: y ① R-squared: 0.466 (treat 결함 값)

Model: OLS Adj. R-squared: 0.458

Method: Least Squares F-statistic: 61.07

Date: Fri, 10 Dec 2021 Prob (F-statistic): 4.01e-11

Time: 17:56:05 Log-Likelihood: -305.71

No. Observations: 72 AIC: 615.4

Df Residuals: 70 BIC: 620.0

Df Model: 1

Covariance Type: nonrobust

②

coef std err t P>|t| [0.025 0.975]

Intercept -2.9037 4.225 -0.687 0.494 -11.330 5.522

x 0.1868 0.024 7.815 0.000 0.139 0.234

Omnibus: 2.045 Durbin-Watson: 0.969

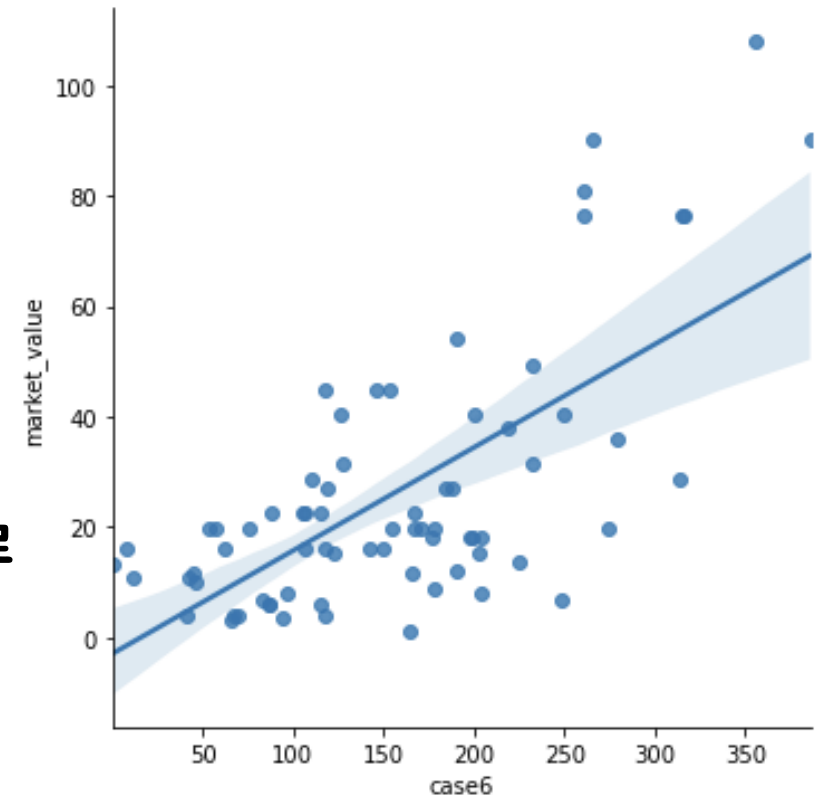
Prob(Omnibus): 0.360 Jarque-Bera (JB): 1.766

Skew: 0.383 Prob(JB): 0.414

Kurtosis: 2.955 Cond. No. 370.

① R-squared = 0.466이므로  
'ICT Index'은 '시장 가치'를  
46.6%만큼 설명한다.

② 유의확률이 0.05보다 작으므로  
'ICT Index'가 '시장가치'에  
유의하게 영향을 미친다.



## 4. 분석 과정 및 코드 설명

### OLS Regression Results

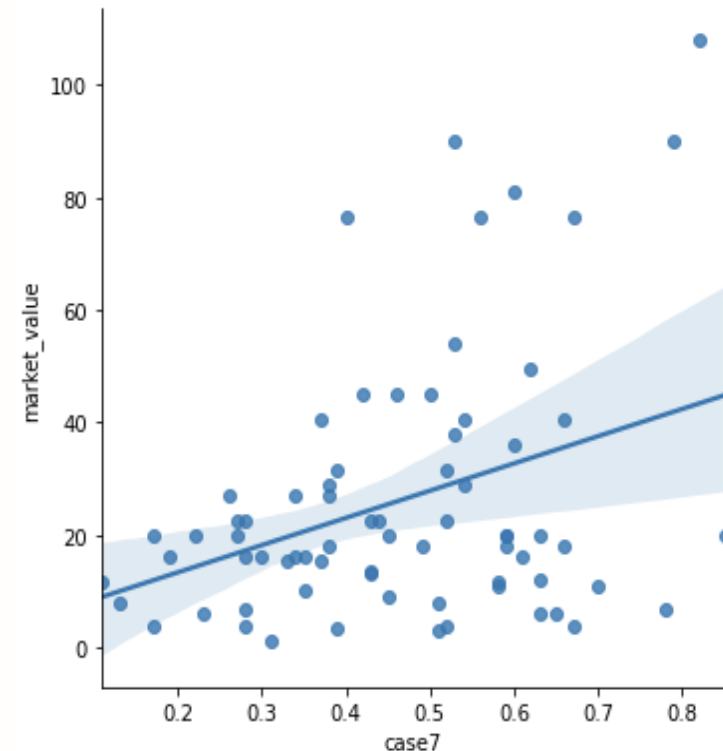
Dep. Variable:	1	R-squared:	0.125
Model:	OLS	Adj. R-squared:	0.113
Method:	Least Squares	F-statistic:	10.04
Date:	Wed, 08 Dec 2021	Prob (F-statistic):	0.00228
Time:	19:03:01	Log-Likelihood:	-323.46
No. Observations:	72	AIC:	650.9
Df Residuals:	70	BIC:	655.5
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
Intercept	3.6940	7.529	0.491	0.625	-11.322	18.710
x	48.3958	15.276	3.168	0.002	17.928	78.863

Omnibus:	16.272	Durbin-Watson:	0.310
Prob(Omnibus):	0.000	Jarque-Bera (JB):	18.759
Skew:	1.115	Prob(JB):	8.45e-05
Kurtosis:	4.130	Cond. No.	7.21

① R-squared = 0.125이므로  
'기대 득점+기대 어시스트'는  
'시장 가치'를 12.5%만큼 설명한다.

② 유의확률이 0.05보다 작으므로  
'xG+xA'가 '시장가치'에  
유의하게 영향을 미친다.



## # 객관적 지표 중 R제곱 값이 가장 높은 case2의 비선형그래프 그려보기

```
import statsmodels.api as sm
import statsmodels.formula.api as smf
from sklearn.metrics import r2_score
from matplotlib import style
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import PolynomialFeatures
```

```
plt.bar(x, Rsq,
x=join2['case2'][:, np.newaxis]
y=join2['market_value']
```

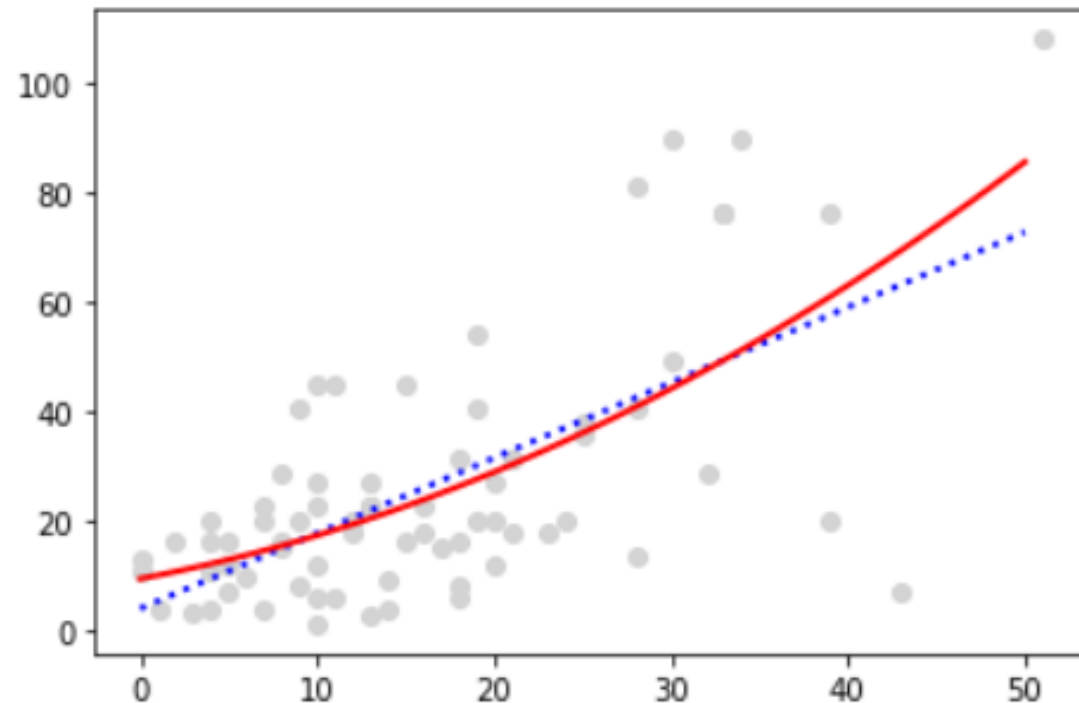
```
plt.xticks(x, case)
lr=LinearRegression()
pr = LinearRegression()
quadratic = PolynomialFeatures(degree=2)
```

```
x_quad = quadratic.fit_transform(x)
```

```
x_fit = np.arange(x.min(), x.max(), 1)[:, np.newaxis]
lr.fit(x,y)
y_lin_fit = lr.predict(x_fit)
l_r2=r2_score(y,lr.predict(x))
```

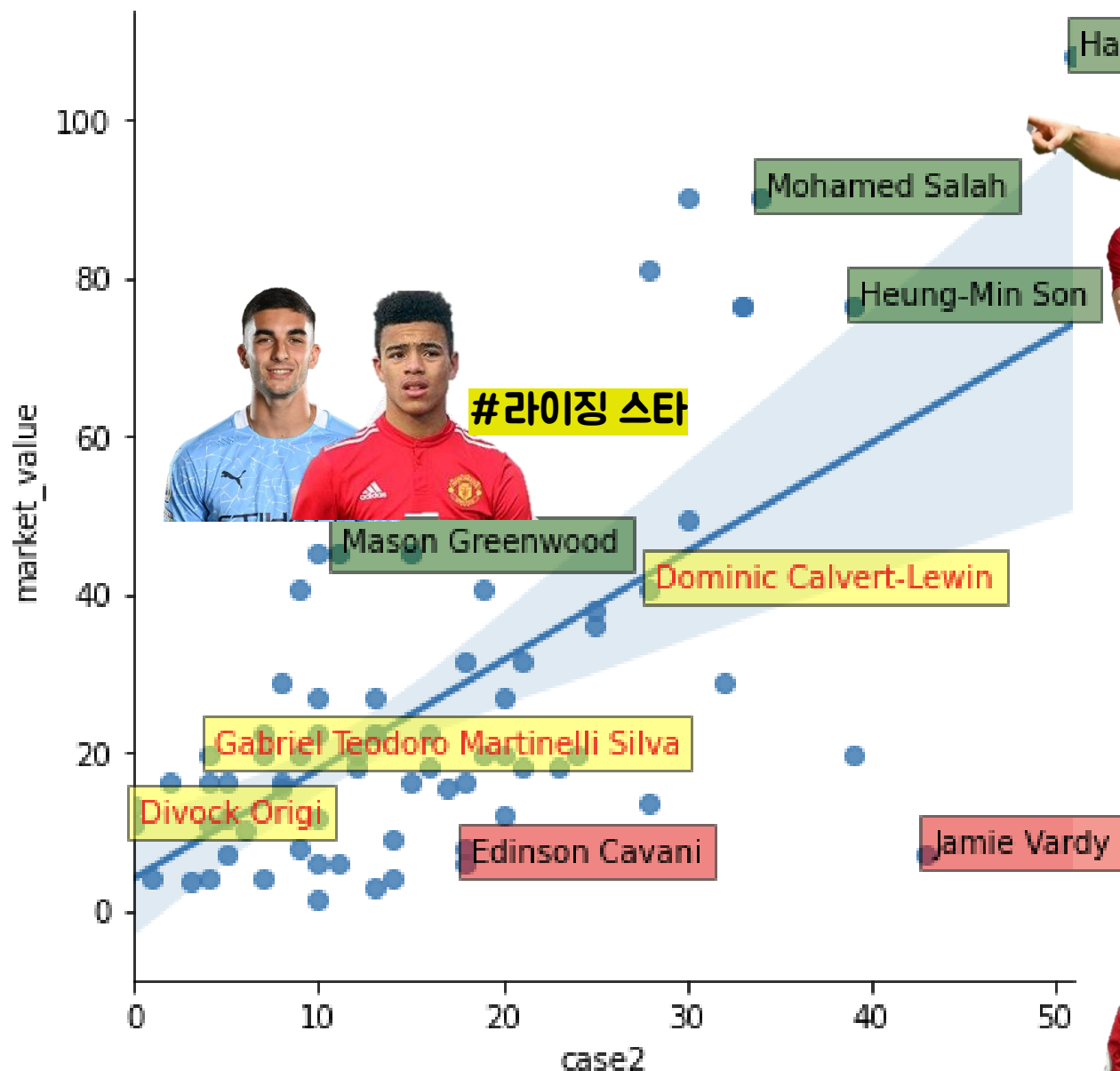
```
lr.fit(x_quad,y)
y_quad_fit=lr.predict(quadratic.fit_transform(x_fit))
q_r2=r2_score(y,lr.predict(x_quad))
```

```
plt.scatter(x,y, c='lightgray')
plt.plot(x_fit, y_lin_fit, linestyle=':', label='linear fit(d=1), $R^2$=%.2f$' %l_r2, c='blue', lw=2)
plt.plot(x_fit, y_quad_fit, linestyle='-', label='linear fit(d=2), $R^2$=%.2f$' %q_r2, c='red', lw=2)
```



```
# Case 6 : 1차 지수 => 0.466
```

```
# Case 7 : 기대 득점 + 기대 어시스트 => 0.125
```



# Case 2 회귀 그래프에 따른  
시장 가치 적정성 평가  
및 이상치(Outlier) 분석

#월드 클래스

#월드클래스

스타성, 마케팅 수익 등  
전문학적인 부가가치로 인해  
시장 가치가 예측 값보다 높음

#라이징 스타

경기 기록은 저조하더라도  
잠재력이 월등히 뛰어나서  
시장 가치가 예측 값보다 높음

#노장 선수

#노장 선수

경기 기록은 좋더라도  
은퇴를 앞두고 있어(잠재력 X)  
시장 가치가 예측 값보다 낮음





## 5. 결론 및 제언

- ① 프리미어리그 공격수 73명의 경기 기록과 시장 가치 데이터의 상관성 분석을 통해 '득점', '어시스트', '평균' 등 총 10가지 경기 지표가 '시장 가치와 상당한 관련성이 있다.'는 것을 발견했다.
- ② 회귀 예측 모형에서 시장 가치를 40% 이상 설명하는 변수는 'ICT 지수', '평균', '골+2\*어시스트'이며, '골+2\*어시스트' 회귀 그래프를 통해 선수들의 시장 가치 적합성을 평가해보았다.
- ③ 회귀 그래프 상 월드클래스, 라이징 스타, 노장 선수 등 경기 기록 이외에 스타성, 잠재력, 시장상황 등 다양하고 복잡한 요소가 반영되지 못해 회귀 그래프 상 '이상치(Outlier)'로 표현되었다고 해석할 수 있다.
- ④ 경기 기록 데이터만으로 시장 가치의 적합성을 완전히 설명하는 데 한계가 있을 수 있지만, 경기 기록을 통해 선수 가치의 적합성을 평가하는 새로운 모델을 제공했다는 측면에서 큰 의미가 있다. 추후 연구에서 여러 지표를 추가 고려하여 R-square값이 더 높은 회귀 예측 모델이 개발된다면 시장 가치 예측, 합리적인 구단 경영 및 선수 영입, 마케팅 등 다양한 분야에서 활용될 것으로 기대된다.

A football player in a white Tottenham Hotspur jersey is celebrating with his arms raised in a 'V' sign. The jersey features the Nike logo, the Tottenham Hotspur crest, and the AIA sponsor logo. The background is a blurred crowd of spectators.

**THANK YOU !**