eda-pr-2-ml-course

October 2, 2023

1 Exploratory Data Analysis (EDA)

importing libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

To explore data, it is necessary to load some data

LOADING DATA (Virat kohlis centuries)

[57]:
```
df = pd.read_csv("71 Centuries of Virat Kohli.csv")
df
```

[57]:

	Score	Out/Not Out	Against	Batting Order	Inn.	Strike Rate	\
0	116	Out	Australia	6	2	NaN	
1	103	Out	New Zealand	5	2	NaN	
2	103	Out	England	5	2	NaN	
3	107	Out	Australia	5	2	NaN	
4	119	Out	South Africa	4	1	NaN	
..	…	…	…	…	…	…	
66	116	Out	Australia	3	1	96.67	
67	123	Out	Australia	3	2	129.47	
68	120	Out	West Indies	3	1	96.00	
69	114	Not Out	West Indies	3	2	115.15	
70	122	Not Out	Afganistan	1	1	200.00	

	Venue	Column1	H/A	Date	\
0	Adelaide Oval	Adelaide	Away	24-01-2012	
1	M. Chinnaswamy Stadium	Bangalore	Home	31-08-2012	
2	Vidarbha Cricket Association Stadium	Nagpur	Home	13-12-2012	
3	M. A. Chidambaram Stadium	Chennai	Home	22-02-2013	
4	Wanderers Stadium	Johannesburg	Away	18-12-2013	
..	…	…	…	…	
66	Vidarbha Cricket Association Stadium	Nagpur	Home	05-03-2019	
67	JSCA International Stadium	Ranchi	Home	08-03-2019	
68	Queen's Park Oval	Port of Spain	Away	11-08-2019	

```
69                     Queen's Park Oval   Port of Spain  Away  14-08-2019
70   Dubai International Cricket Stadium           Dubai  Away  08-09-2022

   Result Format Man of the Match Captain  Unnamed: 14
0    Lost   Test               No      No          NaN
1     Won   Test              Yes      No          NaN
2   Drawn   Test               No      No          NaN
3     Won   Test               No      No          NaN
4   Drawn   Test               No      No          NaN
..    …    …                 …       …           …
66    Won    ODI              Yes     Yes          NaN
67   Lost    ODI               No     Yes          NaN
68    Won    ODI              Yes     Yes          NaN
69    Won    ODI              Yes     Yes          NaN
70    Won   T20I              Yes      No          NaN

[71 rows x 15 columns]
```

checking for data information i.e rows & coulmns

[46]:
```
print("rows:", df.shape[0],"\ncolumns:", df.shape[1])
df.info()
```

```
rows: 71
columns: 15
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 71 entries, 0 to 70
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Score             71 non-null     int64
 1   Out/Not Out       71 non-null     object
 2   Against           71 non-null     object
 3   Batting Order     71 non-null     int64
 4   Inn.              71 non-null     int64
 5   Strike Rate       44 non-null     float64
 6   Venue             71 non-null     object
 7   Column1           71 non-null     object
 8   H/A               71 non-null     object
 9   Date              71 non-null     object
 10  Result            71 non-null     object
 11  Format            71 non-null     object
 12  Man of the Match  71 non-null     object
 13  Captain           71 non-null     object
 14  Unnamed: 14       0 non-null      float64
dtypes: float64(2), int64(3), object(10)
memory usage: 8.4+ KB
```

DATA CLEANING

1. checking for missing values

[33]:
```
null_values = df.isnull().sum().sort_values(ascending= False)
null_values
```

[33]:
```
Unnamed: 14         71
Strike Rate         27
Score                0
Out/Not Out          0
Against              0
Batting Order        0
Inn.                 0
Venue                0
Column1              0
H/A                  0
Date                 0
Result               0
Format               0
Man of the Match     0
Captain              0
dtype: int64
```

checking percentage of missing values

[34]:
```
per_null = ((null_values / df.shape[0]) * 100).sort_values(ascending = False)
per_null
```

[34]:
```
Unnamed: 14         100.000000
Strike Rate          38.028169
Score                 0.000000
Out/Not Out           0.000000
Against               0.000000
Batting Order         0.000000
Inn.                  0.000000
Venue                 0.000000
Column1               0.000000
H/A                   0.000000
Date                  0.000000
Result                0.000000
Format                0.000000
Man of the Match      0.000000
Captain               0.000000
dtype: float64
```

[23]:
```
df.columns
```

[23]:
```
Index(['Out/Not Out', 'Against', 'Batting Order', 'Inn.', 'Strike Rate',
       'Venue', 'Column1', 'H/A', 'Date', 'Result', 'Format',
       'Man of the Match', 'Captain'],
      dtype='object')
```

FILLING MISSING DATA

[37]:
```
# Filling strike rate column with median
df["Strike Rate"].fillna(df["Strike Rate"].median(), inplace = True)
```

[38]:
```
df.isnull().sum().sort_values(ascending= False)
```

[38]:
```
Unnamed: 14         71
Score                0
Out/Not Out          0
Against              0
Batting Order        0
Inn.                 0
Strike Rate          0
Venue                0
Column1              0
H/A                  0
Date                 0
Result               0
Format               0
Man of the Match     0
Captain              0
dtype: int64
```

CHECKING FOR DUPLICATE VALUES

[71]:
```
df.duplicated().sum()
```

[71]:
```
0
```

1.1 HYPOTHESIS

score of virat kohli depends on the number he played

[68]:
```
# score VS batting order
df.groupby('Batting Order')['Score'].sum()
```

[68]:
```
Batting Order
1     122
3    4380
4    4451
5     313
6     116
```

```
Name: Score, dtype: int64
```

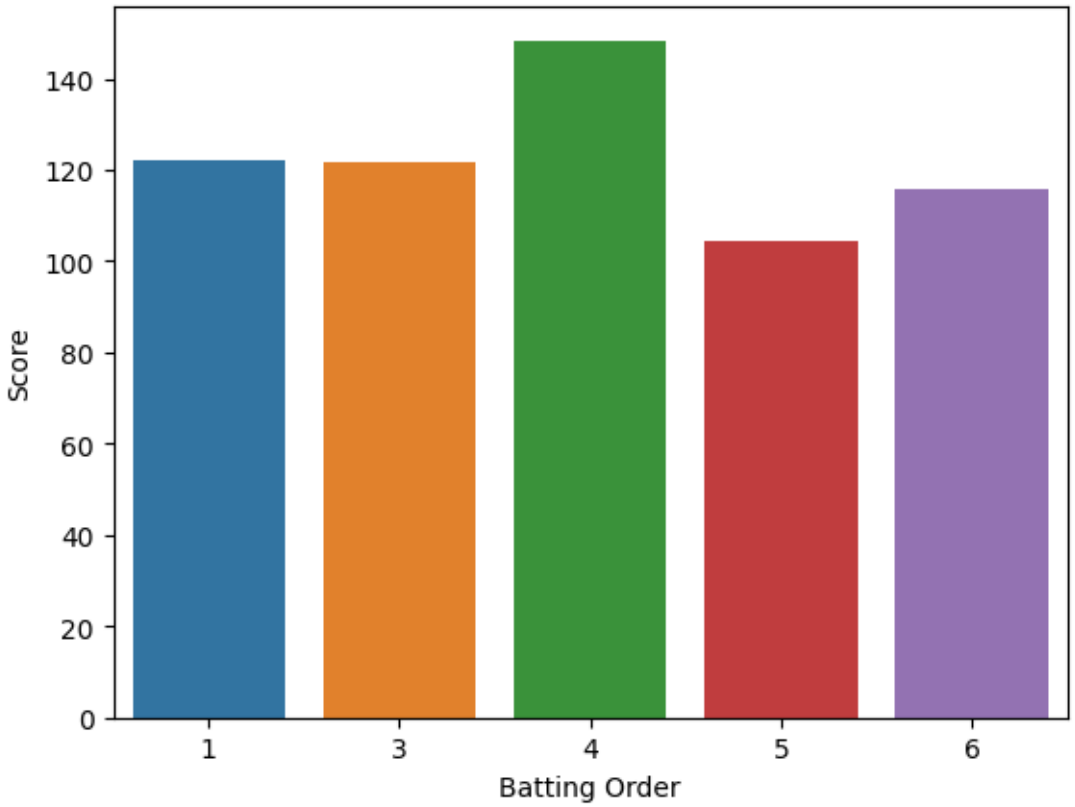
[70]:
```
sns.barplot(x='Batting Order', y='Score', data=df, ci= False)
plt.show
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_1320\588000527.py:1: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=('ci', False)` for the same
effect.

  sns.barplot(x='Batting Order', y='Score', data=df, ci= False)
```

[70]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



[ ]:
```

```