MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

Project Report

on

IPL Match Win Predictor

Submitted By:

Goutam Parmar

0901AI211032

Faculty Mentor:

Dr. Neelam Arya

CENTRE FOR ARTIFICIAL INTELLIGENCE

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR - 474005 (MP) est. 1957

JULY-DEC. 2023

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

CERTIFICATE

This is certified that **Goutam Parmar** 0901AI211032 has submitted the project report titled IPL Match Win Predictor under the mentorship of Dr. Neelam Arya in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in **Artificial Intelligence and Robotics** from Madhav Institute of Technology and Science, Gwalior.

Neelam
23/11/23

Dr. Neelam Arya
Faculty Mentor
Professor
Centre for Artificial Intelligence

23/11/22

Dr. R. R. Singh
Coordinator
Centre for Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfilment of requirement for the award of the degree of Bachelor of Technology in **Artificial Intelligence and Robotics** at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of Dr. Neelam Arya, Professor, Centre for Artificial Intelligence

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

Goutam Parmar
0901AI211032
3 Year,
Centre for Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Centre for Artificial Intelligence,** for allowing me to explore this project. I humbly thank **Dr. R. R. Singh,** Coordinator, Centre for Artificial Intelligence, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors. I am grateful to the guidance of **Dr. Neelam Arya,** Professor, Centre for Artificial Intelligence, for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

Goutam

Goutam Parmar
0901AI211032
3 Year,
Centre for Artificial Intelligence

TABLE OF CONTENTS

Abstract

The IPL Match Win Predictor is a web-based application that harnesses machine learning techniques to estimate the likelihood of a team winning or losing a cricket match in the context of the Indian Premier League (IPL). Each IPL match result depends on numerous factors, including the participating teams, the match location, the target score, the current score, overs played, and wickets taken. The IPL Match Outcome Predictor offers an intuitive and user-friendly interface, making it a valuable tool for cricket enthusiasts, analysts, and fans looking to gain insights into the potential results of IPL matches.

सारः

आईपीएल मैच विन प्रिडिक्टर एक वेब-आधारित एप्लिकेशन है जो इंडियन प्रीमियर लीग (आईपीएल) के संदर्भ में किसी टीम के क्रिकेट मैच जीतने या हारने की संभावना का अनुमान लगाने के लिए मशीन लर्निंग तकनीकों का उपयोग करता है। प्रत्येक आईपीएल मैच का परिणाम कई कारकों पर निर्भर करता है, जिसमें भाग लेने वाली टीमें, मैच का स्थान; लक्ष्य स्कोर, वर्तमान स्कोर, खेले गए ओवर और लिए गए विकेट शामिल हैं। आईपीएल मैच परिणाम पूर्वानुमानक एक सहज और उपयोगकर्ता के अनुकूल इंटरफ़ेस प्रदान करता है, जो इसे क्रिकेट प्रेमियों, विश्लेषकों और आईपीएल मैचों के संभावित परिणामों के बारे में जानकारी प्राप्त करने के इच्छुक प्रशंसकों के लिए एक मूल्यवान उपकरण बनाता है।

Chapter 1:Project Overview

1.1 Introduction:

Cricket, one of the most popular sports in the world, has a fanatical following in countries like India, and its premier tournament, the Indian Premier League (IPL), is a sporting spectacle that captivates millions of fans each year.The project relies on a logistic regression model that has been trained on historical IPL match data, enabling it to consider several critical factors.

1.2 Objective and Scope:

1.2.1 Objective:

The objective of this project is to empowers users to engage with data-driven insights and enhance their understanding of the dynamics that govern the game of cricket in the context of the IPL. This project stands as a testament to the ever-expanding boundaries of technology and its applications in the world of sports.

1.2.2 Scope:

The scope of the "IPL Win Predictor" encompasses various aspects, including data collection, model development, evaluation, and potential deployment. Key elements of the project scope include:

1.3 Project Requirement:

The requirement for this Python project is, you need to have Python (3.6 version recommended) installed on your system, then using pip, you can install the necessary packages. And Also, I am using Jupyter notebook and Streamlit for generating UI.

Chapter 2:Literature and review

2.1 Python

Python is a high-level programming language known for its readability and versatility. It's favored for its simplicity and ease of use, making it a go-to language for various applications, including web development, data analysis, artificial intelligence, and more. With a rich ecosystem of libraries and frameworks, Python simplifies complex tasks and encourages clean, efficient code. Its syntax emphasizes readability, which means that even newcomers find it accessible and understandable. Overall, Python's popularity stems from its combination of power, simplicity, and extensive community support.

2.2 Machine Learning

Machine learning is a subset of artificial intelligence that enables computers to learn and make predictions or decisions without explicit programming. It involves creating algorithms and models that can learn from data, identify patterns, and make predictions or take actions based on that learned information. Machine learning can be categorized into supervised learning, unsupervised learning, and reinforcement learning.

Chapter3:Methodology/ Flow Chart

3.1 Collection of data

Data collection is needed before any kind of machine learning research is carried out. Dataset validity is a must otherwise there is no point in analyzing the data.The data of this project was collected from Kaggle, it is an online community of data scientist and ML engineers. It allows users to find datasets they want to use in building AI models.

Data Collection Category:-

1. Matches result
2. Deliveries result

3.2 Data preprocessing

Data preprocessing is the process of cleaning our data set. There might be missing values or outliers in the dataset. These can be handled by data cleaning. If there are many missing values in a variable we will drop values or substitute it with the average value.

3.3 Training the model

Since the data is broken down into two modules: a Training set and Test set, we must initially train the model. The training set includes the target variable. Machine learning algorithms are applied to the training data set.

3.4 Flow Chart:

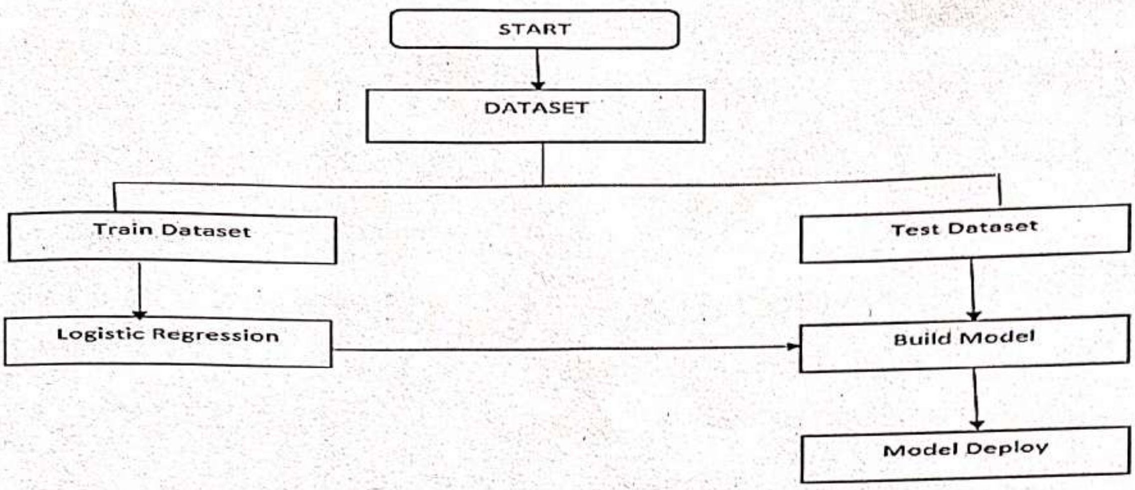
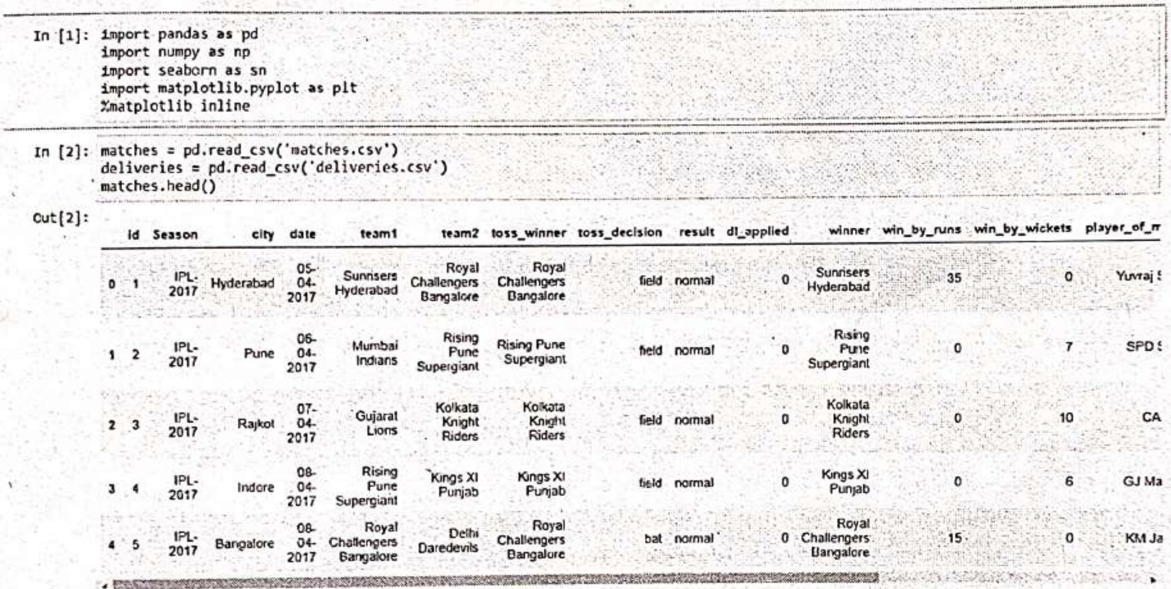


Fig 3.4

Chapter 4:Code for Project

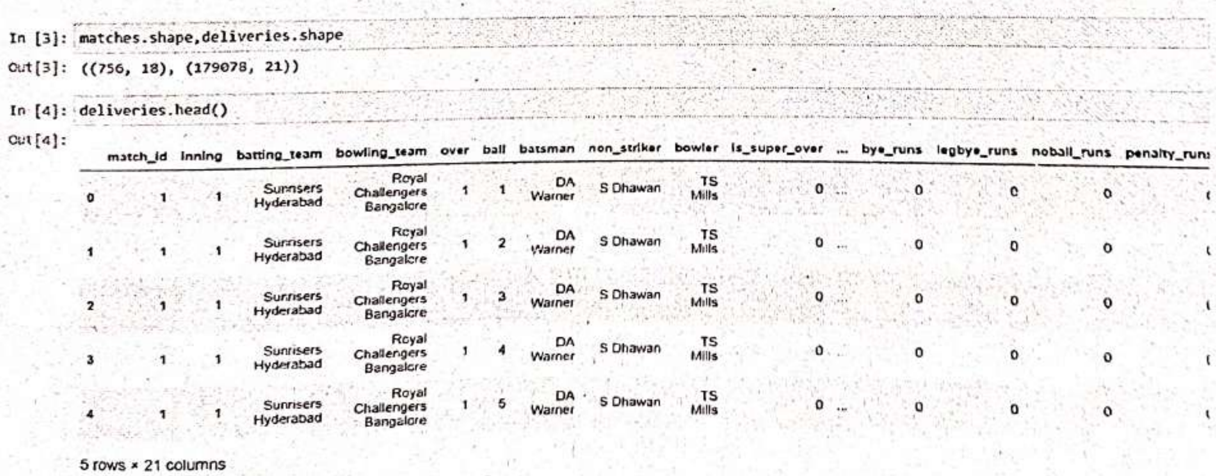
4.1 Data Preparation Code



```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sn
        import matplotlib.pyplot as plt
        %matplotlib inline

In [2]: matches = pd.read_csv('matches.csv')
        deliveries = pd.read_csv('deliveries.csv')
        matches.head()

Out[2]:
```

	id	Season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets	player_of_m
0	1	IPL-2017	Hyderabad	05-04-2017	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	35	0	Yuvraj S
1	2	IPL-2017	Pune	06-04-2017	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	0	Rising Pune Supergiant	0	7	SPD S
2	3	IPL-2017	Rajkot	07-04-2017	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	0	Kolkata Knight Riders	0	10	CA
3	4	IPL-2017	Indore	08-04-2017	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	0	Kings XI Punjab	0	6	GJ Ma
4	5	IPL-2017	Bangalore	08-04-2017	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	0	Royal Challengers Bangalore	15	0	KM Ja

Fig 4.1.1



```
In [3]: matches.shape,deliveries.shape

Out[3]: ((756, 18), (179078, 21))

In [4]: deliveries.head()

Out[4]:
```

	match_id	inning	batting_team	bowling_team	over	ball	batsman	non_striker	bowler	is_super_over	...	bye_runs	legbye_runs	noball_runs	penalty_runs
0	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	1	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	0
1	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	2	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	0
2	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	3	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	0
3	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	4	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	0
4	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	5	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	0

5 rows × 21 columns

In [5]:
```
deliveries.columns
```

Out[5]:
```
Index(['match_id', 'inning', 'batting_team', 'bowling_team', 'over', 'ball',
       'batsman', 'non_striker', 'bowler', 'is_super_over', 'wide_runs',
       'bye_runs', 'legbye_runs', 'noball_runs', 'penalty_runs',
       'batsman_runs', 'extra_runs', 'total_runs', 'player_dismissed',
       'dismissal_kind', 'fielder'],
      dtype='object')
```

In [6]:
```
# grouping the 1st innings,2nd innings score in a particular matchid
# lets say match id = 1,so inning 1 score = 207,inning 2 score = 172,in that way

totalrun_df = deliveries.groupby(['match_id','inning']).sum()['total_runs'].reset_index()

totalrun_df
```

Out[6]:

	match_id	inning	total_runs
0	1	1	207
1	1	2	172
2	2	1	184
3	2	2	187
4	3	1	183
...	...	...	...
1523	11413	2	170
1524	11414	1	155
1525	11414	2	162
1526	11415	1	152
1527	11415	2	157

In [7]:
```
# capturing only the first innings,as we will be predicting for the second innnigs

totalrun_df = totalrun_df[totalrun_df['inning']==1]
totalrun_df['total_runs'] = totalrun_df['total_runs'].apply(lambda x:x+1)
totalrun_df
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_19212\3234705694.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/
rsus-a-copy
  totalrun_df['total_runs'] = totalrun_df['total_runs'].apply(lambda x:x+1)
```

Out[7]:

	match_id	inning	total_runs
0	1	1	208
2	2	1	185
4	3	1	184
6	4	1	164
8	5	1	158
...	...	...	...
1518	11347	1	144
1520	11412	1	137
1522	11413	1	172
1524	11414	1	156
1526	11415	1	153

756 rows × 3 columns

```
In [8]: matches.columns
```

```
Out[8]: Index(['id', 'Season', 'city', 'date', 'team1', 'team2', 'toss_winner',
               'toss_decision', 'result', 'dl_applied', 'winner', 'win_by_runs',
               'win_by_wickets', 'player_of_match', 'venue', 'umpire1', 'umpire2',
               'umpire3'],
              dtype='object')
```

```
In [9]: '''
        Merging the total first innings score df with the matches df,
        where left side merging is done on "id" column of the matches
        and right side merging is done on "match_id" column of the totalrun_df

        '''

        match_df = matches.merge(totalrun_df[['match_id','total_runs']],
                                 left_on='id',right_on='match_id')

        match_df
```

Out[9]:

	id	Season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets
0	1	IPL-2017	Hyderabad	05-04-2017	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	35	0
1	2	IPL-2017	Pune	06-04-2017	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	0	Rising Pune Supergiant	0	7
2	3	IPL-2017	Rajkot	07-04-2017	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	0	Kolkata Knight Riders	0	10

```
In [10]: match_df['team1'].unique()
```

```
Out[10]: array(['Sunrisers Hyderabad', 'Mumbai Indians', 'Gujarat Lions',
                'Rising Pune Supergiant', 'Royal Challengers Bangalore',
                'Kolkata Knight Riders', 'Delhi Daredevils', 'Kings XI Punjab',
                'Chennai Super Kings', 'Rajasthan Royals', 'Deccan Chargers',
                'Kochi Tuskers Kerala', 'Pune Warriors', 'Rising Pune Supergiants',
                'Delhi Capitals'], dtype=object)
```

```
In [11]: teams = [
             'Sunrisers Hyderabad',
             'Mumbai Indians',
             'Royal Challengers Bangalore',
             'Kolkata Knight Riders',
             'Kings XI Punjab',
             'Chennai Super Kings',
             'Rajasthan Royals',
             'Delhi Capitals'
         ]
```

```
In [12]: # replacing the Delhi Daredevils with Delhi Capitals

         match_df['team1'] = match_df['team1'].str.replace('Delhi Daredevils','Delhi Capitals')
         match_df['team2'] = match_df['team2'].str.replace('Delhi Daredevils','Delhi Capitals')


         # replacing the Deccan Chargers with Sunrises Hyderabad

         match_df['team1'] = match_df['team1'].str.replace('Deccan Chargers','Sunrisers Hyderabad')
         match_df['team2'] = match_df['team2'].str.replace('Deccan Chargers','Sunrisers Hyderabad')
```

```
In [13]: # will consider only frequently occuring teams,
         # which are mentioned in the teams list

         match_df = match_df[match_df['team1'].isin(teams)]
         match_df = match_df[match_df['team2'].isin(teams)]

         match_df['team1'].unique()

Out[13]: array(['Sunrisers Hyderabad', 'Royal Challengers Bangalore',
                'Kolkata Knight Riders', 'Kings XI Punjab', 'Delhi Capitals',
                'Mumbai Indians', 'Chennai Super Kings', 'Rajasthan Royals'],
               dtype=object)
```

```
In [14]: match_df.shape

Out[14]: (641, 20)
```

```
In [15]: match_df.head()

Out[15]:
```

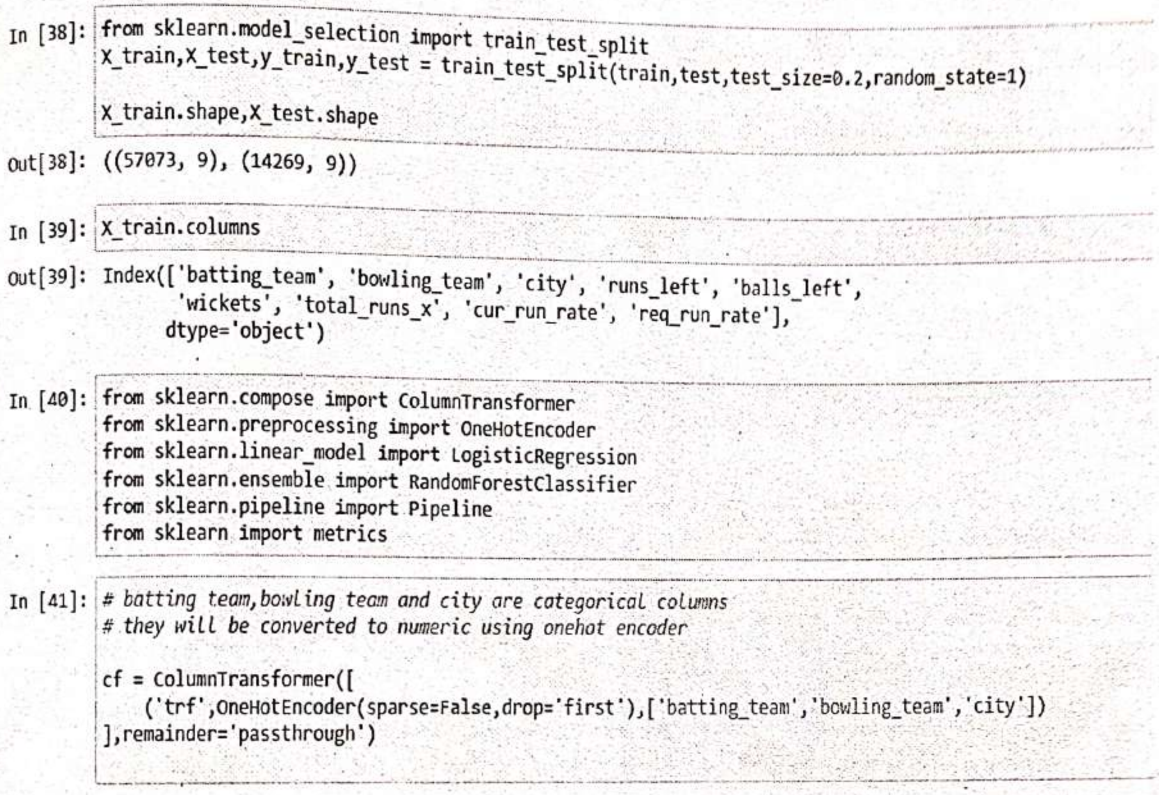
	id	Season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by
0	1	IPL-2017	Hyderabad	05-04-2017	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	
4	5	IPL-2017	Bangalore	08-04-2017	Royal Challengers Bangalore	Delhi Capitals	Royal Challengers Bangalore	bat	normal	0	Royal Challengers Bangalore	
6	7	IPL-2017	Mumbai	09-04-2017	Kolkata Knight Riders	Mumbai Indians	Mumbai Indians	field	normal	0	Mumbai Indians	
7	8	IPL-2017	Indore	10-04-2017	Royal Challengers Bangalore	Kings XI Punjab	Royal Challengers Bangalore	bat	normal	0	Kings XI Punjab	

```
In [16]: # checking the matches which resulted in dl method

         match_df[match_df['dl_applied']==1].style.background_gradient(cmap = 'plasma')

Out[16]:
```

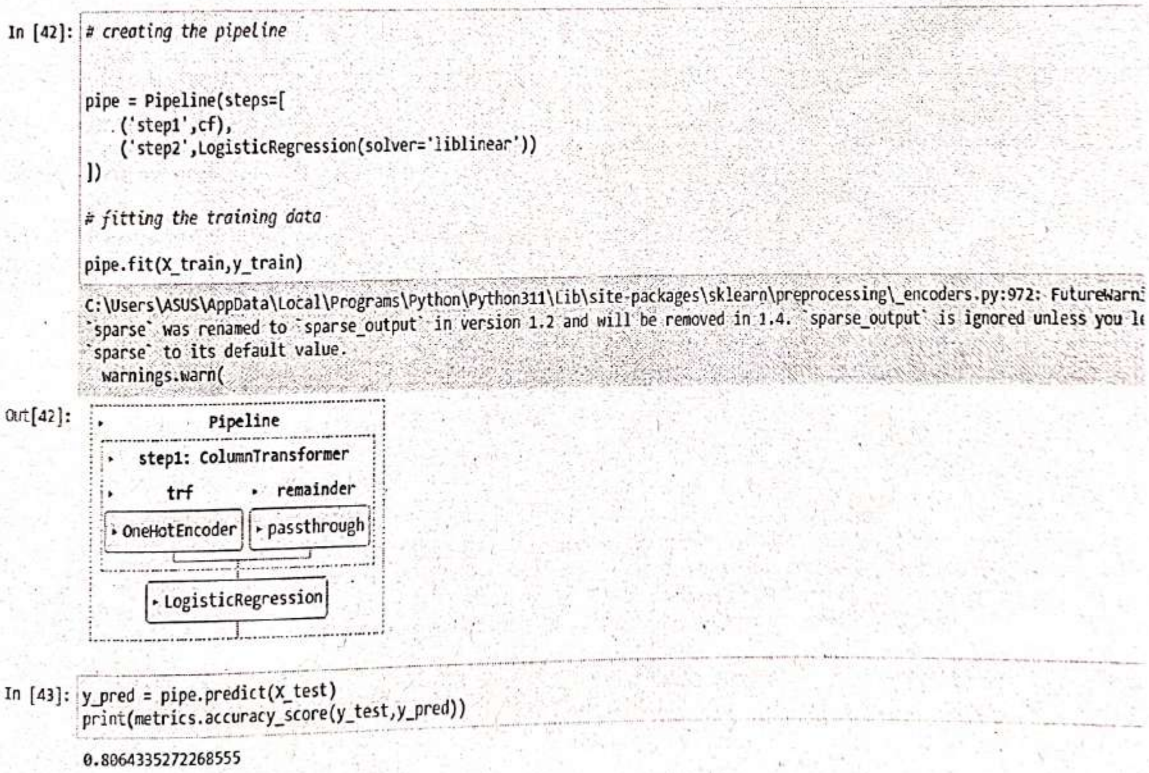
	id	Season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets
56	57	IPL-2017	Bangalore	17-05-2017	Sunrisers Hyderabad	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	1	Kolkata Knight Riders	0	7
99	100	IPL-2008	Delhi	17-05-2008	Delhi Capitals	Kings XI Punjab	Delhi Daredevils	bat	normal	1	Kings XI Punjab	6	0
102	103	IPL-2008	Kolkata	18-05-2008	Kolkata Knight Riders	Chennai Super Kings	Kolkata Knight Riders	bat	normal	1	Chennai Super Kings	3	0
119	120	IPL-2009	Cape Town	19-04-2009	Kings XI Punjab	Delhi Capitals	Delhi Daredevils	field	normal	1	Delhi Daredevils	0	10
122	123	IPL-2009	Durban	21-04-2009	Kings XI Punjab	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	1	Kolkata Knight Riders	11	0

```
In [17]: # ignoring the rows which were DL method

         match_df = match_df[match_df['dl_applied']==0]

         # considering the match_id,city,winner and total runs

         match_df = match_df[['match_id','city','winner','total_runs']]

         match_df
```

4.2 Model Training Code:



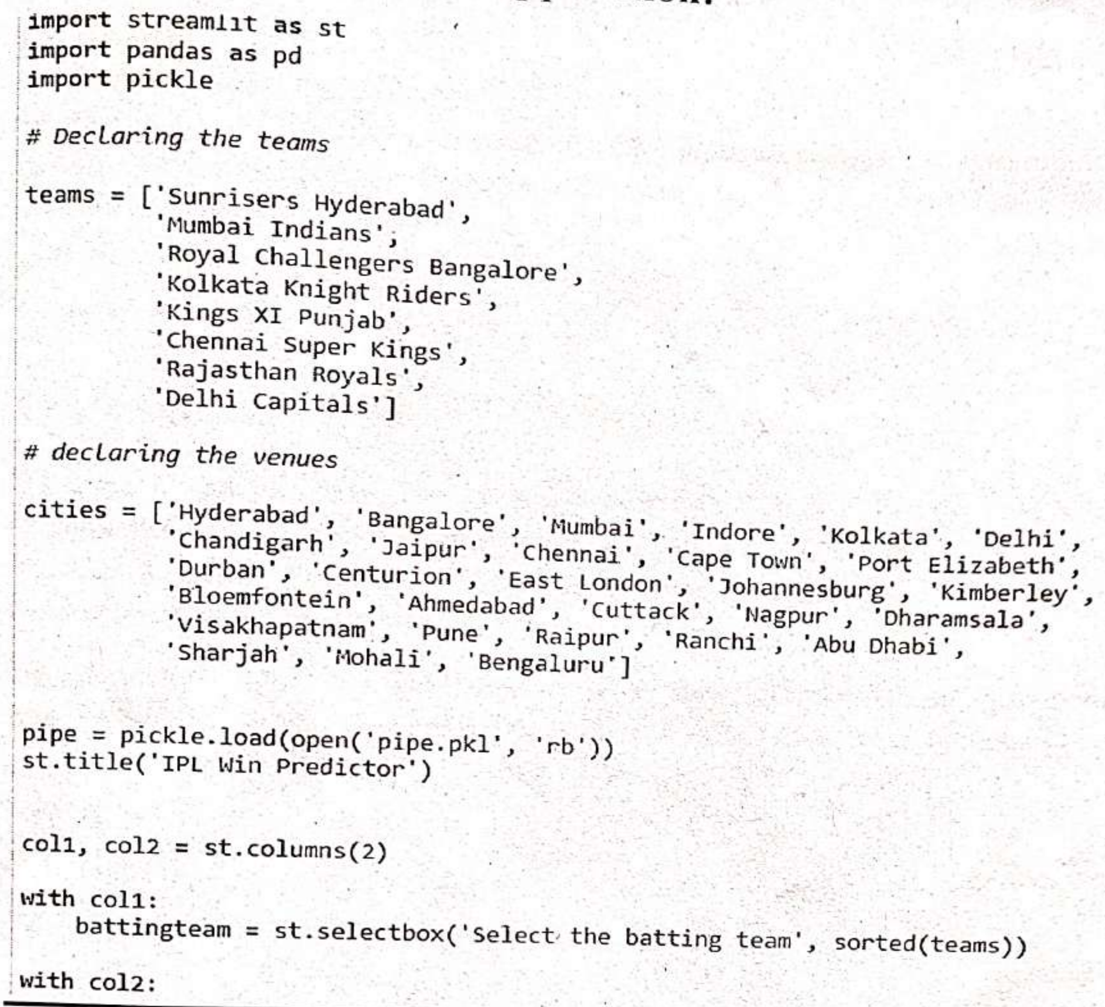
```
In [38]: from sklearn.model_selection import train_test_split
         X_train,X_test,y_train,y_test = train_test_split(train,test,test_size=0.2,random_state=1)

         X_train.shape,X_test.shape

Out[38]: ((57073, 9), (14269, 9))

In [39]: X_train.columns

Out[39]: Index(['batting_team', 'bowling_team', 'city', 'runs_left', 'balls_left',
                'wickets', 'total_runs_x', 'cur_run_rate', 'req_run_rate'],
               dtype='object')

In [40]: from sklearn.compose import ColumnTransformer
         from sklearn.preprocessing import OneHotEncoder
         from sklearn.linear_model import LogisticRegression
         from sklearn.ensemble import RandomForestClassifier
         from sklearn.pipeline import Pipeline
         from sklearn import metrics

In [41]: # batting team,bowling team and city are categorical columns
         # they will be converted to numeric using onehot encoder

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



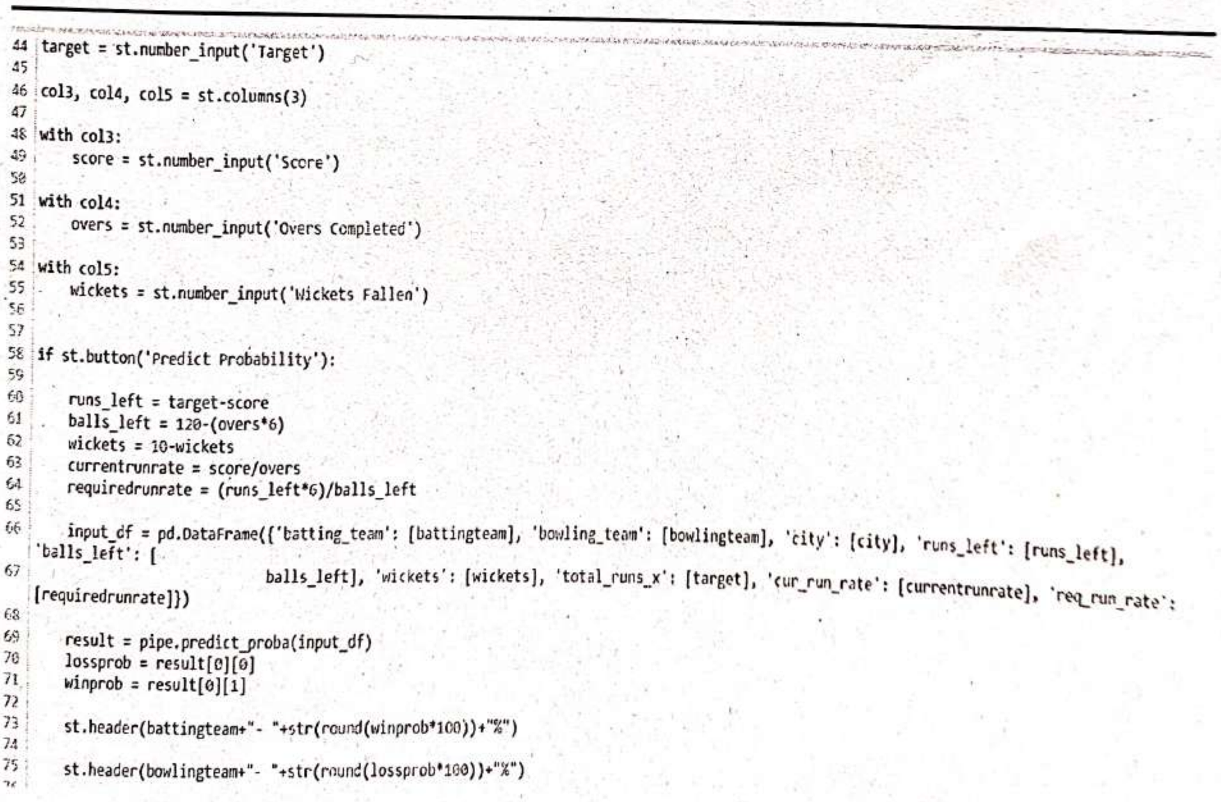
```
In [42]: # creating the pipeline


         pipe = Pipeline(steps=[
             ('step1',cf),
             ('step2',LogisticRegression(solver='liblinear'))
         ])

         # fitting the training data

         pipe.fit(X_train,y_train)

         C:\Users\ASUS\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\preprocessing\_encoders.py:972: FutureWarni
         `sparse` was renamed to `sparse_output` in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you le
         `sparse` to its default value.
           warnings.warn(

Out[42]:
```

```
In [43]: y_pred = pipe.predict(X_test)
         print(metrics.accuracy_score(y_test,y_pred))

         0.8064335272268555
```

4.3 Creating Model Web Application:



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

# Declaring the teams

teams = ['Sunrisers Hyderabad',
         'Mumbai Indians',
         'Royal Challengers Bangalore',
         'Kolkata Knight Riders',
         'Kings XI Punjab',
         'Chennai Super Kings',
         'Rajasthan Royals',
         'Delhi Capitals']

# declaring the venues

cities = ['Hyderabad', 'Bangalore', 'Mumbai', 'Indore', 'Kolkata', 'Delhi',
          'Chandigarh', 'Jaipur', 'Chennai', 'Cape Town', 'Port Elizabeth',
          'Durban', 'Centurion', 'East London', 'Johannesburg', 'Kimberley',
          'Bloemfontein', 'Ahmedabad', 'Cuttack', 'Nagpur', 'Dharamsala',
          'Visakhapatnam', 'Pune', 'Raipur', 'Ranchi', 'Abu Dhabi',
          'Sharjah', 'Mohali', 'Bengaluru']


pipe = pickle.load(open('pipe.pkl', 'rb'))
st.title('IPL Win Predictor')


col1, col2 = st.columns(2)

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

with col2:
```



```
target = st.number_input('Target')

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

with col3:
    score = st.number_input('Score')

with col4:
    overs = st.number_input('Overs Completed')

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


if st.button('Predict Probability'):

    runs_left = target-score
    balls_left = 120-(overs*6)
    wickets = 10-wickets
    currentrunrate = score/overs
    requiredrunrate = (runs_left*6)/balls_left

    input_df = pd.DataFrame({'batting_team': [battingteam], 'bowling_team': [bowlingteam], 'city': [city], 'runs_left': [runs_left],
'balls_left': [
                              balls_left], 'wickets': [wickets], 'total_runs_x': [target], 'cur_run_rate': [currentrunrate], 'req_run_rate':
[requiredrunrate]})

    result = pipe.predict_proba(input_df)
    lossprob = result[0][0]
    winprob = result[0][1]

    st.header(battingteam+"- "+str(round(winprob*100))+"%")

    st.header(bowlingteam+"- "+str(round(lossprob*100))+"%")
```

Chapter 5: Final analysis and design

5.1 Result :

The end result of the " IPL Match Outcome Predictor" project is a web-based application that allows users to input specific match conditions for an IPL cricket match and receive predictions regarding the probability of a team winning or losing the match.

5.2 Conclusion

The " IPL Match Win Predictor " is a practical demonstration of how data science and technology can be leveraged to make complex predictions accessible to a wide audience. It stands as a testament to the expanding horizons of technology, with applications extending beyond traditional domains into the realm of sports.

The scope of the " IPL Match Win Predictor " project extends beyond its initial implementation, offering various opportunities and potential directions for future development and expansion.

5.3 Refrence:

Link 1:https://youtu.be/Ok_zkfWC0gI?si=bKTuQTGOyanUow1S

Link2:https://www.analyticsvidhya.com/blog/2022/05/ipl-team-win-prediction-project-using-machine-learning/