

NBA, 포지션 별 은퇴 시기

1971422 권수지

주제 선정 이유

- 평소 농구에 관심이 많아 친구와 대화하던 중 포지션이 과연 경력과 연관이 있을지에 대해 얘기를 하다가 주제를 선정
- 경력과 연관이 있을 것이라 생각한 이유는, 포지션에 따라 신체에 무리가 가는 정도가 다르며 퍼포먼스에 따라 달라질 것이라 예측해서
- 본인은 포지션 및 다른 것보다 개개인의 역량 및 관리에 초점이 있어 데이터 상으로 아무것도 나오지 않을 것이라는 의견

데이터 셋

player_data.csv

▲ Player	# From	# To	▲ Pos	▲ Ht	# Wt	📅 Birth Date	▲ Colleges
----------	--------	------	-------	------	------	--------------	------------

seasons_stats.csv

Seasons_stats에는 실제로 더 많은 데이터 셋(51가지)가 있음. 생략 된 year도 꽤 중요함.

# Year	▲ Player	▲ Pos	# G	# FGA	# FTA	# AST	# PF	# PTS
--------	----------	-------	-----	-------	-------	-------	------	-------

데이터 불러와서 저장하기

✓
25
초

```
[1] from google.colab import drive  
drive.mount('/content/drive')
```



Mounted at /content/drive

✓
초

```
[2] import pandas as pd  
import numpy as np
```

✓
초

```
[3] df=pd.read_csv('/content/drive/MyDrive/PPT/player_data.csv')  
seasons_df=pd.read_csv('/content/drive/MyDrive/PPT/seasons_stats.csv', encoding='latin-1')  
  
pd.set_option('display.max_columns', None) # 결과물로 보여주는 열 갯수 최대화  
pd.set_option('display.max_rows', None) # 결과물로 보여주는 행 갯수 최대화
```

생년월일 데이터를 태어난 년도(int) 값으로 바꾸기

```
[4] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 4979 entries, 0 to 4978  
Data columns (total 8 columns):  
#   Column      Non-Null Count  Dtype    
---  ---        
0   Player      4979 non-null   object   
1   From        4979 non-null   int64    
2   To          4979 non-null   int64    
3   Pos         4979 non-null   object   
4   Ht          4979 non-null   object   
5   Wt          4974 non-null   float64  
6   Birth Date  4961 non-null   object   
7   Colleges    4628 non-null   object   
dtypes: float64(1), int64(2), object(5)  
memory usage: 311.3+ KB
```

```
[5] df['Birth Date'] = df['Birth Date'].str.slice(-4)  
df['Birth Date'] = pd.to_numeric(df['Birth Date'], errors='coerce')  
df = df.dropna(subset=['Birth Date'])  
df['Birth Date'] = df['Birth Date'].astype(int)
```

```
<ipython-input-5-8e42cfaecfb6>:4: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df['Birth Date'] = df['Birth Date'].astype(int)

생년월일이 Object 값으로,

DD MM(글자) YYYY

폼이었기에, 뒤에 YYYY만 추출하여 int 값
으로 변경, 태어난 년도로 저장함.

경력과 은퇴 나이 생성

```
[6] df['Period'] = df['To'] - df['From']  
df['retire'] = df['To'] - df['Birth Date']  
cols=['Player', 'Period', 'Pos', 'retire']  
player_df=df[cols]  
player_df.head()
```



	Player	Period	Pos	retire
0	Alaa Abdelnaby	4	F-C	27
1	Zaid Abdul-Aziz	9	C-F	32
2	Kareem Abdul-Jabbar*	19	C	42
3	Mahmoud Abdul-Rauf	10	G	32
4	Tariq Abdul-Wahad	5	F	29



Player DB에 경력 기간이 따로 있지 않기에,
NBA에 입성한 년도와 은퇴한 년도를 통해 계산함.

은퇴 나이 또한 비슷한 방법을 이용하여 int 값으로 생성

[7] player_df.info()

```
<class 'pandas.core.frame.DataFrame'>  
Index: 4961 entries, 0 to 4978  
Data columns (total 4 columns):  
#   Column  Non-Null Count  Dtype  
---  ---  
0   Player  4961 non-null   object  
1   Period  4961 non-null   int64  
2   Pos     4961 non-null   object  
3   retire  4961 non-null   int64  
dtypes: int64(2), object(2)  
memory usage: 193.8+ KB
```

[8] seasons_df.info()

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 28057 entries, 0 to 28056  
Data columns (total 51 columns):  
#   Column  Non-Null Count  Dtype  
---  ---  
0   Unnamed: 0  28057 non-null   int64  
1   Year       28057 non-null   int64  
2   Player     28057 non-null   object  
3   Pos        28057 non-null   object  
4   Age        28049 non-null   float64  
5   Tm         28057 non-null   object  
6   G          28057 non-null   int64  
7   GS         21666 non-null   float64  
8   MP         27571 non-null   float64  
9   FG         28057 non-null   int64  
10  FGA        28057 non-null   int64  
11  FG%        27942 non-null   float64  
12  3P         22360 non-null   float64  
13  3PA        22360 non-null   float64  
14  3P%        18632 non-null   float64  
15  2P         28057 non-null   int64  
16  2PA        28057 non-null   int64
```

(캡처가 한번에 다 되지 않아 생략함)
필요한 것들은 모두 int나 float 값임.

Seasons_df에 필요한 것만 빼기



```
seasons_clos=['Player', 'G', 'FGA', 'FTA', 'AST', 'PF', 'PTS']  
  
seasons_df=seasons_df[seasons_clos]  
seasons_df.head()
```



	Player	G	FGA	FTA	AST	PF	PTS
0	Curly Armstrong	63	516	241	176	217	458
1	Cliff Barker	49	274	106	109	99	279
2	Leo Barnhorst	67	499	129	140	192	438
3	Ed Bartels	15	86	34	20	29	63
4	Ed Bartels	13	82	31	20	27	59



선수 신체에 무리가 갈 수 있는 것들인

Game에 참여한 횟수(G)
골(2P, 3P)을 넣으려고 시도한 횟수(FGA)
자유투를 넣으려고 시도한 횟수(FTA)
어시스트(AST)
Personal Fouls(PF)
포인트(PTS)

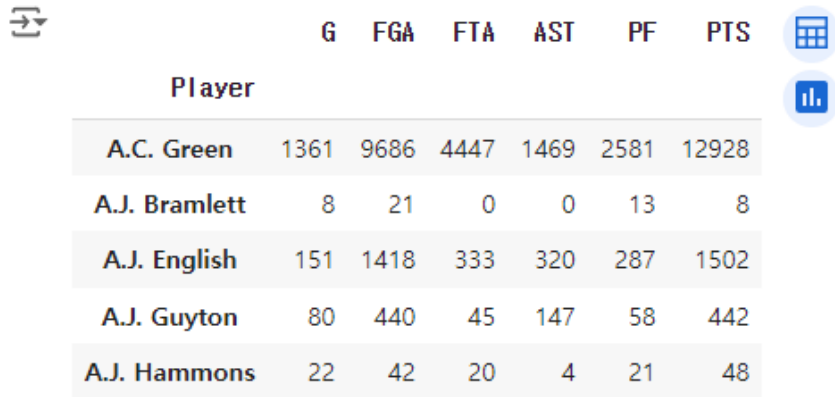
으로 축약

Seasons_df에 필요한 것만 빼기

- G => 실질적으로 게임에 많이 참여할 수록 신체에 무리가 감
- FGA, FTA, AST, PTS => 퍼포먼스를 보였다는 실질적인 지표. 리바운드도 있지만, 일부만 측정하여 자료가 적어 오히려 방해가 될 것을 고려해 완전히 배제함.
- Personal Fouls(PF) => 파울은 당하는 것도 문제지만 본인이 할 때도 신체에 무리가 갈 수도 있기에 포함함.

년도별로 나누어져 있는 걸 합친 뒤 두 DB Merge 함

```
[10] seasons_df=seasons_df.groupby('Player').sum()  
seasons_df.head()
```



A table with 7 columns: Player, G, FGA, FTA, AST, PF, PTS. It shows the sum of statistics for five players: A.C. Green, A.J. Bramlett, A.J. English, A.J. Guyton, and A.J. Hammons.

	Player	G	FGA	FTA	AST	PF	PTS
	A.C. Green	1361	9686	4447	1469	2581	12928
	A.J. Bramlett	8	21	0	0	13	8
	A.J. English	151	1418	333	320	287	1502
	A.J. Guyton	80	440	45	147	58	442
	A.J. Hammons	22	42	20	4	21	48

다음 단계: [seasons_df 변수로 코드 생성](#) [추천 차트 보기](#)

```
[11] NBA_df = pd.merge(player_df, seasons_df, on='Player')  
NBA_df.head()
```



A table with 12 columns: Player, Period, Pos, retire, G, FGA, FTA, AST, PF, PTS. It shows the first 5 rows of the merged dataset for players like Alaa Abdelnaby, Zaid Abdul-Aziz, Kareem Abdul-Jabbar*, Mahmoud Abdul-Rauf, and Tariq Abdul-Wahad.

	Player	Period	Pos	retire	G	FGA	FTA	AST	PF	PTS
0	Alaa Abdelnaby	4	F-C	27	385	1940	472	125	777	2299
1	Zaid Abdul-Aziz	9	C-F	32	570	4588	1536	648	1264	4978
2	Kareem Abdul-Jabbar*	19	C	42	1560	28307	9304	5660	4657	38387
3	Mahmoud Abdul-Rauf	10	G	32	586	7943	1161	2079	1106	8553
4	Tariq Abdul-Wahad	5	F	29	321	2519	755	388	688	2662

Season은 player 별로 있는 것이 아닌, 시즌 + player 별로 정리를 한 것이기 때문에 이미 지운 year은 내버려둔 채 Player(Unique Key)를 이용하여 같은 Player의 퍼포먼스 값을 SUM함.

Player Name이 Unique Key라는 건
Player DB와 seasons DB를 교차 검증
하여 얻어낸 것임

Player(Unique Key)라는 키 값을 이용하여 Player
와 Season(퍼포먼스) DF를 Merge함

포지션 별로 나눔. 필요 없어진 Pos는 삭제

```
[12] def F(x):  
      if 'F' in x:  
          return 1  
      else:  
          return 0  
  
      def G(x):  
          if 'G' in x:  
              return 1  
          else:  
              return 0  
  
      def C(x):  
          if 'C' in x:  
              return 1  
          else:  
              return 0  
  
      NBA_df['F']=NBA_df['Pos'].apply(F)  
      NBA_df['G']=NBA_df['Pos'].apply(G)  
      NBA_df['C']=NBA_df['Pos'].apply(C)  
      NBA_df.head()
```

	Player	Period	Pos	retire	G	FGA	FTA	AST	PF	PTS	F	Gu	C
0	Alaa Abdelnaby	4	F-C	27	385	1940	472	125	777	2299	1	0	1
1	Zaid Abdul-Aziz	9	C-F	32	570	4588	1536	648	1264	4978	1	0	1
2	Kareem Abdul-Jabbar*	19	C	42	1560	28307	9304	5660	4657	38387	0	0	1
3	Mahmoud Abdul-Rauf	10	G	32	586	7943	1161	2079	1106	8553	0	1	0
4	Tariq Abdul-Wahad	5	F	29	321	2519	755	388	688	2662	1	0	0

다음 단계: [NBA_df변수로 코드 생성](#) [추천 차트 보기](#)

```
[13] NBA_df.drop(['Pos'], axis=1, inplace=True)  
      NBA_df.head()
```

Pos는 Object 값으로 G, F, C 등으로 이루어져 있으며 실질적으로 C-F(*빅맨), F-G(*스윙맨)와 같은 값도 있기에 따로 C, F, G 열을 만들어 Pos에 각 문자가 존재하는 지 확인 후, 있으면 1, 없으면 0 값을 넣도록 함

실제로는 C, PF, SF, PG, SG, C-F, F-G 등등 다양한 값이 있었지만, 분류를 하기 위해 크게 C, F, G만으로 구별.
그 외에 존재하는 문자의 종류는 P, S만이 존재하기에 이렇게 분류했을 시 오류도 없을 것이라 판단

```
NBA_df['G/P'] = NBA_df['G'] / NBA_df['Period']
NBA_df.head()
```

	Player	Period	retire	G	FGA	FTA	AST	PF	PTS	F	Gu	C	G/P
0	Alaa Abdelnaby	4	27	385	1940	472	125	777	2299	1	0	1	96.250000
1	Zaid Abdul-Aziz	9	32	570	4588	1536	648	1264	4978	1			
2	Kareem Abdul-Jabbar*	19	42	1560	28307	9304	5660	4657	38387	0			
3	Mahmoud Abdul-Rauf	10	32	586	7943	1161	2079	1106	8553	0			
4	Tariq Abdul-Wahad	5	29	321	2519	755	388	688	2662	1			

```
NBA_df['FGA/G'] = NBA_df['FGA'] / NBA_df['G']
NBA_df['FTA/G'] = NBA_df['FTA'] / NBA_df['G']
NBA_df['AST/G'] = NBA_df['AST'] / NBA_df['G']
NBA_df['PF/G'] = NBA_df['PF'] / NBA_df['G']
NBA_df['PTS/G'] = NBA_df['PTS'] / NBA_df['G']
```

Now drop the original columns

```
NBA_df.drop(['FGA', 'FTA', 'AST', 'PF', 'PTS'], axis=1, inplace=True)
```

```
NBA_df.head()
```

	Player	Period	retire	G	F	Gu	C	G/P	FGA/G	FTA/G	AST/G	PF/G	PTS/G
0	Alaa Abdelnaby	4	27	385	1	0	1	96.250000	5.038961	1.2259			
1	Zaid Abdul-Aziz	9	32	570	1	0	1	63.333333	8.049123	2.6947			
2	Kareem Abdul-Jabbar*	19	42	1560	0	0	1	82.105263	18.145513	5.9641			
3	Mahmoud Abdul-Rauf	10	32	586	0	1	0	58.600000	13.554608	1.981229	3.547782	1.887372	14.595563
4	Tariq Abdul-Wahad	5	29	321	1	0	0	64.200000	7.847352	2.352025	1.208723	2.143302	8.292835

게임을 많이 해서 경력이 긴 것이 아닌
경력이 길기에 게임 횟수가 많은 것으로 판단
원인과 결과가 반대가 되었기에
(게임 횟수)/(경력)을 통하여,
'평균적으로 한 해에 몇 번의 게임을 했는지'로
판단하기로 함.

FGA, FTA, AST, PF, PTS 또한
게임을 많이 하면 많이 얻을 수 있는 것이기에
게임을 한 번 했을 때의 평균으로 하는 것이
좋다고 판단

각각 게임 횟수로 나누어 평소 퍼포먼스의
평균을 구함

```
[16] NBA_df['Period'] = pd.to_numeric(NBA_df['Period'], errors='coerce')
      NBA_df = NBA_df.dropna(subset=['Period'])
      NBA_df['Period'] = NBA_df['Period'].astype(int)
```

```
[17] NBA_df.info()
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4194 entries, 0 to 4193
Data columns (total 13 columns):
 #   Column  Non-Null Count  Dtype
---  -
 0   Player  4194 non-null   object
 1   Period  4194 non-null   int64
 2   retire  4194 non-null   int64
 3   G       4194 non-null   int64
 4   F       4194 non-null   int64
 5   Gu      4194 non-null   int64
 6   C       4194 non-null   int64
 7   G/P     4194 non-null   float64
 8   FGA/G   4194 non-null   float64
 9   FTA/G   4194 non-null   float64
10  AST/G   4194 non-null   float64
11  PF/G    4194 non-null   float64
12  PTS/G   4194 non-null   float64
dtypes: float64(6), int64(6), object(1)
memory usage: 426.1+ KB
```

혹시 모를 오류를 대비하여 넣음

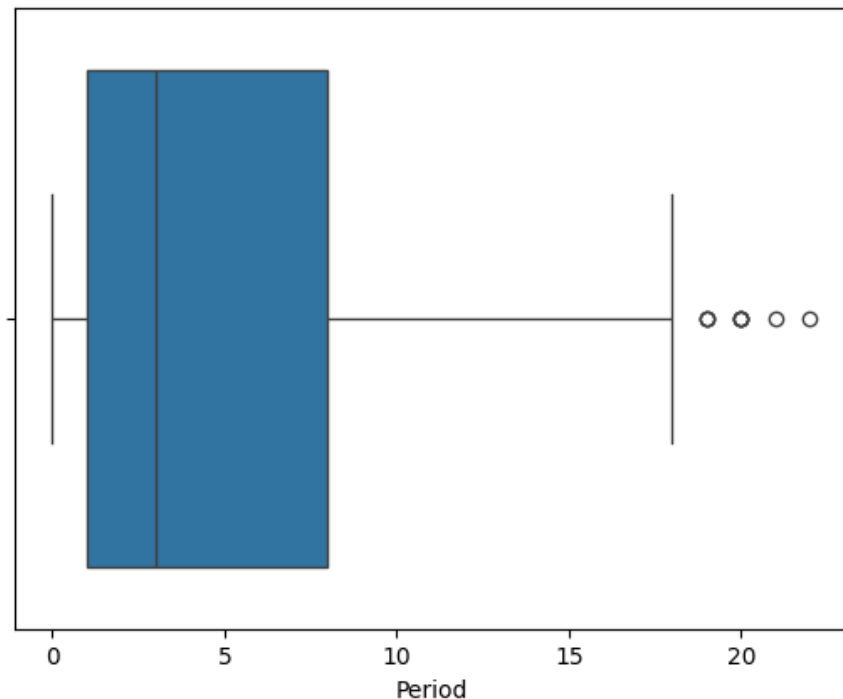
이상값 확인

```
skew = NBA_df['Period'].skew()  
kurt = NBA_df['Period'].kurtosis()  
print('Period - Skewness: {0}, Kurtosis: {1}'.format(skew, kurt))
```

Period - Skewness: 0.8448547166448356, Kurtosis: -0.2453413500542747

```
import seaborn as sns  
sns.boxplot(x = 'Period', data = NBA_df)
```

<Axes: xlabel='Period'>



조금 어려울 것으로 예상
이곳엔 나오지 않았지만, 한 번 Period 값을 훑어봄

이상값 제거(1)

```
[21] Q1 = NBA_df['Period'].quantile(0.25)
      Q3 = NBA_df['Period'].quantile(0.75)
```

```
IQR = Q3-Q1
```

```
Upper = Q3 + 1.5*IQR
```

```
Lower = Q1 - 1.5*IQR
```

```
print(Upper)
```

```
print(Lower)
```

```
⇒ 18.5
   -9.5
```

```
[30] c1 = NBA_df['Period'] <= Upper
      c2 = NBA_df['Period'] > 0
      NBA_df = NBA_df[c1&c2]
      NBA_df.shape
```

Period 이상값을 제거, 3은 숫자가 너무 크기에 1.5로 정함.
(이상값 제거 전, Period의 평균값은 3이었음)

퍼포먼스가 경력에 미치는 영향이기에
1년 미만의 값들 또한 이상값으로 판단하여 모두 없애기로 함.

이상값 제거(2)

```
Q1 = NBA_df['G'].quantile(0.25)
Q3 = NBA_df['G'].quantile(0.75)
```

```
IQR = Q3-Q1
```

```
Upper = Q3 + 1.5*IQR
Lower = Q1 - 1.5*IQR
```

```
print(Upper)
print(Lower)
```

```
1214.625
-666.375
```

```
c1 = NBA_df['G'] <= Upper
c2 = NBA_df['G'] > Lower
NBA_df = NBA_df[c1&c2]
NBA_df.shape
```

게임 횟수의 이상값 제거,
이 또한 Period와 같은 연유로 최소 1번보다 많이 플레이하는
플레이어만을 꼽음.

더 큰 값으로 잡을 수도 있지만 이 이상은 원하는 결과를 위해
데이터를 조작하는 것처럼 느껴져 2번 이상도 포함을 시킴

이상값 제거(3)

```
[24] Q1 = NBA_df['retire'].quantile(0.25)
      Q3 = NBA_df['retire'].quantile(0.75)

      IQR = Q3-Q1

      Upper = Q3 + 1.5*IQR
      Lower = Q1 - 1.5*IQR

      print(Upper)
      print(Lower)
```

⇒ 43.5
15.5

```
[25] c1 = NBA_df['retire'] <= Upper
      c2 = NBA_df['retire'] >= Lower
      NBA_df = NBA_df[c1&c2]
      NBA_df.shape
```

⇒ (3229, 13)

은퇴 나이 이상값 제거.
모두 균일하게 1.5로 이상값을 제거함



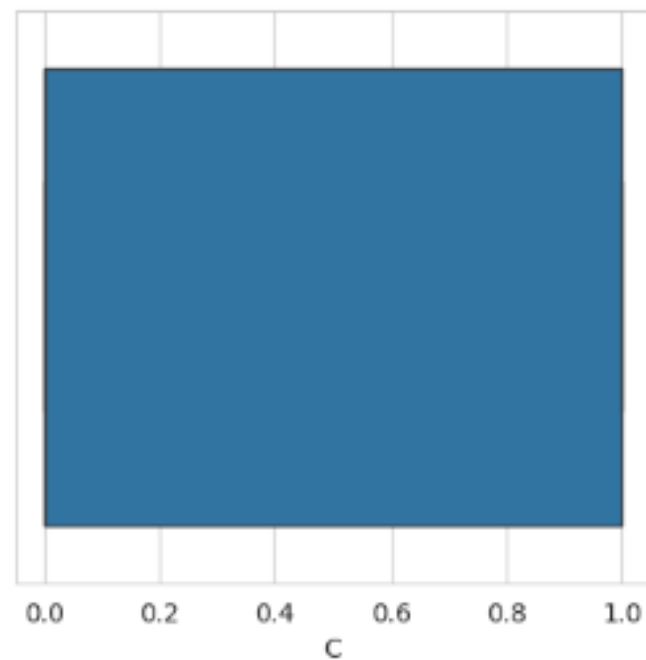
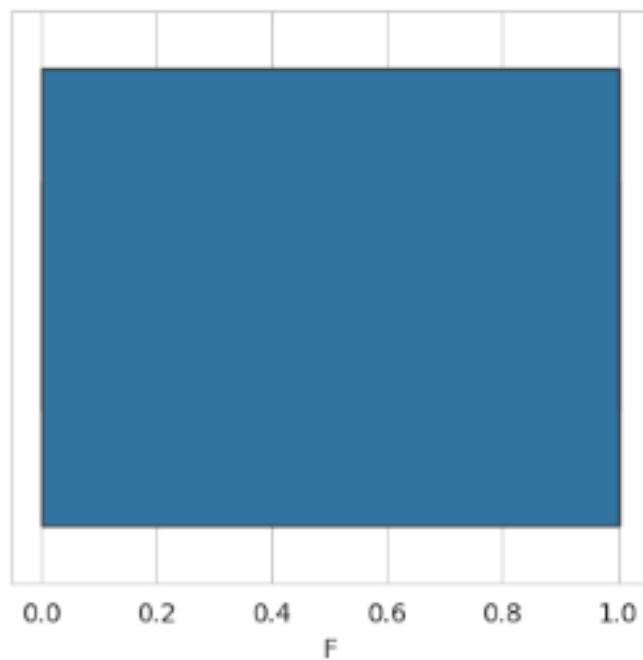
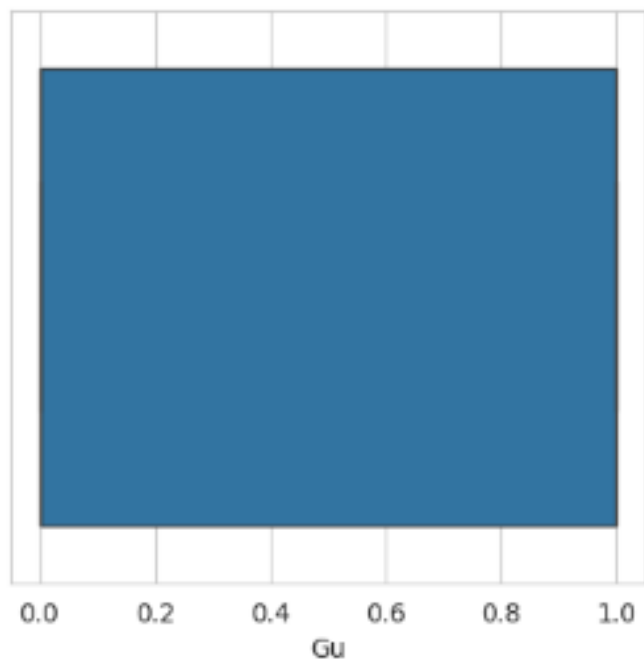
```
sns.set_style('whitegrid')  
fig, axes = plt.subplots(1,3,figsize=(15,4))  
  
sns.boxplot(ax=axes[0],x = 'Gu', data = NBA_df)  
sns.boxplot(ax=axes[1],x = 'F', data = NBA_df)  
sns.boxplot(ax=axes[2],x = 'C', data = NBA_df)
```

시각화를 해보았음.

(혹시 몰라 다시 작성하면서 해보니 그래프가 다르게 나옴)
하지만 크게 의미가 있어 보이지는 않음.



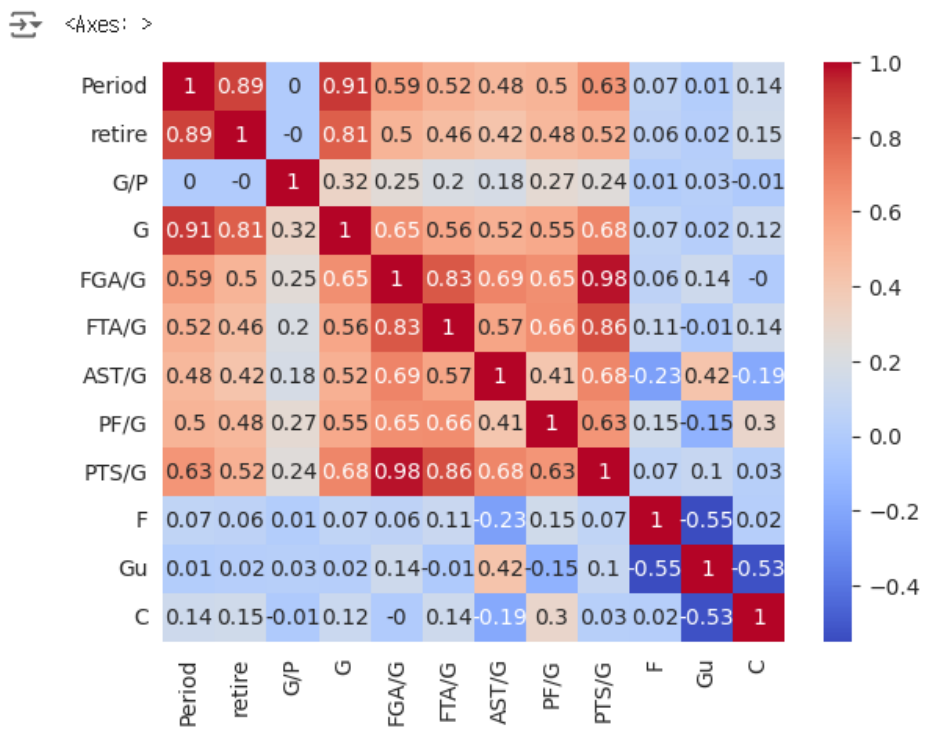
<Axes: xlabel='C'>



상관계수

```
[37] cols3= ['Period', 'retire', 'G/P', 'G', 'FGA/G', 'FTA/G', 'AST/G', 'PF/G', 'PTS/G', 'F', 'Gu', 'C']
```

```
corr=NBA_df[cols3].corr().round(2)  
sns.heatmap(corr, annot=True, cmap='coolwarm')
```



상관계수를 확인하니 Pos는 Period와 거의 상관이 없었음
의외인 점은 점수를 내려 시도하는 것은 물론이며
점수 또한 포지션에 따라 크게 다르지 않다는 점을 알게 됨
그나마 의미를 둘 수 있는 건 AST지만, 이 또한 값이 크지 않음

실직적으로 Pos에 영향을 미치는 것은
Pos밖에 없는 것처럼 보임

T-검정

```
[30] data_1=NBA_df[NBA_df['F']==1]['Period']  
data_0=NBA_df[NBA_df['F']==0]['Period']  
  
stats.ttest_ind(data_1,data_0)
```

```
⇒ TtestResult(statistic=3.342042392015646, pvalue=0.0008411164295426446, df=3227.0)
```

```
[31] data_1=NBA_df[(NBA_df['C']==1)]['Period']  
data_0=NBA_df[(NBA_df['C']==0)]['Period']  
  
stats.ttest_ind(data_1,data_0)
```

```
⇒ TtestResult(statistic=7.044631852184421, pvalue=2.2631315223632014e-12, df=3227.0)
```

```
[32] data_1=NBA_df[NBA_df['Gu']==1]['Period']  
data_0=NBA_df[NBA_df['Gu']==0]['Period']  
  
stats.ttest_ind(data_1,data_0)
```

```
⇒ TtestResult(statistic=-0.7635852888721695, pvalue=0.44517023993101246, df=3227.0)
```

대부분 매우 작은 숫자가 나옴.
가드 값이 크기는 하지만,
이는 단순히 가드가 많은 걸로 추정
(실제로 가드는 1561명으로 40% 이상임)

```
data_1=NBA_df[NBA_df['Gu']==1]  
data_1.count()
```

⇒ Player	1561
Period	1561
retire	1561
G	1561

시각화

이 또한 썸 의미가 있어 보이지는 않음

```
[33] fig, axes = plt.subplots(1,3,figsize=(15,4))
```

```
sns.histplot(ax=axes[0],x="Period", hue="Gu", data = NBA_df)
```

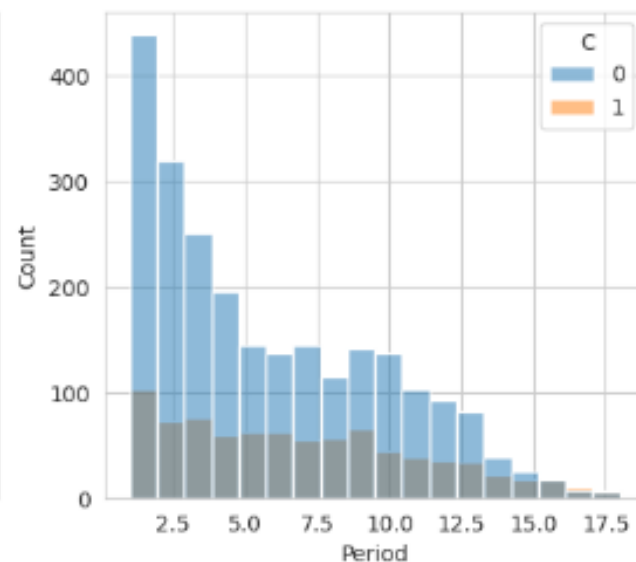
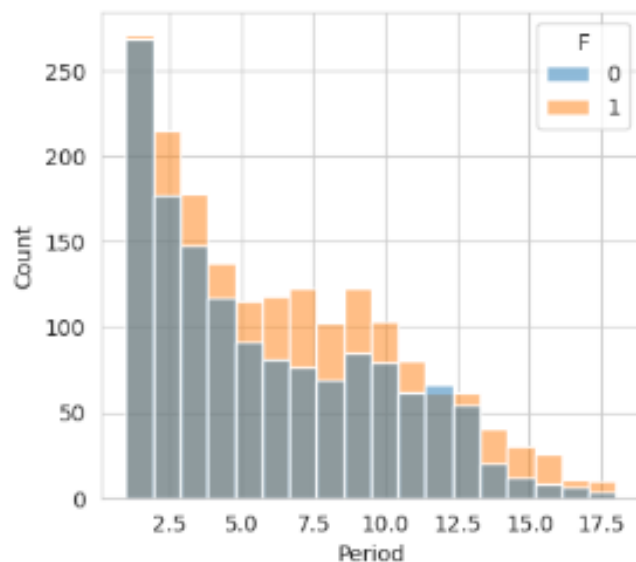
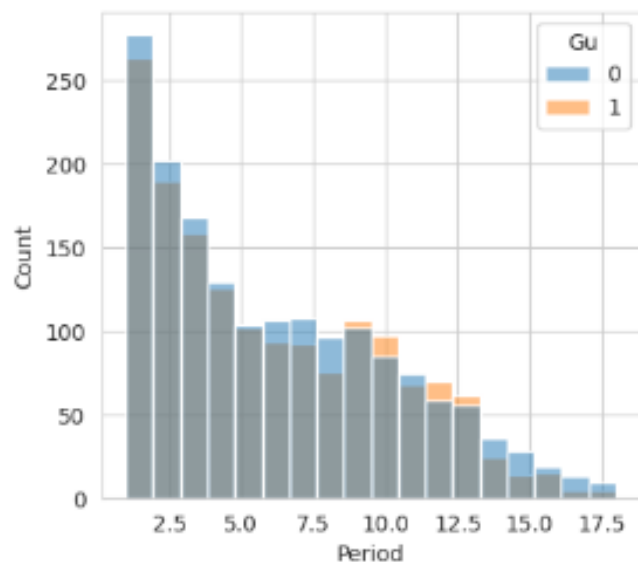
```
sns.histplot(ax=axes[1],x="Period", hue="F", data = NBA_df)
```

```
sns.histplot(ax=axes[2],x="Period", hue="C", data = NBA_df)
```

```
#sns.histplot(data=NBA_df, x="Period", hue="Gu", bins=20)
```



<Axes: xlabel='Period', ylabel='Count'>



시각화

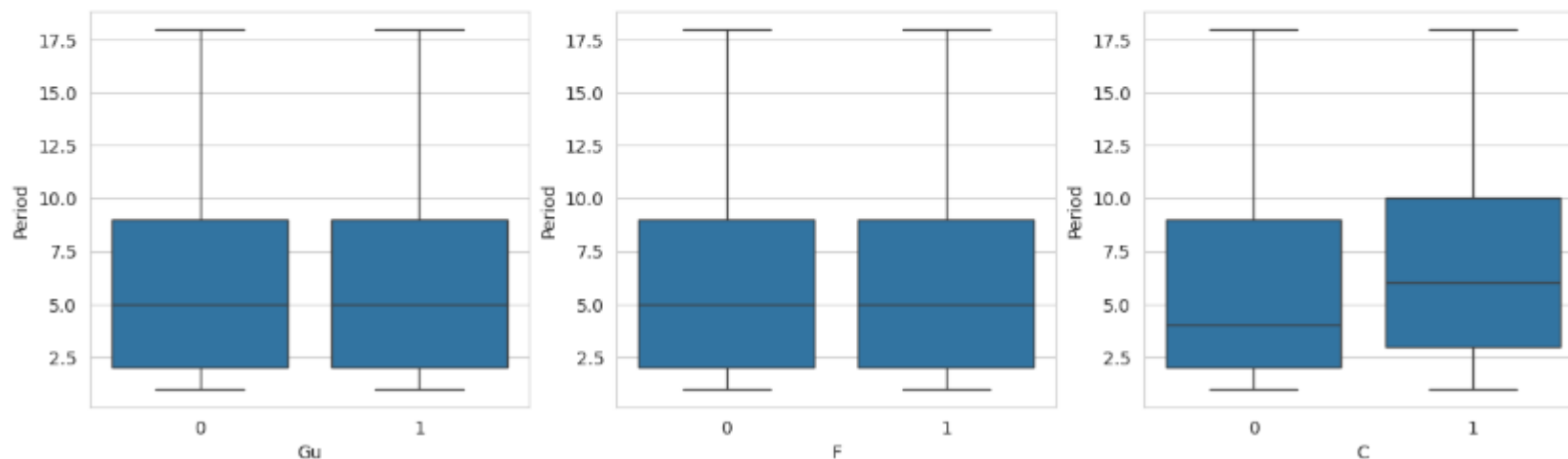
수정을 거쳐가며 지금은 보이지 않지만,
연도 또한 확인했음에도 유의미한 결과값을 내는 것은
크게 없었음.

결국 획일적이거나 보이는 이유보다는 개인의 관리나
역량이 선수 기간에 큰 영향을 미치는 것으로 보임

```
[34] sns.set_style('whitegrid')  
fig, axes = plt.subplots(1,3,figsize=(15,4))  
  
sns.boxplot(ax=axes[0],x = 'Gu', y='Period',data = NBA_df)  
sns.boxplot(ax=axes[1],x = 'F',y='Period', data = NBA_df)  
sns.boxplot(ax=axes[2],x = 'C', y='Period',data = NBA_df)  
  
#sns.boxplot(x='Gu',y='Period',data=NBA_df)
```



<Axes: xlabel='C', ylabel='Period'>



결론

- 포지션은 실질적으로 경력은 물론이며 득점에도 실질적인 의미가 크게 없음
- 결국 포지션이 중요한 것이 아닌 개인 역량으로 보임
- (+)득점이 높고 시도를 많이 하거나 게임의 횟수가 많은 것은 신체가 무리가 되는 마이너스 요소가 아니라 경력을 이어 나갈 수 있는 실력의 증거로 보임

결론(TMI)

```
NBA_df['Period'].median()
```

5.0

- NBA의 선수 경력은 평균 5년으로 그다지 길지 않음

```
NBA_df['retire'].median()
```

29.0

- 은퇴 나이 또한 29살이 평균으로 젊음

- 퍼포먼스(혹은 포지션)에 따르는 유의미한 결과값을 받기엔 실제 선수들의 선수 생활이 애초에 그다지 길지 않다는 결론이 날 수도 있음.

그렇기에 이는 개인의 관리로 충분히 커버가 가능한 것으로 보임

감사합니다