PROJECT REPORT

T20 SCORE PREDICTION

INTRODUCTION:

Overview: The T20 Cricket Score Prediction project aims to leverage Artificial Intelligence (AI) and Machine Learning (ML) techniques to predict the scores of T20 cricket matches. This project is designed to provide cricket enthusiasts, analysts, and stakeholders with a tool that can generate accurate predictions for the total runs a team is likely to score in a T20 match.

Key Objectives:

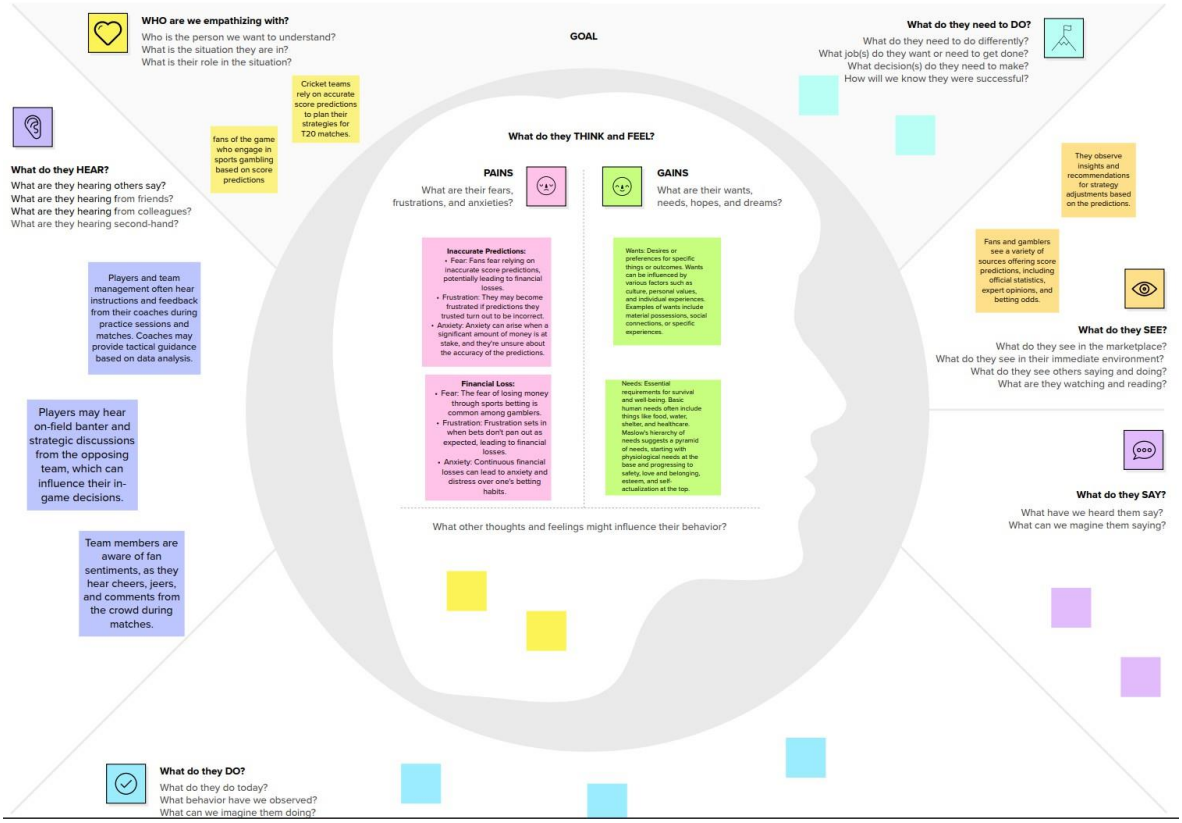
1. **Data Collection:**
   - Gather comprehensive and relevant datasets containing historical T20 match data, including details such as team performance, player statistics, venue characteristics, and match outcomes.
2. **Data Preprocessing:**
   - Clean and preprocess the collected data to handle missing values, outliers, and ensure consistency.
   - Feature engineering to extract relevant information and create meaningful features for the prediction model.
3. **Model Development:**
   - Explore and implement various machine learning algorithms such as regression models, ensemble methods, and neural networks to build the predictive model.
   - Train the model on historical data, using features like team composition, player form, past performance, and match conditions.
4. **Evaluation Metrics:**
   - Define appropriate evaluation metrics to assess the performance of the prediction model, considering factors like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and accuracy.
5. **Hyperparameter Tuning:**
   - Optimize the model by fine-tuning hyperparameters to enhance prediction accuracy.
6. **Validation and Testing:**
   - Validate the model using a separate dataset not used during training to ensure its generalization capability.
   - Conduct thorough testing to assess the model's accuracy and reliability in predicting T20 cricket scores.

7. **User Interface (Optional):**
    - Develop a user-friendly interface for cricket enthusiasts to input match details and receive predicted scores.
8. **Documentation:**
    - Document the entire process, including data sources, preprocessing steps, model architecture, and evaluation metrics.
9. **Deployment:**
    - Deploy the trained model as a web service or application to make predictions in real-time or on-demand.
10. **Continuous Improvement:**
    - Implement mechanisms for continuous learning and improvement, incorporating new data to enhance the model's predictive capabilities
    - over time.

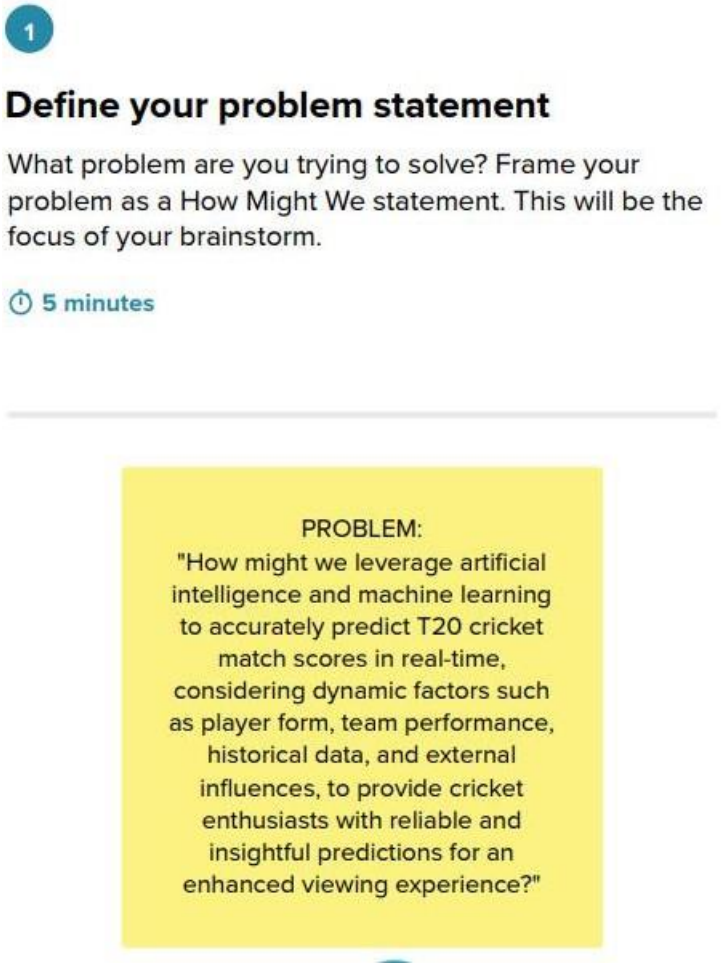
A literature survey is a crucial step in any research project, helping to identify existing work, methodologies, and advancements in the chosen field. Here is a brief literature survey for T20 Cricket Score Prediction using AI and ML:

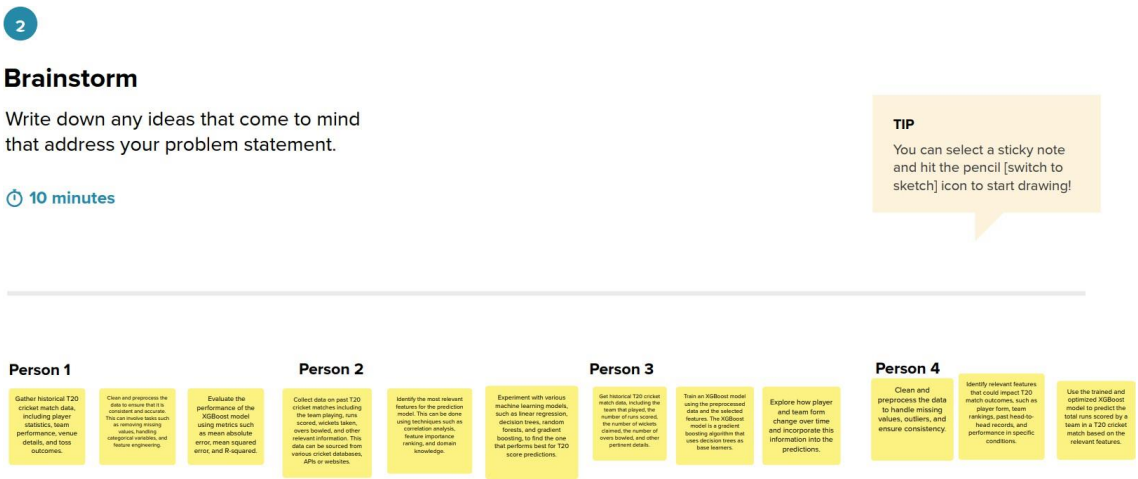
1. **"Cricket Outcome Prediction using Machine Learning" by A. B. Dahanayake, et al. (2018):**
    - This research explores the application of machine learning techniques to predict cricket match outcomes, focusing on T20 matches. It provides insights into feature selection and model evaluation metrics, which could be beneficial for our project.
2. **"Predicting the outcome of one day international cricket matches" by A. De Silva, et al. (2018):**
    - The paper discusses the prediction of cricket match outcomes, a related area to T20 score prediction. It covers feature engineering and model selection techniques, offering valuable guidance for our project.

Empathy Map Canvas



Ideation & Brainstorming





3

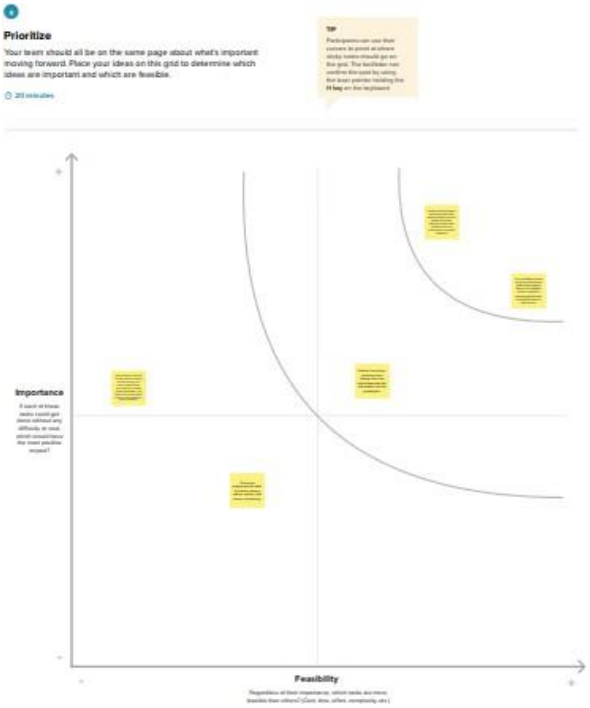
Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.



Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	In the realm of T20 cricket, Accurate prediction of T20 cricket scores is a challenging task due to the dynamic nature of the sport and the involvement of numerous factors that influence the outcome of match. Traditional prediction methods often lack the ability to capture the complex interaction between these factors, leading to suboptimal prediction accuracy.
2.	Idea / Solution description	This project processes an end-to-end machine learning solution that leverages the power of deep learning algorithms, specifically Convolutional Neural Networks (CNNs), to extract relevant features from a comprehensive dataset of historical match data, player statistics, and pitch conditions. These extracted feature will then be used to train a prediction model that can accurately predict the T20 score of the batting team.
3.	Novelty / Uniqueness	The uniqueness of this project lies in the application of CNNs to cricket score prediction. By harnessing the power of deep learning, the model can discern complex patterns in cricket match data, providing a sophisticated and innovative approach to T20 score predictions.
4.	Social Impact / Customer Satisfaction	Accurate T20 score predictions can enhance the overall experience for cricket enthusiasts, providing insights into match dynamics. Sports analysts, teams, and fans can benefit from more informed decision-making, ultimately elevating the enjoyment of T20 cricket. The user-friendly web application ensures accessibility, making it a valuable tool for cricket enthusiasts.

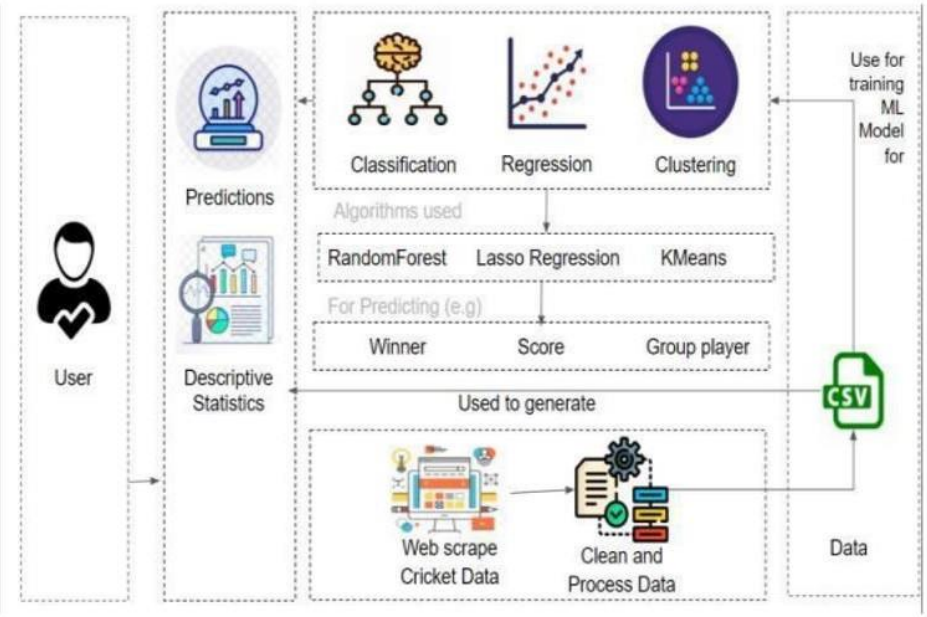
5.	Business Model (Revenue Model)	The business model revolves around offering the machine learning solution as a service. Potential revenue streams include subscription-based access to the predictive analytics platform, premium features, and collaboration opportunities with cricket teams for customized insights. Strategic partnerships with sports media and betting platforms can also be explored.
6.	Scalability of the Solution	The machine learning solution is designed with scalability in mind. As more cricket match data becomes available, the model can be retrained to adapt to evolving patterns and trends. The web application can handle increased user traffic, ensuring scalability in terms of both technology and user engagement.

SOLUTION ARCHITECTURE

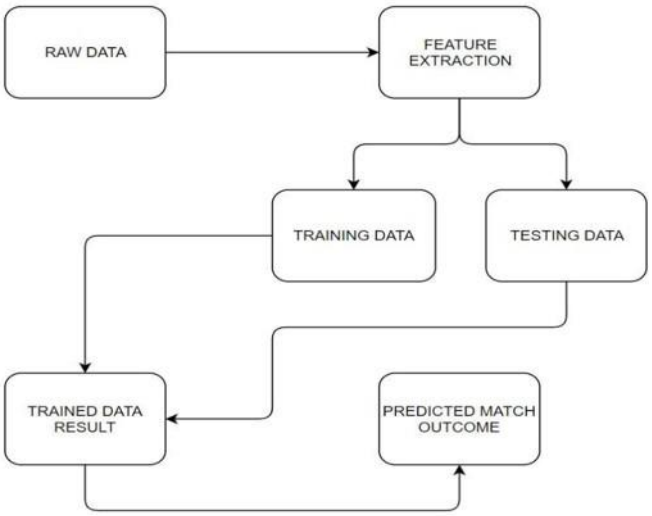
Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.



DATA FLOW DIAGRAM



User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
T20 score prediction	Accessing the data	USN-1	As a user, I can access the analysis dataset by logging into with my credentials	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can filter and search the analysis dataset of prediction based on the user input	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Work Email		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
Prediction	Support and Assistance	USN-6	As a customer care executive, I can view summary reports and conditions of the stadium of the analysis datasets	I can access summary reports displaying overall statistics and trends from the dataset	High	Sprint -2
		USN-7	As a customer care executive, I can generate custom reports based on specific criteria.		Medium	Sprint-3
		USN-8	As a cricket enthusiast, I want to input historical T20 match data, including team performance, player statistics, and match conditions, so that the AI model can learn from past games and make accurate predictions.	I can acess the historical T20 match dataset and performance	High	Sprint-3
Dashboard		USN-9	As an administrator, I can manage user access levels and permissions.	I can get the prediction and datasets of the score	High	Sprint-4
Sport analyst		USN-10	As a sport analyst I want give the venue details	I can acess the location and venue details of the stadium	Medium	Sprint-4

Product Backlog, Sprint Schedule, and Estimation

Spíi→ıt	Ïu→ıctio→ıal Rcquiícmc→ıt (Epic)	Uscí StoíQ Numbcí	Uscí StoíQ / l'ask	StoíQ Poi→ıts	PíioíitQ	I'cam Mcmbcís
Spíi→ıt-1	Rcgistíatio→ı	l'20-USN-1	As a "scí, I ca→ı ícgistcí roí tkc l'20 applicatio→ı bQ c→ıtcíi→ıg mQ cmail, passwoíd, a→ıd co→ıriími→ıg mQ passwoíd.	2	Higk	Siddkaítk, Abi→ıcsk
Spíi→ıt-1	Rcgistíatio→ı	l'20-USN-2	As a "scí, I will íccci:c a co→ıriímatio→ı cmail o→ıcc I ka:c ícgistcícd roí tkc l'20 applicatio→ı.	1	Higk	R"pa Kíisk→ıa, Rakcsk
Spíi→ıt-2	Pícdictio→ı	l'20-USN-«	As a "scí, I ca→ı pícdict scoícs roí l'20 matckcs i→ı tkc applicatio→ı.	«	Mcdi"m	Siddkaítk
Spíi→ıt-2	Pícdictio→ı	l'20-USN-4	As a "scí, I ca→ı :icw pícdictio→ıs madc bQ otkcí "scís.	2	Low	Abi→ıcsk
Spíi→ıt-«	Lcadcíboaíd	l'20-USN-5	As a "scí, I ca→ı :icw tkc lcadcíboaíd to scc tkc top pícdictoís i→ı tkc l'20 applicatio→ı.	2	Higk	R"pa Kíisk→ıa
Spíi→ıt-«	A"tkc→ıticatio→ı	l'20-USN-6	E→ıka→ıcc "scí a"tkc→ıticatio→ı roí bcttcí scc"íitQ.	2	Higk	Rakcsk

Spíi→ıt-4	Notiricatio→ıs	l'20-USN-7	As a "scí, I íccci:c →ıotiricatio→ıs roí matck ícs"lts a→ıd lcadcíboaíd "pdatcs.	«	Mcdi"m	Siddkaítk, Rakcsk
Spíi→ıt-4	Daskboaíd	l'20-USN-8	As a "scí, I ca→ı acccss a pcíso→ıalizcd daskboaíd witk a s"mmaíQ or mQ pícdictio→ıs a→ıd pcíroíma→ıcc.	4	Higk	Abi→ıcsk, R"pa Kíisk→ıa

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Velocity:

Spíi→ıt	I'otal StoíQ Poi→ıts	Duíatio→ı	Spíi→ıt Staít Datc	Spíi→ıt E→ıd Datc (Pla→ı→ıcd)	StoíQ Poi→ıts Complctcd (as o→ı Pla→ı→ıcd E→ıd Datc)	Spíi→ıt Rclcasc Datc (Actual)
Spíi→ıt-1	20	6 DaQs	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Spíi→ıt-2	20	6 DaQs	«1 Oct 2022	05 No: 2022	15	04 No: 2022
Spíi→ıt-«	20	6 DaQs	07 No: 2022	12 No: 2022	18	11 No: 2022
Spíi→ıt-4	20	6 DaQs	14 No: 2022	19 No: 2022	20	19 No: 2022

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

Average Velocity= 73/4

Average Velocity=18.25

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

https://www.visual-paradigm.com/scrum/scrum-burndown-chart/

https://www.atlassian.com/agile/tutorials/burndown-charts

Reference:

https://www.atlassian.com/agile/project-management

https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software

https://www.atlassian.com/agile/tutorials/epics

https://www.atlassian.com/agile/tutorials/sprints

https://www.atlassian.com/agile/project-management/estimation

https://www.atlassian.com/agile/tutorials/burndown-charts

Table-1: Components & Technologies

S.No	Component	Description	Technology
1	User Interface	Prediction Interface for users	React.js, Redux, HTML, CSS
2	Prediction Engine	Core logic for score predictions	Python, TensorFlow, Scikit-learn
3	Data Storage	Storage for historical match and player data	MongoDB, Redis
4	External API-1	Cricket Match Data API for real-time updates	Cricbuzz API, ESPN API
5	External API-2	Player Statistics API for player performance	Cricinfo API, Cricket API
6	Machine Learning Model	Score Prediction Machine Learning Model	Linear Regression, Random Forest, XGBoost
7	Notification Service	Notify users of match results and predictions	Push Notifications, Firebase Cloud Messaging
8	User Authentication	Secure user authentication and authorization	OAuth 2.0, JWT
9	Cloud Infrastructure	Hosting and scaling the application	AWS (EC2, S3), Docker, Kubernetes
10	Logging and Monitoring	Track application performance and errors	ELK Stack (Elasticsearch, Logstash, Kibana)

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology
1	Open-Source Frameworks	Utilize open-source frameworks for efficiency	Django, Flask (Python frameworks), Bootstrap
2	Security Implementations	Implement security measures for data protection	HTTPS, SSL/TLS, Hashing (SHA-256), OWASP
3	Scalable Architecture	Design the application for scalability	Microservices Architecture, Load Balancers, CDN
4	Availability	Ensure high availability for users	Load Balancers, Redundancy, Disaster Recovery
5	Performance	Optimize performance for quick predictions	Caching (Redis), Content Delivery Networks (CDN)

End-to-end machine learning project to predict T20 score.

The objective of this project is to develop an end-to-end machine learning solution for predicting the t20 score of the batting team. The proposed solution involves the use of Machine learning algorithms to extract relevant features from the input and predict the accurate score.

Creating an end-to-end machine learning project to predict T20 cricket scores can offer a range of benefits, both from a sports analytics perspective and as a showcase of machine learning capabilities. Here are some of the key advantages:

1. **Insightful Analytics:** Developing a T20 score prediction model can provide deep insights into the factors that influence team performance and scoring in cricket matches. This could include player statistics, pitch conditions, team composition, weather conditions, and more.

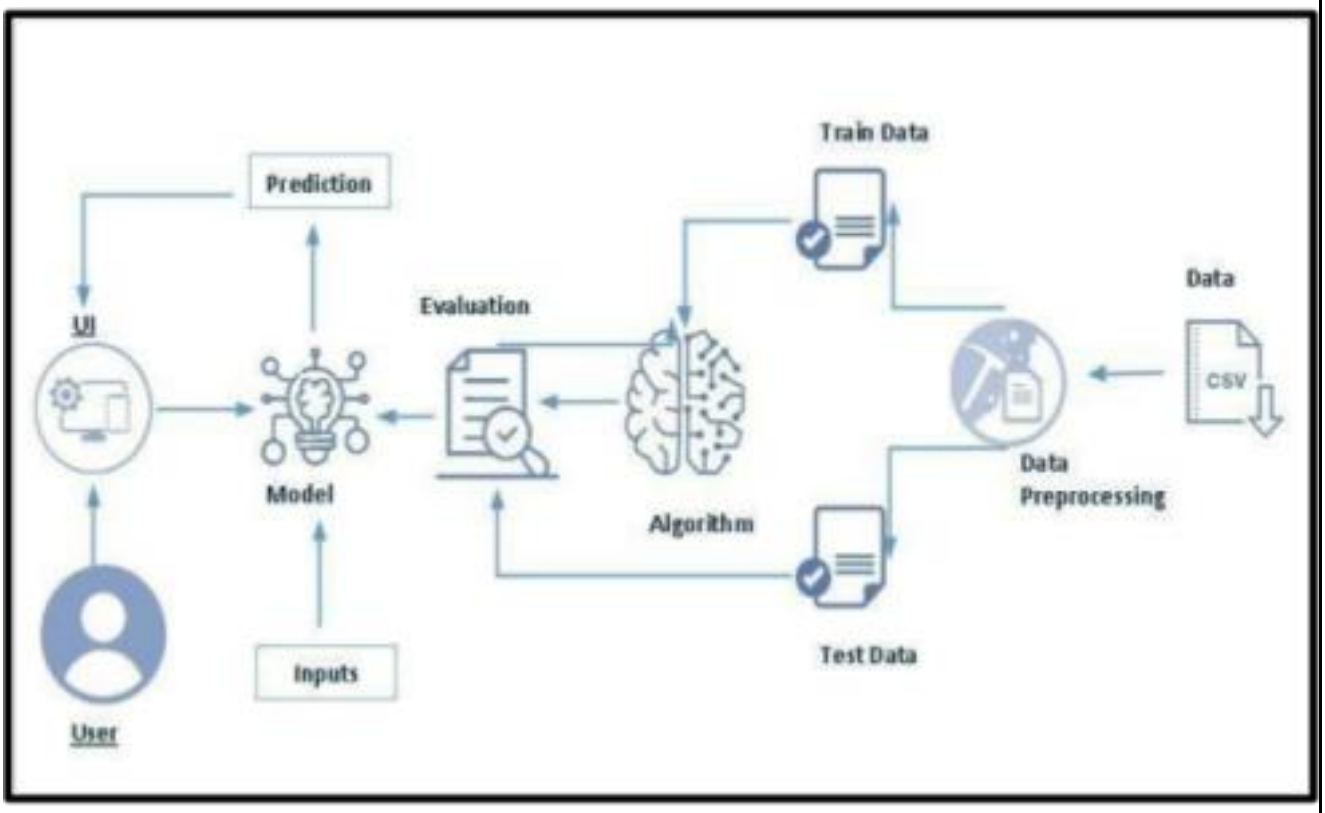
2. **Strategic Decision Making:** Cricket teams, coaches, and analysts can use the predictive model to make informed decisions during matches. This could involve adjusting strategies based on predicted scores, understanding the impact of different factors on the outcome, and optimizing team compositions.

3. **Engaging Fan Experience:** Fans of the sport can benefit from more engaging and interactive experiences. Predicted scores can be displayed in real-time during live broadcasts, enhancing fan engagement and offering viewers a better understanding of the ongoing match dynamics.

4. **Betting and Fantasy Sports:** Predictive models are often sought after by individuals engaged in sports betting and fantasy sports leagues. Accurate score predictions can be used to make informed betting decisions or create more competitive fantasy teams.

Let us look at the Technical Architecture of the project.

Technical Architecture:



Data Collection & Preparation

ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download the required dataset.

Activity 1: Collect the dataset.

Importing the libraries

Import the necessary libraries as shown in the image.

```
import numpy as np
import pandas as pd
from yaml import safe_load
import os
from tqdm import tqdm
```

Converting .yaml files to dataframes.

```
filenames = []
for file in os.listdir('data'):
    filenames.append(os.path.join('data',file))
```

```
filenames[0:5]
```

```
['data\\1001349.yaml',
 'data\\1001351.yaml',
 'data\\1001353.yaml',
 'data\\1004729.yaml',
 'data\\1007655.yaml']
```

Conversion :

```
final_df = pd.DataFrame()
counter = 1
for file in tqdm(filenames):
    with open(file, 'r') as f:
        df = pd.json_normalize(safe_load(f))
        df['match_id'] = counter
        final_df = final_df.append(df)
        counter+=1

final_df
```

Below shown dataframe should be you output.

	innings	meta.data_version	meta.created	meta.revision	info.dates	info.gender	info.match_type	info.outcome.by.wickets	info.outcome.winner	info.overs	...	info.o
0	[{'1st innings': {'team': 'Australia', 'delive...	0.9	2017-02-18	2	[2017-02-17]	male	T20	5.0	Sri Lanka	20	...	
0	[{'1st innings': {'team': 'Australia', 'delive...	0.9	2017-02-19	2	[2017-02-19]	male	T20	2.0	Sri Lanka	20	...	
0	[{'1st innings': {'team': 'Australia', 'delive...	0.9	2017-02-23	1	[2017-02-22]	male	T20	NaN	Australia	20	...	
0	[{'1st innings': {'team': 'Hong Kong', 'delive...	0.9	2016-09-12	1	[2016-09-05]	male	T20	NaN	Hong Kong	20	...	
0	[{'1st innings': {'team': 'Zimbabwe', 'deliver...	0.9	2016-06-19	1	[2016-06-18]	male	T20	NaN	Zimbabwe	20	...	
...	...	...	...	...	...	...	...	...	...	...	...	
0	[{'1st innings': {'team': 'Sri Lanka', 'delive...	0.9	2016-03-05	2	[2016-03-04]	male	T20	6.0	Pakistan	20	...	

Exploratory Data Analysis

Activity 1: Dropping unnecessary columns

```
final_df.drop(columns=[
        'meta.data_version',
        'meta.created',
        'meta.revision',
        'info.outcome.bowl_out',
        'info.bowl_out',
        'info.supersubs.South Africa',
        'info.supersubs.New Zealand',
        'info.outcome.eliminator',
        'info.outcome.result',
        'info.outcome.method',
        'info.neutral_venue',
        'info.match_type_number',
        'info.outcome.by.runs',
        'info.outcome.by.wickets'
    ],inplace=True)
```

The above mentioned columns are unnecessary for our prediction. So, let's drop them.

- We'll be predicting the scores for men's t20 only, because there is no sufficient data to train the machine learning model for women's t20 also.

```
final_df['info.gender'].value_counts()
```

```
male      966
female    466
Name: info.gender, dtype: int64
```

```
final_df = final_df[final_df['info.gender'] == 'male']
final_df.drop(columns=['info.gender'],inplace=True)
final_df
```

Now we have dropped the gender column and reduced the number of rows.

- The given dataset also contains some 50 over matches.

```
final_df['info.match_type'].value_counts()
```

```
T20    966
Name: info.match_type, dtype: int64
```

```
final_df['info.overs'].value_counts()
```

```
20    963
50      3
Name: info.overs, dtype: int64
```

We've found three 50 over matches. Let's remove them.

```
final_df = final_df[final_df['info.overs'] == 20]
final_df.drop(columns=['info.overs','info.match_type'],inplace=True)
final_df
```

```
count = 1
delivery_df = pd.DataFrame()
for index, row in final_df.iterrows():
    if count in [75,108,150,180,268,360,443,458,584,748,982,1052,1111,1226,1345]:
        count+=1
        continue
    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_dismissed':player_of_dismissed,
            'city':city,
            'venue':venue
        })
    delivery_df = delivery_df.append(loop_df)
```

You'll get some 1.15 Lakh rows after extracting it.
Now, Let's extract the bowling team from the teams column and drop the teams column.

```
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
```

Let's remove the unbalanced data i.e teams which played less no.of matches.

```
delivery_df['batting_team'].value_counts()
```

```
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
```

We've reduced the no.of rows by eliminating unbalanced data.

The following data is required for our model.

```
output = delivery_df[['match_id','batting_team','bowling_team','ball','runs','player_dismissed','city','venue']]
```

```
output
```

	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue
0	2	Australia	Sri Lanka	0.1	0	0	NaN	Melbourne Cricket Ground
1	2	Australia	Sri Lanka	0.2	0	0	NaN	Melbourne Cricket Ground
2	2	Australia	Sri Lanka	0.3	1	0	NaN	Melbourne Cricket Ground
3	2	Australia	Sri Lanka	0.4	2	0	NaN	Melbourne Cricket Ground
4	2	Australia	Sri Lanka	0.5	0	0	NaN	Melbourne Cricket Ground
...	...	...	...	...	...	...	...	...
121	964	Sri Lanka	Australia	19.3	1	0	Colombo	R Premadasa Stadium
122	964	Sri Lanka	Australia	19.4	0	0	Colombo	R Premadasa Stadium
123	964	Sri Lanka	Australia	19.5	0	DM de Silva	Colombo	R Premadasa Stadium
124	964	Sri Lanka	Australia	19.6	2	0	Colombo	R Premadasa Stadium
125	964	Sri Lanka	Australia	19.7	1	0	Colombo	R Premadasa Stadium

63888 rows × 8 columns

Activity 2: Feature Extraction

- Looking for null values

```
df.isnull().sum()
```

```
Unnamed: 0             0
match_id               0
batting_team           0
bowling_team           0
ball                   0
runs                   0
player_dismissed       0
city                8548
venue                  0
dtype: int64
```

- Extracting city names using venue column and dropping venue column

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

```
df['city'] = cities
```

```
df.isnull().sum()
```

```
Unnamed: 0          0
match_id            0
batting_team        0
bowling_team        0
ball                0
runs                0
player_dismissed    0
city                0
venue               0
dtype: int64
```

- Filtering the cities based on the number of balls thrown in each city. If the no.of matches played in each stadium is less than 5 i.e 600 balls, we'll remove that city.

```
eligible_cities = df['city'].value_counts()[df['city'].value_counts() > 600].index.tolist()
```

```
df = df[df['city'].isin(eligible_cities)]
```

```
df
```

	Unnamed: 0	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue
0	0	2	Australia	Sri Lanka	0.1	0	0	Melbourne	Melbourne Cricket Ground
1	1	2	Australia	Sri Lanka	0.2	0	0	Melbourne	Melbourne Cricket Ground
2	2	2	Australia	Sri Lanka	0.3	1	0	Melbourne	Melbourne Cricket Ground
3	3	2	Australia	Sri Lanka	0.4	2	0	Melbourne	Melbourne Cricket Ground
4	4	2	Australia	Sri Lanka	0.5	0	0	Melbourne	Melbourne Cricket Ground
...	...	...	...	...	...	...	...	...	...
63883	121	964	Sri Lanka	Australia	19.3	1	0	Colombo	R Premadasa Stadium
63884	122	964	Sri Lanka	Australia	19.4	0	0	Colombo	R Premadasa Stadium
63885	123	964	Sri Lanka	Australia	19.5	0	DM de Silva	Colombo	R Premadasa Stadium
63886	124	964	Sri Lanka	Australia	19.6	2	0	Colombo	R Premadasa Stadium
63887	125	964	Sri Lanka	Australia	19.7	1	0	Colombo	R Premadasa Stadium

- Creating a new column with the current score. Let's write the code to extract the current score.

```
df['current_score'] = df.groupby('match_id').cumsum()['runs']
```

- Now let's create two more columns, namely 'over' and 'ball_no'.

```
df['over'] = df['ball'].apply(lambda x : str(x).split(".")[0])
df['ball_no'] = df['ball'].apply(lambda x : str(x).split(".")[1])
```

- Similarly, we'll write the code for 'balls_bowled' and 'balls_left'.

```
df['balls_bowled'] = (df['over'].astype('int')*6 + df['ball_no'].astype('int'))
```

```
df['balls_left'] = 120 - df['balls_bowled']


df['balls_left'] = df['balls_left'].apply(lambda x: 0 if x < 0 else x)
```

- Again the same thing for ‘player_dismissed’,‘wickets_left’ and current run rate(‘crr’).

```
df['player_dismissed'].apply(lambda x: 1 if x != '0' else '0')
```

```
0        0
1        0
2        0
3        0
4        0
        ..
63883    0
63884    0
63885    1
63886    0
63887    0
Name: player_dismissed, Length: 50501, dtype: object
```

```
df['player_dismissed'] = df['player_dismissed'].astype('int')


df['player_dismissed'] = df.groupby('match_id').cumsum()['player_dismissed']


df['wickets_left'] = 10 - df['player_dismissed']
```

```
df['crr'] = (df['current_score']*6) / df['balls_bowled']
```

- Extracting the runs in the last 5 overs of every match and adding it as a new column.

```
groups = df.groupby('match_id')

match_ids = df['match_id'].unique()
last_five = []
for id in match_ids:
    last_five.extend(groups.get_group(id).rolling(window=30).sum()['runs'].values.tolist())


df['last_five'] = last_five
```

- Selecting only required features from the dataframe.

  final_df = final_df[['batting_team', 'bowling_team', 'city', 'current_score', 'balls_left', 'wickets_left', 'crr', 'last_five', 'runs_x']]

- Checking for null values in the final dataframe and drop them.

```
final_df.isnull().sum()
```

```
batting_team      0
bowling_team      0
city              0
current_score     0
balls_left        0
wickets_left      0
crr               0
last_five         0
runs_x            0
dtype: int64
```

- Let's take a look at the final data frame which is ready for training.

```
final_df = final_df.sample(final_df.shape[0])
```

```
final_df
```

	batting_team	bowling_team	city	current_score	balls_left	wickets_left	crr	last_five	runs_x
33284	Sri Lanka	England	Pallekele	82	52	7	7.235294	32.0	169
46289	Sri Lanka	West Indies	Pallekele	188	12	7	10.444444	64.0	215
34111	South Africa	England	Manchester	42	89	7	8.129032	42.0	77
1080	India	England	Bangalore	136	29	7	8.967033	57.0	202
13235	Australia	India	Sydney	79	63	9	8.315789	41.0	186
...	...	...	...	...	...	...	...	...	...
36741	West Indies	Bangladesh	Mirpur	69	68	8	7.961538	43.0	197
48172	India	Bangladesh	Bangalore	104	32	7	7.090909	47.0	146
46731	South Africa	Sri Lanka	Cape Town	154	4	5	7.965517	38.0	169
45891	Pakistan	New Zealand	Auckland	105	34	7	7.325581	36.0	171
40497	India	Australia	Mirpur	53	68	7	6.115385	25.0	159

38477 rows × 9 columns

Activity 3: Splitting data into train and test sets

Now let's split the Dataset into train and test sets.
The split will be in 8:2 ratio - train : test respectively.

```
X = final_df.drop(columns=['runs_x'])
y = final_df['runs_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=1)
```

```
X_train
```

	batting_team	bowling_team	city	current_score	balls_left	wickets_left	crr	last_five
24147	Australia	Pakistan	St Lucia	87	59	8	8.557377	38.0
19058	Australia	England	Manchester	132	8	6	7.071429	49.0
41484	Pakistan	South Africa	Cape Town	45	74	8	5.869565	32.0
12794	Sri Lanka	Pakistan	Lahore	140	8	5	7.500000	45.0
6751	Pakistan	South Africa	Centurion	61	76	8	8.318182	37.0
...	...	...	...	...	...	...	...	...
17825	India	Australia	Durban	40	75	9	5.333333	31.0
46596	South Africa	Sri Lanka	Johannesburg	104	19	3	6.178218	30.0
8047	New Zealand	India	Auckland	119	32	6	8.113636	59.0
19053	Australia	England	Manchester	130	13	7	7.289720	53.0
30723	Sri Lanka	Pakistan	Abu Dhabi	130	18	6	7.647059	30.0

30781 rows × 8 columns

Activity 4: Importing required packages and Encoding.

- Importing required packaged

```
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 xgboost import XGBRegressor
from sklearn.metrics import r2_score,mean_absolute_error
```

- One hot encoding the data using the Column transfer method.

```
trf = ColumnTransformer([
    ('trf',OneHotEncoder(sparse=False,drop='first'),['batting_team','bowling_team','city'])
]
,remainder='passthrough')
```

Milestone 4: Model Training

In this step we'll select models and train them, step by step.

We'll use 3 machine learning algos and find the best one out.

Model 1: Linear Regression

Creating pipeline

```
pipe = Pipeline(steps=[
    ('step1',trf),
    ('step2',StandardScaler()),
    ('step3',XGBRegressor(n_estimators=1000,learning_rate=0.2,max_depth=12,random_state=1))
])
```

Training and calculating the accuracy

```
pipe.fit(X_train,y_train)
y_pred = pipe.predict(X_test)
print(r2_score(y_test,y_pred))
print(mean_absolute_error(y_test,y_pred))
```

```
0.6885977930742969
13.160578311496517
```

Accuracy is near 68%

Model 2: Random Forest Regressor

Creating Pipeline

```
pipe = Pipeline(steps=[
    ('step1',trf),
    ('step2',StandardScaler()),
    ('step3',RandomForestRegressor())
])
```

Training and calculating the accuracy

```
pipe.fit(X_train, y_train)
y_pred = pipe.predict(X_test)
print(r2_score(y_test,y_pred))
print(mean_absolute_error(y_test,y_pred))
```

```
0.9767823737527738
2.1144844106136844
```

Accuracy is somewhere around 97%

Model 3: XGBRegressor

Creating Pipeline

```
pipe = Pipeline(steps=[
    ('step1',trf),
    ('step2',StandardScaler()),
    ('step3',XGBRegressor(n_estimators=1000,learning_rate=0.2,max_depth=12,random_state=1))
])
```

Training and calculating the accuracy

```
pipe.fit(X_train,y_train)
y_pred = pipe.predict(X_test)
print(r2_score(y_test,y_pred))
print(mean_absolute_error(y_test,y_pred))
```

```
0.9863469919711688
1.6940391234406624
```

Accuracy is somewhere around 98%.

Model Deployment

Integrate with Web Framework

In this section, we will be building a web application that is integrated to the model we built.

We will be using the streamlit package for our website development.

Streamlit is a free and open-source framework to rapidly build and share beautiful machine learning and data science web apps.

Create a web.py file and import necessary packages:

```
import streamlit as st
import pickle
import pandas as pd
import numpy as np
```

Activity 2.2: Defining teams/cities names and loading the pipeline:

```
pipe = pickle.load(open('pipe.pkl', 'rb'))

teams = [
    'Australia',
    'India',
    'Bangladesh',
    'New Zealand',
    'South Africa',
    'England',
    'West Indies',
    'Afghanistan',
    'Pakistan',
    'Sri Lanka'
]

cities = ['Colombo',
    'Mirpur',
    'Johannesburg',
    'Dubai',
    'Auckland',
    'Cape Town',
    'London',
    'Pallekele',
    'Barbados',
    'Sydney',
    'Melbourne',
    'Durban',
```

Activity 2.3: Accepting the input from user and prediction

```python
st.title('Cricket Score Predictor')

col1, col2 = st.columns(2)

with col1:
    batting_team = st.selectbox('Select batting team', sorted(teams))

with col2:
    bowling_team = st.selectbox('Select bowling team', sorted(teams))

city = st.selectbox('Select city', sorted(cities))

col3,col4,col5 = st.columns(3)

with col3:
    current_score = st.number_input('Current Score')

with col4:
    overs = st.number_input('Overs Done (works for over > 5)')

with col5:
    wickets = st.number_input('Wickets Out')

last_five = st.number_input("Runs scored in last 5 overs")

if st.button('Predict Score'):
    balls_left = 120 - (overs * 6)
    wickets_left = 10 - wickets
    crr = current_score/overs

    input_df = pd.DataFrame(
        {'batting_team': [batting_team], 'bowling_team': [bowling_team], 'city': city, 'current_score': [current_score],
         'balls_left': [balls_left], 'wickets_left': [wickets], 'crr': [crr], 'last_five': [last_five]})
    result = pipe.predict(input_df)
    st.header("Predicted Score - " + str(int(result[0])))
```

To run your website go to your terminal with your respective directory and run the command :

- “streamlit run web.py”

```
PS D:\cricket_score_pred-main\cricket_score_pred-main> streamlit run app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.29.67:8501
```

On successful execution you’ll get to see this in your terminal.

HOW DOES YOUR WEBSITE LOOK LIKE ?

- **Before entering the data :**

- **After entering the data :**

Project Demonstration & Documentation

Below mentioned deliverables to be submitted along with other deliverables. **Activity 1: - Record explanation Video for project end to end solution.**

Summary:

Creating a T20 score prediction model using machine learning involves leveraging historical data, player statistics, and various features to make accurate predictions about a team's performance in a T20 cricket match. Here's a summary of the process:

1. **Data Collection:**
    - **Gather historical data from T20 matches, including team performance, player statistics, venue details, and match outcomes.**
    - **Features can include batting averages, strike rates, bowling economy rates, player form, team composition, and more.**
2. **Data Preprocessing:**
    - **Clean and preprocess the data to handle missing values, outliers, and irrelevant information.**
    - **Normalize or scale numerical features to ensure consistency in the model training.**
3. **Feature Selection:**
    - **Identify and select relevant features that contribute significantly to predicting T20 scores.**
    - **Consider factors like recent player form, head-to-head team performance, and venue conditions.**
4. **Model Selection:**
    - **Choose an appropriate machine learning algorithm for score prediction. Common choices include regression models such as linear regression, decision trees, or ensemble methods like random forests.**
5. **Model Training:**
    - **Split the dataset into training and testing sets to train and evaluate the model.**
    - **Utilize techniques like cross-validation to ensure robustness and avoid overfitting.**

6. **Parameter Tuning:**

   - **Fine-tune the model parameters to enhance its predictive accuracy.**
   - **Use techniques like grid search or random search to find optimal hyperparameters.**

7. **Evaluation:**

   - **Evaluate the model's performance using metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), or R-squared.**
   - **Assess the model's ability to generalize to new, unseen data.**

8. **Prediction:**

   - **Apply the trained model to predict T20 scores for upcoming matches.**
   - **Continuously update the model with new data to improve its accuracy over time.**

9. **Deployment:**

   - **Implement the model in a production environment for real-time score predictions.**
   - **Ensure that the model is regularly updated to adapt to changing player form, team dynamics, and other factors.**

10. **Monitoring and Maintenance:**

    - **Monitor the model's performance and recalibrate as needed to account for shifts in player performance or changes in the game environment.**

Video Link:

https://drive.google.com/file/d/1dAY1GqRDBZvKs6-D73J9Pcm6rOGxZ1EU/view?usp=sharing