nba

November 18, 2024

1 Imports

[1]:
```
# import kagglehub
import pandas as pd
import numpy as np
import os
import zipfile
```

[2]:
```
import pandas as pd
import plotly.express as px
import ipywidgets as widgets
from IPython.display import display
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

2 Loading CSV from Kaggle API

[3]:
```
# # Download latest version
# path = kagglehub.dataset_download("justinas/nba-players-data")
# print("Path to dataset files:", path)
```

[8]:
```
# # List all files in the directory to find the specific file
# files = os.listdir(path) #--> ['file name']
# print("Files in the dataset directory:", files)

# # Assuming you want to read a CSV file, find the relevant file and load it
# csv_file_path = os.path.join(path, files[0])  # Replace with the actual␣
 ↪filename
df = pd.read_csv('all_seasons.csv')
# df = pd.read_csv(csv_file_path)
df.head()
%config InlineBackend.print_rows = 5
```

[9]:
```
b = False
for i in df.index:
    if (i != df.iloc[i, 0]):
```

```
        print(i)
        break
    b = True
if b==True:
    df.drop(df.columns[0], axis = 1, inplace = True)
```

[10]:
```
df['college'].fillna('Untraditional', inplace = True)
```

[11]:
```
nba_top10 = df.copy()
#storing highest colleges having largest number of draftees
arr_10 = nba_top10.groupby('college').agg({'age':'count'}).sort_values('age',␣
 ↪ascending = False).head(25).index.to_numpy()
```

[12]:
```
nba_top10 = nba_top10[nba_top10['college'].apply(lambda x: x in arr_10)]
```

[13]:
```
nba_top10.columns[11: 21]
```

[13]:
```
Index(['pts', 'reb', 'ast', 'net_rating', 'oreb_pct', 'dreb_pct', 'usg_pct',
       'ts_pct', 'ast_pct', 'season'],
      dtype='object')
```

[14]:
```
category_dropdown = widgets.Dropdown(
    options=nba_top10['college'].unique(),
    description='College:',
    value='UCLA'
)
#columns containing stats located from index 11 --> 21 (options for stats in␣
 ↪our dropdown)
stats_dropdown = widgets.Dropdown(
    options=nba_top10.columns[11: 21],
    description='Stat:',
    value='pts'
)
stats_dropdown #example
```

[14]:
```
Dropdown(description='Stat:', options=('pts', 'reb', 'ast', 'net_rating',
'oreb_pct', 'dreb_pct', 'usg_pct', '…
```

[15]:
```
#confirm approach to round any year down to a divisor of 5
for i in nba_top10['season'].value_counts().index:
    print((int(i.split('-')[0])//5)*5)
```

```
2015
2015
2015
2010
2015
```

```
2010
2010
2020
2020
2015
2020
2010
2010
2005
2005
2005
2005
2005
2000
2000
2000
2000
2000
2000
1995
1995
1995
1995
```

[16]:
```
#roudning player's year down into 5 year buckets
def fiveYear(szn):
    return (int(szn.split('-')[0])//5)*5
```

[17]:
```
nba_top10['5Year'] = nba_top10['season'].apply(lambda x: fiveYear(x))
```

[18]:
```
nba_top10.columns
```

[18]:
```
Index(['player_name', 'team_abbreviation', 'age', 'player_height',
       'player_weight', 'college', 'country', 'draft_year', 'draft_round',
       'draft_number', 'gp', 'pts', 'reb', 'ast', 'net_rating', 'oreb_pct',
       'dreb_pct', 'usg_pct', 'ts_pct', 'ast_pct', 'season', '5Year'],
      dtype='object')
```

[19]:
```
nba_top10[nba_top10['college'] == 'Kansas'].groupby('5Year',␣
 ↪as_index=False)['pts'].mean()
```

[19]:

	5Year	pts
0	1995	6.634615
1	2000	9.665714
2	2005	7.887500
3	2010	7.119118
4	2015	8.847297
5	2020	10.357895

3 Create column that has years in league during a respective season

[20]:
```
#type of column is unorthodox and must be processed unconvetnionally as:
# nba_top10['season'].str.split('-')[0] refuses to extract first column of the␣
 ↪matrix
#shape of matrix is currently [m, none]?
j = 0
for i in nba_top10['season'].str.split('-'):

    print(i[0])
    j+=1
    if j==10:
        break

# above currectly extracts first year in the season
arr = []
split = nba_top10['season'].str.split('-')
for i in split:
    arr.append(int(i[0]))

np.array(arr).shape
```

```
1996
1996
1996
1996
1996
1996
1996
1996
1996
1996
```

[20]:
```
(6751,)
```

[21]:
```
len(arr) == len(nba_top10['draft_year'])
```

[21]:
```
True
```

[22]:
```
# those who were undracted or had unconventional route to the nba (i.e -->␣
 ↪notable names Luka Doncic, Wemby must be undrafted_
# and thus have NA values under draft_year --> replace these null val and␣
 ↪represent them as 0
nba_top10['draft_year'].value_counts().head(2)
```

[22]:
```
draft_year
Undrafted    750
2003         303
Name: count, dtype: int64
```

[23]:
```
nba_top10['draft_year'].replace('Undrafted', None, inplace=True)
nba_top10['draft_year'].fillna(0, inplace = True)
nba_top10['start_season'] = arr
#undrafted players can now be filtered numerically:
nba_top10[nba_top10['draft_year'] == 0]
```

[23]:

	player_name	team_abbreviation	age	player_height	player_weight	\
88	Jaren Jackson	WAS	29.0	198.12	95.254320	
165	Duane Ferrell	IND	32.0	200.66	97.522280	
176	Darrick Martin	LAC	26.0	180.34	77.110640	
197	Keith Askins	MIA	29.0	203.20	101.604608	
199	Scott Williams	PHI	29.0	208.28	104.326160	
…	…	…	…	…	…	
12795	Jack White	DEN	25.0	200.66	102.058200	
12809	Jose Alvarado	NOP	25.0	182.88	81.192968	
12818	Johnny Juzang	UTA	22.0	198.12	94.800728	
12837	Joe Ingles	MIL	35.0	205.74	99.790240	
12843	JaMychal Green	GSW	33.0	205.74	102.965384	

	college	country	draft_year	draft_round	draft_number	…	\
88	Georgetown	USA	0	Undrafted	Undrafted	…	
165	Georgia Tech	USA	0	Undrafted	Undrafted	…	
176	UCLA	USA	0	Undrafted	Undrafted	…	
197	Alabama	USA	0	Undrafted	Undrafted	…	
199	North Carolina	USA	0	Undrafted	Undrafted	…	
…	…	…	…	…	…	…	
12795	Duke	Australia	0	Undrafted	Undrafted	…	
12809	Georgia Tech	USA	0	Undrafted	Undrafted	…	
12818	UCLA	USA	0	Undrafted	Undrafted	…	
12837	Untraditional	Australia	0	Undrafted	Undrafted	…	
12843	Alabama	USA	0	Undrafted	Undrafted	…	

	ast	net_rating	oreb_pct	dreb_pct	usg_pct	ts_pct	ast_pct	season	\
88	0.9	1.7	0.030	0.104	0.167	0.520	0.090	1996-97	
165	1.1	2.3	0.066	0.091	0.181	0.521	0.100	1996-97	
176	4.1	-4.5	0.016	0.060	0.232	0.539	0.292	1996-97	
197	1.0	6.1	0.059	0.127	0.108	0.557	0.068	1996-97	
199	0.7	-4.6	0.126	0.205	0.127	0.529	0.049	1996-97	
…	…	…	…	…	…	…	…	…	
12795	0.2	-14.1	0.086	0.177	0.133	0.517	0.093	2022-23	
12809	3.0	3.7	0.022	0.087	0.191	0.525	0.199	2022-23	
12818	0.4	-17.7	0.029	0.122	0.187	0.411	0.051	2022-23	

```
12837  3.3         2.5     0.012     0.102     0.122   0.616     0.181  2022-23
12843  0.9        -8.2     0.087     0.164     0.169   0.650     0.094  2022-23

       5Year  start_season
88      1995          1996
165     1995          1996
176     1995          1996
197     1995          1996
199     1995          1996
…       …             …
12795   2020          2022
12809   2020          2022
12818   2020          2022
12837   2020          2022
12843   2020          2022

[750 rows x 23 columns]
```

[24]:
```
nba_top10.isnull().sum() == 0 #confirm no outstanding null vals
```

[24]:
```
player_name          True
team_abbreviation    True
age                  True
player_height        True
player_weight        True
college              True
country              True
draft_year           True
draft_round          True
draft_number         True
gp                   True
pts                  True
reb                  True
ast                  True
net_rating           True
oreb_pct             True
dreb_pct             True
usg_pct              True
ts_pct               True
ast_pct              True
season               True
5Year                True
start_season         True
dtype: bool
```

[25]:
```
nba_top10.columns
```

[25]:
```
Index(['player_name', 'team_abbreviation', 'age', 'player_height',
       'player_weight', 'college', 'country', 'draft_year', 'draft_round',
       'draft_number', 'gp', 'pts', 'reb', 'ast', 'net_rating', 'oreb_pct',
       'dreb_pct', 'usg_pct', 'ts_pct', 'ast_pct', 'season', '5Year',
       'start_season'],
      dtype='object')
```

[26]:
```
#Clean draft years for undrafted players, this confirms non-digit values are␣
 ↪uniquely only undrafted, locate and replace undrafted
#groups by each undrafted player and pulls their minimum start season to find␣
 ↪the year they played first nba game
nba_top10[nba_top10['draft_year'].astype(str).str.isdigit() ==␣
 ↪False]['draft_year'].value_counts()
mapped = nba_top10[nba_top10['draft_year']==0].groupby('player_name').
 ↪agg(min)['start_season']
print(f'{mapped.shape[0]} unique undrafted players in top 10 schools')
```

```
258 unique undrafted players in top 10 schools
```

[27]:
```
nba_top10[nba_top10['draft_year'] == 0]
```

[27]:

	player_name	team_abbreviation	age	player_height	player_weight	…
88	Jaren Jackson	WAS	29.0	198.12	95.254320	
165	Duane Ferrell	IND	32.0	200.66	97.522280	
176	Darrick Martin	LAC	26.0	180.34	77.110640	
197	Keith Askins	MIA	29.0	203.20	101.604608	
199	Scott Williams	PHI	29.0	208.28	104.326160	
…	…	…	…	…	…	
12795	Jack White	DEN	25.0	200.66	102.058200	
12809	Jose Alvarado	NOP	25.0	182.88	81.192968	
12818	Johnny Juzang	UTA	22.0	198.12	94.800728	
12837	Joe Ingles	MIL	35.0	205.74	99.790240	
12843	JaMychal Green	GSW	33.0	205.74	102.965384	

	college	country	draft_year	draft_round	draft_number	…
88	Georgetown	USA	0	Undrafted	Undrafted	…
165	Georgia Tech	USA	0	Undrafted	Undrafted	…
176	UCLA	USA	0	Undrafted	Undrafted	…
197	Alabama	USA	0	Undrafted	Undrafted	…
199	North Carolina	USA	0	Undrafted	Undrafted	…
…	…	…	…	…	…	…
12795	Duke	Australia	0	Undrafted	Undrafted	…
12809	Georgia Tech	USA	0	Undrafted	Undrafted	…
12818	UCLA	USA	0	Undrafted	Undrafted	…
12837	Untraditional	Australia	0	Undrafted	Undrafted	…
12843	Alabama	USA	0	Undrafted	Undrafted	…

```
       ast  net_rating  oreb_pct  dreb_pct  usg_pct  ts_pct  ast_pct    season  \
88     0.9         1.7     0.030     0.104    0.167   0.520    0.090   1996-97
165    1.1         2.3     0.066     0.091    0.181   0.521    0.100   1996-97
176    4.1        -4.5     0.016     0.060    0.232   0.539    0.292   1996-97
197    1.0         6.1     0.059     0.127    0.108   0.557    0.068   1996-97
199    0.7        -4.6     0.126     0.205    0.127   0.529    0.049   1996-97
…      …          …        …         …        …       …        …
12795  0.2       -14.1     0.086     0.177    0.133   0.517    0.093   2022-23
12809  3.0         3.7     0.022     0.087    0.191   0.525    0.199   2022-23
12818  0.4       -17.7     0.029     0.122    0.187   0.411    0.051   2022-23
12837  3.3         2.5     0.012     0.102    0.122   0.616    0.181   2022-23
12843  0.9        -8.2     0.087     0.164    0.169   0.650    0.094   2022-23

       5Year  start_season
88      1995          1996
165     1995          1996
176     1995          1996
197     1995          1996
199     1995          1996
…       …             …
12795   2020          2022
12809   2020          2022
12818   2020          2022
12837   2020          2022
12843   2020          2022

[750 rows x 23 columns]
```

[28]:
```
#use series of uniwue undrafted players and their start season to map onto␣
 ↪reald df and update their start season
#maps based on our index which in ther nace --> access player names and map␣
 ↪year values onto draft_year of undrafted
nba_top10.loc[nba_top10['draft_year'] == 0, 'draft_year'] =␣
 ↪nba_top10['player_name'].map(mapped)
#convert in case floats are present
nba_top10['draft_year'] = nba_top10['draft_year'].astype(int)

# nba_top10[(nba_top10['draft_round'] == 'Undrafted')] #confirms all have been␣
 ↪now mapped

nba_top10['draft_year'].value_counts()
#check if mapped correctly (i.e ensuring no non-year values in col

# nba_top10[nba_top10['draft_year'].isna()].apply(lambda
```

[28]:
```
draft_year
2005    375
2003    332
2008    311
1996    307
2001    307
2004    284
1998    265
2006    262
2014    255
2009    253
2012    231
2002    230
2011    225
2013    224
2000    217
2007    210
2010    202
2017    189
2015    189
2016    188
1995    187
1999    184
1997    173
2018    172
2019    132
1992    122
1994    111
2020    106
1993     91
2021     69
1991     65
1989     63
1990     56
2022     40
1988     37
1987     33
1984     19
1986     16
1985     12
1981      5
1983      1
1982      1
Name: count, dtype: int64
```

[29]:
```
nba_top10['yearInLeague'] = nba_top10['start_season'] - nba_top10['draft_year']
```

[30]:
```
#rookie seasons
def rookieSeason(name):
    return nba_top10[(nba_top10['yearsLeague'] == 0) &␣
 ↪(nba_top10['player_name'].str.contains(name))]
rookies = nba_top10[nba_top10['yearInLeague'] == 0]
```

[31]:
```
best_player = rookies[nba_top10['college'] == 'UCLA'].sort_values('pts',␣
 ↪ascending = False).reset_index()
best_year = best_player.iloc[0, list(rookies.columns).index('net_rating')+1].
 ↪astype(int)
print(best_year)
```

```
-5
```

```
/tmp/ipykernel_1032/1741623170.py:1: UserWarning: Boolean Series key will be
reindexed to match DataFrame index.
  best_player = rookies[nba_top10['college'] == 'UCLA'].sort_values('pts',
ascending = False).reset_index()
```

[32]:
```
from plotly.subplots import make_subplots
import plotly.graph_objects as go

def outputPrint(category, stat):
    filtered = rookies[nba_top10['college'] == category].groupby('5Year',␣
 ↪as_index=False)[stat].mean()
    timeY = rookies[rookies['college'] == category].groupby('start_season',␣
 ↪as_index=False)[stat].mean()
    avg_stat = rookies[rookies['college'] == category][stat].mean()
    figure1 = px.bar(filtered, x = '5Year', y = stat, title=f'{category} School␣
 ↪Average Rookie Points')
    figure2 = px.line(timeY, x = 'start_season', y = stat)
    #average stat for the selected school
    figure2.add_hline(y = avg_stat, line_dash = 'dot', annotation_text=␣
 ↪category + ' average ' + stat + ': ' + str(np.round(avg_stat, 2)),
              annotation_position="bottom right")
    #best player in the selected dropdowns
    best_player = rookies[nba_top10['college'] == category].sort_values(stat,␣
 ↪ascending = False).reset_index()
    name = best_player.iloc[0, 1]
    best_stat = best_player.iloc[0, list(rookies.columns).index(stat)+1].
 ↪astype(int)
    name = name + ' best ' + stat +' :' +  str(best_stat)
    best_year = best_player.iloc[0, list(rookies.columns).
 ↪index('draft_year')+1].astype(int)
    # print(best_player.shape)
    figure2.add_hline(y = avg_stat, line_dash = 'dot', annotation_text=␣
 ↪category + ' average ' + stat + ': ' + str(np.round(avg_stat, 2)),
```

```
            annotation_position="bottom right")

    figure2.add_vline(x=best_year, line_dash = 'dot', line_color = 'red',␣
 ↪annotation_text = name, annotation_position = "top right")


    # print(f'School Being Described: {category}')
    figure1.show()
    figure2.update_layout(xaxis=dict(range=[1995, 2025]))
    figure2.show()
```

[33]:
```
import warnings
warnings.filterwarnings("ignore")

widgets.interactive(outputPrint, category=category_dropdown, stat =␣
 ↪stats_dropdown)
```

[33]:
```
interactive(children=(Dropdown(description='College:', index=2,
options=('Louisiana State', 'North Carolina', …
```

4 US College Athletes vs Nontraditional Athletes drafted into NBA Trends

Nontraditional defined as players direct from high school, overseas, G-League, etc.

[34]:
```
df[df['player_name'] == 'Luka Doncic']
```

[34]:

	player_name	team_abbreviation	age	player_height	player_weight	\
10605	Luka Doncic	DAL	20.0	200.66	98.883056	
11092	Luka Doncic	DAL	21.0	200.66	104.326160	
11670	Luka Doncic	DAL	22.0	200.66	104.326160	
12099	Luka Doncic	DAL	23.0	200.66	104.326160	
12740	Luka Doncic	DAL	24.0	200.66	104.326160	

	college	country	draft_year	draft_round	draft_number	…	pts	\
10605	Untraditional	Slovenia	2018	1	3	…	21.2	
11092	Untraditional	Slovenia	2018	1	3	…	28.8	
11670	Untraditional	Slovenia	2018	1	3	…	27.7	
12099	Untraditional	Slovenia	2018	1	3	…	28.4	
12740	Untraditional	Slovenia	2018	1	3	…	32.4	

	reb	ast	net_rating	oreb_pct	dreb_pct	usg_pct	ts_pct	ast_pct	\
10605	7.8	6.0	-3.1	0.036	0.198	0.296	0.545	0.317	
11092	9.4	8.8	5.3	0.036	0.224	0.355	0.585	0.454	
11670	8.0	8.6	3.9	0.024	0.202	0.350	0.587	0.425	
12099	9.1	8.7	3.5	0.025	0.230	0.368	0.571	0.458	
12740	8.6	8.0	2.1	0.024	0.224	0.368	0.609	0.408	

```
        season
10605  2018-19
11092  2019-20
11670  2020-21
12099  2021-22
12740  2022-23

[5 rows x 21 columns]
```

[35]:
```
df['draft_year'].replace('Undrafted', 0, inplace=True)
lol = df[df['college'].astype(str) == 'Untraditional'].groupby('player_name').
 ↪agg({'pts':'mean'}).sort_values('pts', ascending = False)
lol.shape
```

[35]:
```
(361, 1)
```

[36]:
```
lol.head()
```

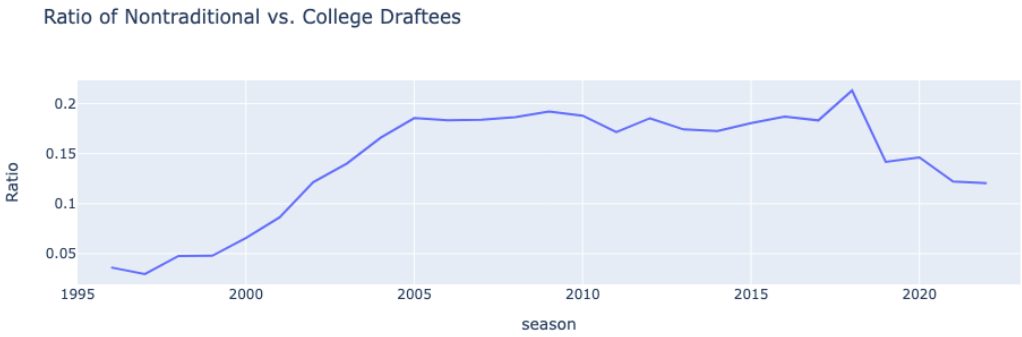
[36]:
```
                          pts
player_name
Luka Doncic             27.70
LeBron James            27.20
Kobe Bryant             24.20
Giannis Antetokounmpo   23.25
Nikola Jokic            20.40
```

[37]:
```
#ratio of overseas players over total players per year currently active in nba␣
 ↪over time
ratio = []

trend = df.groupby('season').apply(lambda x: x[x['college'] == 'Untraditional'].
 ↪shape[0]/x.shape[0]).reset_index()

arr = []
split = trend['season'].str.split('-')
for i in split:
    arr.append(int(i[0]))
trend['season'] = arr
# trend['season'].apply(lambda x: fiveYear(x))

fig = px.line(data_frame=trend, x='season', y=0, title="Ratio of Nontraditional␣
 ↪vs. College Draftees")
fig.update_layout(xaxis=dict(range=[1995, 2023]), yaxis_title = 'Ratio')
fig.show()
```

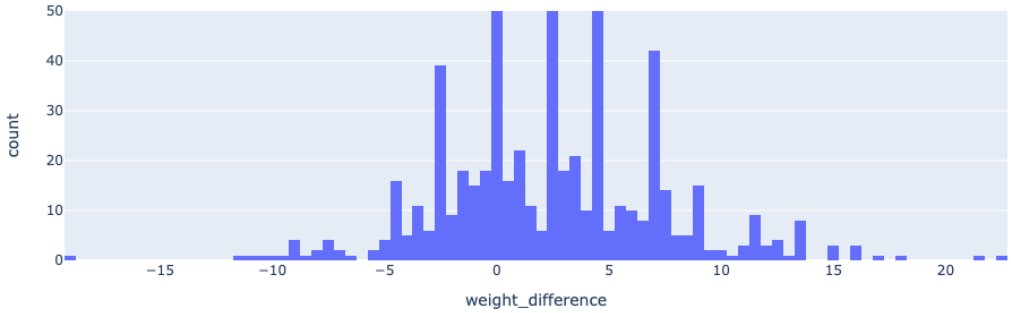


5 Average Change in weight over Career of nba players

[38]:
```
#gather the weight at start of career and end of career and compute differesce,␣
 ↪need years in league col
arr = []
split = df['season'].str.split('-')
for i in split:
    arr.append(int(i[0]))
df['start'] = np.array(arr).astype(int)
df['yearsInLeague'] = arr - np.array(df['draft_year']).astype(int)
```

[39]:
```
diff = df.groupby('player_name').apply(lambda x: x.loc[x['yearsInLeague'] ==␣
 ↪x['yearsInLeague'].max(), 'player_weight'].values[0] -
       x.loc[x['yearsInLeague'] == 0, 'player_weight'].values[0]
       if not x.loc[x['yearsInLeague'] == 0, 'player_weight'].empty and not x.
 ↪loc[x['yearsInLeague'] == x['yearsInLeague'].max(), 'player_weight'].empty
       else 0  # Default value if either condition fails
    ).reset_index(name='weight_difference')
# diff.sort_values(, ascending = False)
diff.sort_values('weight_difference', ascending = False)

fig = px.histogram(data_frame=diff, x='weight_difference'
       )
fig.update_layout(yaxis = dict(range = [0,50]))
```


50
40
30
20
10
0
count
−15
−10
−5
0
5
10
15
20
weight_difference