

00_data_download

December 18, 2025

1 NBA 24-25 Data Import, Cleaning, and Processing

This notebook prepares the 2024-2025 NBA player game dataset from Kaggle. We do minimal preprocessing because the dataset is already clean.

In this notebook, I: - load the Kaggle NBA Player Stats 24/25 dataset - convert the “Data” column into a datetime - create per-game and per-season statistics - aggregate team-level game information

Import and create paths for raw CSV data.

```
[3]: import pandas as pd
import numpy as np
from pathlib import Path
from utils import add_gamekey_and_win, add_per_minute_stats

df = pd.read_csv("data/nba.csv")
df = add_gamekey_and_win(df)
df = add_per_minute_stats(df)
df.head()
```

```
[3]:
```

	Player	Tm	Opp	Res	MP	FG	FGA	FG%	3P	3PA	...	TOV	PF	\
0	Jayson Tatum	BOS	NYK	W	30.30	14	18	0.778	8	11	...	1	1	
1	Anthony Davis	LAL	MIN	W	37.58	11	23	0.478	1	3	...	1	1	
2	Derrick White	BOS	NYK	W	26.63	8	13	0.615	6	10	...	0	1	
3	Jrue Holiday	BOS	NYK	W	30.52	7	9	0.778	4	6	...	0	2	
4	Miles McBride	NYK	BOS	L	25.85	8	10	0.800	4	5	...	1	1	

	PTS	GmSc	Data	Win	GameKey	PTS_per_min	TRB_per_min	\
0	37	38.1	2024-10-22	1	2024-10-22_BOS_vs_NYK	1.221122	0.132013	
1	36	34.0	2024-10-22	1	2024-10-22_LAL_vs_MIN	0.957956	0.425758	
2	24	22.4	2024-10-22	1	2024-10-22_BOS_vs_NYK	0.901239	0.112655	
3	18	19.5	2024-10-22	1	2024-10-22_BOS_vs_NYK	0.589777	0.131062	
4	22	17.8	2024-10-22	0	2024-10-22_NYK_vs_BOS	0.851064	0.000000	

	AST_per_min
0	0.330033
1	0.106440
2	0.150207

```
3      0.131062
4      0.077369
```

```
[5 rows x 30 columns]
```

Create new columns that will help our analysis later on, including winning indicators, game ID, and per-minute stats.

```
[4]: # Convert date column
df["Data"] = pd.to_datetime(df["Data"], errors="coerce")

# Winning indicator
df["Win"] = (df["Res"] == "W").astype(int)

# Game ID
df["GameKey"] = (
    df["Data"].dt.strftime("%Y-%m-%d") + "_" + df["Tm"] + "_vs_" + df["Opp"])

df[["Player", "Tm", "Opp", "GameKey"]].head()

# Minutes played (avoiding division by zero) + per minute stats
mp_safe = df["MP"].clip(lower=1)

df["PTS_per_min"] = df["PTS"] / mp_safe
df["TRB_per_min"] = df["TRB"] / mp_safe
df["AST_per_min"] = df["AST"] / mp_safe

df[["Player", "MP", "PTS", "PTS_per_min"]].head()
```

```
[4]:
```

	Player	MP	PTS	PTS_per_min
0	Jayson Tatum	30.30	37	1.221122
1	Anthony Davis	37.58	36	0.957956
2	Derrick White	26.63	24	0.901239
3	Jrue Holiday	30.52	18	0.589777
4	Miles McBride	25.85	22	0.851064

Create per-player metrics by first building a player_game_stats table. - Create a player season summary by using a groupby to aggregate by player and team.

```
[5]: player_game_stats = df.copy

group_cols = ["Player", "Tm"]

player_season_summary = (
    df.groupby(group_cols)
    .agg(
        GamesPlayed = ("PTS", "count"),
        MP_mean = ("MP", "mean"),
```

```

        PTS_mean = ("PTS", "mean"),
        PTS_std = ("PTS", "std"),
        TRB_mean = ("TRB", "mean"),
        TRB_std = ("TRB", "std"),
        AST_mean = ("AST", "mean"),
        AST_std = ("AST", "std"),
        FGpct_mean = ("FG%", "mean"),
        ThreePct_mean = ("3P%", "mean"),
        FTpct_mean = ("FT%", "mean")
    )
    .reset_index()

print("player_season_summary shape:", player_season_summary.shape)
player_season_summary.head()

```

player_season_summary shape: (583, 13)

```

[5]:
      Player  Tm  GamesPlayed  MP_mean  PTS_mean  PTS_std  TRB_mean  \
0   A.J. Green  MIL           44  21.99750   7.659091  5.382760  2.250000
1   A.J. Lawson  TOR            4   3.75750   2.750000  2.500000  0.750000
2    AJ Johnson  MIL            8   5.67125   2.500000  4.598136  1.000000
3    AJ Johnson  WAS            1   8.83000   2.000000         NaN  1.000000
4  Aaron Gordon  DEN           30  26.66600  12.333333  6.608946  4.733333

      TRB_std  AST_mean  AST_std  FGpct_mean  ThreePct_mean  FTpct_mean
0  1.780057  1.272727  1.318273   0.426455         0.395273   0.140159
1  0.957427  0.000000  0.000000   0.666750         0.500000   0.125000
2  1.309307  0.875000  1.457738   0.260500         0.187500   0.062500
3         NaN  4.000000         NaN   0.250000         0.000000   0.000000
4  2.981938  3.066667  2.531639   0.510900         0.401000   0.630700

```

Now aggregate by team, gameID, data, and opponent to build the stats for all players from the same team in the same game. Compute the game stats including: - Team field goals percentage - Team 3 point shooting percentage - Team free throws percentage

```

[6]: team_group_cols = ["Tm", "GameKey", "Data", "Opp"]

team_game_stats = (
    df.groupby(team_group_cols)
    .agg(
        Team_PTS = ("PTS", "sum"),
        Team_TRB = ("TRB", "sum"),
        Team_AST = ("AST", "sum"),
        Team_TOV = ("TOV", "sum"),
        Team_FGM = ("FG", "sum"),
        Team_FGA = ("FGA", "sum"),
        Team_3PM = ("3P", "sum"),
    )
)

```

```

        Team_3PA = ("3PA", "sum"),
        Team_FTM = ("FT", "sum"),
        Team_FTA = ("FTA", "sum"),
        Team_Win = ("Win", "max")
    )
    .reset_index()

# Compute team shooting%
team_game_stats["Team_FG%"] = team_game_stats["Team_FGM"] / \
    ↪team_game_stats["Team_FGA"].replace(0, np.nan)
team_game_stats["Team_3P%"] = team_game_stats["Team_3PM"] / \
    ↪team_game_stats["Team_3PA"].replace(0, np.nan)
team_game_stats["Team_FT%"] = team_game_stats["Team_FTM"] / \
    ↪team_game_stats["Team_FTA"].replace(0, np.nan)

team_game_stats.head()

```

```

[6]:      Tm      GameKey      Data Opp Team_PTS Team_TRB Team_AST \
0  ATL  2024-10-23_ATL_vs_BRK  2024-10-23 BRK      120      45      25
1  ATL  2024-10-25_ATL_vs_CHO  2024-10-25 CHO      125      39      25
2  ATL  2024-10-27_ATL_vs_OKC  2024-10-27 OKC      104      49      24
3  ATL  2024-10-28_ATL_vs_WAS  2024-10-28 WAS      119      39      32
4  ATL  2024-10-30_ATL_vs_WAS  2024-10-30 WAS      120      41      28

      Team_TOV Team_FGM Team_FGA Team_3PM Team_3PA Team_FTM Team_FTA \
0           16      39      80          9      28      33      46
1           13      39      81         14      38      33      38
2           19      36      91         10      31      22      29
3           16      39      81         15      40      26      36
4           15      45      95         12      39      18      21

      Team_Win Team_FG% Team_3P% Team_FT%
0            1  0.487500  0.321429  0.717391
1            1  0.481481  0.368421  0.868421
2            0  0.395604  0.322581  0.758621
3            0  0.481481  0.375000  0.722222
4            0  0.473684  0.307692  0.857143

```

```

[7]: df.to_csv("data/player_game_stats_clean.csv", index=False)

```

— exports: - format: pdf —

```

[ ]:

```