IPL DATA ANALYSIS

GALGOTIAS UNIVERSITY

IPL

AKASH: 23SCSE1011203
ONKAR DUREJA: 23SCSE1180036

SANDEEP YADAV: 23SCSE1010514
YASH KUMAR GUPTA: 23SCSE1180115

INTRODUCTION

The Indian Premier League (IPL) is one of the most popular and competitive cricket leagues globally. With rich historical data and numerous matches played over the years, it provides a great opportunity for data analysis. This project involves analyzing IPL data to uncover trends, performance metrics, and winning strategies using data science techniques.

This project explores IPL match and delivery data using Python tools. We perform exploratory data analysis (EDA), visualize key metrics, and identify patterns among teams and players. The goal is to derive meaningful insights that help understand the game better from a data perspective.

PROBLEM STATEMENT

How can historical IPL data be analyzed to:

Understand team and player performances?

Discover the impact of toss decisions and venues?

Identify factors contributing to winning matches?

By answering these questions, we aim to turn raw data into actionable insights.

Why We choose this ?

Cricket is a passion for millions, and IPL data is widely available and rich in patterns. Being a cricket enthusiast and aspiring data analyst, we chose this project to combine both interests. It also serves as an ideal case for applying and improving our data analysis and visualization skills.

Project Scope

1. Analyze IPL matches from 2008 to recent seasons.
2. Focus on team performance, player stats, and match outcomes.
3. Visualize findings using charts and graphs.
4. Scope does not include prediction or live data analysis (but can be extended in the future).

DA.py

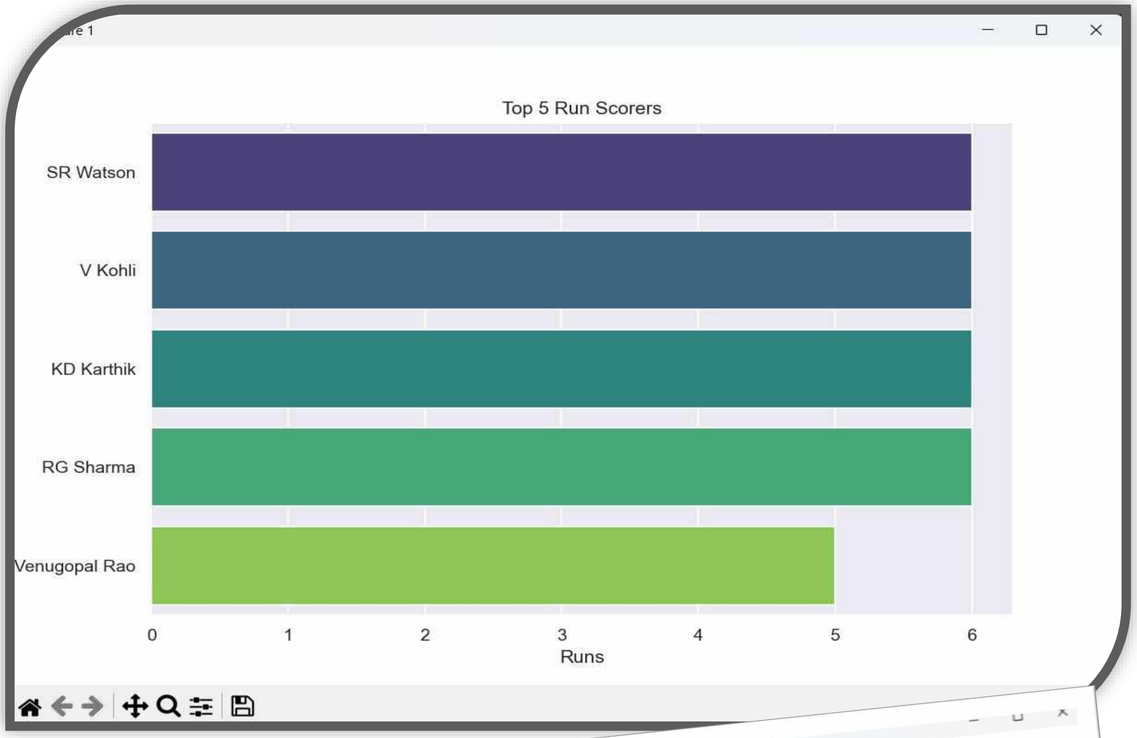
```python
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')

# Set plot styles
sns.set(style='darkgrid')
plt.rcParams['figure.figsize'] = (12, 6)

# Load datasets
matches = pd.read_csv('matches.csv')
deliveries = pd.read_csv('deliveries.csv')

# Display initial data overview
print("Matches Dataset:")
print(matches.head())
print("\nDeliveries Dataset:")
print(deliveries.head())

# ------------------------------
# 1. Data Cleaning and Handling Missing Values
# ------------------------------

# Check for missing values
print("\nMissing values in matches dataset:")
print(matches.isnull().sum())
print("\nMissing values in deliveries dataset:")
print(deliveries.isnull().sum())

# Fill missing 'player_of_match' with 'Unknown'
matches['player_of_match'].fillna('Unknown', inplace=True)

# Drop rows with missing values in deliveries
deliveries.dropna(inplace=True)

# ------------------------------
# 2. Feature Selection and Engineering
# ------------------------------
```

DA.py

```python

    # Identify outliers in 'total_runs' using IQR
    Q1 = deliveries['total_runs'].quantile(0.25)
    Q3 = deliveries['total_runs'].quantile(0.75)
    IQR = Q3 - Q1
    outliers = deliveries[(deliveries['total_runs'] < Q1 - 1.5 * IQR) | (deliveries['total_runs'] > Q3 + 1.5 * IQR)]
    print(f"\nNumber of outlier deliveries: {outliers.shape[0]}")

    # ------------------------------
    # 7. Initial Visual Representation of Key Findings
    # ------------------------------

    # Top 5 run scorers bar plot
    plt.figure(figsize=(10,6))
    sns.barplot(x=top_scorers.values, y=top_scorers.index, palette='viridis')
    plt.title('Top 5 Run Scorers')
    plt.xlabel('Runs')
    plt.ylabel('Batsman')
    plt.show()

    # Runs per over line plot
    plt.figure(figsize=(10,6))
    sns.lineplot(x=runs_per_over.index, y=runs_per_over.values, marker='o')
    plt.title('Average Runs per Over')
    plt.xlabel('Over')
    plt.ylabel('Average Runs')
    plt.show()
```

DA.py

```python
    # Convert 'date' column to datetime
    matches['date'] = pd.to_datetime(matches['date'], dayfirst=True, errors='coerce')


    # Create a new feature: 'match_year'
    matches['match_year'] = matches['date'].dt.year

    # Merge matches and deliveries data on 'match_id'
    combined_df = deliveries.merge(matches, left_on='match_id', right_on='id')


    # Calculate total runs per match
    total_runs_per_match = combined_df.groupby('match_id')['total_runs'].sum().reset_index()
    total_runs_per_match.rename(columns={'total_runs': 'total_runs_in_match'}, inplace=True)

    # Merge total runs per match back to matches dataframe
    matches = matches.merge(total_runs_per_match, left_on='id', right_on='match_id')


    # ------------------------------
    # 3. Ensuring Data Integrity and Consistency
    # ------------------------------

    # Standardize team names
    team_replacements = {
        'Delhi Daredevils': 'Delhi Capitals',
        'Kings XI Punjab': 'Punjab Kings'
    }
    matches.replace({'team1': team_replacements, 'team2': team_replacements, 'winner': team_replacements}, inplace=True)
    deliveries.replace({'batting_team': team_replacements, 'bowling_team': team_replacements}, inplace=True)

    # ------------------------------
    # 4. Summary Statistics and Insights
    # ------------------------------

    # Total matches played
    total_matches = matches.shape[0]
    print(f"\nTotal matches played: {total_matches}")

    # Total runs scored
    total_runs = deliveries['total_runs'].sum()
    print(f"Total runs scored: {total_runs}")
```


Top 5 Run Scorers
SR Watson
V Kohli
KD Karthik
RG Sharma
Venugopal Rao
0
1
2
3
4
5
6
Runs

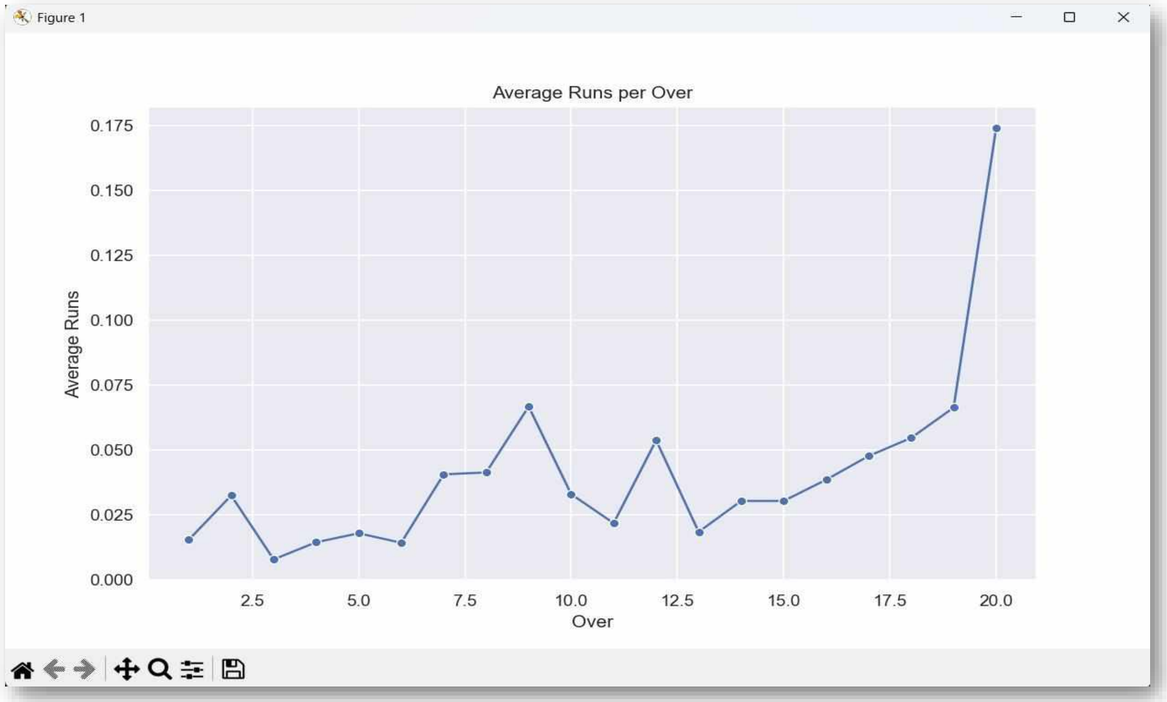
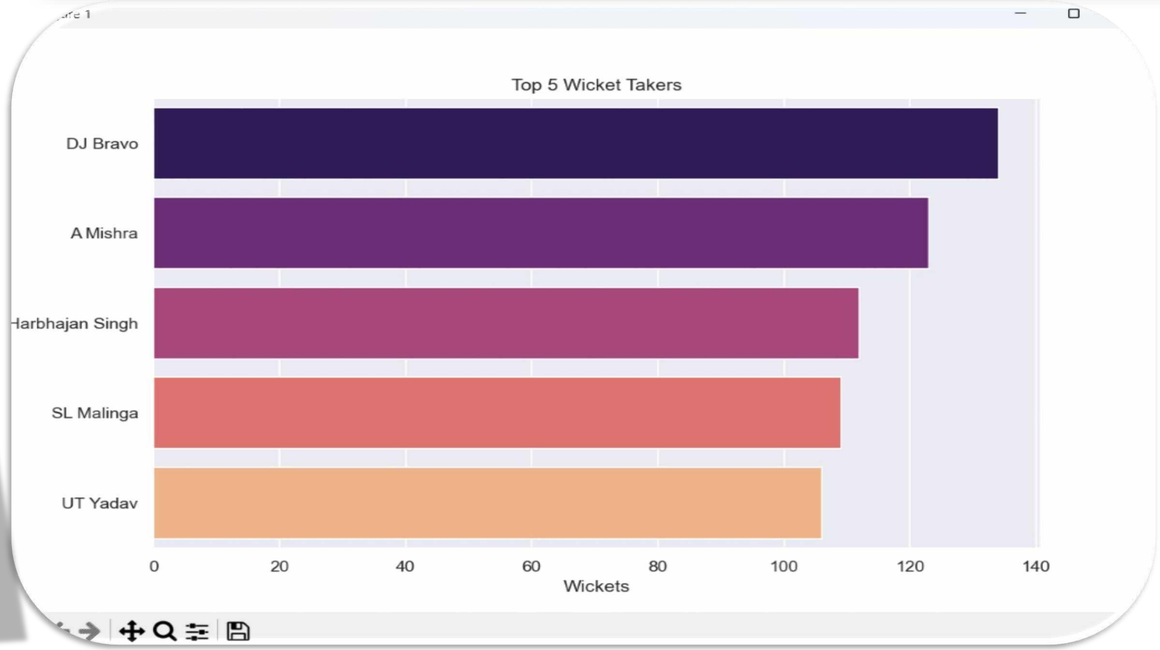
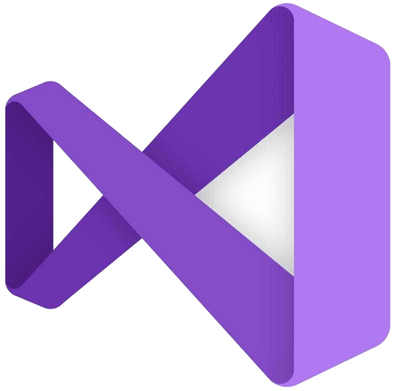


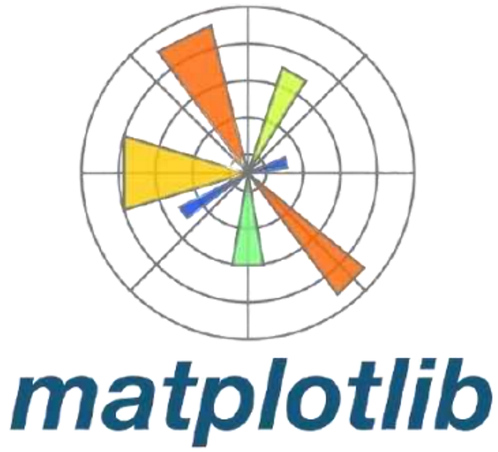
Figure 1
Average Runs per Over
Average Runs
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Over


Figure 1
Total Runs by Each Team
kata Knight Riders
llengers Bangalore
Mumbai Indians
Rajasthan Royals
Punjab Kings
Delhi Capitals
ennai Super Kings
nrisers Hyderabad
Deccan Chargers
Pune Warriors
Gujarat Lions
chi Tuskers Kerala
g Pune Supergiant
Pune Supergiants
0
5
10
15
20
25
30
35
40
Total Runs


Top 5 Wicket Takers
DJ Bravo
A Mishra
Harbhajan Singh
SL Malinga
UT Yadav
0
20
40
60
80
100
120
140
Wickets

TECHNOLOGY USED





KEY CONCEPT USED

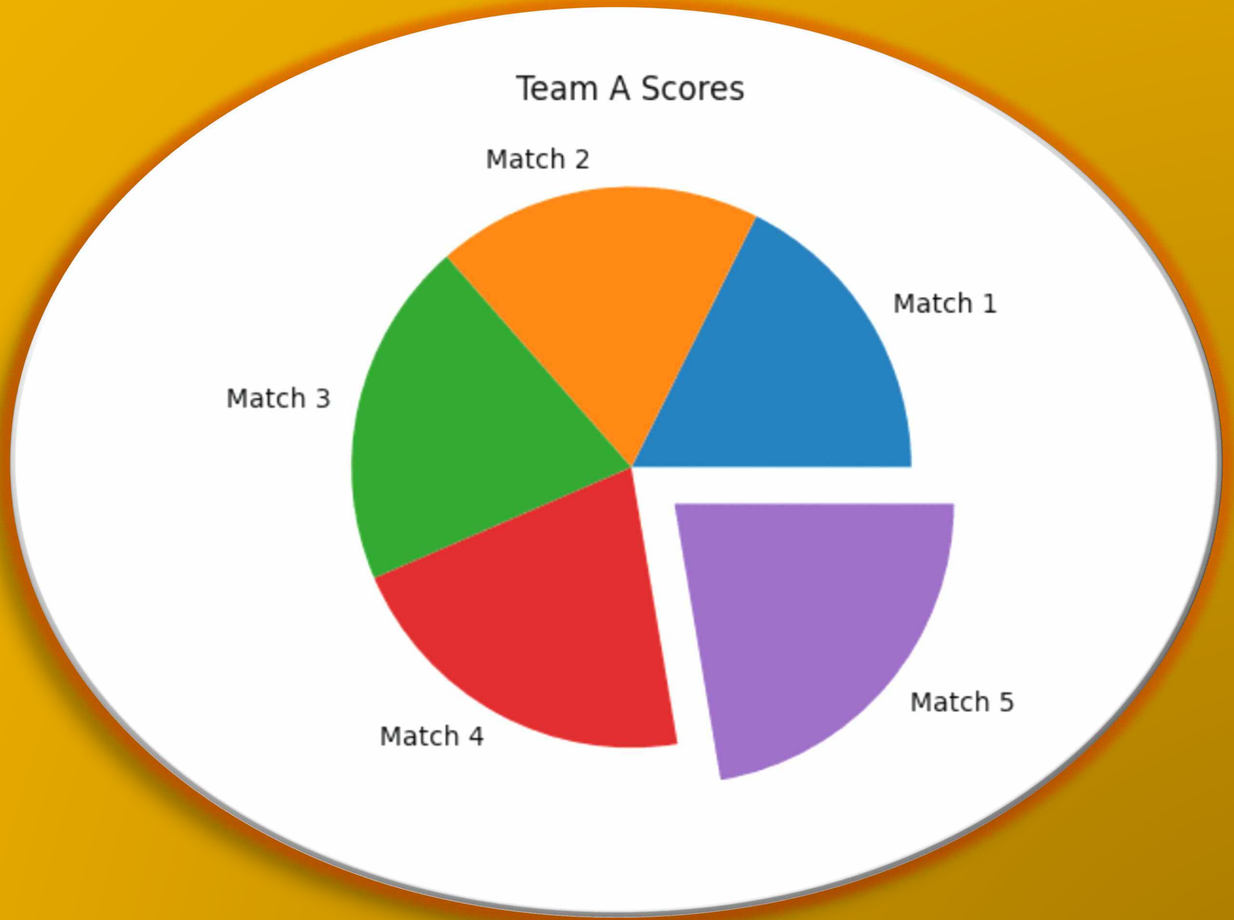
I. Data Cleaning: Handling null values, formatting
II. Exploratory Data Analysis (EDA): Summary statistics, data grouping
III. Visualization: Bar plots, line graphs, heatmaps
IV. Aggregation: GroupBy functions for team and player metrics
V. Comparative Analysis: Toss vs result, batting first vs second, etc.

CHALLENGES FACED

- Inconsistent data formats in early seasons
- Handling large delivery-level dataset efficiently
- Interpreting cricket-specific statistics in data context
- Choosing the right visualizations to represent insights clearly

CONCLUSION

This project provided valuable insights into IPL matches and player performance. It reinforced the importance of data preprocessing and visualization in uncovering patterns. The project also strengthened our analytical thinking and Python skills, while opening possibilities for future work in predictive modeling.



THANK YOU