IPL

April 26, 2023

1 Cricket Hackathon 2023

- https://iitm-ipl.web.app/
- https://drive.google.com/drive/folders/1UA8LLt_D1W4dN-XrfUrbFg5PMrbuBzM4
- https://www.sportskeeda.com/cricket/yesterday-ipl-match-result
- https://www.iplt20.com/matches/schedule/men
- https://aiimsexams.org/ipl-schedule/
- https://www.icccricketschedule.com/ipl-2023-schedule-team-venue-time-table-pdf-point-table-ranking-winning-prediction/
- https://www.timesofsports.com/cricket/ipl/squad-2023/

[ ]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

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

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer

from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.decomposition import PCA

from sklearn.linear_model import LinearRegression, Ridge, Lasso, ElasticNet
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.neural_network import MLPRegressor
```

[ ]:
```
def log(*args):
    print(' ', *args)
```

[ ]:
```
def to_kebab_case(string):
    return '-'.join(
        string.replace(",", "").replace(".", "").split()
    ).lower()
```

[ ]:
```
ball_by_ball = pd.read_csv('./Data/IPL_Ball_by_Ball_2008_2022.csv')

matches_result = pd.read_csv('./Data/IPL_Matches_Result_2008_2022.csv')

data__ipl_2023_venues = pd.read_csv('./Data/Ipl_2023 _cricketers - Venue.csv').
 ↪rename(columns={
    'Venue': 'venue'
})
data__ipl_2023_teams = pd.read_csv('./Data/Ipl_2023 _cricketers - Team name.
 ↪csv').rename(columns={
    'Teams': 'team'
})
data__ipl_2023_players = pd.read_csv('./Data/Ipl_2023 _cricketers - Players.
 ↪csv').rename(columns={
    'Team ': 'team'
})
```

2 Cleaning `ball_by_ball`, `matches_result`

-

2.1 Change column names, drop unnecessary columns [in ball__by__ball, matches__result]

[ ]:
```
ball_by_ball_orig = ball_by_ball
matches_result_orig = matches_result

ball_by_ball = ball_by_ball.rename(columns={
    'ID': 'match_id',
    'ballnumber': 'ball_number',
    'non-striker': 'non_striker',
    'BattingTeam': 'batting_team',
}).loc[:, [
    'match_id',
    'innings',
    'batting_team',
    'overs',
    'ball_number',
    'batter',
    'bowler',
    'total_run',
```

```
]]

matches_result = matches_result.rename(columns={
    'ID': 'match_id',
    'Team1': 'team_1',
    'Team2': 'team_2',
    'Venue': 'venue',
}).loc[:, [
    'match_id',
    'team_1',
    'team_2',
    'venue',
]]
```

[ ]:
```
ball_by_ball_orig.shape
ball_by_ball_orig.head()
matches_result_orig.shape
matches_result_orig.head()
```

[ ]:
```
(225954, 17)
```

[ ]:
```
        ID  innings  overs  ballnumber       batter          bowler  \
0  1312200        1      0           1  YBK Jaiswal  Mohammed Shami
1  1312200        1      0           2  YBK Jaiswal  Mohammed Shami
2  1312200        1      0           3   JC Buttler  Mohammed Shami
3  1312200        1      0           4  YBK Jaiswal  Mohammed Shami
4  1312200        1      0           5  YBK Jaiswal  Mohammed Shami

   non-striker extra_type  batsman_run  extras_run  total_run  non_boundary  \
0   JC Buttler        NaN            0           0          0             0
1   JC Buttler    legbyes            0           1          1             0
2  YBK Jaiswal        NaN            1           0          1             0
3   JC Buttler        NaN            0           0          0             0
4   JC Buttler        NaN            0           0          0             0

   isWicketDelivery player_out kind fielders_involved       BattingTeam
0                 0        NaN  NaN               NaN  Rajasthan Royals
1                 0        NaN  NaN               NaN  Rajasthan Royals
2                 0        NaN  NaN               NaN  Rajasthan Royals
3                 0        NaN  NaN               NaN  Rajasthan Royals
4                 0        NaN  NaN               NaN  Rajasthan Royals
```

[ ]:
```
(950, 20)
```

[ ]:
```
        ID       City        Date Season  MatchNumber  \
0  1312200  Ahmedabad  2022-05-29   2022        Final
1  1312199  Ahmedabad  2022-05-27   2022  Qualifier 2
```

```
2  1312198    Kolkata  2022-05-25   2022   Eliminator
3  1312197    Kolkata  2022-05-24   2022  Qualifier 1
4  1304116     Mumbai  2022-05-22   2022           70

                        Team1                Team2  \
0            Rajasthan Royals        Gujarat Titans
1  Royal Challengers Bangalore      Rajasthan Royals
2  Royal Challengers Bangalore  Lucknow Super Giants
3            Rajasthan Royals        Gujarat Titans
4         Sunrisers Hyderabad          Punjab Kings

                             Venue            TossWinner TossDecision  \
0  Narendra Modi Stadium, Ahmedabad      Rajasthan Royals          bat
1  Narendra Modi Stadium, Ahmedabad      Rajasthan Royals        field
2            Eden Gardens, Kolkata  Lucknow Super Giants        field
3            Eden Gardens, Kolkata        Gujarat Titans        field
4         Wankhede Stadium, Mumbai   Sunrisers Hyderabad          bat

  SuperOver                  WinningTeam    WonBy  Margin method  \
0         N                Gujarat Titans  Wickets     7.0    NaN
1         N              Rajasthan Royals  Wickets     7.0    NaN
2         N  Royal Challengers Bangalore     Runs    14.0    NaN
3         N                Gujarat Titans  Wickets     7.0    NaN
4         N                  Punjab Kings  Wickets     5.0    NaN

  Player_of_Match                                       Team1Players  \
0       HH Pandya  ['YBK Jaiswal', 'JC Buttler', 'SV Samson', 'D …
1      JC Buttler  ['V Kohli', 'F du Plessis', 'RM Patidar', 'GJ …
2      RM Patidar  ['V Kohli', 'F du Plessis', 'RM Patidar', 'GJ …
3       DA Miller  ['YBK Jaiswal', 'JC Buttler', 'SV Samson', 'D …
4   Harpreet Brar  ['PK Garg', 'Abhishek Sharma', 'RA Tripathi', …

                                      Team2Players        Umpire1  \
0  ['WP Saha', 'Shubman Gill', 'MS Wade', 'HH Pan…    CB Gaffaney
1  ['YBK Jaiswal', 'JC Buttler', 'SV Samson', 'D …    CB Gaffaney
2  ['Q de Kock', 'KL Rahul', 'M Vohra', 'DJ Hooda…  J Madanagopal
3  ['WP Saha', 'Shubman Gill', 'MS Wade', 'HH Pan…   BNJ Oxenford
4  ['JM Bairstow', 'S Dhawan', 'M Shahrukh Khan',…   AK Chaudhary

         Umpire2
0    Nitin Menon
1    Nitin Menon
2       MA Gough
3      VK Sharma
4  NA Patwardhan
```

[ ]:
```
ball_by_ball.shape
ball_by_ball.head()
matches_result.shape
matches_result.head()
```

[ ]: (225954, 8)

[ ]:

	match_id	innings	batting_team	overs	ball_number	batter
0	1312200	1	Rajasthan Royals	0	1	YBK Jaiswal
1	1312200	1	Rajasthan Royals	0	2	YBK Jaiswal
2	1312200	1	Rajasthan Royals	0	3	JC Buttler
3	1312200	1	Rajasthan Royals	0	4	YBK Jaiswal
4	1312200	1	Rajasthan Royals	0	5	YBK Jaiswal

	bowler	total_run
0	Mohammed Shami	0
1	Mohammed Shami	1
2	Mohammed Shami	1
3	Mohammed Shami	0
4	Mohammed Shami	0

[ ]: (950, 4)

[ ]:

	match_id	team_1	team_2	venue
0	1312200	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad
1	1312199	Royal Challengers Bangalore	Rajasthan Royals	Narendra Modi Stadium, Ahmedabad
2	1312198	Royal Challengers Bangalore	Lucknow Super Giants	Eden Gardens, Kolkata
3	1312197	Rajasthan Royals	Gujarat Titans	Eden Gardens, Kolkata
4	1304116	Sunrisers Hyderabad	Punjab Kings	Wankhede Stadium, Mumbai

-

2.2 Some stats

[ ]:
```
log('ball_by_ball match_id.nunique:', ball_by_ball.match_id.nunique())
log('ball_by_ball batting_team.nunique:', ball_by_ball.batting_team.nunique())
log('ball_by_ball union1d(batter, bowler).shape:', np.union1d(
    ball_by_ball.batter.unique(), ball_by_ball.bowler.unique()
).shape)
log('ball_by_ball innings.unique:', ball_by_ball.innings.unique())
log('ball_by_ball overs.unique:', ball_by_ball.overs.unique())
```

```
 ball_by_ball match_id.nunique: 950
 ball_by_ball batting_team.nunique: 18
 ball_by_ball union1d(batter, bowler).shape: (652,)
 ball_by_ball innings.unique: [1 2 3 4 5 6]
 ball_by_ball overs.unique: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
17 18 19]
```

[ ]:
```
log('matches_result match_id.nunique:', matches_result.match_id.nunique())
log('matches_result venue.nunique:', matches_result.venue.nunique())
log('matches_result union1d(team_1, team_2).shape:', np.union1d(
    matches_result.team_1.unique(), matches_result.team_2.unique()
).shape)
```

```
 matches_result match_id.nunique: 950
 matches_result venue.nunique: 49
 matches_result union1d(team_1, team_2).shape: (18,)
```

-

2.3 Venues Mapping

[ ]:
```
# matches_result_orig.groupby(['City', 'Venue'], dropna=False)['Venue'].
 ↪describe()
```

: https://www.iplt20.com/matches/schedule/men

[ ]:
```
venue_mapping_normal = {
    "Arun Jaitley Stadium": "Arun Jaitley Stadium",
    "Arun Jaitley Stadium, Delhi": "Arun Jaitley Stadium",
    "Feroz Shah Kotla": "Arun Jaitley Stadium",
    "Barsapara Cricket Stadium": "Barsapara Cricket Stadium",
    "Barsapara Cricket Stadium, Guwahati": "Barsapara Cricket Stadium",
    "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium": "Bharat␣
 ↪Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
    "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium, Lucknow":␣
 ↪"Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
    "Eden Gardens": "Eden Gardens",
    "Eden Gardens, Kolkata": "Eden Gardens",
    "Himachal Pradesh Cricket Association Stadium": "Himachal Pradesh Cricket␣
 ↪Association Stadium",
    "Himachal Pradesh Cricket Association Stadium, Dharamsala": "Himachal␣
 ↪Pradesh Cricket Association Stadium",
    "M Chinnaswamy Stadium": "M Chinnaswamy Stadium",
    "M Chinnaswamy Stadium, Bengaluru": "M Chinnaswamy Stadium",
    "M Chinnaswamy Stadium, Bangalore": "M Chinnaswamy Stadium",
    "M.Chinnaswamy Stadium": "M Chinnaswamy Stadium",
    "M.Chinnaswamy Stadium, Bengaluru": "M Chinnaswamy Stadium",
    "M.Chinnaswamy Stadium, Bangalore": "M Chinnaswamy Stadium",
    "MA Chidambaram Stadium": "MA Chidambaram Stadium",
```

```
    "MA Chidambaram Stadium, Chennai": "MA Chidambaram Stadium",
    "MA Chidambaram Stadium, Chepauk": "MA Chidambaram Stadium",
    "MA Chidambaram Stadium, Chepauk, Chennai": "MA Chidambaram Stadium",
    "Narendra Modi Stadium": "Narendra Modi Stadium",
    "Narendra Modi Stadium, Ahmedabad": "Narendra Modi Stadium",
    "Punjab Cricket Association IS Bindra Stadium": "Punjab Cricket Association␣
 ↪IS Bindra Stadium",
    "Punjab Cricket Association IS Bindra Stadium, Mohali": "Punjab Cricket␣
 ↪Association IS Bindra Stadium",
    "Punjab Cricket Association Stadium, Mohali": "Punjab Cricket Association␣
 ↪IS Bindra Stadium",
    "Rajiv Gandhi International Stadium": "Rajiv Gandhi International Stadium",
    "Rajiv Gandhi International Stadium, Hyderabad": "Rajiv Gandhi␣
 ↪International Stadium",
    "Rajiv Gandhi International Stadium, Uppal": "Rajiv Gandhi International␣
 ↪Stadium",
    "Sawai Mansingh Stadium": "Sawai Mansingh Stadium",
    "Sawai Mansingh Stadium, Jaipur": "Sawai Mansingh Stadium",
    "Wankhede Stadium": "Wankhede Stadium",
    "Wankhede Stadium, Mumbai": "Wankhede Stadium"
}
```

[ ]:
```
venue_mapping_kebab = {
    "arun-jaitley-stadium": "Arun Jaitley Stadium",
    "arun-jaitley-stadium-delhi": "Arun Jaitley Stadium",
    "feroz-shah-kotla": "Arun Jaitley Stadium",
    "barsapara-cricket-stadium": "Barsapara Cricket Stadium",
    "barsapara-cricket-stadium-guwahati": "Barsapara Cricket Stadium",
    "bharat-ratna-shri-atal-bihari-vajpayee-ekana-cricket-stadium": "Bharat␣
 ↪Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
    "bharat-ratna-shri-atal-bihari-vajpayee-ekana-cricket-stadium-lucknow":␣
 ↪"Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
    "eden-gardens": "Eden Gardens",
    "eden-gardens-kolkata": "Eden Gardens",
    "himachal-pradesh-cricket-association-stadium": "Himachal Pradesh Cricket␣
 ↪Association Stadium",
    "himachal-pradesh-cricket-association-stadium-dharamsala": "Himachal␣
 ↪Pradesh Cricket Association Stadium",
    "m-chinnaswamy-stadium": "M Chinnaswamy Stadium",
    "m-chinnaswamy-stadium-bengaluru": "M Chinnaswamy Stadium",
    "m-chinnaswamy-stadium-bangalore": "M Chinnaswamy Stadium",
    "mchinnaswamy-stadium": "M Chinnaswamy Stadium",
    "mchinnaswamy-stadium-bengaluru": "M Chinnaswamy Stadium",
    "mchinnaswamy-stadium-bangalore": "M Chinnaswamy Stadium",
    "ma-chidambaram-stadium": "MA Chidambaram Stadium",
    "ma-chidambaram-stadium-chennai": "MA Chidambaram Stadium",
```

```
    "ma-chidambaram-stadium-chepauk": "MA Chidambaram Stadium",
    "ma-chidambaram-stadium-chepauk-chennai": "MA Chidambaram Stadium",
    "narendra-modi-stadium": "Narendra Modi Stadium",
    "narendra-modi-stadium-ahmedabad": "Narendra Modi Stadium",
    "punjab-cricket-association-is-bindra-stadium": "Punjab Cricket Association␣
↪IS Bindra Stadium",
    "punjab-cricket-association-is-bindra-stadium-mohali": "Punjab Cricket␣
↪Association IS Bindra Stadium",
    "punjab-cricket-association-stadium-mohali": "Punjab Cricket Association IS␣
↪Bindra Stadium",
    "rajiv-gandhi-international-stadium": "Rajiv Gandhi International Stadium",
    "rajiv-gandhi-international-stadium-hyderabad": "Rajiv Gandhi International␣
↪Stadium",
    "rajiv-gandhi-international-stadium-uppal": "Rajiv Gandhi International␣
↪Stadium",
    "sawai-mansingh-stadium": "Sawai Mansingh Stadium",
    "sawai-mansingh-stadium-jaipur": "Sawai Mansingh Stadium",
    "wankhede-stadium": "Wankhede Stadium",
    "wankhede-stadium-mumbai": "Wankhede Stadium"
}
```

[ ]:
```
venue_mapping_tags = {
    "delhi": "Arun Jaitley Stadium",
    "arun jaitley": "Arun Jaitley Stadium",
    "guwahati": "Barsapara Cricket Stadium",
    "barsapara": "Barsapara Cricket Stadium",
    "bhupen hazarika": "Barsapara Cricket Stadium",
    "assam cricket association stadium": "Barsapara Cricket Stadium",
    "lucknow": "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
    "ekana": "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
    "atal bihari": "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket␣
↪Stadium",
    "kolkata": "Eden Gardens",
    "eden gardens": "Eden Gardens",
    "dharamsala": "Himachal Pradesh Cricket Association Stadium",
    "himachal pradesh": "Himachal Pradesh Cricket Association Stadium",
    "bengaluru": "M Chinnaswamy Stadium",
    "bengalore": "M Chinnaswamy Stadium",
    "chinnaswamy": "M Chinnaswamy Stadium",
    "chennai": "MA Chidambaram Stadium",
    "chepauk": "MA Chidambaram Stadium",
    "chidambaram": "MA Chidambaram Stadium",
    "ahmedabad": "Narendra Modi Stadium",
    "narendra modi": "Narendra Modi Stadium",
    "mohali": "Punjab Cricket Association IS Bindra Stadium",
    "punjab cricket association": "Punjab Cricket Association IS Bindra␣
↪Stadium",
```

```
    "is bindra": "Punjab Cricket Association IS Bindra Stadium",
    "hyderabad": "Rajiv Gandhi International Stadium",
    "rajiv gandhi": "Rajiv Gandhi International Stadium",
    "jaipur": "Sawai Mansingh Stadium",
    "sawai mansingh": "Sawai Mansingh Stadium",
    "mumbai": "Wankhede Stadium",
    "wankhede": "Wankhede Stadium"
}
```

[ ]:
```
# venues in `matches_result` where ipl23 isn't happening (collectively␣
↪considered as 'Other' venue after mapping)
# np.setdiff1d(matches_result.venue.unique(), list(venue_mapping_normal.keys()))
```

-

2.4 Teams Mapping

[ ]:
```
set(matches_result['team_1'].unique()) == set(
    matches_result['team_2'].unique()) == set(ball_by_ball['batting_team'].
↪unique())
```

[ ]:
```
True
```

[ ]:
```
team_mapping = {
    'Rajasthan Royals': 'Rajasthan Royals',
    'Gujarat Titans': 'Gujarat Titans',
    'Royal Challengers Bangalore': 'Royal Challengers Bangalore',
    'Lucknow Super Giants': 'Lucknow Super Giants',
    'Sunrisers Hyderabad': 'Sunrisers Hyderabad',
    'Mumbai Indians': 'Mumbai Indians',
    'Chennai Super Kings': 'Chennai Super Kings',
    'Kolkata Knight Riders': 'Kolkata Knight Riders',
    'Kings XI Punjab': 'Punjab Kings',
    'Punjab Kings': 'Punjab Kings',
    'Delhi Daredevils': 'Delhi Capitals',
    'Delhi Capitals': 'Delhi Capitals',
}
```

[ ]:
```
# unused keys of `team_mapping` (teams which are listed in `team_mapping`, but␣
↪not present in `matches_result`)
# [] before mapping, ['Delhi Daredevils', 'Kings XI Punjab'] after mapping
np.setdiff1d(
    list(team_mapping.keys()), matches_result['team_1'].unique()
)
```

[ ]:
```
array([], dtype='<U27')
```

[ ]:
```
# old ipl teams (collectively considered as 'Other' team after mapping)
np.setdiff1d(
    matches_result['team_1'].unique(), list(team_mapping.keys())
)
```

[ ]:
```
array(['Deccan Chargers', 'Gujarat Lions', 'Kochi Tuskers Kerala',
       'Pune Warriors', 'Rising Pune Supergiant',
       'Rising Pune Supergiants'], dtype=object)
```

-

2.5 Apply Venues/Teams Mapping [in matches_result, ball_by_ball]

[ ]:
```
matches_result.venue = matches_result.venue.map(
    venue_mapping_normal).fillna('Other')

matches_result.team_1 = matches_result.team_1.map(team_mapping).fillna('Other')
matches_result.team_2 = matches_result.team_2.map(team_mapping).fillna('Other')

ball_by_ball.batting_team = ball_by_ball.batting_team.map(
    team_mapping).fillna('Other')
```

[ ]:
```
# matches_result.shape
# matches_result.venue[matches_result.venue == 'Other'].shape
# matches_result.team_1[matches_result.team_1 == 'Other'].shape
# matches_result.team_2[matches_result.team_2 == 'Other'].shape
```

[ ]:
```
# ball_by_ball.shape
# ball_by_ball.batting_team[ball_by_ball.batting_team == 'Other'].shape
```

-

2.6 list_of_ipl23_[venues, teams, players]

[ ]:
```
list_of_ipl23_venues = ['Arun Jaitley Stadium', 'Barsapara Cricket Stadium',
                        'Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket␣
 ↪Stadium',
                        'Eden Gardens', 'Himachal Pradesh Cricket Association␣
 ↪Stadium',
                        'M Chinnaswamy Stadium', 'MA Chidambaram Stadium',
                        'Narendra Modi Stadium',
                        'Punjab Cricket Association IS Bindra Stadium',
                        'Rajiv Gandhi International Stadium', 'Sawai Mansingh␣
 ↪Stadium',
                        'Wankhede Stadium']

list_of_ipl23_teams = ['Chennai Super Kings', 'Delhi Capitals', 'Gujarat␣
 ↪Titans',
```

```
                    'Kolkata Knight Riders', 'Lucknow Super Giants', 'Mumbai␣
 ↪Indians',
                    'Punjab Kings', 'Rajasthan Royals', 'Royal Challengers␣
 ↪Bangalore',
                    'Sunrisers Hyderabad']

list_of_ipl23_players = data__ipl_2023_players[data__ipl_2023_players.team.
 ↪notnull()].player.values
```

[ ]:
```
# True: list_of_ipl23_venues  venue_mapping_normal
np.array_equal(np.unique(
    list(venue_mapping_normal.values())
), list_of_ipl23_venues)

# True: list_of_ipl23_teams  team_mapping
np.array_equal(np.unique(
    list(team_mapping.values())
), list_of_ipl23_teams)
```

[ ]:
```
True
```

[ ]:
```
True
```

[ ]:
```
# ipl23 venues which are not present in `matches_result`
# ['Barsapara Cricket Stadium', 'Bharat Ratna Shri Atal Bihari Vajpayee Ekana␣
 ↪Cricket Stadium']
np.setdiff1d(
    list(venue_mapping_normal.values()), matches_result.venue.unique()
)
```

[ ]:
```
array(['Barsapara Cricket Stadium',
       'Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium'],
      dtype='<U60')
```

-

2.7 Select first 6 overs, Select innings 1 & 2, Map innings (1,2) to (0,1) [in `ball_by_ball`]

[ ]:
```
ball_by_ball = ball_by_ball.loc[(
    ball_by_ball.overs <= 5) & (ball_by_ball.innings <= 2)]
ball_by_ball.loc[ball_by_ball.innings == 1, 'innings'] = 0
ball_by_ball.loc[ball_by_ball.innings == 2, 'innings'] = 1
```

[ ]:
```
# ball_by_ball.shape
# ball_by_ball.innings.unique()
# ball_by_ball.overs.unique()
```

-

2.8 Grouping (Integration of `ball_by_ball` and `matches_result`) [create `data`]

[ ]:
```
data = ball_by_ball.groupby(['match_id', 'innings', 'batting_team']).agg({
    'total_run': 'sum',
    'batter': 'unique',
    'bowler': 'unique',
}).reset_index().rename(columns={
    'batter': 'batsmen', 'bowler': 'bowlers', 'total_run': 'total_runs'
}).merge(matches_result, on=['match_id'])
```

[ ]:
```
mask = data['batting_team'] == data['team_1']
data.loc[mask, 'bowling_team'] = data['team_2']
data.loc[~mask, 'bowling_team'] = data['team_1']
```

[ ]:
```
# match_id == 829763, data for one innings is missing
# data.query('match_id == 829763')

# match_id == 829813, total_runs for one innings is 2 (probably a mistake in␣
 ↪data entry)
# data.query('match_id == 829813')

data = data.drop(data[
    (data['match_id'] == 829763) | (data['match_id'] == 829813)
].index)
```

[ ]:
```
# get count of batsmen & bowlers for each innings
data['count_batsmen'] = [len(x) for x in data['batsmen']]
data['count_bowlers'] = [len(x) for x in data['bowlers']]
```

[ ]:
```
data = data[
    ['match_id', 'venue', 'innings', 'batting_team', 'bowling_team', 'batsmen',␣
 ↪'count_batsmen', 'bowlers', 'count_bowlers', 'total_runs']
]
```

[ ]:
```
data
```

[ ]:
```
      match_id                                     venue  innings  \
0       335982                     M Chinnaswamy Stadium        0
1       335982                     M Chinnaswamy Stadium        1
2       335983  Punjab Cricket Association IS Bindra Stadium        0
3       335983  Punjab Cricket Association IS Bindra Stadium        1
4       335984                       Arun Jaitley Stadium        0
…          …                                          …      …
1893   1312198                              Eden Gardens        1
```

```
1894   1312199                        Narendra Modi Stadium        0
1895   1312199                        Narendra Modi Stadium        1
1896   1312200                        Narendra Modi Stadium        0
1897   1312200                        Narendra Modi Stadium        1

                    batting_team                 bowling_team  \
0           Kolkata Knight Riders  Royal Challengers Bangalore
1     Royal Challengers Bangalore        Kolkata Knight Riders
2             Chennai Super Kings                Punjab Kings
3                   Punjab Kings          Chennai Super Kings
4                Rajasthan Royals               Delhi Capitals
…                             …                            …
1893         Lucknow Super Giants  Royal Challengers Bangalore
1894  Royal Challengers Bangalore             Rajasthan Royals
1895             Rajasthan Royals  Royal Challengers Bangalore
1896             Rajasthan Royals               Gujarat Titans
1897               Gujarat Titans             Rajasthan Royals

                                              batsmen  count_batsmen  \
0                  [SC Ganguly, BB McCullum, RT Ponting]              3
1     [R Dravid, W Jaffer, V Kohli, JH Kallis, CL Wh…              6
2                       [PA Patel, ML Hayden, MEK Hussey]              3
3                                  [K Goel, JR Hopes]              2
4                [T Kohli, YK Pathan, SR Watson, M Kaif]              4
…                                                    …            …
1893           [Q de Kock, KL Rahul, M Vohra, DJ Hooda]              4
1894                [V Kohli, F du Plessis, RM Patidar]              3
1895               [YBK Jaiswal, JC Buttler, SV Samson]              3
1896               [YBK Jaiswal, JC Buttler, SV Samson]              3
1897         [WP Saha, Shubman Gill, MS Wade, HH Pandya]              4

                                              bowlers  count_bowlers  \
0                           [P Kumar, Z Khan, AA Noffke]              3
1                        [AB Dinda, I Sharma, AB Agarkar]              3
2                          [B Lee, S Sreesanth, JR Hopes]              3
3                                    [JDP Oram, MS Gony]              2
4                    [GD McGrath, B Geeves, MF Maharoof]              3
…                                                    …            …
1893      [Mohammed Siraj, JR Hazlewood, Shahbaz Ahmed]              3
1894                         [TA Boult, M Prasidh Krishna]              2
1895  [Mohammed Siraj, JR Hazlewood, GJ Maxwell, Sha…              4
1896  [Mohammed Shami, Yash Dayal, LH Ferguson, Rash…              4
1897            [TA Boult, M Prasidh Krishna, YS Chahal]              3

      total_runs
0             61
1             26
```

```
2              53
3              63
4              40
…             …
1893           62
1894           46
1895           67
1896           44
1897           31

[1895 rows x 10 columns]
```

3 Create a seperate dataset: data_extended, add batted__<player>, bowled__<player>, opener_batsman__<player>

<player> should be in top 4 batsmen/bowlers of respective innings and must present in all_ipl23_players list

[ ]:
```
batted__ = []
opener_batsman__ = []
for x in data.batsmen:
    batted__.append(x[0:4])
    opener_batsman__.append(x[0])

bowled__ = []
for x in data.bowlers:
    bowled__.append(x[0:4])

batted__ = np.unique(np.concatenate(batted__))
batted__ = [f'batted__{player}' for player in batted__ if player in␣
 ↪list_of_ipl23_players]

opener_batsman__ = np.unique(opener_batsman__)
opener_batsman__ = [f'opener_batsman__{player}' for player in opener_batsman__␣
 ↪if player in list_of_ipl23_players]

bowled__ = np.unique(np.concatenate(bowled__))
bowled__ = [f'bowled__{player}' for player in bowled__ if player in␣
 ↪list_of_ipl23_players]
```

[ ]:
```
batted__df = pd.DataFrame(0, index=data.index, columns=batted__)
bowled__df = pd.DataFrame(0, index=data.index, columns=bowled__)
opener_batsman__df = pd.DataFrame(0, index=data.index, columns=opener_batsman__)
data_extended = pd.concat([data.copy(), opener_batsman__df, batted__df,␣
 ↪bowled__df], axis=1)
```

```
len(batted__), len(opener_batsman__), len(bowled__)
data.shape
data_extended.shape
```

[ ]: (90, 45, 109)

[ ]: (1895, 10)

[ ]: (1895, 254)

[ ]:
```
data_extended = data_extended.drop(columns=['match_id', 'batsmen', 'bowlers'])
data = data.drop(columns=['match_id', 'batsmen', 'bowlers'])
```

4 Final training datasets: data, data_extended

[ ]:
```
data_extended.head()
data.head()
```

[ ]:
```
                                     venue  innings  \
0                    M Chinnaswamy Stadium        0
1                    M Chinnaswamy Stadium        1
2  Punjab Cricket Association IS Bindra Stadium        0
3  Punjab Cricket Association IS Bindra Stadium        1
4                        Arun Jaitley Stadium        0

                 batting_team                bowling_team  count_batsmen  \
0       Kolkata Knight Riders  Royal Challengers Bangalore              3
1  Royal Challengers Bangalore        Kolkata Knight Riders              6
2         Chennai Super Kings                Punjab Kings              3
3                Punjab Kings          Chennai Super Kings              2
4             Rajasthan Royals               Delhi Capitals              4

   count_bowlers  total_runs  opener_batsman__AM Rahane  \
0              3          61                          0
1              3          26                          0
2              3          53                          0
3              2          63                          0
4              3          40                          0

   opener_batsman__AT Rayudu  opener_batsman__Abhishek Sharma  …  \
0                          0                                0  …
1                          0                                0  …
2                          0                                0  …
3                          0                                0  …
4                          0                                0  …

   bowled__TG Southee  bowled__TU Deshpande  bowled__Tilak Varma  \
```

```
0                    0                    0                    0
1                    0                    0                    0
2                    0                    0                    0
3                    0                    0                    0
4                    0                    0                    0

   bowled__UT Yadav  bowled__Umran Malik  bowled__V Shankar  bowled__VG Arora  \
0                 0                    0                  0                 0
1                 0                    0                  0                 0
2                 0                    0                  0                 0
3                 0                    0                  0                 0
4                 0                    0                  0                 0

   bowled__Washington Sundar  bowled__YS Chahal  bowled__Yash Dayal
0                          0                  0                   0
1                          0                  0                   0
2                          0                  0                   0
3                          0                  0                   0
4                          0                  0                   0

[5 rows x 251 columns]
```

[ ]:
```
                                    venue  innings  \
0                        M Chinnaswamy Stadium        0
1                        M Chinnaswamy Stadium        1
2  Punjab Cricket Association IS Bindra Stadium        0
3  Punjab Cricket Association IS Bindra Stadium        1
4                          Arun Jaitley Stadium        0

                 batting_team                bowling_team  count_batsmen  \
0       Kolkata Knight Riders  Royal Challengers Bangalore              3
1  Royal Challengers Bangalore       Kolkata Knight Riders              6
2         Chennai Super Kings                Punjab Kings              3
3                Punjab Kings         Chennai Super Kings              2
4            Rajasthan Royals              Delhi Capitals              4

   count_bowlers  total_runs
0              3          61
1              3          26
2              3          53
3              2          63
4              3          40
```

[ ]:
```
# data.groupby(['count_batsmen']).total_runs.describe()[['count', 'mean',␣
↪'75%']].sort_values(by='75%')
# data.groupby(['count_bowlers']).total_runs.describe()[['count', 'mean',␣
↪'75%']].sort_values(by='75%')
```

5 Data preprocessing

-

5.1 Encoding of categorical inputs and feature scaling

[ ]:
```
X_data = data.drop(columns=['total_runs'])
y_data = data["total_runs"]

X_data_extended = data_extended.drop(columns=['total_runs'])
y_data_extended = data_extended["total_runs"]
```

Normalization scales the data to a range of 0 to 1, while standardization scales the data to have a mean of 0 and a standard deviation of 1.

[ ]:
```
venues_categories = list_of_ipl23_venues + ['Other']
teams_categories = list_of_ipl23_teams + ['Other']

preprocessor__onehot_scaler = ColumnTransformer([
    ("onehot", OneHotEncoder(sparse_output=False, drop='first', categories=[
        venues_categories, teams_categories, teams_categories
    ]), ["venue", "batting_team", "bowling_team"]),

    ("scaler", StandardScaler(), ["count_batsmen", "count_bowlers"])
], remainder='passthrough')

X_data_preprocessed_by_preprocessor__onehot_scaler =␣
 ↪preprocessor__onehot_scaler.fit_transform(X_data)
X_data_extended_preprocessed_by_preprocessor__onehot_scaler =␣
 ↪preprocessor__onehot_scaler.fit_transform(X_data_extended)
```

[ ]:
```
# 13 venues_categories + 11 batting_team + 11 bowling_team + (innings,␣
 ↪count_batsmen, count_bowlers) - (3 OneHotEncoder drop='first') = 35
len(venues_categories), len(teams_categories)
X_data_preprocessed_by_preprocessor__onehot_scaler.shape

# 90 batted__ + 45 opener_batsman__ + 109 bowled__ + 35 = 279 (251 before␣
 ↪OneHotEncoder -> 251 - 7 + 13 + 11 + 11 = 279)
len(batted__), len(opener_batsman__), len(bowled__)
X_data_extended_preprocessed_by_preprocessor__onehot_scaler.shape
```

[ ]:
```
(13, 11)
```

[ ]:
```
(1895, 35)
```

[ ]:
```
(90, 45, 109)
```

[ ]:
```
(1895, 279)
```

-

5.2 SelectKBest & PCA

6 Evaluate

[ ]:
```
models = {
    'Linear Regression': LinearRegression(),

    'Ridge Regression': Ridge(alpha=1),

    'Lasso Regression': Lasso(alpha=1),

    'Elastic Net Regression': ElasticNet(alpha=1, l1_ratio=0.5),

    'Support Vector Regression': SVR(C=1, epsilon=0.1, kernel='rbf'),

    'Decision Tree Regression': DecisionTreeRegressor(max_depth=None,␣
 ↪min_samples_split=2, min_samples_leaf=1),

    'Random Forest Regression': RandomForestRegressor(n_estimators=100,␣
 ↪max_depth=None, min_samples_split=2, min_samples_leaf=1),

    'Gradient Boosting Regression': GradientBoostingRegressor(n_estimators=100,␣
 ↪learning_rate=0.1, max_depth=3, min_samples_split=2, min_samples_leaf=1),

    'Neural Network Regression': MLPRegressor(hidden_layer_sizes=(100,),␣
 ↪activation='relu', solver='adam', alpha=0.0001, learning_rate='constant',␣
 ↪max_iter=100)
}
```

[ ]:
```
def get_test_results(y_test, y_pred):
    y_pred = np.round(y_pred).astype(int)

    mae = mean_absolute_error(y_test, y_pred)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    r2 = r2_score(y_test, y_pred)
    total_absolute_error = np.abs(y_test - y_pred).sum()

    return {
        'MAE': mae,
        'RMSE': rmse,
        'R2': r2,
        'Total Absolute Error': total_absolute_error
    }
```

[ ]:
```python
from sklearn.model_selection import cross_val_predict

def evaluate(X_preprocessed, y, models, do_cross_val_predict=False):
    if do_cross_val_predict == False:
        X_train, X_test, y_train, y_test = train_test_split(X_preprocessed, y,␣
 ↪test_size=0.2)

    results = []
    for model_name, model in models.items():
        if do_cross_val_predict == False:
            y_pred = model.fit(X_train, y_train).predict(X_test)
        else:
            y_pred = cross_val_predict(model, X_preprocessed, y, cv=10)

        if do_cross_val_predict:
            y_test = y

        results.append({
            'Model': model_name, **get_test_results(y_test, y_pred),
        })

    return pd.DataFrame(results)
```

[ ]:
```python
evaluate(X_preprocessed, y, models, True)
```

```
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
```

```
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:686:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and
the optimization hasn't converged yet.
  warnings.warn(
```

[ ]:

```
                         Model        MAE       RMSE        R2  \
0            Linear Regression   8.309235  10.651049  0.203177
1             Ridge Regression   8.311346  10.646788  0.203814
2             Lasso Regression   8.410026  10.760256  0.186753
3       Elastic Net Regression   8.520317  10.868317  0.170337
4     Support Vector Regression   8.311873  10.698088  0.196123
5      Decision Tree Regression  11.433245  14.622713 -0.501873
6      Random Forest Regression   9.096570  11.633724  0.049363
7   Gradient Boosting Regression   8.268602  10.630841  0.206198
8     Neural Network Regression   9.509763  12.144273 -0.035906

   Total Absolute Error
0                 15746
1                 15750
2                 15937
3                 16146
4                 15751
5                 21666
6                 17238
```

```
7                    15669
8                    18021
```

[ ]:
```
# X = X_preprocessed

# # Define the regression models
# models = [
#     ('Linear Regression', LinearRegression(), param_grids['Linear␣
 ↪Regression']),
#     ('Ridge Regression', Ridge(), param_grids['Ridge Regression']),
#     ('Lasso Regression', Lasso(), param_grids['Lasso Regression']),
#     ('Elastic Net Regression', ElasticNet(), param_grids['Elastic Net␣
 ↪Regression']),
#     ('Support Vector Regression', SVR(), param_grids['Support Vector␣
 ↪Regression']),
#     ('Decision Tree Regression', DecisionTreeRegressor(),␣
 ↪param_grids['Decision Tree Regression']),
#     ('Random Forest Regression', RandomForestRegressor(), param_grids['Random␣
 ↪Forest Regression']),
#     ('Gradient Boosting Regression', GradientBoostingRegressor(),␣
 ↪param_grids['Gradient Boosting Regression']),
#     ('Neural Network Regression', MLPRegressor(), param_grids['Neural Network␣
 ↪Regression'])
# ]

# # Perform grid search for each model and print the best parameters and mean␣
 ↪squared error
# best_model = None
# best_mse = np.inf
# for name, model, param_grid in models:
#     grid_search = GridSearchCV(estimator=model, param_grid=param_grid,␣
 ↪scoring='neg_mean_squared_error', cv=5, n_jobs=-1)
#     grid_search.fit(X, y)
#     mse = -grid_search.best_score_
#     if mse < best_mse:
#         best_model = name
#         best_params = grid_search.best_params_
#         best_mse = mse
#     print(f"{name}: Best parameters = {grid_search.best_params_}, Best mean␣
 ↪squared error = {mse:.2f}")

# # Print the best model and its corresponding best parameters and mean squared␣
 ↪error
# print(f"\nBest model: {best_model}")
# print(f"Best parameters: {best_params}")
# print(f"Best mean squared error: {best_mse:.2f}")
```

[ ]:
```python
# Create a list of regression models
models0 = [
    ('Linear Regression', LinearRegression()),
    ('Ridge Regression', Ridge(alpha=1.0)),
    ('Lasso Regression', Lasso(alpha=0.1)),
    ('Elastic Net Regression', ElasticNet(alpha=0.1, l1_ratio=0.5)),
    ('Support Vector Regression', SVR(kernel='linear', C=1.0)),
    ('Decision Tree Regression', DecisionTreeRegressor(random_state=42)),
    ('Random Forest Regression', RandomForestRegressor(
        n_estimators=100, random_state=42)),
    ('Gradient Boosting Regression', GradientBoostingRegressor(
        n_estimators=100, random_state=42)),
    ('Neural Network Regression', MLPRegressor(
        hidden_layer_sizes=(50, 50), max_iter=1000, random_state=42))
]

# Train and evaluate each model
for name, model in models0:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)

    # Calculate the mean absolute error (MAE)
    mae = mean_absolute_error(y_test, y_pred)

    print(f"{name}: mae = {mae:.2f}")
```

[ ]:
```
LinearRegression()
```

```
Linear Regression: mae = 8.30
```

[ ]:
```
Ridge()
```

```
Ridge Regression: mae = 8.22
```

[ ]:
```
Lasso(alpha=0.1)
```

```
Lasso Regression: mae = 8.15
```

[ ]:
```
ElasticNet(alpha=0.1)
```

```
Elastic Net Regression: mae = 8.16
```

[ ]:
```
SVR(kernel='linear')
```

```
Support Vector Regression: mae = 8.21
```

[ ]:
```
DecisionTreeRegressor(random_state=42)
```

```
Decision Tree Regression: mae = 11.52
```

[ ]: RandomForestRegressor(random_state=42)

Random Forest Regression: mae = 8.66

[ ]: GradientBoostingRegressor(random_state=42)

Gradient Boosting Regression: mae = 8.08

C:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\neural_network_multilayer_perceptron.py:686: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (1000) reached and the optimization hasn't converged yet.
warnings.warn(

[ ]: MLPRegressor(hidden_layer_sizes=(50, 50), max_iter=1000, random_state=42)

Neural Network Regression: mae = 9.83

-

6.1 Models

[ ]:
```
models = {}
```

[ ]:
```
from sklearn.ensemble import AdaBoostRegressor
models['AdaBoostRegressor'] = regressor = AdaBoostRegressor(
    learning_rate=1, loss='exponential', n_estimators=100
)
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

[ ]: AdaBoostRegressor(learning_rate=1, loss='exponential', n_estimators=100)

C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pandas doesn't allow columns to be created via a new attribute name - see https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pandas doesn't allow columns to be created via a new attribute name - see https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
results_df.Predicted = y_pred

[ ]: (Empty DataFrame
Columns: []
Index: [],
9.469656992084433,
11.806151967275635,
-0.041224385997210344,
3589)

[ ]:
```
from sklearn.linear_model import LinearRegression
models['LinearRegression'] = regressor = LinearRegression()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

[ ]:
```
LinearRegression()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 8.313984168865435,
 10.285758784508173,
 0.20968492995380883,
 3151)
```

[ ]:
```
from sklearn.tree import DecisionTreeRegressor
models['DecisionTreeRegressor'] = regressor = DecisionTreeRegressor()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

[ ]:
```
DecisionTreeRegressor()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 11.522427440633246,
 14.781879372516919,
 -0.6322508391085426,
 4367)
```

[ ]:
```
from sklearn.ensemble import RandomForestRegressor
models['RandomForestRegressor'] = regressor = RandomForestRegressor()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

[ ]:
```
RandomForestRegressor()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 8.633245382585752,
 11.007073762958935,
 0.09495255539364522,
 3272)
```

[ ]:
```
from sklearn.neighbors import KNeighborsRegressor
models['KNeighborsRegressor'] = regressor = KNeighborsRegressor()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

[ ]:
```
KNeighborsRegressor()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 8.831134564643799,
 10.969975815522103,
 0.10104297005420015,
 3347)
```

[ ]:
```
from sklearn.svm import SVR
models['SVR'] = regressor = SVR()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

[ ]:
```
SVR()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 8.131926121372032,
 10.141999719549673,
 0.2316222487796913,
 3082)
```

[ ]:
```
import xgboost as xgb
models['XGBRegressor'] = regressor = xgb.XGBRegressor()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

[ ]:
```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             n_estimators=100, n_jobs=None, num_parallel_tree=None,
             predictor=None, random_state=None, …)
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]: (Empty DataFrame
Columns: []
Index: [],
9.100263852242744,
11.651831251093443,
-0.014182156501153953,
3449)

•

7 Evaluation [using IPL-2023 dataset]

[ ]:
```
import os
files = os.listdir('./FilesUsed')
all_X = []
all_y = []
for file in files:
    if 'test_file_matchid' in file:
        match_no = file[-6:-4]

        if int(match_no) < 20:
            continue

        X_file_name = './FilesUsed/' + file
        y_file_name = './FilesUsed/' + 'test_file_labels_matchid_' + match_no +␣
 ↪'.csv'

        X = pd.read_csv(X_file_name).drop(columns=['Unnamed: 0'])
        y = pd.read_csv(y_file_name)['actual_runs']

        all_X += [X]
        all_y += [y]

        print(match_no, X_file_name, y_file_name)

X_IPL23 = pd.concat(all_X, axis=0, ignore_index=True)
y_IPL23 = pd.concat(all_y, axis=0, ignore_index=True)
```

```
20 ./FilesUsed/test_file_matchid_20.csv
./FilesUsed/test_file_labels_matchid_20.csv
21 ./FilesUsed/test_file_matchid_21.csv
./FilesUsed/test_file_labels_matchid_21.csv
22 ./FilesUsed/test_file_matchid_22.csv
./FilesUsed/test_file_labels_matchid_22.csv
23 ./FilesUsed/test_file_matchid_23.csv
./FilesUsed/test_file_labels_matchid_23.csv
24 ./FilesUsed/test_file_matchid_24.csv
```

```
./FilesUsed/test_file_labels_matchid_24.csv
25 ./FilesUsed/test_file_matchid_25.csv
./FilesUsed/test_file_labels_matchid_25.csv
26 ./FilesUsed/test_file_matchid_26.csv
./FilesUsed/test_file_labels_matchid_26.csv
27 ./FilesUsed/test_file_matchid_27.csv
./FilesUsed/test_file_labels_matchid_27.csv
28 ./FilesUsed/test_file_matchid_28.csv
./FilesUsed/test_file_labels_matchid_28.csv
29 ./FilesUsed/test_file_matchid_29.csv
./FilesUsed/test_file_labels_matchid_29.csv
30 ./FilesUsed/test_file_matchid_30.csv
./FilesUsed/test_file_labels_matchid_30.csv
31 ./FilesUsed/test_file_matchid_31.csv
./FilesUsed/test_file_labels_matchid_31.csv
32 ./FilesUsed/test_file_matchid_32.csv
./FilesUsed/test_file_labels_matchid_32.csv
33 ./FilesUsed/test_file_matchid_33.csv
./FilesUsed/test_file_labels_matchid_33.csv
```

[ ]:
```
len(all_X)
```

[ ]:
```
14
```

[ ]:
```
X_IPL23.innings = X_IPL23.innings.replace({1: 0, 2: 1})

# get count of batsmen & bowlers for each innings
X_IPL23['count_batsmen'] = [len(x.split(",")) for x in X_IPL23['batsmen']]
X_IPL23['count_bowlers'] = [len(x.split(",")) for x in X_IPL23['bowlers']]
X_IPL23 = X_IPL23.drop(columns=['batsmen', 'bowlers'])[
    ['venue', 'innings', 'batting_team', 'bowling_team',
        'count_batsmen', 'count_bowlers']
]
```

[ ]:
```
ambiguous_venues = np.setdiff1d(
    X_IPL23.venue.unique(), list(venue_mapping_normal.keys()))
ambiguous_venues_mapping = {}
for venue in ambiguous_venues:
    venue_kebab_case = to_kebab_case(venue)
    if venue_kebab_case in venue_mapping_kebab:
        ambiguous_venues_mapping[venue] = venue_mapping_kebab[venue_kebab_case]
    else:
        venue_lower = venue.lower()
        for tag in venue_mapping_tags:
            if tag in venue_lower:
                ambiguous_venues_mapping[venue] = venue_mapping_tags[tag]

venue_mapping_final = {**venue_mapping_normal, **ambiguous_venues_mapping}
```

```
np.setdiff1d(X_IPL23.venue.unique(), list(venue_mapping_final.keys()))
```

[ ]:
```
array([], dtype=object)
```

[ ]:
```
X_IPL23.venue = X_IPL23.venue.map(venue_mapping_final).fillna('Other').replace({
    'Barsapara Cricket Stadium': 'Other',
    'Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium': 'Other'
})
```

[ ]:
```
X_IPL23
```

[ ]:

	venue	innings	batting_team	bowling_team	count_batsmen
0	M Chinnaswamy Stadium	0	Royal Challengers Bangalore	Delhi Capitals	3
1	M Chinnaswamy Stadium	1	Delhi Capitals	Royal Challengers Bangalore	3
2	Other	0	Lucknow Super Giants	Punjab Kings	2
3	Other	1	Punjab Kings	Lucknow Super Giants	4
4	Wankhede Stadium	0	Kolkata Knight Riders	Mumbai Indians	4
5	Wankhede Stadium	1			
6	Narendra Modi Stadium	0			
7	Narendra Modi Stadium	1			
8	M Chinnaswamy Stadium	0			
9	M Chinnaswamy Stadium	1			
10	Rajiv Gandhi International Stadium	0			
11	Rajiv Gandhi International Stadium	1			
12	Sawai Mansingh Stadium	0			
13	Sawai Mansingh Stadium	1			
14	Punjab Cricket Association IS Bindra Stadium	0			
15	Punjab Cricket Association IS Bindra Stadium	1			
16	Arun Jaitley Stadium	0			
17	Arun Jaitley Stadium	1			
18	MA Chidambaram Stadium	0			
19	MA Chidambaram Stadium	1			
20	Other	0			
21	Other	1			
22	Wankhede Stadium	0			
23	Wankhede Stadium	1			
24	M Chinnaswamy Stadium	0			
25	M Chinnaswamy Stadium	1			
26	Eden Gardens	0			
27	Eden Gardens	1			

5	Mumbai Indians	Kolkata Knight Riders	3
6	Gujarat Titans	Rajasthan Royals	4
7	Rajasthan Royals	Gujarat Titans	4
8	Chennai Super Kings	Royal Challengers Bangalore	3
9	Royal Challengers Bangalore	Chennai Super Kings	4
10	Mumbai Indians	Sunrisers Hyderabad	3
11	Sunrisers Hyderabad	Mumbai Indians	4
12	Lucknow Super Giants	Rajasthan Royals	2
13	Rajasthan Royals	Lucknow Super Giants	2
14	Royal Challengers Bangalore	Punjab Kings	2
15	Punjab Kings	Royal Challengers Bangalore	6
16	Kolkata Knight Riders	Delhi Capitals	5
17	Delhi Capitals	Kolkata Knight Riders	3
18	Sunrisers Hyderabad	Chennai Super Kings	3
19	Chennai Super Kings	Sunrisers Hyderabad	2
20	Gujarat Titans	Lucknow Super Giants	3
21	Lucknow Super Giants	Gujarat Titans	2
22	Punjab Kings	Mumbai Indians	3
23	Mumbai Indians	Punjab Kings	3
24	Royal Challengers Bangalore	Rajasthan Royals	4
25	Rajasthan Royals	Royal Challengers Bangalore	3
26	Chennai Super Kings	Kolkata Knight Riders	2
27	Kolkata Knight Riders	Chennai Super Kings	4

	count_bowlers
0	5
1	2
2	4
3	4
4	4
5	4
6	4
7	2
8	3
9	3
10	4
11	3
12	3
13	3
14	4
15	4
16	3
17	5
18	3
19	3
20	4
21	4

```
22              5
23              5
24              3
25              4
26              4
27              3
```

[ ]:
```
X_IPL23_preprocessed = preprocessor.transform(X_IPL23)
```

[ ]:
```
X_IPL23_preprocessed.shape
```

[ ]:
```
(28, 36)
```

[ ]:
```
X_IPL23_preprocessed[0]
```

[ ]:
```
array([ 0.        ,  0.        ,  0.        ,  1.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        1.        ,  0.        ,  0.        ,  1.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        , -0.31740491,  2.03573722,
        0.        ])
```

[ ]:
```
evaluate(models['LinearRegression'], X_IPL23_preprocessed, y_IPL23)
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 7.928571428571429,
 10.579630023236703,
 0.113857836751593,
 222.0)
```

[ ]:
```
# evaluate(models0['LinearRegression'], X_IPL23_preprocessed, y_IPL23)

# Train and evaluate each model
for name, model in models0:
    y_pred = np.round(
```

```
        model.predict(X_IPL23_preprocessed)
    ).astype(int)

    y_test = y_IPL23

    # Calculate the mean absolute error (MAE)
    mae = mean_absolute_error(y_test, y_pred)

    # Calculate the root mean squared error (RMSE)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))

    # Calculate the R-squared score
    r2 = r2_score(y_test, y_pred)

    # Calculate total absolute error
    total_absolute_error = np.abs(y_test - y_pred).sum()

    print(f"{name}: total_absolute_error = {total_absolute_error:.2f}", r2)
```

```
Linear Regression: total_absolute_error = 222.00 0.113857836751593
Ridge Regression: total_absolute_error = 213.00 0.14750522584749615
Lasso Regression: total_absolute_error = 225.00 0.09830652246357052
Elastic Net Regression: total_absolute_error = 218.00 0.1268643905197573
Support Vector Regression: total_absolute_error = 220.00 0.06579013804316003
Decision Tree Regression: total_absolute_error = 326.00 -0.6388257747886938
Random Forest Regression: total_absolute_error = 268.00 -0.12478414977733343
Gradient Boosting Regression: total_absolute_error = 239.00 0.012915669463883672
Neural Network Regression: total_absolute_error = 332.00 -0.6987690225897987
```

[ ]:
```
class ConstantRegressor:
    def __init__(self, n):
        self.n = n

    def predict(self, X):
        return np.repeat(self.n, X.shape[0])
```

[ ]:
```
evaluate(ConstantRegressor(40), X_IPL23_preprocessed, y_IPL23)
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]: (Empty DataFrame
Columns: []
Index: [],
12.178571428571429,
14.972594011345242,
-0.7748290870166721,
341.0)

[ ]:
```
evaluate(ConstantRegressor(46), X_IPL23_preprocessed, y_IPL23)
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning:
Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

[ ]: (Empty DataFrame
Columns: []
Index: [],
9.464285714285714,
11.893875975235563,
-0.11997737990649004,
265.0)