In [32]:

```
import pybaseball as pyb
from pybaseball import statcast, pitching_stats, playerid_lookup, stat
import numpy as np
import math
import pandas as pd
import glob
import os
import re
import unicodedata
from datetime import datetime
from itertools import groupby
from operator import itemgetter
from sklearn.preprocessing import OneHotEncoder
```

First, load in DF from data_cleaning_notebook_1 'cleaning_filtered_df.csv'

Drop all info prior to 2008.

Then rename 'Year' to 'season'.

In [23]:

```
cleaning_filtered_df = pd.read_csv('cleaning_filtered_df.csv', index_c
cleaning_filtered_df
```

Out[23]:

	Name	Age	Year	Throws	IP	G	GS	CG	SHO	sDR	Career Start	Career End	Inact Ye
0	ed acosta	28	1972	1	89.0	46	2	0	0	1	1972	1972	
1	doyle alexander	21	1972	1	106.1	35	9	2	2	3	1972	1989	
2	lloyd allen	22	1972	1	85.1	42	6	0	0	5	1972	1975	
3	steve arlin	26	1972	1	250.0	38	37	12	3	22	1972	1974	
4	stan bahnsen	27	1972	1	252.1	43	41	5	1	32	1972	1981	[19 19
...	...	...	...	...	...	...	...	...	...	...	...	...	
16566	brandon woodruff	30	2023	1	67.0	11	11	1	1	0	2017	2023	
16567	kyle wright	27	2023	1	31.0	9	7	0	0	0	2019	2023	
16568	ryan yarbrough	31	2023	0	89.2	25	9	0	0	0	2018	2023	
16571	rob zastryzny	31	2023	0	20.2	21	1	0	0	0	2016	2023	[20 20 20 20 20 20
16572	angel zerpa	23	2023	0	42.2	15	3	0	0	1	2021	2023	

15060 rows × 13 columns

In [24]:
```
cleaning_filtered_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 15060 entries, 0 to 16572
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Name            15060 non-null  object 
 1   Age             15060 non-null  int64  
 2   Year            15060 non-null  int64  
 3   Throws          15060 non-null  int64  
 4   IP              15060 non-null  float64
 5   G               15060 non-null  int64  
 6   GS              15060 non-null  int64  
 7   CG              15060 non-null  int64  
 8   SHO             15060 non-null  int64  
 9   sDR             15060 non-null  int64  
 10  Career Start    15060 non-null  int64  
 11  Career End      15060 non-null  int64  
 12  Inactive Years  15060 non-null  object 
dtypes: float64(1), int64(10), object(2)
memory usage: 1.6+ MB
```

In [25]:
```
cleaning_filtered_df = cleaning_filtered_df[cleaning_filtered_df['Year
```

In [26]:
```
cleaning_filtered_df
```

Out[26]:

	Name	Age	Year	Throws	IP	G	GS	CG	SHO	sDR	Career Start	Career End	Inactiv Yea
10745	alfredo aceves	25	2008	1	30.0	6	4	0	0	0	2008	2013	[201 201
10746	nick adenhart	21	2008	1	12.0	3	3	0	0	0	2008	2009	
10747	matt albers	25	2008	1	49.0	28	3	0	0	2	2006	2016	[200 201 201 201 201 201 201
10748	alberto arias	24	2008	1	21.2	15	2	0	0	0	2008	2008	
10749	alberto arias	24	2008	1	8.0	3	2	0	0	0	2008	2008	
...	...	...	...	...	...	...	...	...	...	...	...	...	
16566	brandon woodruff	30	2023	1	67.0	11	11	1	1	0	2017	2023	
16567	kyle wright	27	2023	1	31.0	9	7	0	0	0	2019	2023	
16568	ryan yarbrough	31	2023	0	89.2	25	9	0	0	0	2018	2023	
16571	rob zastryzny	31	2023	0	20.2	21	1	0	0	0	2016	2023	[201 201 201 202 202 202
16572	angel zerpa	23	2023	0	42.2	15	3	0	0	1	2021	2023	

5305 rows × 13 columns

In [27]:
```
cleaning_filtered_df = cleaning_filtered_df.rename(columns={'Year': 's
```

Will merge cleaning_filtered_df with yearly DF later.

Now, load in data by year.

Start with 2008.

In [9]:
```
all_2008_stats_df = pd.read_csv('all_2008_stats_df.csv', index_col=0)
```

In [10]:

```
all_2008_stats_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 459185 entries, 0 to 459184
Data columns (total 92 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   pitch_type                      437522 non-null  object
 1   game_date                       459185 non-null  object
 2   release_speed                   437503 non-null  float64
 3   release_pos_x                   437492 non-null  float64
 4   release_pos_z                   437492 non-null  float64
 5   player_name                     459185 non-null  object
 6   batter                          459185 non-null  int64
 7   pitcher                         459185 non-null  int64
 8   events                          122796 non-null  object
 9   description                     459185 non-null  object
 10  spin_dir                        0 non-null       float64
 11  spin_rate_deprecated            0 non-null       float64
 12  break_angle_deprecated          0 non-null       float64
 13  break_length_deprecated         0 non-null       float64
 14  zone                            437503 non-null  float64
 15  des                             459184 non-null  object
 16  game_type                       459185 non-null  object
 17  stand                           459185 non-null  object
 18  p_throws                        459185 non-null  object
 19  home_team                       459185 non-null  object
 20  away_team                       459185 non-null  object
 21  type                            459185 non-null  object
 22  hit_location                    108130 non-null  float64
 23  bb_type                         90752 non-null   object
 24  balls                           459185 non-null  int64
 25  strikes                         459185 non-null  int64
 26  game_year                       459185 non-null  int64
 27  pfx_x                           437492 non-null  float64
 28  pfx_z                           437492 non-null  float64
 29  plate_x                         437503 non-null  float64
 30  plate_z                         437503 non-null  float64
 31  on_3b                           43982 non-null   float64
 32  on_2b                           87192 non-null   float64
 33  on_1b                           138014 non-null  float64
 34  outs_when_up                    459185 non-null  int64
 35  inning                          459185 non-null  int64
 36  inning_topbot                   459185 non-null  object
 37  hc_x                            82377 non-null   float64
 38  hc_y                            82377 non-null   float64
 39  tfs_deprecated                  0 non-null       float64
 40  tfs_zulu_deprecated             0 non-null       float64
 41  fielder_2                       459185 non-null  int64
 42  umpire                          0 non-null       float64
 43  sv_id                           437521 non-null  object
 44  vx0                             437492 non-null  float64
 45  vy0                             437492 non-null  float64
 46  vz0                             437492 non-null  float64
 47  ax                              437503 non-null  float64
 48  ay                              437503 non-null  float64
 49  az                              437503 non-null  float64
 50  sz_top                          437503 non-null  float64
 51  sz_bot                          437503 non-null  float64
```

```
 52  hit_distance_sc                  0 non-null       float64
 53  launch_speed                     0 non-null       float64
 54  launch_angle                     0 non-null       float64
 55  effective_speed                  0 non-null       float64
 56  release_spin_rate                0 non-null       float64
 57  release_extension                0 non-null       float64
 58  game_pk                          459185 non-null  int64
 59  pitcher.1                        459185 non-null  int64
 60  fielder_2.1                      459185 non-null  int64
 61  fielder_3                        459185 non-null  int64
 62  fielder_4                        459185 non-null  int64
 63  fielder_5                        459185 non-null  int64
 64  fielder_6                        459185 non-null  int64
 65  fielder_7                        459185 non-null  int64
 66  fielder_8                        459185 non-null  int64
 67  fielder_9                        459185 non-null  int64
 68  release_pos_y                    437492 non-null  float64
 69  estimated_ba_using_speedangle    0 non-null       float64
 70  estimated_woba_using_speedangle  0 non-null       float64
 71  woba_value                       122797 non-null  float64
 72  woba_denom                       0 non-null       float64
 73  babip_value                      122797 non-null  float64
 74  iso_value                        122797 non-null  float64
 75  launch_speed_angle               0 non-null       float64
 76  at_bat_number                    459185 non-null  int64
 77  pitch_number                     459185 non-null  int64
 78  pitch_name                       437522 non-null  object
 79  home_score                       459185 non-null  int64
 80  away_score                       459185 non-null  int64
 81  bat_score                        459185 non-null  int64
 82  fld_score                        459185 non-null  int64
 83  post_away_score                  459185 non-null  int64
 84  post_home_score                  459185 non-null  int64
 85  post_bat_score                   459185 non-null  int64
 86  post_fld_score                   459185 non-null  int64
 87  if_fielding_alignment            0 non-null       float64
 88  of_fielding_alignment            0 non-null       float64
 89  spin_axis                        0 non-null       float64
 90  delta_home_win_exp               459185 non-null  float64
 91  delta_run_exp                    449456 non-null  float64
dtypes: float64(48), int64(28), object(16)
memory usage: 325.8+ MB
```

In [11]:

```
all_2008_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprecat
                               'spin_axis', 'effective_speed', 'releas
all_2008_stats_df.head()
```

Out[11]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player_name	pitche
0	FF	2008-09-28	94.4	2.17	6.08	Rhodes, Arthur	12112
1	FF	2008-09-28	93.0	2.21	6.03	Rhodes, Arthur	12112
2	NaN	2008-09-26	NaN	NaN	NaN	Rhodes, Arthur	12112
3	NaN	2008-09-26	NaN	NaN	NaN	Rhodes, Arthur	12112
4	FF	2008-09-22	92.2	1.87	6.72	Rhodes, Arthur	12112

In [12]:
```
all_2008_stats_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 459185 entries, 0 to 459184
Data columns (total 17 columns):
 #   Column         Non-Null Count   Dtype  
---  ------         --------------   -----  
 0   pitch_type     437522 non-null  object 
 1   game_date      459185 non-null  object 
 2   release_speed  437503 non-null  float64
 3   release_pos_x  437492 non-null  float64
 4   release_pos_z  437492 non-null  float64
 5   player_name    459185 non-null  object 
 6   pitcher        459185 non-null  int64  
 7   p_throws       459185 non-null  object 
 8   game_year      459185 non-null  int64  
 9   vx0            437492 non-null  float64
 10  vy0            437492 non-null  float64
 11  vz0            437492 non-null  float64
 12  ax             437503 non-null  float64
 13  ay             437503 non-null  float64
 14  az             437503 non-null  float64
 15  release_pos_y  437492 non-null  float64
 16  pitch_name     437522 non-null  object 
dtypes: float64(10), int64(2), object(5)
memory usage: 63.1+ MB
```

In [13]:
```
all_2008_stats_df = all_2008_stats_df.dropna(axis=0)
```

In [14]:
```
all_2008_stats_df.reset_index(inplace=True)
```

In [15]:
```
all_2008_stats_df.drop('index', axis=1)
```

Out[15]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player_name
0	FF	2008-09-28	94.4	2.17	6.08	Rhodes, Arthur
1	FF	2008-09-28	93.0	2.21	6.03	Rhodes, Arthur
2	FF	2008-09-22	92.2	1.87	6.72	Rhodes, Arthur
3	SL	2008-09-22	83.5	1.67	6.77	Rhodes, Arthur
4	FF	2008-09-22	92.7	1.94	6.68	Rhodes, Arthur
...	...	...	...	...	...	...
437487	SI	2008-04-05	91.7	-1.14	6.77	Wainwright, Adam
437488	SI	2008-04-05	91.2	-1.23	6.71	Wainwright, Adam
437489	SI	2008-04-05	91.5	-0.97	6.65	Wainwright, Adam
437490	CU	2008-04-05	72.3	-1.15	6.93	Wainwright, Adam
437491	SI	2008-04-05	90.0	-1.01	6.71	Wainwright, Adam

437492 rows × 17 columns

In [16]:

```
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2008_stats_df.groupby(['game_date', 'pitcher', 'pl

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the s
total_pitches_by_type = all_2008_stats_df.groupby(['game_date', 'pitch

# Calculate averages of the specified metrics for each pitch type, gro
avg_metrics = all_2008_stats_df.groupby(['game_date', 'pitcher', 'play
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2008_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2008_df = grouped_2008_df.merge(avg_metrics, on=['game_date',

grouped_2008_df
```

Out[16]:

	game_date	pitcher	player_name	total_pitches	pitch_type	count_by_pitch_type	rele
0	2008-03-28	112526	Colon, Bartolo	39	FF	10	
1	2008-03-28	112526	Colon, Bartolo	39	SI	17	
2	2008-03-28	112526	Colon, Bartolo	39	SL	12	
3	2008-03-28	118120	Maddux, Greg	1	SI	1	
4	2008-03-28	121556	Rusch, Glendon	8	FF	4	
...	...	...	...	...	...	...	
28419	2008-09-30	448147	Blackburn, Nick	89	CH	3	
28420	2008-09-30	448147	Blackburn, Nick	89	CU	4	
28421	2008-09-30	448147	Blackburn, Nick	89	FC	21	
28422	2008-09-30	448147	Blackburn, Nick	89	IN	4	
28423	2008-09-30	448147	Blackburn, Nick	89	SI	57	

28424 rows × 16 columns

In [17]:

```
grouped_2008_df['game_date'] = pd.to_datetime(grouped_2008_df['game_da
grouped_2008_df['season'] = grouped_2008_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2008_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2008_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos
for col in weighted_avg_columns:
    grouped_2008_df[f'{col}_product'] = grouped_2008_df[col] * grouped

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2008_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg

# Merge season totals and weighted averages
final_2008_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2008_df = pd.merge(final_2008_df, weighted_avg_df, on=['pitcher'

final_2008_df.head()
```

Out[17]:

	pitcher	player_name	season	season_total_pitches	pitch_type	season_total_count_by_pit
0	110683	Batista, Miguel	2008	10287	CH	
1	110683	Batista, Miguel	2008	10287	CU	
2	110683	Batista, Miguel	2008	10287	FC	
3	110683	Batista, Miguel	2008	10287	FF	
4	110683	Batista, Miguel	2008	10287	IN	

In [18]:

```
final_2008_df['player_name'] = final_2008_df['player_name'].str.lower(
final_2008_df
```

Out[18]:

	pitcher	player_name	season	season_total_pitches	pitch_type	season_total_count_by
0	110683	batista, miguel	2008	10287	CH	
1	110683	batista, miguel	2008	10287	CU	
2	110683	batista, miguel	2008	10287	FC	
3	110683	batista, miguel	2008	10287	FF	
4	110683	batista, miguel	2008	10287	IN	
...	...	...	...	...	...	
1574	502188	samardzija, jeff	2008	1820	FS	
1575	502188	samardzija, jeff	2008	1820	IN	
1576	502188	samardzija, jeff	2008	1820	PO	
1577	502188	samardzija, jeff	2008	1820	SI	
1578	502188	samardzija, jeff	2008	1820	SL	

1579 rows × 17 columns

In [19]:

```
print(final_2008_df['player_name'].unique())
```

```
['batista, miguel' 'brocail, doug' 'byrd, paul' 'carpenter, chris'
 'colon, bartolo' 'dessens, elmer' 'estes, shawn' 'eyre, scott'
 'glavine, tom' 'gordon, tom' 'hampton, mike' 'hawkins, latroy'
 'hernandez, livan' 'isringhausen, jason' 'lieber, jon' 'loaiza, esteban'
 'lowe, derek' 'maddux, greg' 'martinez, pedro' 'mercker, kent'
 'millwood, kevin' 'moehler, brian' 'morris, matt' 'moyer, jamie'
 'mussina, mike' 'nomo, hideo' 'oliver, darren' 'park, chan ho'
 'pettitte, andy' 'reyes, dennys' 'rhodes, arthur' 'rogers, kenny'
 'rusch, glendon' 'smoltz, john' 'springer, russ' 'sturtze, tanyon'
 'suppan, jeff' 'tavarez, julian' 'tomko, brett' 'torres, salomon'
 'trachsel, steve' 'villone, ron' 'wakefield, tim' 'weathers, david'
 'williams, woody' 'wright, jamey' 'looper, braden' 'washburn, jarrod'
 'ponson, sidney' 'dempster, ryan' 'elarton, scott' 'wood, kerry'
 'vazquez, javier' 'pavano, carl' 'perez, odalis' 'nunez, vladimir'
 'halladay, roy' 'schoeneweis, scott' 'wolf, randy' 'figueroa, nelson'
 'redman, mark' 'nathan, joe' 'davis, doug' 'marquis, jason'
 'moss, damian' 'fogg, josh' 'wells, kip' 'burnett, a.j.' 'armas, tony'
 'lilly, ted' 'westbrook, jake' 'mulder, mark' 'glover, gary'
 'penny, brad' 'franklin, ryan' 'zito, barry' 'hudson, tim'
 'padilla, vicente' 'meche, gil' 'durbin, chad' 'downs, scott'
 'chacon, shawn' 'santana, johan' 'arroyo, bronson' 'benoit, joaquín'
 'beckett, josh' 'belisle, matt' 'garland, jon' 'buehrle, mark'
 'sabathia, cc' 'vargas, claudio' 'sheets, ben' 'eaton, adam'
 'dickey, r.a.' 'pineiro, joel' 'affeldt, jeremy' 'lohse, kyle'
 'duckworth, brandon' 'cook, aaron' 'oswalt, roy' 'redding, tim'
 'silva, carlos' 'ramirez, horacio' 'moseley, dustin' 'fossum, casey'
 'zambrano, carlos' 'jennings, jason' 'lackey, john' 'de la rosa, jorge'
 'backe, brandon' 'bedard, erik' 'sosa, jorge' 'myers, brett'
 'peavy, jake' 'harang, aaron' 'pérez, oliver' 'saarloos, kirk'
 'hernandez, runelvys' 'hendrickson, mark' 'robertson, nate'
 'davis, jason' 'tallet, brian' 'guthrie, jeremy' 'wang, chien-ming'
 'gobble, jimmy' 'wellemeyer, todd' 'mcclung, seth' 'kuo, hung-chih'
 'cabrera, daniel' 'webb, brandon' 'carrasco, d.j.' 'ledezma, wilfredo'
 'contreras, jose' 'wainwright, adam' 'bonser, boof' 'bonderman, jeremy'
 'greinke, zack' 'harden, rich' 'floyd, gavin' 'willis, dontrelle'
 'haren, dan' 'hill, shawn' 'jackson, edwin' 'maine, john'
 'santana, ervin' 'correia, kevin' 'gaudin, chad' 'simon, alfredo'
 'blanton, joe' 'snell, ian' 'mcgowan, dustin' 'germano, justin'
 'maholm, paul' 'cain, matt' 'hamels, cole' 'kazmir, scott' 'young, chri
s'
 'danks, john' 'thompson, brad' 'hernandez, roberto' 'francis, jeff'
 'hernández, félix' 'petit, yusmeiro' 'bush, dave' 'loe, kameron'
 'verlander, justin' 'liriano, francisco' 'tejeda, robinson'
 'saunders, joe' 'jiménez, ubaldo' 'hammel, jason' 'burres, brian'
 'rodriguez, wandy' 'mendoza, luis' 'sánchez, aníbal' 'davies, kyle'
 'rowland-smith, ryan' 'reyes, anthony' 'duke, zach' 'olsen, scott'
 'mccarthy, brandon' 'miner, zach' 'niemann, jeff' 'karstens, jeff'
 'laffey, aaron' 'feldman, scott' 'nolasco, ricky' 'marshall, sean'
 'stults, eric' 'chavez, jesse' 'eveland, dana' 'litsch, jesse'
 'bannister, brian' 'blackburn, nick' 'parra, manny' 'hill, rich'
 'shields, james' 'wells, randy' 'garcía, jaime' 'volquez, edinson'
 'weaver, jered' 'wilson, c.j.' 'galarraga, armando' 'billingsley, chad'
 'gallardo, yovani' 'marcum, shaun' 'owings, micah' 'lester, jon'
 'kendrick, kyle' 'gorzelanny, tom' 'kennedy, ian' 'miller, andrew'
 'leblanc, wade' 'scherzer, max' 'lincecum, tim' 'buchholz, clay'
 'morrow, brandon' 'richard, clayton' 'villanueva, carlos' 'hensley, cla
y'
```

```
'ohlendorf, ross' 'price, david' 'sánchez, jonathan' 'cueto, johnny'
'bergmann, jason' 'bailey, homer' 'harrison, matt' 'jurrjens, jair'
'reyes, jo-jo' 'happ, j.a.' 'sonnanstine, andy' 'volstad, chris'
'lannan, john' 'slowey, kevin' 'humber, philip' 'hochevar, luke'
'pelfrey, mike' 'sowers, jeremy' 'braden, dallas' 'gonzález, gio'
'hughes, phil' 'estrada, marco' 'morales, franklin' 'masterson, justin'
'niese, jonathon' 'kershaw, clayton' 'hunter, tommy' 'garza, matt'
'kuroda, hiroki' 'matsuzaka, daisuke' 'samardzija, jeff']
```

In [20]:

```
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2008_df['player_name'] = final_2008_df['player_name'].apply(clea
```

In [21]:

```
# Convert 'player_name' from "last name, first name" to "first name la
final_2008_df['Name'] = final_2008_df['player_name'].apply(lambda x: '
```

Finally!

Now merge the two DF on 'Name' and 'season'.

In [28]:

```
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2008_df = pd.merge(final_2008_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
```

In [29]:
```
better_2008_df
```

Out[29]:

	pitcher	player_name	season	season_total_pitches	pitch_type	season_total_count_by
0	110683	batista, miguel	2008	10287	CH	
1	110683	batista, miguel	2008	10287	CU	
2	110683	batista, miguel	2008	10287	FC	
3	110683	batista, miguel	2008	10287	FF	
4	110683	batista, miguel	2008	10287	IN	
...	...	...	...	...	...	
1621	502188	samardzija, jeff	2008	1820	FS	
1622	502188	samardzija, jeff	2008	1820	IN	
1623	502188	samardzija, jeff	2008	1820	PO	
1624	502188	samardzija, jeff	2008	1820	SI	
1625	502188	samardzija, jeff	2008	1820	SL	

1626 rows × 19 columns

In [30]:
```
better_2008_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1626 entries, 0 to 1625
Data columns (total 19 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   pitcher                           1626 non-null   int64
 1   player_name                       1626 non-null   object
 2   season                            1626 non-null   int32
 3   season_total_pitches              1626 non-null   int64
 4   pitch_type                        1626 non-null   object
 5   season_total_count_by_pitch_type  1626 non-null   int64
 6   count_by_pitch_type               1626 non-null   int64
 7   release_speed_weighted_avg        1626 non-null   float64
 8   release_pos_x_weighted_avg        1626 non-null   float64
 9   release_pos_z_weighted_avg        1626 non-null   float64
 10  vx0_weighted_avg                  1626 non-null   float64
 11  vy0_weighted_avg                  1626 non-null   float64
 12  vz0_weighted_avg                  1626 non-null   float64
 13  ax_weighted_avg                   1626 non-null   float64
 14  ay_weighted_avg                   1626 non-null   float64
 15  az_weighted_avg                   1626 non-null   float64
 16  release_pos_y_weighted_avg        1626 non-null   float64
 17  Name                              1626 non-null   object
 18  Age                               1358 non-null   float64
dtypes: float64(11), int32(1), int64(4), object(3)
memory usage: 235.1+ KB
```

In [31]:
```
"""
better_2008_df.to_csv('better_2008_df.csv')
"""
```

Out[31]:
```
"\nbetter_2008_df.to_csv('better_2008_df.csv')\n"
```

This process was done for all years from 2008-2023.

Will concat all 'better' DF in data_cleaning_notebook_4

In [2]:
```
"""
all_2010_stats_df = pd.read_csv('all_2010_stats_df.csv', index_col=0)
"""
```

In [3]:

```
"""
all_2010_stats_df.info()
"""
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 474758 entries, 0 to 474757
Data columns (total 92 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   pitch_type                      471786 non-null  object
 1   game_date                       474758 non-null  object
 2   release_speed                   471783 non-null  float64
 3   release_pos_x                   471770 non-null  float64
 4   release_pos_z                   471770 non-null  float64
 5   player_name                     474758 non-null  object
 6   batter                          474758 non-null  int64
 7   pitcher                         474758 non-null  int64
 8   events                          124976 non-null  object
 9   description                     474758 non-null  object
 10  spin_dir                        0 non-null       float64
 11  spin_rate_deprecated            0 non-null       float64
 12  break_angle_deprecated          0 non-null       float64
 13  break_length_deprecated         0 non-null       float64
 14  zone                            471783 non-null  float64
 15  des                             474757 non-null  object
 16  game_type                       474758 non-null  object
 17  stand                           474758 non-null  object
 18  p_throws                        474758 non-null  object
 19  home_team                       474758 non-null  object
 20  away_team                       474758 non-null  object
 21  type                            474758 non-null  object
 22  hit_location                    110347 non-null  float64
 23  bb_type                         91270 non-null   object
 24  balls                           474758 non-null  int64
 25  strikes                         474758 non-null  int64
 26  game_year                       474758 non-null  int64
 27  pfx_x                           471770 non-null  float64
 28  pfx_z                           471770 non-null  float64
 29  plate_x                         471783 non-null  float64
 30  plate_z                         471783 non-null  float64
 31  on_3b                           44276 non-null   float64
 32  on_2b                           87596 non-null   float64
 33  on_1b                           140420 non-null  float64
 34  outs_when_up                    474758 non-null  int64
 35  inning                          474758 non-null  int64
 36  inning_topbot                   474758 non-null  object
 37  hc_x                            84545 non-null   float64
 38  hc_y                            84545 non-null   float64
 39  tfs_deprecated                  0 non-null       float64
 40  tfs_zulu_deprecated             0 non-null       float64
 41  fielder_2                       474758 non-null  int64
 42  umpire                          0 non-null       float64
 43  sv_id                           471806 non-null  object
 44  vx0                             471770 non-null  float64
 45  vy0                             471770 non-null  float64
 46  vz0                             471770 non-null  float64
 47  ax                              471783 non-null  float64
 48  ay                              471783 non-null  float64
 49  az                              471783 non-null  float64
 50  sz_top                          471783 non-null  float64
 51  sz_bot                          471783 non-null  float64
```

```
 52  hit_distance_sc                  0 non-null       float64
 53  launch_speed                     0 non-null       float64
 54  launch_angle                     0 non-null       float64
 55  effective_speed                  0 non-null       float64
 56  release_spin_rate                0 non-null       float64
 57  release_extension                0 non-null       float64
 58  game_pk                          474758 non-null  int64
 59  pitcher.1                        474758 non-null  int64
 60  fielder_2.1                      474758 non-null  int64
 61  fielder_3                        474758 non-null  int64
 62  fielder_4                        474758 non-null  int64
 63  fielder_5                        474758 non-null  int64
 64  fielder_6                        474758 non-null  int64
 65  fielder_7                        474758 non-null  int64
 66  fielder_8                        474758 non-null  int64
 67  fielder_9                        474758 non-null  int64
 68  release_pos_y                    471770 non-null  float64
 69  estimated_ba_using_speedangle    0 non-null       float64
 70  estimated_woba_using_speedangle  0 non-null       float64
 71  woba_value                       124977 non-null  float64
 72  woba_denom                       0 non-null       float64
 73  babip_value                      124977 non-null  float64
 74  iso_value                        124977 non-null  float64
 75  launch_speed_angle               0 non-null       float64
 76  at_bat_number                    474758 non-null  int64
 77  pitch_number                     474758 non-null  int64
 78  pitch_name                       471786 non-null  object
 79  home_score                       474758 non-null  int64
 80  away_score                       474758 non-null  int64
 81  bat_score                        474758 non-null  int64
 82  fld_score                        474758 non-null  int64
 83  post_away_score                  474758 non-null  int64
 84  post_home_score                  474758 non-null  int64
 85  post_bat_score                   474758 non-null  int64
 86  post_fld_score                   474758 non-null  int64
 87  if_fielding_alignment            0 non-null       float64
 88  of_fielding_alignment            0 non-null       float64
 89  spin_axis                        0 non-null       float64
 90  delta_home_win_exp               474758 non-null  float64
 91  delta_run_exp                    472124 non-null  float64
dtypes: float64(48), int64(28), object(16)
memory usage: 336.9+ MB
```

In [4]:

```
"""
all_2010_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2010_stats_df.head()
"""
```

Out[4]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player_name	pitche
0	SL	2010-10-03	82.8	1.82	6.45	Rhodes, Arthur	12112
1	SL	2010-10-03	81.1	1.76	6.46	Rhodes, Arthur	12112
2	FF	2010-10-03	90.3	1.94	6.31	Rhodes, Arthur	12112
3	SL	2010-10-03	80.4	1.71	6.55	Rhodes, Arthur	12112
4	SL	2010-10-03	80.7	1.76	6.52	Rhodes, Arthur	12112

5 rows × 21 columns

In [5]:

```
"""
all_2010_stats_df.info()
"""
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 474758 entries, 0 to 474757
Data columns (total 21 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   pitch_type         471786 non-null  object
 1   game_date          474758 non-null  object
 2   release_speed      471783 non-null  float64
 3   release_pos_x      471770 non-null  float64
 4   release_pos_z      471770 non-null  float64
 5   player_name        474758 non-null  object
 6   pitcher            474758 non-null  int64
 7   p_throws           474758 non-null  object
 8   game_year          474758 non-null  int64
 9   vx0                471770 non-null  float64
 10  vy0                471770 non-null  float64
 11  vz0                471770 non-null  float64
 12  ax                 471783 non-null  float64
 13  ay                 471783 non-null  float64
 14  az                 471783 non-null  float64
 15  effective_speed    0 non-null       float64
 16  release_spin_rate  0 non-null       float64
 17  release_extension  0 non-null       float64
 18  release_pos_y      471770 non-null  float64
 19  pitch_name         471786 non-null  object
 20  spin_axis          0 non-null       float64
dtypes: float64(14), int64(2), object(5)
memory usage: 79.7+ MB
```

In [5]:

```
"""
all_2010_stats_df.drop(columns=['spin_axis', 'effective_speed',
                                'release_spin_rate', 'release_extension
all_2010_stats_df.head()
"""
```

Out[5]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player_name	pitche
0	SL	2010-10-03	82.8	1.82	6.45	Rhodes, Arthur	12112
1	SL	2010-10-03	81.1	1.76	6.46	Rhodes, Arthur	12112
2	FF	2010-10-03	90.3	1.94	6.31	Rhodes, Arthur	12112
3	SL	2010-10-03	80.4	1.71	6.55	Rhodes, Arthur	12112
4	SL	2010-10-03	80.7	1.76	6.52	Rhodes, Arthur	12112

In [6]:
```
"""
all_2010_stats_df = all_2010_stats_df.dropna(axis=0)
"""
```

In [7]:
```
"""
all_2010_stats_df.reset_index(inplace=True)
"""
```

In [8]:
```
"""
all_2010_stats_df.drop('index', axis=1)
"""
```

Out[8]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player_name
0	SL	2010-10-03	82.8	1.82	6.45	Rhodes, Arthur
1	SL	2010-10-03	81.1	1.76	6.46	Rhodes, Arthur
2	FF	2010-10-03	90.3	1.94	6.31	Rhodes, Arthur
3	SL	2010-10-03	80.4	1.71	6.55	Rhodes, Arthur
4	SL	2010-10-03	80.7	1.76	6.52	Rhodes, Arthur
...	...	...	...	...	...	...
471765	SI	2010-04-07	91.6	-1.45	6.52	Wainwright, Adam
471766	SI	2010-04-07	92.1	-1.29	6.58	Wainwright, Adam
471767	SI	2010-04-07	91.8	-1.33	6.57	Wainwright, Adam
471768	SI	2010-04-07	90.7	-1.28	6.65	Wainwright, Adam
471769	SI	2010-04-07	92.2	-1.20	6.58	Wainwright, Adam

471770 rows × 17 columns

In [10]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2010_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2010_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2010_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2010_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2010_df = grouped_2010_df.merge(avg_metrics, on=['game_date',

grouped_2010_df
"""
```

Out[10]:

	game_date	pitcher	player_name	total_pitches	pitch_type	count_by_pitch_type	rele
0	2010-04-04	120221	Park, Chan Ho	21	CH	2	
1	2010-04-04	120221	Park, Chan Ho	21	CU	1	
2	2010-04-04	120221	Park, Chan Ho	21	FF	3	
3	2010-04-04	120221	Park, Chan Ho	21	SI	8	
4	2010-04-04	120221	Park, Chan Ho	21	SL	7	
...	...	...	...	...	...	...	
30332	2010-10-03	502009	Latos, Mat	82	SI	20	
30333	2010-10-03	502009	Latos, Mat	82	SL	20	
30334	2010-10-03	519242	Sale, Chris	30	CH	1	
30335	2010-10-03	519242	Sale, Chris	30	SI	16	
30336	2010-10-03	519242	Sale, Chris	30	SL	13	

30337 rows × 16 columns

In [11]:

```
"""
grouped_2010_df['game_date'] = pd.to_datetime(grouped_2010_df['game_da
grouped_2010_df['season'] = grouped_2010_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2010_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2010_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2010_df[f'{col}_product'] = grouped_2010_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2010_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2010_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2010_df = pd.merge(final_2010_df, weighted_avg_df, on=['pitcher'

final_2010_df.head()
"""
```

Out[11]:

	pitcher	player_name	season	season_total_pitches	pitch_type	season_total_count_by_pit
0	110683	Batista, Miguel	2010	4616	CH	
1	110683	Batista, Miguel	2010	4616	CU	
2	110683	Batista, Miguel	2010	4616	FF	
3	110683	Batista, Miguel	2010	4616	IN	
4	110683	Batista, Miguel	2010	4616	PO	

In [12]:

```
#final_2010_df.info()
```

. . .

In [13]:
```
#final_2010_df
```

...

In [20]:
```
"""
final_2010_df['player_name'] = final_2010_df['player_name'].str.lower(
final_2010_df
"""
```

...

In [21]:
```
#print(final_2010_df['player_name'].unique())
```

...

In [34]:
```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2010_df['player_name'] = final_2010_df['player_name'].apply(clea
"""
```

Out[34]:
```
'\ndef remove_accents(input_str):\n    nfkd_form = unicodedata.normalize
(\'NFKD\', input_str)\n    return "".join([c for c in nfkd_form if not un
icodedata.combining(c)])\n\ndef clean_name(name):\n    name = name.lower
()\n    name = remove_accents(name)\n    name = re.sub(r\'[-.]\', \'\', n
ame)\n    name = re.sub(r\'\\s+\', \' \', name).strip()\n    return name
\n\nfinal_2010_df[\'player_name\'] = final_2010_df[\'player_name\'].apply
(clean_name)\n'
```

In [23]:
```
#final_2010_df
```

...

In [29]:
```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2010_df['Name'] = final_2010_df['player_name'].apply(lambda x: '
"""
```

In [30]:
```
#final_2010_df
```

...

In [35]:
```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2010_df = pd.merge(final_2010_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [36]:
```
#better_2010_df
```

...

In [37]:
```
#better_2010_df.info()
```

...

In [39]:
```
"""
# Filter the DataFrame to show only rows where 'Age' is NaN
nan_age_rows = better_2010_df[better_2010_df['Age'].isna()]
nan_age_rows
"""
```

...

In [41]:
```
#better_2010_df.to_csv('better_2010_df.csv')
```

2009 DF

In [42]:
```
#all_2009_stats_df = pd.read_csv('all_2009_stats_df.csv', index_col=0)
```

In [43]:

```
"""
all_2009_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2009_stats_df.head()
"""
```

...

In [44]:

```
#all_2009_stats_df.info()
```

...

In [45]:

```
"""
all_2009_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
all_2009_stats_df.head()
"""
```

...

In [46]:

```
#all_2009_stats_df = all_2009_stats_df.dropna(axis=0)
```

In [49]:

```
#all_2009_stats_df.reset_index(inplace=True)
```

In [50]:

```
#all_2009_stats_df.drop('index', axis=1)
```

...

In [51]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2009_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2009_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2009_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2009_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2009_df = grouped_2009_df.merge(avg_metrics, on=['game_date',

grouped_2009_df
"""
```

...

In [52]:

```
"""
grouped_2009_df['game_date'] = pd.to_datetime(grouped_2009_df['game_da
grouped_2009_df['season'] = grouped_2009_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2009_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2009_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2009_df[f'{col}_product'] = grouped_2009_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2009_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2009_df = pd.merge(season_total_pitches, season_total_by_pitch_ty
final_2009_df = pd.merge(final_2009_df, weighted_avg_df, on=['pitcher'

final_2009_df.head()
"""
```

. . .

In [53]:

```
#final_2009_df.info()
```

. . .

In [54]:

```
#final_2009_df
```

. . .

In [55]:

```
"""
final_2009_df['player_name'] = final_2009_df['player_name'].str.lower(
final_2009_df
"""
```

. . .

In [56]:
```
#print(final_2010_df['player_name'].unique())
```

...

In [58]:
```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2009_df['player_name'] = final_2009_df['player_name'].apply(clea
"""
```

In [59]:
```
#final_2009_df
```

...

In [60]:
```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2009_df['Name'] = final_2009_df['player_name'].apply(lambda x: '
"""
```

In [61]:
```
#final_2009_df
```

...

In [62]:
```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2009_df = pd.merge(final_2009_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [63]:
```
#better_2009_df
```

...

In [64]:
```
#better_2009_df.info()
```

...

In [65]:
```
#better_2009_df.to_csv('better_2009_df.csv')
```

2011

In [84]:
```
#all_2011_stats_df = pd.read_csv('all_2011_stats_df.csv', index_col=0)
```

In [85]:
```
"""
all_2011_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_depreca
all_2011_stats_df.head()
"""
```

. . .

In [86]:
```
"""
all_2011_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
all_2011_stats_df.head()
"""
```

. . .

In [87]:
```
#all_2011_stats_df = all_2011_stats_df.dropna(axis=0)
```

In [88]:
```
#all_2011_stats_df.reset_index(inplace=True)
```

In [89]:
```
#all_2011_stats_df.drop('index', axis=1)
```

. . .

In [90]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2011_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2011_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2011_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2011_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2011_df = grouped_2011_df.merge(avg_metrics, on=['game_date',

grouped_2011_df
"""
```

...

In [91]:

```
"""
grouped_2011_df['game_date'] = pd.to_datetime(grouped_2011_df['game_da
grouped_2011_df['season'] = grouped_2011_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2011_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2011_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2011_df[f'{col}_product'] = grouped_2011_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2011_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2011_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2011_df = pd.merge(final_2011_df, weighted_avg_df, on=['pitcher'

final_2011_df.head()
"""
```

...

In [92]:

```
"""
final_2011_df['player_name'] = final_2011_df['player_name'].str.lower(
final_2011_df
"""
```

...

In [93]:

```
#print(final_2011_df['player_name'].unique())
```

...

In [94]:
```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2011_df['player_name'] = final_2011_df['player_name'].apply(clea
"""
```

In [95]:
```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2011_df['Name'] = final_2011_df['player_name'].apply(lambda x: '
"""
```

In [96]:
```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2011_df = pd.merge(final_2011_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [114]:
```
#better_2011_df
```

...

In [115]:
```
#better_2011_df.info()
```

...

In [99]:
```
#better_2011_df.to_csv('better_2011_df.csv')
```

2012

In [100]:
```
#all_2012_stats_df = pd.read_csv('all_2012_stats_df.csv', index_col=0)
```

In [101]:

```
"""
all_2012_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2012_stats_df.head()
"""
```

. . .

In [102]:

```
"""
all_2012_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
all_2012_stats_df.head()
"""
```

. . .

In [104]:

```
#all_2012_stats_df = all_2012_stats_df.dropna(axis=0)
```

In [105]:

```
#all_2012_stats_df.reset_index(inplace=True)
```

In [106]:

```
#all_2012_stats_df.drop('index', axis=1)
```

. . .

In [107]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2012_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2012_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2012_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2012_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2012_df = grouped_2012_df.merge(avg_metrics, on=['game_date',

grouped_2012_df
"""
```

...

In [108]:

```
"""
grouped_2012_df['game_date'] = pd.to_datetime(grouped_2012_df['game_da
grouped_2012_df['season'] = grouped_2012_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2012_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2012_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2012_df[f'{col}_product'] = grouped_2012_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2012_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2012_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2012_df = pd.merge(final_2012_df, weighted_avg_df, on=['pitcher'

final_2012_df.head()
"""
```

...

In [109]:

```
"""
final_2012_df['player_name'] = final_2012_df['player_name'].str.lower(
final_2012_df
"""
```

...

In [110]:

```
#print(final_2012_df['player_name'].unique())
```

...

In [111]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2012_df['player_name'] = final_2012_df['player_name'].apply(clea
"""
```

In [112]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2012_df['Name'] = final_2012_df['player_name'].apply(lambda x: '
"""
```

In [113]:

```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2012_df = pd.merge(final_2012_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [116]:

```
#better_2012_df
```

...

In [117]:

```
#better_2012_df.info()
```

...

In [118]:

```
#better_2012_df.to_csv('better_2012_df.csv')
```

2013

In [119]:

```
#all_2013_stats_df = pd.read_csv('all_2013_stats_df.csv', index_col=0)
```

In [120]:

```
"""all_2013_stats_df.drop(columns=['batter', 'events', 'description',
                                   'des', 'game_type', 'stand', 'home_tea
                                   'away_team', 'type', 'hit_location', '
                                   'balls', 'strikes', 'pfx_x', 'spin_dir
                                   'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                   'on_2b', 'on_1b', 'outs_when_up', 'inn
                                   'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                   'umpire', 'sv_id', 'hit_distance_sc',
                                    'sz_bot', 'launch_speed', 'launch_ang
                                   'pitcher.1', 'fielder_2.1', 'fielder_3
                                   'fielder_5', 'fielder_6', 'fielder_7',
                                   'fielder_9', 'estimated_ba_using_speed
                                   'estimated_woba_using_speedangle', 'ba
                                   'launch_speed_angle', 'woba_value', 'w
                                   'at_bat_number', 'pitch_number', 'home
                                   'bat_score', 'fld_score', 'post_home_s
                                   'post_fld_score', 'post_away_score', '
                                   'of_fielding_alignment', 'delta_home_w
                                   'delta_run_exp', 'spin_rate_deprecated
                                  'break_length_deprecated', 'tfs_deprec
all_2013_stats_df.head()
"""
```

. . .

In [121]:

```
"""
all_2013_stats_df.drop(columns=['spin_axis', 'effective_speed',
                                 'release_spin_rate', 'release_extension
"""
```

In [122]:

```
#all_2013_stats_df = all_2013_stats_df.dropna(axis=0)
```

In [123]:

```
#all_2013_stats_df.reset_index(inplace=True)
```

In [124]:

```
#all_2013_stats_df.drop('index', axis=1)
```

. . .

In [125]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2013_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2013_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2013_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2013_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2013_df = grouped_2013_df.merge(avg_metrics, on=['game_date',

grouped_2013_df
"""
```

. . .

In [126]:

```
"""
grouped_2013_df['game_date'] = pd.to_datetime(grouped_2013_df['game_da
grouped_2013_df['season'] = grouped_2013_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2013_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2013_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2013_df[f'{col}_product'] = grouped_2013_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2013_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2013_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2013_df = pd.merge(final_2013_df, weighted_avg_df, on=['pitcher'

final_2013_df.head()
"""
```

. . .

In [127]:

```
"""
final_2013_df['player_name'] = final_2013_df['player_name'].str.lower(
final_2013_df
"""
```

. . .

In [128]:

```
#print(final_2013_df['player_name'].unique())
```

. . .

In [129]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2013_df['player_name'] = final_2013_df['player_name'].apply(clea
"""
```

In [130]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2013_df['Name'] = final_2013_df['player_name'].apply(lambda x: '
"""
```

In [131]:

```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2013_df = pd.merge(final_2013_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [133]:

```
#better_2013_df
```

...

In [134]:

```
#better_2013_df.info()
```

...

In [135]:

```
#better_2013_df.to_csv('better_2013_df.csv')
```

2014

In [136]:

```
#all_2014_stats_df = pd.read_csv('all_2014_stats_df.csv', index_col=0)
```

In [137]:

```
"""
all_2014_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2014_stats_df.head()
"""
```

...

In [138]:

```
"""
all_2014_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
"""
```

In [139]:

```
#all_2014_stats_df = all_2014_stats_df.dropna(axis=0)
```

In [140]:

```
#all_2014_stats_df.reset_index(inplace=True)
```

In [141]:

```
#all_2014_stats_df.drop('index', axis=1)
```

...

In [142]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2014_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2014_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2014_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2014_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2014_df = grouped_2014_df.merge(avg_metrics, on=['game_date',

grouped_2014_df.head()
"""
```

...

In [143]:

```
"""
grouped_2014_df['game_date'] = pd.to_datetime(grouped_2014_df['game_da
grouped_2014_df['season'] = grouped_2014_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2014_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2014_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2014_df[f'{col}_product'] = grouped_2014_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2014_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2014_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2014_df = pd.merge(final_2014_df, weighted_avg_df, on=['pitcher'

final_2014_df.head()
"""
```

...

In [144]:

```
"""
final_2014_df['player_name'] = final_2014_df['player_name'].str.lower(
final_2014_df.head()
"""
```

...

In [145]:

```
#print(final_2014_df['player_name'].unique())
```

...

In [146]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2014_df['player_name'] = final_2014_df['player_name'].apply(clea
"""
```

In [147]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2014_df['Name'] = final_2014_df['player_name'].apply(lambda x: '
"""
```

In [148]:

```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2014_df = pd.merge(final_2014_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [151]:

```
#better_2014_df.head()
```

...

In [152]:

```
#better_2014_df.info()
```

...

In [153]:

```
#better_2014_df.to_csv('better_2014_df.csv')
```

2015

In [154]:

```
#all_2015_stats_df = pd.read_csv('all_2015_stats_df.csv', index_col=0)
```

In [155]:

```
"""
all_2015_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2015_stats_df.head()
"""
```

...

In [156]:

```
"""
all_2015_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
"""
```

In [157]:

```
#all_2015_stats_df = all_2015_stats_df.dropna(axis=0)
```

In [158]:

```
#all_2015_stats_df.reset_index(inplace=True)
```

In [159]:

```
#all_2015_stats_df.drop('index', axis=1)
```

...

In [160]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2015_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2015_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2015_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2015_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2015_df = grouped_2015_df.merge(avg_metrics, on=['game_date',

grouped_2015_df.head()
"""
```

...

In [161]:

```
"""
grouped_2015_df['game_date'] = pd.to_datetime(grouped_2015_df['game_da
grouped_2015_df['season'] = grouped_2015_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2015_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2015_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2015_df[f'{col}_product'] = grouped_2015_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2015_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2015_df = pd.merge(season_total_pitches, season_total_by_pitch_ty
final_2015_df = pd.merge(final_2015_df, weighted_avg_df, on=['pitcher'

final_2015_df.head()
"""
```

...

In [162]:

```
"""
final_2015_df['player_name'] = final_2015_df['player_name'].str.lower(
final_2015_df.head()
"""
```

...

In [163]:

```
#print(final_2015_df['player_name'].unique())
```

...

In [164]:
```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2015_df['player_name'] = final_2015_df['player_name'].apply(clea
"""
```

In [165]:
```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2015_df['Name'] = final_2015_df['player_name'].apply(lambda x: '
"""
```

In [166]:
```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2015_df = pd.merge(final_2015_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [167]:
```
#better_2015_df.head()
```

...

In [168]:
```
#better_2015_df.info()
```

...

In [169]:
```
#better_2015_df.to_csv('better_2015_df.csv')
```

2016

In [279]:
```
#all_2016_stats_df = pd.read_csv('all_2016_stats_df.csv', index_col=0)
```

In [280]:

```
"""
all_2016_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2016_stats_df.head()
"""
```

Out[280]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player_name	pitche
0	SI	2016-09-27	91.5	-2.44	6.38	Nathan, Joe	15027
1	CU	2016-09-27	80.6	-2.11	6.73	Nathan, Joe	15027
2	FF	2016-09-27	91.5	-1.98	6.80	Nathan, Joe	15027
3	FF	2016-09-27	91.4	-2.10	6.78	Nathan, Joe	15027
4	CU	2016-09-27	79.3	-2.02	6.99	Nathan, Joe	15027

5 rows × 21 columns

In [281]:

```
"""
all_2016_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
"""
```

In [282]:

```
#all_2016_stats_df = all_2016_stats_df.dropna(axis=0)
```

In [283]:

```
#all_2016_stats_df.reset_index(inplace=True)
```

In [284]:

```
#all_2016_stats_df.drop('index', axis=1)
```

. . .

In [285]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2016_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2016_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2016_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2016_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2016_df = grouped_2016_df.merge(avg_metrics, on=['game_date',

grouped_2016_df.head()
"""
```

. . .

In [286]:

```
"""
grouped_2016_df['game_date'] = pd.to_datetime(grouped_2016_df['game_da
grouped_2016_df['season'] = grouped_2016_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2016_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2016_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2016_df[f'{col}_product'] = grouped_2016_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2016_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2016_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2016_df = pd.merge(final_2016_df, weighted_avg_df, on=['pitcher'

final_2016_df.head()
"""
```

. . .

In [287]:

```
"""
final_2016_df['player_name'] = final_2016_df['player_name'].str.lower(
final_2016_df.head()
"""
```

Out[287]:

	pitcher	player_name	season	season_total_pitches	pitch_type	season_total_count_by_pit
0	112526	colon, bartolo	2016	11394	CH	
1	112526	colon, bartolo	2016	11394	FF	
2	112526	colon, bartolo	2016	11394	IN	
3	112526	colon, bartolo	2016	11394	SI	
4	112526	colon, bartolo	2016	11394	SL	

In [288]:
```
#print(final_2016_df['player_name'].unique())
```

...

In [289]:
```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2016_df['player_name'] = final_2016_df['player_name'].apply(clea
"""
```

In [290]:
```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2016_df['Name'] = final_2016_df['player_name'].apply(lambda x: '
"""
```

In [291]:
```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2016_df = pd.merge(final_2016_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [292]:
```
#better_2016_df.to_csv('better_2016_df.csv')
```

In [184]:
```
#better_2016_df.info()
```

...

In [185]:
```
#better_2016_df.head()
```

...

2017

In [186]:
```
#all_2017_stats_df = pd.read_csv('all_2017_stats_df.csv', index_col=0)
```

In [187]:

```
"""all_2017_stats_df.drop(columns=['batter', 'events', 'description',
                                   'des', 'game_type', 'stand', 'home_tea
                                   'away_team', 'type', 'hit_location', '
                                   'balls', 'strikes', 'pfx_x', 'spin_dir
                                   'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                   'on_2b', 'on_1b', 'outs_when_up', 'inn
                                   'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                   'umpire', 'sv_id', 'hit_distance_sc',
                                    'sz_bot', 'launch_speed', 'launch_ang
                                   'pitcher.1', 'fielder_2.1', 'fielder_3
                                   'fielder_5', 'fielder_6', 'fielder_7',
                                   'fielder_9', 'estimated_ba_using_speed
                                   'estimated_woba_using_speedangle', 'ba
                                   'launch_speed_angle', 'woba_value', 'w
                                   'at_bat_number', 'pitch_number', 'home
                                   'bat_score', 'fld_score', 'post_home_s
                                   'post_fld_score', 'post_away_score', '
                                   'of_fielding_alignment', 'delta_home_w
                                   'delta_run_exp', 'spin_rate_deprecated
                                  'break_length_deprecated', 'tfs_deprec
all_2017_stats_df.head()
"""
```

. . .

In [188]:

```
"""
all_2017_stats_df.drop(columns=['spin_axis', 'effective_speed',
                                'release_spin_rate', 'release_extension
"""
```

In [189]:

```
#all_2017_stats_df = all_2017_stats_df.dropna(axis=0)
```

In [190]:

```
#all_2017_stats_df.reset_index(inplace=True)
```

In [191]:

```
#all_2017_stats_df.drop('index', axis=1)
```

. . .

In [192]:

```
"""# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2017_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2017_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2017_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2017_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2017_df = grouped_2017_df.merge(avg_metrics, on=['game_date',

grouped_2017_df.head()
"""
```

. . .

In [193]:

```
"""
grouped_2017_df['game_date'] = pd.to_datetime(grouped_2017_df['game_da
grouped_2017_df['season'] = grouped_2017_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2017_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2017_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2017_df[f'{col}_product'] = grouped_2017_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2017_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2017_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2017_df = pd.merge(final_2017_df, weighted_avg_df, on=['pitcher'

final_2017_df.head()
"""
```

. . .

In [194]:

```
"""final_2017_df['player_name'] = final_2017_df['player_name'].str.low
final_2017_df.head()
"""
```

. . .

In [195]:

```
#print(final_2017_df['player_name'].unique())
```

. . .

In [196]:
```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2017_df['player_name'] = final_2017_df['player_name'].apply(clea
"""
```

In [197]:
```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2017_df['Name'] = final_2017_df['player_name'].apply(lambda x: '
"""
```

In [198]:
```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2017_df = pd.merge(final_2017_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [199]:
```
#better_2017_df.head()
```

...

In [200]:
```
#better_2017_df.info()
```

...

In [201]:
```
#better_2017_df.to_csv('better_2017_df.csv')
```

2018

In [6]:
```
#all_2018_stats_df = pd.read_csv('all_2018_stats_df.csv', index_col=0)
```

In [8]:
```
#all_2018_stats_df.info()
```

...

In [203]:

```
"""
all_2018_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2018_stats_df.head()
"""
```

...

In [204]:

```
"""
all_2018_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
"""
```

In [205]:

```
#all_2018_stats_df = all_2018_stats_df.dropna(axis=0)
```

In [206]:

```
#all_2018_stats_df.reset_index(inplace=True)
```

In [207]:

```
#all_2018_stats_df.drop('index', axis=1)
```

...

In [208]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2018_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2018_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2018_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2018_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2018_df = grouped_2018_df.merge(avg_metrics, on=['game_date',

grouped_2018_df.head()
"""
```

. . .

In [209]:

```
"""grouped_2018_df['game_date'] = pd.to_datetime(grouped_2018_df['game_
grouped_2018_df['season'] = grouped_2018_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2018_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2018_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2018_df[f'{col}_product'] = grouped_2018_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2018_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2018_df = pd.merge(season_total_pitches, season_total_by_pitch_ty
final_2018_df = pd.merge(final_2018_df, weighted_avg_df, on=['pitcher'

final_2018_df.head()
"""
```

. . .

In [210]:

```
"""
final_2018_df['player_name'] = final_2018_df['player_name'].str.lower(
final_2018_df.head()
"""
```

. . .

In [211]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2018_df['player_name'] = final_2018_df['player_name'].apply(clea
"""
```

In [212]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2018_df['Name'] = final_2018_df['player_name'].apply(lambda x: '
"""
```

In [213]:

```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2018_df = pd.merge(final_2018_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [214]:

```
#better_2018_df.head()
```

...

In [215]:

```
#better_2018_df.info()
```

...

In [216]:

```
#better_2018_df.to_csv('better_2018_df.csv')
```

2019

In [217]:

```
#all_2019_stats_df = pd.read_csv('all_2019_stats_df.csv', index_col=0)
```

In [218]:

```
"""
all_2019_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2019_stats_df.head()
"""
```

...

In [219]:

```
"""
all_2019_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
"""
```

In [220]:

```
#all_2019_stats_df = all_2019_stats_df.dropna(axis=0)
```

In [221]:

```
#all_2019_stats_df.reset_index(inplace=True)
```

In [222]:

```
#all_2019_stats_df.drop('index', axis=1)
```

...

In [223]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2019_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2019_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2019_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2019_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2019_df = grouped_2019_df.merge(avg_metrics, on=['game_date',

grouped_2019_df.head()
"""
```

. . .

In [224]:

```
"""
grouped_2019_df['game_date'] = pd.to_datetime(grouped_2019_df['game_da
grouped_2019_df['season'] = grouped_2019_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2019_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2019_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2019_df[f'{col}_product'] = grouped_2019_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2019_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2019_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2019_df = pd.merge(final_2019_df, weighted_avg_df, on=['pitcher'

final_2019_df.head()
"""
```

. . .

In [225]:

```
"""
final_2019_df['player_name'] = final_2019_df['player_name'].str.lower(
final_2019_df.head()
"""
```

. . .

In [226]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2019_df['player_name'] = final_2019_df['player_name'].apply(clea
"""
```

In [227]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2019_df['Name'] = final_2019_df['player_name'].apply(lambda x: '
"""
```

In [228]:

```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2019_df = pd.merge(final_2019_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [229]:

```
#better_2019_df.info()
```

...

In [230]:

```
#better_2019_df.head()
```

...

In [231]:

```
#better_2019_df.to_csv('better_2019_df.csv')
```

2020

In [232]:

```
#all_2020_stats_df = pd.read_csv('all_2020_stats_df.csv', index_col=0)
```

In [233]:

```
"""all_2020_stats_df.drop(columns=['batter', 'events', 'description',
                                   'des', 'game_type', 'stand', 'home_tea
                                   'away_team', 'type', 'hit_location', '
                                   'balls', 'strikes', 'pfx_x', 'spin_dir
                                   'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                   'on_2b', 'on_1b', 'outs_when_up', 'inn
                                   'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                   'umpire', 'sv_id', 'hit_distance_sc',
                                    'sz_bot', 'launch_speed', 'launch_ang
                                   'pitcher.1', 'fielder_2.1', 'fielder_3
                                   'fielder_5', 'fielder_6', 'fielder_7',
                                   'fielder_9', 'estimated_ba_using_speed
                                   'estimated_woba_using_speedangle', 'ba
                                   'launch_speed_angle', 'woba_value', 'w
                                   'at_bat_number', 'pitch_number', 'home
                                   'bat_score', 'fld_score', 'post_home_s
                                   'post_fld_score', 'post_away_score', '
                                   'of_fielding_alignment', 'delta_home_w
                                   'delta_run_exp', 'spin_rate_deprecated
                                  'break_length_deprecated', 'tfs_deprec
all_2020_stats_df.head()
"""
```

...

In [234]:

```
"""
all_2020_stats_df.drop(columns=['spin_axis', 'effective_speed',
                                 'release_spin_rate', 'release_extension
"""
```

In [235]:

```
#all_2020_stats_df = all_2020_stats_df.dropna(axis=0)
```

In [236]:

```
#all_2020_stats_df.reset_index(inplace=True)
```

In [237]:

```
#all_2020_stats_df.drop('index', axis=1)
```

...

In [238]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2020_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2020_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2020_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2020_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2020_df = grouped_2020_df.merge(avg_metrics, on=['game_date',

grouped_2020_df.head()
"""
```

. . .

In [239]:

```
"""
grouped_2020_df['game_date'] = pd.to_datetime(grouped_2020_df['game_da
grouped_2020_df['season'] = grouped_2020_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2020_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2020_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2020_df[f'{col}_product'] = grouped_2020_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2020_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2020_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2020_df = pd.merge(final_2020_df, weighted_avg_df, on=['pitcher'

final_2020_df.head()
"""
```

. . .

In [240]:

```
"""
final_2020_df['player_name'] = final_2020_df['player_name'].str.lower(
final_2020_df.head()
"""
```

. . .

In [241]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2020_df['player_name'] = final_2020_df['player_name'].apply(clea
"""
```

In [242]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2020_df['Name'] = final_2020_df['player_name'].apply(lambda x: '
"""
```

In [243]:

```
"""# Now, you can perform the merge using 'Name' and 'season' as the k
better_2020_df = pd.merge(final_2020_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [244]:

```
#better_2020_df.info()
```

...

In [245]:

```
#better_2020_df.head()
```

...

In [246]:

```
#better_2020_df.to_csv('better_2020_df.csv')
```

2021

In [247]:

```
#all_2021_stats_df = pd.read_csv('all_2021_stats_df.csv', index_col=0)
```

In [248]:

```
"""
all_2021_stats_df.drop(columns=['batter', 'events', 'description', 'zo
                                'des', 'game_type', 'stand', 'home_tea
                                'away_team', 'type', 'hit_location', '
                                'balls', 'strikes', 'pfx_x', 'spin_dir
                                'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                'on_2b', 'on_1b', 'outs_when_up', 'inn
                                'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                'umpire', 'sv_id', 'hit_distance_sc',
                                 'sz_bot', 'launch_speed', 'launch_ang
                                'pitcher.1', 'fielder_2.1', 'fielder_3
                                'fielder_5', 'fielder_6', 'fielder_7',
                                'fielder_9', 'estimated_ba_using_speed
                                'estimated_woba_using_speedangle', 'ba
                                'launch_speed_angle', 'woba_value', 'w
                                'at_bat_number', 'pitch_number', 'home
                                'bat_score', 'fld_score', 'post_home_s
                                'post_fld_score', 'post_away_score', '
                                'of_fielding_alignment', 'delta_home_w
                                'delta_run_exp', 'spin_rate_deprecated
                               'break_length_deprecated', 'tfs_deprec
all_2021_stats_df.head()
"""
```

...

In [249]:

```
"""
all_2021_stats_df.drop(columns=['spin_axis', 'effective_speed',
                               'release_spin_rate', 'release_extension
"""
```

In [250]:

```
#all_2021_stats_df = all_2021_stats_df.dropna(axis=0)
```

In [251]:

```
#all_2021_stats_df.reset_index(inplace=True)
```

In [252]:

```
#all_2021_stats_df.drop('index', axis=1)
```

...

In [253]:

```
"""# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2021_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2021_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2021_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2021_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2021_df = grouped_2021_df.merge(avg_metrics, on=['game_date',

grouped_2021_df.head()
"""
```

. . .

In [254]:

```
"""
grouped_2021_df['game_date'] = pd.to_datetime(grouped_2021_df['game_da
grouped_2021_df['season'] = grouped_2021_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2021_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2021_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2021_df[f'{col}_product'] = grouped_2021_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2021_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2021_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2021_df = pd.merge(final_2021_df, weighted_avg_df, on=['pitcher'

final_2021_df.head()
"""
```

. . .

In [255]:

```
"""
final_2021_df['player_name'] = final_2021_df['player_name'].str.lower(
final_2021_df.head()
"""
```

. . .

In [256]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2021_df['player_name'] = final_2021_df['player_name'].apply(clea
"""
```

In [257]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2021_df['Name'] = final_2021_df['player_name'].apply(lambda x: '
"""
```

In [258]:

```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2021_df = pd.merge(final_2021_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [259]:

```
#better_2021_df.head()
```

...

In [260]:

```
#better_2021_df.info()
```

...

In [261]:

```
#better_2021_df.to_csv('better_2021_df.csv')
```

2022

In [262]:

```
#all_2022_stats_df = pd.read_csv('all_2022_stats_df.csv', index_col=0)
```

In [263]:

```
"""all_2022_stats_df.drop(columns=['batter', 'events', 'description',
                                   'des', 'game_type', 'stand', 'home_tea
                                   'away_team', 'type', 'hit_location', '
                                   'balls', 'strikes', 'pfx_x', 'spin_dir
                                   'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                   'on_2b', 'on_1b', 'outs_when_up', 'inn
                                   'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                   'umpire', 'sv_id', 'hit_distance_sc',
                                    'sz_bot', 'launch_speed', 'launch_ang
                                   'pitcher.1', 'fielder_2.1', 'fielder_3
                                   'fielder_5', 'fielder_6', 'fielder_7',
                                   'fielder_9', 'estimated_ba_using_speed
                                   'estimated_woba_using_speedangle', 'ba
                                   'launch_speed_angle', 'woba_value', 'w
                                   'at_bat_number', 'pitch_number', 'home
                                   'bat_score', 'fld_score', 'post_home_s
                                   'post_fld_score', 'post_away_score', '
                                   'of_fielding_alignment', 'delta_home_w
                                   'delta_run_exp', 'spin_rate_deprecated
                                  'break_length_deprecated', 'tfs_deprec
all_2022_stats_df.head()
"""
```

...

In [264]:

```
"""all_2022_stats_df.drop(columns=['spin_axis', 'effective_speed',
                                  'release_spin_rate', 'release_extension
"""
```

In [265]:

```
#all_2022_stats_df = all_2022_stats_df.dropna(axis=0)
```

In [266]:

```
#all_2022_stats_df.reset_index(inplace=True)
```

In [268]:

```
#all_2022_stats_df.drop('index', axis=1)
```

...

In [269]:

```
"""
# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2022_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2022_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2022_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2022_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2022_df = grouped_2022_df.merge(avg_metrics, on=['game_date',

grouped_2022_df.head()
"""
```

...

In [270]:

```
"""
grouped_2022_df['game_date'] = pd.to_datetime(grouped_2022_df['game_da
grouped_2022_df['season'] = grouped_2022_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2022_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2022_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2022_df[f'{col}_product'] = grouped_2022_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2022_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2022_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2022_df = pd.merge(final_2022_df, weighted_avg_df, on=['pitcher'

final_2022_df.head()
"""
```

. . .

In [271]:

```
"""
final_2022_df['player_name'] = final_2022_df['player_name'].str.lower(
final_2022_df.head()
"""
```

. . .

In [272]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2022_df['player_name'] = final_2022_df['player_name'].apply(clea
"""
```

In [273]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2022_df['Name'] = final_2022_df['player_name'].apply(lambda x: '
"""
```

In [274]:

```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2022_df = pd.merge(final_2022_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [275]:

```
#better_2022_df.head()
```

...

In [276]:

```
#better_2022_df.info()
```

...

In [277]:

```
#better_2022_df.to_csv('better_2022_df.csv')
```

2023

In [278]:

```
#all_2023_stats_df = pd.read_csv('all_2023_stats_df.csv', index_col=0)
```

In [293]:

```
"""all_2023_stats_df.drop(columns=['batter', 'events', 'description',
                                   'des', 'game_type', 'stand', 'home_tea
                                   'away_team', 'type', 'hit_location', '
                                   'balls', 'strikes', 'pfx_x', 'spin_dir
                                   'pfx_z', 'plate_x', 'plate_z', 'on_3b'
                                   'on_2b', 'on_1b', 'outs_when_up', 'inn
                                   'inning_topbot', 'hc_x', 'hc_y', 'fiel
                                   'umpire', 'sv_id', 'hit_distance_sc',
                                    'sz_bot', 'launch_speed', 'launch_ang
                                   'pitcher.1', 'fielder_2.1', 'fielder_3
                                   'fielder_5', 'fielder_6', 'fielder_7',
                                   'fielder_9', 'estimated_ba_using_speed
                                   'estimated_woba_using_speedangle', 'ba
                                   'launch_speed_angle', 'woba_value', 'w
                                   'at_bat_number', 'pitch_number', 'home
                                   'bat_score', 'fld_score', 'post_home_s
                                   'post_fld_score', 'post_away_score', '
                                   'of_fielding_alignment', 'delta_home_w
                                   'delta_run_exp', 'spin_rate_deprecated
                                  'break_length_deprecated', 'tfs_deprec
all_2023_stats_df.head()
"""
```

Out[293]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player_name	pitche
0	FA	2023-04-14	52.1	-1.99	6.71	Pérez, Carlos	54220
1	FA	2023-04-14	69.6	-2.30	6.58	Pérez, Carlos	54220
2	FA	2023-04-14	65.1	-2.25	6.52	Pérez, Carlos	54220
3	FA	2023-04-14	64.2	-2.37	6.50	Pérez, Carlos	54220
4	FA	2023-04-14	72.0	-2.45	6.49	Pérez, Carlos	54220

5 rows × 21 columns

In [294]:

```
"""
all_2023_stats_df.drop(columns=['spin_axis', 'effective_speed',
                                'release_spin_rate', 'release_extension
"""
```

In [295]:

```
#all_2023_stats_df = all_2023_stats_df.dropna(axis=0)
```

In [296]:

```
#all_2023_stats_df.reset_index(inplace=True)
```

In [297]:
```
#all_2023_stats_df.drop('index', axis=1)
```

Out[297]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player_name
0	FA	2023-04-14	52.1	-1.99	6.71	Pérez, Carlos
1	FA	2023-04-14	69.6	-2.30	6.58	Pérez, Carlos
2	FA	2023-04-14	65.1	-2.25	6.52	Pérez, Carlos
3	FA	2023-04-14	64.2	-2.37	6.50	Pérez, Carlos
4	FA	2023-04-14	72.0	-2.45	6.49	Pérez, Carlos
...	...	...	...	...	...	...
290106	CU	2023-04-01	70.1	3.02	5.35	Yarbrough, Ryan
290107	CU	2023-04-01	71.3	2.90	5.33	Yarbrough, Ryan
290108	CH	2023-04-01	79.6	2.71	5.48	Yarbrough, Ryan
290109	SI	2023-04-01	87.8	2.84	5.59	Yarbrough, Ryan
290110	SI	2023-04-01	86.6	2.88	5.61	Yarbrough, Ryan

290111 rows × 17 columns

In [298]:

```
"""# Group by 'game_date' and 'pitcher' to calculate the total pitches
total_pitches = all_2023_stats_df.groupby(['game_date', 'pitcher', 'pla

# Group by 'game_date', 'pitcher', and 'pitch_type' to calculate the su
total_pitches_by_type = all_2023_stats_df.groupby(['game_date', 'pitche

# Calculate averages of the specified metrics for each pitch type, grou
avg_metrics = all_2023_stats_df.groupby(['game_date', 'pitcher', 'playe
    'release_speed': 'mean',
    'release_pos_x': 'mean',
    'release_pos_z': 'mean',
    'vx0': 'mean',
    'vy0': 'mean',
    'vz0': 'mean',
    'ax': 'mean',
    'ay': 'mean',
    'az': 'mean',
    'release_pos_y': 'mean',
}).reset_index()

grouped_2023_df = total_pitches.merge(total_pitches_by_type, on=['game_
grouped_2023_df = grouped_2023_df.merge(avg_metrics, on=['game_date',

grouped_2023_df.head()
"""
```

Out[298]:

	game_date	pitcher	player_name	total_pitches	pitch_type	count_by_pitch_type	release_
0	2023-03-30	425844	Greinke, Zack	80	CH	11	87.6
1	2023-03-30	425844	Greinke, Zack	80	CU	22	73.8
2	2023-03-30	425844	Greinke, Zack	80	FC	9	86.
3	2023-03-30	425844	Greinke, Zack	80	FF	6	90.3
4	2023-03-30	425844	Greinke, Zack	80	SI	13	90.7

In [299]:

```
"""
grouped_2023_df['game_date'] = pd.to_datetime(grouped_2023_df['game_da
grouped_2023_df['season'] = grouped_2023_df['game_date'].dt.year

# Step 1: Season Total Pitches
season_total_pitches = grouped_2023_df.groupby(['pitcher', 'player_nam

# Step 2: Season Total by Pitch Type
season_total_by_pitch_type = grouped_2023_df.groupby(['pitcher', 'play

# Weighted Averages Calculation Setup
weighted_avg_columns = ['release_speed', 'release_pos_x', 'release_pos_
for col in weighted_avg_columns:
    grouped_2023_df[f'{col}_product'] = grouped_2023_df[col] * grouped_

weighted_avg_aggregations = {f'{col}_product': 'sum' for col in weight
weighted_avg_aggregations['count_by_pitch_type'] = 'sum'

# Aggregate for weighted averages
weighted_avg_df = grouped_2023_df.groupby(['pitcher', 'player_name', '

# Calculate weighted averages
for col in weighted_avg_columns:
    weighted_avg_df[f'{col}_weighted_avg'] = weighted_avg_df[f'{col}_p

# Cleanup
weighted_avg_df.drop(columns=[f'{col}_product' for col in weighted_avg_

# Merge season totals and weighted averages
final_2023_df = pd.merge(season_total_pitches, season_total_by_pitch_t
final_2023_df = pd.merge(final_2023_df, weighted_avg_df, on=['pitcher'

final_2023_df.head()
"""
```

Out[299]:

	pitcher	player_name	season	season_total_pitches	pitch_type	season_total_count_by_pit
0	425794	Wainwright, Adam	2023	9498	CH	
1	425794	Wainwright, Adam	2023	9498	CS	
2	425794	Wainwright, Adam	2023	9498	CU	
3	425794	Wainwright, Adam	2023	9498	FC	
4	425794	Wainwright, Adam	2023	9498	FF	

In [300]:

```
"""
final_2023_df['player_name'] = final_2023_df['player_name'].str.lower(
final_2023_df.head()
"""
```

Out[300]:

	pitcher	player_name	season	season_total_pitches	pitch_type	season_total_count_by_pit
0	425794	wainwright, adam	2023	9498	CH	
1	425794	wainwright, adam	2023	9498	CS	
2	425794	wainwright, adam	2023	9498	CU	
3	425794	wainwright, adam	2023	9498	FC	
4	425794	wainwright, adam	2023	9498	FF	

In [301]:

```
"""
def remove_accents(input_str):
    nfkd_form = unicodedata.normalize('NFKD', input_str)
    return "".join([c for c in nfkd_form if not unicodedata.combining(

def clean_name(name):
    name = name.lower()
    name = remove_accents(name)
    name = re.sub(r'[-.]', '', name)
    name = re.sub(r'\s+', ' ', name).strip()
    return name

final_2023_df['player_name'] = final_2023_df['player_name'].apply(clea
"""
```

In [302]:

```
"""
# Convert 'player_name' from "last name, first name" to "first name la
final_2023_df['Name'] = final_2023_df['player_name'].apply(lambda x: '
"""
```

In [303]:

```
"""
# Now, you can perform the merge using 'Name' and 'season' as the keys
better_2023_df = pd.merge(final_2023_df,
                          cleaning_filtered_df[['Name', 'season', 'Age
                          on=['Name', 'season'],
                          how='left')
"""
```

In [304]:
```
#better_2023_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 877 entries, 0 to 876
Data columns (total 19 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   pitcher                           877 non-null    int64
 1   player_name                       877 non-null    object
 2   season                            877 non-null    int32
 3   season_total_pitches              877 non-null    int64
 4   pitch_type                        877 non-null    object
 5   season_total_count_by_pitch_type  877 non-null    int64
 6   count_by_pitch_type               877 non-null    int64
 7   release_speed_weighted_avg        877 non-null    float64
 8   release_pos_x_weighted_avg        877 non-null    float64
 9   release_pos_z_weighted_avg        877 non-null    float64
 10  vx0_weighted_avg                  877 non-null    float64
 11  vy0_weighted_avg                  877 non-null    float64
 12  vz0_weighted_avg                  877 non-null    float64
 13  ax_weighted_avg                   877 non-null    float64
 14  ay_weighted_avg                   877 non-null    float64
 15  az_weighted_avg                   877 non-null    float64
 16  release_pos_y_weighted_avg        877 non-null    float64
 17  Name                              877 non-null    object
 18  Age                               815 non-null    float64
dtypes: float64(11), int32(1), int64(4), object(3)
memory usage: 126.9+ KB
```

In [305]:
```
#better_2023_df.head()
```

Out[305]:

	pitcher	player_name	season	season_total_pitches	pitch_type	season_total_count_by_pit
0	425794	wainwright, adam	2023	9498	CH	
1	425794	wainwright, adam	2023	9498	CS	
2	425794	wainwright, adam	2023	9498	CU	
3	425794	wainwright, adam	2023	9498	FC	
4	425794	wainwright, adam	2023	9498	FF	

In [306]:
```
#better_2023_df.to_csv('better_2023_df.csv')
```

END.