EDS > VIVEK.PY > ...

```python
import pandas as pd

# Load the dataset
df = pd.read_csv('teamwise_home_and_away (3).csv')

# Clean column names (optional but safe)
df.columns = df.columns.str.strip()

# Check the column names to ensure proper naming
print("Columns in dataset:", df.columns)

# 1. Find team with maximum home wins
max_home_wins_team = df[df['home_wins'] == df['home_wins'].max()]['team'].values[0]
print("1. Team with most home wins:", max_home_wins_team)
print('\n')


# 2. Find team with maximum away losses
# Ensure the column name is correct and exists
if 'away_losses' in df.columns:
    max_away_losses_team = df[df['away_losses'] == df['away_losses'].max()]['team'].values[0]
    print("2. Team with most away losses:", max_away_losses_team)
else:
    print("Column 'away_losses' not found in dataset.")
print('\n')


# 3. Find average home matches
```

```
Click to add a breakpoint
                                    ge home matches
avg_home_matches = df['home_matches'].mean()
print("3. Average home matches:", avg_home_matches)
print('\n')


# 4. Find team with highest win percentage
# Calculate total wins and total matches based on available columns
df['total_wins'] = df['home_wins'] + df['away_wins']
df['total_matches'] = df['home_matches'] + df['away_matches']
df['win_percentage'] = (df['total_wins'] / df['total_matches']) * 100
highest_win_team = df[df['win_percentage'] == df['win_percentage'].max()]['team'].values[0]
print("4. Team with highest win %:", highest_win_team)
print('\n')


# 5. Count teams with home win rate > 50%
df['home_win_rate'] = df['home_wins'] / df['home_matches']
teams_high_home_win = df[df['home_win_rate'] > 0.5]['team'].count()
print("5. Teams with >50% home win rate:", teams_high_home_win)
print('\n')


# 6. Find team with minimum away wins
min_away_wins_team = df[df['away_wins'] == df['away_wins'].min()]['team'].values[0]
print("6. Team with least away wins:", min_away_wins_team)
print('\n')
```

```python
# 7. Find total number of draws
# Assuming draws are available as columns or calculated as needed
# For example, using home_draws and away_draws (if available), or leaving this task out
total_draws = 0  # Update with actual calculation if data is available
print("7. Total draws:", total_draws)
print('\n')


# 8. Team with largest difference between home and away wins
df['win_diff'] = abs(df['home_wins'] - df['away_wins'])
largest_diff_team = df[df['win_diff'] == df['win_diff'].max()]['team'].values[0]
print("8. Team with largest win difference:", largest_diff_team)
print('\n')


# 9. Find correlation between home and away wins
correlation = df['home_wins'].corr(df['away_wins'])
print("9. Correlation between home and away wins:", correlation)
print('\n')


# 10. Top 5 teams with most total losses
# If total_losses column is available, use it; otherwise, calculate based on wins
# Example assuming total_losses = total_matches - (home_wins + away_wins)
df['total_losses'] = df['total_matches'] - df['total_wins']
top5_losses = df[['team', 'total_losses']].sort_values('total_losses', ascending=False).head(5)
print("10. Top 5 teams with most losses:\n", top5_losses)
print('\n')
```

```
87  # 11. Mean away draws (if data is available for away_draws)
C:\Users\Engeers\Desktop\python_practice\python_practice\EDS\VIVEK.PY culation if away_draws data is available
89  print("11. Mean away draws:", mean_away_draws)
90  print('\n')
91
92
93  # 12. Team with maximum total matches
94  max_matches_team = df[df['total_matches'] == df['total_matches'].max()]['team'].values[0]
95  print("12. Team with most matches:", max_matches_team)
96  print('\n')
97
98
99  # 13. Teams with more home wins than away wins
100 teams_more_home = df[df['home_wins'] > df['away_wins']]['team'].tolist()
101 print("13. Teams with more home wins:", teams_more_home)
102 print('\n')
103
104
105 # 14. Check total matches consistency
106 df['computed_matches'] = df['home_matches'] + df['away_matches']
107 consistent = (df['computed_matches'] == df['total_matches']).all()
108 print("14. Matches consistency correct?:", consistent)
109 print('\n')
110
111
112
```

```python
# 15. Standard deviation of total wins
std_total_wins = df['total_wins'].std()
print("15. Std dev of total wins:", std_total_wins)
print('\n')


# 16. Teams with away win rate < 30%
df['away_win_rate'] = df['away_wins'] / df['away_matches']
teams_low_away_win = df[df['away_win_rate'] < 0.3]['team'].tolist()
print("16. Teams with away win rate <30%:", teams_low_away_win)
print('\n')



# 17. Most balanced team (equal win, draw, loss %)
# Skip calculation if 'total_draws' is missing
if 'total_draws' in df.columns:
    df['draw_percentage'] = (df['total_draws'] / df['total_matches']) * 100
    df['loss_percentage'] = (df['total_losses'] / df['total_matches']) * 100
    df['balance_std'] = df[['win_percentage', 'draw_percentage', 'loss_percentage']].std(axis=1)
    most_balanced_team = df[df['balance_std'] == df['balance_std'].min()]['team'].values[0]
    print("17. Most balanced team:", most_balanced_team)
else:
    print("17. Cannot calculate most balanced team - 'total_draws' column missing.")
print('\n')
```

Click to add a breakpoint

```python
# 18. Ratio of total home wins to away wins
total_home_wins = df['home_wins'].sum()
total_away_wins = df['away_wins'].sum()
home_away_ratio = total_home_wins / total_away_wins
print("18. Home to away win ratio:", home_away_ratio)
print('\n')

# 19. Skewness of total matches
skewness = df['total_matches'].skew()
print("19. Skewness of total matches:", skewness)
print('\n')


# 20. Group teams into 'High Win Rate' and 'Others'
df['win_group'] = df['win_percentage'].apply(lambda x: 'High Win Rate' if x > 60 else 'Others')
print("20. Teams grouped by win rate:\n", df[['team', 'win_group']])
print('\n')
```

```
PS C:\Users\Engeers\Desktop\python_practice\python_practice> python -u "c:\Users\Engeers\Desktop\python_practice\python_practice\EDS\vivek.py"
Columns in dataset: Index(['team', 'home_wins', 'away_wins', 'home_matches', 'away_matches',
       'home_win_percentage', 'away_win_percentage'],
      dtype='object')
1. Team with most home wins: Mumbai Indians


Column 'away_losses' not found in dataset.


3. Average home matches: 53.5


4. Team with highest win %: Rising Pune Supergiant


5. Teams with >50% home win rate: 3


6. Team with least away wins: Kochi Tuskers Kerala


7. Total draws: 0


8. Team with largest win difference: Kolkata Knight Riders


9. Correlation between home and away wins: 0.8862442050777292


10. Top 5 teams with most losses:
                         team  total_losses
8   Royal Challengers Bangalore           96
7              Kings XI Punjab           94
10            Delhi Daredevils           94
9         Kolkata Knight Riders           86
1               Mumbai Indians           78
```

6. Team with least away wins: Kochi Tuskers Kerala

7. Total draws: 0

8. Team with largest win difference: Kolkata Knight Riders

9. Correlation between home and away wins: 0.8862442050777292

10. Top 5 teams with most losses:

	team	total_losses
8	Royal Challengers Bangalore	96
7	Kings XI Punjab	94
10	Delhi Daredevils	94
9	Kolkata Knight Riders	86
1	Mumbai Indians	78

11. Mean away draws: 0

12. Team with most matches: Mumbai Indians

13. Teams with more home wins: ['Mumbai Indians', 'Chennai Super Kings', 'Sunrisers Hyderabad', 'Deccan Chargers']

14. Matches consistency correct?: True

15. Std dev of total wins: 38.36586452094571

16. Teams with away win rate <30%: ['Pune Warriors']

16. Teams with away win rate <30%: ['Pune Warriors']

17. Cannot calculate most balanced team - 'total_draws' column missing.

18. Home to away win ratio: 0.8131067961165048

19. Skewness of total matches: -0.2612856611972961

20. Teams grouped by win rate:

	team	win_group
0	Rising Pune Supergiant	High Win Rate
1	Mumbai Indians	Others
2	Chennai Super Kings	High Win Rate
3	Delhi Capitals	High Win Rate
4	Sunrisers Hyderabad	Others
5	Rajasthan Royals	Others
6	Deccan Chargers	Others
7	Kings XI Punjab	Others
8	Royal Challengers Bangalore	Others
9	Kolkata Knight Riders	Others
10	Delhi Daredevils	Others
11	Pune Warriors	Others
12	Kochi Tuskers Kerala	Others
13	Gujarat Lions	Others

PS C:\Users\Engeers\Desktop\python_practice\python_practice>

```
20. Teams grouped by win rate:
                         team      win_group
0        Rising Pune Supergiant  High Win Rate
1                Mumbai Indians        Others
2           Chennai Super Kings  High Win Rate
3                Delhi Capitals  High Win Rate
4            Sunrisers Hyderabad        Others
5               Rajasthan Royals        Others
6                Deccan Chargers        Others
7                Kings XI Punjab        Others
8    Royal Challengers Bangalore        Others
9          Kolkata Knight Riders        Others
10              Delhi Daredevils        Others
11                Pune Warriors        Others
12         Kochi Tuskers Kerala        Others
13                Gujarat Lions        Others


PS C:\Users\Engeers\Desktop\python_practice\python_practice> python -u "c:\Users\Engeers\Desktop\python_practice\python_practice\EDS\VIVEK.PY"
Columns in dataset: Index(['team', 'home_wins', 'away_wins', 'home_matches', 'away_matches',
       'home_win_percentage', 'away_win_percentage'],
      dtype='object')
1. Team with most home wins: Mumbai Indians


Column 'away_losses' not found in dataset.


3. Average home matches: 53.5


4. Team with highest win %: Rising Pune Supergiant


5. Teams with >50% home win rate: 3


6. Team with least away wins: Kochi Tuskers Kerala
```

```
PS C:\Users\Engeers\Desktop\python_practice\python_practice> python -u "c:\Users\Engeers\Desktop\python_practice\python_practice\EDS\vivek.py"
Columns in dataset: Index(['team', 'home_wins', 'away_wins', 'home_matches', 'away_matches',
       'home_win_percentage', 'away_win_percentage'],
      dtype='object')
1. Team with most home wins: Mumbai Indians


Column 'away_losses' not found in dataset.


3. Average home matches: 53.5


4. Team with highest win %: Rising Pune Supergiant


5. Teams with >50% home win rate: 3


6. Team with least away wins: Kochi Tuskers Kerala


7. Total draws: 0


8. Team with largest win difference: Kolkata Knight Riders


9. Correlation between home and away wins: 0.8862442050777292


10. Top 5 teams with most losses:
                         team  total_losses
8   Royal Challengers Bangalore           96
7              Kings XI Punjab           94
10            Delhi Daredevils           94
9         Kolkata Knight Riders           86
1               Mumbai Indians           78
```

11. Mean away draws: 0

12. Team with most matches: Mumbai Indians

13. Teams with more home wins: ['Mumbai Indians', 'Chennai Super Kings', 'Sunrisers Hyderabad', 'Deccan Chargers']

14. Matches consistency correct?: True

15. Std dev of total wins: 38.36586452094571

16. Teams with away win rate <30%: ['Pune Warriors']

17. Cannot calculate most balanced team - 'total_draws' column missing.

18. Home to away win ratio: 0.8131067961165048

19. Skewness of total matches: -0.2612856611972961

20. Teams grouped by win rate:

	team	win_group
0	Rising Pune Supergiant	High Win Rate
1	Mumbai Indians	Others
2	Chennai Super Kings	High Win Rate
3	Delhi Capitals	High Win Rate
4	Sunrisers Hyderabad	Others
5	Rajasthan Royals	Others
6	Deccan Chargers	Others
7	Kings XI Punjab	Others
8	Royal Challengers Bangalore	Others
9	Kolkata Knight Riders	Others
10	Delhi Daredevils	Others
11	Pune Warriors	Others