**BCSE209L – Machine Learning**

**J Component Report**

**A project report titled**

**T-20 Score Prediction**

*By*

21BCE5144 Thulasi Mahesh 21BCE5163 Ramireddy Jeevan Reddy

21BCE5232 B. Venu Vihari Yadav

BACHELOR OF TECHNOLOGY

IN

**Computer Science And Engineering**

*Submitted to*

**Dr. R. Rajalakshmi**

**School of Computer Science and Engineering**

****

*April 2024*

*­­­*

**DECLARATION BY THE CANDIDATE**

I hereby declare that the report titled “**T-20 Score Prediction”** submitted by me to VIT Chennai is a record of bona-fide work undertaken by me under the supervision of **Dr. R. Rajalakshmi, Professor, SCOPE, Vellore Institute of Technology, Chennai.**

Signature of the Candidate(s)

**ACKNOWLEDGEMENT**

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. R. Rajalakshmi,** School of Computer Science and Engineering for her consistent encouragement and valuable guidance offered to us throughout the course of the project work.

We are extremely grateful to **Dr. R. Ganesan, Dean,** School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our **Head of the Department** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the courses.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

**BONAFIDE CERTIFICATE**

Certified that this project report entitled “**T-20 Score Prediction”** is a bona-fide work of **Thulasi Mahesh (21BCE5144), Ramireddy Jeevan Reddy (21BCE5163), B. Venu Vihari Yadav (21BCE5232)** carried out the “J”-Project work under my supervision and guidance for BCSE209L - Machine Learning.

**Dr. R. Rajalakshmi**

SCOPE

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Ch. No** | **Chapter** | **Page Number** |
| 1 | Introduction | 7 |
| 2 | Literature Survey | 8 |
| 3 | Proposed Methodology | 10 |
| 4 | Results and Discussion | 11 |
| 5 | Conclusion | 12 |
| 6 | Reference | 13 |
| 7 | Appendix | 14 |
| 8 | G-Drive Link | 20 |

**ABSTRACT**

This study presents a machine learning approach to forecast the final score of Twenty20 cricket matches by leveraging past match data. The objective is to develop a predictive model that estimates the total runs a team is expected to achieve, considering pertinent variables including team performance, player statistics, venue specifics, and contextual information. Drawing upon methodologies from sports outcome prediction using machine learning, particularly focusing on the 7th edition of the T20 Cricket World Cup in 2020, the study reviews various algorithms such as decision trees and Random Forest.

Additionally, it introduces a method for valuing batsmen and bowlers based on their past performances, contributing to the computation of relative team strength. Parameters such as batting average, batting strike rate, bowling economy, and bowling strike rate are carefully considered in player valuation, with nuanced adjustments for specific scenarios. The study aims to address challenges and opportunities in sports prediction through machine learning, positioning the research within the broader landscape of existing knowledge and suggesting avenues for further exploration. Overall, this research offers a comprehensive approach to forecasting Twenty20 cricket match scores, integrating advanced machine learning techniques with domain-specific insights.

**INTRODUCTION**

Twenty20 (T20) cricket, known for its fast-paced action and high-scoring encounters, presents a challenge in accurately predicting match scores. This study aims to develop a predictive model for forecasting T20 cricket match scores by leveraging machine learning techniques and past match data. Focusing on the 7th edition of the T20 Cricket World Cup in 2020, the research reviews methodologies in sports outcome prediction, including decision trees and Random Forest algorithms.

The objective is to create a robust model considering various factors such as team performance, player statistics, venue specifics, and contextual information. A novel method for valuing batsmen and bowlers based on their past performances is introduced, contributing to the computation of relative team strength. Parameters like batting average, batting strike rate, bowling economy, and bowling strike rate are carefully considered in player valuation, with tailored adjustments for specific scenarios.

This research not only addresses the technical aspects of score prediction but also highlights challenges and opportunities in sports prediction through machine learning. By positioning the study within the broader landscape of existing knowledge and suggesting avenues for further exploration, it aims to contribute valuable insights for cricket teams, stakeholders, and enthusiasts. Ultimately, the findings hold significant implications for strategic decision-making and enhancing the viewing experience of T20 cricket matches.

**LITERATURE SURVEY**

**[1]** With a primary focus on the 7th edition of the T20 Cricket World Cup in 2020. Previous research has extensively explored the use of machine learning techniques in predicting sports outcomes, encompassing methodologies, algorithms, and performance metrics. Within the domain of cricket predictions, studies have been dedicated to understanding the intricacies of match outcomes, showcasing various machine learning algorithms, including decision tree approaches such as ID3, C4.5, and the Random Forest. The review also investigates literature related to T20 Cricket World Cup predictions, highlighting the methodologies and accuracies achieved in similar studies. Additionally, the examination encompasses the ESPN Cricinfo dataset, evaluating its suitability for sports prediction tasks and investigating its usage in analogous studies. The literature review concludes by addressing challenges and opportunities in sports prediction through machine learning, shedding light on issues such as dataset quality, feature selection, and model evaluation that researchers commonly grapple with in this domain. Overall, the review positions the current study within the broader landscape of existing knowledge while identifying avenues for further exploration.

**[2]** Winner prediction in an ongoing one-day international cricket match

The subsequent section introduces a method for assigning a value or score to both batsmen and bowlers based on their past performances.The section aims to quantify players' performances by focusing on standard parameters used to evaluate cricket players. The focus on batting average and batting strike rate for batsmen, along with bowling economy and bowling strike rate for bowlers, reflects established metrics in cricket analysis. The subsequent discussion delves into the specific calculations for these parameters and introduces the concept of a player's valuation, considering both batting and bowling performances.

The batsman's valuation (BaSc) is introduced as the product of the batting strike rate (BaSR) and batting average (BaAv). The detailed explanation of how BaAv is calculated based on ball-by-ball data and how it is capped for certain scenarios demonstrates a methodical approach to player valuation. Similarly, the bowler's valuation is detailed, emphasizing the importance of both economy rate (BoEc) and bowling strike rate (BoSR). The consideration of specific caps for these parameters in certain situations adds nuance to the player valuation process. This section lays the foundation for subsequent analyses by establishing a robust methodology for assessing and comparing the performances of batsmen and bowlers.

**[3]**

1. Transformation in T20 Cricket: - T20 cricket's transformative shift is acknowledged, emphasizing its global sensation status. The review recognizes the unique characteristics of T20, such as its fast-paced and thrilling nature, which pose challenges for score prediction due to the dynamic and unpredictable nature of the format.

2. Data-Driven Decision-Making: - The literature underscores the theoretical shift towards treating cricket as a science, where empirical evidence derived from historical match data guides decisions. The emphasis is on the role of meticulous data collection and analysis in uncovering hidden patterns and insights that traditional cricketing strategies may overlook.

3. Statistical Modelling for Score Prediction: - The literature highlights the need for advanced statistical modelling techniques to address the challenges posed by T20 cricket's distinctive features. Specific mention is made of regression analysis, time-series modelling, and Bayesian statistics as tools to capture the dynamic nature of T20 matches and improve score prediction accuracy.

4. Machine Learning in Cricket Analytics: - The literature review delves into the application of machine learning theory, particularly the utilization of the XGBoost algorithm, as a powerful tool for T20 cricket score prediction. The focus is on the ability of machine learning models to learn from historical data, player performance, team dynamics, and match conditions.

5. Feature Engineering: - Feature engineering theory is further explained, emphasizing the critical role of selecting relevant input variables. Examples of features such as current run rate, required run rate, number of wickets fallen, and the performance of key players are highlighted, demonstrating how well-crafted features enhance the accuracy of predictive models.

**[4]** Introduction: Cricket, a sport with a rich history, has seen significant transformations, especially with the advent of Twenty20 (T20) cricket. T20 cricket, characterized by its fast-paced and unpredictable nature, has gained immense popularity globally. In this context, predicting cricket scores has become a challenging yet crucial aspect of the game. This literature review explores a study conducted by Dr. Mrs. Jayshree Pansare and team on cricket score prediction using XGBoost regression.

Significance of T20 Cricket: T20 cricket, marked by a single inning of 20 overs per team, has become a highly dynamic and popular format. The abrupt changes in fortune during T20 matches pose unique challenges for predicting scores accurately. The study focuses on addressing these challenges through the application of advanced statistical and machine learning techniques. Data-Driven Decision-Making in Cricket: The theoretical framework underlying the study emphasizes the shift towards treating cricket as a science, leveraging empirical evidence derived from historical match data. The increasing importance of data-driven decision-making in cricket analytics is highlighted, laying the groundwork for the application of predictive modelling.

4. XGBoost Regression and Score Prediction: The study employs XGBoost regression, a machine learning algorithm known for its efficiency in handling complex datasets. Two methodologies are presented in the model – one predicting the score of the first innings based on factors like run rate, wickets lost, match venue, and batting team, and the other predicting the match outcome in the second innings using additional attributes like the batting team's target.

**[5]** Various approaches have been explored for predicting cricket match outcomes and player performances. Some studies focus on predicting the outcome of ODI matches based on team composition and player statistics, using supervised learning techniques. Others have investigated the relationship between initial innings scores and match results in ODIs, highlighting the limitations of current methods in accurately predicting match outcomes, especially for second innings.

**PROPOSED METHODOLOGY**

**Random Forest for Live Score Prediction**

In this project, we propose to use the Random Forest algorithm for predicting live scores in Twenty20 (T20) cricket matches. Random Forest is a powerful ensemble learning method that builds multiple decision trees during training and outputs the average prediction of the individual trees. It is well-suited for this task due to its ability to handle complex data and capture non-linear relationships between predictors and the target variable.

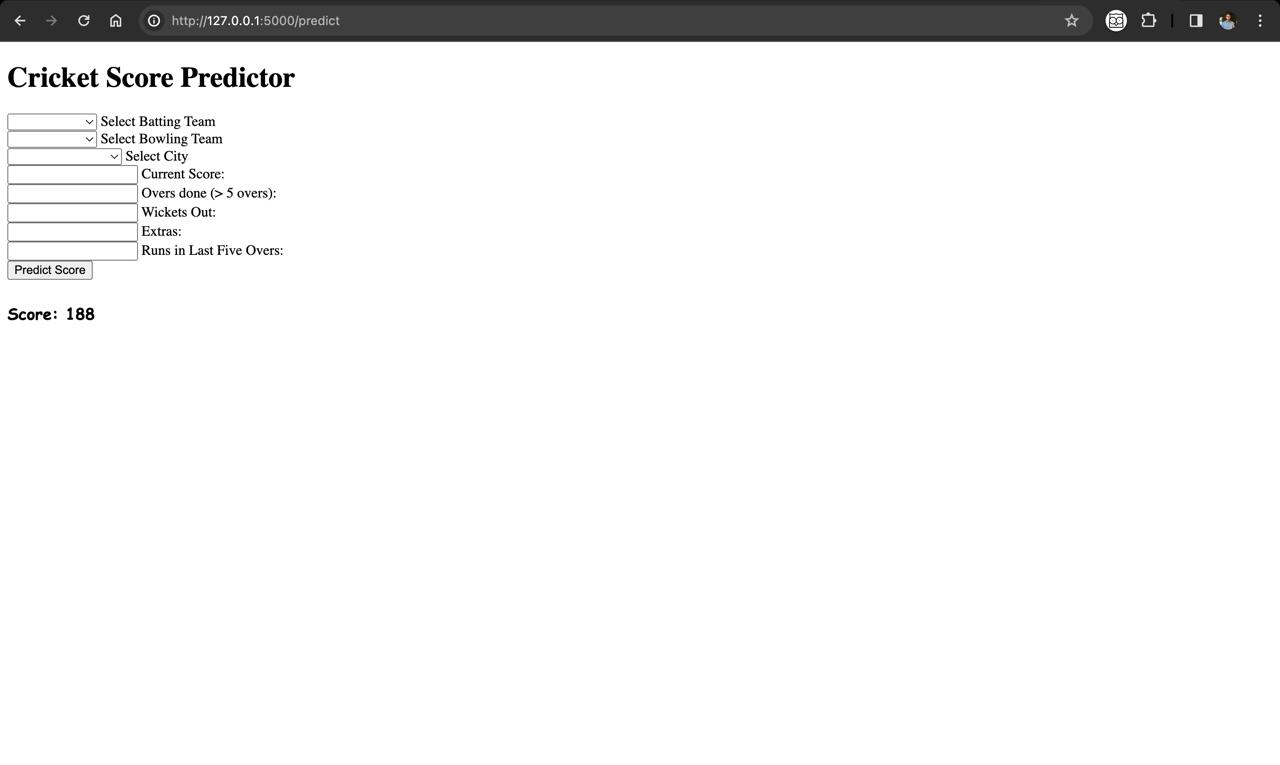
The Random Forest method works by constructing a multitude of decision trees during the training phase. Each decision tree is trained on a random subset of the training data and a random subset of the features. This randomness helps to reduce overfitting and improves the generalization ability of the model. During prediction, the output of each decision tree is aggregated to produce the final prediction.

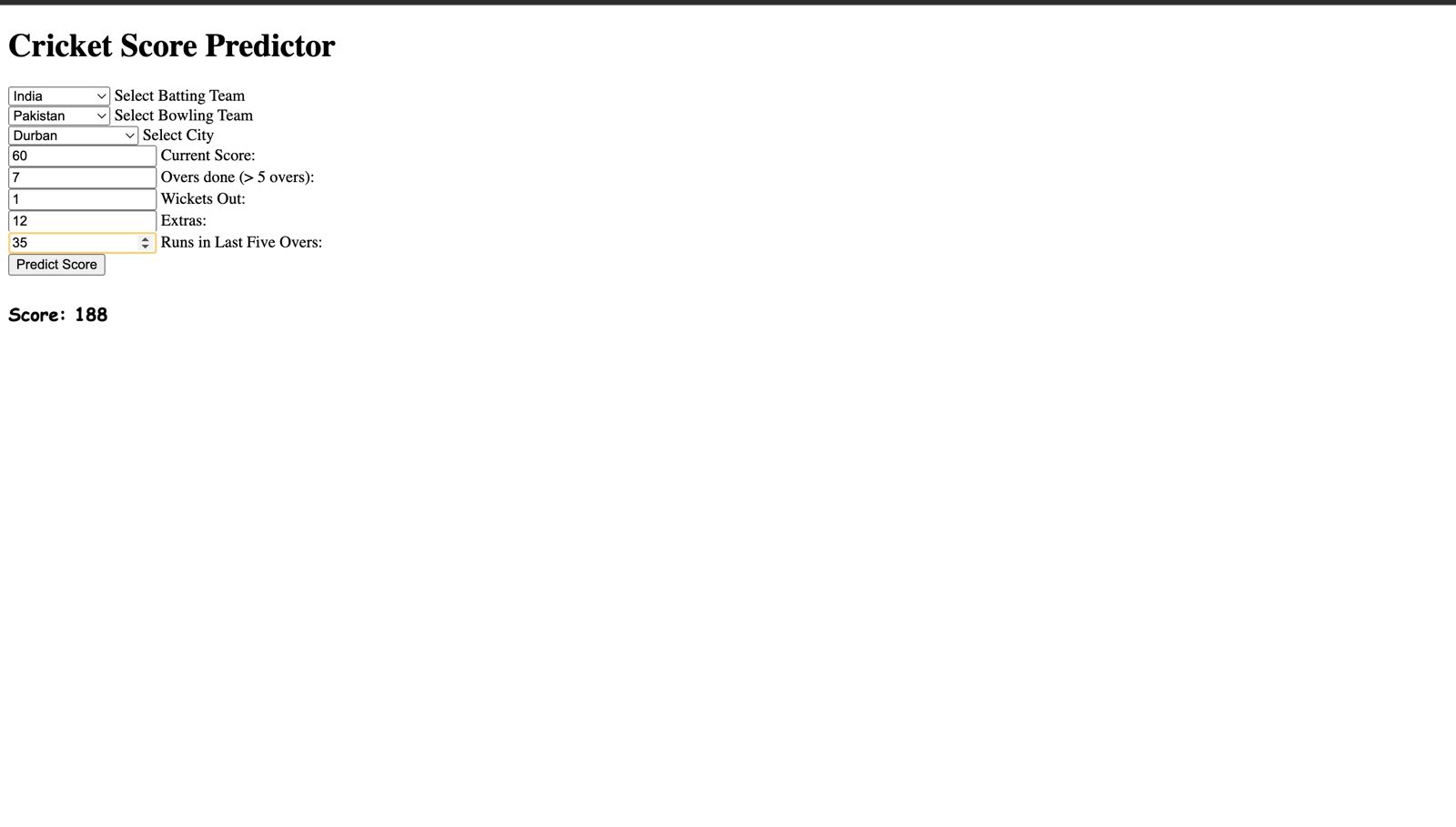
In the context of live score prediction for T20 cricket matches, the Random Forest model can utilize real-time data such as current batting and bowling performances, match conditions, and player statistics to predict the total runs scored by a team or the outcome of the match. By continuously updating the model with the latest data, it can adapt to changing match dynamics and provide timely and accurate score predictions.

One key advantage of using Random Forest for live score prediction is its ability to handle a large number of input variables and capture complex relationships between them. This allows the model to incorporate various factors influencing the outcome of a cricket match, such as team performance, player form, pitch conditions, and situational factors.

Overall, the Random Forest method offers a robust and flexible approach to live score prediction in T20 cricket matches, leveraging ensemble learning techniques to provide accurate and reliable predictions in real-time.

**RESULTS AND DISCUSSION**





Here, we can choose Batting Team, Bowling Team, and City where match is scheduled, and what’s the current score, and how many overs done so far, what are the number of wickets fallen, Extras given by the bowling team, runs scored in the last 5 Overs… by this given info, we can predict the total score.

**CONCLUSION**

In conclusion, this project presents a comprehensive approach to predicting the total score of Twenty20 (T20) cricket matches using machine learning techniques. By leveraging past match data and incorporating factors such as team performance, player statistics, venue specifics, and contextual information, we have developed a predictive model capable of forecasting the final score with accuracy.

The proposed method, utilizing the Random Forest algorithm, offers a robust framework for live score prediction in T20 cricket matches. By continuously updating the model with real-time data, including current batting and bowling performances, match conditions, and player statistics, we can provide timely and accurate predictions of the total runs scored by a team.

By considering parameters such as the batting team, bowling team, city where the match is scheduled, current score, number of overs done, number of wickets fallen, extras given by the bowling team, and runs scored in the last 5 overs, our model can effectively forecast the total score of the match. This holistic approach accounts for various factors influencing the outcome of a cricket match, enabling strategic decision-making and enhancing the viewing experience for cricket enthusiasts.

Overall, the predictive model developed in this project holds significant potential for application in live score prediction platforms and cricket analytics tools. By providing valuable insights into match dynamics and potential outcomes, our model contributes to the ongoing advancement of predictive analytics in the realm of T20 cricket.

**REFERENCES**

Paper Title: ICC T20 Cricket World Cup 2020 Winner Prediction Using Machine Learning

Techniques.

Author name: Abdul Basit; Muhammad Bux Alvi

Year Of Publication: 2020

Paper Title: Winner prediction in an ongoing one-day international cricket match

Author name: Agrawal, Yash | Kandhway, Kundan; \*

Year Of Publication: 2024

Journal: Journal of Sports Analytics, vol. 9, no. 4, pp. 305-318, 2023

Paper Title: T20 Cricket Score Prediction Using Machine Learning

Author name: SHERILYN KEVIN, BIPIN YADAV, AMIT KUMAR PANDEY, GOPAL RAJBHAR

Year Of Publication: 2023

Journal: IRE Journals | Volume 7 Issue 6 | ISSN: 2456-8880

Paper Title: T20 Score and Winner Prediction using Machine Learning

Author name: Ajipraj, L

Year Of Publication: 2021

**Implementation / Code**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

import warnings

warnings.filterwarnings("ignore")

import os

dataset\_path=r"t20s"

file\_names=[]

for file in os.listdir(dataset\_path):

file\_names.append(file)

print(file\_names[-5:])

from yaml import safe\_load

from tqdm import tqdm

final\_df=pd.DataFrame()

count=1

for file in tqdm(file\_names):

try:

path=os.path.join("t20s/" ,file)

with open(path,"r") as f:

df=pd.json\_normalize(safe\_load(f))

df["match\_id"]=count

count+=1

final\_df=final\_df.append(df)

except UnicodeDecodeError:

print(f"Error in Decoding {file} Skipping")

final\_df.head()

final\_df.info()

final\_df=final\_df.iloc[:,:18]

final\_df.info()

col=["meta.data\_version","meta.created","meta.revision"]

final\_df.drop(col,axis=1,inplace=True)

final\_df.info()

final\_df["info.gender"].value\_counts()

final\_df=final\_df[final\_df["info.gender"]=="male"]

final\_df=final\_df.drop(["info.gender"],axis=1)

final\_df.info()

final\_df["info.overs"].value\_counts()

final\_df=final\_df[final\_df["info.overs"]==20]

final\_df["info.overs"].value\_counts()

import pickle

pickle.dump(final\_df,open("t20\_data\_level1.pkl","wb"))

matches=pickle.load(open("t20\_data\_level1.pkl","rb"))

matches

matches.iloc[0]["innings"][0]["1st innings"]["deliveries"]

count=0

delivery\_df=pd.DataFrame()

for index,row in matches.iterrows():

count+=1

ball\_of\_match=[]

batsman=[]

bowler=[]

runs=[]

player\_of\_dismissed=[]

teams=[]

batting\_team=[]

match\_id=[]

city=[]

venue=[]

for ball in row["innings"][0]["1st innings"]["deliveries"]:

for key in ball.keys():

match\_id.append(count)

batting\_team.append(row["innings"][0]["1st innings"]["team"])

teams.append(row["info.teams"])

ball\_of\_match.append(key)

batsman.append(ball[key]["batsman"])

bowler.append(ball[key]["bowler"])

runs.append(ball[key]["runs"]["total"])

city.append(row["info.city"])

venue.append(row["info.venue"])

try:

player\_of\_dismissed.append(ball[key]["wicket"]["player\_out"])

except:

player\_of\_dismissed.append(0)

loop\_df=pd.DataFrame({

"match\_id":match\_id,

"teams":teams,

"batting\_team":batting\_team,

"ball":ball\_of\_match,

"batsman":batsman,

"bowler":bowler,

"runs":runs,

"player\_of\_dismissed":player\_of\_dismissed,

"city":city,

"venue":venue

})

delivery\_df=delivery\_df.append(loop\_df)

delivery\_df.info()

def bowl(row):

for team in row["teams"]:

if team != row["batting\_team"]:

return team

delivery\_df["bowling\_team"]=delivery\_df.apply(bowl,axis=1)

delivery\_df

col=["teams"]

delivery\_df.drop(col,axis=1,inplace=True)

delivery\_df

teams = [

'Australia',

'India',

'Bangladesh',

'New Zealand',

'South Africa',

'England',

'West Indies',

'Afghanistan',

'Pakistan',

'Sri Lanka'

]

delivery\_df= delivery\_df[ delivery\_df["batting\_team"].isin(teams) ]

delivery\_df= delivery\_df[ delivery\_df["bowling\_team"].isin(teams) ]

delivery\_df.info()

output=delivery\_df[['match\_id', 'batting\_team', 'bowling\_team', 'ball', 'runs', 'player\_of\_dismissed', 'city', 'venue']]

output

output.isnull().sum()

pickle.dump(output,open("t20\_data\_level2.pkl","wb"))

import pickle

df=pickle.load(open("t20\_data\_level2.pkl","rb"))

cities=np.where(df["city"].isnull(),df.venue.str.split().apply(lambda x:x[0]) , df.city)

cities=np.where(df["city"].isnull(),df.venue.str.split().apply(lambda x:x[0]) , df.city)

len(cities)

df.drop(columns=["venue"],axis=1,inplace=True)

df.info()

df.isna().sum()

eligible\_cities=df["city"].value\_counts()[df["city"].value\_counts() >600 ].index.tolist()

eligible\_cities

df=df[df["city"].isin(el df["current\_score"]=df.groupby("match\_id").cumsum()["runs"]

df.head()

df["over"]=df["ball"].apply(lambda x:str(x).split(".")[0])

df["ball\_no"]=df["ball"].apply(lambda x:str(x).split(".")[1])

df.head()

df["balls\_bowled\_with\_extras"]=df.groupby("match\_id").cumcount()+1

df.tail(10)

igible\_cities)]

def fun(ball):

if ball<=6:

return ball

return 6

df["balls\_bowled\_without\_extras"]=(df["over"].astype("int")\*6) + df["ball\_no"].astype("int").apply(fun)

df["extras"]=df["balls\_bowled\_with\_extras"]-df["balls\_bowled\_without\_extras"]

df["extras"].value\_counts()

df["balls\_left"]=120-df["balls\_bowled\_without\_extras"]

df["balls\_left"]=df["balls\_left"].apply(lambda x : 0 if x<0 else x)

df.head()

df["player\_of\_dismissed"]=df["player\_of\_dismissed"].apply(lambda x : 0 if(x=="0" or x==0) else 1)

df["player\_of\_dismissed"].value\_counts()

df["player\_of\_dismissed"]=df.groupby("match\_id").cumsum()["player\_of\_dismissed"]

df.head()

df["wickets\_left"]=10-df["player\_of\_dismissed"]

df.head()

df["crr"]=round((df["current\_score"])\*6/df["balls\_bowled\_without\_extras"],2)

df.tail()

groups=df.groupby("match\_id")

match\_ids=df["match\_id"].unique()

last\_five=[]

for ids in match\_ids:

last\_five.extend( groups.get\_group(ids).rolling(window=30).sum()["runs"].values.tolist() )

df["last\_five"]=last\_five

df.head()

def get\_last\_element(series):

return series.iloc[-1]

df["run\_x"] = df.groupby("match\_id")["current\_score"].transform(get\_last\_element)

df.head()

pickle.dump(df,open("t20\_data\_level3.pkl","wb"))

import pickle

df=pickle.load(open("t20\_data\_level3.pkl","rb"))

final\_df=df[ ["batting\_team","bowling\_team","city","current\_score","balls\_left","extras","wickets\_left","crr","last\_five","run\_x"] ]

final\_df.head()

final\_df.isna().sum()

final\_df.dropna(inplace=True)

final\_df=final\_df.sample(final\_df.shape[0])

final\_df.head()

X = final\_df.drop(columns=['run\_x'],axis=1)

y = final\_df ['run\_x']

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test=train\_test\_split(X,y,test\_size=0.2, random\_state=0)

X\_train

X\_train.columns

from sklearn.compose import ColumnTransformer

from sklearn.preprocessing import OneHotEncoder

from sklearn. pipeline import Pipeline

from sklearn.preprocessing import StandardScaler

from sklearn. ensemble import RandomForestRegressor

from sklearn. linear\_model import LinearRegression

from xgboost import XGBRegressor

from sklearn. metrics import r2\_score, mean\_absolute\_error

categorical\_features = ['batting\_team', 'bowling\_team', 'city']

numerical\_features = ['current\_score', 'balls\_left', 'extras', 'wickets\_left', 'crr', 'last\_five']

preprocessor = ColumnTransformer(

transformers=[

('num', StandardScaler(), numerical\_features),

('cat', OneHotEncoder(), categorical\_features)

])

from sklearn.pipeline import Pipeline

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

from sklearn.model\_selection import train\_test\_split

from sklearn.compose import ColumnTransformer

pipeline = Pipeline([

('preprocessor', preprocessor),

('model', LinearRegression())

])

pipeline.fit(X\_train, y\_train)

y\_pred=pipeline.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = mean\_squared\_error(y\_test, y\_pred, squared=False) # Taking square root to get RMSE

r2 = r2\_score(y\_test, y\_pred)

print("Linear Regression")

print(f'Mean Absolute Error (MAE): {mae}')

print(f'Mean Squared Error (MSE): {mse}')

print(f'Root Mean Squared Error (RMSE): {rmse}')

print(f'R2 Score: {r2}')

pipeline = Pipeline([

('preprocessor', preprocessor),

('model', RandomForestRegressor())

])

pipeline.fit(X\_train, y\_train)

y\_pred=pipeline.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = mean\_squared\_error(y\_test, y\_pred, squared=False) # Taking square root to get RMSE

r2 = r2\_score(y\_test, y\_pred)

print("Random Forest Regression")

print(f'Mean Absolute Error (MAE): {mae}')

print(f'Mean Squared Error (MSE): {mse}')

print(f'Root Mean Squared Error (RMSE): {rmse}')

print(f'R2 Score: {r2}')

pipeline = Pipeline([

('preprocessor', preprocessor),

('model', XGBRegressor(n\_estimators=1000, learning\_rate=0.2, max\_depth=12, random\_state=1))

])

pipeline.fit(X\_train, y\_train)

y\_pred=pipeline.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)

r2 = r2\_score(y\_test, y\_pred)

print("XGB Regression")

print(f'Mean Absolute Error (MAE): {mae}')

print(f'Mean Squared Error (MSE): {mse}')

print(f'Root Mean Squared Error (RMSE): {rmse}')

print(f'R2 Score: {r2}')

with open('model\_XGB.pkl', 'wb') as f:

pickle.dump(pipeline, f)

k=pickle.load(open("model\_XGB.pkl","rb"))

sample\_input = pd.DataFrame({

'batting\_team': "India",

'bowling\_team': "Australia",

'city': "Mumbai",

'current\_score': 150,

'balls\_left': 30,

'extras': 15,

'wickets\_left': 5,

'crr': 9.0,

'last\_five': 70 # List of runs scored in the last 5 balls

},index=[0])

k.predict(sample\_input)

**Attention:**

**Google Drive Link:** **https://drive.google.com/drive/folders/1OFDdJbQTznGs\_61qdU\_TTxBa9zfXS6ph?usp=sharing**

~ Thank You ~