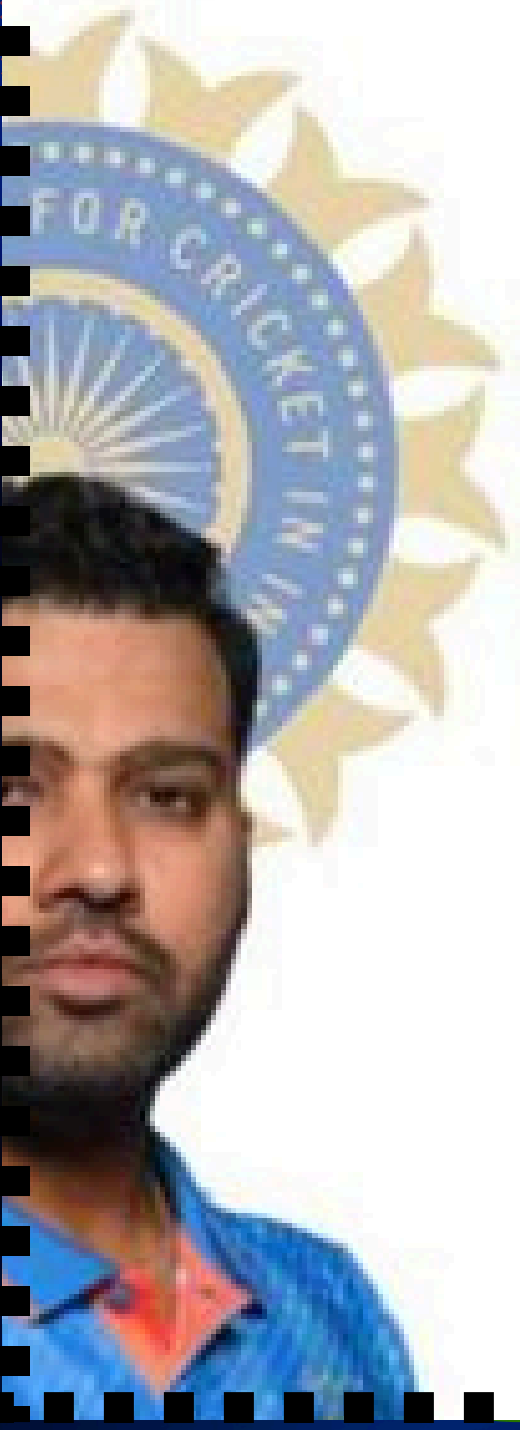
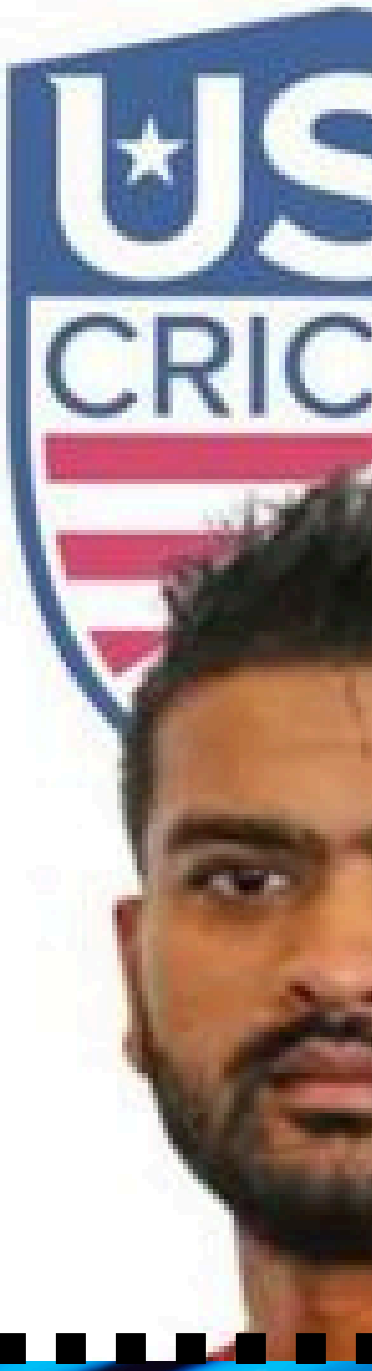
T20 WORLD CUP 2024

UJJAWAL VALIYA PROJECT





TEAM INDIA VS TEAM USA

ABOUT THE PROJECT

BELOW IS INFORMATION ABOUT ALL THE COLUMNS IN THE DATASET:

BATTER: THE NAME OF THE BATSMAN.
BOWLER: THE NAME OF THE BOWLER.
NON_STRIKER: THE NAME OF THE NON-STRIKER BATSMAN.
RUNS_BATTER: THE RUNS SCORED BY THE BATSMAN ON THAT BALL.
RUNS_EXTRAS: THE EXTRA RUNS GIVEN (LIKE WIDES, NO-BALLS).
RUNS_TOTAL: THE TOTAL RUNS SCORED ON THAT BALL (BATSMAN RUNS + EXTRAS).
WICKETS_0_PLAYER_OUT: THE NAME OF THE PLAYER WHO GOT OUT.
WICKETS_0_KIND: THE MODE OF DISMISSAL (E.G., LBW, BOWLED).
TEAM: THE TEAM PLAYING THE INNINGS.
OVER: THE OVER NUMBER IN WHICH THE BALL WAS BOWLED.
BALL: THE BALL NUMBER IN THAT OVER.
EXTRAS_WIDES: EXTRA RUNS DUE TO WIDE BALLS.
EXTRAS_BYES: EXTRA RUNS DUE TO BYES.
EXTRAS_NOBALLS: EXTRA RUNS DUE TO NO-BALLS.
EXTRAS_LEGBYES: EXTRA RUNS DUE TO LEG BYES.
EXTRAS_PENALTY: PENALTY RUNS AWARDED.
WICKETS_0_FIELDERS_0_NAME: NAME OF THE FIELDER INVOLVED IN THE DISMISSAL (IF ANY).
WICKETS_0_FIELDERS_1_NAME: ADDITIONAL FIELDER INVOLVED IN THE DISMISSAL.
REVIEW_BY: THE TEAM OR PLAYER REQUESTING A REVIEW.
REVIEW_UMPIRE: THE UMPIRE INVOLVED IN THE REVIEW.
REVIEW_BATTER: THE BATSMAN INVOLVED IN THE REVIEW.
REVIEW_DECISION: THE DECISION MADE AFTER THE REVIEW.
REVIEW_TYPE: THE TYPE OF REVIEW (E.G., DRS).

```
import pandas as pd

data = pd.read_csv("india-usa_innings_data.csv")

print(data.head())
```

```
           batter          bowler non_striker  runs_batter  runs_extras  \
0  Shayan Jahangir  Arshdeep Singh   SR Taylor            0            0
1         AGS Gous  Arshdeep Singh   SR Taylor            0            0
2         AGS Gous  Arshdeep Singh   SR Taylor            0            0
3         AGS Gous  Arshdeep Singh   SR Taylor            0            1
4         AGS Gous  Arshdeep Singh   SR Taylor            2            0

   runs_total wickets_0_player_out wickets_0_kind                    team  \
0           0      Shayan Jahangir            lbw  United States of America
1           0                  NaN            NaN  United States of America
2           0                  NaN            NaN  United States of America
3           1                  NaN            NaN  United States of America
4           2                  NaN            NaN  United States of America

   over  ...  wickets_0_fielders_0_name review_by review_umpire review_batter  \
0     0  ...                       NaN       NaN           NaN            NaN
1     0  ...                       NaN       NaN           NaN            NaN
2     0  ...                       NaN       NaN           NaN            NaN
3     0  ...                       NaN       NaN           NaN            NaN
4     0  ...                       NaN       NaN           NaN            NaN

  review_decision review_type extras_legbyes  wickets_0_fielders_1_name  \
0             NaN         NaN            NaN                        NaN
1             NaN         NaN            NaN                        NaN
2             NaN         NaN            NaN                        NaN
3             NaN         NaN            NaN                        NaN
4             NaN         NaN            NaN                        NaN

  extras_noballs  extras_penalty
0            NaN             NaN
1            NaN             NaN
2            NaN             NaN
3            NaN             NaN
4            NaN             NaN
```

Let's have a look at the missing values and data types:

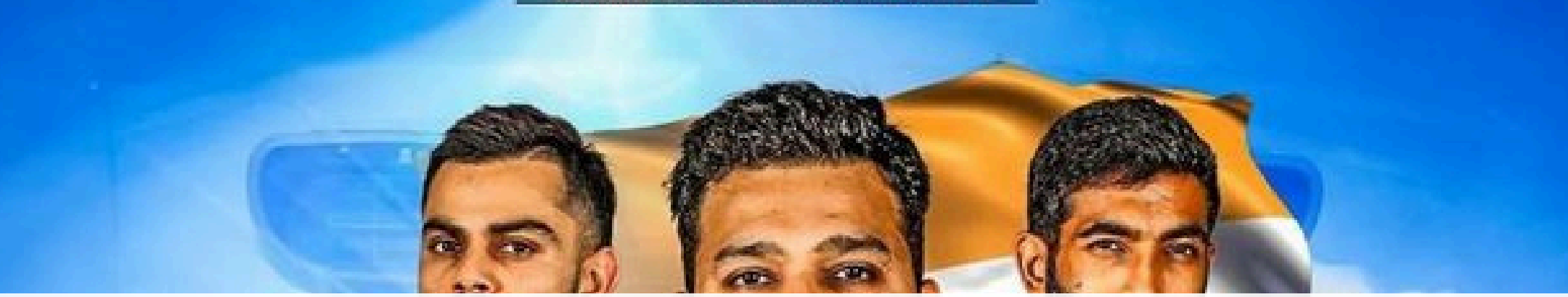
```
[ ]: # checking for missing values in the dataset
     missing_values = data.isnull().sum()

     # checking data types of the columns
     data_types = data.dtypes

     missing_values
```

```
[ ]: batter                      0
     bowler                      0
     non_striker                 0
     runs_batter                 0
     runs_extras                 0
     runs_total                  0
     wickets_0_player_out      225
     wickets_0_kind            225
     team                        0
     over                        0
     extras_wides              231
     wickets_0_fielders_0_name 228
     review_by                 235
     review_umpire             235
     review_batter             235
     review_decision           235
     review_type               235
     extras_legbyes            234
     wickets_0_fielders_1_name 235
     extras_noballs            235
     extras_penalty            235
     dtype: int64
```

data_types

```
batter                    object
bowler                    object
non_striker               object
runs_batter                int64
runs_extras                int64
runs_total                 int64
wickets_0_player_out      object
wickets_0_kind            object
team                      object
over                       int64
extras_wides             float64
wickets_0_fielders_0_name  object
review_by                  object
review_umpire              object
review_batter              object
review_decision            object
review_type                object
extras_legbyes           float64
wickets_0_fielders_1_name  object
extras_noballs           float64
extras_penalty           float64
dtype: object
```

```python
# total runs scored by each team
total_runs = data.groupby('team')['runs_total'].sum()

# total wickets taken by each team
total_wickets = data['wickets_0_player_out'].notna().groupby(data['team']).sum()

# total extras
total_extras = data[['team', 'runs_extras', 'extras_wides', 'extras_noballs', 'extras_legbyes', 'extras_penalty']].groupby('team').sum()

# runs scored by each batter
batter_runs = data.groupby('batter')['runs_batter'].sum()

# balls faced by each batter
balls_faced = data.groupby('batter').size()

# strike rate of each batter
strike_rate = (batter_runs / balls_faced) * 100

# boundaries hit by each batter
boundaries = data[(data['runs_batter'] == 4) | (data['runs_batter'] == 6)].groupby(['batter', 'runs_batter']).size().unstack(fill_value=0)

# wickets taken by each bowler
wickets_taken = data['wickets_0_player_out'].notna().groupby(data['bowler']).sum()

# runs conceded by each bowler
runs_conceded = data.groupby('bowler')['runs_total'].sum()

# balls bowled by each bowler
balls_bowled = data.groupby('bowler').size()

# economy rate of each bowler
economy_rate = runs_conceded / (balls_bowled / 6)

# dott balls bowled by each bowler
dot_balls = data[data['runs_total'] == 0].groupby('bowler').size()

# combine all these statistics into dataframes for batters and bowlers
batter_stats = pd.DataFrame({
    'Runs': batter_runs,
    'Balls Faced': balls_faced,
    'Strike Rate': strike_rate,
}).join(boundaries)

bowler_stats = pd.DataFrame({
    'Wickets': wickets_taken,
    'Runs Conceded': runs_conceded,
    'Balls Bowled': balls_bowled,
    'Economy Rate': economy_rate,
    'Dot Balls': dot_balls,
})
```



I have grouped the data that we needed to analyze this data properly. You can look at all grouped insights one by one. Here's an example:

```
total_runs
```

```
team
India                          111
United States of America       110
Name: runs_total, dtype: int64
```

```
total_wickets
```

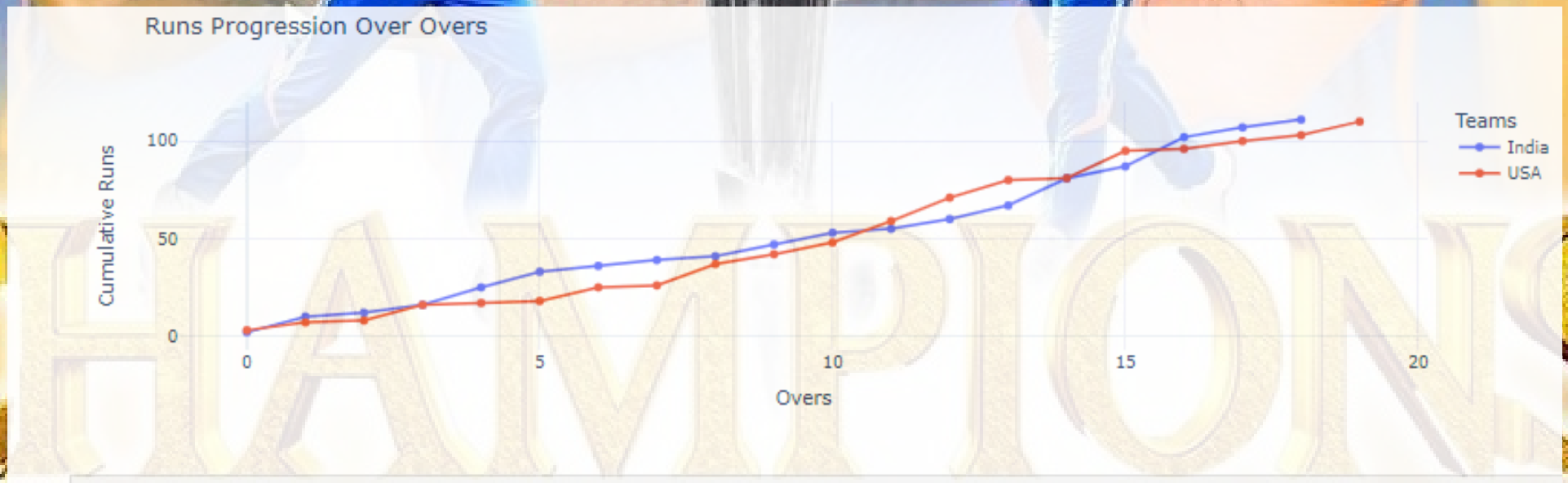
```
team
India                          3
United States of America       8
Name: wickets_0_player_out, dtype: int64
```

```
total_extras
```

team	runs_extras	extras_wides	extras_noballs	extras_legbyes	extras_penalty
India	9	2.0	1.0	1.0	5.0
United States of America	8	7.0	0.0	1.0	0.0

Now, let's have a look at the progression of the run over overs:

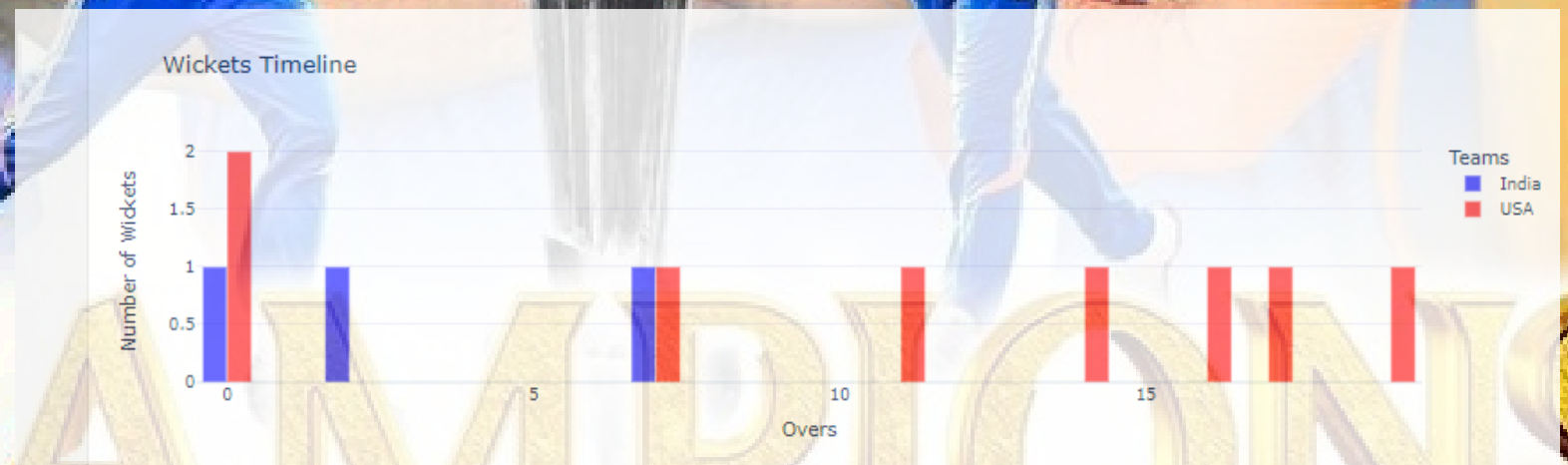
```
import plotly.graph_objects as go

india_runs_progression = data[data['team'] == 'India'].groupby('over')['runs_total'].sum().cumsum()
usa_runs_progression = data[data['team'] == 'United States of America'].groupby('over')['runs_total'].sum().cumsum()

fig = go.Figure()

fig.add_trace(go.Scatter(
    x=india_runs_progression.index,
    y=india_runs_progression.values,
    mode='lines+markers',
    name='India'
))

fig.add_trace(go.Scatter(
    x=usa_runs_progression.index,
    y=usa_runs_progression.values,
    mode='lines+markers',
    name='USA'
))

fig.update_layout(
    title='Runs Progression Over Overs',
    xaxis_title='Overs',
    yaxis_title='Cumulative Runs',
    legend_title='Teams',
    template='plotly_white'
)

fig.show()
```



[ ]:
```
The graph shows the progression of the cumulative run over the overs for both India and the USA in their T20 World Cup match.
Initially, both teams had a steady run rate, with India slightly ahead in the early overs.
As the innings progressed, USA gained momentum and took the lead briefly around the middle overs.
However, India accelerated their scoring in the later overs, surpassing the USA and maintaining the lead until the end.
The key takeaway is India's strong finish, which enabled them to secure the win by consistently increasing their run rate in the final overs.
```

Now, let's have a look at the wickets timeline:

```
india_wickets = data[(data['team'] == 'India') & data['wickets_0_player_out'].notna()].groupby('over').size()
usa_wickets = data[(data['team'] == 'United States of America') & data['wickets_0_player_out'].notna()].groupby('over').size()

fig = go.Figure()

fig.add_trace(go.Bar(
    x=india_wickets.index,
    y=india_wickets.values,
    name='India',
    marker_color='blue',
    opacity=0.7
))

fig.add_trace(go.Bar(
    x=usa_wickets.index,
    y=usa_wickets.values,
    name='USA',
    marker_color='red',
    opacity=0.7
))

fig.update_layout(
    title='Wickets Timeline',
    xaxis_title='Overs',
    yaxis_title='Number of Wickets',
    barmode='group',
    template='plotly_white',
    legend_title='Teams'
)

fig.show()
```



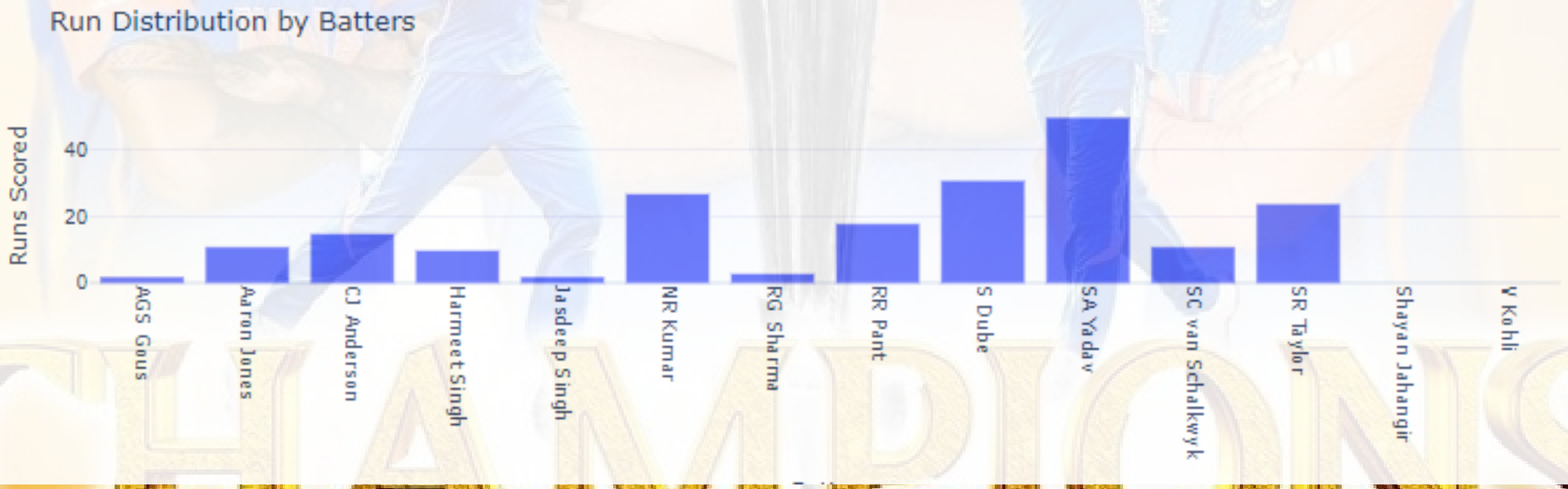
[ ]:
The wickets timeline graph illustrates the distribution of wickets taken over the overs for both India and the USA.
The USA lost wickets more frequently, especially in the early overs, with two wickets falling in the first over,
followed by consistent wicket losses throughout their innings.

In contrast, India experienced their wicket losses more evenly spread across their innings, with a couple of early wickets but maintaining longer partnerships in the middle overs.

The frequent loss of wickets by the USA disrupted their momentum, while India's ability to avoid clusters of wickets falling in succession helped them maintain a steady scoring rate and ultimately secure the win.

Now, let's have a look at the run distribution by batters:

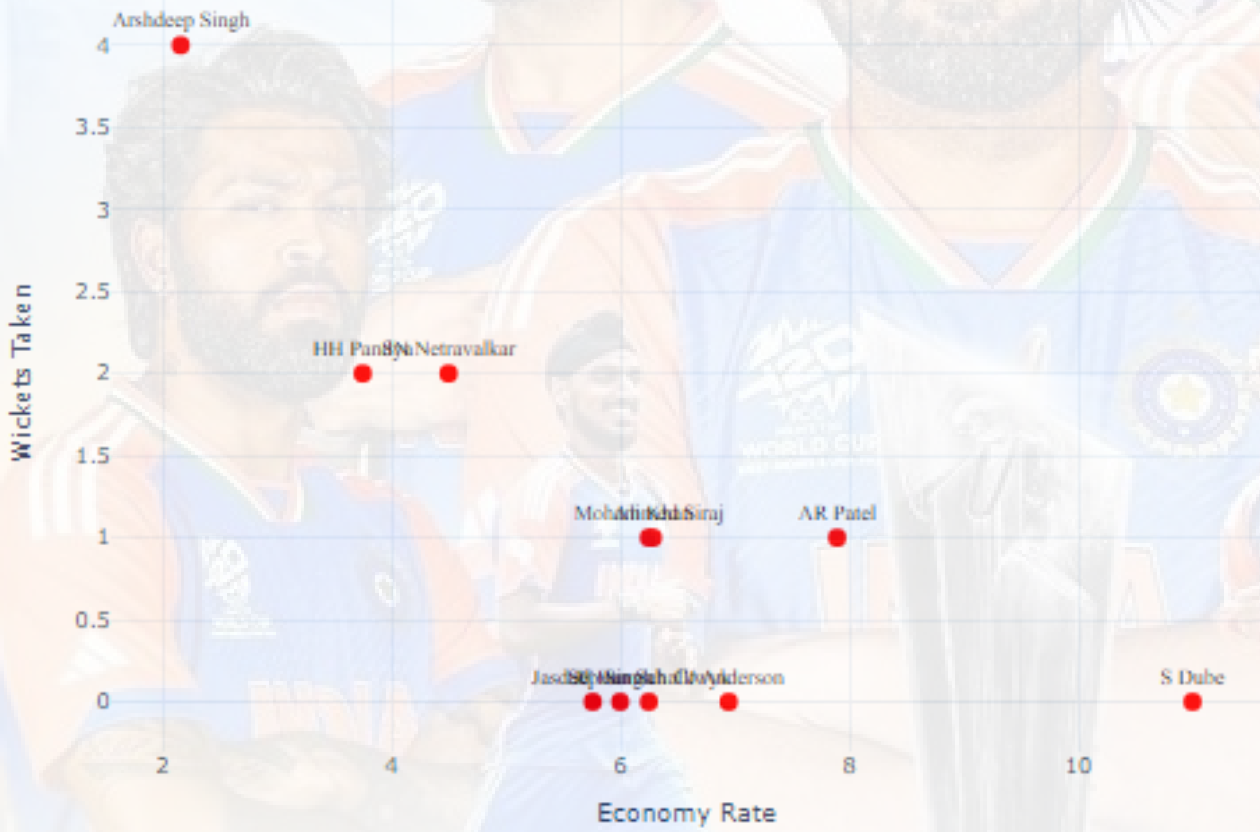
```
import plotly.express as px

fig = px.bar(
    batter_stats,
    x=batter_stats.index,
    y='Runs',
    title='Run Distribution by Batters',
    labels={'x': 'Batter', 'Runs': 'Runs Scored'},
    template='plotly_white'
)

fig.update_layout(
    xaxis_title='Batter',
    yaxis_title='Runs Scored',
    xaxis=dict(tickangle=90)
)

fig.show()
```



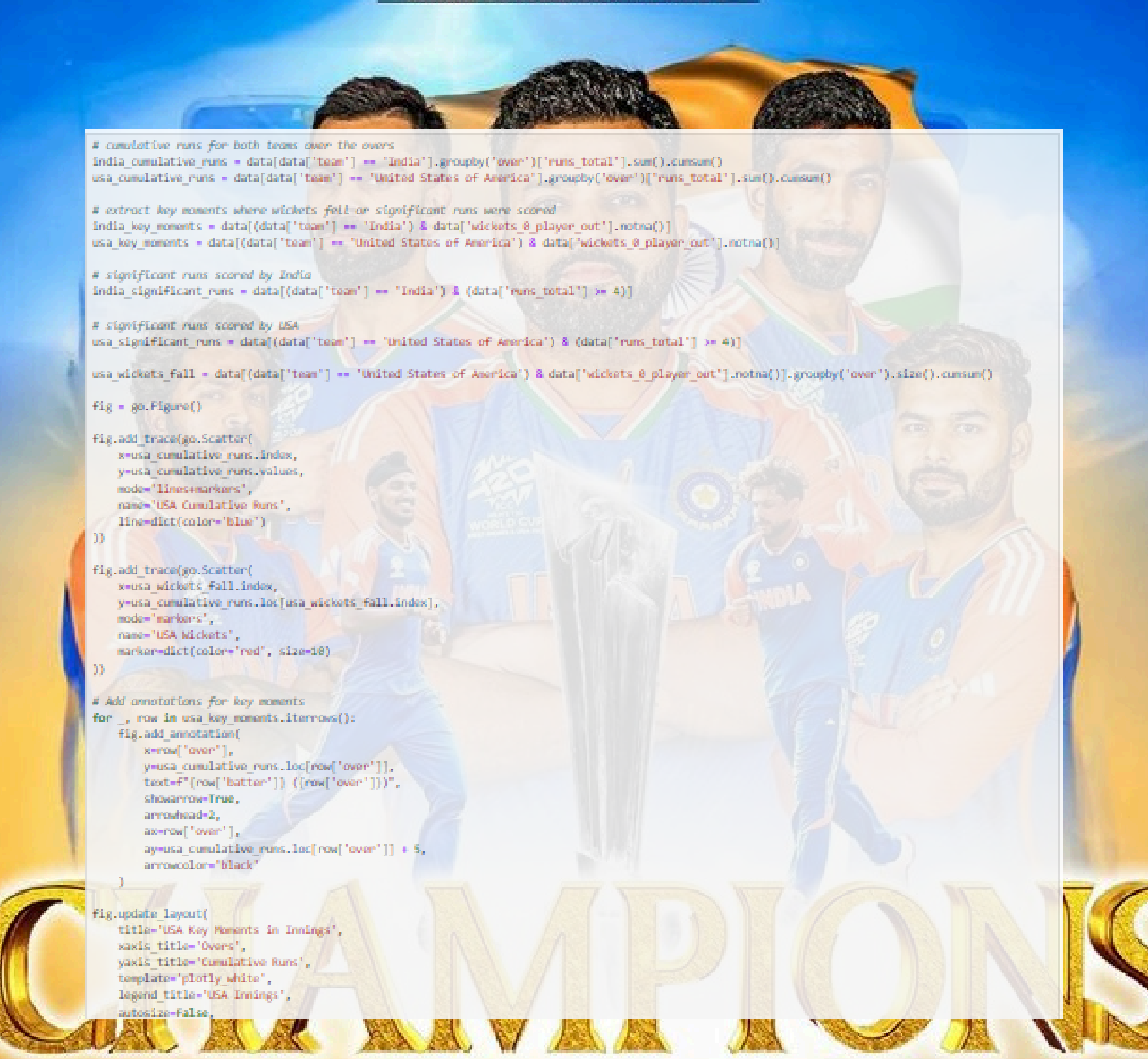
Now, let's have a look at the bowling performance:

```
fig = go.Figure()

fig.add_trace(go.Scatter(
    x=bowler_stats['Economy Rate'],
    y=bowler_stats['Wickets'],
    mode='markers+text',
    text=bowler_stats.index,
    textposition='top center',
    textfont=dict(
        family="sans serif",
        size=12,
        color="black"
    ),
    marker=dict(color='red', size=10),
    name='Bowlers'
))

fig.update_layout(
    title='Bowling Performance',
    xaxis_title='Economy Rate',
    yaxis_title='Wickets Taken',
    template='plotly_white',
    autosize=False,
    width=800,
    height=600
)

fig.show()
```

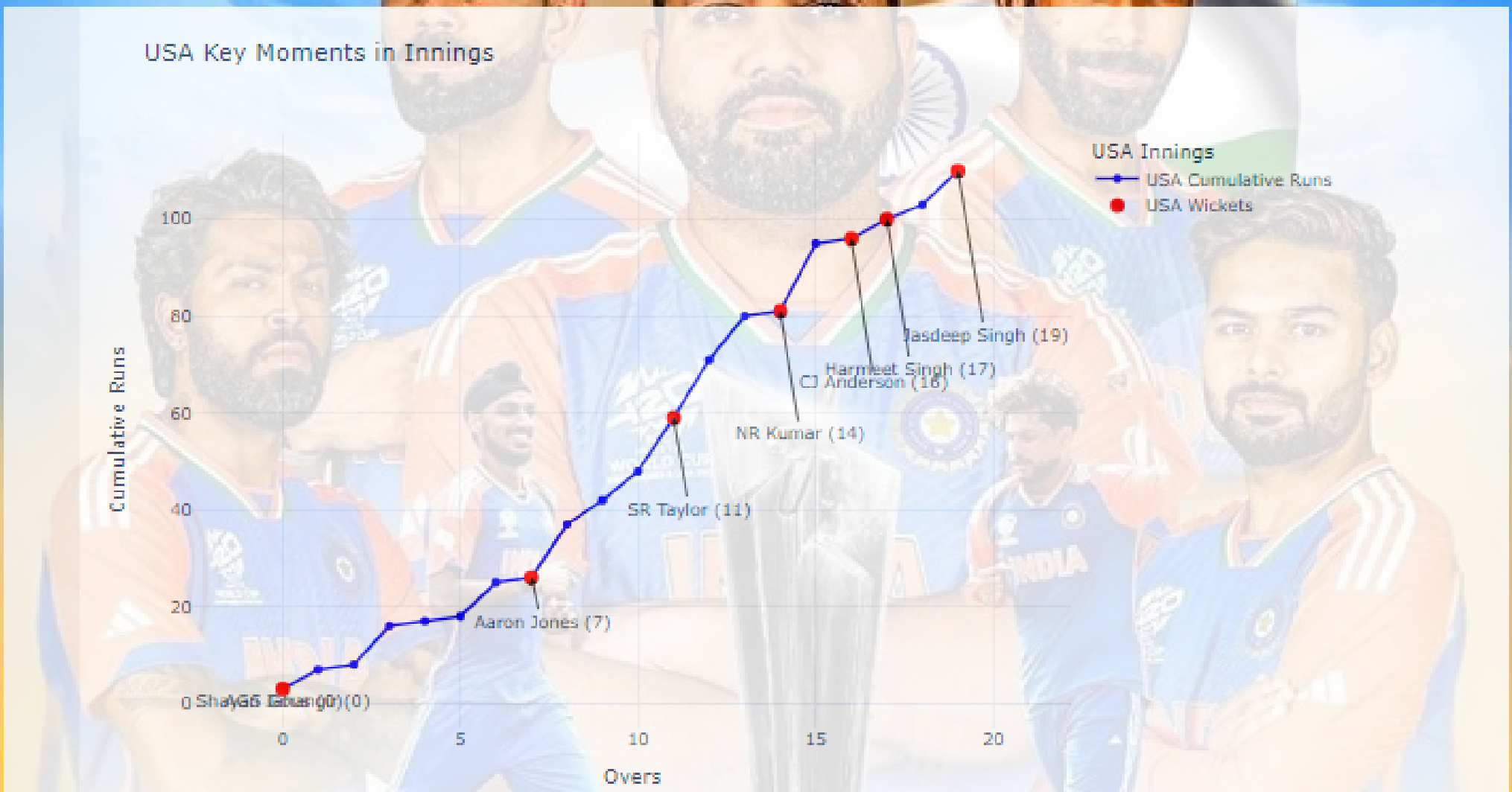
Bowling Performance



The bowling performance graph compares the economy rate and wickets taken by various bowlers in the match between India and the USA. Arshdeep Singh stands out as the most effective bowler, taking the highest number of wickets (4) with a commendable economy rate. Other notable performances include HH Pandya and SN Netravalkar, both taking 2 wickets each with moderate economy rates. Bowlers like S Dube, having a higher economy rate, contributed less in terms of wickets.



```python
# cumulative runs for both teams over the overs
india_cumulative_runs = data[data['team'] == 'India'].groupby('over')['runs_total'].sum().cumsum()
usa_cumulative_runs = data[data['team'] == 'United States of America'].groupby('over')['runs_total'].sum().cumsum()

# extract key moments where wickets fell or significant runs were scored
india_key_moments = data[(data['team'] == 'India') & data['wickets_@_player_out'].notna()]
usa_key_moments = data[(data['team'] == 'United States of America') & data['wickets_@_player_out'].notna()]

# significant runs scored by India
india_significant_runs = data[(data['team'] == 'India') & (data['runs_total'] >= 4)]

# significant runs scored by USA
usa_significant_runs = data[(data['team'] == 'United States of America') & (data['runs_total'] >= 4)]

usa_wickets_fall = data[(data['team'] == 'United States of America') & data['wickets_@_player_out'].notna()].groupby('over').size().cumsum()

fig = go.Figure()

fig.add_trace(go.Scatter(
    x=usa_cumulative_runs.index,
    y=usa_cumulative_runs.values,
    mode='lines+markers',
    name='USA Cumulative Runs',
    line=dict(color='blue')
))

fig.add_trace(go.Scatter(
    x=usa_wickets_fall.index,
    y=usa_cumulative_runs.loc[usa_wickets_fall.index],
    mode='markers',
    name='USA Wickets',
    marker=dict(color='red', size=10)
))

# Add annotations for key moments
for _, row in usa_key_moments.iterrows():
    fig.add_annotation(
        x=row['over'],
        y=usa_cumulative_runs.loc[row['over']],
        text=f"{row['batter']} ({row['over']})",
        showarrow=True,
        arrowhead=2,
        ax=row['over'],
        ay=usa_cumulative_runs.loc[row['over']] + 5,
        arrowcolor='black'
    )

fig.update_layout(
    title='USA Key Moments in Innings',
    xaxis_title='Overs',
    yaxis_title='Cumulative Runs',
    template='plotly_white',
    legend_title='USA Innings',
    autosize=False,
```



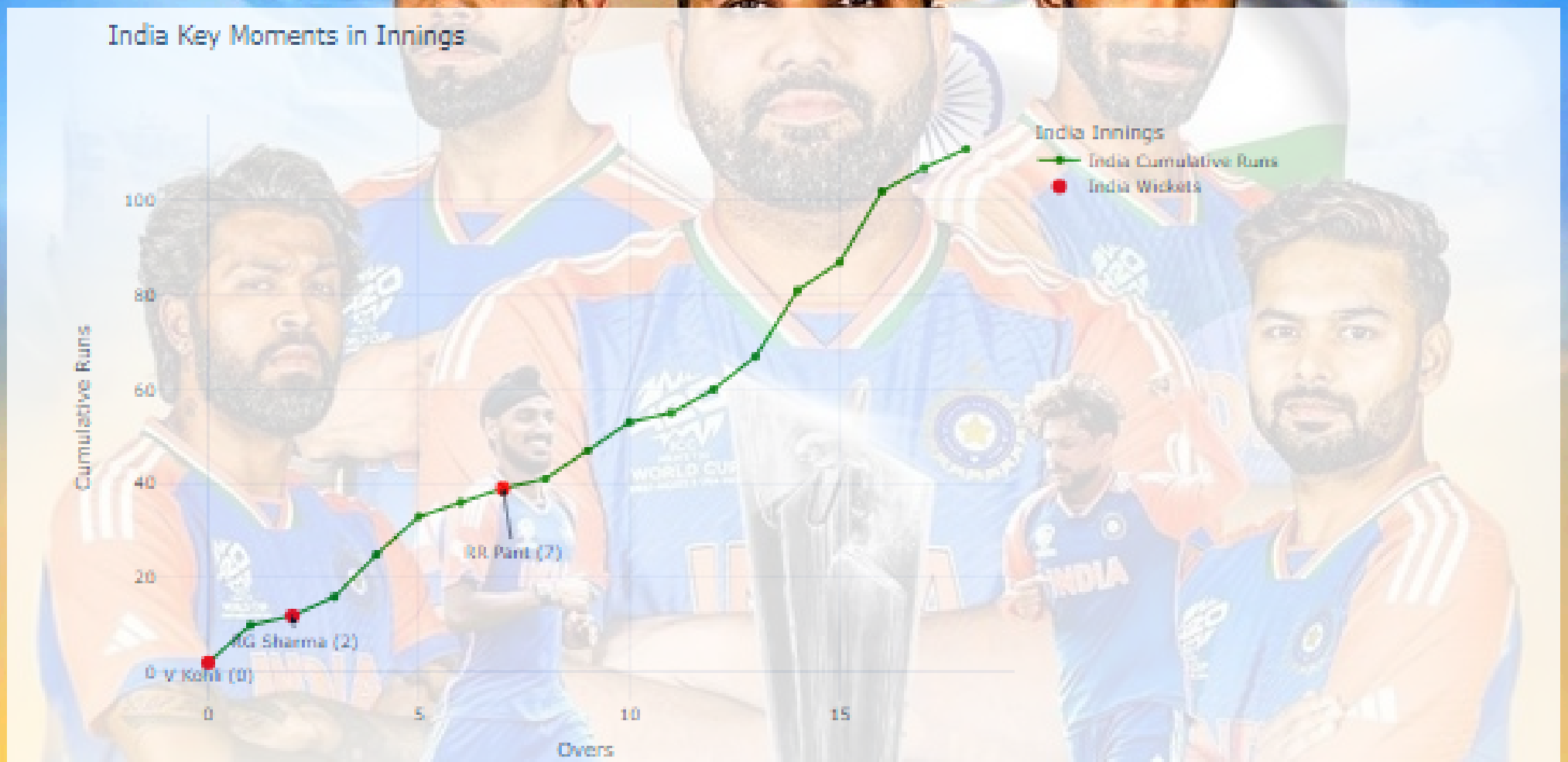
[ ]:

The graph highlights the key moments in the USA's innings, showing the progression of the cumulative run with wickets marked.
Early wickets, such as those of Shayan Jahangir and AGS Gous in the first over, set back the USA's momentum.
Despite recoveries led by partnerships involving SR Taylor and NR Kumar, regular wickets in the middle and late overs,
particularly around the 14th to 19th overs, hindered their progress.

The dismissals of key players like Aaron Jones, SR Taylor, and later batsmen such as Harmeet Singh and CJ Anderson,
prevented the USA from building a substantial and uninterrupted run flow, ultimately impacting their total score.

Now, let's have a look at the key moments for India:

```
india_cumulative_runs = data[data['team'] == 'India'].groupby('over')['runs_total'].sum().cumsum()
india_wickets_fall = data[(data['team'] == 'India') & data['wickets_@_player_out'].notna()].groupby('over').size().cumsum()
india_key_moments = data[(data['team'] == 'India') & data['wickets_@_player_out'].notna()].reset_index()

fig = go.Figure()

fig.add_trace(go.Scatter(
    x=india_cumulative_runs.index,
    y=india_cumulative_runs.values,
    mode='lines+markers',
    name='India Cumulative Runs',
    line=dict(color='green')
))

fig.add_trace(go.Scatter(
    x=india_wickets_fall.index,
    y=india_cumulative_runs.loc[india_wickets_fall.index],
    mode='markers',
    name='India Wickets',
    marker=dict(color='red', size=10)
))

for _, row in india_key_moments.iterrows():
    fig.add_annotation(
        x=row['over'],
        y=india_cumulative_runs.loc[row['over']],
        text=f"{row['batter']} ({row['over']})",
        showarrow=True,
        arrowhead=2,
        ax=row['over'],
        ay=india_cumulative_runs.loc[row['over']] + 5,
        arrowcolor='black'
    )

fig.update_layout(
    title='India Key Moments in Innings',
    xaxis_title='Overs',
    yaxis_title='Cumulative Runs',
    template='plotly_white',
    legend_title='India Innings',
    autosize=False,
    width=800,
    height=600
```



Despite an early setback with the dismissals of V. Kohli and RG Sharma in the first two overs, India managed to maintain a steady run rate. The wicket of RR Pant in the 7th over was another crucial moment, but subsequent partnerships helped stabilize the innings.

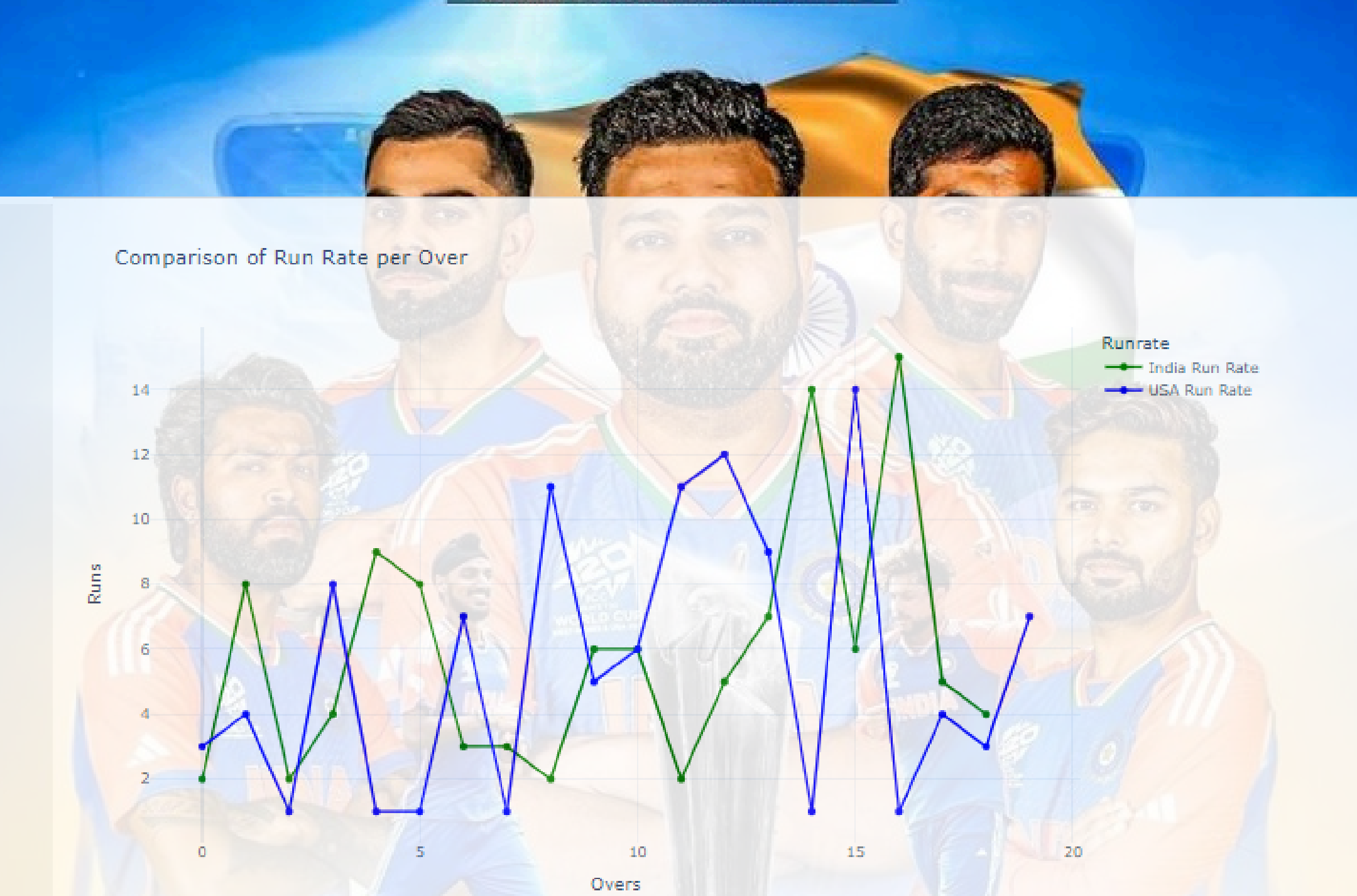
CHAMPIONS

Now, let's compare the average run rate for both teams:

```
india_run_rate = data[data['team'] == 'India'].groupby('over')['runs_total'].sum().mean()
usa_run_rate = data[data['team'] == 'United States of America'].groupby('over')['runs_total'].sum().mean()

fig = go.Figure()

fig.add_trace(go.Bar(
    x=['India', 'USA'],
    y=[india_run_rate, usa_run_rate],
    marker_color=['green', 'blue']
))

fig.add_annotation(
    x='India',
    y=india_run_rate,
    text=f"{india_run_rate:.2f}",
    showarrow=False,
    yshift=10
)

fig.add_annotation(
    x='USA',
    y=usa_run_rate,
    text=f"{usa_run_rate:.2f}",
    showarrow=False,
    yshift=10
)

fig.update_layout(
    title='Comparison of Average Run Rate per Over',
    xaxis_title='Team',
    yaxis_title='Average Run Rate per Over',
    template='plotly_white'
)

fig.show()
```

Now, let's finish with a comparison of the run rate per over:

```
india_run_rate_per_over = data[data['team'] == 'India'].groupby('over')['runs_total'].sum()
usa_run_rate_per_over = data[data['team'] == 'United States of America'].groupby('over')['runs_total'].sum()

fig = go.Figure()

fig.add_trace(go.Scatter(
    x=india_run_rate_per_over.index,
    y=india_run_rate_per_over.values,
    mode='lines+markers',
    name='India Run Rate',
    line=dict(color='green')
))

fig.add_trace(go.Scatter(
    x=usa_run_rate_per_over.index,
    y=usa_run_rate_per_over.values,
    mode='lines+markers',
    name='USA Run Rate',
    line=dict(color='blue')
))

fig.update_layout(
    title='Comparison of Run Rate per Over',
    xaxis_title='Overs',
    yaxis_title='Runs',
    template='plotly_white',
    legend_title='Runrate',
    autosize=False,
    width=1000,
    height=600
)

fig.show()
```



The USA experienced significant fluctuations in their run rate, with peaks in the 10th and 15th overs, but also several low-scoring overs, indicating inconsistency. India's run rate was relatively more stable, with a notable increase towards the end of their innings. This stability in India's run rate, especially in the death overs, allowed them to maintain pressure and chase the target successfully. The graph highlights India's ability to keep a more consistent scoring pace, while the USA's variable run rate reflects periods of struggle to maintain momentum

UJJAWAL VALIYA