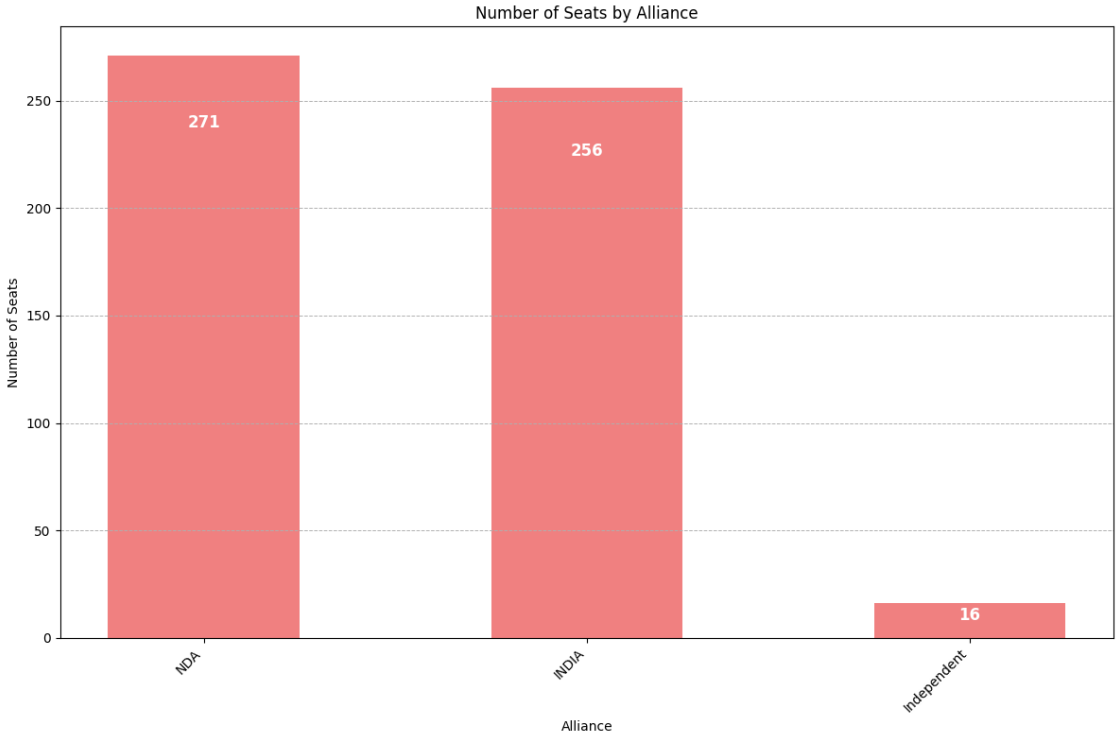
Report

July 1, 2024

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
import matplotlib.patches as patches
import squarify
file_path = 'merged_election_results.xlsx'
election_data = pd.read_excel(file_path)
```

[3]:
```
# Group the data by alliance and count the number of seats won by each alliance
alliance_seat_count = election_data['Alliance'].value_counts()

plt.figure(figsize=(12, 8))
bars = plt.bar(alliance_seat_count.index, alliance_seat_count.values,␣
 ↪color='lightcoral', width=0.5)
plt.xlabel('Alliance')
plt.ylabel('Number of Seats')
plt.title('Number of Seats by Alliance')
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.grid(axis='y', linestyle='--', linewidth=0.7)

for bar in bars:
    yval = bar.get_height()
    plt.text(
        bar.get_x() + bar.get_width() / 2.0,
        yval - (0.1 * yval),
        f'{yval}',
        ha='center',
        va='top',
        color='white',
        fontsize=12,
        fontweight='bold'
    )

plt.tight_layout()
plt.show()
```



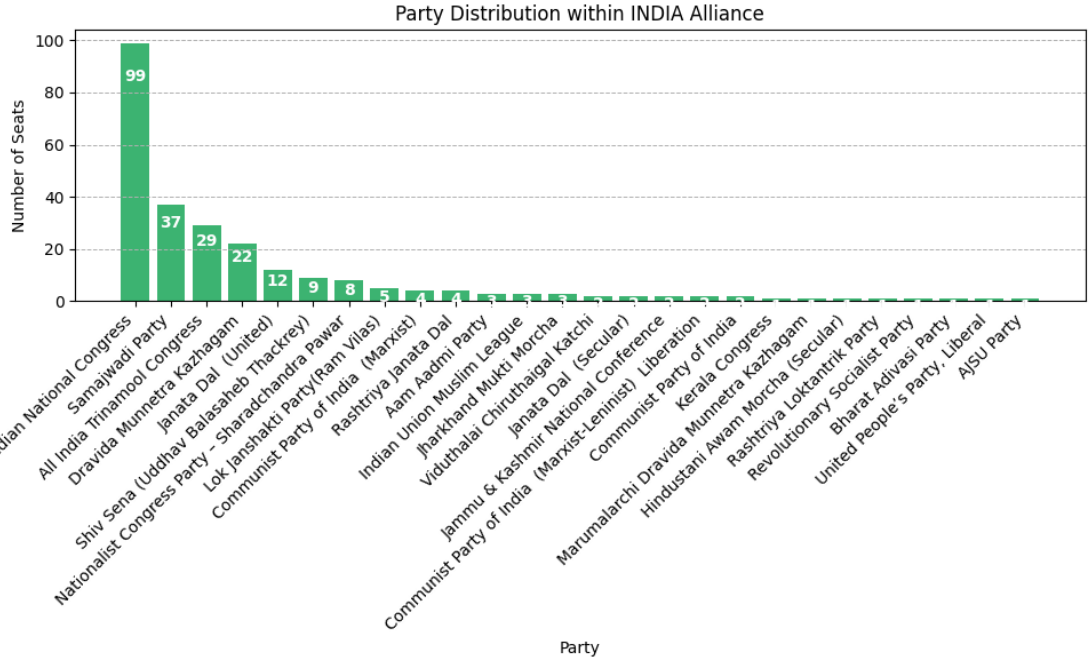
0.0.1 Inference

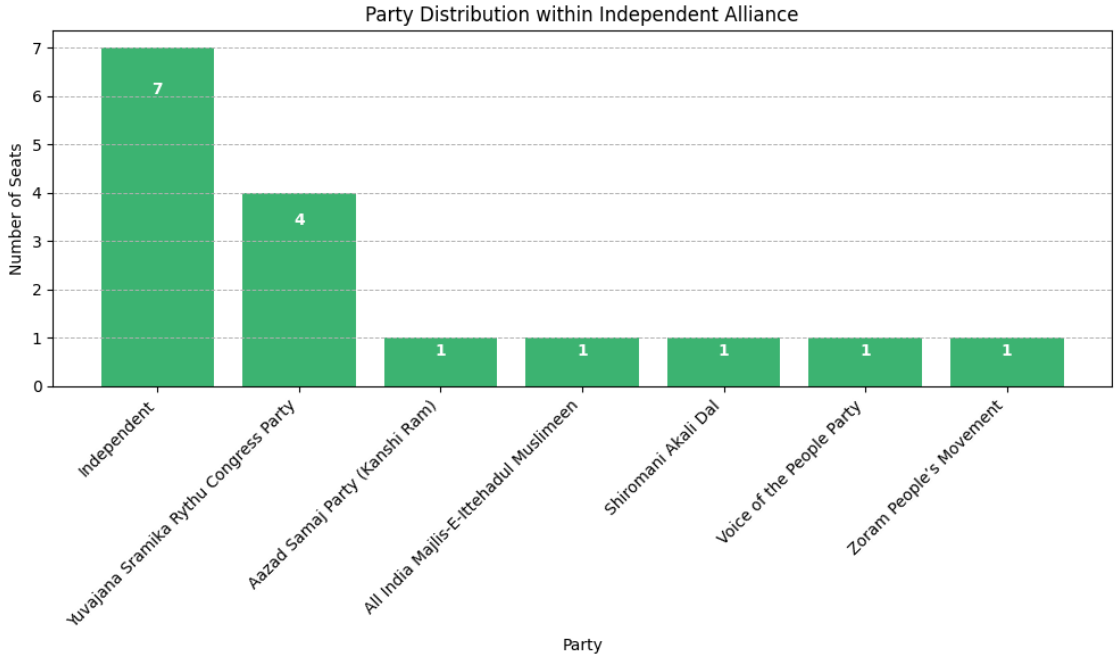
The bar chart vividly illustrates the distribution of seats won by each alliance in the election.

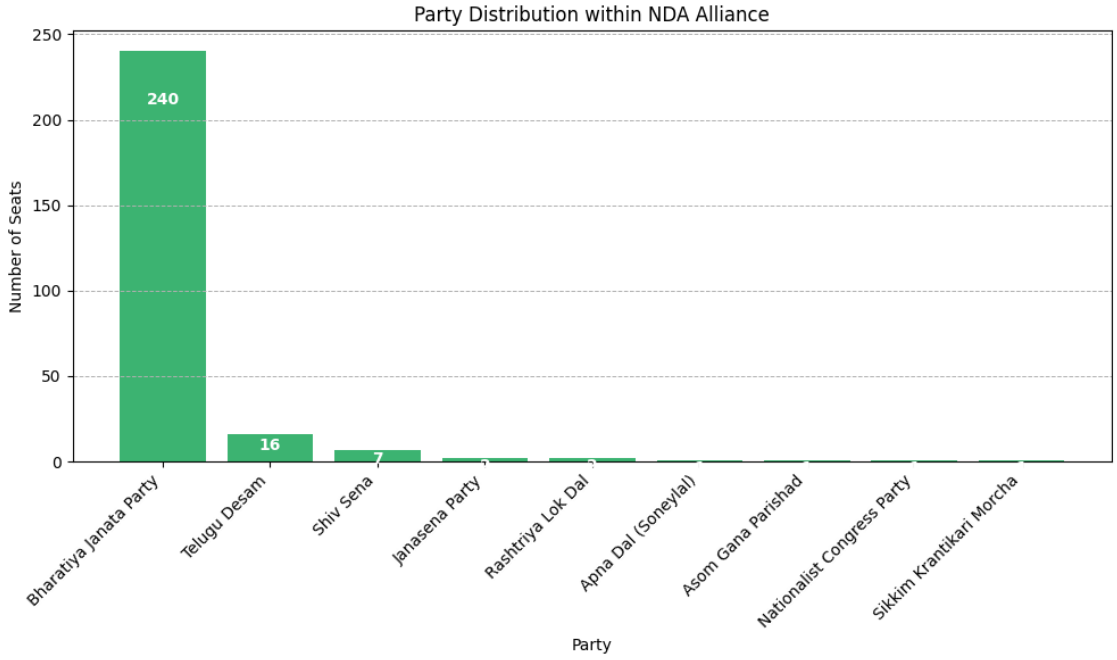
Some alliances have clearly secured a substantial number of seats, towering over the rest. This dominance suggests a significant political influence or strong voter base for these groups.

[8]:
```
# Group the data by Alliance and then by Party to count the number of seats won␣
 ↪by each party within each alliance
alliance_party_count = election_data.groupby(['Alliance', 'Party']).size().
 ↪unstack(fill_value=0)

for alliance in alliance_party_count.index:
    plt.figure(figsize=(10, 6))
    data = alliance_party_count.loc[alliance]
    data = data[data > 0].sort_values(ascending=False)
    bars = plt.bar(data.index, data.values, color='mediumseagreen')
    plt.xlabel('Party')
    plt.ylabel('Number of Seats')
    plt.title(f'Party Distribution within {alliance} Alliance')
    plt.xticks(rotation=45, ha='right', fontsize=10)
    plt.grid(axis='y', linestyle='--', linewidth=0.7)

    # Annotating the bars
```

```
    for bar in bars:
        yval = bar.get_height()
        plt.text(
            bar.get_x() + bar.get_width() / 2.0,
            yval - (0.1 * yval),
            f'{yval}',
            ha='center',
            va='top',
            color='white',
            fontsize=10,
            fontweight='bold'
        )
    plt.tight_layout()
    plt.show()
```





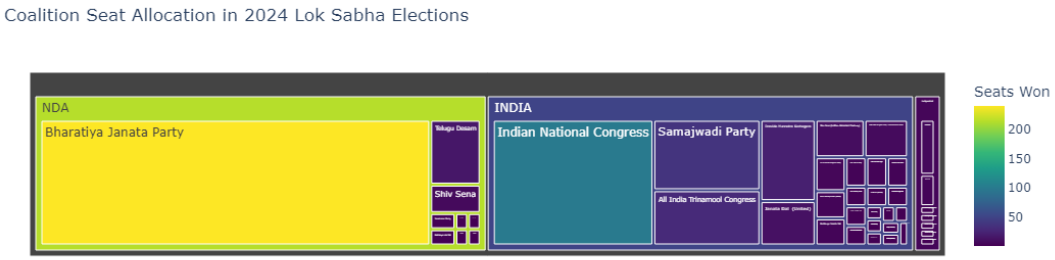


0.0.2 Inference

The visualizations reveal that within each alliance, the distribution of seats among member parties varies significantly. **Some alliances are dominated by a single party, while others show**

a more balanced contribution from multiple parties. This highlights the diverse power structures and internal dynamics within political alliances.

[14]:
```
# Aggregate data by coalition and party to count the number of seats won
coalition_data = election_data.groupby(['Alliance', 'Party']).size().
 ↪reset_index(name='Seats Won')

# Step 3: Create the Tree Map
fig = px.treemap(
    coalition_data,
    path=['Alliance', 'Party'],
    values='Seats Won',
    color='Seats Won',
    color_continuous_scale='Viridis',
    title='Coalition Seat Allocation in 2024 Lok Sabha Elections'
)
fig.show()
```

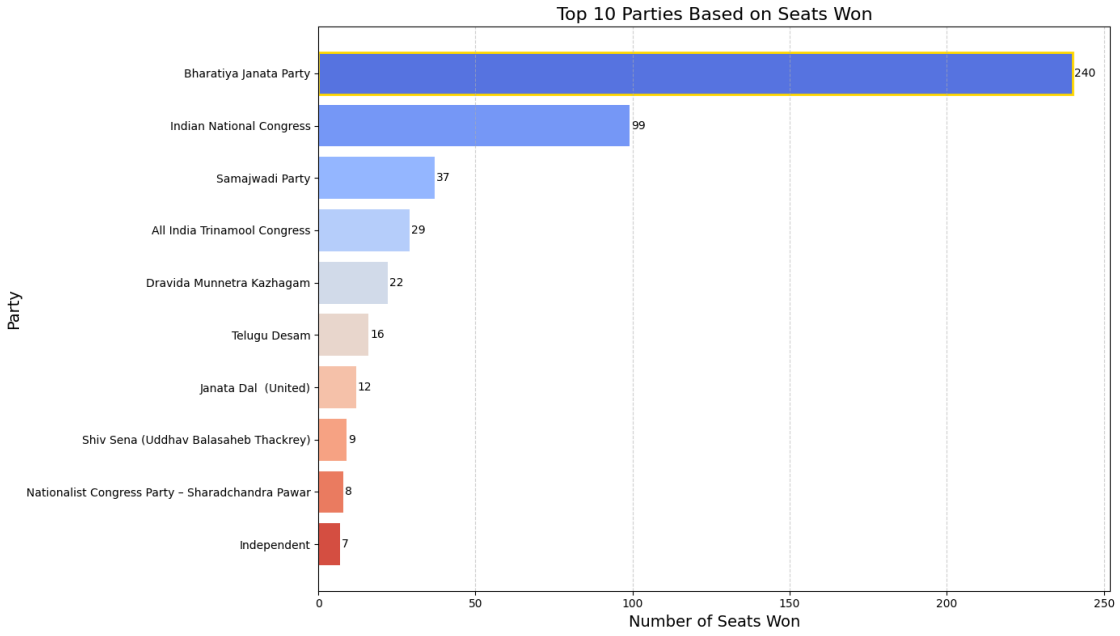


0.0.3 Inference

The tree map vividly illustrates the distribution of seats among parties within each alliance. **Larger blocks indicate dominant parties, while smaller ones highlight the minor players.** This visualization effectively captures the hierarchical structure and relative strengths of parties within their alliances.

[19]:
```
file_path = "merged_election_results.xlsx"
excel_data = pd.ExcelFile(file_path)
sheet_names = excel_data.sheet_names

df = pd.read_excel(file_path, sheet_name='Sheet1')

seats_won = df['Party'].value_counts()

top_10_parties = seats_won.head(10)
```

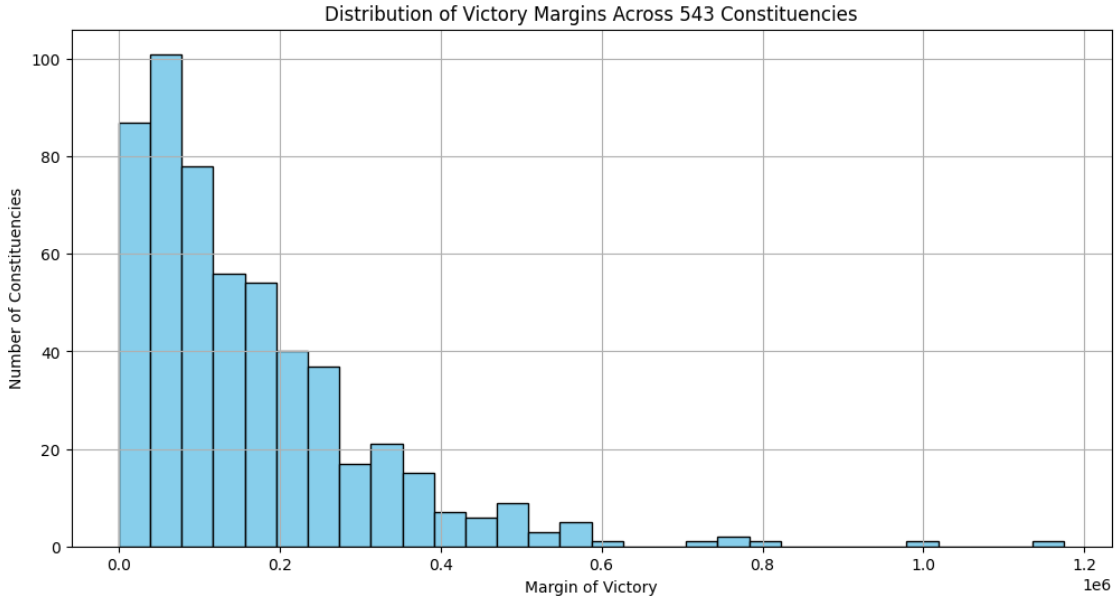
```
plt.figure(figsize=(14, 8))
colors = sns.color_palette("coolwarm", len(top_10_parties))

bars = plt.barh(top_10_parties.index, top_10_parties.values, color=colors)


for bar in bars:
    width = bar.get_width()
    plt.text(width + 0.5, bar.get_y() + bar.get_height() / 2, f'{width}',␣
 ↪va='center', ha='left', fontsize=10, color='black')

max_value = top_10_parties.max()
max_index = top_10_parties.idxmax()
highlight = patches.Rectangle((0, bars[0].get_y()), max_value, bars[0].
 ↪get_height(), fill=False, edgecolor='gold', linewidth=2)
plt.gca().add_patch(highlight)

plt.title('Top 10 Parties Based on Seats Won', fontsize=16)
plt.xlabel('Number of Seats Won', fontsize=14)
plt.ylabel('Party', fontsize=14)
plt.grid(axis='x', linestyle='--', alpha=0.6)
plt.gca().invert_yaxis()

plt.tight_layout()
plt.show()
```



0.0.4 Inference

The horizontal bar chart highlights the top 10 parties by the number of seats won. **Notably[Bharatiya Janata Partye] stands out with the highest seats, emphasized by the golden outline, showcasing its significant lead over others.** This visualization effectively captures the major players in the election, demonstrating their relative political strength.

[27]:
```
plt.figure(figsize=(12, 6))
plt.hist(election_data['Margin'], bins=30, color='skyblue', edgecolor='black')

plt.title('Distribution of Victory Margins Across 543 Constituencies')
plt.xlabel('Margin of Victory')
plt.ylabel('Number of Constituencies')

plt.grid(True)

plt.show()
```

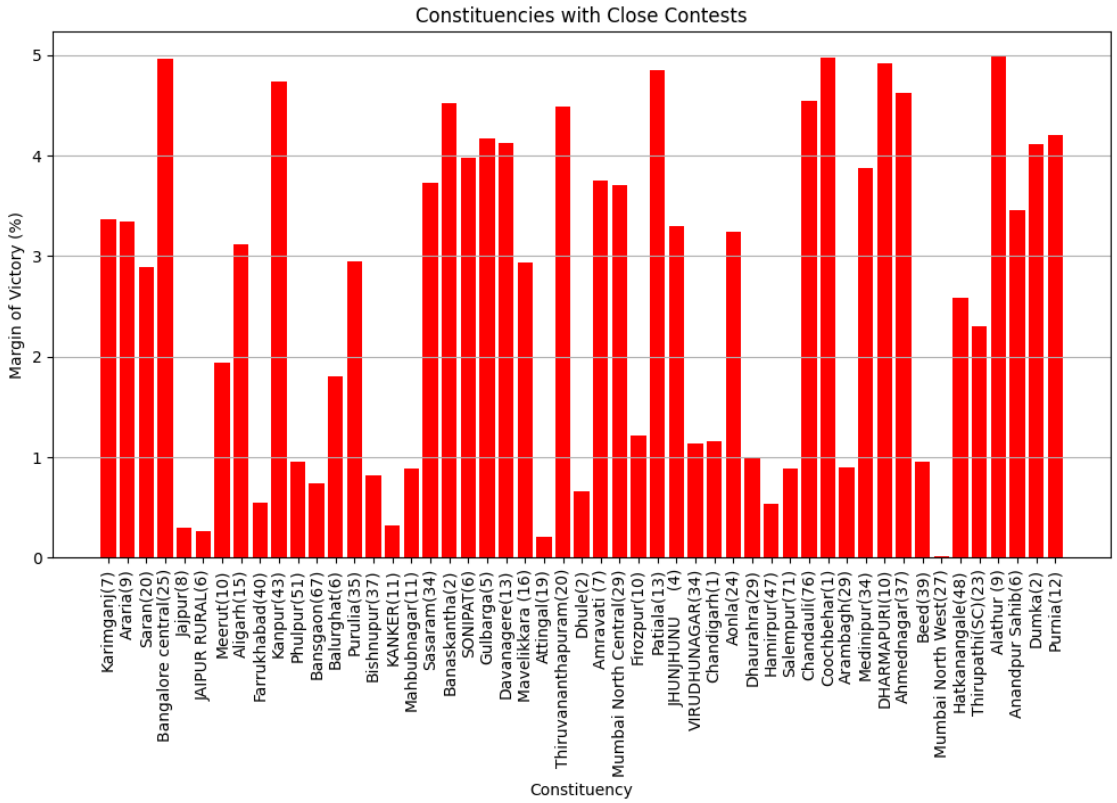


0.0.5 Inference

The histogram illustrates the distribution of victory margins across the 543 constituencies. **Most constituencies have relatively small margins of victory, indicating closely contested races.** This suggests a highly competitive election with numerous tight contests.

[30]:
```
# Define close contests as those with margins less than 5% of total votes
election_data['Close Contest'] = election_data['Margin'] / election_data['Total␣
↪Votes'] * 100
close_contests = election_data[election_data['Close Contest'] < 5]
```

```
plt.figure(figsize=(12, 6))
plt.bar(close_contests['Parliament Constituency'], close_contests['Close␣
 ↪Contest'], color='red')
plt.title('Constituencies with Close Contests')
plt.xlabel('Constituency')
plt.ylabel('Margin of Victory (%)')
plt.xticks(rotation=90)
plt.grid(True, axis='y')
plt.show()
```

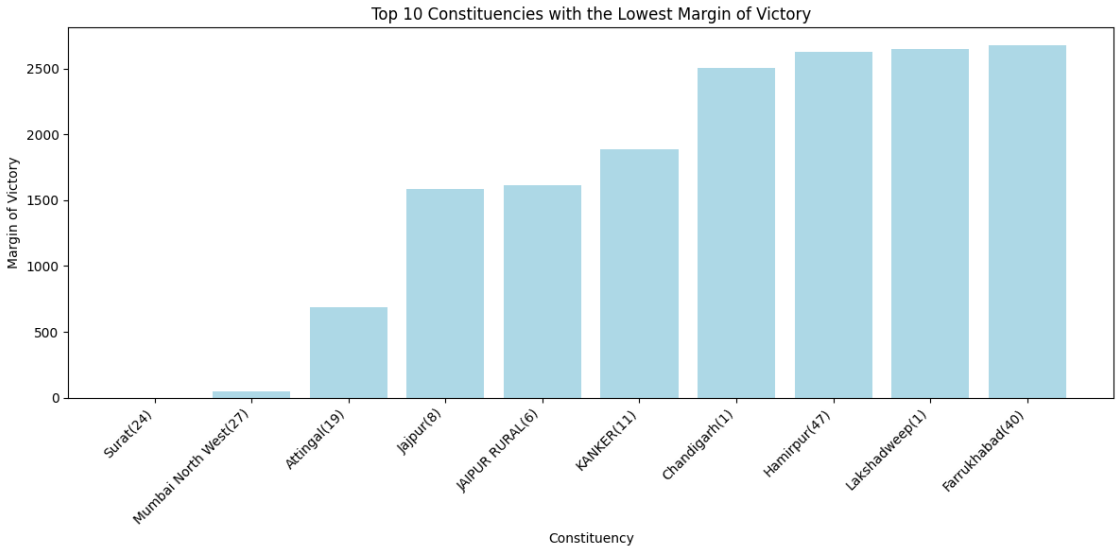


0.0.6 Inference

The bar chart identifies constituencies with close contests, defined as margins of victory less than 5% of total votes. **These constituencies, marked in red, represent crucial battlegrounds where electoral outcomes were highly competitive and could have swung in favor of either candidate.** This visualization highlights key areas of electoral tension and strategic importance in the overall election landscape.

[33]:
```
# Sort the data by 'Margin' in ascending order and select the top 10 rows
top_10_lowest_margin = election_data.sort_values('Margin').head(10)
```

```
constituencies = top_10_lowest_margin['Parliament Constituency']
margins = top_10_lowest_margin['Margin']

plt.figure(figsize=(12, 6))
plt.bar(constituencies, margins, color='lightblue')
plt.xlabel('Constituency')
plt.ylabel('Margin of Victory')
plt.title('Top 10 Constituencies with the Lowest Margin of Victory')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

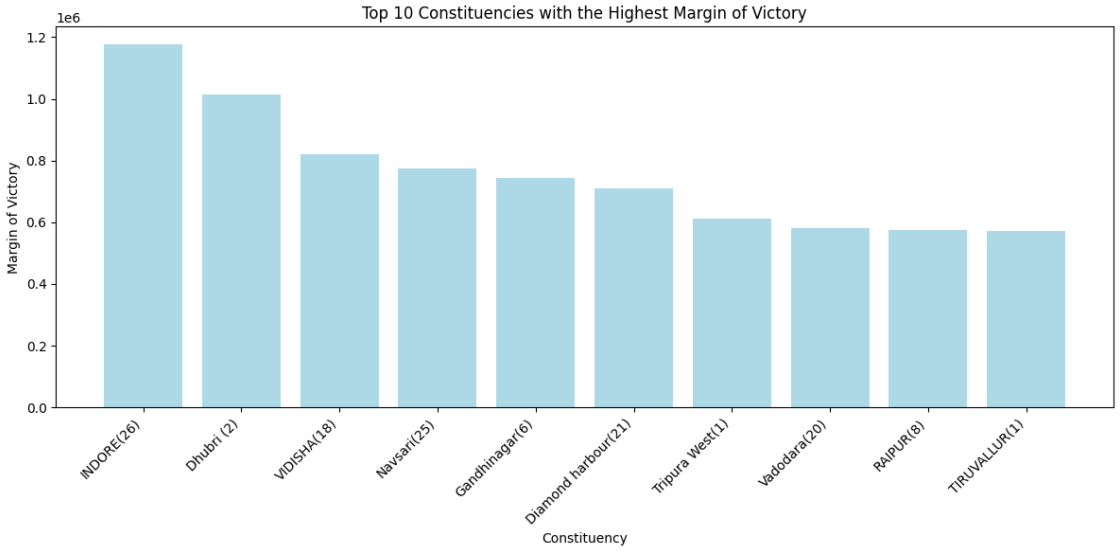


0.0.7 Inference

The bar chart displays the top 10 constituencies with the lowest margins of victory, sorted in ascending order. **These constituencies experienced extremely tight races, with the smallest margins between winning and losing candidates.** This visualization underscores the competitive nature of these electoral battles, where every vote played a crucial role in determining the outcome.

[36]:
```
# Sort the data by 'Margin' in descending order and select the top 10 rows
top_10_highest_margin = election_data.sort_values('Margin', ascending=False).
↪head(10)

constituencies = top_10_highest_margin['Parliament Constituency']
margins = top_10_highest_margin['Margin']

plt.figure(figsize=(12, 6))
```

```
plt.bar(constituencies, margins, color='lightblue')
plt.xlabel('Constituency')
plt.ylabel('Margin of Victory')
plt.title('Top 10 Constituencies with the Highest Margin of Victory')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

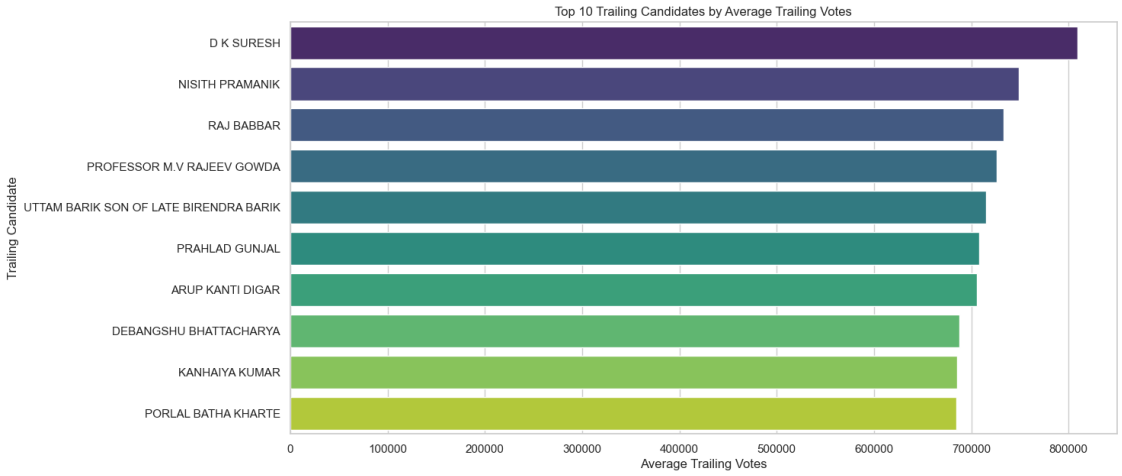


0.0.8 Inference

The bar chart highlights the top 10 constituencies with the highest margins of victory, sorted in descending order. **These constituencies witnessed decisive victories with substantial margins between the winning and runner-up candidates.** This visualization indicates areas where candidates secured strong mandates from voters, reflecting significant support and potentially influencing future electoral strategies.

[39]:
```
election_data['Trailing Candidate Votes'] = election_data['Total Votes'] -␣
↪election_data['Margin']

# Define a threshold for close contests
threshold = 10000
close_contests = election_data[election_data['Margin'] < threshold]

trailing_candidate_analysis = election_data.groupby('Trailing Candidate').agg({
    'Trailing Candidate Votes': ['mean', 'max', 'count'],
    'Margin': 'mean'
}).reset_index()
```

```
trailing_candidate_analysis.columns = ['Trailing Candidate', 'Avg Trailing␣
 ↪Votes', 'Max Trailing Votes', 'Number of Constituencies', 'Avg Margin']

trailing_candidate_analysis = trailing_candidate_analysis.sort_values(by='Avg␣
 ↪Trailing Votes', ascending=False)

sns.set(style="whitegrid")

plt.figure(figsize=(14, 7))
sns.barplot(x='Avg Trailing Votes', y='Trailing Candidate',␣
 ↪data=trailing_candidate_analysis.head(10), palette="viridis")
plt.title('Top 10 Trailing Candidates by Average Trailing Votes')
plt.xlabel('Average Trailing Votes')
plt.ylabel('Trailing Candidate')
plt.show()
```

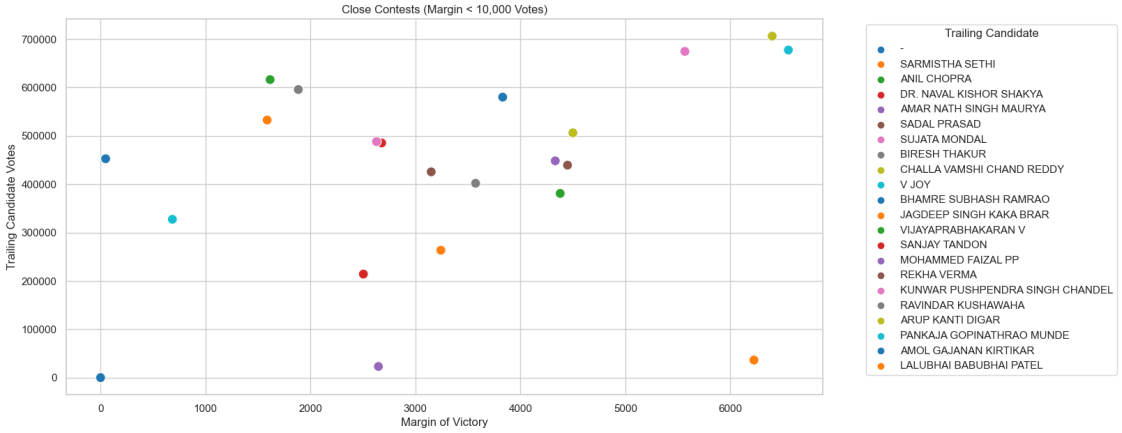


0.0.9 Inference

The bar chart ranks the top 10 trailing candidates by their average trailing votes across constituencies. **Candidates with higher average trailing votes indicate strong performances despite not winning, potentially influencing future electoral strategies and coalition dynamics.** This visualization provides insights into the resilience and support base of candidates who narrowly missed victory in closely contested elections.

[44]:
```
# Scatter plot for close contests
plt.figure(figsize=(14, 7))
sns.scatterplot(x='Margin', y='Trailing Candidate Votes', hue='Trailing␣
 ↪Candidate', data=close_contests, palette="tab10", s=100)
plt.title('Close Contests (Margin < 10,000 Votes)')
plt.xlabel('Margin of Victory')
```

```
plt.ylabel('Trailing Candidate Votes')
plt.legend(title='Trailing Candidate', bbox_to_anchor=(1.05, 1), loc='upper␣
 ↪left')
plt.show()
```



0.0.10 Inference

The scatter plot visualizes close contests where the margin of victory was less than 10,000 votes. **It illustrates the relationship between the margin of victory and the votes received by the trailing candidate, highlighting instances where candidates narrowly missed winning.** This analysis is crucial for understanding the competitive nature of these elections and assessing the impact of margin thresholds on electoral outcomes.