semesterProject

December 3, 2025

1 NBA retirement prediction

[572]:
```
# Install dependencies as needed:
# pip install kagglehub
import kagglehub
from kagglehub import KaggleDatasetAdapter
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
```

2 DATA:

3 Player Totals

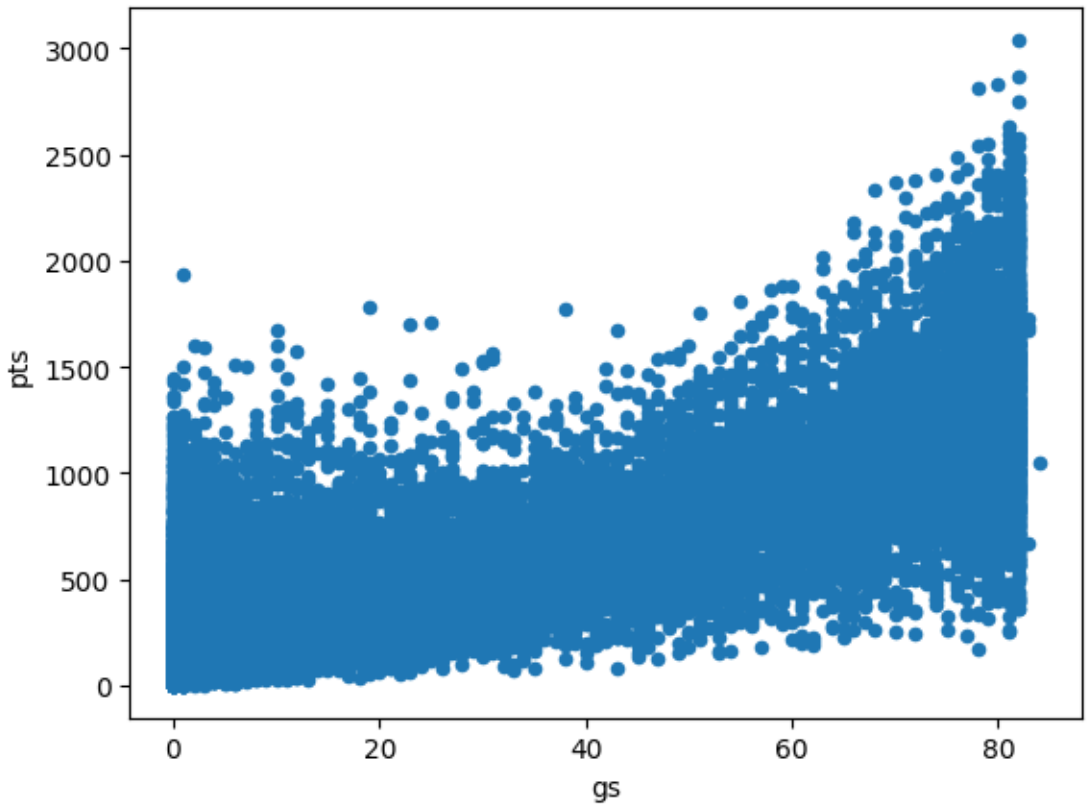
[573]:
```
# Set the path to the file you'd like to load
file_path = "Player Totals.csv"

# Load the latest version
playerTotalsDf = kagglehub.dataset_load(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
  # Provide any additional arguments like
  # sql_query or pandas_kwargs. See the
  # documenation for more information:
  # https://github.com/Kaggle/kagglehub/blob/main/README.
  ↪md#kaggledatasetadapterpandas
)

print(playerTotalsDf.head())
```

```
   season   lg            player  player_id   age team pos   g    gs     mp  \
0    2026  NBA  Precious Achiuwa  achiupr01  26.0  SAC   C  14   6.0  293.0
1    2026  NBA      Steven Adams  adamsst01  32.0  HOU   C  15   4.0  324.0
```

```
2    2026  NBA       Bam Adebayo  adebaba01  28.0  MIA   C  14  14.0  425.0
3    2026  NBA      Ochai Agbaji  agbajoc01  25.0  TOR  SG  15   1.0  204.0
4    2026  NBA      Santi Aldama  aldamsa01  25.0  MEM  PF  21   2.0  548.0

   …   orb    drb    trb  ast   stl   blk   tov    pf  pts  trp_dbl
0  …  25.0   48.0   73.0   14   9.0   6.0   7.0  24.0  102      0.0
1  …  74.0   67.0  141.0   23  10.0  10.0  15.0  29.0   97      0.0
2  …  18.0  101.0  119.0   36  13.0  11.0  26.0  20.0  264      0.0
3  …   5.0   23.0   28.0   10   6.0   2.0   5.0  26.0   46      0.0
4  …  31.0  108.0  139.0   63  20.0  16.0  23.0  26.0  282      0.0

[5 rows x 33 columns]
```

[574]:
```
playerTotalsDf.plot.scatter(x='gs', y='pts')
```

[574]: <Axes: xlabel='gs', ylabel='pts'>



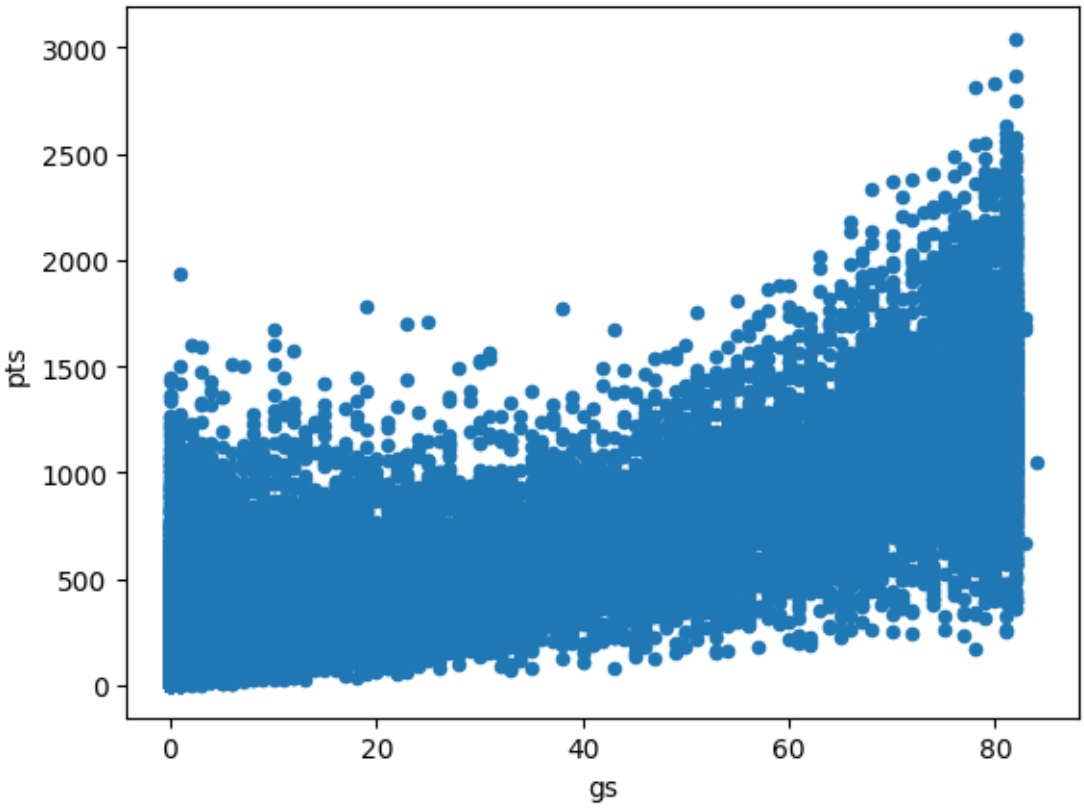
4 Player Per Game

[575]:
```
# Set the path to the file you'd like to load
file_path = "Player Per Game.csv"

# Load the latest version
playerPerGameDf = kagglehub.dataset_load(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
)

print(playerPerGameDf.head())
```

```
   season   lg            player  player_id   age team pos   g    gs  \
0    2026  NBA  Precious Achiuwa  achiupr01  26.0  SAC   C  14   6.0
1    2026  NBA      Steven Adams  adamsst01  32.0  HOU   C  15   4.0
2    2026  NBA       Bam Adebayo  adebaba01  28.0  MIA   C  14  14.0
3    2026  NBA      Ochai Agbaji  agbajoc01  25.0  TOR  SG  15   1.0
4    2026  NBA      Santi Aldama  aldamsa01  25.0  MEM  PF  21   2.0

   mp_per_game  …  ft_percent  orb_per_game  drb_per_game  trb_per_game  \
0         20.9  …       0.458           1.8           3.4           5.2
1         21.6  …       0.775           4.9           4.5           9.4
2         30.4  …       0.806           1.3           7.2           8.5
3         13.6  …       0.700           0.3           1.5           1.9
4         26.1  …       0.673           1.5           5.1           6.6

   ast_per_game  stl_per_game  blk_per_game  tov_per_game  pf_per_game  \
0           1.0           0.6           0.4           0.5          1.7
1           1.5           0.7           0.7           1.0          1.9
2           2.6           0.9           0.8           1.9          1.4
3           0.7           0.4           0.1           0.3          1.7
4           3.0           1.0           0.8           1.1          1.2

   pts_per_game
0           7.3
1           6.5
2          18.9
3           3.1
4          13.4

[5 rows x 32 columns]
```

[576]:
```
playerTotalsDf.plot.scatter(x='gs', y='pts')
```

[576]:
```
<Axes: xlabel='gs', ylabel='pts'>
```



5 Player Per 36 Mins

[577]:
```
# Set the path to the file you'd like to load
file_path = "Per 36 Minutes.csv"

# Load the latest version
per36MinsDf = kagglehub.dataset_load(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
)

print(per36MinsDf.head())
```

```
   season   lg            player  player_id   age team pos   g    gs   mp  \
0    2026  NBA  Precious Achiuwa  achiupr01  26.0  SAC   C  14   6.0  293
1    2026  NBA      Steven Adams  adamsst01  32.0  HOU   C  15   4.0  324
2    2026  NBA       Bam Adebayo  adebaba01  28.0  MIA   C  14  14.0  425
3    2026  NBA      Ochai Agbaji  agbajoc01  25.0  TOR  SG  15   1.0  204
4    2026  NBA      Santi Aldama  aldamsa01  25.0  MEM  PF  21   2.0  548
```

```
   …  ft_percent  orb_per_36_min  drb_per_36_min  trb_per_36_min  \
0  …        0.458             3.1             5.9             9.0
1  …        0.775             8.2             7.4            15.7
2  …        0.806             1.5             8.6            10.1
3  …        0.700             0.9             4.1             4.9
4  …        0.673             2.0             7.1             9.1

   ast_per_36_min  stl_per_36_min  blk_per_36_min  tov_per_36_min  \
0             1.7             1.1             0.7             0.9
1             2.6             1.1             1.1             1.7
2             3.0             1.1             0.9             2.2
3             1.8             1.1             0.4             0.9
4             4.1             1.3             1.1             1.5

   pf_per_36_min  pts_per_36_min
0            2.9            12.5
1            3.2            10.8
2            1.7            22.4
3            4.6             8.1
4            1.7            18.5

[5 rows x 32 columns]
```

6 Player Per 100 Possesions

[578]:
```
# Set the path to the file you'd like to load
file_path = "Per 100 Poss.csv"

# Load the latest version
per100PossDf = kagglehub.dataset_load(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
)

print(per100PossDf.head())
```

```
   season   lg            player  player_id  age team pos   g    gs   mp  …  \
0    2026  NBA  Precious Achiuwa  achiupr01   26  SAC   C  14   6.0  293  …
1    2026  NBA      Steven Adams  adamsst01   32  HOU   C  15   4.0  324  …
2    2026  NBA       Bam Adebayo  adebaba01   28  MIA   C  14  14.0  425  …
3    2026  NBA      Ochai Agbaji  agbajoc01   25  TOR  SG  15   1.0  204  …
4    2026  NBA      Santi Aldama  aldamsa01   25  MEM  PF  21   2.0  548  …

   drb_per_100_poss  trb_per_100_poss  ast_per_100_poss  stl_per_100_poss  \
0               7.8              11.8               2.3               1.5
1              10.3              21.7               3.5               1.5
```

```
2              10.8              12.7               3.9               1.4
3               5.4               6.6               2.4               1.4
4               9.4              12.1               5.5               1.7

   blk_per_100_poss  tov_per_100_poss  pf_per_100_poss  pts_per_100_poss  \
0               1.0               1.1              3.9              16.5
1               1.5               2.3              4.5              14.9
2               1.2               2.8              2.1              28.3
3               0.5               1.2              6.1              10.8
4               1.4               2.0              2.3              24.6

   o_rtg  d_rtg
0  117.0  120.0
1  138.0  110.0
2  112.0  109.0
3   94.0  114.0
4  116.0  113.0

[5 rows x 34 columns]
```

7 Player Advanced Stats

[579]:
```
# Set the path to the file you'd like to load
file_path = "Advanced.csv"

# Load the latest version
advancedStatsDf = kagglehub.dataset_load(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
)

print(advancedStatsDf.head())
```

```
   season   lg            player  player_id   age team pos   g    gs     mp  \
0    2026  NBA  Precious Achiuwa  achiupr01  26.0  SAC   C  14   6.0  293.0
1    2026  NBA      Steven Adams  adamsst01  32.0  HOU   C  15   4.0  324.0
2    2026  NBA       Bam Adebayo  adebaba01  28.0  MIA   C  14  14.0  425.0
3    2026  NBA      Ochai Agbaji  agbajoc01  25.0  TOR  SG  15   1.0  204.0
4    2026  NBA      Santi Aldama  aldamsa01  25.0  MEM  PF  21   2.0  548.0

   …  tov_percent  usg_percent  ows  dws   ws  ws_48  obpm  dbpm  bpm  vorp
0  …          7.1         14.3  0.3  0.2  0.4  0.068  -1.4  -0.8 -2.2   0.0
1  …         15.4         12.4  0.9  0.5  1.4  0.208   2.3  -0.3  1.9   0.3
2  …         10.1         24.5  0.3  0.8  1.1  0.119   0.9   1.1  2.0   0.4
3  …          8.6         12.1 -0.2  0.2  0.1  0.017  -6.6  -0.3 -6.9  -0.3
4  …          8.3         20.9  0.6  0.8  1.3  0.118   1.9   1.2  3.1   0.7
```

[5 rows x 30 columns]

8 Player play by play stats

[580]:
```
# Set the path to the file you'd like to load
file_path = "Player Play By Play.csv"

# Load the latest version
playByPlayDf = kagglehub.dataset_load(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
)

print(playByPlayDf.head())
```

```
   season   lg            player  player_id  age team pos   g  gs   mp  … \
0    2026  NBA  Precious Achiuwa  achiupr01   26  SAC   C  14   6  293  …
1    2026  NBA      Steven Adams  adamsst01   32  HOU   C  15   4  324  …
2    2026  NBA       Bam Adebayo  adebaba01   28  MIA   C  14  14  425  …
3    2026  NBA      Ochai Agbaji  agbajoc01   25  TOR  SG  15   1  204  …
4    2026  NBA      Santi Aldama  aldamsa01   25  MEM  PF  21   2  548  …

   net_plus_minus_per_100_poss  bad_pass_turnover  lost_ball_turnover  \
0                          6.4                  2                   1
1                          9.8                  5                   3
2                          8.5                 10                  11
3                          1.3                  2                   2
4                         -3.5                 11                   7

   shooting_foul_committed  offensive_foul_committed  shooting_foul_drawn  \
0                       12                         3                   11
1                        9                         5                   14
2                        6                         2                   33
3                       13                         1                    4
4                       15                         4                   21

   offensive_foul_drawn  points_generated_by_assists  and1  fga_blocked
0                   2.0                           31     6            4
1                   0.0                           52     1            2
2                   2.0                           83     3            7
3                   2.0                           23     0            5
4                   3.0                          160     8            7

[5 rows x 26 columns]
```

9 Height and Weight Data

[581]:
```
# Set the path to the file you'd like to load
file_path = "Players.csv"

# Load the latest version
HeightAndWeightDf = kagglehub.dataset_load(
  KaggleDatasetAdapter.PANDAS,
  "drgilermo/nba-players-stats",
  file_path,
)

HeightAndWeightDf = HeightAndWeightDf.rename(columns={'Player': 'player'})
print(HeightAndWeightDf.head())
```

```
   Unnamed: 0           player  height  weight  \
0           0  Curly Armstrong   180.0    77.0
1           1     Cliff Barker   188.0    83.0
2           2    Leo Barnhorst   193.0    86.0
3           3       Ed Bartels   196.0    88.0
4           4      Ralph Beard   178.0    79.0

                         collage    born   birth_city birth_state
0              Indiana University  1918.0          NaN         NaN
1          University of Kentucky  1921.0     Yorktown     Indiana
2        University of Notre Dame  1924.0          NaN         NaN
3  North Carolina State University  1925.0          NaN         NaN
4          University of Kentucky  1927.0  Hardinsburg    Kentucky
```

10 Injury Data

[582]:
```
# Set the path to the file you'd like to load
file_path = "injuries_2010-2020.csv"

# Load the latest version
injuryDF = kagglehub.dataset_load(
  KaggleDatasetAdapter.PANDAS,
  "ghopkins/nba-injuries-2010-2018",
  file_path,
)
injuryDF = injuryDF.rename(columns={'Relinquished': 'player'})

injury_counts = injuryDF['player'].value_counts().reset_index()
injury_counts.columns = ['player', 'num_injuries']

print(injuryDF.head())
```

```
         Date     Team Acquired         player  \
0  2010-10-03    Bulls      NaN  Carlos Boozer
1  2010-10-06  Pistons      NaN  Jonas Jerebko
2  2010-10-06  Pistons      NaN  Terrico White
3  2010-10-08  Blazers      NaN     Jeff Ayres
4  2010-10-08     Nets      NaN    Troy Murphy

                                          Notes
0  fractured bone in right pinky finger (out inde…
1      torn right Achilles tendon (out indefinitely)
2  broken fifth metatarsal in right foot (out ind…
3           torn ACL in right knee (out indefinitely)
4              strained lower back (out indefinitely)
```

11 Merging data into 1 dataset

[583]:
```
#Merge data into one big dataset-Amelia
mergedDF = playerTotalsDf
mergedDF = pd.merge(mergedDF, playerPerGameDf, on=["player_id", "season"],␣
 ↪how="outer", suffixes=("", "_per_game"))
mergedDF = pd.merge(mergedDF, per36MinsDf, on=["player_id", "season"],␣
 ↪how="outer", suffixes=("", "_per_36_mins"))
mergedDF = pd.merge(mergedDF, per100PossDf, on=["player_id", "season"],␣
 ↪how="outer", suffixes=("", "_per_100_poss"))
mergedDF = pd.merge(mergedDF, advancedStatsDf, on=["player_id", "season"],␣
 ↪how="outer", suffixes=("", "_advanced"))
finalDF = mergedDF.drop_duplicates(subset=['season', 'player'])


print(finalDF[26700:26730])
```

```
        season   lg             player  player_id   age team pos   g    gs  \
803813    1960  NBA     George Yardley  yardlge01  31.0  SYR  SF  73   NaN
803814    1972  NBA        Barry Yates  yatesba01  26.0  PHI  PF  24   1.0
803815    1962  NBA        Wayne Yates  yateswa01  24.0  LAL   C  37   NaN
803816    1972  NBA  Charlie Yelverton  yelvech01  23.0  POR  SG  69   NaN
803817    1982  NBA       Rich Yonakor  yonakri01  23.0  SAS  PF  10   0.0
803818    2022  NBA          Gabe York   yorkga01  28.0  IND  SG   2   0.0
803819    2023  NBA          Gabe York   yorkga01  29.0  IND  SG   3   0.0
803820    2026  NBA   Chris Youngblood  youngch01  23.0  OKC  SG  16   0.0
803821    1985  NBA        Danny Young  youngda01  22.0  SEA  PG   3   0.0
803822    1986  NBA        Danny Young  youngda01  23.0  SEA  PG  82  29.0
803823    1987  NBA        Danny Young  youngda01  24.0  SEA  PG  73  26.0
803824    1988  NBA        Danny Young  youngda01  25.0  SEA  PG  77   0.0
803825    1989  NBA        Danny Young  youngda01  26.0  POR  PG  48   2.0
803826    1990  NBA        Danny Young  youngda01  27.0  POR  PG  82   8.0
803827    1991  NBA        Danny Young  youngda01  28.0  POR  PG  75   1.0
```

803828	1992	NBA	Danny Young	youngda01	29.0	2TM	PG	62	5.0
804071	1993	NBA	Danny Young	youngda01	30.0	DET	PG	65	2.0
804072	1995	NBA	Danny Young	youngda01	32.0	MIL	PG	7	0.0
804073	2015	NBA	James Young	youngja01	19.0	BOS	SG	31	0.0
804074	2016	NBA	James Young	youngja01	20.0	BOS	SG	29	0.0
804075	2017	NBA	James Young	youngja01	21.0	BOS	SG	29	0.0
804076	2018	NBA	James Young	youngja01	22.0	PHI	SG	6	0.0
804077	2025	NBA	Jahmir Young	youngja05	24.0	CHI	PG	6	0.0
804078	2026	NBA	Jahmir Young	youngja05	25.0	MIA	PG	4	0.0
804079	2016	NBA	Joe Young	youngjo01	23.0	IND	PG	41	0.0
804080	2017	NBA	Joe Young	youngjo01	24.0	IND	PG	33	0.0
804081	2018	NBA	Joe Young	youngjo01	25.0	IND	PG	53	1.0
804082	1999	NBA	Korleone Young	youngko01	20.0	DET	SF	3	0.0
804083	1985	NBA	Michael Young	youngmi01	24.0	PHO	SF	2	0.0
804084	1986	NBA	Michael Young	youngmi01	25.0	PHI	SF	2	0.0

	mp	…	tov_percent	usg_percent	ows	dws	ws	ws_48	obpm	\
803813	2402.0	…	NaN	NaN	6.9	2.1	9.0	0.179	NaN	
803814	144.0	…	NaN	NaN	-0.4	0.1	-0.2	-0.075	NaN	
803815	263.0	…	NaN	NaN	-0.8	0.3	-0.5	-0.085	NaN	
803816	1227.0	…	NaN	NaN	-0.7	0.3	-0.5	-0.018	NaN	
803817	70.0	…	6.4	17.7	0.2	0.1	0.3	0.201	-1.2	
803818	21.0	…	9.8	20.8	0.0	0.0	0.0	0.059	-3.7	
803819	56.0	…	0.0	16.4	0.1	0.0	0.1	0.091	-1.7	
803820	80.0	…	3.9	13.8	0.1	0.1	0.2	0.114	-4.6	
803821	26.0	…	16.7	19.3	-0.2	0.0	-0.1	-0.188	-9.9	
803822	1901.0	…	15.7	12.9	2.7	2.1	4.8	0.121	0.2	
803823	1482.0	…	21.0	10.7	2.0	0.9	2.9	0.095	0.8	
803824	949.0	…	13.3	11.8	1.6	0.8	2.4	0.122	0.4	
803825	952.0	…	13.9	13.1	1.0	0.8	1.8	0.091	-1.1	
803826	1393.0	…	17.5	12.9	1.0	1.9	3.0	0.103	-1.2	
803827	897.0	…	14.7	15.4	0.6	1.2	1.8	0.096	-0.6	
803828	1023.0	…	14.2	13.7	0.8	1.1	1.9	0.089	-1.4	
804071	836.0	…	14.2	10.8	0.9	0.5	1.4	0.081	-1.5	
804072	77.0	…	18.7	12.3	0.2	0.0	0.2	0.141	0.9	
804073	332.0	…	4.2	15.8	0.1	0.3	0.3	0.047	-2.5	
804074	199.0	…	11.7	9.1	-0.2	0.2	0.1	0.013	-5.3	
804075	220.0	…	6.1	13.3	0.1	0.2	0.3	0.076	-1.7	
804076	61.0	…	5.7	12.3	0.0	0.0	0.1	0.041	-4.5	
804077	30.0	…	14.5	9.6	0.1	0.0	0.1	0.220	1.2	
804078	17.0	…	9.1	26.2	0.0	0.0	0.0	-0.001	-4.3	
804079	384.0	…	15.5	24.4	-0.5	0.5	-0.1	-0.009	-3.9	
804080	135.0	…	6.0	27.8	-0.2	0.1	-0.1	-0.022	-3.1	
804081	558.0	…	11.6	18.0	0.1	0.4	0.5	0.043	-2.3	
804082	15.0	…	8.4	37.6	0.1	0.0	0.1	0.280	9.9	
804083	11.0	…	0.0	22.2	0.0	0.0	0.0	-0.072	-2.5	
804084	2.0	…	0.0	40.3	0.0	0.0	0.0	-0.818	-28.9	

```
        dbpm   bpm  vorp
803813   NaN   NaN   NaN
803814   NaN   NaN   NaN
803815   NaN   NaN   NaN
803816   NaN   NaN   NaN
803817  -0.6  -1.8   0.1
803818   2.2  -1.5   0.0
803819  -1.8  -3.5   0.0
803820  -1.0  -5.6  -0.1
803821   3.8  -6.1   0.0
803822   1.7   1.9   1.9
803823   0.3   1.1   1.2
803824   0.7   1.1   0.8
803825   1.2   0.0   0.5
803826   2.1   0.9   1.0
803827   1.1   0.5   0.6
803828   0.6  -0.8   0.3
804071   1.0  -0.5   0.3
804072   2.4   3.3   0.1
804073  -0.9  -3.4  -0.1
804074   0.8  -4.5  -0.1
804075   0.4  -1.3   0.0
804076  -2.4  -6.9  -0.1
804077  -0.8   0.5   0.0
804078  -2.6  -7.0   0.0
804079  -0.8  -4.7  -0.3
804080  -2.1  -5.2  -0.1
804081  -1.2  -3.5  -0.2
804082  -0.5   9.4   0.0
804083  -5.6  -8.0   0.0
804084 -19.1 -48.0   0.0

[30 rows x 153 columns]
```

12 Creating column for year retired, NaN if still playing

[584]:
```
# players not retired yet
activePlayers = finalDF.groupby("player")["season"].max().eq(2026)

retirementYears = finalDF.groupby("player")["season"].max().
  ↪where(~activePlayers)

finalDF['retired'] = finalDF['player'].map(retirementYears)

finalDF[26700:26730]
```

```
/var/folders/_b/qm30jgm53fqdy5bnwf_lv5x00000gn/T/ipykernel_76804/2776414382.py:6
```

```
: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  finalDF['retired'] = finalDF['player'].map(retirementYears)
```

[584]:

	season	lg	player	player_id	age	team	pos	g	gs	\
803813	1960	NBA	George Yardley	yardlge01	31.0	SYR	SF	73	NaN	
803814	1972	NBA	Barry Yates	yatesba01	26.0	PHI	PF	24	1.0	
803815	1962	NBA	Wayne Yates	yateswa01	24.0	LAL	C	37	NaN	
803816	1972	NBA	Charlie Yelverton	yelvech01	23.0	POR	SG	69	NaN	
803817	1982	NBA	Rich Yonakor	yonakri01	23.0	SAS	PF	10	0.0	
803818	2022	NBA	Gabe York	yorkga01	28.0	IND	SG	2	0.0	
803819	2023	NBA	Gabe York	yorkga01	29.0	IND	SG	3	0.0	
803820	2026	NBA	Chris Youngblood	youngch01	23.0	OKC	SG	16	0.0	
803821	1985	NBA	Danny Young	youngda01	22.0	SEA	PG	3	0.0	
803822	1986	NBA	Danny Young	youngda01	23.0	SEA	PG	82	29.0	
803823	1987	NBA	Danny Young	youngda01	24.0	SEA	PG	73	26.0	
803824	1988	NBA	Danny Young	youngda01	25.0	SEA	PG	77	0.0	
803825	1989	NBA	Danny Young	youngda01	26.0	POR	PG	48	2.0	
803826	1990	NBA	Danny Young	youngda01	27.0	POR	PG	82	8.0	
803827	1991	NBA	Danny Young	youngda01	28.0	POR	PG	75	1.0	
803828	1992	NBA	Danny Young	youngda01	29.0	2TM	PG	62	5.0	
804071	1993	NBA	Danny Young	youngda01	30.0	DET	PG	65	2.0	
804072	1995	NBA	Danny Young	youngda01	32.0	MIL	PG	7	0.0	
804073	2015	NBA	James Young	youngja01	19.0	BOS	SG	31	0.0	
804074	2016	NBA	James Young	youngja01	20.0	BOS	SG	29	0.0	
804075	2017	NBA	James Young	youngja01	21.0	BOS	SG	29	0.0	
804076	2018	NBA	James Young	youngja01	22.0	PHI	SG	6	0.0	
804077	2025	NBA	Jahmir Young	youngja05	24.0	CHI	PG	6	0.0	
804078	2026	NBA	Jahmir Young	youngja05	25.0	MIA	PG	4	0.0	
804079	2016	NBA	Joe Young	youngjo01	23.0	IND	PG	41	0.0	
804080	2017	NBA	Joe Young	youngjo01	24.0	IND	PG	33	0.0	
804081	2018	NBA	Joe Young	youngjo01	25.0	IND	PG	53	1.0	
804082	1999	NBA	Korleone Young	youngko01	20.0	DET	SF	3	0.0	
804083	1985	NBA	Michael Young	youngmi01	24.0	PHO	SF	2	0.0	
804084	1986	NBA	Michael Young	youngmi01	25.0	PHI	SF	2	0.0	

	mp	…	usg_percent	ows	dws	ws	ws_48	obpm	dbpm	bpm	\
803813	2402.0	…	NaN	6.9	2.1	9.0	0.179	NaN	NaN	NaN	
803814	144.0	…	NaN	-0.4	0.1	-0.2	-0.075	NaN	NaN	NaN	
803815	263.0	…	NaN	-0.8	0.3	-0.5	-0.085	NaN	NaN	NaN	
803816	1227.0	…	NaN	-0.7	0.3	-0.5	-0.018	NaN	NaN	NaN	
803817	70.0	…	17.7	0.2	0.1	0.3	0.201	-1.2	-0.6	-1.8	
803818	21.0	…	20.8	0.0	0.0	0.0	0.059	-3.7	2.2	-1.5	

803819	56.0	…	16.4	0.1	0.0	0.1	0.091	-1.7	-1.8	-3.5
803820	80.0	…	13.8	0.1	0.1	0.2	0.114	-4.6	-1.0	-5.6
803821	26.0	…	19.3	-0.2	0.0	-0.1	-0.188	-9.9	3.8	-6.1
803822	1901.0	…	12.9	2.7	2.1	4.8	0.121	0.2	1.7	1.9
803823	1482.0	…	10.7	2.0	0.9	2.9	0.095	0.8	0.3	1.1
803824	949.0	…	11.8	1.6	0.8	2.4	0.122	0.4	0.7	1.1
803825	952.0	…	13.1	1.0	0.8	1.8	0.091	-1.1	1.2	0.0
803826	1393.0	…	12.9	1.0	1.9	3.0	0.103	-1.2	2.1	0.9
803827	897.0	…	15.4	0.6	1.2	1.8	0.096	-0.6	1.1	0.5
803828	1023.0	…	13.7	0.8	1.1	1.9	0.089	-1.4	0.6	-0.8
804071	836.0	…	10.8	0.9	0.5	1.4	0.081	-1.5	1.0	-0.5
804072	77.0	…	12.3	0.2	0.0	0.2	0.141	0.9	2.4	3.3
804073	332.0	…	15.8	0.1	0.3	0.3	0.047	-2.5	-0.9	-3.4
804074	199.0	…	9.1	-0.2	0.2	0.1	0.013	-5.3	0.8	-4.5
804075	220.0	…	13.3	0.1	0.2	0.3	0.076	-1.7	0.4	-1.3
804076	61.0	…	12.3	0.0	0.0	0.1	0.041	-4.5	-2.4	-6.9
804077	30.0	…	9.6	0.1	0.0	0.1	0.220	1.2	-0.8	0.5
804078	17.0	…	26.2	0.0	0.0	0.0	-0.001	-4.3	-2.6	-7.0
804079	384.0	…	24.4	-0.5	0.5	-0.1	-0.009	-3.9	-0.8	-4.7
804080	135.0	…	27.8	-0.2	0.1	-0.1	-0.022	-3.1	-2.1	-5.2
804081	558.0	…	18.0	0.1	0.4	0.5	0.043	-2.3	-1.2	-3.5
804082	15.0	…	37.6	0.1	0.0	0.1	0.280	9.9	-0.5	9.4
804083	11.0	…	22.2	0.0	0.0	0.0	-0.072	-2.5	-5.6	-8.0
804084	2.0	…	40.3	0.0	0.0	0.0	-0.818	-28.9	-19.1	-48.0

	vorp	retired
803813	NaN	1960.0
803814	NaN	1972.0
803815	NaN	1962.0
803816	NaN	1972.0
803817	0.1	1982.0
803818	0.0	2023.0
803819	0.0	2023.0
803820	-0.1	NaN
803821	0.0	1995.0
803822	1.9	1995.0
803823	1.2	1995.0
803824	0.8	1995.0
803825	0.5	1995.0
803826	1.0	1995.0
803827	0.6	1995.0
803828	0.3	1995.0
804071	0.3	1995.0
804072	0.1	1995.0
804073	-0.1	2018.0
804074	-0.1	2018.0
804075	0.0	2018.0

```
804076  -0.1   2018.0
804077   0.0      NaN
804078   0.0      NaN
804079  -0.3   2018.0
804080  -0.1   2018.0
804081  -0.2   2018.0
804082   0.0   1999.0
804083   0.0   1990.0
804084   0.0   1990.0

[30 rows x 154 columns]
```

[585]:
```
ret_year_mp = finalDF.loc[finalDF['season'] == finalDF['retired'],␣
  ↪['player_id', 'retired', 'mp']].rename(columns={'mp': 'mp_at_retirement'})
finalDF = finalDF.merge(ret_year_mp, on=['player_id', 'retired'], how='left')
finalDF[26700:26730]
```

[585]:

	season	lg	player	player_id	age	team	pos	g	gs	\
26700	1960	NBA	George Yardley	yardlge01	31.0	SYR	SF	73	NaN	
26701	1972	NBA	Barry Yates	yatesba01	26.0	PHI	PF	24	1.0	
26702	1962	NBA	Wayne Yates	yateswa01	24.0	LAL	C	37	NaN	
26703	1972	NBA	Charlie Yelverton	yelvech01	23.0	POR	SG	69	NaN	
26704	1982	NBA	Rich Yonakor	yonakri01	23.0	SAS	PF	10	0.0	
26705	2022	NBA	Gabe York	yorkga01	28.0	IND	SG	2	0.0	
26706	2023	NBA	Gabe York	yorkga01	29.0	IND	SG	3	0.0	
26707	2026	NBA	Chris Youngblood	youngch01	23.0	OKC	SG	16	0.0	
26708	1985	NBA	Danny Young	youngda01	22.0	SEA	PG	3	0.0	
26709	1986	NBA	Danny Young	youngda01	23.0	SEA	PG	82	29.0	
26710	1987	NBA	Danny Young	youngda01	24.0	SEA	PG	73	26.0	
26711	1988	NBA	Danny Young	youngda01	25.0	SEA	PG	77	0.0	
26712	1989	NBA	Danny Young	youngda01	26.0	POR	PG	48	2.0	
26713	1990	NBA	Danny Young	youngda01	27.0	POR	PG	82	8.0	
26714	1991	NBA	Danny Young	youngda01	28.0	POR	PG	75	1.0	
26715	1992	NBA	Danny Young	youngda01	29.0	2TM	PG	62	5.0	
26716	1993	NBA	Danny Young	youngda01	30.0	DET	PG	65	2.0	
26717	1995	NBA	Danny Young	youngda01	32.0	MIL	PG	7	0.0	
26718	2015	NBA	James Young	youngja01	19.0	BOS	SG	31	0.0	
26719	2016	NBA	James Young	youngja01	20.0	BOS	SG	29	0.0	
26720	2017	NBA	James Young	youngja01	21.0	BOS	SG	29	0.0	
26721	2018	NBA	James Young	youngja01	22.0	PHI	SG	6	0.0	
26722	2025	NBA	Jahmir Young	youngja05	24.0	CHI	PG	6	0.0	
26723	2026	NBA	Jahmir Young	youngja05	25.0	MIA	PG	4	0.0	
26724	2016	NBA	Joe Young	youngjo01	23.0	IND	PG	41	0.0	
26725	2017	NBA	Joe Young	youngjo01	24.0	IND	PG	33	0.0	
26726	2018	NBA	Joe Young	youngjo01	25.0	IND	PG	53	1.0	
26727	1999	NBA	Korleone Young	youngko01	20.0	DET	SF	3	0.0	
26728	1985	NBA	Michael Young	youngmi01	24.0	PHO	SF	2	0.0	

```
26729    1986  NBA      Michael Young  youngmi01  25.0  PHI  SF   2   0.0

          mp  …  ows  dws   ws  ws_48  obpm  dbpm   bpm  vorp  retired  \
26700  2402.0  …  6.9  2.1  9.0  0.179   NaN   NaN   NaN   NaN   1960.0
26701   144.0  … -0.4  0.1 -0.2 -0.075   NaN   NaN   NaN   NaN   1972.0
26702   263.0  … -0.8  0.3 -0.5 -0.085   NaN   NaN   NaN   NaN   1962.0
26703  1227.0  … -0.7  0.3 -0.5 -0.018   NaN   NaN   NaN   NaN   1972.0
26704    70.0  …  0.2  0.1  0.3  0.201  -1.2  -0.6  -1.8   0.1   1982.0
26705    21.0  …  0.0  0.0  0.0  0.059  -3.7   2.2  -1.5   0.0   2023.0
26706    56.0  …  0.1  0.0  0.1  0.091  -1.7  -1.8  -3.5   0.0   2023.0
26707    80.0  …  0.1  0.1  0.2  0.114  -4.6  -1.0  -5.6  -0.1      NaN
26708    26.0  … -0.2  0.0 -0.1 -0.188  -9.9   3.8  -6.1   0.0   1995.0
26709  1901.0  …  2.7  2.1  4.8  0.121   0.2   1.7   1.9   1.9   1995.0
26710  1482.0  …  2.0  0.9  2.9  0.095   0.8   0.3   1.1   1.2   1995.0
26711   949.0  …  1.6  0.8  2.4  0.122   0.4   0.7   1.1   0.8   1995.0
26712   952.0  …  1.0  0.8  1.8  0.091  -1.1   1.2   0.0   0.5   1995.0
26713  1393.0  …  1.0  1.9  3.0  0.103  -1.2   2.1   0.9   1.0   1995.0
26714   897.0  …  0.6  1.2  1.8  0.096  -0.6   1.1   0.5   0.6   1995.0
26715  1023.0  …  0.8  1.1  1.9  0.089  -1.4   0.6  -0.8   0.3   1995.0
26716   836.0  …  0.9  0.5  1.4  0.081  -1.5   1.0  -0.5   0.3   1995.0
26717    77.0  …  0.2  0.0  0.2  0.141   0.9   2.4   3.3   0.1   1995.0
26718   332.0  …  0.1  0.3  0.3  0.047  -2.5  -0.9  -3.4  -0.1   2018.0
26719   199.0  … -0.2  0.2  0.1  0.013  -5.3   0.8  -4.5  -0.1   2018.0
26720   220.0  …  0.1  0.2  0.3  0.076  -1.7   0.4  -1.3   0.0   2018.0
26721    61.0  …  0.0  0.0  0.1  0.041  -4.5  -2.4  -6.9  -0.1   2018.0
26722    30.0  …  0.1  0.0  0.1  0.220   1.2  -0.8   0.5   0.0      NaN
26723    17.0  …  0.0  0.0  0.0 -0.001  -4.3  -2.6  -7.0   0.0      NaN
26724   384.0  … -0.5  0.5 -0.1 -0.009  -3.9  -0.8  -4.7  -0.3   2018.0
26725   135.0  … -0.2  0.1 -0.1 -0.022  -3.1  -2.1  -5.2  -0.1   2018.0
26726   558.0  …  0.1  0.4  0.5  0.043  -2.3  -1.2  -3.5  -0.2   2018.0
26727    15.0  …  0.1  0.0  0.1  0.280   9.9  -0.5   9.4   0.0   1999.0
26728    11.0  …  0.0  0.0  0.0 -0.072  -2.5  -5.6  -8.0   0.0   1990.0
26729     2.0  …  0.0  0.0  0.0 -0.818 -28.9 -19.1 -48.0   0.0   1990.0

       mp_at_retirement
26700            2402.0
26701             144.0
26702             263.0
26703            1227.0
26704              70.0
26705              56.0
26706              56.0
26707               NaN
26708              77.0
26709              77.0
26710              77.0
26711              77.0
```

```
26712              77.0
26713              77.0
26714              77.0
26715              77.0
26716              77.0
26717              77.0
26718              61.0
26719              61.0
26720              61.0
26721              61.0
26722               NaN
26723               NaN
26724             558.0
26725             558.0
26726             558.0
26727              15.0
26728             459.0
26729             459.0

[30 rows x 155 columns]
```

[586]:
```
#Make a variable that holds career length(ex: 10 years)-make a method to␣
 ↪calculate this for each player(value_count?)-Amelia
careerLen = finalDF.groupby("player_id")["season"].nunique().reset_index()

careerLen.columns = ["player_id", "career_length"]
finalDF = finalDF.merge(careerLen, on="player_id", how="left")
finalDF[["player","career_length"]][26703:26713]
```

[586]:
```
                  player  career_length
26703  Charlie Yelverton              1
26704       Rich Yonakor              1
26705          Gabe York              2
26706          Gabe York              2
26707   Chris Youngblood              1
26708        Danny Young             10
26709        Danny Young             10
26710        Danny Young             10
26711        Danny Young             10
26712        Danny Young             10
```

[587]:
```
#Create a copy of the DF that removes all the active players so we can gather␣
 ↪more accurate predictive conclusions about retirement.
retiredDF = finalDF.copy()
retiredDF = retiredDF[retiredDF['retired'].notna()]
retiredDF = retiredDF.reset_index(drop=True)
retiredDF
```

[587]:
```
       season   lg          player  player_id   age team  pos   g    gs  \
0        1991  NBA  Alaa Abdelnaby  abdelal01  22.0  POR   PF  43   0.0
1        1992  NBA  Alaa Abdelnaby  abdelal01  23.0  POR   PF  71   1.0
2        1993  NBA  Alaa Abdelnaby  abdelal01  24.0  2TM   PF  75  52.0
3        1994  NBA  Alaa Abdelnaby  abdelal01  25.0  BOS   PF  13   0.0
4        1995  NBA  Alaa Abdelnaby  abdelal01  26.0  2TM   PF  54   0.0
…         …   …              …         …    …   …  …  ..   …
24079    2019  NBA      Ante Žižić  zizican01  22.0  CLE    C  59  25.0
24080    2020  NBA      Ante Žižić  zizican01  23.0  CLE    C  22   0.0
24081    1983  NBA        Jim Zoet   zoetji01  29.0  DET    C   7   0.0
24082    1971  NBA       Bill Zopf   zopfbi01  22.0  MIL   PG  53   NaN
24083    1949  BAA      Matt Zunic  zunicma01  29.0  WSC  NaN  56   NaN

             mp  …  dws   ws  ws_48  obpm  dbpm  bpm  vorp  retired  \
0         290.0  …  0.5  0.5  0.079  -3.4  -1.2 -4.6  -0.2   1995.0
1         934.0  …  1.5  2.1  0.110  -2.3  -0.4 -2.6  -0.1   1995.0
2        1311.0  …  1.3  2.0  0.074  -2.4  -1.5 -3.9  -0.6   1995.0
3         159.0  …  0.1 -0.1 -0.032  -5.3  -2.2 -7.4  -0.2   1995.0
4         506.0  …  0.7  0.3  0.027  -4.4   0.1 -4.3  -0.3   1995.0
…          …   …  …  …    …   …   …  …   …      …
24079    1082.0  …  0.3  2.0  0.087  -1.1  -2.1 -3.2  -0.3   2020.0
24080     221.0  …  0.2  0.5  0.106  -1.7  -1.5 -3.2  -0.1   2020.0
24081      30.0  …  0.0 -0.1 -0.123  -5.6   0.2 -5.4  -0.1   1983.0
24082     398.0  …  0.4 -0.1 -0.011   NaN   NaN  NaN   NaN   1971.0
24083       NaN  …  1.8  2.0    NaN   NaN   NaN  NaN   NaN   1949.0

       mp_at_retirement  career_length
0                 506.0              5
1                 506.0              5
2                 506.0              5
3                 506.0              5
4                 506.0              5
…                   …              …
24079             221.0              3
24080             221.0              3
24081              30.0              1
24082             398.0              1
24083               NaN              1

[24084 rows x 156 columns]
```

[588]:
```
avgData = retiredDF.copy()
avgData["gs_g_ratio"] = avgData["gs"]/avgData["g"].replace(0, np.nan)

numeric_cols = [
    'career_length','mp','gs_g_ratio','fg','fga','fg_percent',
    'x3p','x3pa','x3p_percent',
```

```
    'x2p','x2pa','x2p_percent',
    'e_fg_percent','ft','fta','ft_percent'
]
non_numeric_cols = ['player', 'pos']
numeric_means = avgData.groupby('player_id')[numeric_cols].mean().reset_index()
non_numeric_first = avgData.groupby('player_id')[non_numeric_cols].first().
 ↪reset_index()
retirement_age = avgData.groupby('player_id')['age'].max().
 ↪reset_index(name='retirementAge')
avgData = numeric_means.merge(non_numeric_first, on='player_id')
avgData = avgData.merge(retirement_age, on='player_id')

# add heights and weights
avgData = avgData.merge(HeightAndWeightDf[['player', 'height', 'weight']],␣
 ↪on='player', how='left')
# add injuries
avgData = avgData.merge(injury_counts, on='player', how='left')

avgData
```

[588]:

	player_id	career_length	mp	gs_g_ratio	fg	\
0	abdelal01	5.0	640.000000	0.141484	124.000000	
1	abdulka01	20.0	2872.300000	1.000000	791.850000	
2	abdulma01	10.0	1920.200000	NaN	359.700000	
3	abdulma02	9.0	1736.444444	0.510361	390.444444	
4	abdulta01	6.0	801.166667	0.517169	120.000000	
…	…	…	…	…	…	
4881	zipsepa01	2.0	833.500000	0.315657	84.500000	
4882	zizican01	3.0	505.666667	0.162076	91.000000	
4883	zoetji01	1.0	30.000000	0.000000	1.000000	
4884	zopfbi01	1.0	398.000000	NaN	49.000000	
4885	zunicma01	1.0	NaN	NaN	98.000000	

	fga	fg_percent	x3p	x3pa	x3p_percent	…	\
0	247.200000	0.486400	0.000000	0.600000	0.000000	…	
1	1415.350000	0.558350	0.100000	1.800000	0.033300	…	
2	816.200000	0.432000	NaN	NaN	NaN	…	
3	882.555556	0.439778	52.666667	148.777778	0.316444	…	
4	287.666667	0.414833	3.000000	12.666667	0.254500	…	
…	…	…	…	…	…	…	
4881	227.500000	0.372000	35.000000	104.500000	0.334500	…	
4882	156.666667	0.617667	0.000000	0.000000	NaN	…	
4883	5.000000	0.200000	0.000000	0.000000	NaN	…	
4884	135.000000	0.363000	NaN	NaN	NaN	…	
4885	323.000000	0.303000	NaN	NaN	NaN	…	

	e_fg_percent	ft	fta	ft_percent	player	\

```
0         0.486400   45.000000   64.200000    0.658000       Alaa Abdelnaby
1         0.565700  335.600000  465.200000    0.727700  Kareem Abdul-Jabbar
2              NaN  189.300000  250.200000    0.749700         Walt Hazzard
3         0.466889  116.777778  129.000000    0.893111   Mahmoud Abdul-Rauf
4         0.421000   62.000000   88.166667    0.654833    Tariq Abdul-Wahad
…              …          …           …          …                    …
4881      0.449000   25.000000   32.500000    0.767500          Paul Zipser
4882      0.617667   42.666667   60.000000    0.722000           Ante Žižić
4883      0.200000    0.000000    0.000000         NaN            Jim Zoet
4884           NaN   20.000000   36.000000    0.556000           Bill Zopf
4885           NaN   77.000000  109.000000    0.706000           Matt Zunic

      pos  retirementAge height weight  num_injuries
0      PF           26.0  208.0  108.0           NaN
1       C           41.0    NaN    NaN           NaN
2      SG           31.0  201.0  102.0           NaN
3      PG           31.0  188.0   83.0           NaN
4      SG           28.0  198.0  101.0           NaN
…   …             …    …    …          …
4881   SF           23.0  203.0   97.0          12.0
4882    C           23.0    NaN    NaN           NaN
4883    C           29.0  216.0  108.0           NaN
4884   PG           22.0  185.0   77.0           NaN
4885  None          29.0    NaN    NaN           NaN

[4886 rows x 23 columns]
```

[589]:
```
predCols = ['fg_percent', 'x3p_percent', 'x2p_percent', 'e_fg_percent',␣
 ↪'ft_percent',
           'mp','gs_g_ratio','fg','fga','x3p','x3pa','x2p','x2pa','ft','fta',␣
 ↪'height', 'weight', 'num_injuries']

X = avgData[predCols].fillna(avgData[predCols].mean())
y = avgData['career_length'].fillna(avgData['career_length'].mean())
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[590]:
```
X_train
```

[590]:
```
      fg_percent  x3p_percent  x2p_percent  e_fg_percent  ft_percent  \
679     0.422500     0.346333     0.448500      0.473000    0.788333
4602    0.440200     0.224136     0.440029      0.440403    0.585600
1894    0.466000     0.176000     0.490000      0.473000    0.667000
3299    0.667000     0.224136     0.667000      0.667000    0.690108
1261    0.457750     0.370625     0.526375      0.538875    0.790000
…            …            …            …             …           …
4426    0.322600     0.224136     0.440029      0.440403    0.677600
```

```
466     0.444727     0.296000     0.459091      0.463545    0.870000
3092    0.275000     0.224136     0.440029      0.440403    0.667000
3772    0.230500     0.224136     0.440029      0.440403    0.726000
860     0.380000     0.100000     0.404500      0.383500    0.893000

                mp  gs_g_ratio          fg         fga        x3p         x3pa  \
679     750.833333    0.211742  121.666667  292.166667  26.500000   78.333333
4602   1030.600000    0.000000  200.200000  461.400000  13.886148   41.210164
1894    467.000000    0.216121  104.000000  223.000000   3.000000   17.000000
3299      6.000000    0.000000    2.000000    3.000000   0.000000    0.000000
1261   1002.750000    0.330943  113.375000  248.125000  39.625000  110.000000
…            …           …           …           …          …           …
4426    118.000000    0.216121  109.600000  347.800000  13.886148   41.210164
466    1958.636364    0.703244  355.272727  792.727273  35.818182  100.909091
3092    803.591551    0.216121   11.000000   40.000000  13.886148   41.210164
3772    803.591551    0.216121   27.000000   92.500000  13.886148   41.210164
860      79.000000    0.000000   19.000000   45.500000   0.500000    3.500000

              x2p         x2pa          ft         fta      height      weight  \
679     95.166667   213.833333   61.500000   76.166667  196.000000   93.000000
4602   107.157474   226.933226   99.800000  158.500000  211.000000   99.000000
1894   101.000000   206.000000   20.000000   30.000000  198.393443   94.397055
3299     2.000000     3.000000    0.000000    0.000000  201.000000  111.000000
1261    73.750000   138.125000   62.250000   78.125000  198.393443   94.397055
…            …           …           …           …          …           …
4426   107.157474   226.933226   69.800000  101.600000  190.000000   88.000000
466    319.454545   691.818182  162.181818  185.727273  180.000000   81.000000
3092   107.157474   226.933226   10.000000   15.000000  198.393443   94.397055
3772   107.157474   226.933226   13.750000   23.250000  178.000000   74.000000
860     18.500000    42.000000    6.500000    8.000000  198.000000   86.000000

      num_injuries
679      15.046458
4602     15.046458
1894     15.046458
3299     15.046458
1261     15.046458
…              …
4426     15.046458
466      15.046458
3092     15.046458
3772     15.046458
860      15.046458

[3908 rows x 18 columns]
```

[591]:
```
X_test
```

[591]:

	fg_percent	x3p_percent	x2p_percent	e_fg_percent	ft_percent
1149	0.437909	0.135714	0.438091	0.439545	0.694909
393	0.406000	0.139200	0.430900	0.430900	0.560750
1268	0.444000	0.224136	0.444000	0.444000	0.628250
4233	0.500000	0.000000	0.501500	0.500000	0.776000
4181	0.418100	0.284000	0.438100	0.439800	0.816500
…	…	…	…	…	…
3112	0.000000	0.224136	0.440029	0.440403	0.500000
84	0.467000	0.224136	0.440029	0.440403	0.851500
2086	0.370000	0.261000	0.407000	0.403000	0.858000
4126	0.250000	0.000000	0.500000	0.250000	0.500000
3614	0.404000	0.334000	0.445000	0.467250	0.798000

	mp	gs_g_ratio	fg	fga	x3p	x3pa
1149	1587.454545	0.697676	292.363636	624.545455	0.454545	2.181818
393	1169.800000	0.216285	66.500000	163.500000	4.300000	20.500000
1268	534.500000	0.305125	39.500000	85.250000	0.000000	0.000000
4233	409.000000	0.056818	35.000000	85.000000	0.000000	0.500000
4181	1789.000000	0.432840	290.400000	687.000000	27.600000	92.100000
…	…	…	…	…	…	…
3112	11.000000	0.216121	0.000000	4.000000	13.886148	41.210164
84	533.500000	0.216121	67.500000	143.000000	13.886148	41.210164
2086	2045.500000	0.216121	329.500000	879.500000	61.500000	223.500000
4126	12.000000	0.000000	1.000000	4.000000	0.000000	2.000000
3614	587.500000	0.038377	78.750000	190.000000	24.250000	71.250000

	x2p	x2pa	ft	fta	height	weight
1149	291.909091	622.363636	149.818182	203.545455	213.0	124.0
393	62.200000	143.000000	22.600000	40.300000	231.0	90.0
1268	39.500000	85.250000	26.250000	42.250000	211.0	115.0
4233	35.000000	84.500000	19.500000	28.000000	193.0	79.0
4181	262.800000	594.900000	210.100000	255.100000	196.0	95.0
…	…	…	…	…	…	
3112	107.157474	226.933226	1.000000	2.000000	190.0	83.0
84	107.157474	226.933226	38.500000	45.500000	208.0	104.0
2086	268.000000	656.000000	374.500000	440.000000	193.0	90.0
4126	1.000000	2.000000	1.000000	2.000000	201.0	102.0
3614	54.500000	118.750000	31.000000	38.000000	185.0	88.0

	num_injuries
1149	15.046458
393	15.046458
1268	15.046458
4233	15.046458
4181	43.000000
…	…
3112	15.046458

```
84       15.046458
2086     15.046458
4126     15.046458
3614     15.046458

[978 rows x 18 columns]
```

[592]:
```
y_train
```

[592]:
```
679      6.0
4602    10.0
1894     1.0
3299     1.0
1261     8.0
        …
4426     5.0
466     11.0
3092     1.0
3772     4.0
860      2.0
Name: career_length, Length: 3908, dtype: float64
```

[593]:
```
y_test
```

[593]:
```
1149    11.0
393     10.0
1268     4.0
4233     2.0
4181    10.0
        …
3112     1.0
84       2.0
2086     2.0
4126     1.0
3614     4.0
Name: career_length, Length: 978, dtype: float64
```

[594]:
```
avgCareerLength = avgData['career_length'].mean()
print(avgCareerLength)

avgRetirementAge = avgData['retirementAge'].mean()
print(avgRetirementAge)
```

```
4.929390094146541
27.281930184804928
```

[595]:
```
avgData.loc[avgData['player'] == 'LeBron James']
```

[595]:
```
Empty DataFrame
Columns: [player_id, career_length, mp, gs_g_ratio, fg, fga, fg_percent, x3p,
x3pa, x3p_percent, x2p, x2pa, x2p_percent, e_fg_percent, ft, fta, ft_percent,
player, pos, retirementAge, height, weight, num_injuries]
Index: []

[0 rows x 23 columns]
```

13 Predictions

[596]:
```
#Function to return predicted retirement values
def calculate_y_hat(X_train, X_test, Y_train, with_intercept=True):
    X = np.array(X_train)
    Y = np.array(Y_train)
    reg = LinearRegression(fit_intercept=with_intercept)
    reg.fit(X_train, Y_train)
    #Determining the yHat value given the fit and linear regression
    yHat = reg.predict(X_test)
    return yHat

# Define the columns to use for prediction
#predCols = ['fg_percent', 'x3p_percent', 'x2p_percent', 'e_fg_percent',␣
 ↪'ft_percent',
            #'mp','gs_g_ratio','fg','fga','x3p','x3pa','x2p','x2pa','ft','fta',␣
 ↪'height', 'weight', 'num_injuries']

# Fill missing values with column means
#X_train_features = trainData[predCols].fillna(trainData[predCols].mean())
#X_test_features = testData[predCols].fillna(testData[predCols].mean())
#y_train = trainData['career_length'].fillna(trainData['career_length'].mean())
#y_test = testData['career_length'].fillna(testData['career_length'].mean())

# Use the function to get predicted retirement ages
y_pred = calculate_y_hat(X_train, X_test, y_train, with_intercept=True)

# Round predictions for readability
#avgData['predictedCareerLength'] = avgData['predictedCareerLength'].round(0)

# Display results
print(y_test.head(20))
print(y_pred[:20])
```

```
1149    11.0
393     10.0
1268     4.0
4233     2.0
4181    10.0
```

```
2680    14.0
3949    16.0
4854     2.0
3757     1.0
3979     6.0
3418     4.0
2509     2.0
1371     1.0
3478     2.0
4748     7.0
2577    16.0
538      7.0
718      1.0
4502    16.0
4473     2.0
Name: career_length, dtype: float64
[ 9.48810784  6.83087212  4.90436618  3.10126108  9.79916924 10.05950646
  9.19021371  0.57583574  3.04748741  8.31209126  4.27512803  4.04146995
  0.59106491  1.79122387  4.79168957 14.49667684  8.37077751  2.63653874
  9.89709037  0.55857179]
```

[597]:
```
df = pd.DataFrame()
df['career_length'] = y_test
df['predicted_career_length'] = y_pred
df.head(20)
```

[597]:

	career_length	predicted_career_length
1149	11.0	9.488108
393	10.0	6.830872
1268	4.0	4.904366
4233	2.0	3.101261
4181	10.0	9.799169
2680	14.0	10.059506
3949	16.0	9.190214
4854	2.0	0.575836
3757	1.0	3.047487
3979	6.0	8.312091
3418	4.0	4.275128
2509	2.0	4.041470
1371	1.0	0.591065
3478	2.0	1.791224
4748	7.0	4.791690
2577	16.0	14.496677
538	7.0	8.370778
718	1.0	2.636539
4502	16.0	9.897090
4473	2.0	0.558572

[598]:
```
#y = testData['career_length']
#y_hat = testData['predictedCareerLength']

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse:.2f}")
print(f"R2: {r2:.4f}")
```

```
RMSE: 2.57
R2: 0.6699
```

[599]:
```
print(avgData.loc[avgData['player'] == 'Trae Young'])
```

```
Empty DataFrame
Columns: [player_id, career_length, mp, gs_g_ratio, fg, fga, fg_percent, x3p,
x3pa, x3p_percent, x2p, x2pa, x2p_percent, e_fg_percent, ft, fta, ft_percent,
player, pos, retirementAge, height, weight, num_injuries]
Index: []

[0 rows x 23 columns]
```

[600]:
```
#PLot of true retirement age vs predicted retirement age
plotData = avgData[avgData['career_length'] > 0]

x = plotData['career_length'].values.reshape(-1, 1)
y = plotData['predictedCareerLength'].values
reg = LinearRegression()
reg.fit(x, y)
y_fit = reg.predict(x)

plt.figure(figsize=(8,6))
plt.scatter(x, y, alpha=0.6, color='blue', label='Data points')
plt.plot(x, y_fit, color='green')
plt.xlabel("True career length")
plt.ylabel("Predicted career length")
plt.title("Predicted vs true career length")
plt.show()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~/CS577/577venv1/lib/python3.12/site-packages/pandas/core/indexes/base.py:
↪3812, in Index.get_loc(self, key)
   3811 try:
-> 3812     return self._engine.get_loc(casted_key)
   3813 except KeyError as err:
```

```
File pandas/_libs/index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File pandas/_libs/index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas/_libs/hashtable_class_helper.pxi:7088, in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

File pandas/_libs/hashtable_class_helper.pxi:7096, in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

KeyError: 'predictedCareerLength'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[600], line 5
      2 plotData = avgData[avgData['career_length'] > 0]
      4 x = plotData['career_length'].values.reshape(-1, 1)
----> 5 y = plotData[                          ].values
      6 reg = LinearRegression()
      7 reg.fit(x, y)

File ~/CS577/577venv1/lib/python3.12/site-packages/pandas/core/frame.py:4113, i
 ↪DataFrame.__getitem__(self, key)
   4111 if self.columns.nlevels > 1:
   4112     return self._getitem_multilevel(key)
-> 4113 indexer = self.columns.get_loc(key)
   4114 if is_integer(indexer):
   4115     indexer = [indexer]

File ~/CS577/577venv1/lib/python3.12/site-packages/pandas/core/indexes/base.py:
 ↪3819, in Index.get_loc(self, key)
   3814     if isinstance(casted_key, slice) or (
   3815         isinstance(casted_key, abc.Iterable)
   3816         and any(isinstance(x, slice) for x in casted_key)
   3817     ):
   3818         raise InvalidIndexError(key)
-> 3819     raise KeyError(key) from err
   3820 except TypeError:
   3821     # If we have a listlike key, _check_indexing_error will raise
   3822     #  InvalidIndexError. Otherwise we fall through and re-raise
   3823     #  the TypeError.
   3824     self._check_indexing_error(key)

KeyError: 'predictedCareerLength'
```

[ ]:
```
finalDF.plot.scatter(x='mp', y='pts')
```

[ ]:
```
finalDF = finalDF.dropna()
# Linear regression with mp and pts
MP = np.array(finalDF['mp']).reshape(-1, 1)
PTS = np.array(finalDF['pts']).reshape(-1, 1)
model = LinearRegression()
model.fit(MP, PTS)
prediction = model.predict(MP)
```

[ ]:
```
plt.scatter(finalDF['mp'], finalDF['pts'])
line = prediction
plt.plot(finalDF['mp'], line, color='red')
plt.xlabel('mp')
plt.ylabel('pts')
plt.title('Simple Linear Regression, predicting mp and pts')
plt.show()
```

[ ]:
```
#More plotting
finalDF["age"] = pd.to_numeric(finalDF["age"])
ageDF = finalDF[(finalDF["age"])> 0]
aveAge = ageDF.groupby("age")["mp_per_game"].mean()
youngest = aveAge.index.min()

plt.plot(aveAge.index, aveAge.values)
plt.title("average mins per game")
plt.xlabel("age")
plt.ylabel("mins per game")
plt.xlim(youngest)
plt.show()
```

[ ]:
```
plt.plot(finalDF['career_length'])
```

14 AMELIA CODE

[ ]:
```
# Install dependencies as needed:
# pip install kagglehub[pandas-datasets]
import kagglehub
from kagglehub import KaggleDatasetAdapter

# Set the path to the file you'd like to load
file_path = "Player Totals.csv"

# Load the latest version
totals_df = kagglehub.load_dataset(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
```

```
  file_path,
  # Provide any additional arguments like
  # sql_query or pandas_kwargs. See the
  # documenation for more information:
  # https://github.com/Kaggle/kagglehub/blob/main/README.
 ↪md#kaggledatasetadapterpandas
)

print("First 5 records:", totals_df.head())

# Import kagglehub if not already imported
import kagglehub
from kagglehub import KaggleDatasetAdapter

# Path to the new file
file_path_shooting = "Player Shooting.csv"

# Load the Player Shooting.csv file
df_shooting = kagglehub.load_dataset(
    KaggleDatasetAdapter.PANDAS,
    "sumitrodatta/nba-aba-baa-stats",
    file_path_shooting,
    # You can also provide additional pandas kwargs like index_col,␣
 ↪parse_dates, etc.
)

print("First 5 records of Player Shooting:", df_shooting.head())


# Install dependencies as needed:
# pip install kagglehub[pandas-datasets]
import kagglehub
from kagglehub import KaggleDatasetAdapter

# Set the path to the file you'd like to load
file_path = "Player Season Info.csv"

# Load the latest version
season_info_df = kagglehub.load_dataset(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
  # Provide any additional arguments like
  # sql_query or pandas_kwargs. See the
  # documenation for more information:
  # https://github.com/Kaggle/kagglehub/blob/main/README.
 ↪md#kaggledatasetadapterpandas
```

```
)

print("First 5 records:", season_info_df.head())


# Install dependencies as needed:
# pip install kagglehub[pandas-datasets]
import kagglehub
from kagglehub import KaggleDatasetAdapter

# Set the path to the file you'd like to load
file_path = "Player Award Shares.csv"

# Load the latest version
award_shares_df = kagglehub.load_dataset(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
  # Provide any additional arguments like
  # sql_query or pandas_kwargs. See the
  # documenation for more information:
  # https://github.com/Kaggle/kagglehub/blob/main/README.
  ↪md#kaggledatasetadapterpandas
)

print("First 5 records:", award_shares_df.head())


# Install dependencies as needed:
# pip install kagglehub[pandas-datasets]
import kagglehub
from kagglehub import KaggleDatasetAdapter

# Set the path to the file you'd like to load
file_path = "Player Career Info.csv"

# Load the latest version
career_df = kagglehub.load_dataset(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
  # Provide any additional arguments like
  # sql_query or pandas_kwargs. See the
  # documenation for more information:
  # https://github.com/Kaggle/kagglehub/blob/main/README.
  ↪md#kaggledatasetadapterpandas
)
```

```
print("First 5 records:", career_df.head())



# Install dependencies as needed:
# pip install kagglehub[pandas-datasets]
import kagglehub
from kagglehub import KaggleDatasetAdapter

# Set the path to the file you'd like to load
file_path = "Player Per Game.csv"

# Load the latest version
per_games_df = kagglehub.load_dataset(
  KaggleDatasetAdapter.PANDAS,
  "sumitrodatta/nba-aba-baa-stats",
  file_path,
  # Provide any additional arguments like
  # sql_query or pandas_kwargs. See the
  # documenation for more information:
  # https://github.com/Kaggle/kagglehub/blob/main/README.
  ↪md#kaggledatasetadapterpandas
)

print("First 5 records:", per_games_df.head())
```

[ ]:
```
mergedDF = totals_df
mergedDF = pd.merge(mergedDF,per_games_df,on=["player_id", "season"],␣
  ↪how="outer",suffixes=("", "_per_game"))
mergedDF = pd.merge(mergedDF,df_shooting,on=["player_id", "season"],␣
  ↪how="outer",suffixes=("", "_shooting"))
mergedDF = pd.merge(mergedDF,season_info_df,on=["player_id", "season"],␣
  ↪how="outer",suffixes=("", "_season_info"))
mergedDF = pd.merge(mergedDF,award_shares_df,on=["player_id", "season"],␣
  ↪how="outer",suffixes=("", "_award"))
finalDF = pd.merge(mergedDF,career_df,on="player_id", how="left",suffixes=("",␣
  ↪"_career"))
print(finalDF[223000:223030])
```