Scraping Data

This jupyter notebook will:

- extract raw NBA statistics
- store them in an excel workbook for easy access

In [ ]:
```
# importing libraries and modules needed to scrape the nba player statistics
import pandas as pd
import numpy as np
import requests
```

In [ ]:
```
# set up for nba web app api call
headers  = {
    'Connection': 'keep-alive',
    'Accept': 'application/json, text/plain, */*',
    'x-nba-stats-token': 'true',
    'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKi
    'x-nba-stats-origin': 'stats',
    'Sec-Fetch-Site': 'same-origin',
    'Sec-Fetch-Mode': 'cors',
    'Referer': 'https://stats.nba.com/',
    'Accept-Encoding': 'gzip, deflate, br',
    'Accept-Language': 'en-US,en;q=0.9',
}
```

In [ ]:
```
# preparing a lookup tables of statistics and urls
# statistics are for season 2020-2021 (for a more complete dataset)
stats_url_lookup = {
    'general_traditional': 'https://stats.nba.com/stats/leaguedashplayerstats
    'general_advanced': 'https://stats.nba.com/stats/leaguedashplayerstats?Co
    'defense_dashboard_overall': 'https://stats.nba.com/stats/leaguedashptdef
    'defense_dashboard_3pt_defense': 'https://stats.nba.com/stats/leaguedashp
    'defense_dashboard_<_6ft_defense': 'https://stats.nba.com/stats/leaguedas
    'defense_dashboard_<_10ft_defense': 'https://stats.nba.com/stats/leagueda
    'defense_dashboard_>15ft_defense': 'https://stats.nba.com/stats/leaguedas
    'hustle': 'https://stats.nba.com/stats/leaguehustlestatsplayer?College=&C
    'shooting': 'https://stats.nba.com/stats/leaguedashplayershotlocations?Co
    'playtype_isolation': 'https://stats.nba.com/stats/synergyplaytypes?Leagu
    'playtype_pnr_ball_handler': 'https://stats.nba.com/stats/synergyplaytype
    'playtype_pnr_roll_man': 'https://stats.nba.com/stats/synergyplaytypes?Le
    'playtype_transition': 'https://stats.nba.com/stats/synergyplaytypes?Leag
    'playtype_post_up': 'https://stats.nba.com/stats/synergyplaytypes?LeagueI
    'playtype_spot_up': 'https://stats.nba.com/stats/synergyplaytypes?LeagueI
    'playtype_handoff': 'https://stats.nba.com/stats/synergyplaytypes?LeagueI
    'playtype_cut': 'https://stats.nba.com/stats/synergyplaytypes?LeagueID=00
    'playtype_off_screen': 'https://stats.nba.com/stats/synergyplaytypes?Leag
    'playtype_putback': 'https://stats.nba.com/stats/synergyplaytypes?LeagueI
    'tracking_passing': 'https://stats.nba.com/stats/leaguedashptstats?Colleg
}
```

In [ ]:
```
# fetching statistics from stats.nba.com, and pour the data into the excel wo

def stats_to_xlxs(lookup):
    with pd.ExcelWriter('nba_stats.xlsx', mode='a', if_sheet_exists='new') as
        try:
            pairs = lookup.items()
```

```
            for pair in pairs:
                json = requests.get(pair[1], headers=headers).json()
                if (pair[0] == 'shooting'):
                    stats = json['resultSets']['rowSet']
                    labels = json['resultSets']['headers'][1]['columnNames']
                else:
                    stats = json['resultSets'][0]['rowSet']
                    labels = json['resultSets'][0]['headers']
                df = pd.DataFrame.from_records(stats, columns=labels)
                df.to_excel(writer, sheet_name=pair[0])
        except Exception as e:
            print(e)
            print('error occurs at:', pair[0])

stats_to_xlxs(stats_url_lookup)
```

```
/opt/miniconda3/envs/tf_python/lib/python3.7/site-packages/openpyxl/workbook/child.py:99: UserWarning: Title is more than 31 characters. Some applications may not be able to read the file
  warnings.warn("Title is more than 31 characters. Some applications may not be able to read the file")
```

Selecting Features

This jupyter notebook will:

- converting data in the excel file to multiple dataframes
- cleaning data by dropping unnecessary features
- using correlation matrix to detect similar features and then drop them to reduce dimensionality
- merging tidied dataframes and then exporting it as a pickle file to do further analysis

In [ ]:
```
# importing libraries and modules needed for feature selection
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
# loading nba data and transforming them back to dataframes
# due to the nubmer of characters restriction on excel spreadsheet naming, 'd
nba_stats = pd.ExcelFile('nba_stats.xlsx')
df_list = ['general_traditional', 'general_advanced', 'defense_dashboard_over
dfs = [pd.read_excel(nba_stats, f"{df_name}") for df_name in df_list ]
```

In [ ]:
```
# data scraped from nba api contains hidden data not shown on the website, wh
# then use correlation matrix to cut down the number of variables

# general_traditional df
general_traditional = dfs[0]
general_traditional.drop(['Unnamed: 0', 'PLAYER_ID', 'NICKNAME', 'TEAM_ID',
       'AGE', 'W', 'L', 'NBA_FANTASY_PTS', 'GP_RANK','W_RANK', 'L_RANK',
       'W_PCT_RANK', 'MIN_RANK', 'FGM_RANK', 'FGA_RANK', 'FG_PCT_RANK',
       'FG3M_RANK', 'FG3A_RANK', 'FG3_PCT_RANK', 'FTM_RANK', 'FTA_RANK',
       'FT_PCT_RANK', 'OREB_RANK', 'DREB_RANK', 'REB_RANK', 'AST_RANK',
       'TOV_RANK', 'STL_RANK', 'BLK_RANK', 'BLKA_RANK', 'PF_RANK',
       'PFD_RANK', 'PTS_RANK', 'PLUS_MINUS_RANK', 'NBA_FANTASY_PTS_RANK',
       'DD2_RANK', 'TD3_RANK', 'CFID', 'CFPARAMS'], axis=1, inplace=True)
# limiting sample size by dropping inactive players
general_traditional = general_traditional[(general_traditional['MIN'] >= 12)
general_traditional.reset_index(drop=True, inplace=True)
general_traditional
```

Out[ ]:

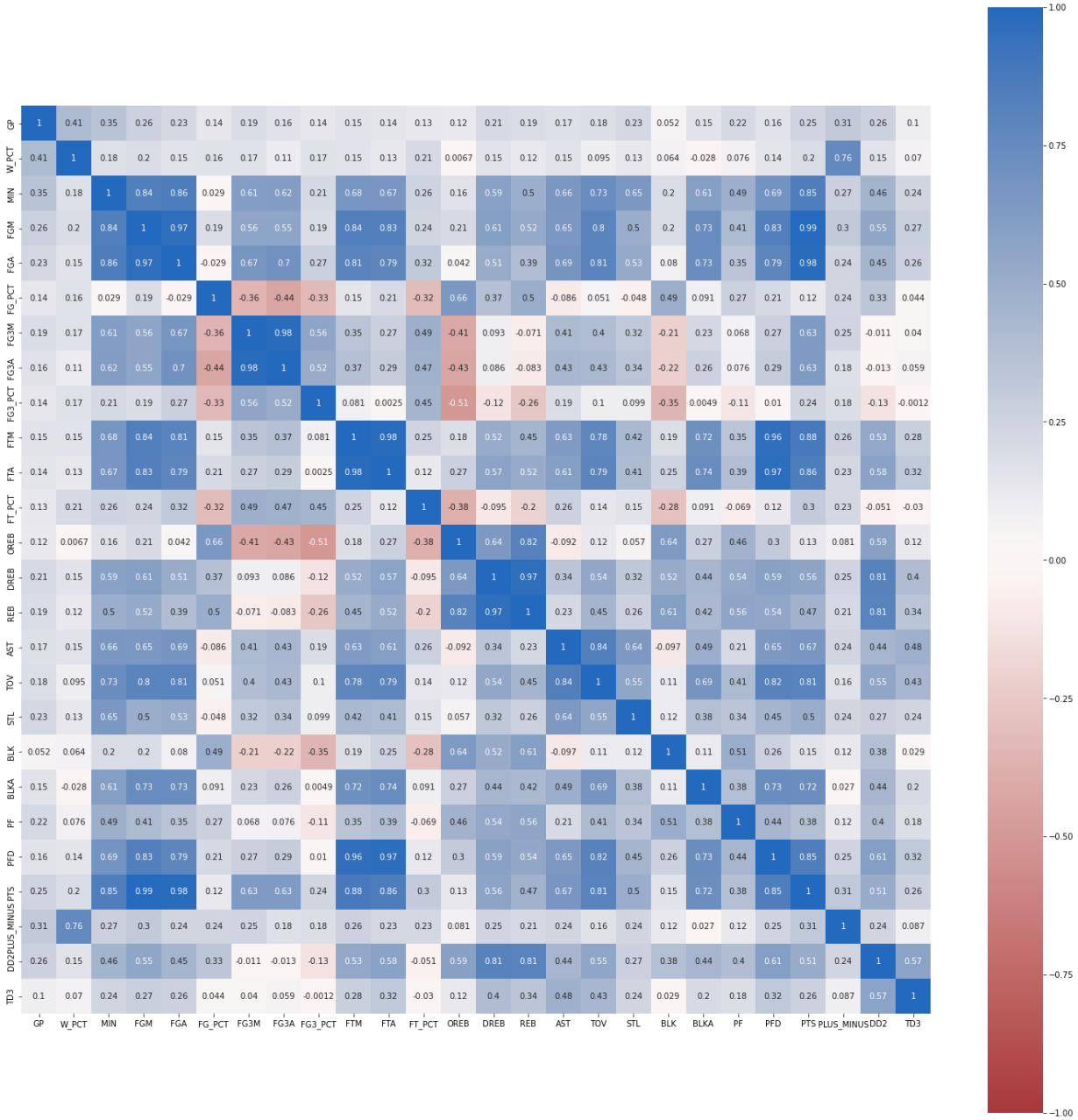
	PLAYER_NAME	TEAM_ABBREVIATION	GP	W_PCT	MIN	FGM	FGA	FG_PCT	FG3M	F
0	Aaron Gordon	DEN	50	0.580	27.7	4.6	10.0	0.463	1.2	
1	Aaron Holiday	IND	66	0.455	17.8	2.6	6.6	0.390	1.0	
2	Aaron Nesmith	BOS	46	0.478	14.5	1.7	3.9	0.438	0.9	
3	Abdel Nader	PHX	24	0.667	14.8	2.4	4.8	0.491	0.8	
4	Al Horford	BOS	28	0.393	27.9	5.8	12.9	0.450	2.0	
...	...	...	...	...	...	...	...	...	...	
391	Xavier Tillman	MEM	59	0.525	18.4	2.8	5.1	0.559	0.4	
392	Yogi Ferrell	LAC	10	0.400	13.7	2.0	5.7	0.351	0.9	
393	Yuta Watanabe	TOR	50	0.480	14.5	1.6	3.6	0.439	0.7	

	PLAYER_NAME	TEAM_ABBREVIATION	GP	W_PCT	MIN	FGM	FGA	FG_PCT	FG3M	F
394	Zach LaVine	CHI	58	0.448	35.1	9.8	19.4	0.507	3.4	
395	Zion Williamson	NOP	61	0.475	33.2	10.4	17.0	0.611	0.2	

396 rows × 28 columns

In [ ]:
```
general_traditional_corr = general_traditional.corr()
plt.figure(figsize=(25,25))
sns.heatmap(general_traditional_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r'
```

Out[ ]: <AxesSubplot:>



In [ ]:
```
general_traditional.drop(['GP', 'W_PCT', 'MIN', 'FGM', 'FGA', 'FG3M', 'FG3A',
```

```
/opt/miniconda3/envs/tf_python/lib/python3.7/site-packages/pandas/core/frame.p
y:4913: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
```

```
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  errors=errors,
```

In [ ]:

```
# general_advanced df
general_advanced = dfs[1]
general_advanced.drop(['Unnamed: 0', 'PLAYER_ID', 'NICKNAME', 'TEAM_ID',
       'AGE', 'GP', 'W', 'L', 'MIN', 'E_OFF_RATING',
       'sp_work_OFF_RATING', 'E_DEF_RATING', 'sp_work_DEF_RATING', 'E_NET_RAT
       'sp_work_NET_RATING', 'E_TOV_PCT', 'E_USG_PCT', 'E_PACE', 'PACE_PER40'
       'POSS', 'FGM', 'FGA', 'FGM_PG', 'FGA_PG', 'FG_PCT', 'GP_RANK',
       'W_RANK', 'L_RANK', 'W_PCT_RANK', 'MIN_RANK', 'E_OFF_RATING_RANK',
       'OFF_RATING_RANK', 'sp_work_OFF_RATING_RANK', 'E_DEF_RATING_RANK',
       'DEF_RATING_RANK', 'sp_work_DEF_RATING_RANK', 'E_NET_RATING_RANK',
       'NET_RATING_RANK', 'sp_work_NET_RATING_RANK', 'AST_PCT_RANK',
       'AST_TO_RANK', 'AST_RATIO_RANK', 'OREB_PCT_RANK', 'DREB_PCT_RANK',
       'REB_PCT_RANK', 'TM_TOV_PCT_RANK', 'E_TOV_PCT_RANK', 'EFG_PCT_RANK',
       'TS_PCT_RANK', 'USG_PCT_RANK', 'E_USG_PCT_RANK', 'E_PACE_RANK',
       'PACE_RANK', 'sp_work_PACE_RANK', 'PIE_RANK', 'FGM_RANK', 'FGA_RANK',
       'FGM_PG_RANK', 'FGA_PG_RANK', 'FG_PCT_RANK', 'CFID', 'CFPARAMS'], axis
general_advanced
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	W_PCT	OFF_RATING	DEF_RATING	NET_RATING
0	Aaron Gordon	DEN	0.580	112.5	110.4	2.1
1	Aaron Holiday	IND	0.455	110.1	110.3	-0.2
2	Aaron Nesmith	BOS	0.478	108.6	109.1	-0.5
3	Abdel Nader	PHX	0.667	106.7	101.7	5.0
4	Adam Mokoka	CHI	0.214	83.2	90.3	-7.1
...	...	...	...	...	...	...
535	Yogi Ferrell	LAC	0.400	106.7	107.1	-0.4
536	Yuta Watanabe	TOR	0.480	105.0	107.8	-2.7
537	Zach LaVine	CHI	0.448	112.4	113.1	-0.8
538	Zeke Nnaji	DEN	0.714	102.0	112.5	-10.5
539	Zion Williamson	NOP	0.475	113.9	111.7	2.1

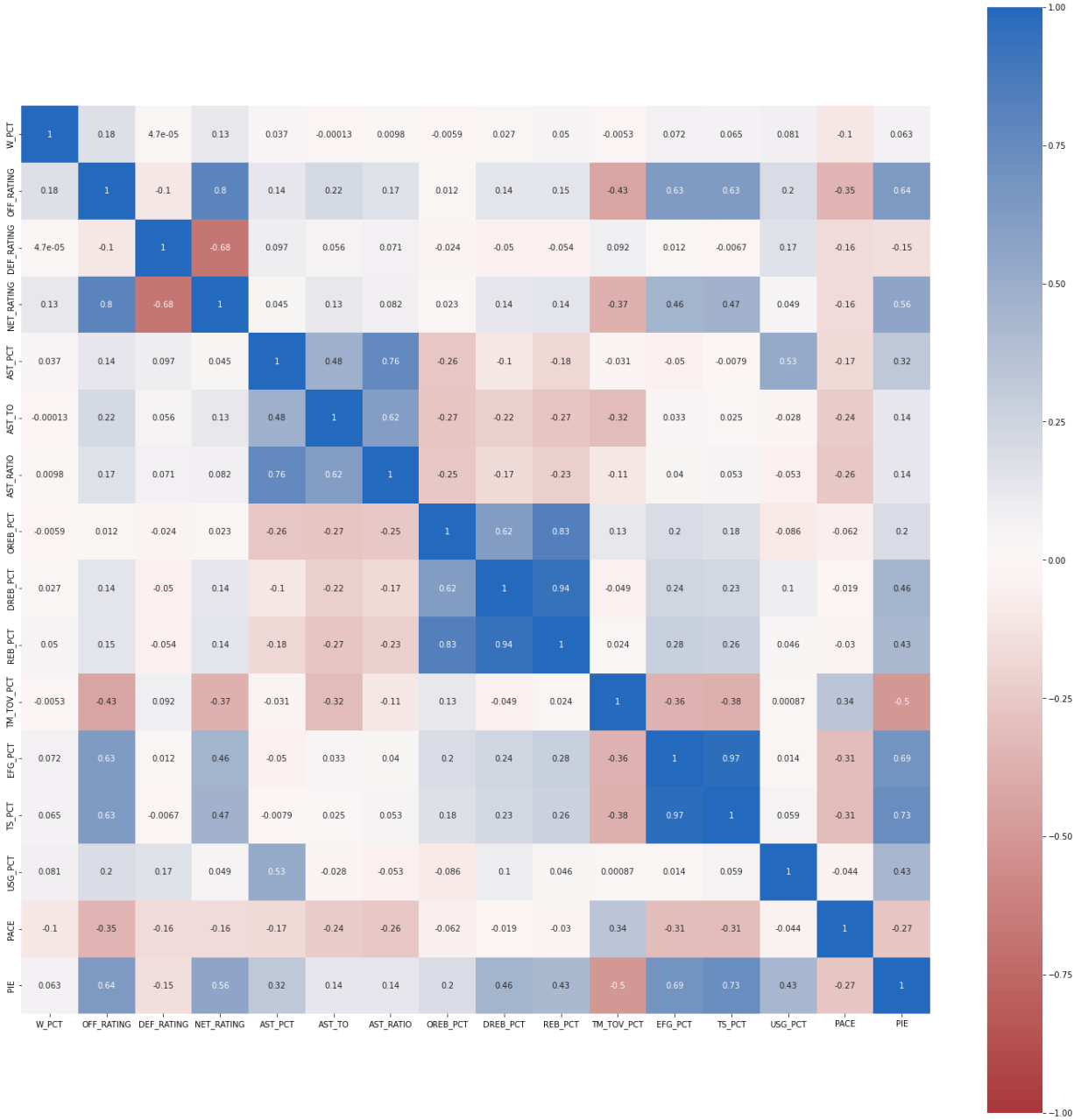
540 rows × 18 columns

In [ ]:

```
general_advanced_corr = general_advanced.corr()
plt.figure(figsize=(25,25))
sns.heatmap(general_advanced_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', s
```

Out[ ]:

```
<AxesSubplot:>
```



In [ ]:
```
general_advanced.drop(['W_PCT', 'OFF_RATING', 'DEF_RATING', 'NET_RATING', 'AS
```

In [ ]:
```
# defense_dashboard_overall df
defense_dashboard_overall = dfs[2]
defense_dashboard_overall.drop(['Unnamed: 0', 'CLOSE_DEF_PERSON_ID', 'PLAYER_
defense_dashboard_overall
```

Out[ ]:

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	GP	G	F
0	Rudy Gobert	UTA	C	29	71	71	
1	Brook Lopez	MIL	C	33	69	69	
2	Domantas Sabonis	IND	F-C	25	61	61	
3	Nikola Jokic	DEN	C	26	71	71	
4	Myles Turner	IND	C-F	25	46	46	
...	...	...	...	...	...	...	...
532	Theo Pinson	NYK	G-F	25	12	12	

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	GP	G	F
533	Keljin Blevins	POR	G	25	14	14	
534	Grant Riller	CHA	G	24	5	5	
535	Robert Woodard II	SAC	F	21	7	7	
536	Greg Whittington	DEN	F	28	1	1	

537 rows × 12 columns

In [ ]:
```
# defense_dashboard_3pt_defense df
defense_dashboard_3pt_defense = dfs[3]
defense_dashboard_3pt_defense.drop(['Unnamed: 0', 'CLOSE_DEF_PERSON_ID', 'PLA
defense_dashboard_3pt_defense
```

Out[ ]:

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	GP	G	F
0	Zion Williamson	NOP	F	20	59	59	C
1	Pascal Siakam	TOR	F	27	56	56	C
2	Darius Bazley	OKC	F-G	21	54	54	C
3	Shai Gilgeous-Alexander	OKC	G-F	22	34	34	C
4	RJ Barrett	NYK	F-G	21	71	70	C
...	...	...	...	...	...	...	
529	Noah Vonleh	BKN	F	25	4	1	C
530	Udoka Azubuike	UTA	C-F	21	12	5	C
531	Bruno Caboclo	HOU	F	25	5	2	C
532	Terrance Ferguson	PHI	G	23	6	2	C
533	Robert Woodard II	SAC	F	21	7	1	C

534 rows × 12 columns

In [ ]:
```
# defense_dashboard_6ft_defense df
defense_dashboard__6ft_defense = dfs[4]
defense_dashboard__6ft_defense.drop(['Unnamed: 0', 'CLOSE_DEF_PERSON_ID', 'PL
defense_dashboard__6ft_defense
```

Out[ ]:

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	GP	G	F
0	Myles Turner	IND	C-F	25	46	46	C
1	Domantas Sabonis	IND	F-C	25	61	61	C
2	Rudy Gobert	UTA	C	29	71	71	C
3	Joel Embiid	PHI	C-F	27	51	51	C
4	Jakob Poeltl	SAS	C	25	69	68	C

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	GP	G	F
...	...	...	...	...	...	...	
530	Jared Harper	NYK	G	23	4	2	0
531	Noah Vonleh	BKN	F	25	4	2	0
532	Devon Dotson	CHI	G	21	7	2	
533	Grant Riller	CHA	G	24	5	2	0
534	Keljin Blevins	POR	G	25	14	5	0

535 rows × 12 columns

In [ ]:
```
# defense_dashboard_10ft_defense df
defense_dashboard_10ft_defens = dfs[5]
defense_dashboard_10ft_defens.drop(['Unnamed: 0', 'CLOSE_DEF_PERSON_ID', 'PLA
defense_dashboard_10ft_defens
```

Out[ ]:

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	GP	G	F
0	Rudy Gobert	UTA	C	29	71	71	0
1	Domantas Sabonis	IND	F-C	25	61	61	0
2	Myles Turner	IND	C-F	25	46	46	0
3	Jakob Poeltl	SAS	C	25	69	68	0
4	Nikola Jokic	DEN	C	26	71	71	0
...	...	...	...	...	...	...	
531	Noah Vonleh	BKN	F	25	4	2	0
532	Jared Harper	NYK	G	23	4	2	0
533	Keljin Blevins	POR	G	25	14	6	0
534	Grant Riller	CHA	G	24	5	2	0
535	Marques Bolden	CLE	C	23	4	1	

536 rows × 12 columns

In [ ]:
```
# defense_dashboard_15ft_defense df
defense_dashboard_15ft_defense = dfs[6]
defense_dashboard_15ft_defense.drop(['Unnamed: 0', 'CLOSE_DEF_PERSON_ID', 'PL
defense_dashboard_15ft_defense
```

Out[ ]:

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	GP	G	F
0	Rudy Gobert	UTA	C	29	71	71	0
1	Brook Lopez	MIL	C	33	69	69	0
2	Christian Wood	HOU	F	25	41	41	0
3	Darius Bazley	OKC	F-G	21	54	54	0
4	Shai Gilgeous-Alexander	OKC	G-F	22	34	34	0
...	...	...	...	...	...	...	

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	GP	G	F
531	Jaylen Adams	MIL	G	25	2	1	0
532	Jalen Lecque	IND	G	21	4	2	0
533	Iman Shumpert	BKN	G	31	2	1	0
534	Terrance Ferguson	PHI	G	23	6	2	0
535	Robert Woodard II	SAC	F	21	7	1	0

536 rows × 12 columns

In [ ]:

```
# as defense area are overlapped, so as the statistics
# we have to merge the defense dataframes and dissect the defense area throug

# renaming and dropping duplicate columns helps merging
defense_dashboard_overall.drop(['GP', 'G'], axis=1, inplace=True)
defense_dashboard_overall.rename(
    {
        'FREQ': 'OVERALL_FREQ',
        'D_FGM': 'OVERALL_DFGM',
        'D_FGA': 'OVERALL_DFGA',
        'D_FG_PCT': 'OVERALL_DFG%',
        'NORMAL_FG_PCT': 'OVERALL_FG%',
        'PCT_PLUSMINUS': 'OVERALL_%DIFF'
    }, axis=1, inplace=True
)

defense_dashboard__6ft_defense.drop(['GP', 'G'], axis=1, inplace=True)
defense_dashboard__6ft_defense.rename(
    {
        'FREQ': 'LT6FT_FREQ',
        'FGM_LT_06': 'LT6FT_DFGM',
        'FGA_LT_06': 'LT6FT_DFGA',
        'LT_06_PCT': 'LT6FT_DFG%',
        'NS_LT_06_PCT': 'LT6FT_FG%',
        'PLUSMINUS': 'LT6FT_%DIFF'
    }, axis=1, inplace=True
)

defense_dashboard_10ft_defens.drop(['GP', 'G'], axis=1, inplace=True)
defense_dashboard_10ft_defens.rename(
    {
        'FREQ': 'LT10FT_FREQ',
        'FGM_LT_10': 'LT10FT_DFGM',
        'FGA_LT_10': 'LT10FT_DFGA',
        'LT_10_PCT': 'LT10FT_DFG%',
        'NS_LT_10_PCT': 'LT10FT_FG%',
        'PLUSMINUS': 'LT10FT_%DIFF'
    }, axis=1, inplace=True
)

defense_dashboard_15ft_defense.drop(['GP', 'G'], axis=1, inplace=True)
defense_dashboard_15ft_defense.rename(
    {
        'FREQ': 'GT15FT_FREQ',
        'FGM_GT_15': 'GT15FT_DFGM',
        'FGA_GT_15': 'GT15FT_DFGA',
        'GT_15_PCT': 'GT15FT_DFG%',
        'NS_GT_15_PCT': 'GT15FT_FG%',
```

```
        'PLUSMINUS': 'GT15FT_%DIFF'
    }, axis=1, inplace=True
)

defense_dashboard_3pt_defense.drop(['GP', 'G'], axis=1, inplace=True)
defense_dashboard_3pt_defense.rename(
    {
        'FREQ': '3PT_FREQ',
        'FG3M': '3PT_DFGM',
        'FG3A': '3PT_DFGA',
        'FG3_PCT': '3PT_DFG%',
        'NS_FG3_PCT': '3PT_FG%',
        'PLUSMINUS': '3PT_%DIFF'
    }, axis=1, inplace=True
)

defense_dashboard = pd.merge(defense_dashboard_overall, defense_dashboard__6f
defense_dashboard = pd.merge(defense_dashboard, defense_dashboard_10ft_defens
defense_dashboard = pd.merge(defense_dashboard, defense_dashboard_15ft_defens
defense_dashboard = pd.merge(defense_dashboard, defense_dashboard_3pt_defense

defense_dashboard.fillna(0, inplace=True)
defense_dashboard
```

Out[ ]:

	PLAYER_NAME	PLAYER_LAST_TEAM_ABBREVIATION	PLAYER_POSITION	AGE	OVERALL_
0	Rudy Gobert	UTA	C	29	
1	Brook Lopez	MIL	C	33	
2	Domantas Sabonis	IND	F-C	25	
3	Nikola Jokic	DEN	C	26	
4	Myles Turner	IND	C-F	25	
...	...	...	...	...	
532	Theo Pinson	NYK	G-F	25	
533	Keljin Blevins	POR	G	25	
534	Grant Riller	CHA	G	24	
535	Robert Woodard II	SAC	F	21	
536	Greg Whittington	DEN	F	28	

537 rows × 34 columns

In [ ]:

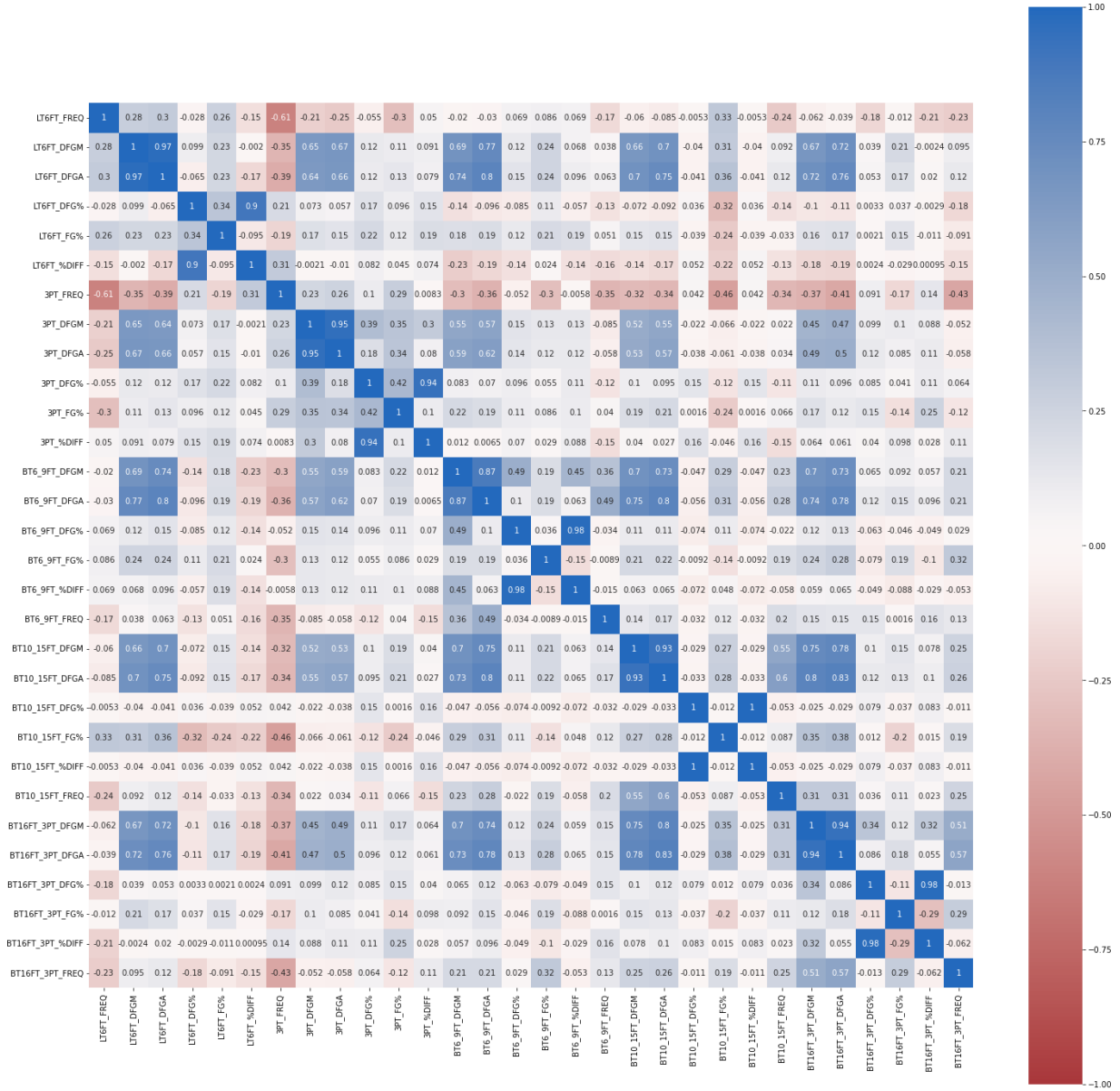
```
# defensive statistics can be reconstructed for different area
# less than 6ft, 6-9ft (paint-area)
# between 10-15ft (mid-range)
# between 16-3pt, from 3pt and beyond (perimeter)
defense_dashboard['BT6_9FT_DFGM'] = defense_dashboard['LT10FT_DFGM'] - defens
defense_dashboard['BT6_9FT_DFGA'] = defense_dashboard['LT10FT_DFGA'] - defens
defense_dashboard['BT6_9FT_DFG%'] = (defense_dashboard['BT6_9FT_DFGM'] / defe
defense_dashboard['BT6_9FT_FG%'] = defense_dashboard['LT10FT_FG%'] * 2 - defe
defense_dashboard['BT6_9FT_%DIFF'] = defense_dashboard['BT6_9FT_DFG%'] - defe
defense_dashboard['BT6_9FT_FREQ'] = (defense_dashboard['BT6_9FT_DFGA'] / defe

defense_dashboard['BT10_15FT_DFGM'] = defense_dashboard['OVERALL_DFGM'] - def
defense_dashboard['BT10_15FT_DFGA'] = defense_dashboard['OVERALL_DFGA'] - def
```

```
defense_dashboard['BT10_15FT_DFG%'] = (defense_dashboard['BT10_15FT_DFGM'] / (
defense_dashboard['BT10_15FT_FG%'] = defense_dashboard['OVERALL_FG%'] * 3 - de
defense_dashboard['BT10_15FT_%DIFF'] = defense_dashboard['BT10_15FT_DFG%'] - (
defense_dashboard['BT10_15FT_FREQ'] = (defense_dashboard['BT10_15FT_DFGA'] / (

defense_dashboard['BT16FT_3PT_DFGM'] = defense_dashboard['GT15FT_DFGM'] - defe
defense_dashboard['BT16FT_3PT_DFGA'] = defense_dashboard['GT15FT_DFGA'] - defe
defense_dashboard['BT16FT_3PT_DFG%'] = (defense_dashboard['BT16FT_3PT_DFGM']
defense_dashboard['BT16FT_3PT_FG%'] = defense_dashboard['GT15FT_FG%'] * 2 - de
defense_dashboard['BT16FT_3PT_%DIFF'] = defense_dashboard['BT16FT_3PT_DFG%'] -
defense_dashboard['BT16FT_3PT_FREQ'] = (defense_dashboard['BT16FT_3PT_DFGA']

defense_dashboard.drop(['AGE', 'OVERALL_FREQ', 'OVERALL_DFGM', 'OVERALL_DFGA'
```

In [ ]:

```
defense_dashboard_corr = defense_dashboard.corr()
plt.figure(figsize=(25,25))
sns.heatmap(defense_dashboard_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', s
```

Out[ ]: <AxesSubplot:>



In [ ]:

```
defense_dashboard.drop([
        'LT6FT_DFGA', 'LT6FT_DFG%', 'LT6FT_FG%','3PT_DFGA',
        '3PT_DFG%', '3PT_FG%', 'BT6_9FT_DFGA', 'BT6_9FT_DFG%', 'BT6_9FT_FG%',
        'BT10_15FT_DFGA', 'BT10_15FT_DFG%', 'BT10_15FT_FG%','BT16FT_3PT_DFGA',
```

```
        'BT16FT_3PT_DFG%', 'BT16FT_3PT_FG%',],
        axis=1, inplace=True)
defense_dashboard.rename({'PLAYER_LAST_TEAM_ABBREVIATION': 'TEAM_ABBREVIATION
defense_dashboard.fillna(0, inplace=True)
defense_dashboard
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	PLAYER_POSITION	LT6FT_FREQ	LT6FT_DFGM	LT
0	Rudy Gobert	UTA	C	0.373	4.10	
1	Brook Lopez	MIL	C	0.346	3.48	
2	Domantas Sabonis	IND	F-C	0.477	4.85	
3	Nikola Jokic	DEN	C	0.428	4.72	
4	Myles Turner	IND	C-F	0.502	4.48	
...	...	...	...	...	...	
532	Theo Pinson	NYK	G-F	0.375	0.33	
533	Keljin Blevins	POR	G	0.278	0.21	
534	Grant Riller	CHA	G	0.333	0.40	
535	Robert Woodard II	SAC	F	0.875	0.43	
536	Greg Whittington	DEN	F	0.000	0.00	

537 rows × 18 columns

In [ ]:

```
#  hustle df
hustle = dfs[7]
hustle.drop(['Unnamed: 0', 'PLAYER_ID', 'TEAM_ID', 'AGE', 'G', 'MIN',
       'OFF_BOXOUTS', 'DEF_BOXOUTS',
       'BOX_OUTS', 'BOX_OUT_PLAYER_TEAM_REBS', 'BOX_OUT_PLAYER_REBS',
       'PCT_BOX_OUTS_OFF', 'PCT_BOX_OUTS_DEF', 'PCT_BOX_OUTS_TEAM_REB',
       'PCT_BOX_OUTS_REB'], axis=1, inplace=True)
hustle
```

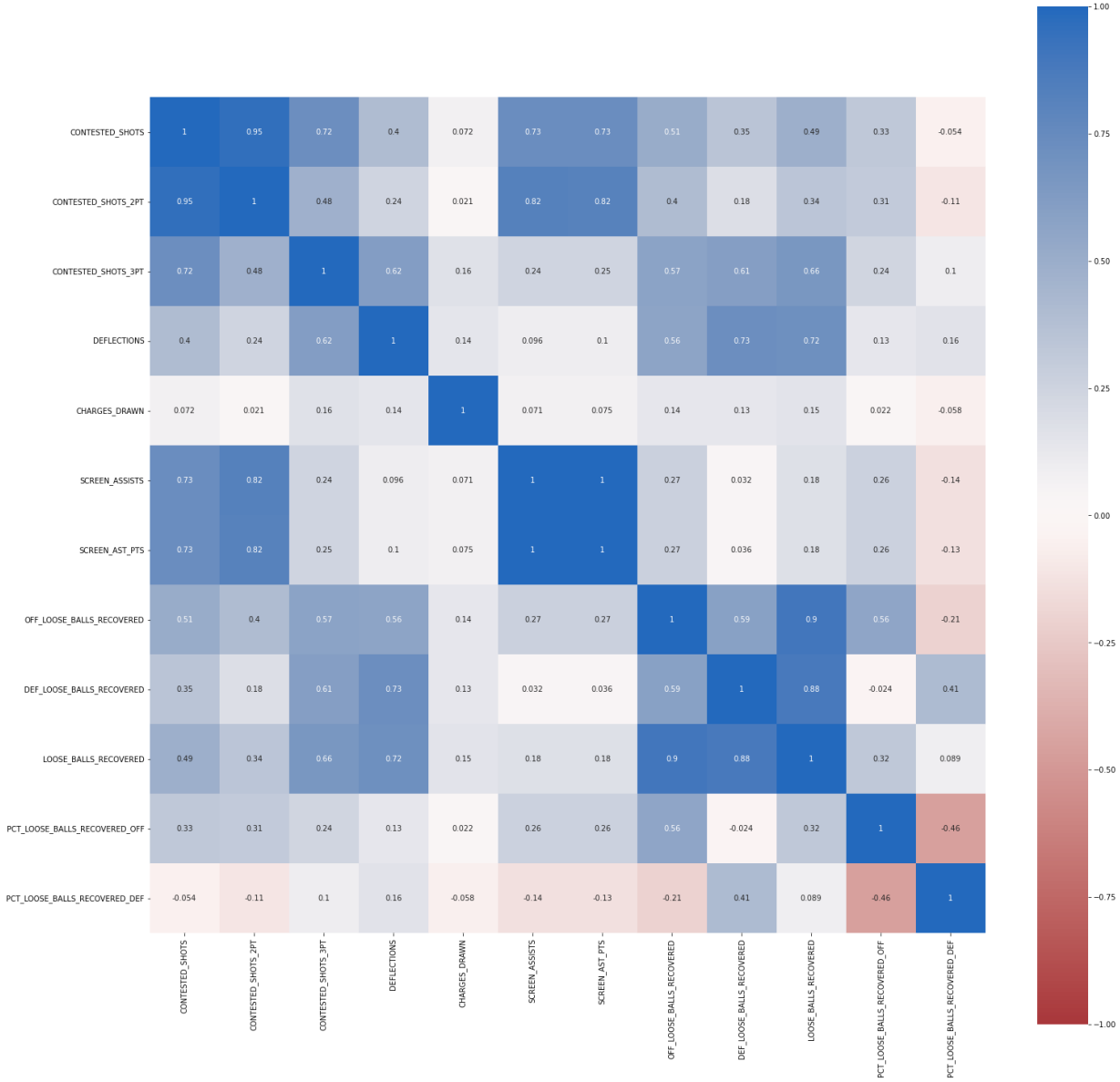
Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	CONTESTED_SHOTS	CONTESTED_SHOTS_2PT	C(
0	Aaron Gordon	DEN	3.78	2.46	
1	Aaron Holiday	IND	3.09	1.29	
2	Aaron Nesmith	BOS	2.87	1.83	
3	Abdel Nader	PHX	3.17	1.92	
4	Adam Mokoka	CHI	0.29	0.21	
...	...	...	...	...	
535	Yogi Ferrell	LAC	1.40	0.80	
536	Yuta Watanabe	TOR	3.92	1.86	
537	Zach LaVine	CHI	6.45	3.26	
538	Zeke Nnaji	DEN	2.52	1.36	
539	Zion Williamson	NOP	7.72	3.54	

540 rows × 14 columns

In [ ]:
```
hustle_corr = hustle.corr()
plt.figure(figsize=(25,25))
sns.heatmap(hustle_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', square=True
```

Out[ ]: <AxesSubplot:>



In [ ]:
```
hustle.drop(['CONTESTED_SHOTS', 'SCREEN_AST_PTS', 'LOOSE_BALLS_RECOVERED',
             'PCT_LOOSE_BALLS_RECOVERED_OFF', 'PCT_LOOSE_BALLS_RECOVERED_DEF']
```

In [ ]:
```
# shooting df
shooting = dfs[8]
shooting.drop(['Unnamed: 0', 'PLAYER_ID', 'TEAM_ID', 'NICKNAME', 'AGE', 'FGM.
               'FGA.6', 'FG_PCT.6', 'FGM.7', 'FGA.7', 'FG_PCT.7', 'FGM.8', 'FG
shooting.rename({
       'FGM': 'LT_5FT_FGM',
       'FGA': 'LT_5FT_FGA',
       'FG_PCT': 'LT_5FT_FG%',
       'FGM.1': 'BT5_9FT_FGM',
       'FGA.1': 'BT5_9FT_FGA',
       'FG_PCT.1': 'BT5_9FT_FG%',
       'FGM.2': 'BT10_14FT_FGM',
```

```
        'FGA.2': 'BT10_14FT_FGA',
        'FG_PCT.2': 'BT10_14FT_FG%',
        'FGM.3': 'BT15_19FT_FGM',
        'FGA.3': 'BT15_19FT_FGA',
        'FG_PCT.3': 'BT15_19FT_FG%',
        'FGM.4': 'BT20_24FT_FGM',
        'FGA.4': 'BT20_24FT_FGA',
        'FG_PCT.4': 'BT20_24FT_FG%',
        'FGM.5': 'BT25_29FT_FGM',
        'FGA.5': 'BT25_29FT_FGA',
        'FG_PCT.5': 'BT25_29FT_FG%',
}, axis=1, inplace=True)
shooting.fillna(0, inplace=True)
shooting
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	LT_5FT_FGM	LT_5FT_FGA	LT_5FT_FG%	BT5_9FT
0	Aaron Gordon	DEN	2.2	3.5	0.631	
1	Aaron Holiday	IND	1.0	2.2	0.426	
2	Aaron Nesmith	BOS	0.6	1.0	0.617	
3	Abdel Nader	PHX	1.3	2.3	0.564	
4	Adam Mokoka	CHI	0.5	0.8	0.667	
...	...	...	...	...	...	
535	Yogi Ferrell	LAC	0.4	0.8	0.571	
536	Yuta Watanabe	TOR	0.6	1.1	0.593	
537	Zach LaVine	CHI	4.5	7.1	0.636	
538	Zeke Nnaji	DEN	0.6	0.8	0.710	
539	Zion Williamson	NOP	9.3	14.3	0.654	

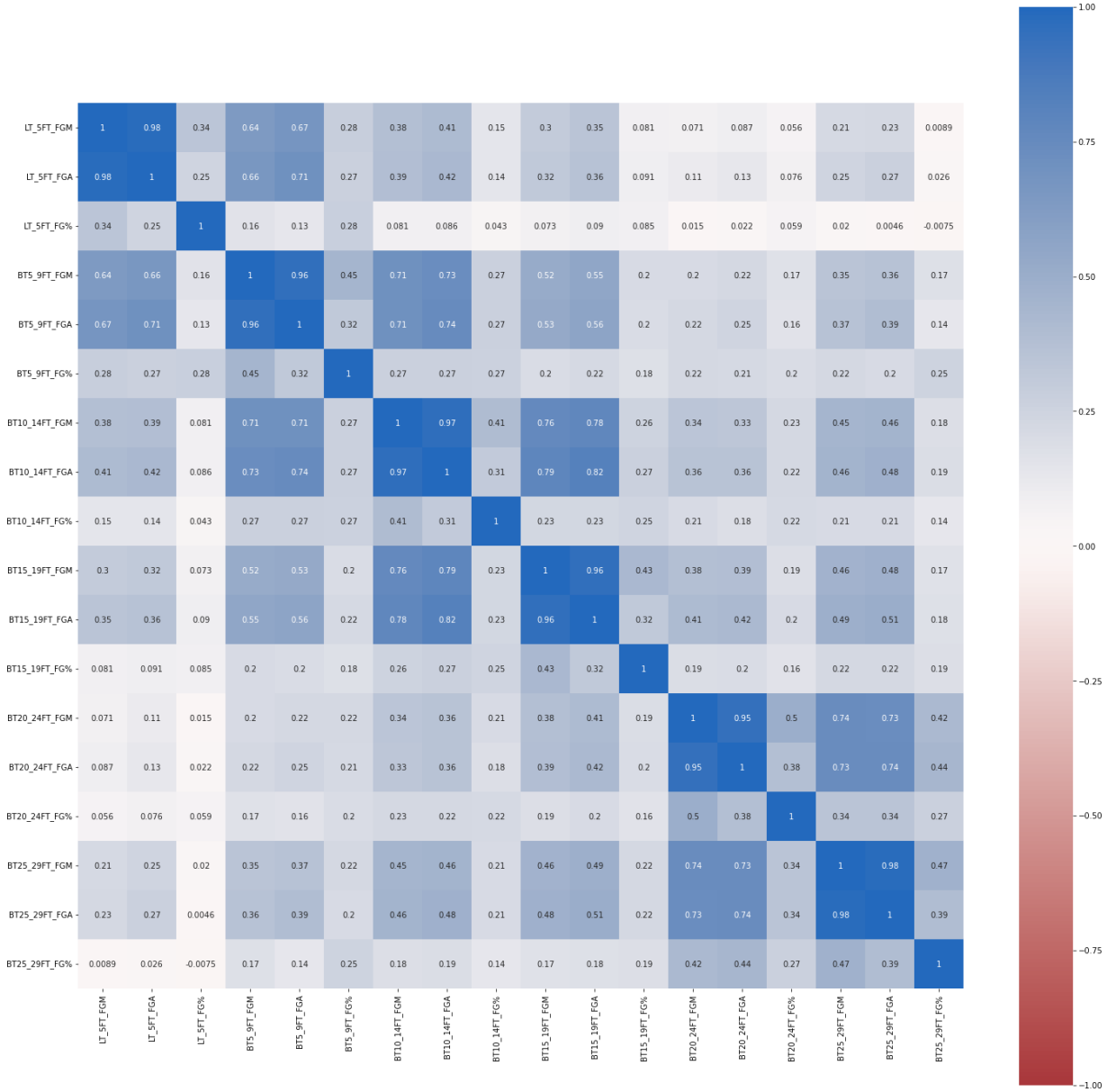
540 rows × 20 columns

In [ ]:

```
shooting_corr = shooting.corr()
plt.figure(figsize=(25,25))
sns.heatmap(shooting_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', square=Tr
```

Out[ ]:

```
<AxesSubplot:>
```



In [ ]:
```
shooting.drop(['LT_5FT_FGA','BT5_9FT_FGA','BT10_14FT_FGA',
               'BT15_19FT_FGA', 'BT20_24FT_FGA','BT25_29FT_FGA'], axis=1, inplac
```

In [ ]:
```
# playtype_isolation df
playtype_isolation = dfs[9]
playtype_isolation.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID', '
     'GP', 'PLAY_TYPE', 'TYPE_GROUPING', 'FGMX'], axis=1, inplace=True)
playtype_isolation
```

Out[ ]:

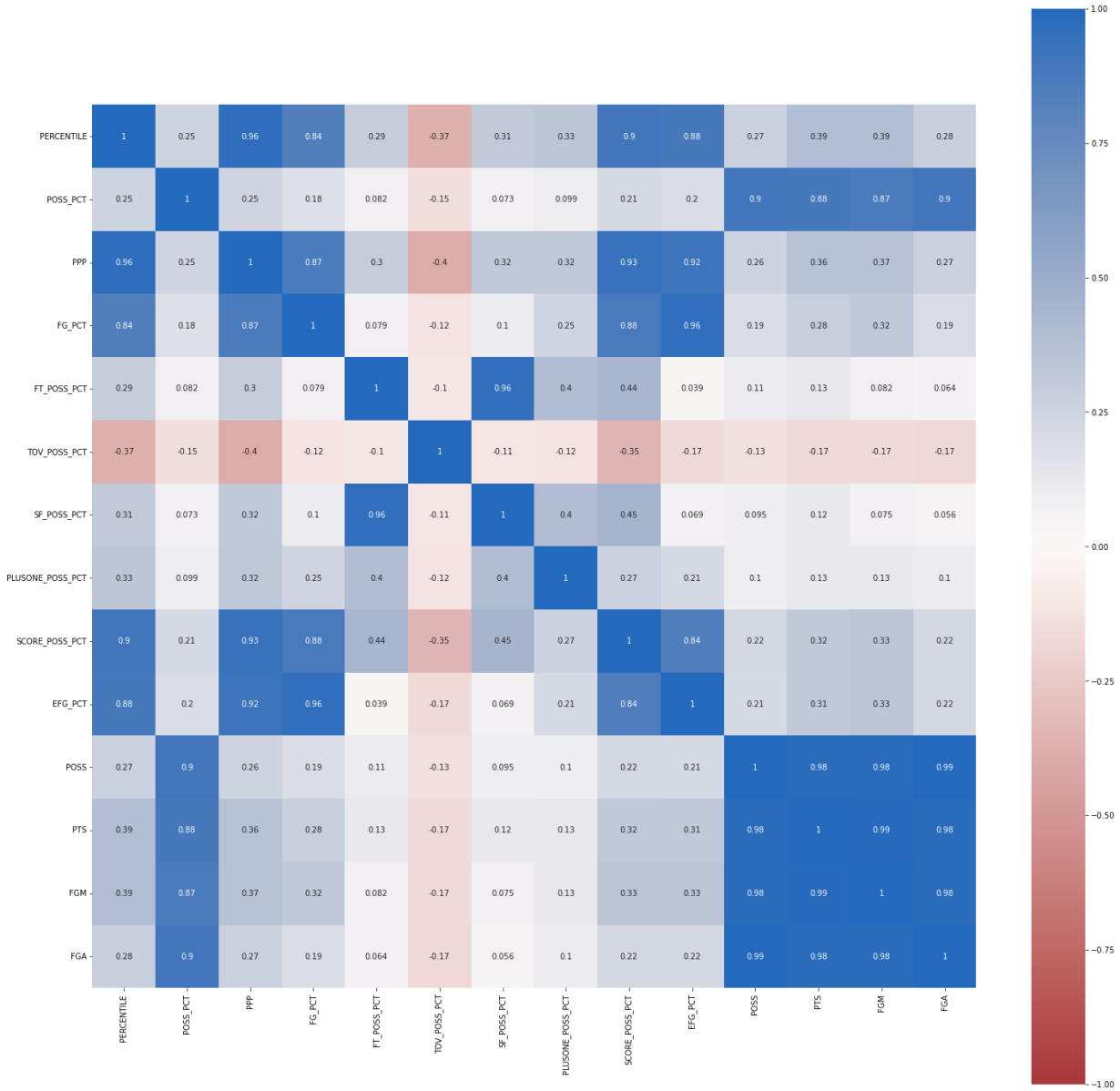
	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
0	Damian Lillard	POR	0.895	0.191	1.106	0.442	
1	Julius Randle	NYK	0.581	0.227	0.901	0.420	
2	Russell Westbrook	WAS	0.335	0.240	0.783	0.395	
3	Luka Doncic	DAL	0.802	0.167	1.046	0.463	
4	James Harden	BKN	0.875	0.336	1.091	0.435	
...	...	...	...	...	...	...	...
241	Seth Curry	PHI	0.020	0.022	0.286	0.077	

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
242	Denzel Valentine	CHI	0.016	0.034	0.250	0.133	
243	Patrick Beverley	LAC	0.012	0.065	0.222	0.000	
244	Draymond Green	GSW	0.004	0.017	0.200	0.143	
245	Larry Nance Jr.	CLE	0.000	0.031	0.000	0.000	

246 rows × 16 columns

In [ ]:
```
playtype_isolation_corr = playtype_isolation.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_isolation_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r',
```

Out[ ]: <AxesSubplot:>



In [ ]:
```
playtype_isolation.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_PC
                         'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PT
```

In [ ]:

```
playtype_isolation.rename({
    'PPP': 'ISO_PPP',
    'FG_PCT': 'ISO_FG%',
    'POSS': 'ISO_POSS'
}, axis=1, inplace=True)
```

In [ ]:

```
# playtype_pnr_ball_handler df
playtype_pnr_ball_handler = dfs[10]
playtype_pnr_ball_handler.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM
                                'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], a
playtype_pnr_ball_handler
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
0	Trae Young	ATL	0.773	0.562	0.980	0.435	
1	Luka Doncic	DAL	0.827	0.477	1.006	0.491	
2	Damian Lillard	POR	0.909	0.463	1.073	0.443	
3	Zach LaVine	CHI	0.820	0.435	1.000	0.478	
4	De'Aaron Fox	SAC	0.691	0.441	0.939	0.461	
...	...	...	...	...	...	...	
304	Langston Galloway	PHX	0.021	0.084	0.429	0.273	
305	Austin Rivers	DEN	0.003	0.146	0.316	0.200	
306	Ignas Brazdeikis	ORL	0.033	0.104	0.500	0.333	
307	Victor Oladipo	MIA	0.006	0.231	0.333	0.222	
308	Tim Frazier	MEM	0.012	0.407	0.364	0.200	

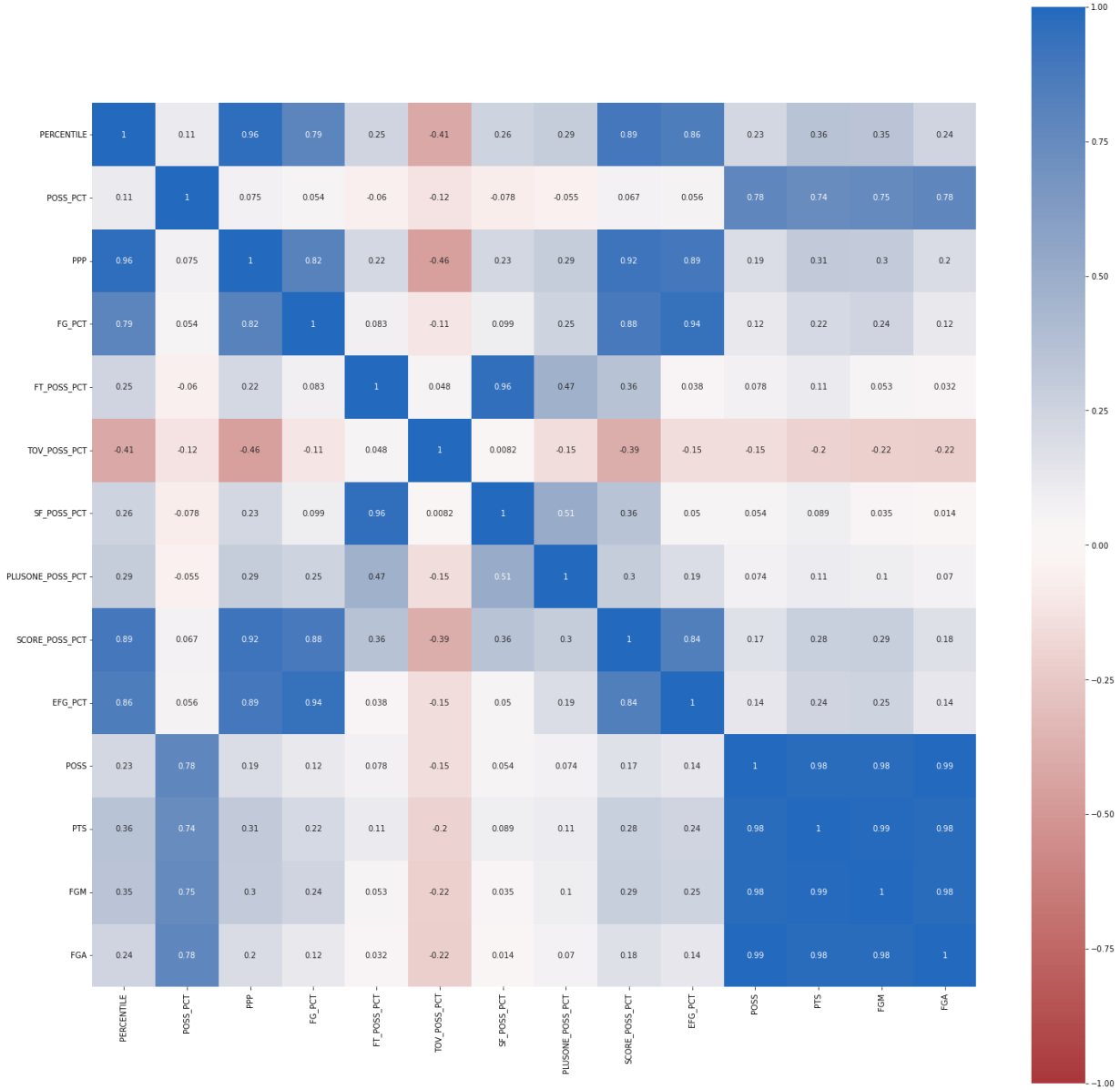
309 rows × 16 columns

In [ ]:

```
playtype_pnr_ball_handler_corr = playtype_pnr_ball_handler.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_pnr_ball_handler_corr, vmin=-1, vmax=1,center=0, cmap='v
```

Out[ ]:

```
<AxesSubplot:>
```



In [ ]:
```
playtype_pnr_ball_handler.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_pnr_ball_handler.rename({
    'PPP': 'PNR_HANDLER_PPP',
    'FG_PCT': 'PNR_HANDLER_FG%',
    'POSS': 'PNR_HANDLER_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# playtype_pnr_roll_man df
playtype_pnr_roll_man = dfs[11]
playtype_pnr_roll_man.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID'
                            'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], axis=
playtype_pnr_roll_man
```

Out[ ]:

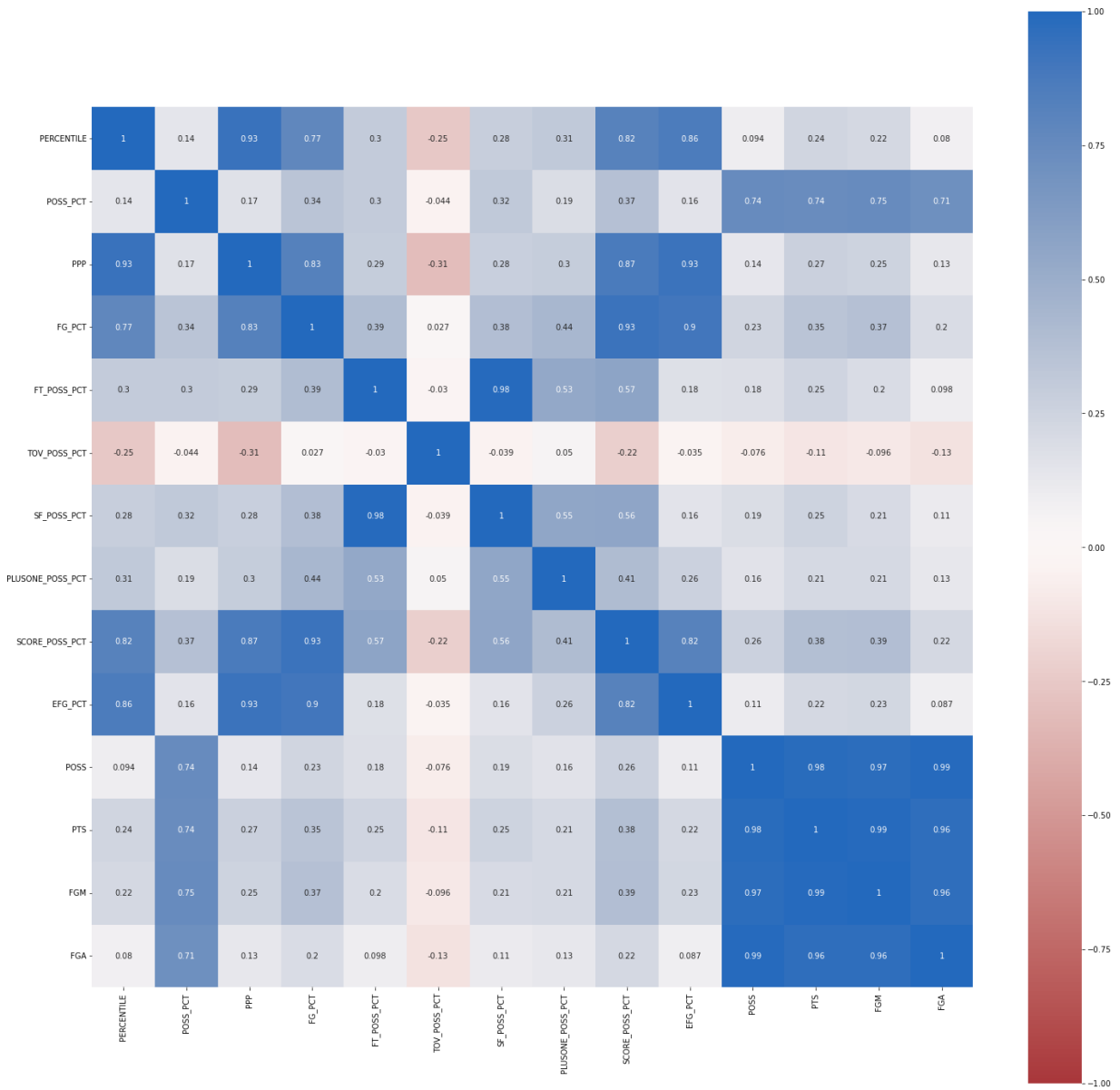
	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
0	Rudy Gobert	UTA	0.866	0.317	1.339	0.714	
1	Nikola Jokic	DEN	0.626	0.171	1.170	0.530	
2	Richaun Holmes	SAC	0.836	0.355	1.301	0.642	

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
3	Nikola Vucevic	ORL	0.542	0.229	1.135	0.489	
4	Bam Adebayo	MIA	0.849	0.171	1.323	0.677	
...	...	...	...	...	...	...	...
223	Trey Burke	DAL	0.021	0.024	0.500	0.222	
224	Marcus Smart	BOS	0.017	0.016	0.364	0.000	
225	Devonte' Graham	CHA	0.008	0.016	0.231	0.077	
226	Nicolo Melli	NOP	0.004	0.211	0.200	0.083	
227	Kentavious Caldwell-Pope	LAL	0.000	0.018	0.182	0.143	

228 rows × 16 columns

In [ ]:
```
playtype_pnr_roll_man_corr = playtype_pnr_roll_man.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_pnr_roll_man_corr, vmin=-1, vmax=1,center=0, cmap='vlag_
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:
```
playtype_pnr_roll_man.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_pnr_roll_man.rename({
    'PPP': 'PNR_ROLLMAN_PPP',
    'FG_PCT': 'PNR_ROLLMAN_FG%',
    'POSS': 'PNR_ROLLMAN_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# playtype_transition df
playtype_transition = dfs[12]
playtype_transition.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID',
                          'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], axis=1,
playtype_transition
```

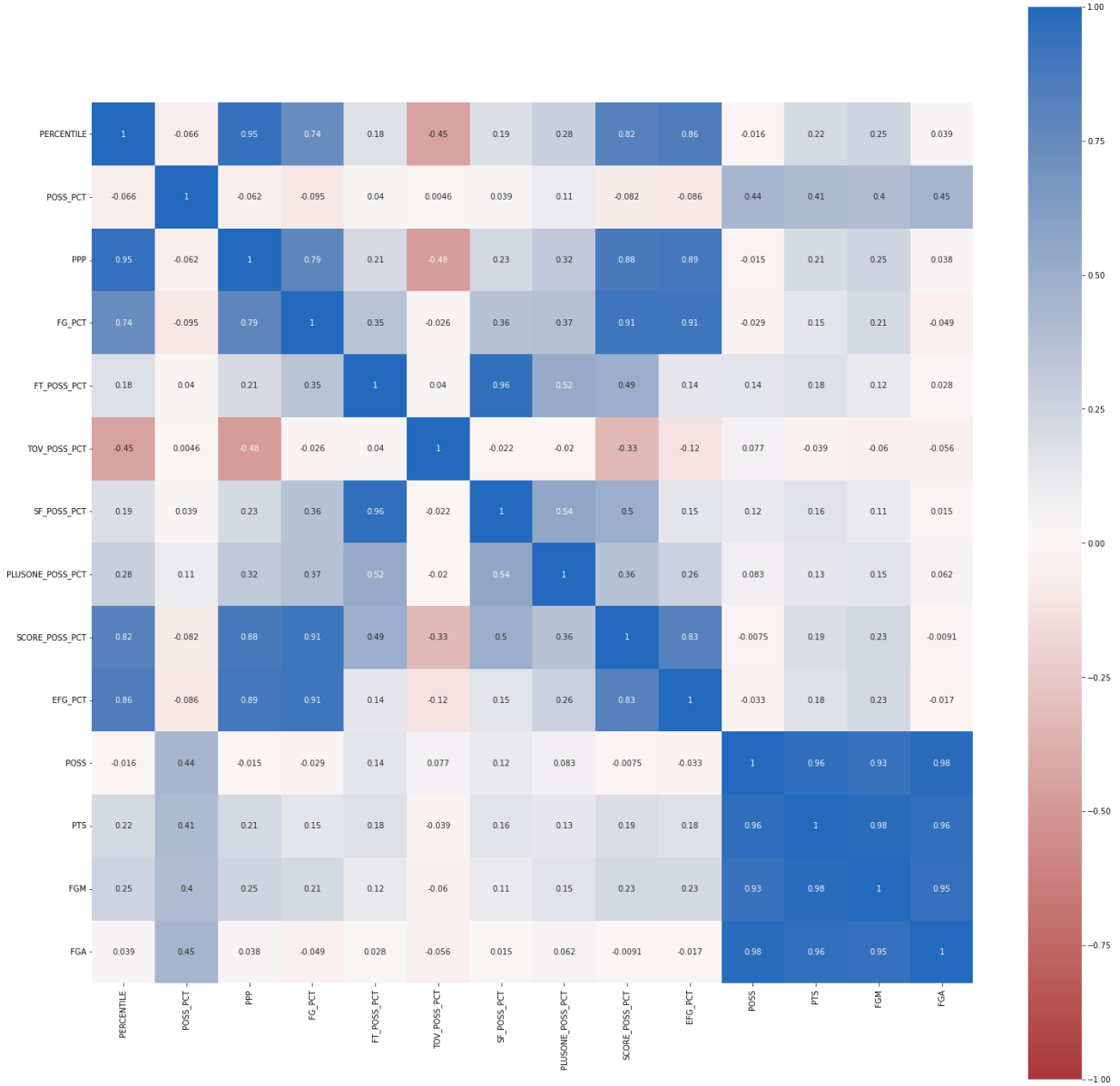
Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
0	Giannis Antetokounmpo	MIL	0.748	0.264	1.234	0.636	
1	Russell Westbrook	WAS	0.201	0.225	0.943	0.529	
2	Terry Rozier	CHA	0.679	0.220	1.201	0.542	
3	Jaylen Brown	BOS	0.551	0.224	1.143	0.573	
4	Devin Booker	PHX	0.535	0.182	1.137	0.500	
...	...	...	...	...	...	...	
446	Nicolo Melli	DAL	0.068	0.104	0.800	0.375	
447	Ignas Brazdeikis	ORL	0.021	0.125	0.667	0.300	
448	P.J. Tucker	HOU	0.006	0.087	0.467	0.300	
449	Frank Ntilikina	NYK	0.000	0.140	0.286	0.100	
450	Jay Scrubb	LAC	0.004	0.267	0.417	0.182	

451 rows × 16 columns

In [ ]:
```
playtype_transition_corr = playtype_transition.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_transition_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r'
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:
```
playtype_transition.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_P
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_transition.rename({
    'PPP': 'TRANSITION_PPP',
    'FG_PCT': 'TRANSITION_FG%',
    'POSS': 'TRANSITION_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# playtype_post_up df
playtype_post_up = dfs[13]
playtype_post_up.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID', 'TE
                       'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], axis=1, i
playtype_post_up
```

Out[ ]:

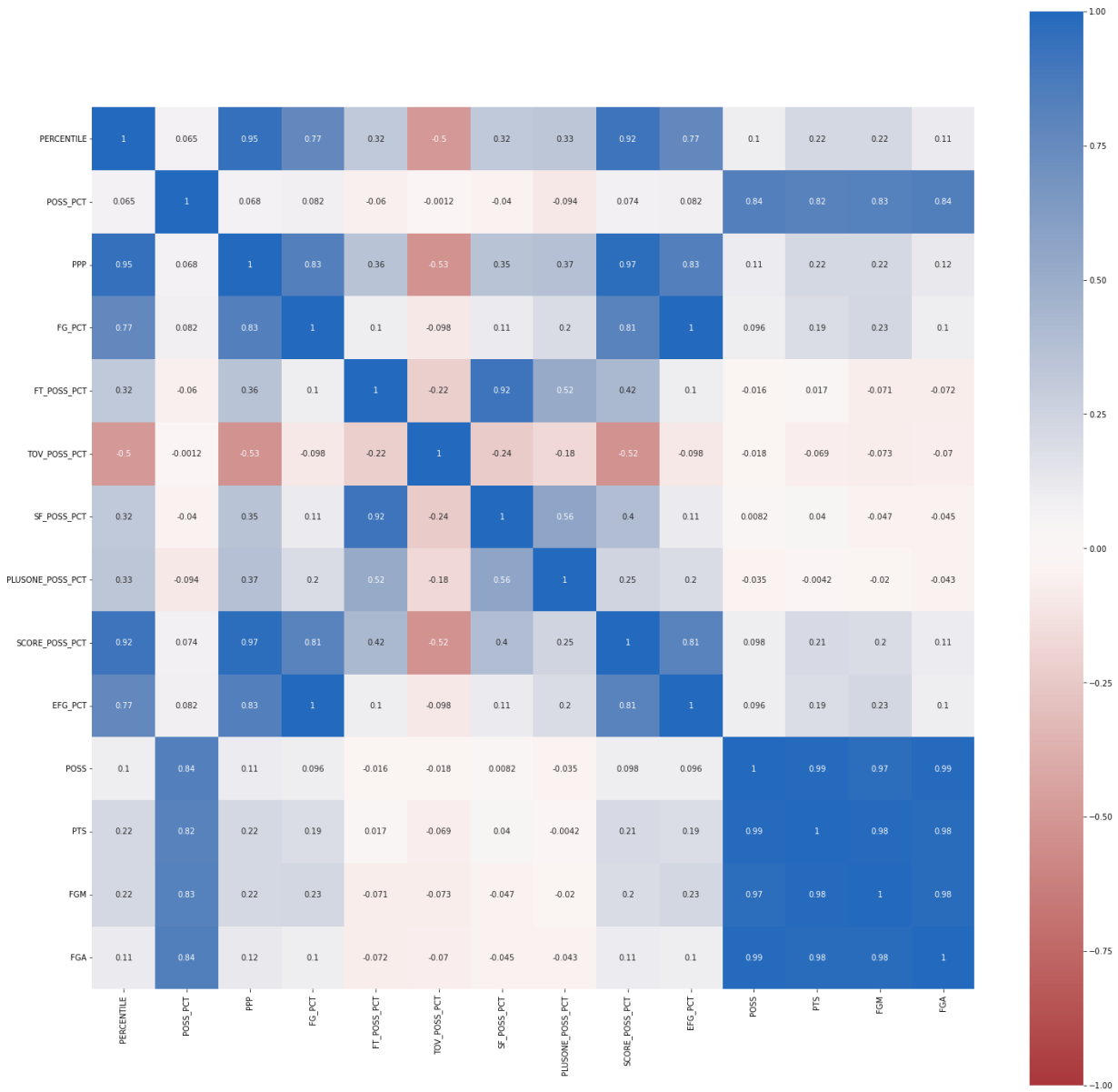
	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POSS
0	Joel Embiid	PHI	0.831	0.365	1.080	0.526	
1	Nikola Jokic	DEN	0.725	0.246	1.039	0.556	
2	Julius Randle	NYK	0.469	0.173	0.930	0.467	

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POSS
3	Jonas Valanciunas	MEM	0.706	0.258	1.033	0.578	
4	Deandre Ayton	PHX	0.488	0.275	0.936	0.500	
...	...	...	...	...	...	...	...
150	Gorgui Dieng	MEM	0.156	0.073	0.727	0.333	
151	Aron Baynes	TOR	0.094	0.035	0.615	0.364	
152	Bismack Biyombo	CHA	0.019	0.044	0.438	0.333	
153	Zach LaVine	CHI	0.013	0.007	0.400	0.222	
154	JaVale McGee	DEN	0.000	0.124	0.091	0.000	

155 rows × 16 columns

In [ ]:
```
playtype_post_up_corr = playtype_post_up.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_post_up_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', s
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:
```
playtype_post_up.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_PCT'
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_post_up.rename({
    'PPP': 'POST_UP_PPP',
    'FG_PCT': 'POST_UP_FG%',
    'POSS': 'POST_UP_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# playtype_spot_up df
playtype_spot_up = dfs[14]
playtype_spot_up.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID', 'TE
                       'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], axis=1, i
playtype_spot_up
```

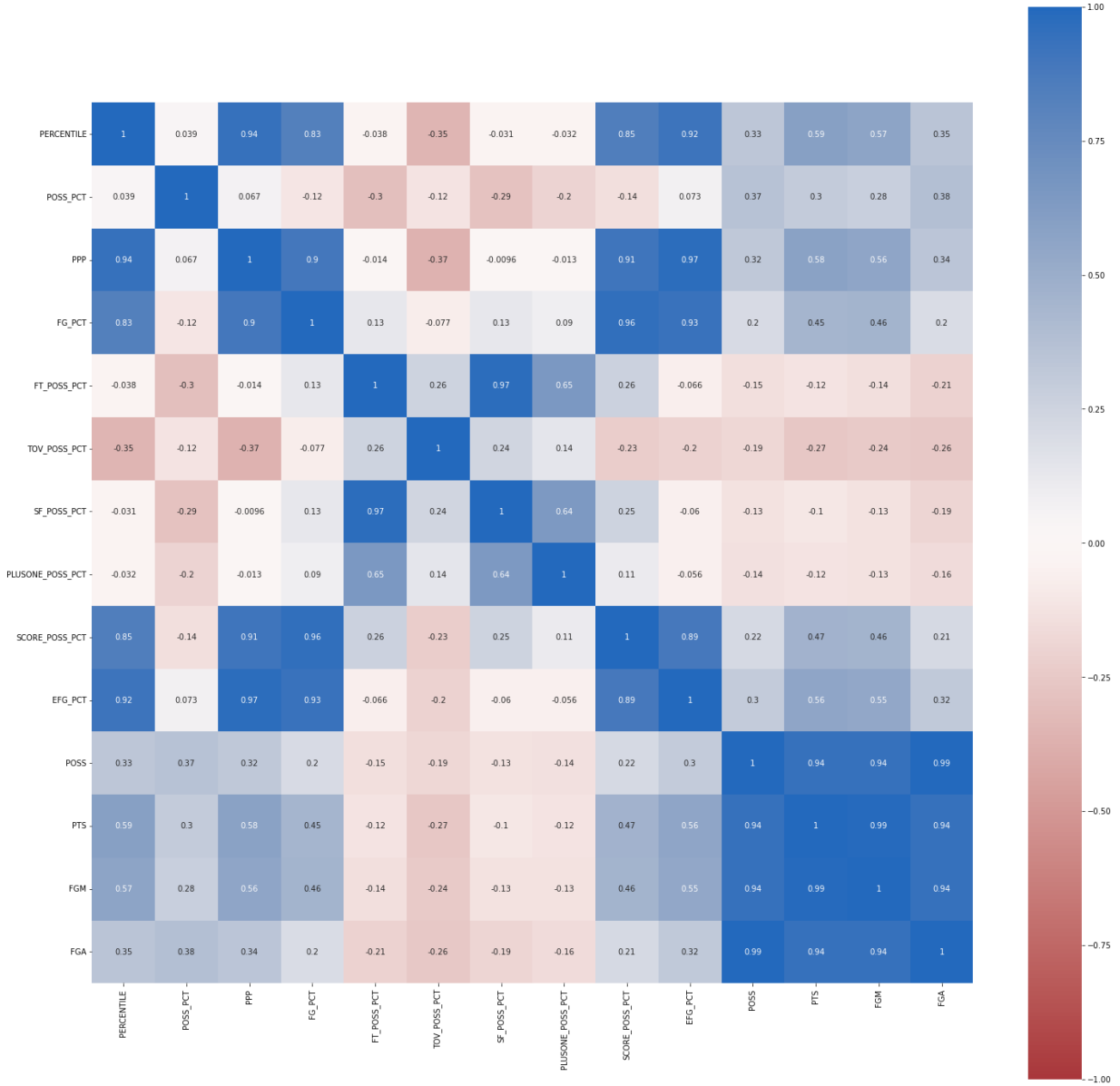
Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
0	Bojan Bogdanovic	UTA	0.732	0.316	1.111	0.426	
1	Julius Randle	NYK	0.770	0.205	1.135	0.445	
2	RJ Barrett	NYK	0.690	0.279	1.092	0.415	
3	Marcus Morris Sr.	LAC	0.990	0.404	1.426	0.524	
4	Saddiq Bey	DET	0.712	0.422	1.102	0.382	
...	...	...	...	...	...	...	...
461	Markelle Fultz	ORL	0.054	0.094	0.583	0.250	
462	Chris Chiozza	BKN	0.018	0.116	0.385	0.182	
463	Khyri Thomas	HOU	0.006	0.211	0.250	0.133	
464	Jabari Parker	BOS	0.008	0.180	0.273	0.100	
465	Derrick Favors	UTA	0.004	0.032	0.200	0.100	

466 rows × 16 columns

In [ ]:
```
playtype_spot_up_corr = playtype_spot_up.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_spot_up_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', s
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:
```
playtype_spot_up.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_PCT'
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_spot_up.rename({
    'PPP': 'SPOT_UP_PPP',
    'FG_PCT': 'SPOT_UP_FG%',
    'POSS': 'SPOT_UP_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# playtype_handoff df
playtype_handoff = dfs[15]
playtype_handoff.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID', 'TE
                       'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], axis=1, i
playtype_handoff
```

Out[ ]:

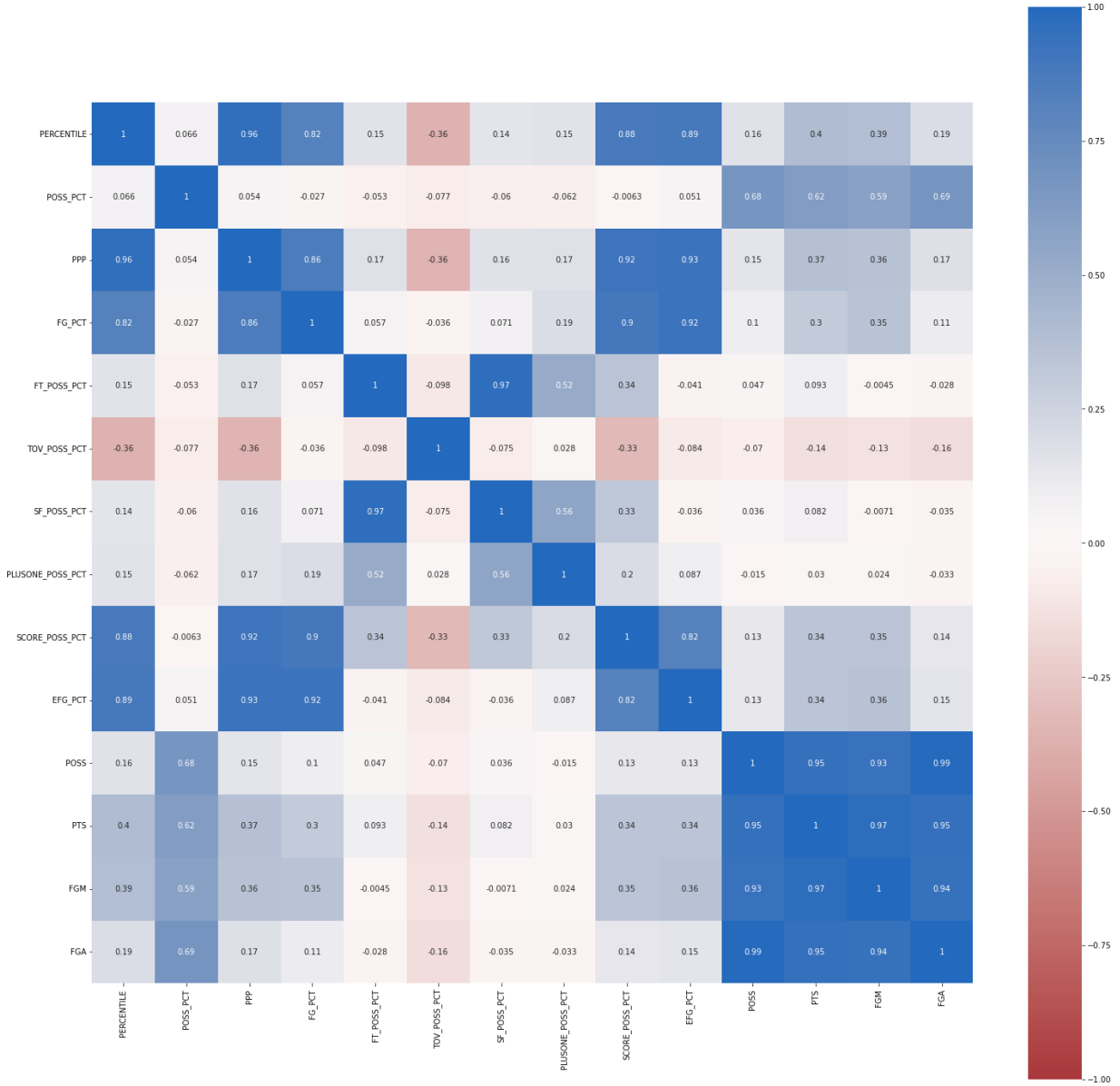
	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
0	Stephen Curry	GSW	0.922	0.118	1.266	0.485	
1	Duncan Robinson	MIA	0.724	0.236	1.062	0.385	

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
2	Doug McDermott	IND	0.795	0.218	1.097	0.458	
3	Jamal Murray	DEN	0.703	0.158	1.053	0.462	
4	Bradley Beal	WAS	0.594	0.087	0.993	0.471	
...	...	...	...	...	...	...	...
268	Naz Reid	MIN	0.025	0.014	0.400	0.143	
269	Dennis Smith Jr.	DET	0.021	0.064	0.364	0.222	
270	Brandon Goodwin	ATL	0.011	0.047	0.231	0.100	
271	Terance Mann	LAC	0.007	0.033	0.143	0.071	
272	Ben McLemore	HOU	0.000	0.057	0.000	0.000	

273 rows × 16 columns

In [ ]:
```
playtype_handoff_corr = playtype_handoff.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_handoff_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', s
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:
```
playtype_handoff.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_PCT'
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_handoff.rename({
    'PPP': 'HANDOFF_PPP',
    'FG_PCT': 'HANDOFF_FG%',
    'POSS': 'HANDOFF_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# playtype_cut df
playtype_cut = dfs[16]
playtype_cut.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID', 'TEAM_NA
                   'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], axis=1, inpla
playtype_cut
```

Out[ ]:

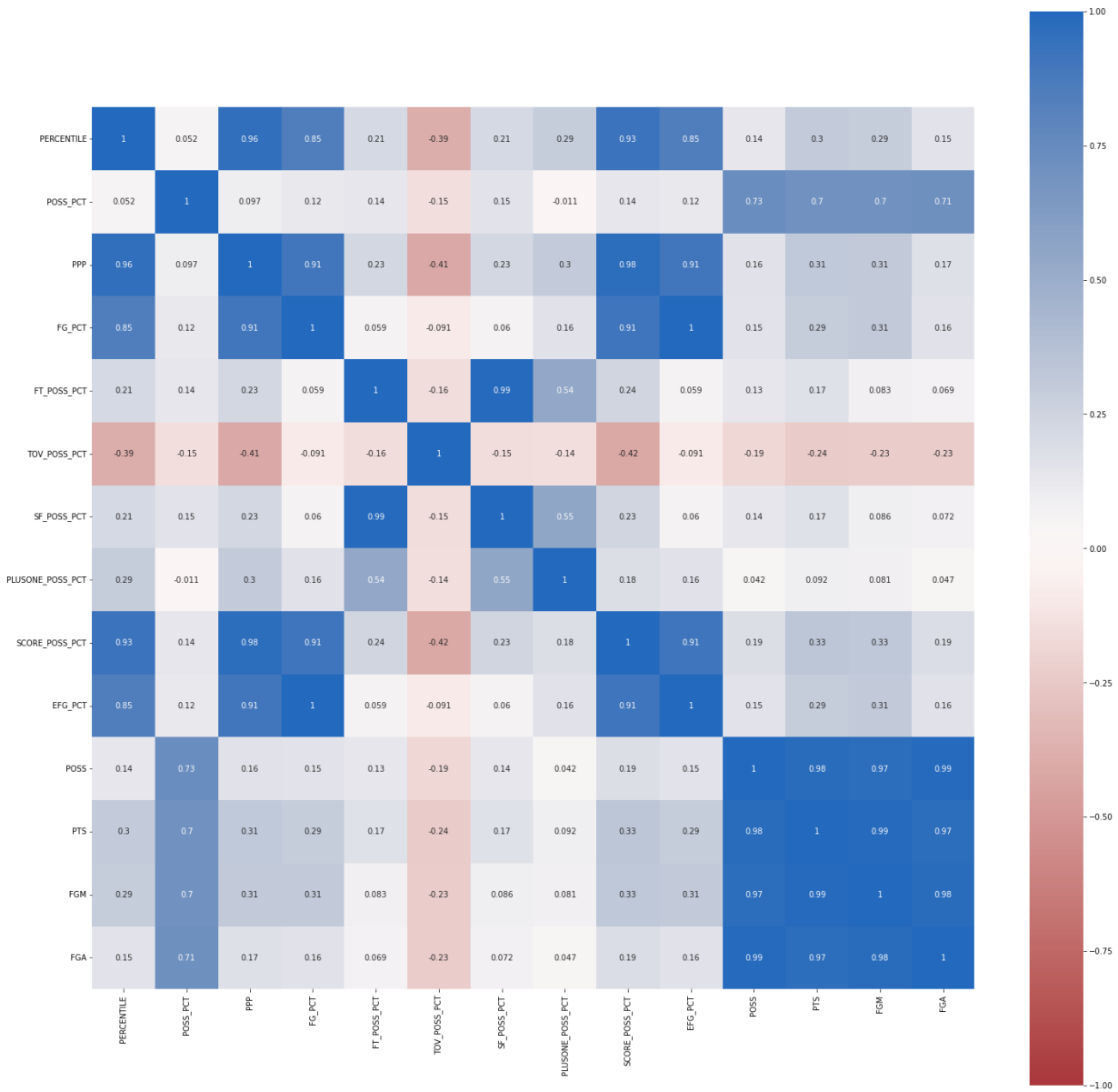
	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
0	Rudy Gobert	UTA	0.509	0.265	1.284	0.688	
1	Zion Williamson	NOP	0.778	0.140	1.420	0.715	
2	Bam Adebayo	MIA	0.337	0.204	1.197	0.595	

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
3	Deandre Ayton	PHX	0.733	0.217	1.398	0.731	
4	Montrezl Harrell	LAL	0.730	0.231	1.398	0.713	
...	...	...	...	...	...	...	...
327	R.J. Hampton	ORL	0.038	0.031	0.800	0.500	
328	Talen Horton-Tucker	LAL	0.026	0.017	0.727	0.444	
329	Al-Farouq Aminu	ORL	0.009	0.100	0.667	0.333	
330	LaMelo Ball	CHA	0.003	0.016	0.571	0.333	
331	Solomon Hill	ATL	0.000	0.036	0.538	0.273	

332 rows × 16 columns

In [ ]:
```
playtype_cut_corr = playtype_cut.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_cut_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', squar
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:
```
playtype_cut.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_PCT', 'S
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_cut.rename({
    'PPP': 'CUT_PPP',
    'FG_PCT': 'CUT_FG%',
    'POSS': 'CUT_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# playtype_off_screen df
playtype_off_screen = dfs[17]
playtype_off_screen.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID',
                          'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], axis=
playtype_off_screen
```

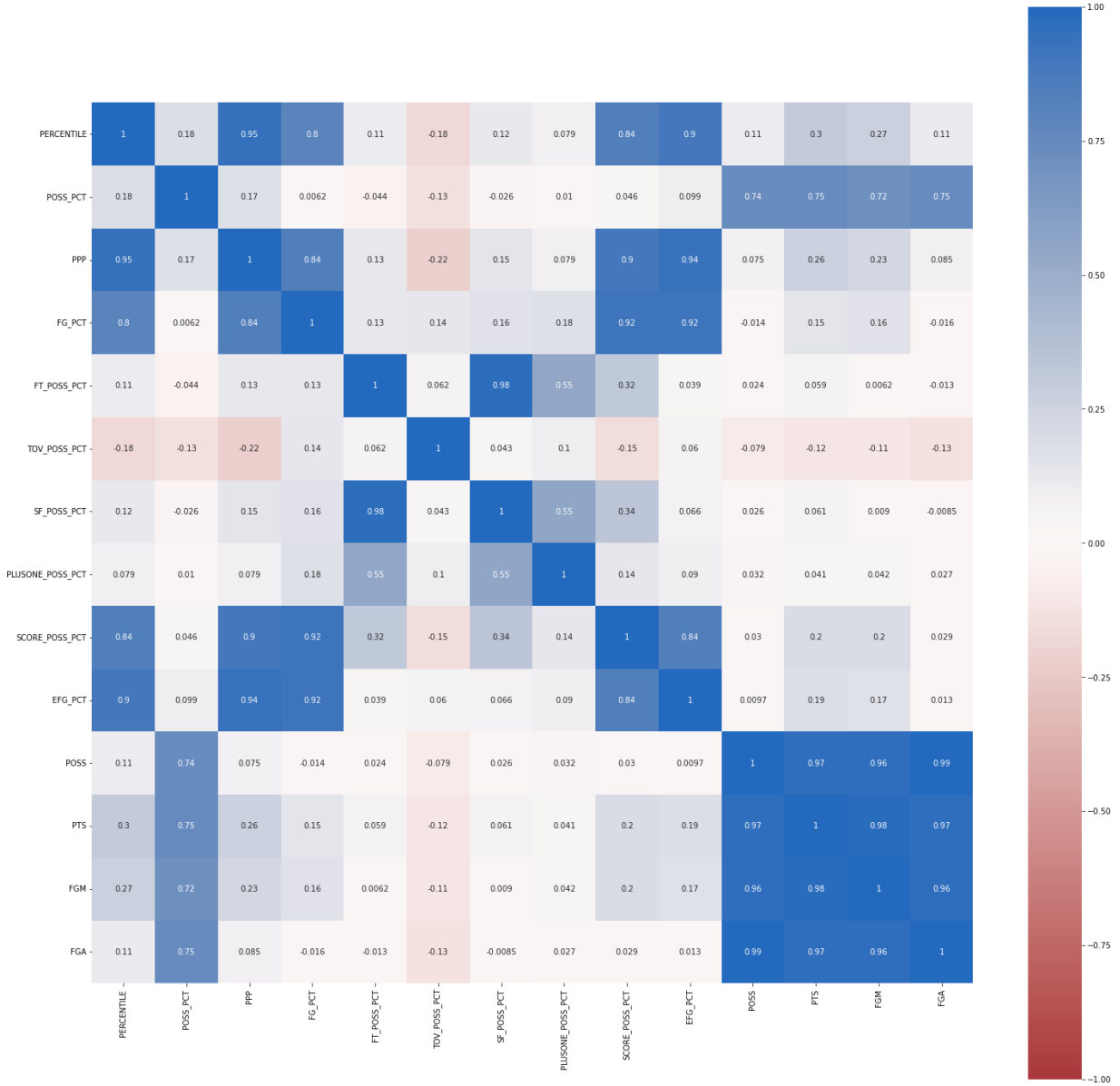
Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POS
0	Stephen Curry	GSW	0.624	0.158	1.048	0.402	
1	Bradley Beal	WAS	0.670	0.136	1.089	0.444	
2	Devin Booker	PHX	0.510	0.112	0.978	0.466	
3	Buddy Hield	SAC	0.405	0.169	0.918	0.335	
4	Duncan Robinson	MIA	0.881	0.167	1.299	0.469	
...	...	...	...	...	...	...	
200	Garrett Temple	CHI	0.014	0.033	0.400	0.182	
201	Tyler Johnson	BKN	0.033	0.053	0.455	0.200	
202	Cole Anthony	ORL	0.005	0.014	0.200	0.100	
203	James Harden	BKN	0.005	0.012	0.200	0.111	
204	Jae'Sean Tate	HOU	0.000	0.017	0.154	0.100	

205 rows × 16 columns

In [ ]:
```
playtype_off_screen_corr = playtype_off_screen.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_off_screen_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r'
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:
```
playtype_off_screen.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_P
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_off_screen.rename({
    'PPP': 'OFF_SCREEN_PPP',
    'FG_PCT': 'OFF_SCREEN_FG%',
    'POSS': 'OFF_SCREEN_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# playtype_putback df
playtype_putback = dfs[18]
playtype_putback.drop(['Unnamed: 0', 'SEASON_ID', 'PLAYER_ID', 'TEAM_ID', 'TE
                       'GP', 'PLAY_TYPE', 'TYPE_GROUPING','FGMX'], axis=1, i
playtype_putback
```

Out[ ]:

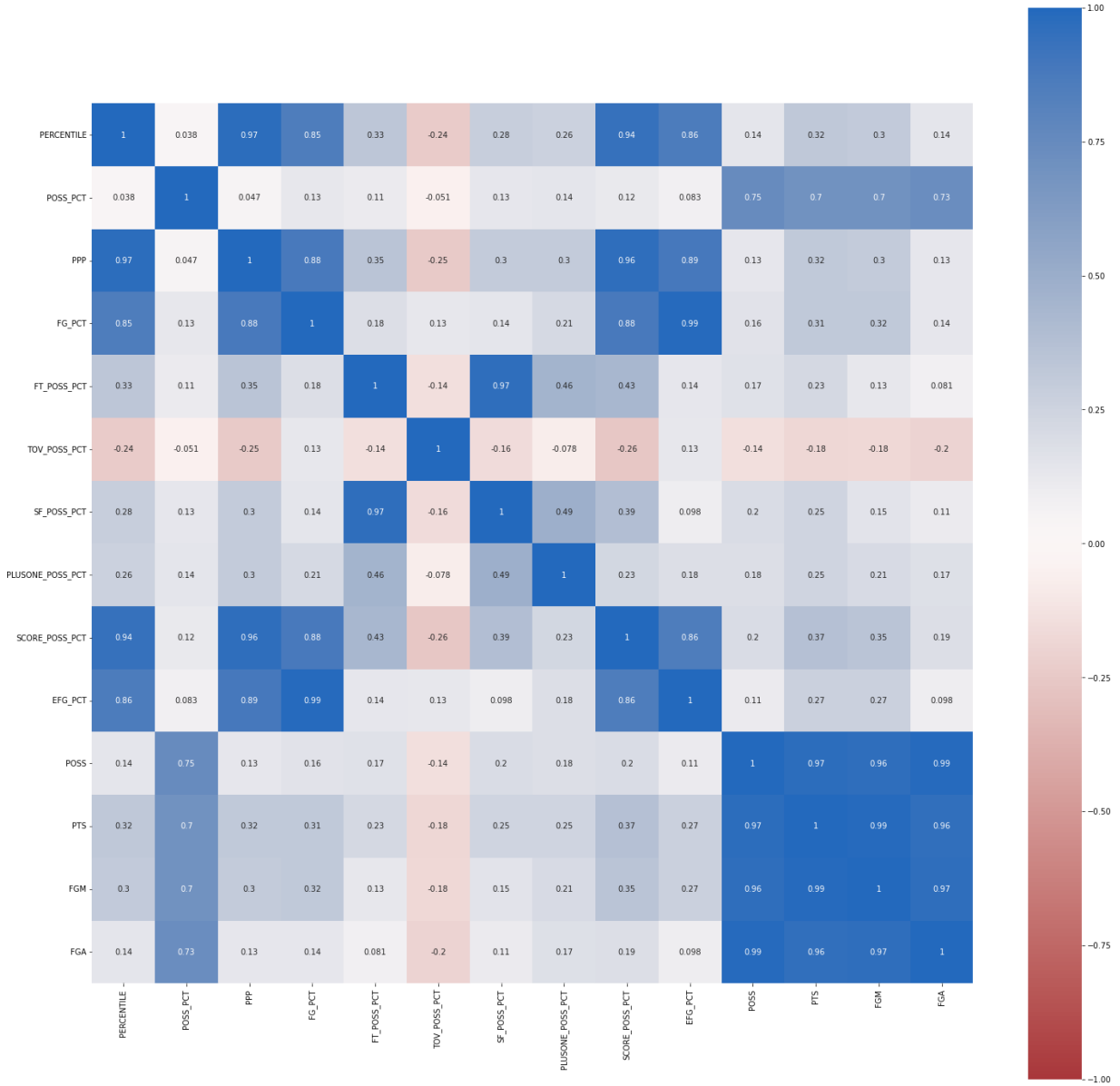
	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POSS
0	Enes Freedom	POR	0.643	0.342	1.177	0.596	
1	Jonas Valanciunas	MEM	0.777	0.224	1.268	0.644	
2	Clint Capela	ATL	0.519	0.273	1.105	0.565	

	PLAYER_NAME	TEAM_ABBREVIATION	PERCENTILE	POSS_PCT	PPP	FG_PCT	FT_POSS
3	Nikola Jokic	DEN	0.880	0.096	1.346	0.649	
4	Rudy Gobert	UTA	0.808	0.192	1.289	0.672	
...	...	...	...	...	...	...	
275	Taurean Prince	CLE	0.017	0.039	0.667	0.333	
276	T.J. McConnell	IND	0.017	0.019	0.667	0.444	
277	Garrett Temple	CHI	0.041	0.022	0.700	0.333	
278	Naji Marshall	NOP	0.007	0.043	0.583	0.333	
279	Caleb Martin	CHA	0.003	0.044	0.500	0.182	

280 rows × 16 columns

In [ ]:
```
playtype_putback_corr = playtype_putback.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_putback_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', s
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:
```
playtype_putback.drop(['PERCENTILE', 'POSS_PCT','FT_POSS_PCT', 'TOV_POSS_PCT'
       'PLUSONE_POSS_PCT', 'SCORE_POSS_PCT', 'EFG_PCT', 'PTS', 'FGM', 'FGA'],
```

In [ ]:
```
playtype_putback.rename({
    'PPP': 'PUTBACK_PPP',
    'FG_PCT': 'PUTBACK_FG%',
    'POSS': 'PUTBACK_POSS'
}, axis=1, inplace=True)
```

In [ ]:
```
# merging all playtypes to have a comprehensive overview of players' playstyl

playtype = playtype_isolation.merge(playtype_pnr_ball_handler, how='outer')
playtype = playtype.merge(playtype_pnr_roll_man, how='outer')
playtype = playtype.merge(playtype_transition, how='outer')
playtype = playtype.merge(playtype_post_up, how='outer')
playtype = playtype.merge(playtype_spot_up, how='outer')
playtype = playtype.merge(playtype_handoff, how='outer')
playtype = playtype.merge(playtype_cut, how='outer')
playtype = playtype.merge(playtype_off_screen, how='outer')
playtype = playtype.merge(playtype_putback, how='outer')

playtype.fillna(0, inplace=True)
playtype
```

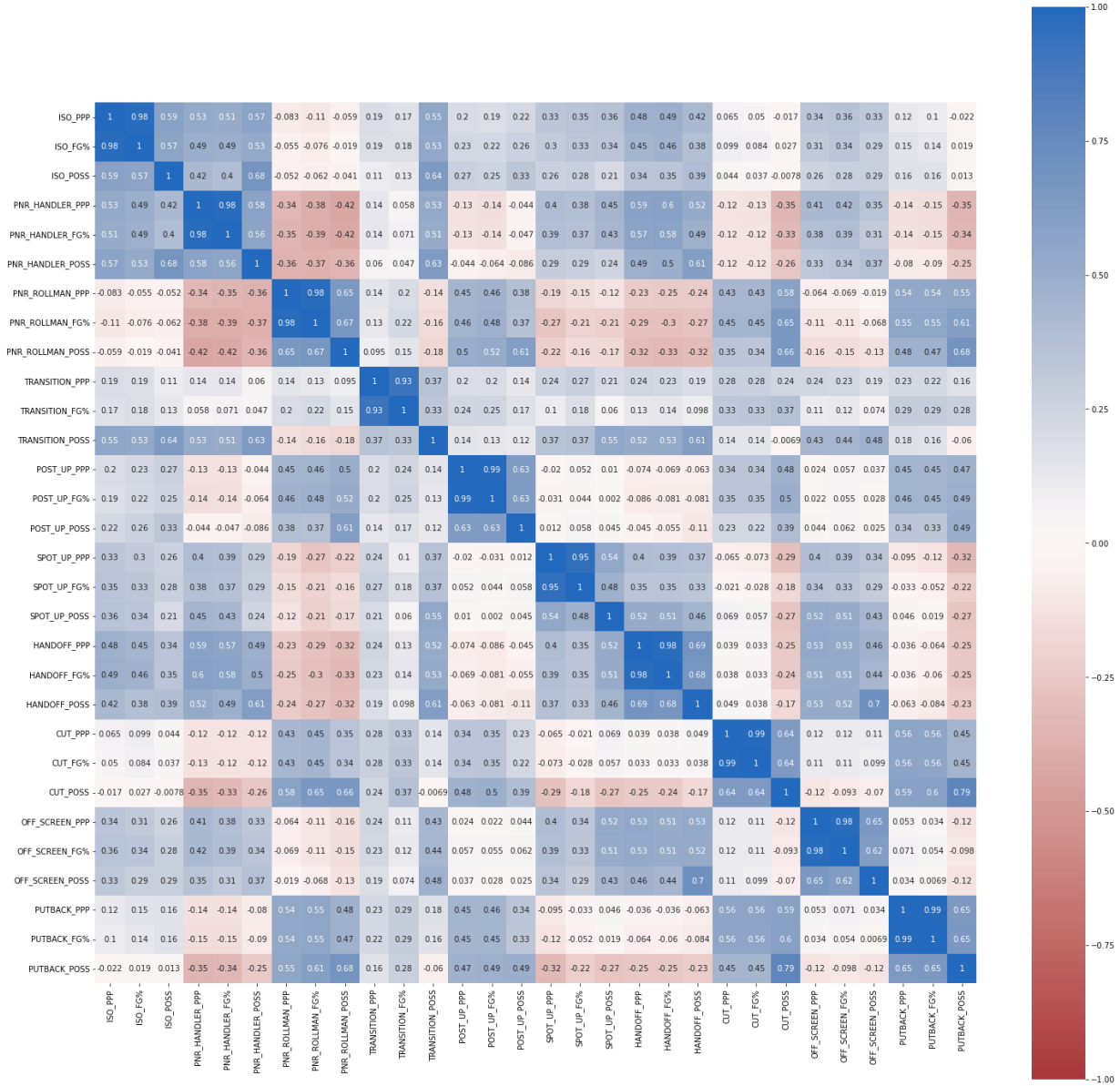
Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	ISO_PPP	ISO_FG%	ISO_POSS	PNR_HANDLER_PI
0	Damian Lillard	POR	1.106	0.442	4.9	1.0
1	Julius Randle	NYK	0.901	0.420	5.5	0.8
2	Russell Westbrook	WAS	0.783	0.395	6.4	0.7
3	Luka Doncic	DAL	1.046	0.463	4.6	1.0
4	James Harden	BKN	1.091	0.435	8.0	0.9
...	...	...	...	...	...	
492	JJ Redick	DAL	0.000	0.000	0.0	0.0
493	Cameron Oliver	HOU	0.000	0.000	0.0	0.0
494	Dante Exum	CLE	0.000	0.000	0.0	0.0
495	Damian Jones	LAL	0.000	0.000	0.0	0.0
496	Ed Davis	MIN	0.000	0.000	0.0	0.0

497 rows × 32 columns

In [ ]:
```
playtype_corr = playtype.corr()
plt.figure(figsize=(25,25))
sns.heatmap(playtype_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', square=Tr
```

Out[ ]:
```
<AxesSubplot:>
```



In [ ]:

```
# tracking_passing df
tracking_passing = dfs[19]
tracking_passing.drop(['Unnamed: 0', 'PLAYER_ID', 'TEAM_ID', 'GP', 'W', 'L',
tracking_passing
```

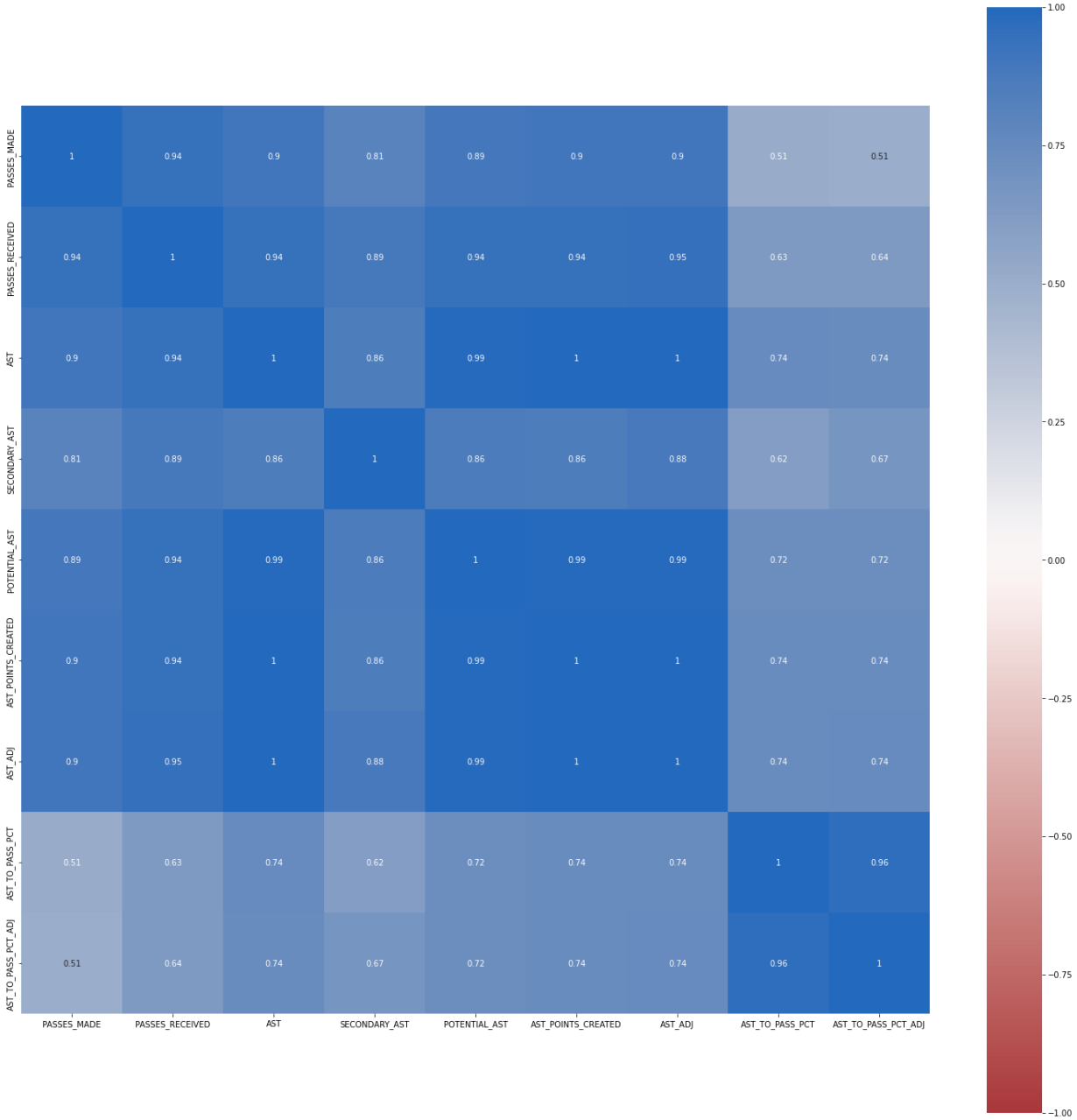
Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	PASSES_MADE	PASSES_RECEIVED	AST	SECOND
0	Aaron Gordon	DEN	32.9	34.0	3.2	
1	Aaron Holiday	IND	15.5	20.5	1.9	
2	Aaron Nesmith	BOS	11.7	9.0	0.5	
3	Abdel Nader	PHX	9.4	11.3	0.8	
4	Adam Mokoka	CHI	2.4	3.1	0.4	
...	...	...	...	...	...	
535	Yogi Ferrell	LAC	17.6	21.4	2.2	
536	Yuta Watanabe	TOR	14.4	10.9	0.8	
537	Zach LaVine	CHI	40.1	56.6	4.9	
538	Zeke Nnaji	DEN	6.6	5.9	0.2	
539	Zion Williamson	NOP	38.2	43.0	3.7	

540 rows × 11 columns

In [ ]:
```
tracking_passing_corr = tracking_passing.corr()
plt.figure(figsize=(25,25))
sns.heatmap(tracking_passing_corr, vmin=-1, vmax=1,center=0, cmap='vlag_r', s
```

Out[ ]: <AxesSubplot:>



In [ ]:
```
tracking_passing.drop(['PASSES_RECEIVED', 'AST','AST_POINTS_CREATED', 'AST_AD
```

In [ ]:
```
general_traditional.head()
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	FG_PCT	FG3_PCT	OREB	DREB	AST	STL	BLK	F
0	Aaron Gordon	DEN	0.463	0.335	1.5	4.1	3.2	0.7	0.7	1
1	Aaron Holiday	IND	0.390	0.368	0.2	1.1	1.9	0.7	0.2	1
2	Aaron Nesmith	BOS	0.438	0.370	0.6	2.2	0.5	0.3	0.2	1

	PLAYER_NAME	TEAM_ABBREVIATION	FG_PCT	FG3_PCT	OREB	DREB	AST	STL	BLK	F
3	Abdel Nader	PHX	0.491	0.419	0.3	2.3	0.8	0.4	0.4	1
4	Al Horford	BOS	0.450	0.368	1.0	5.7	3.4	0.9	0.9	1

In [ ]:
```
general_advanced.head()
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	AST_PCT	AST_TO	OREB_PCT	DREB_PCT	TM_TOV
0	Aaron Gordon	DEN	0.165	1.66	0.055	0.150	
1	Aaron Holiday	IND	0.139	1.86	0.012	0.060	
2	Aaron Nesmith	BOS	0.047	1.00	0.041	0.146	
3	Abdel Nader	PHX	0.078	1.00	0.020	0.151	
4	Adam Mokoka	CHI	0.179	1.00	0.017	0.077	

In [ ]:
```
defense_dashboard.head()
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	PLAYER_POSITION	LT6FT_FREQ	LT6FT_DFGM	LT6F
0	Rudy Gobert	UTA	C	0.373	4.10	
1	Brook Lopez	MIL	C	0.346	3.48	
2	Domantas Sabonis	IND	F-C	0.477	4.85	
3	Nikola Jokic	DEN	C	0.428	4.72	
4	Myles Turner	IND	C-F	0.502	4.48	

In [ ]:
```
hustle.head()
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	CONTESTED_SHOTS_2PT	CONTESTED_SHOTS_3PT
0	Aaron Gordon	DEN	2.46	1.32
1	Aaron Holiday	IND	1.29	1.80
2	Aaron Nesmith	BOS	1.83	1.04
3	Abdel Nader	PHX	1.92	1.25
4	Adam Mokoka	CHI	0.21	0.07

In [ ]:
```
shooting.head()
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	LT_5FT_FGM	LT_5FT_FG%	BT5_9FT_FGM	BT5_9FT
0	Aaron Gordon	DEN	2.2	0.631	0.4	
1	Aaron Holiday	IND	1.0	0.426	0.3	
2	Aaron Nesmith	BOS	0.6	0.617	0.1	
3	Abdel Nader	PHX	1.3	0.564	0.3	
4	Adam Mokoka	CHI	0.5	0.667	0.0	

In [ ]:
```
playtype.head()
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	ISO_PPP	ISO_FG%	ISO_POSS	PNR_HANDLER_PPP
0	Damian Lillard	POR	1.106	0.442	4.9	1.073
1	Julius Randle	NYK	0.901	0.420	5.5	0.886
2	Russell Westbrook	WAS	0.783	0.395	6.4	0.713
3	Luka Doncic	DAL	1.046	0.463	4.6	1.006
4	James Harden	BKN	1.091	0.435	8.0	0.915

5 rows × 32 columns

In [ ]:
```
tracking_passing.head()
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	PASSES_MADE	SECONDARY_AST	POTENTIAL_AST
0	Aaron Gordon	DEN	32.9	0.6	5.7
1	Aaron Holiday	IND	15.5	0.3	3.3
2	Aaron Nesmith	BOS	11.7	0.1	1.0
3	Abdel Nader	PHX	9.4	0.0	1.1
4	Adam Mokoka	CHI	2.4	0.1	0.6

In [ ]:
```
# finally we merge all dataframes into one big dashboard for every player
# followed by some renaming and tidy up work

players_dashboard = general_traditional.merge(general_advanced)
players_dashboard = players_dashboard.merge(defense_dashboard, how='left')
players_dashboard = players_dashboard.merge(hustle, how='left')
players_dashboard = players_dashboard.merge(shooting, how='left')
players_dashboard = players_dashboard.merge(playtype, how='left')
players_dashboard = players_dashboard.merge(tracking_passing)

players_dashboard.fillna(0, inplace=True)
print(players_dashboard.columns)
players_dashboard
```

```
Index(['PLAYER_NAME', 'TEAM_ABBREVIATION', 'FG_PCT', 'FG3_PCT', 'OREB', 'DRE
B',
       'AST', 'STL', 'BLK', 'PF', 'PTS', 'PLUS_MINUS', 'AST_PCT', 'AST_TO',
       'OREB_PCT', 'DREB_PCT', 'TM_TOV_PCT', 'TS_PCT', 'USG_PCT', 'PIE',
       'PLAYER_POSITION', 'LT6FT_FREQ', 'LT6FT_DFGM', 'LT6FT_%DIFF',
       '3PT_FREQ', '3PT_DFGM', '3PT_%DIFF', 'BT6_9FT_DFGM', 'BT6_9FT_%DIFF',
       'BT6_9FT_FREQ', 'BT10_15FT_DFGM', 'BT10_15FT_%DIFF', 'BT10_15FT_FREQ',
       'BT16FT_3PT_DFGM', 'BT16FT_3PT_%DIFF', 'BT16FT_3PT_FREQ',
       'CONTESTED_SHOTS_2PT', 'CONTESTED_SHOTS_3PT', 'DEFLECTIONS',
       'CHARGES_DRAWN', 'SCREEN_ASSISTS', 'OFF_LOOSE_BALLS_RECOVERED',
       'DEF_LOOSE_BALLS_RECOVERED', 'LT_5FT_FGM', 'LT_5FT_FG%', 'BT5_9FT_FGM',
       'BT5_9FT_FG%', 'BT10_14FT_FGM', 'BT10_14FT_FG%', 'BT15_19FT_FGM',
       'BT15_19FT_FG%', 'BT20_24FT_FGM', 'BT20_24FT_FG%', 'BT25_29FT_FGM',
       'BT25_29FT_FG%', 'ISO_PPP', 'ISO_FG%', 'ISO_POSS', 'PNR_HANDLER_PPP',
       'PNR_HANDLER_FG%', 'PNR_HANDLER_POSS', 'PNR_ROLLMAN_PPP',
       'PNR_ROLLMAN_FG%', 'PNR_ROLLMAN_POSS', 'TRANSITION_PPP',
       'TRANSITION_FG%', 'TRANSITION_POSS', 'POST_UP_PPP', 'POST_UP_FG%',
```

```
       'POST_UP_POSS', 'SPOT_UP_PPP', 'SPOT_UP_FG%', 'SPOT_UP_POSS',
       'HANDOFF_PPP', 'HANDOFF_FG%', 'HANDOFF_POSS', 'CUT_PPP', 'CUT_FG%',
       'CUT_POSS', 'OFF_SCREEN_PPP', 'OFF_SCREEN_FG%', 'OFF_SCREEN_POSS',
       'PUTBACK_PPP', 'PUTBACK_FG%', 'PUTBACK_POSS', 'PASSES_MADE',
       'SECONDARY_AST', 'POTENTIAL_AST', 'AST_TO_PASS_PCT'],
      dtype='object')
```

Out[ ]:

	PLAYER_NAME	TEAM_ABBREVIATION	FG_PCT	FG3_PCT	OREB	DREB	AST	STL	BLK
0	Aaron Gordon	DEN	0.463	0.335	1.5	4.1	3.2	0.7	0.7
1	Aaron Holiday	IND	0.390	0.368	0.2	1.1	1.9	0.7	0.2
2	Aaron Nesmith	BOS	0.438	0.370	0.6	2.2	0.5	0.3	0.2
3	Abdel Nader	PHX	0.491	0.419	0.3	2.3	0.8	0.4	0.4
4	Al Horford	BOS	0.450	0.368	1.0	5.7	3.4	0.9	0.9
...	...	...	...	...	...	...	...	...	...
391	Xavier Tillman	MEM	0.559	0.338	1.3	3.1	1.3	0.7	0.6
392	Yogi Ferrell	LAC	0.351	0.321	0.5	1.4	2.2	0.7	0.3
393	Yuta Watanabe	TOR	0.439	0.400	0.7	2.5	0.8	0.5	0.4
394	Zach LaVine	CHI	0.507	0.419	0.6	4.4	4.9	0.8	0.5
395	Zion Williamson	NOP	0.611	0.294	2.7	4.5	3.7	0.9	0.6

396 rows × 89 columns

In [ ]:

```
# final tidy up and eport to a pickle file and excel csv

players_dashboard.rename(
    {
        'TEAM_ABBREVIATION': 'TEAM',
        'FG_PCT': 'FG%',
        'FG3_PCT': 'FG3%',
        'FT_PCT': 'FT%',
        'AST_PCT': 'AST%',
        'OREB_PCT': 'OREB%',
        'DREB_PCT': 'DREB%',
        'TM_TOV_PCT': 'TM_TOV%',
        'TS_PCT': 'TS%',
        'USG_PCT': 'USG%',
        'AST_TO_PASS_PCT': 'AST_TO_PASS%'
    },
    axis=1, inplace=True
)

players_dashboard.to_csv('players_dashboard.csv')
players_dashboard.to_pickle('players_dashboard.pkl')
```

PCA

This jupyter notebook will:

- using PCA to further reduce the dimenionality
- investigating components and finding insights for clustering

In [ ]:
```
# importing libraries and modules needed for PCA
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
# loading the store pickle file and transform it to dataframe
players_dashboard = pd.read_pickle('players_dashboard.pkl')
players_dashboard.reset_index(inplace=True)
players_dashboard.drop(['index'], axis=1, inplace=True)
players_dashboard
```

Out[ ]:

	PLAYER_NAME	TEAM	FG%	FG3%	OREB	DREB	AST	STL	BLK	PF	...	OFF_SCREE
0	Aaron Gordon	DEN	0.463	0.335	1.5	4.1	3.2	0.7	0.7	1.8	...	
1	Aaron Holiday	IND	0.390	0.368	0.2	1.1	1.9	0.7	0.2	1.4	...	
2	Aaron Nesmith	BOS	0.438	0.370	0.6	2.2	0.5	0.3	0.2	1.9	...	
3	Abdel Nader	PHX	0.491	0.419	0.3	2.3	0.8	0.4	0.4	1.4	...	
4	Al Horford	BOS	0.450	0.368	1.0	5.7	3.4	0.9	0.9	1.7	...	
...	...	...	...	...	...	...	...	...	...	...	...	
391	Xavier Tillman	MEM	0.559	0.338	1.3	3.1	1.3	0.7	0.6	2.0	...	
392	Yogi Ferrell	LAC	0.351	0.321	0.5	1.4	2.2	0.7	0.3	1.1	...	
393	Yuta Watanabe	TOR	0.439	0.400	0.7	2.5	0.8	0.5	0.4	1.1	...	
394	Zach LaVine	CHI	0.507	0.419	0.6	4.4	4.9	0.8	0.5	2.4	...	
395	Zion Williamson	NOP	0.611	0.294	2.7	4.5	3.7	0.9	0.6	2.2	...	

396 rows × 89 columns

In [ ]:
```
# standardizing data
players_stats_std = StandardScaler().fit_transform(players_dashboard.drop(['P
players_stats_std = pd.DataFrame(players_stats_std, columns=players_dashboard
players_stats_std['PLAYER'] = players_dashboard['PLAYER_NAME']
players_stats_std['TEAM'] = players_dashboard['TEAM']
players_stats_std['POSITION'] = players_dashboard['PLAYER_POSITION']

players_stats_std
```

Out[ ]:

	FG%	FG3%	OREB	DREB	AST	STL	BLK	PI
0	-0.040703	0.077299	0.694923	0.403816	0.387215	-0.137444	0.487523	-0.17407

	FG%	FG3%	OREB	DREB	AST	STL	BLK	PI
1	-0.995761	0.375892	-1.002166	-1.351519	-0.294749	-0.137444	-0.689666	-0.85155(
2	-0.367777	0.393988	-0.479985	-0.707897	-1.029172	-1.307937	-0.689666	-0.00470!
3	0.325621	0.837354	-0.871620	-0.649385	-0.871796	-1.015314	-0.218791	-0.85155(
4	-0.210781	0.375892	0.042196	1.339995	0.492132	0.447802	0.958398	-0.34344:
...	...	...	...	...	...	...	...	..
391	1.215264	0.104444	0.433832	-0.181296	-0.609502	-0.137444	0.252085	0.16466-
392	-1.505997	-0.049377	-0.610530	-1.175986	-0.137373	-0.137444	-0.454229	-1.35965!
393	-0.354694	0.665437	-0.349439	-0.532363	-0.871796	-0.722691	-0.218791	-1.35965!
394	0.534949	0.837354	-0.479985	0.579349	1.279014	0.155179	0.016647	0.84214
395	1.895579	-0.293680	2.261466	0.637860	0.649508	0.447802	0.252085	0.50340:

396 rows × 89 columns

In [ ]:
```
# fitting the standardizing data into PCA
pca = PCA(random_state=0).fit(players_stats_std.drop(['PLAYER', 'TEAM', 'POSI'
pca_list = [ 'PC' + str(i) for i in range(1, pca.n_components_ + 1)]
pca_variance = pd.DataFrame({
    'Variance explained': pca.explained_variance_,
    'Percentage of variance explained': pca.explained_variance_ratio_
}, index=pca_list)
pca_variance['Cumulative Percentage of variance explained'] = pca_variance['P
pca_variance.head(10)
```

Out[ ]:

	Variance explained	Percentage of variance explained	Cumulative Percentage of variance explained
PC1	19.926538	0.231119	0.231119
PC2	17.214583	0.199664	0.430783
PC3	5.223597	0.060586	0.491369
PC4	3.293922	0.038205	0.529574
PC5	3.045287	0.035321	0.564895
PC6	2.494112	0.028928	0.593823
PC7	2.182078	0.025309	0.619132
PC8	1.833290	0.021263	0.640395
PC9	1.642621	0.019052	0.659447
PC10	1.590081	0.018443	0.677890

In [ ]:
```
pca_data = PCA(random_state=0).fit_transform(players_stats_std.drop(['PLAYER'
pca_data = pd.DataFrame(pca_data, columns=pca_list)
```

```
pca_data['PLAYER'] = players_stats_std['PLAYER']
pca_data['TEAM'] = players_stats_std['TEAM']
pca_data['POSITION'] = players_stats_std['POSITION']
pca_data.to_pickle('pca_data.pkl')
pca_data
```

Out[ ]:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC
0	0.882081	1.494421	-0.103091	-0.276908	-2.034680	0.646940	-0.847650	0.49941
1	-3.105932	-3.972688	-0.099769	-0.275749	0.593826	-0.210098	-0.191335	-2.03373
2	0.288487	-4.568325	-1.555126	-1.268215	0.319178	0.871216	-0.080447	0.53942
3	0.197796	-4.259287	-1.253244	-1.309263	-0.583111	-0.886630	0.408395	-0.68386
4	3.188852	0.733974	5.771234	-3.028903	7.864954	0.795206	2.981832	-0.73131
...	...	...	...	...	...	...	...	.
391	5.263395	0.616630	-0.827514	-1.665981	-0.742167	-0.320279	-1.455114	0.97668
392	-0.849503	-6.196595	6.894639	-2.518939	0.948903	-1.509507	2.918007	5.40245
393	0.115215	-4.537190	-1.805106	0.628896	-1.676585	0.339316	-0.824896	0.94012
394	-6.056749	5.995335	-1.423611	0.042179	-1.203111	0.666256	2.164512	-0.31239
395	-0.134730	9.304845	-1.553399	2.119779	-4.042119	-0.088732	0.720046	-1.60515

396 rows × 89 columns

In [ ]:

```
# Compositions of components
pca_eigen = pd.DataFrame(pca.components_, columns=players_stats_std.drop(['PL
pca_eigen
```

Out[ ]:

	FG%	FG3%	OREB	DREB	AST	STL	BLK	PF
0	0.132868	-0.126104	0.158507	0.054166	-0.132308	-0.092250	0.122940	0.039639
1	0.109902	-0.003523	0.129870	0.195849	0.136542	0.123497	0.115311	0.145285
2	-0.034241	-0.159098	0.050776	0.021449	0.206002	0.100197	-0.008224	-0.011967
3	-0.109978	-0.156954	0.072401	0.046838	-0.019250	0.203138	0.109284	0.191785
4	-0.190210	0.140249	-0.002219	0.085107	-0.028139	0.009997	0.067937	0.112995
...	...	...	...	...	...	...	...	...
81	-0.029857	0.005733	-0.199114	0.199486	0.009342	0.026698	0.003197	-0.032807
82	0.126968	0.005017	0.377099	-0.207857	-0.163478	0.011492	-0.014776	0.006319
83	0.145282	-0.002637	0.128763	-0.096161	0.692418	-0.018873	-0.003542	0.006105
84	0.209222	0.007967	-0.015400	-0.007162	-0.235629	0.027652	0.000907	-0.015654
85	0.003833	0.000181	0.003348	-0.001228	0.004589	0.000649	0.000766	0.000456

86 rows × 86 columns

In [ ]:
```
# PC1
print(pca_eigen.iloc[0][pca_eigen.iloc[0].abs().sort_values(ascending=False).
print('\n')
print(pca_data.nlargest(10,'PC1')[['PC1', 'PLAYER']])
```

```
OREB%                  0.190918
PNR_HANDLER_PPP       -0.172078
SCREEN_ASSISTS         0.170267
PNR_HANDLER_FG%       -0.167128
3PT_FREQ              -0.166189
PNR_HANDLER_POSS      -0.162927
AST_TO_PASS%          -0.161203
OREB                   0.158507
HANDOFF_FG%           -0.154675
HANDOFF_PPP           -0.153156
HANDOFF_POSS          -0.152183
CONTESTED_SHOTS_2PT    0.151777
BT25_29FT_FGM         -0.149828
PUTBACK_POSS           0.147825
PNR_ROLLMAN_POSS       0.147169
CUT_POSS               0.145109
BT20_24FT_FGM         -0.144609
DREB%                  0.143462
PNR_ROLLMAN_FG%        0.143120
SECONDARY_AST         -0.142942
Name: 0, dtype: float64


           PC1             PLAYER
335  12.873855        Rudy Gobert
54   11.996865       Clint Capela
121  11.349243       Enes Freedom
281  10.692030  Mitchell Robinson
82   10.270398     DeAndre Jordan
159  10.267981        Ivica Zubac
69    9.728899     Daniel Gafford
87    9.727703      Deandre Ayton
106   9.611996         Donta Hall
170   9.590553       Jakob Poeltl
```

In [ ]:
```
# PC2
print(pca_eigen.iloc[1][pca_eigen.iloc[1].abs().sort_values(ascending=False).
print('\n')
print(pca_data.nlargest(10,'PC2')[['PC2', 'PLAYER']])
```

```
PTS                          0.199246
PIE                          0.197299
DREB                         0.195849
LT_5FT_FGM                   0.194865
LT6FT_DFGM                   0.176635
OFF_LOOSE_BALLS_RECOVERED    0.172353
BT5_9FT_FGM                  0.169840
PASSES_MADE                  0.168043
USG%                         0.154592
POST_UP_POSS                 0.152815
BT16FT_3PT_DFGM              0.147127
PF                           0.145285
ISO_POSS                     0.144150
POST_UP_FG%                  0.143802
```

```
POST_UP_PPP                  0.143711
BT6_9FT_DFGM                 0.142565
BT10_15FT_DFGM               0.141681
3PT_DFGM                     0.138752
BT10_14FT_FGM                0.137622
PUTBACK_POSS                 0.137485
Name: 1, dtype: float64
```

	PC2	PLAYER
299	14.356474	Nikola Jokic
196	12.008666	Joel Embiid
141	11.304216	Giannis Antetokounmpo
104	11.239786	Domantas Sabonis
220	10.655708	Karl-Anthony Towns
300	10.439976	Nikola Vucevic
200	9.790050	Jonas Valanciunas
395	9.304845	Zion Williamson
337	9.287998	Russell Westbrook
213	9.249537	Julius Randle

In [ ]:
```
# PC3
print(pca_eigen.iloc[2][pca_eigen.iloc[2].abs().sort_values(ascending=False).
print('\n')
print(pca_data.nlargest(10,'PC3')[['PC3', 'PLAYER']])
```

```
AST%                  0.262885
TRANSITION_PPP       -0.224155
SPOT_UP_POSS         -0.220048
OFF_SCREEN_PPP       -0.208578
AST                   0.206002
POTENTIAL_AST         0.203186
OFF_SCREEN_FG%       -0.195065
SPOT_UP_PPP          -0.187985
BT20_24FT_FGM        -0.183993
CUT_PPP              -0.180213
CUT_FG%              -0.177778
AST_TO_PASS%          0.175983
OFF_SCREEN_POSS      -0.173029
SPOT_UP_FG%          -0.165801
PASSES_MADE           0.164734
AST_TO                0.164278
TS%                  -0.160068
FG3%                 -0.159098
TM_TOV%               0.154398
TRANSITION_FG%       -0.143226
Name: 2, dtype: float64
```

	PC3	PLAYER
269	7.781874	Matthew Dellavedova
360	7.173065	Thaddeus Young
337	6.904351	Russell Westbrook
392	6.894639	Yogi Ferrell
225	5.912303	Kemba Walker
4	5.771234	Al Horford
51	5.663023	Chris Paul
176	5.585118	James Harden
238	5.558224	Killian Hayes
199	5.328880	John Wall

In [ ]:
```
# PC4
print(pca_eigen.iloc[3][pca_eigen.iloc[3].abs().sort_values(ascending=False).
```

```
print('\n')
print(pca_data.nlargest(10,'PC4')[['PC4', 'PLAYER']])
```

```
3PT_DFGM                     0.235342
CONTESTED_SHOTS_3PT          0.234037
DEFLECTIONS                  0.229566
BT5_9FT_FG%                 -0.226953
BT10_15FT_FREQ              -0.203619
STL                          0.203138
BT10_14FT_FG%               -0.198230
PF                           0.191785
DEF_LOOSE_BALLS_RECOVERED    0.191544
BT6_9FT_FREQ                -0.181773
TS%                         -0.179039
PIE                         -0.177798
PLUS_MINUS                  -0.168380
BT25_29FT_FG%               -0.158535
FG3%                        -0.156954
BT10_14FT_FGM               -0.156444
BT20_24FT_FG%               -0.152994
LT6FT_DFGM                   0.143435
BT15_19FT_FGM               -0.139305
SPOT_UP_POSS                 0.130932
Name: 3, dtype: float64


          PC4             PLAYER
41   5.850233        Cam Reddish
130  4.910813  Freddie Gillespie
327  4.792714   Robert Covington
208  4.156879        Josh Okogie
154  4.028645        Isaac Okoro
255  4.018052      Luguentz Dort
75   4.003638      Darius Bazley
383  3.815782     Victor Oladipo
10   3.812376     Andre Drummond
167  3.749377      Jae'Sean Tate
```

In [ ]:

```
# PC5
print(pca_eigen.iloc[4][pca_eigen.iloc[4].abs().sort_values(ascending=False).
print('\n')
print(pca_data.nlargest(10,'PC5')[['PC5', 'PLAYER']])
```

```
BT10_15FT_DFGM          0.266591
BT16FT_3PT_DFGM         0.244504
TRANSITION_FG%         -0.238977
BT6_9FT_DFGM            0.237226
BT10_15FT_FREQ          0.201572
FG%                    -0.190210
BT20_24FT_FGM           0.187854
LT6FT_DFGM              0.183436
CONTESTED_SHOTS_2PT     0.169635
TRANSITION_PPP         -0.167486
BT16FT_3PT_FREQ         0.166731
BT25_29FT_FG%           0.163317
BT25_29FT_FGM           0.161226
CUT_FG%                -0.159986
CUT_PPP                -0.151867
3PT_DFGM                0.149510
LT_5FT_FG%             -0.146282
CONTESTED_SHOTS_3PT     0.145390
TM_TOV%                -0.144197
PUTBACK_FG%            -0.142976
```

```
Name: 4, dtype: float64
```

```
          PC5             PLAYER
4    7.864954         Al Horford
245  6.943303  LaMarcus Aldridge
35   5.466913       Brook Lopez
225  5.056500       Kemba Walker
300  4.559126     Nikola Vucevic
272  4.446643        Maxi Kleber
216  4.410811      Justin Patton
307  4.251908        P.J. Tucker
72   3.471365        Danny Green
97   3.329617      Derrick White
```

In [ ]:
```
# PC6
print(pca_eigen.iloc[5][pca_eigen.iloc[5].abs().sort_values(ascending=False).
print('\n')
print(pca_data.nlargest(10,'PC6')[['PC6', 'PLAYER']])
```

```
USG%                          0.249094
TRANSITION_FG%               -0.239501
BT6_9FT_FREQ                 -0.238163
PLUS_MINUS                   -0.228004
AST_TO                       -0.221013
TRANSITION_PPP               -0.217828
BT15_19FT_FGM                 0.211465
DEFLECTIONS                  -0.190139
POST_UP_PPP                   0.185737
TS%                          -0.183391
POST_UP_POSS                  0.176228
LT6FT_FREQ                    0.171487
POST_UP_FG%                   0.171385
BT10_15FT_FREQ               -0.169348
STL                          -0.168888
BT6_9FT_DFGM                 -0.161365
BT10_14FT_FGM                 0.154152
SPOT_UP_FG%                  -0.151380
OFF_SCREEN_POSS               0.149595
DEF_LOOSE_BALLS_RECOVERED    -0.144072
Name: 5, dtype: float64
```

```
          PC6              PLAYER
179  4.811652   Jaren Jackson Jr.
205  4.729569           Josh Hall
305  3.808730     Otto Porter Jr.
196  3.787251         Joel Embiid
160  3.748483           JJ Redick
7    3.739252  Aleksej Pokusevski
233  3.671968          Kevin Love
240  3.644402  Kristaps Porzingis
99   3.545302        Devin Booker
215  3.532858      Justin Jackson
```

In [ ]:
```
# PC7
print(pca_eigen.iloc[6][pca_eigen.iloc[6].abs().sort_values(ascending=False).
print('\n')
print(pca_data.nlargest(10,'PC7')[['PC7', 'PLAYER']])
```

```
SPOT_UP_FG%       -0.292292
SPOT_UP_PPP       -0.257323
OFF_SCREEN_POSS    0.252616
```

```
HANDOFF_POSS       0.245817
OFF_SCREEN_PPP     0.216160
OFF_SCREEN_FG%     0.209664
POST_UP_POSS      -0.202481
POST_UP_FG%       -0.198843
POST_UP_PPP       -0.197817
TS%                0.191781
FG3%              -0.181129
FG%                0.173070
LT_5FT_FG%         0.166794
TRANSITION_PPP    -0.163536
TRANSITION_FG%    -0.161116
BT25_29FT_FG%     -0.151558
HANDOFF_PPP        0.130928
BT6_9FT_FREQ       0.129270
BT25_29FT_FGM      0.128454
SPOT_UP_POSS      -0.126851
Name: 6, dtype: float64
```

	PC7	PLAYER
106	6.069927	Donta Hall
82	4.341050	DeAndre Jordan
281	4.317686	Mitchell Robinson
348	4.286849	Stephen Curry
102	4.180922	Dewayne Dedmon
286	3.739035	Moses Brown
328	3.540419	Robert Williams III
69	3.387298	Daniel Gafford
30	3.155734	Bradley Beal
360	2.985152	Thaddeus Young

Clustering

This jupyter notebook will:

- perform k means clustering
- perform Gaussian Mixture model

In [ ]:
```
# importing libraries and modules needed for clustering
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture
from mpl_toolkits.mplot3d import Axes3D
```

In [ ]:
```
# import pca_data
pca_data = pd.read_pickle('pca_data.pkl')
pca_data_clustering = pca_data[['PLAYER', 'TEAM', 'POSITION', 'PC1', 'PC2', '
x = pca_data_clustering.drop(['PLAYER', 'TEAM', 'POSITION'], axis=1)
x
```

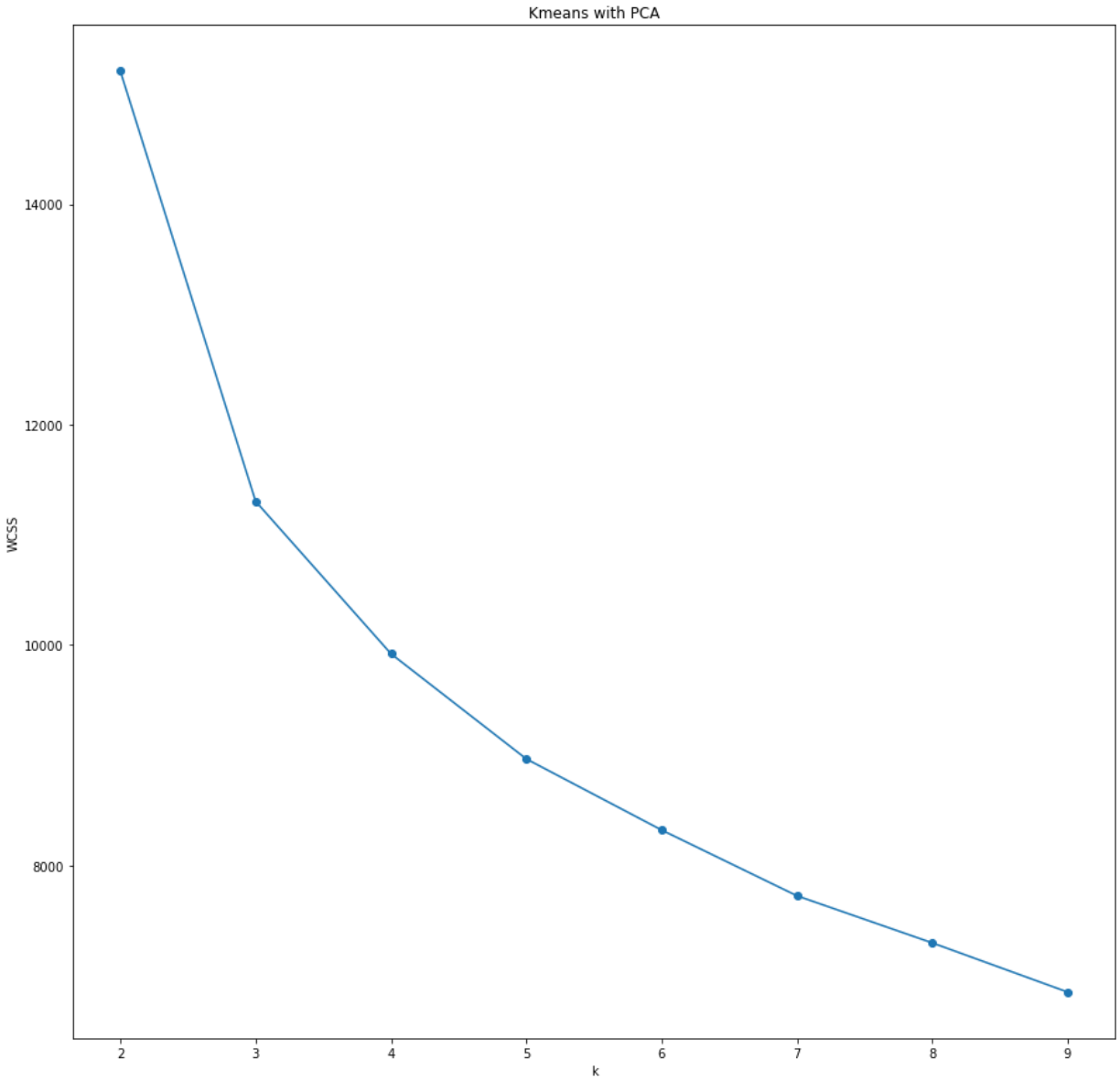
Out[ ]:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
0	0.882081	1.494421	-0.103091	-0.276908	-2.034680	0.646940	-0.847650
1	-3.105932	-3.972688	-0.099769	-0.275749	0.593826	-0.210098	-0.191335
2	0.288487	-4.568325	-1.555126	-1.268215	0.319178	0.871216	-0.080447
3	0.197796	-4.259287	-1.253244	-1.309263	-0.583111	-0.886630	0.408395
4	3.188852	0.733974	5.771234	-3.028903	7.864954	0.795206	2.981832
...	...	...	...	...	...	...	...
391	5.263395	0.616630	-0.827514	-1.665981	-0.742167	-0.320279	-1.455114
392	-0.849503	-6.196595	6.894639	-2.518939	0.948903	-1.509507	2.918007
393	0.115215	-4.537190	-1.805106	0.628896	-1.676585	0.339316	-0.824896
394	-6.056749	5.995335	-1.423611	0.042179	-1.203111	0.666256	2.164512
395	-0.134730	9.304845	-1.553399	2.119779	-4.042119	-0.088732	0.720046

396 rows × 7 columns

In [ ]:
```
# applying elbow method to find the best number of K
# testing the algorithm with up to 10 clusters
wcss = []
for i in range(2, 10):
    km = KMeans(n_clusters=i, init='k-means++')
    km.fit(x)
    wcss.append(km.inertia_)

plt.figure(figsize=(15,15))
plt.plot(range(2,10), wcss, marker='o')
plt.xlabel('k')
plt.ylabel('WCSS')
```

```
plt.title('Kmeans with PCA')
plt.show()
```



In [ ]:
```
# fitting pca_data into k-means
km = KMeans(n_clusters=3, init='k-means++')
km.fit(x)
```

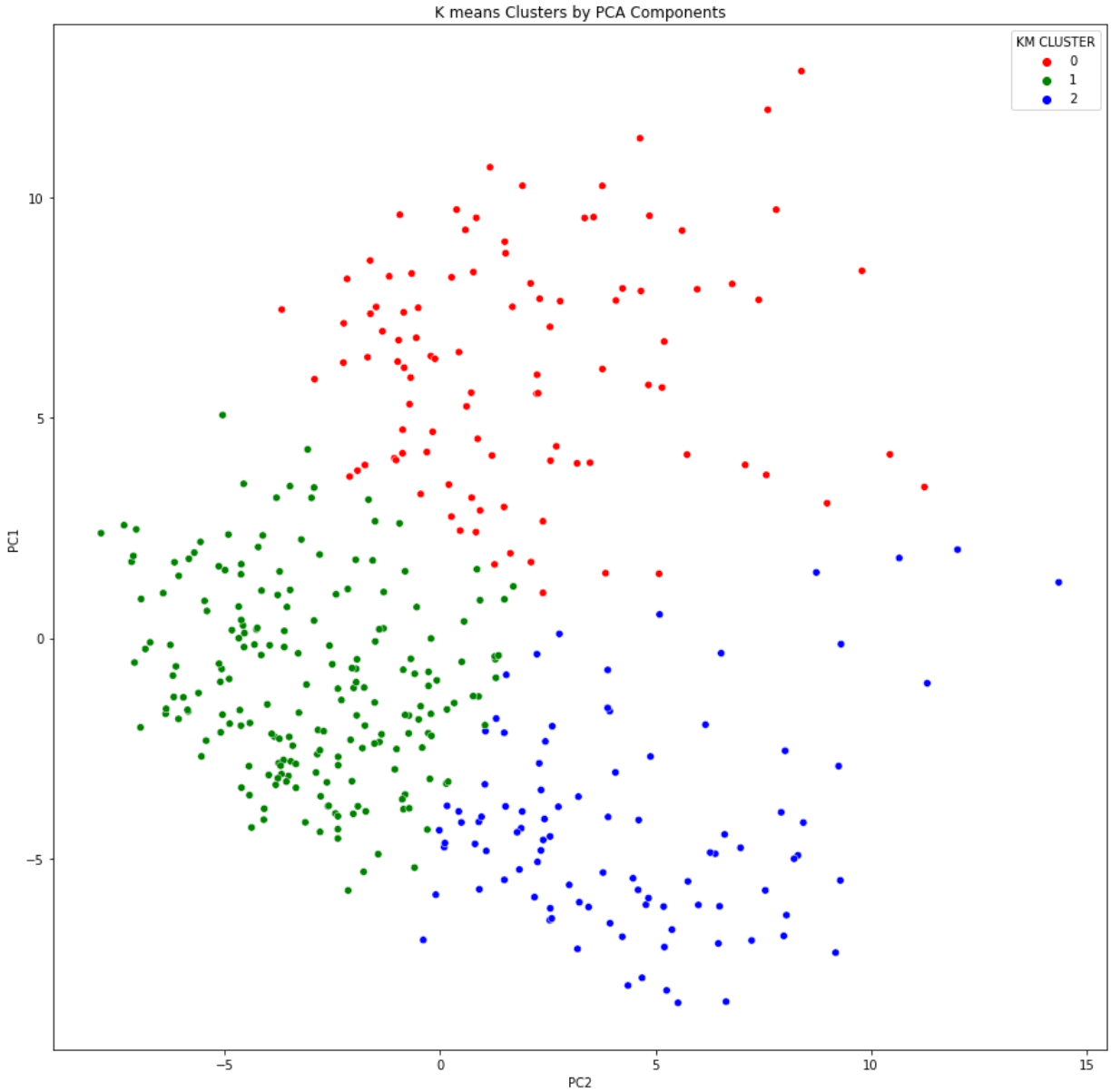
Out[ ]:
```
KMeans(n_clusters=3)
```

In [ ]:
```
model = pca_data_clustering.copy()
model['KM CLUSTER'] = km.labels_
```

In [ ]:
```
# plotting into 2D graph
plt.figure(figsize=(15,15))
sns.scatterplot(model['PC2'], model['PC1'], hue=model['KM CLUSTER'], palette=
plt.xlabel('PC2')
plt.ylabel('PC1')
plt.title('K means Clusters by PCA Components')
plt.show()
```

```
/opt/miniconda3/envs/tf_python/lib/python3.7/site-packages/seaborn/_decorator
s.py:43: FutureWarning: Pass the following variables as keyword args: x, y. Fr
```

```
om version 0.12, the only valid positional argument will be `data`, and passin
g other arguments without an explicit keyword will result in an error or misin
terpretation.
  FutureWarning
```

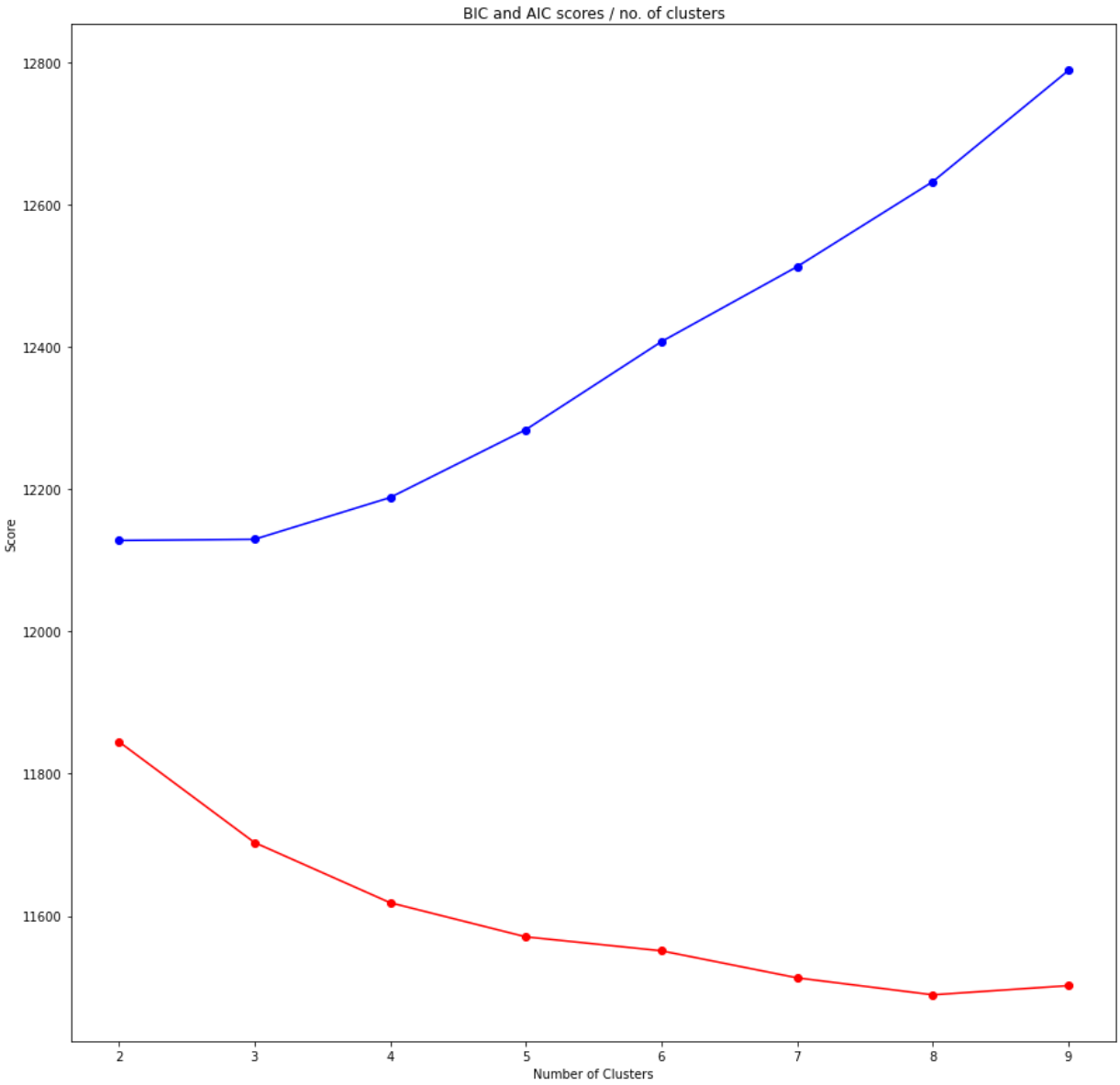


In [ ]:

```
# applying BIC and AIC scores to find the best number of clusters for GMM
n_range = range(2, 10)
bic_scores = []
aic_scores = []
for n in n_range:
    gm = GaussianMixture(n_components=n, n_init=10)
    gm.fit(x)
    bic_scores.append(gm.bic(x))
    aic_scores.append(gm.aic(x))

fig, ax = plt.subplots(figsize=(15,15), nrows=1)
ax.plot(n_range, bic_scores, '-o', color='blue')
ax.plot(n_range, aic_scores, '-o', color='red')
ax.set(xlabel='Number of Clusters', ylabel='Score')
ax.set_xticks(n_range)
ax.set_title('BIC and AIC scores / no. of clusters')
```

Out[ ]:

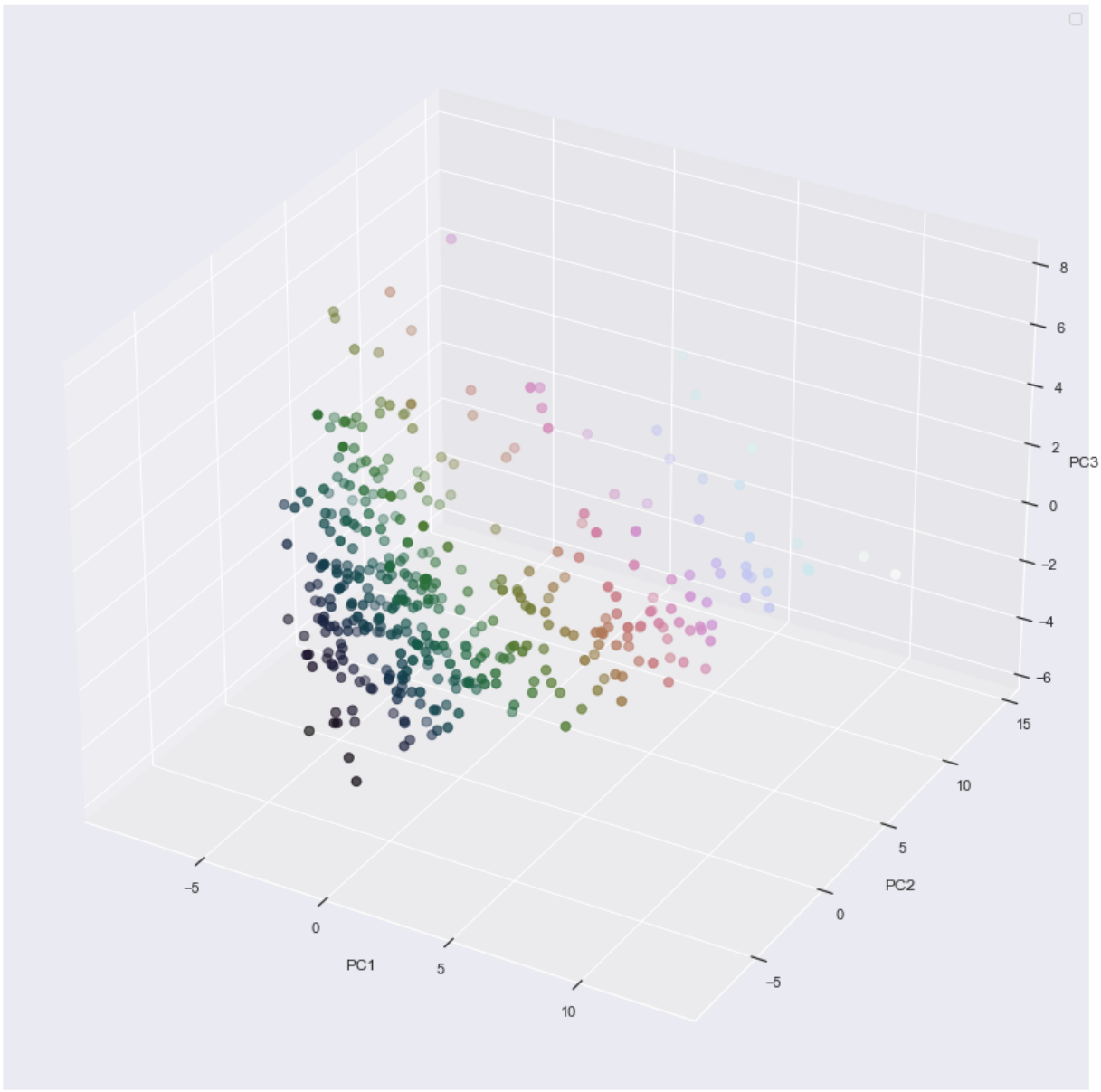
```
Text(0.5, 1.0, 'BIC and AIC scores / no. of clusters')
```



In [ ]:
```
# fitting pca_data into GMM
gm = GaussianMixture(n_components=8, n_init=10)
gm_labels = gm.fit_predict(x)
model['GMM CLUSTER'] = gm_labels
```

In [ ]:
```
# plotting into 3D graph
sns.set(style='darkgrid')
fig = plt.figure(figsize=(15,15))
ax = fig.add_subplot(111, projection='3d')
x = model['PC1']
y = model['PC2']
z = model['PC3']
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
threed_cmap = plt.get_cmap('cubehelix')
ax.scatter(x, y, z, c=(x + y + z), cmap=threed_cmap, s=50)
ax.legend()
plt.show()
```

```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



In [ ]:

```
# Clustering with PCA
model.to_pickle('model.pkl')
model
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
0	Aaron Gordon	DEN	F	0.882081	1.494421	-0.103091	-0.276908	-2.034680	(
1	Aaron Holiday	IND	G	-3.105932	-3.972688	-0.099769	-0.275749	0.593826	-
2	Aaron Nesmith	BOS	G-F	0.288487	-4.568325	-1.555126	-1.268215	0.319178	
3	Abdel Nader	PHX	F	0.197796	-4.259287	-1.253244	-1.309263	-0.583111	-(
4	Al Horford	BOS	C-F	3.188852	0.733974	5.771234	-3.028903	7.864954	
...	...	...	...	...	...	...	...	...	
391	Xavier Tillman	MEM	F	5.263395	0.616630	-0.827514	-1.665981	-0.742167	-

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
392	Yogi Ferrell	LAC	G	-0.849503	-6.196595	6.894639	-2.518939	0.948903	-
393	Yuta Watanabe	TOR	G-F	0.115215	-4.537190	-1.805106	0.628896	-1.676585	
394	Zach LaVine	CHI	G-F	-6.056749	5.995335	-1.423611	0.042179	-1.203111	(
395	Zion Williamson	NOP	F	-0.134730	9.304845	-1.553399	2.119779	-4.042119	-

396 rows × 12 columns

Analysis

This jupyter notebook will:

- examine clusters and analysis for the two unsupervised learning method

In [ ]:
```
# importing libraries and modules needed for clustering
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
# loading model
model =  pd.read_pickle('model.pkl')
model
```

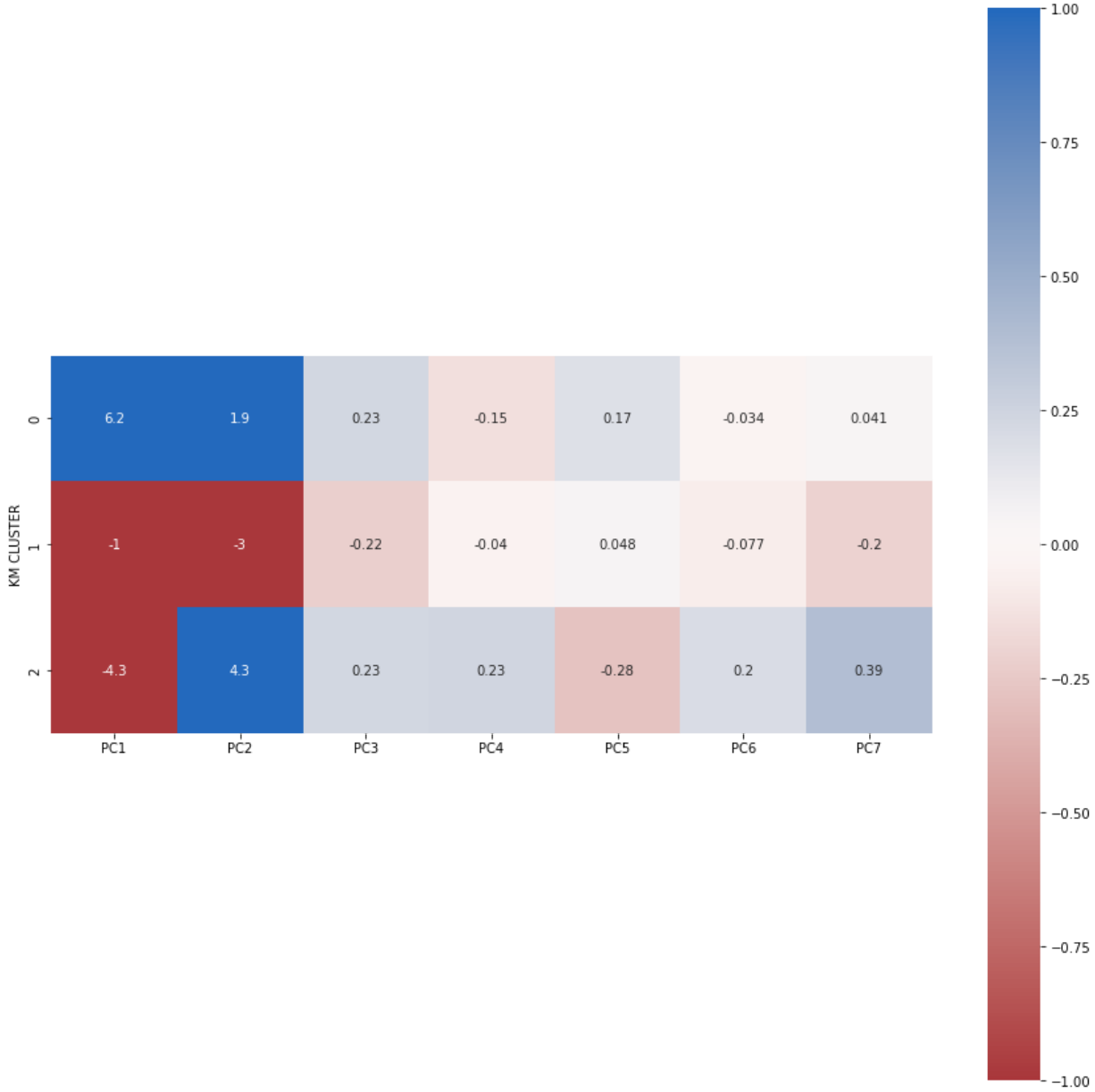
Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
0	Aaron Gordon	DEN	F	0.882081	1.494421	-0.103091	-0.276908	-2.034680	(
1	Aaron Holiday	IND	G	-3.105932	-3.972688	-0.099769	-0.275749	0.593826	-
2	Aaron Nesmith	BOS	G-F	0.288487	-4.568325	-1.555126	-1.268215	0.319178	
3	Abdel Nader	PHX	F	0.197796	-4.259287	-1.253244	-1.309263	-0.583111	-(
4	Al Horford	BOS	C-F	3.188852	0.733974	5.771234	-3.028903	7.864954	
...	...	...	...	...	...	...	...	...	
391	Xavier Tillman	MEM	F	5.263395	0.616630	-0.827514	-1.665981	-0.742167	-
392	Yogi Ferrell	LAC	G	-0.849503	-6.196595	6.894639	-2.518939	0.948903	-
393	Yuta Watanabe	TOR	G-F	0.115215	-4.537190	-1.805106	0.628896	-1.676585	
394	Zach LaVine	CHI	G-F	-6.056749	5.995335	-1.423611	0.042179	-1.203111	(
395	Zion Williamson	NOP	F	-0.134730	9.304845	-1.553399	2.119779	-4.042119	-

396 rows × 12 columns

In [ ]:
```
# Interpreting K mean clusters
k_clusters = model.drop(['GMM CLUSTER'], axis=1)
k_clusters_pc_mean = k_clusters.groupby('KM CLUSTER').mean()
plt.figure(figsize=(15,15))
sns.heatmap(k_clusters_pc_mean, vmin=-1, vmax=1, center=0, cmap='vlag_r', squ
```

Out[ ]:
```
<AxesSubplot:ylabel='KM CLUSTER'>
```



In [ ]:
```
k_clusters[k_clusters['KM CLUSTER'] == 0].sort_values(by=['PC1', 'PC2', 'PC3'
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5
335	Rudy Gobert	UTA	C	12.873855	8.385346	0.234198	1.080648	2.680176
54	Clint Capela	ATL	C	11.996865	7.604640	0.940569	1.523086	0.159426
121	Enes Freedom	POR	C	11.349243	4.640469	1.483398	-0.739032	0.853819
281	Mitchell Robinson	NYK	C-F	10.692030	1.160703	2.630472	3.454518	0.374932
82	DeAndre Jordan	BKN	C	10.270398	1.913565	1.445750	0.764318	-1.600349
159	Ivica Zubac	LAC	C	10.267981	3.765455	0.356786	-2.007991	0.986681
69	Daniel Gafford	WAS	F-C	9.728899	0.384740	0.514264	1.071766	-3.506124
87	Deandre Ayton	PHX	C	9.727703	7.799906	-0.083409	-0.574043	1.454436

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5
106	Donta Hall	ORL	C	9.611996	-0.932226	2.496713	-2.121195	-1.042581
170	Jakob Poeltl	SAS	C	9.590553	4.861557	0.284585	0.672420	1.120626
350	Steven Adams	NOP	C	9.558986	3.563386	2.602140	2.072194	-0.654317
367	Tony Bradley	OKC	C-F	9.542795	0.841716	0.812417	-1.039636	-4.041709
328	Robert Williams III	BOS	C-F	9.539935	3.356070	1.718561	-0.265790	-3.042063
94	Derrick Favors	UTA	F	9.268330	0.588730	0.682220	-0.730310	-2.035432
181	Jarrett Allen	CLE	C	9.254632	5.617097	0.509729	-0.050557	0.315371
151	Hassan Whiteside	SAC	C	9.001005	1.502829	1.149105	-2.428216	-0.574292
114	Dwight Howard	PHI	C-F	8.739024	1.521361	0.279331	-2.272299	-1.236697
303	Onyeka Okongwu	ATL	F-C	8.574074	-1.621130	0.333279	0.037167	-4.343379
200	Jonas Valanciunas	MEM	C	8.341097	9.790050	-0.393832	-3.099371	3.047453
293	Nerlens Noel	NYK	C-F	8.312320	0.774900	0.613290	3.374784	-2.608550

In [ ]:
```
k_clusters[k_clusters['KM CLUSTER'] == 1].sort_values(by=['PC5'], ascending=F
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5
225	Kemba Walker	OKC	G	-1.544602	-0.448122	5.912303	0.411749	5.056500
272	Maxi Kleber	DAL	F	2.654486	-1.505496	-1.073625	-0.403339	4.446643
307	P.J. Tucker	MIL	F	1.633215	-5.131241	2.603162	2.641958	4.251908
72	Danny Green	PHI	G	-2.354823	-1.404126	-1.301161	2.407567	3.471365
78	Davis Bertans	WAS	F	-0.591921	-2.498479	-4.257572	0.420658	2.970933
146	Gorgui Dieng	SAS	C	3.419983	-2.920254	1.975849	-2.613379	2.823035
269	Matthew Dellavedova	CLE	G	-1.717622	-6.364928	7.781874	0.642838	2.640250
46	Cedi Osman	CLE	F	0.183075	-4.827425	5.014799	0.889352	2.639766
214	Justin Holiday	IND	F-G	-2.181634	-1.356089	-2.757691	2.191131	2.577116
327	Robert Covington	POR	F	-0.392738	1.359101	-0.763611	4.792714	2.508700
368	Tony Snell	ATL	G	-1.632580	-4.644225	-2.593250	-2.774624	2.404449

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
323	Reggie Bullock	NYK	G-F	-2.495285	-1.800047	-3.302069	-0.660040	2.338655	-
17	Armoni Brooks	HOU	G	-3.128518	-3.515029	-1.877806	0.134642	2.316948	
371	Trevor Ariza	MIA	F	-0.678389	-2.042265	-1.116794	3.556728	2.194702	-
138	Gary Trent Jr.	TOR	G-F	-5.303256	-1.770635	-2.211473	0.011551	2.131999	
136	Gary Clark	PHI	F	2.383728	-7.864474	3.416644	1.638047	2.113470	
135	Garrison Mathews	WAS	G	-1.334041	-6.175586	-2.293793	-1.381614	1.931453	-
154	Isaac Okoro	CLE	F-G	-0.957411	-0.074062	-0.583107	4.028645	1.876749	-
206	Josh Hart	NOP	G	-1.470819	0.330123	-0.186516	2.636390	1.742409	-
112	Duncan Robinson	MIA	F	-2.484732	-0.413742	-4.736633	0.576462	1.705962	

In [ ]:
```
k_clusters[k_clusters['KM CLUSTER'] == 2].sort_values(by=['PC2', 'PC7', 'PC3'
```

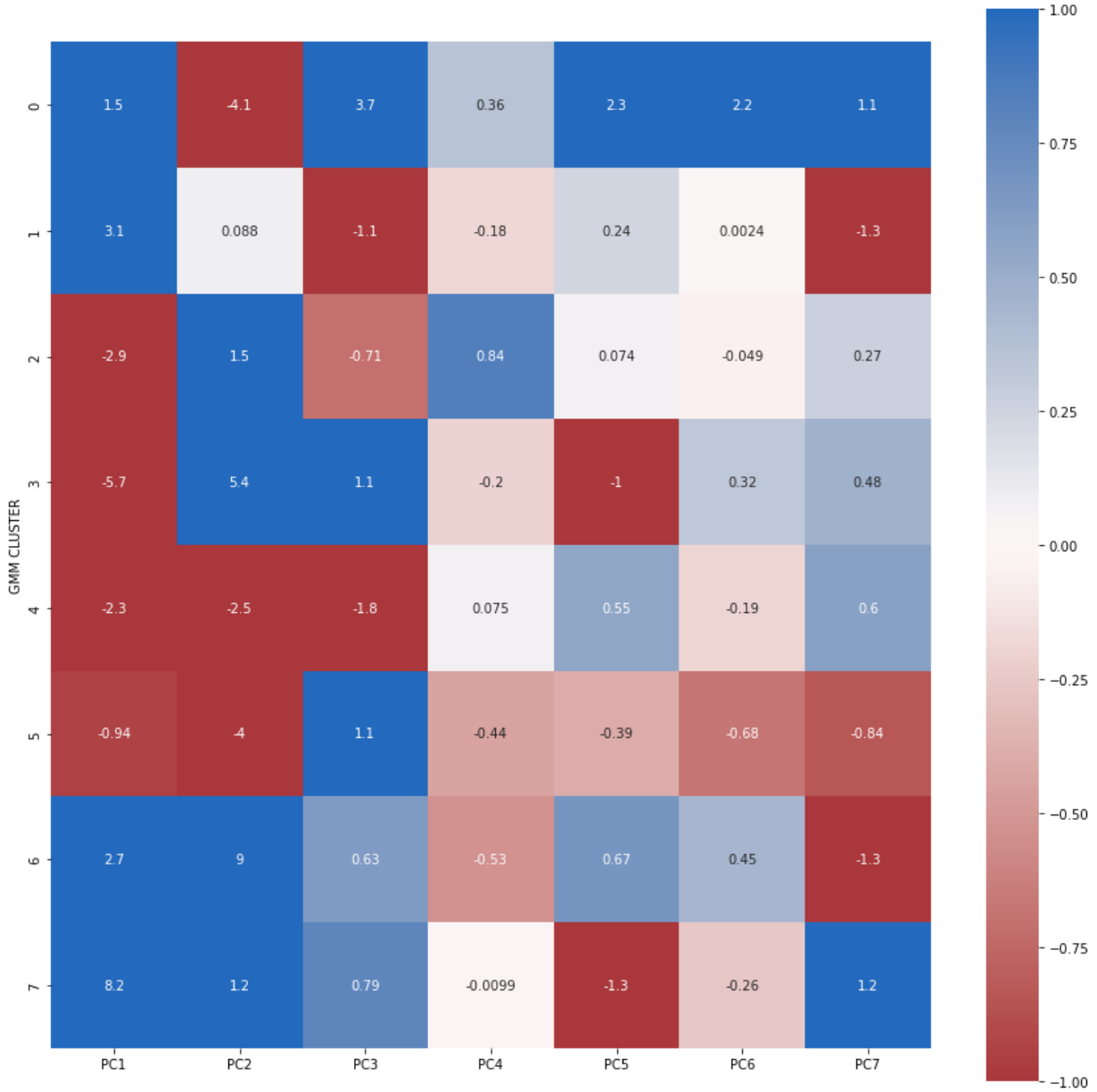
Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC
299	Nikola Jokic	DEN	C	1.267944	14.356474	2.619054	-4.372658	1.82298
196	Joel Embiid	PHI	C-F	2.011224	12.008666	0.127056	-3.041007	0.52344
141	Giannis Antetokounmpo	MIL	F	-1.022489	11.304216	0.544927	0.842661	-2.99262
220	Karl-Anthony Towns	MIN	C-F	1.819898	10.655708	-0.929568	-0.503866	2.66536
395	Zion Williamson	NOP	F	-0.134730	9.304845	-1.553399	2.119779	-4.04211
337	Russell Westbrook	WAS	G	-5.498704	9.287998	6.904351	0.370163	-1.94955
213	Julius Randle	NYK	F-C	-2.904486	9.249537	0.410074	-1.316475	0.73739
256	Luka Doncic	DAL	F-G	-7.139944	9.178708	3.513625	-1.587461	-2.23490
14	Anthony Davis	LAL	F-C	1.493251	8.730289	0.070448	-0.162658	-0.88999
251	LeBron James	LAL	F	-4.186861	8.430257	1.561553	-1.782775	-4.02507
221	Kawhi Leonard	LAC	F	-4.930372	8.305679	-0.239014	-2.449812	-2.75546
231	Kevin Durant	BKN	F	-5.007492	8.215086	-1.306907	-4.385556	-0.65345
30	Bradley Beal	WAS	G	-6.287549	8.038585	-0.734552	-0.421143	-1.02782
309	Pascal Siakam	TOR	F	-2.556227	8.007599	0.651933	2.381070	0.12476
244	Kyrie Irving	BKN	G	-6.759551	7.976042	1.193528	-1.454828	-2.03125
193	Jimmy Butler	MIA	F	-3.952633	7.916208	2.662290	0.374920	-3.26589
315	Paul George	LAC	F	-5.725444	7.547462	-0.572104	-0.748874	0.00291
176	James Harden	BKN	G	-6.862489	7.230448	5.585118	-2.096802	-0.98784
187	Jayson Tatum	BOS	F-G	-4.759261	6.973759	-1.094956	-0.513909	-2.46561

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC
348	Stephen Curry	GSW	G	-8.252439	6.636225	-1.639729	-1.370838	-2.10628

In [ ]:
```
# Interpreting GMM clusters
gmm_clusters = model.drop(['KM CLUSTER'], axis=1)
gmm_clusters_pc_mean = gmm_clusters.groupby('GMM CLUSTER').mean()
plt.figure(figsize=(15,15))
sns.heatmap(gmm_clusters_pc_mean, vmin=-1, vmax=1, center=0, cmap='vlag_r', s
```

Out[ ]:
```
<AxesSubplot:ylabel='GMM CLUSTER'>
```



In [ ]:
```
gmm_clusters[gmm_clusters['GMM CLUSTER'] == 0].sort_values(by=['PC3', 'PC5',
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
269	Matthew Dellavedova	CLE	G	-1.717622	-6.364928	7.781874	0.642838	2.640250	0
360	Thaddeus Young	CHI	F	2.759759	0.264625	7.173065	0.383883	1.946996	0

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
392	Yogi Ferrell	LAC	G	-0.849503	-6.196595	6.894639	-2.518939	0.948903	-1
225	Kemba Walker	OKC	G	-1.544602	-0.448122	5.912303	0.411749	5.056500	(
4	Al Horford	BOS	C-F	3.188852	0.733974	5.771234	-3.028903	7.864954	0
118	Ed Davis	MIN	C-F	7.461362	-3.671116	5.142611	1.378149	1.358409	1
46	Cedi Osman	CLE	F	0.183075	-4.827425	5.014799	0.889352	2.639766	2
5	Al-Farouq Aminu	CHI	F	1.942920	-5.700718	4.822251	2.145999	1.120658	3
286	Moses Brown	BOS	C	8.155696	-2.155121	4.751715	3.260757	3.098450	3
305	Otto Porter Jr.	ORL	F	0.716493	-4.671051	3.580875	-0.480701	0.926217	3

In [ ]:
```
gmm_clusters[gmm_clusters['GMM CLUSTER'] == 1].sort_values(by=['PC1', 'PC5',
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
114	Dwight Howard	PHI	C-F	8.739024	1.521361	0.279331	-2.272299	-1.236697	-0.0
362	Thomas Bryant	WAS	C-F	7.666506	4.081237	-2.492309	-1.866177	1.275553	-3.2
282	Mo Bamba	ORL	C	6.494846	0.439230	-0.190260	-1.818791	0.432113	2.1
235	Kevon Looney	GSW	F	6.404801	-0.213135	0.834283	-0.918246	0.625715	-3.1
18	Aron Baynes	TOR	C-F	6.278761	-0.980246	0.447687	-0.620040	1.659780	0.
342	Serge Ibaka	LAC	F	6.110759	3.768475	-1.350698	-3.098031	2.819633	0.5
178	James Wiseman	GSW	C	5.981029	2.251652	-1.889457	0.269619	-0.103574	2.7
288	Myles Turner	IND	C-F	5.749734	4.836246	-2.277020	3.026985	2.867196	0.0
35	Brook Lopez	MIL	C	5.690583	5.151027	-3.072704	-0.934646	5.466913	-1.2
236	Khem Birch	TOR	C	5.573408	0.723897	-0.216497	1.388958	-0.453773	-0.6

In [ ]:
```
gmm_clusters[gmm_clusters['GMM CLUSTER'] == 2].sort_values(by=['PC2', 'PC4',
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
12	Andrew Wiggins	GSW	F	-2.687518	4.883545	-2.009962	1.559640	0.199792	1.0

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
39	CJ McCollum	POR	G	-7.884882	4.360873	-0.404131	-1.533708	1.717335	-0.2
190	Jerami Grant	DET	F	-3.047262	4.070724	-2.762546	0.121638	-0.209722	3.4
274	Michael Porter Jr.	DEN	F	-1.656981	3.938716	-5.309825	0.357939	-0.987476	0.9
150	Harrison Barnes	SAC	F	-0.721526	3.891876	-1.631948	-0.546591	-0.018055	-0.6
302	OG Anunoby	TOR	F	-1.583089	3.890812	-3.482111	2.848698	0.557479	-1.3
91	Dennis Schroder	LAL	G	-5.601030	2.998187	2.914423	1.064203	0.484797	-0.8
280	Miles Bridges	CHA	F	0.097323	2.769323	-3.472732	1.543079	-0.603446	-0.2
275	Mikal Bridges	PHX	F	-1.998919	2.604579	-3.767077	0.731654	-0.475014	-2.4
76	Darius Garland	CLE	G	-6.362431	2.592409	1.850134	0.555979	0.549082	-1.1

In [ ]:
```
gmm_clusters[gmm_clusters['GMM CLUSTER'] == 3].sort_values(by=['PC2', 'PC3',
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
337	Russell Westbrook	WAS	G	-5.498704	9.287998	6.904351	0.370163	-1.949559	0.
256	Luka Doncic	DAL	F-G	-7.139944	9.178708	3.513625	-1.587461	-2.234909	2.
251	LeBron James	LAL	F	-4.186861	8.430257	1.561553	-1.782775	-4.025076	0
221	Kawhi Leonard	LAC	F	-4.930372	8.305679	-0.239014	-2.449812	-2.755462	1
231	Kevin Durant	BKN	F	-5.007492	8.215086	-1.306907	-4.385556	-0.653450	1.
30	Bradley Beal	WAS	G	-6.287549	8.038585	-0.734552	-0.421143	-1.027824	2.
244	Kyrie Irving	BKN	G	-6.759551	7.976042	1.193528	-1.454828	-2.031253	1
193	Jimmy Butler	MIA	F	-3.952633	7.916208	2.662290	0.374920	-3.265899	-2.
315	Paul George	LAC	F	-5.725444	7.547462	-0.572104	-0.748874	0.002918	0.
176	James Harden	BKN	G	-6.862489	7.230448	5.585118	-2.096802	-0.987843	-1

In [ ]:
```
gmm_clusters[gmm_clusters['GMM CLUSTER'] == 4].sort_values(by=['PC7', 'PC5',
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
109	Doug McDermott	IND	F	-0.808237	-0.585936	-4.967559	-0.356529	-2.644026	
112	Duncan Robinson	MIA	F	-2.484732	-0.413742	-4.736633	0.576462	1.705962	-
123	Eric Gordon	HOU	G	-4.902916	-1.434279	-0.483317	-0.643356	-0.005010	
271	Max Strus	MIA	G-F	-1.243497	-5.602876	-3.691229	-0.831104	-2.332817	-
287	Mychal Mulder	GSW	G	-0.640181	-6.126593	-4.112638	-1.486331	-0.806139	-
384	Wayne Ellington	DET	G	-2.895157	-3.697502	-3.399859	-0.991707	0.735094	-
17	Armoni Brooks	HOU	G	-3.128518	-3.515029	-1.877806	0.134642	2.316948	
363	Tim Hardaway Jr.	DAL	G-F	-4.183128	0.495778	-4.413643	-0.082643	0.910476	(
116	Dylan Windler	CLE	G-F	-0.200277	-3.613943	-1.092720	0.485780	-1.410310	-
204	Jordan Poole	GSW	G	-3.978574	-2.425509	-2.371958	-1.712350	-2.212846	-

In [ ]:
```
gmm_clusters[gmm_clusters['GMM CLUSTER'] == 5].sort_values(by=['PC3'], ascend
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
321	Rajon Rondo	LAC	G	-3.255169	-3.564492	3.942869	-1.794456	-0.549538	-
277	Mike James	BKN	G	-3.871474	-4.078353	3.900296	-1.008869	-1.694004	
92	Dennis Smith Jr.	DET	G	-2.084257	-2.830573	3.891196	-1.404663	0.326265	-
158	Ish Smith	WAS	G	-1.984379	-1.741550	3.390870	-2.216083	1.397866	-
382	Tyus Jones	MEM	G	-4.120395	-4.088154	3.329030	-2.569202	-1.371765	-
89	Delon Wright	SAC	G	-3.195484	-0.235515	3.227535	0.107181	0.226727	-2
29	Brad Wanamaker	CHA	G	-1.736282	-5.046183	3.195546	0.799393	-0.646670	(
254	Lou Williams	ATL	G	-4.180300	-3.125484	3.136087	-2.154878	-1.044837	(
258	Malachi Flynn	TOR	G	-2.439843	-3.416596	3.048996	-0.766920	0.033281	-(
296	Nico Mannion	GSW	G	-1.603090	-6.353302	2.984094	-0.753914	-0.849449	

In [ ]:
```
gmm_clusters[gmm_clusters['GMM CLUSTER'] == 6].sort_values(by=['PC2', 'PC1',
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC
299	Nikola Jokic	DEN	C	1.267944	14.356474	2.619054	-4.372658	1.82298
196	Joel Embiid	PHI	C-F	2.011224	12.008666	0.127056	-3.041007	0.52344
141	Giannis Antetokounmpo	MIL	F	-1.022489	11.304216	0.544927	0.842661	-2.99262
104	Domantas Sabonis	IND	F-C	3.432684	11.239786	2.926418	0.765334	0.80183
220	Karl-Anthony Towns	MIN	C-F	1.819898	10.655708	-0.929568	-0.503866	2.66536
300	Nikola Vucevic	CHI	C	4.172533	10.439976	0.559052	-3.757390	4.55912
200	Jonas Valanciunas	MEM	C	8.341097	9.790050	-0.393832	-3.099371	3.04745
395	Zion Williamson	NOP	F	-0.134730	9.304845	-1.553399	2.119779	-4.04211
213	Julius Randle	NYK	F-C	-2.904486	9.249537	0.410074	-1.316475	0.73739
21	Bam Adebayo	MIA	C-F	3.064523	8.981061	2.514558	-0.697403	-1.43937

In [ ]:

```
gmm_clusters[gmm_clusters['GMM CLUSTER'] == 7].sort_values(by=['PC1', 'PC2',
```

Out[ ]:

	PLAYER	TEAM	POSITION	PC1	PC2	PC3	PC4	PC5	
335	Rudy Gobert	UTA	C	12.873855	8.385346	0.234198	1.080648	2.680176	-3.3
54	Clint Capela	ATL	C	11.996865	7.604640	0.940569	1.523086	0.159426	-1.0
121	Enes Freedom	POR	C	11.349243	4.640469	1.483398	-0.739032	0.853819	0.2
281	Mitchell Robinson	NYK	C-F	10.692030	1.160703	2.630472	3.454518	0.374932	0.3
82	DeAndre Jordan	BKN	C	10.270398	1.913565	1.445750	0.764318	-1.600349	-2.4
159	Ivica Zubac	LAC	C	10.267981	3.765455	0.356786	-2.007991	0.986681	-1.9
69	Daniel Gafford	WAS	F-C	9.728899	0.384740	0.514264	1.071766	-3.506124	-0.9
106	Donta Hall	ORL	C	9.611996	-0.932226	2.496713	-2.121195	-1.042581	-0.3
170	Jakob Poeltl	SAS	C	9.590553	4.861557	0.284585	0.672420	1.120626	-1.7
350	Steven Adams	NOP	C	9.558986	3.563386	2.602140	2.072194	-0.654317	-1.2