

beautifulsoup-1

March 17, 2024

```
[71]: import requests
from bs4 import BeautifulSoup
import pandas as pd

# Step 1: Send a GET request to the URL
url = 'https://www.basketball-reference.com/leagues/NBA_2020_per_game.html'
response = requests.get(url)

# Check if the request was successful
if response.status_code == 200:
    # Step 2: Parse the HTML content
    soup = BeautifulSoup(response.content, 'html.parser')

    # Find the table containing player statistics
    table = soup.find('table', {'id': 'per_game_stats'})

    # Step 3: Extract data and perform data analytics
    if table:
        # Convert the table HTML to a pandas DataFrame
        df = pd.read_html(str(table))[0]

        # Display the first few rows of the DataFrame
        print("First few rows of the DataFrame:")
        print(df.head())

        # Perform data analytics or further processing as needed
        # For example, calculate summary statistics
        summary_stats = df.describe()
        print("\nSummary statistics:")
        print(summary_stats)

        # Save the DataFrame to a CSV file
        df.to_csv('nba_2020_player_stats.csv', index=False)
        print("\nData saved to 'nba_2020_player_stats.csv'")
    else:
        print("Table not found on the webpage.")
else:
```

```
print("Failed to retrieve the webpage. Status code:", response.status_code)
```

First few rows of the DataFrame:

```
Rk      Player Pos Age  Tm  G  GS  MP  FG  FGA ... \
0  1      Steven Adams  C  26  OKC  63  63  26.7  4.5  7.6 ...
1  2      Bam Adebayo  PF  22  MIA  72  72  33.6  6.1  11.0 ...
2  3  LaMarcus Aldridge  C  34  SAS  53  53  33.1  7.4  15.0 ...
3  4      Kyle Alexander  C  23  MIA  2  0  6.5  0.5  1.0 ...
4  5  Nickeil Alexander-Walker  SG  21  NOP  47  1  12.6  2.1  5.7 ...
```

```
      FT%  ORB  DRB  TRB  AST  STL  BLK  TOV  PF  PTS
0  .582  3.3  6.0  9.3  2.3  0.8  1.1  1.5  1.9  10.9
1  .691  2.4  7.8  10.2  5.1  1.1  1.3  2.8  2.5  15.9
2  .827  1.9  5.5  7.4  2.4  0.7  1.6  1.4  2.4  18.9
3  NaN  1.0  0.5  1.5  0.0  0.0  0.0  0.5  0.5  1.0
4  .676  0.2  1.6  1.8  1.9  0.4  0.2  1.1  1.2  5.7
```

[5 rows x 30 columns]

Summary statistics:

```
      Rk  Player  Pos  Age  Tm  G  GS  MP  FG  FGA ...  FT%  ORB  \
count  677    677  677  677  677  677  677  677  677  677 ...  644  677
unique  530    530  15  23  32  75  73  277  96  176 ...  272  41
top     Rk  Player  SG  24  TOT  G  0  MP  1.3  FGA ...  1.000  0.3
freq    26    26  141  79  60  26  191  26  28  26 ...  34  74
```

```
      DRB  TRB  AST  STL  BLK  TOV  PF  PTS
count  677  677  677  677  677  677  677
unique  87  104  73  23  26  44  42  208
top     DRB  TRB  0.8  0.4  0.1  0.8  1.9  PTS
freq    26  26  34  77  109  49  39  26
```

[4 rows x 30 columns]

Data saved to 'nba_2020_player_stats.csv'

```
[72]: ## loading the dataset
df=pd.read_csv('C:/Users/KSK/Downloads/nba_2020_player_stats.csv')
```

```
[73]: df.head()
```

```
[73]:  Rk      Player Pos Age  Tm  G  GS  MP  FG  FGA ... \
0  1      Steven Adams  C  26  OKC  63  63  26.7  4.5  7.6 ...
1  2      Bam Adebayo  PF  22  MIA  72  72  33.6  6.1  11.0 ...
2  3  LaMarcus Aldridge  C  34  SAS  53  53  33.1  7.4  15.0 ...
3  4      Kyle Alexander  C  23  MIA  2  0  6.5  0.5  1.0 ...
4  5  Nickeil Alexander-Walker  SG  21  NOP  47  1  12.6  2.1  5.7 ...
```

	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS
0	.582	3.3	6.0	9.3	2.3	0.8	1.1	1.5	1.9	10.9
1	.691	2.4	7.8	10.2	5.1	1.1	1.3	2.8	2.5	15.9
2	.827	1.9	5.5	7.4	2.4	0.7	1.6	1.4	2.4	18.9
3	NaN	1.0	0.5	1.5	0.0	0.0	0.0	0.5	0.5	1.0
4	.676	0.2	1.6	1.8	1.9	0.4	0.2	1.1	1.2	5.7

[5 rows x 30 columns]

```
[74]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Step 1: Load the dataset
#df = pd.read_csv('nba_2020_player_stats.csv')

# Step 2: Inspect the structure and summary statistics
print("Dataset information:")
print(df.info())
print("\nSummary statistics:")
print(df.describe())
```

```
Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 677 entries, 0 to 676
Data columns (total 30 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Rk           677 non-null    object
1   Player      677 non-null    object
2   Pos         677 non-null    object
3   Age         677 non-null    object
4   Tm          677 non-null    object
5   G           677 non-null    object
6   GS          677 non-null    object
7   MP          677 non-null    object
8   FG          677 non-null    object
9   FGA         677 non-null    object
10  FG%         675 non-null    object
11  3P          677 non-null    object
12  3PA         677 non-null    object
13  3P%         642 non-null    object
14  2P          677 non-null    object
15  2PA         677 non-null    object
16  2P%         671 non-null    object
17  eFG%        675 non-null    object
```

```

18 FT      677 non-null  object
19 FTA     677 non-null  object
20 FT%     644 non-null  object
21 ORB     677 non-null  object
22 DRB     677 non-null  object
23 TRB     677 non-null  object
24 AST     677 non-null  object
25 STL     677 non-null  object
26 BLK     677 non-null  object
27 TOV     677 non-null  object
28 PF      677 non-null  object
29 PTS     677 non-null  object

```

dtypes: object(30)

memory usage: 158.8+ KB

None

Summary statistics:

	Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	...	FT%	ORB	\
count	677	677	677	677	677	677	677	677	677	677	...	644	677	
unique	530	530	15	23	32	75	73	277	96	176	...	272	41	
top	Rk	Player	SG	24	TOT	G	0	MP	1.3	FGA	...	1.000	0.3	
freq	26	26	141	79	60	26	191	26	28	26	...	34	74	

	DRB	TRB	AST	STL	BLK	TOV	PF	PTS
count	677	677	677	677	677	677	677	677
unique	87	104	73	23	26	44	42	208
top	DRB	TRB	0.8	0.4	0.1	0.8	1.9	PTS
freq	26	26	34	77	109	49	39	26

[4 rows x 30 columns]

```

[75]: # Step 3: Handle missing or inconsistent data
      # Check for missing values
      print("\nMissing values:")
      print(df.isnull().sum())

```

Missing values:

```

Rk      0
Player  0
Pos     0
Age     0
Tm      0
G       0
GS      0
MP      0
FG      0
FGA     0

```

```
FG%      2
3P       0
3PA      0
3P%     35
2P       0
2PA      0
2P%      6
eFG%     2
FT       0
FTA      0
FT%     33
ORB      0
DRB      0
TRB      0
AST      0
STL      0
BLK      0
TOV      0
PF       0
PTS      0
dtype: int64
```

```
[76]: df.dtypes
```

```
[76]: Rk      object
      Player  object
      Pos    object
      Age    object
      Tm     object
      G      object
      GS    object
      MP    object
      FG    object
      FGA   object
      FG%   object
      3P    object
      3PA   object
      3P%   object
      2P    object
      2PA   object
      2P%   object
      eFG%  object
      FT    object
      FTA   object
      FT%   object
      ORB   object
      DRB   object
```

```
TRB      object
AST      object
STL      object
BLK      object
TOV      object
PF       object
PTS      object
dtype: object
```

```
[77]: # Step 1: Convert non-numeric values to NaN
df_numeric = df.apply(pd.to_numeric, errors='coerce')
```

```
[78]: df_numeric.head()
```

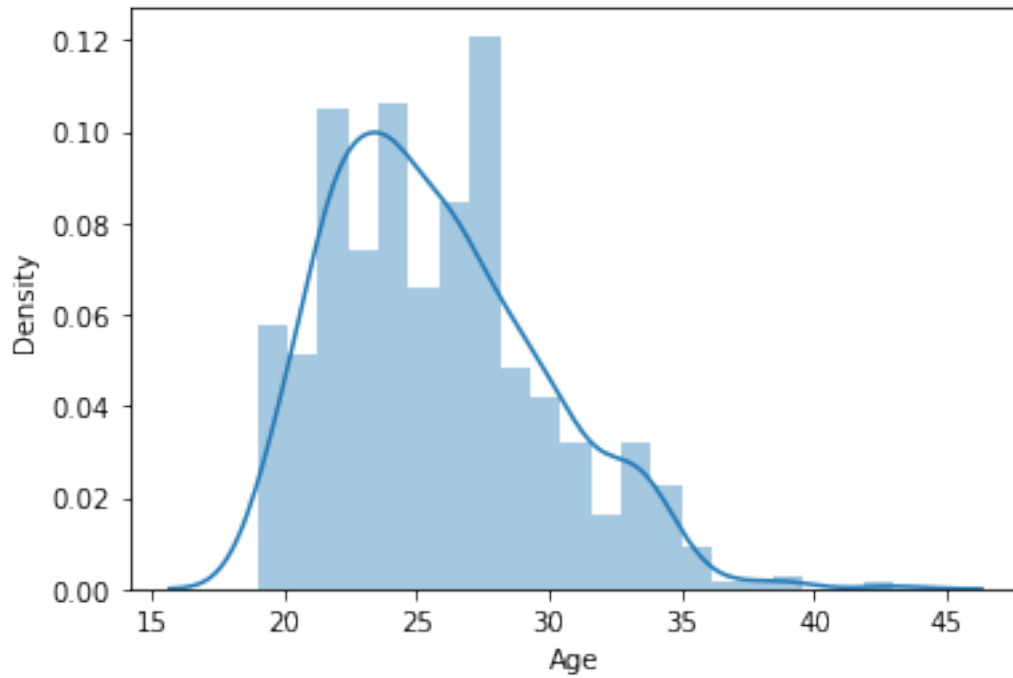
```
[78]:
```

	Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	...	FT%	ORB	\
0	1.0	NaN	NaN	26.0	NaN	63.0	63.0	26.7	4.5	7.6	...	0.582	3.3	
1	2.0	NaN	NaN	22.0	NaN	72.0	72.0	33.6	6.1	11.0	...	0.691	2.4	
2	3.0	NaN	NaN	34.0	NaN	53.0	53.0	33.1	7.4	15.0	...	0.827	1.9	
3	4.0	NaN	NaN	23.0	NaN	2.0	0.0	6.5	0.5	1.0	...	NaN	1.0	
4	5.0	NaN	NaN	21.0	NaN	47.0	1.0	12.6	2.1	5.7	...	0.676	0.2	

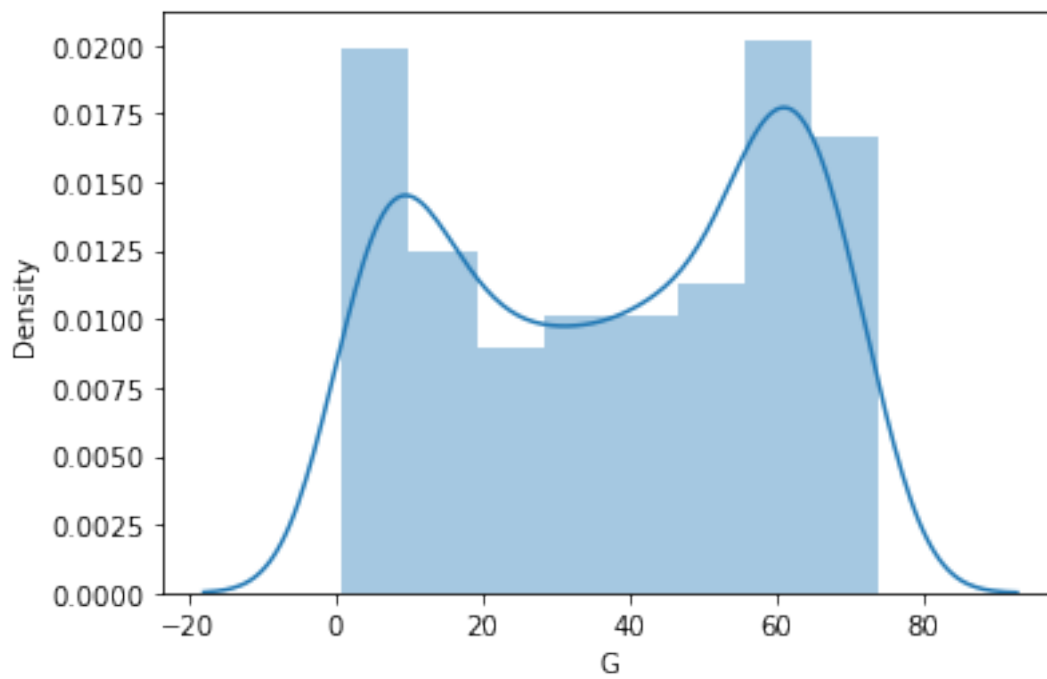
	DRB	TRB	AST	STL	BLK	TOV	PF	PTS
0	6.0	9.3	2.3	0.8	1.1	1.5	1.9	10.9
1	7.8	10.2	5.1	1.1	1.3	2.8	2.5	15.9
2	5.5	7.4	2.4	0.7	1.6	1.4	2.4	18.9
3	0.5	1.5	0.0	0.0	0.0	0.5	0.5	1.0
4	1.6	1.8	1.9	0.4	0.2	1.1	1.2	5.7

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

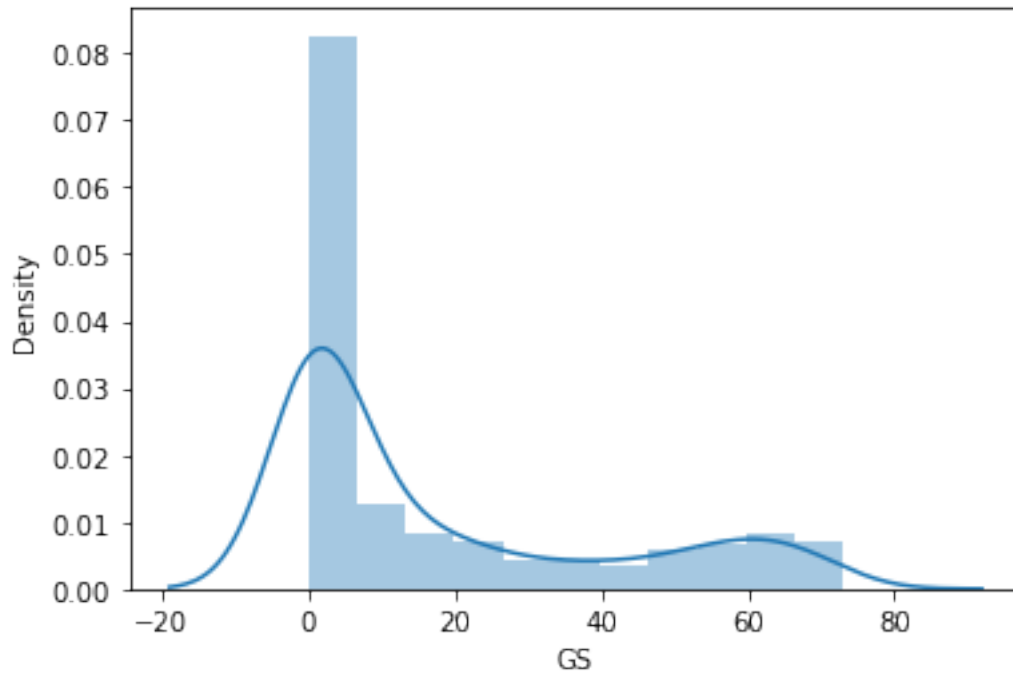
```
[80]: import warnings
warnings.filterwarnings('ignore')
sns.distplot(df_numeric['Age'])
plt.show()
```



```
[81]: import warnings
warnings.filterwarnings('ignore')
sns.distplot(df_numeric['G'])
plt.show()
```



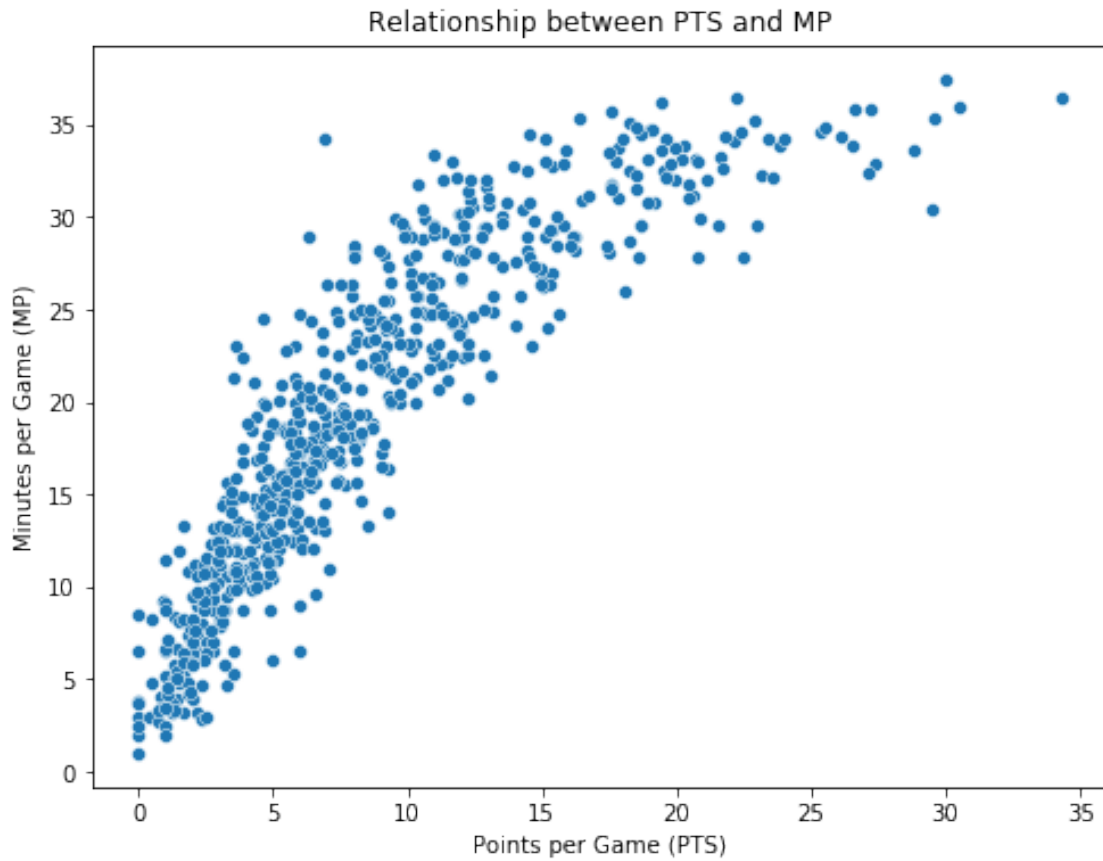
```
[82]: import warnings
warnings.filterwarnings('ignore')
sns.distplot(df_numeric['GS'])
plt.show()
```



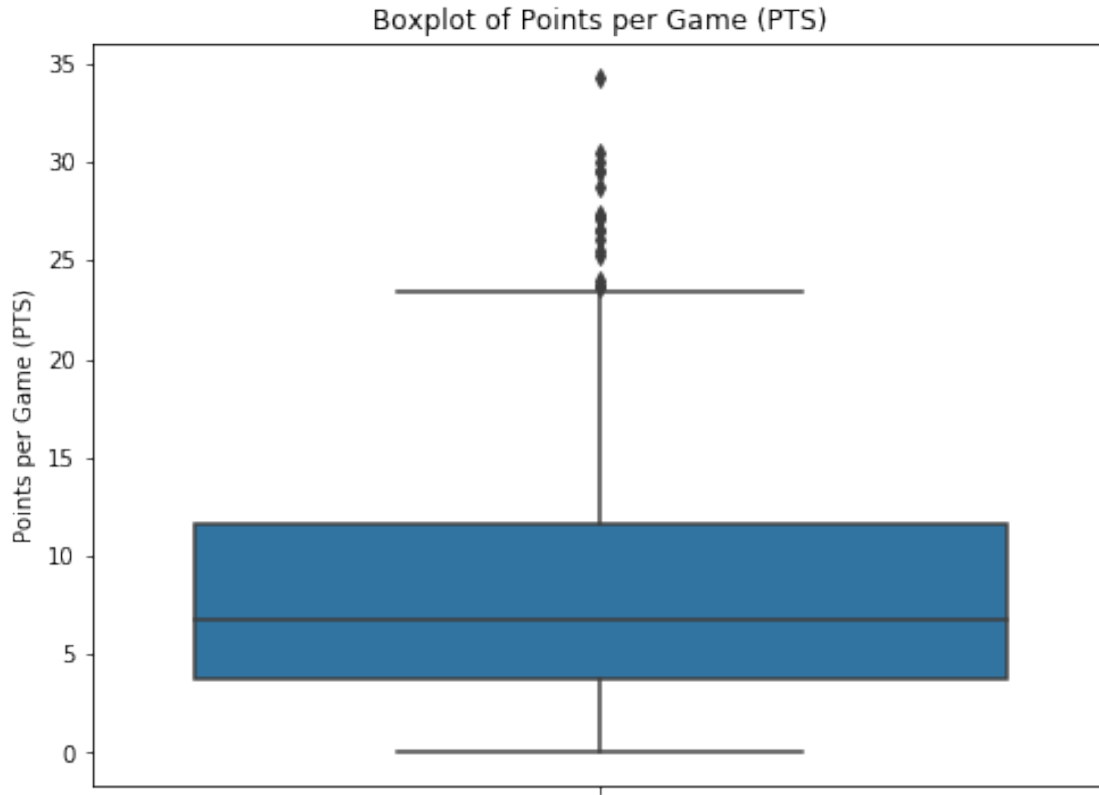
```
[83]: # Step 5: Analyze categorical variables
# Unique values in categorical variables
print("\nUnique values in categorical variables:")
for column in df_numeric.select_dtypes(include='object'):
    print(f"{column}: {df_numeric[column].unique()}")
```

Unique values in categorical variables:

```
[84]: # Step 6: Visualize relationships between variables
# Example: Scatter plot between points per game (PTS) and minutes per game (MP)
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df_numeric, x='PTS', y='MP')
plt.title('Relationship between PTS and MP')
plt.xlabel('Points per Game (PTS)')
plt.ylabel('Minutes per Game (MP)')
plt.show()
```

```
[85]: # Step 7: Identify outliers or anomalies
# Example: Boxplot for points per game (PTS)
plt.figure(figsize=(8, 6))
sns.boxplot(data=df_numeric, y='PTS')
plt.title('Boxplot of Points per Game (PTS)')
plt.ylabel('Points per Game (PTS)')
plt.show()
```



```
[86]: # Step 8: Derive additional features if necessary
# Example: Calculate field goal percentage (FG%)
df_numeric['FG%'] = df_numeric['FG'] / df_numeric['FGA']
```

```
[87]: # Step 9: Draw conclusions and insights from the analysis
# Example: Analyze correlation between numerical variables
correlation_matrix = df_numeric.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
```

