Winning Space Race with Data Science

Name:- Janmejaya Pharande
Date:- 15/04/2023

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

Summary of methodologies

1. Data Collection through API
2. Data Collection with Web Scraping
3. Data Wrangling
4. Exploratory Data Analysis with SQL
5. Exploratory Data Analysis with Data Visualization
6. Interactive Visual Analytics with Folium
7. Machine Learning Prediction

- Summary of all results

I. Exploratory Data Analysis result

II. Interactive analytics in screenshots

III. Predictive Analytics result

Introduction

The goal of this project is to develop a machine learning pipeline that can predict the probability of a successful landing of the first stage of a Falcon 9 rocket. The successful landing of the first stage is crucial for reducing the cost of rocket launches, as Space X can reuse the first stage. Therefore, understanding the factors that contribute to a successful landing is critical for predicting launch costs accurately and developing a competitive bidding strategy for rocket launch services.
To achieve this goal, we need to identify the critical features that affect the landing of the first stage. These features may include environmental conditions, such as wind speed and direction, sea state, and visibility, as well as technical factors, such as the performance of the engines, the guidance system, and the control systems.

Problems you want to find answers

i. What factors determine the successful landing of the Rocket?

ii. Developing an accurate machine learning model for predicting the success rate of a first-stage landing.

iii. What operating conditions needs to be in followed to have a successful landing flight.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

1. Data collection methods: Data was collected using get requests to the SpaceX API and web scraping from the Internet.
2. Converting data to Pandas DataFrame: The response content was decoded as a JSON and then converted into a Pandas DataFrame using .json_normalize() function call.
3. Data cleaning: The data was cleaned and checked for missing values. Missing values were filled in where necessary.
4. Web scraping: We used BeautifulSoup to extract the launch records as an HTML table from Wikipedia, parsed the table and converted it to a Pandas DataFrame for analysis.
5. Objective: The objective of data collection and cleaning was to develop a robust dataset that can be used for analysis and developing a machine learning model for predicting the success rate of a first-stage landing.

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.
- The link to the notebook is https://github.com/Janmejaya03/janmejayapharandecapstonestory/blob/main/janmejaya-spacex-data-collection-api.ipynb

1. Get request for rocket launch data using API

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

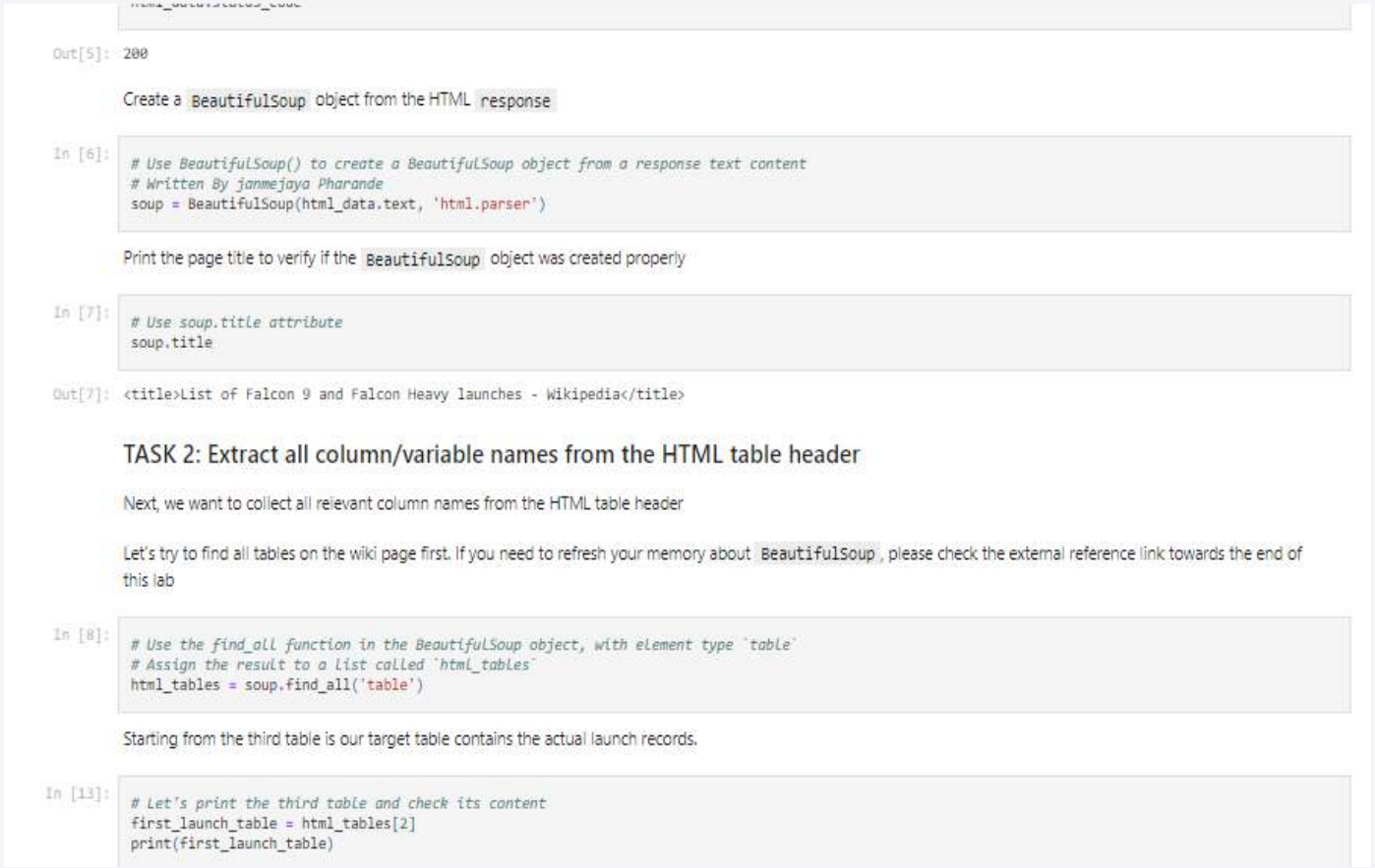
In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- We applied web scrapping on webscraping Falcon 9 launch records using BeautifulSoup
- We prepared the table and converted it into a pandas dataframe.
- The link to the notebook is https://github.com/Janmejaya03/janmejayapharandecapstonestory/blob/main/janmejaya-webscraping.ipynb



Out[5]: 200

Create a `BeautifulSoup` object from the HTML `response`

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
# Written By janmejaya Pharande
soup = BeautifulSoup(html_data.text, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

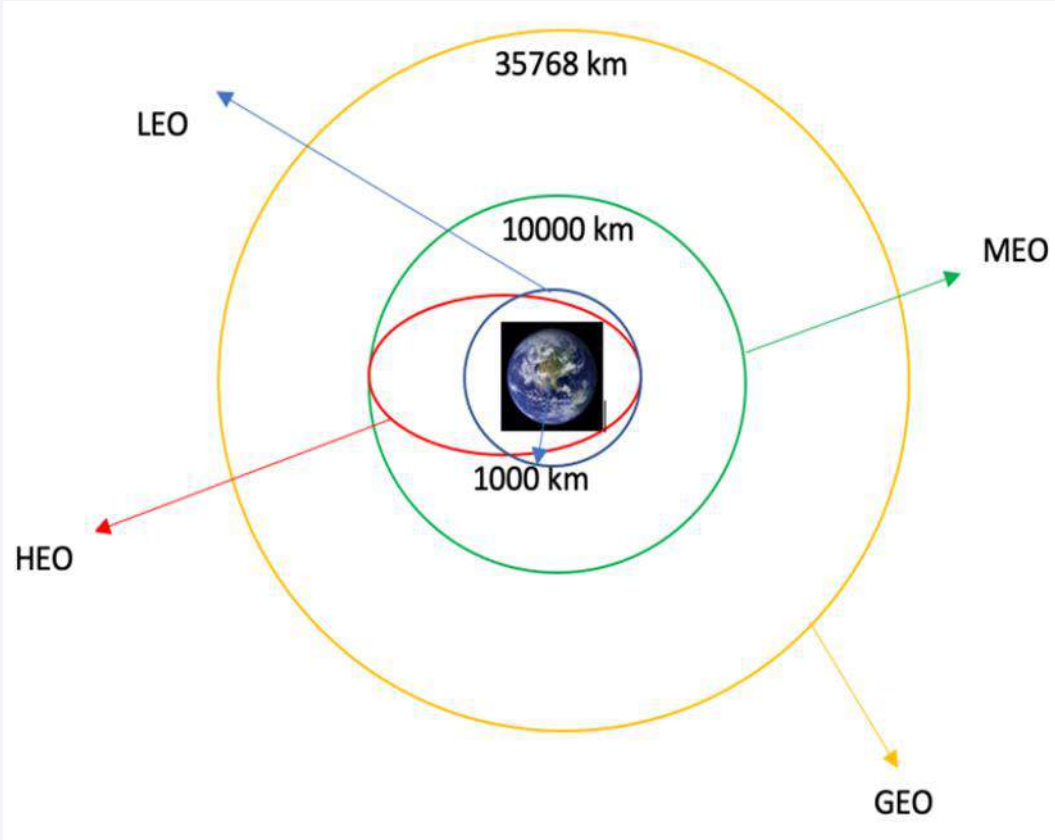
In [8]:
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

In [13]:
```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

Data Wrangling

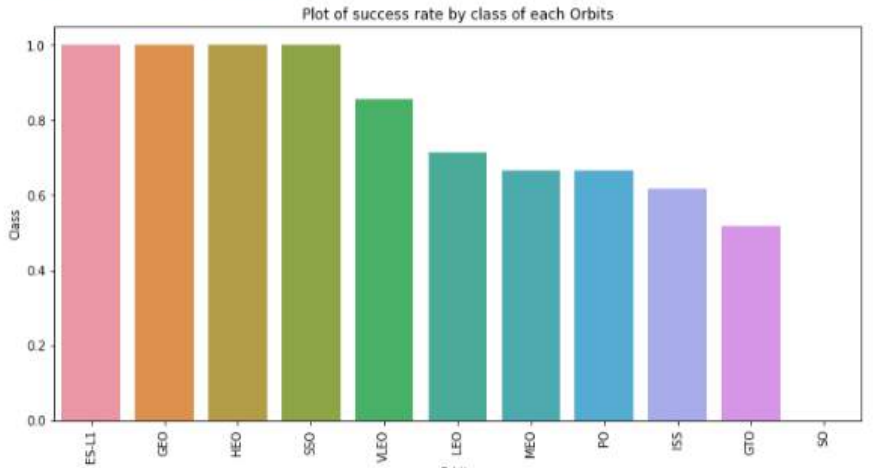
- Exploratory Data Analysis (EDA) was conducted to gain insights into the data. This involved analyzing various aspects of the data, such as the number of launches at each site and the occurrence of different orbits.
- Training labels were determined through the EDA process. This involved identifying patterns and trends in the data, which were used to label the data for the purpose of machine learning.
- The resulting data was exported to a CSV file, which included a landing outcome label that was derived from the outcome column. This file can now be used for further analysis and machine learning tasks.
- The link to the notebook is https://github.com/Janmejaya03/janmejayapharandecapstonestory/blob/main/janmejaya-data-wrangling.ipynb



EDA with Data Visualization

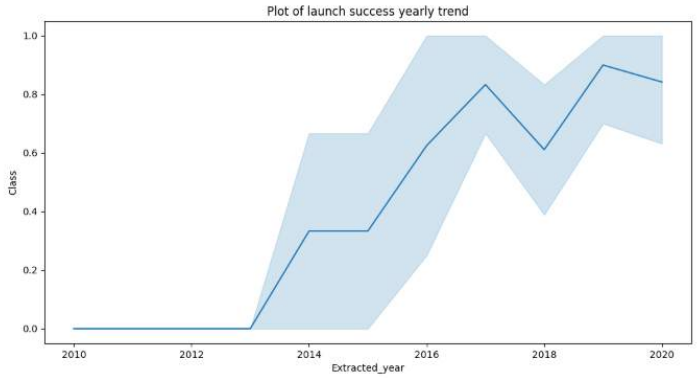
- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly

```
# Use groupby method on Orbit column and get the mean of Class column
# Written By janmejaya Pharande
grouped_orbits = df.groupby(by=['Orbit'])['Class'].mean().sort_values(ascending=False).reset_index()
fig, ax=plt.subplots(figsize=(12,6))
ax = sns.barplot(x = 'Orbit', y = 'Class', data=grouped_orbits)
ax.set_title('Plot of success rate by class of each Orbits', fontdict={'size':12})
ax.set_ylabel('Class', fontsize = 10)
ax.set_xlabel('Orbits', fontsize = 10)
ax.set_xticklabels(ax.get_xticklabels(), fontsize = 10, rotation=90);
```




```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
# Written By janmejaya Pharande
df_copy = df.copy()
df_copy['Extracted_year'] = pd.DatetimeIndex(df['Date']).year

# plot line chart
fig, ax=plt.subplots(figsize=(12,6))
sns.lineplot(data=df_copy, x='Extracted_year', y='Class')
plt.title('Plot of launch success yearly trend');
plt.show()
```



The link to the notebook is https://github.com/Janmejaya03/janmejayapharandecapstonestory/blob/main/janmejaya-eda-visualization.jupyterlite.ipynb

EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook is https://github.com/Janmejaya03/janmejayapharandecapstonestory/blob/main/janmejaya-EDA-with-SQL.ipynb

Build an Interactive Map with Folium

- Launch sites were marked and map objects such as markers, circles, and lines were added to a Folium map to visualize the success or failure of launches for each site. This helped to identify patterns and trends in the data that could be used to make predictions about future launches.
- Launch outcomes were assigned to classes 0 and 1, where 0 represents failure and 1 represents success. This allowed for easy classification and analysis of the data.
- Using the color-labeled marker clusters, high success rate launch sites were identified. This information can be used to inform decisions about where to focus resources for future launches.
- Distances between launch sites and their proximities were calculated, which helped to answer questions such as which launch sites are closest to each other, and which launch sites are most suitable for certain types of missions. This information can be used to optimize launch strategies and reduce costs.
    - Are launch sites near railways, highways and coastlines.
    - Do launch sites keep certain distance away from cities.

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The link to the notebook is https://github.com/Janmejaya03/janmejayapharandecapstonestory/blob/main/Plotly%20Janmejaya%20code.py

Predictive Analysis (Classification)

- The data was loaded and transformed using NumPy and Pandas. The data was then split into training and testing sets to prepare for the machine learning models.
- Multiple machine learning models were built, and hyperparameters were tuned using GridSearchCV. This allowed for optimization of the models and improved their accuracy.
- Feature engineering and algorithm tuning were used to further improve the model's accuracy. The accuracy metric was used to evaluate the performance of the model and guide the optimization process.
- We found the best performing classification model.
- The link to the notebook is https://github.com/Janmejaya03/janmejayapharandecapstonestory/blob/main/Janmejaya-SpaceX-Machine-Learning.ipynb

Results

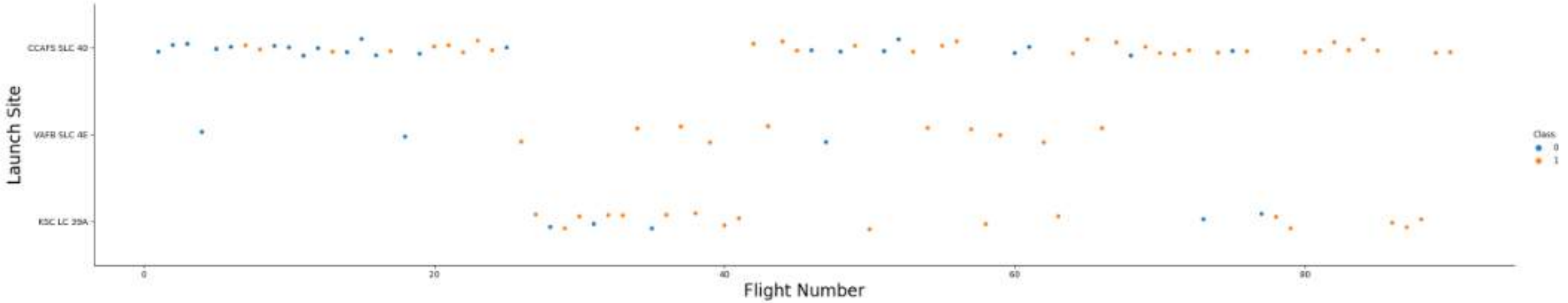
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

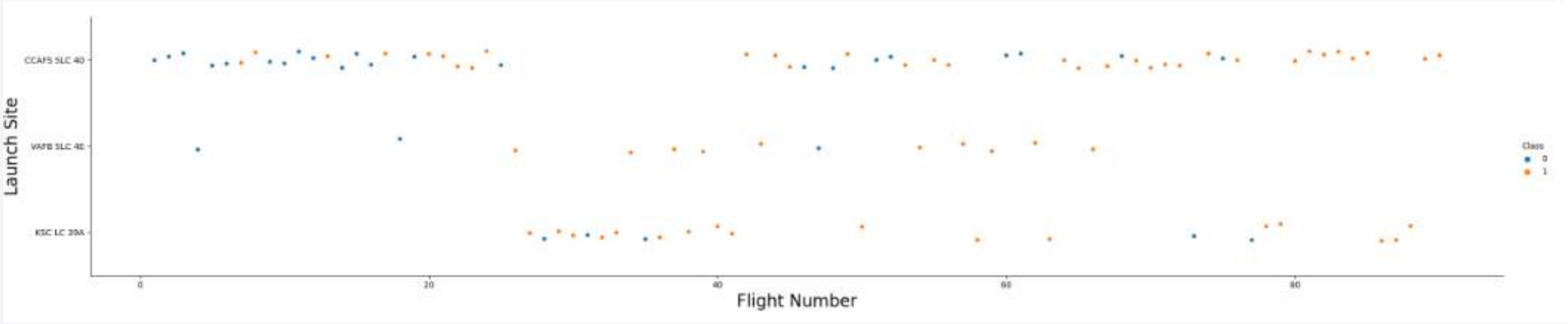
Flight Number vs. Launch Site

```
### TASK 1: Visualize the relationship between Flight Number and Launch Site
# Written By janmejaya Pharande
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```



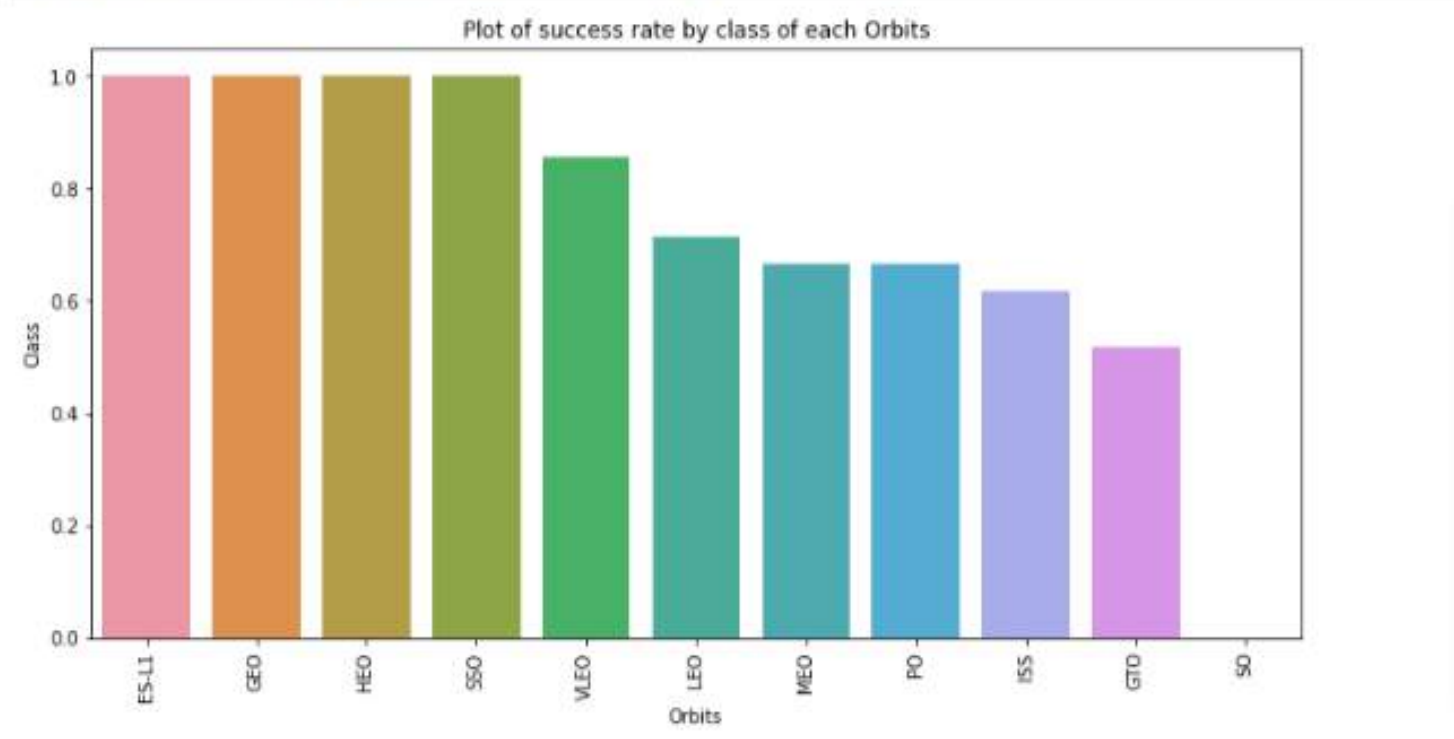
Payload vs. Launch Site

```
### TASK 2: Visualize the relationship between Payload and Launch Site
# Written By janmejaya Pharande
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



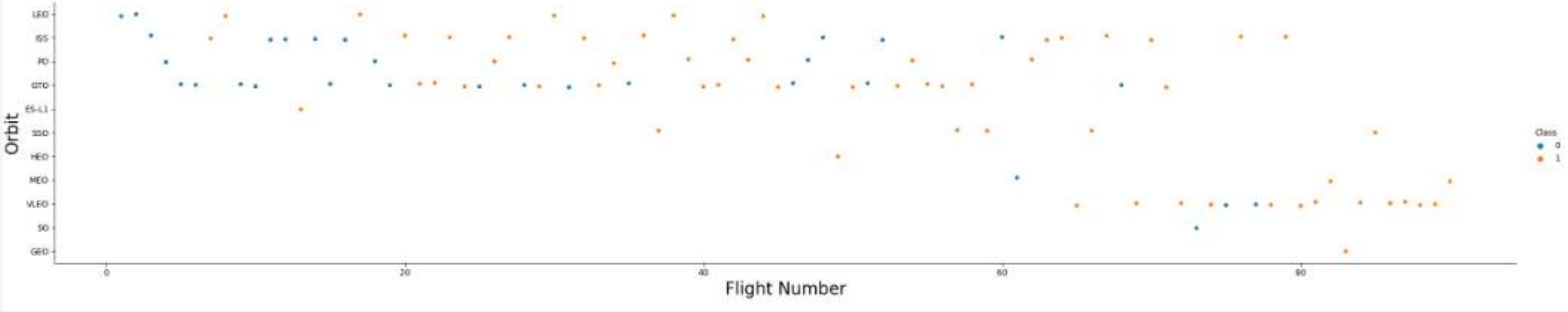
Success Rate vs. Orbit Type

```
# Use groupby method on Orbit column and get the mean of Class column
# Written By janmejaya Pharande
grouped_orbits = df.groupby(by=['Orbit'])['Class'].mean().sort_values(ascending=False).reset_index()
fig, ax=plt.subplots(figsize=(12,6))
ax = sns.barplot(x = 'Orbit', y = 'Class', data=grouped_orbits)
ax.set_title('Plot of success rate by class of each Orbits', fontdict={'size':12})
ax.set_ylabel('Class', fontsize = 10)
ax.set_xlabel('Orbits', fontsize = 10)
ax.set_xticklabels(ax.get_xticklabels(), fontsize = 10, rotation=90);
```



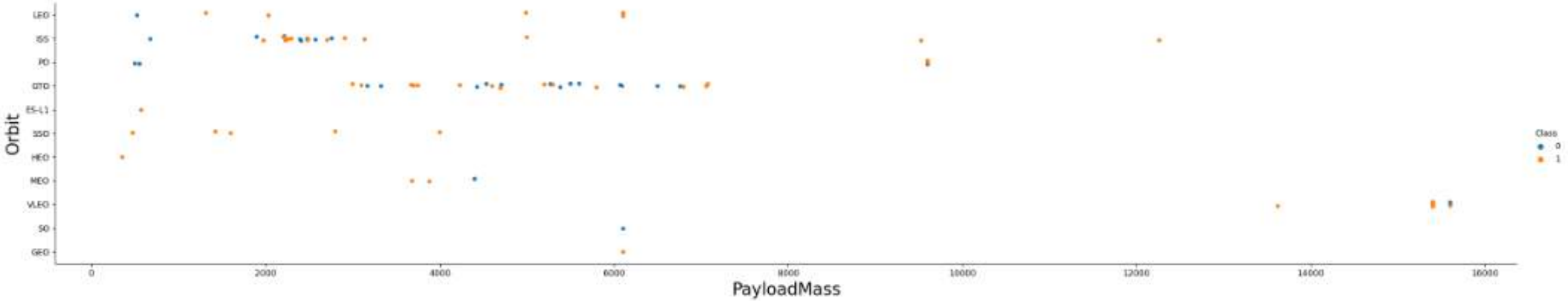
Flight Number vs. Orbit Type

```
### TASK  4: Visualize the relationship between FlightNumber and Orbit type
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Payload vs. Orbit Type

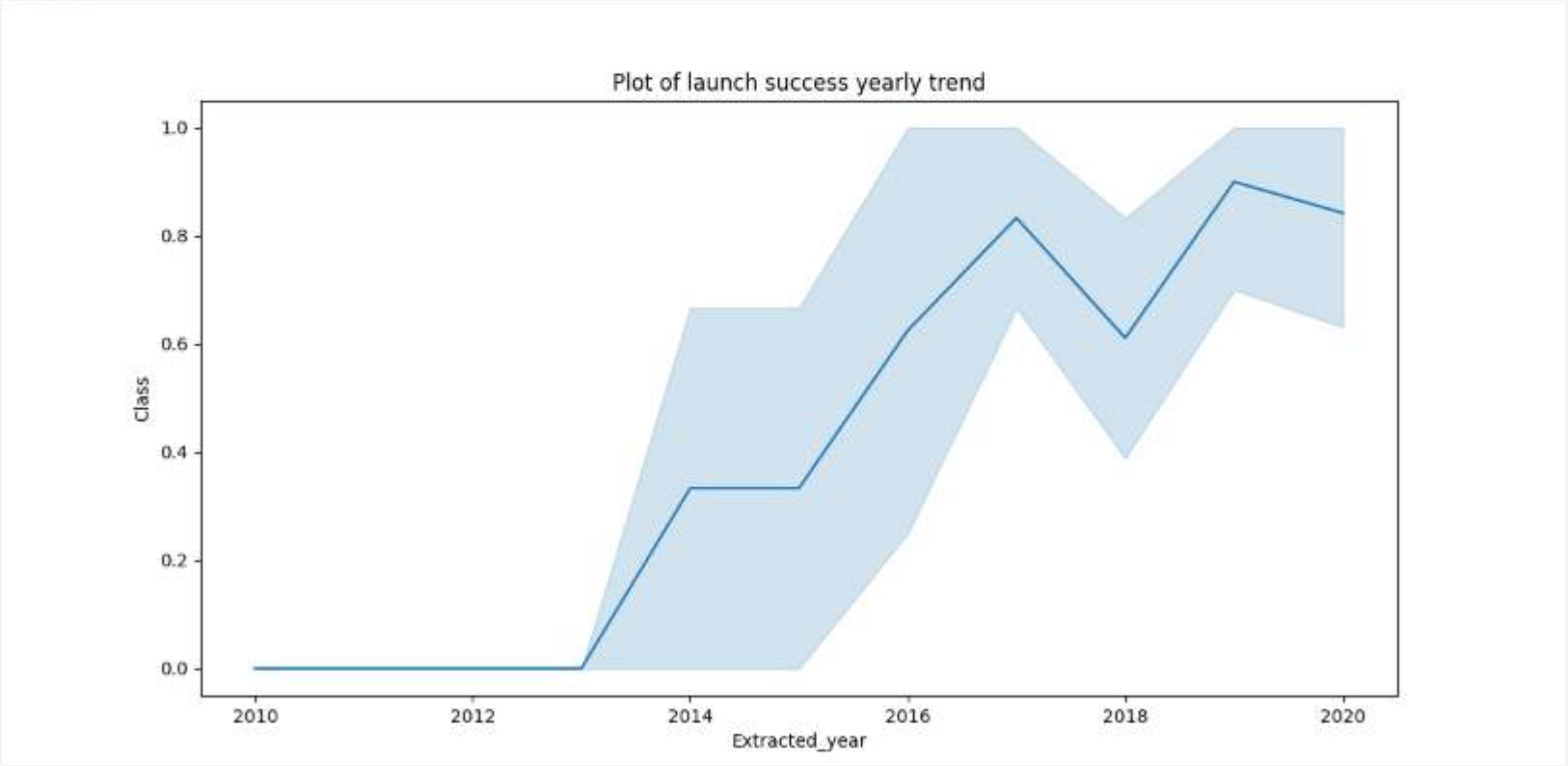
```
### TASK  5: Visualize the relationship between Payload and Orbit type
# Written By janmejaya Pharande
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
# Written By Janmejaya Pharande
df_copy = df.copy()
df_copy['Extracted_year'] = pd.DatetimeIndex(df['Date']).year

# plot line chart
fig, ax=plt.subplots(figsize=(12,6))
sns.lineplot(data=df_copy, x='Extracted_year', y='Class')
plt.title('Plot of launch success yearly trend');
plt.show()
```



All Launch Site Names

- We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
# Written By janmejaya Pharande
```

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

Task 2

Display 5 records where launch sites begin with the string 'CCA'

```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
# Written By janmejaya Pharande
```

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
# Written By janmejaya Pharande
```

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

Task 4

Display average payload mass carried by booster version F9 v1.1

```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
# Written By janmejaya Pharande
```

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

Task 5

List the date when the first successful landing outcome in ground pad was acheived.

Hint:Use min function

```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
# Written By janmejaya Pharande
```

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
# Written By janmejaya Pharande
```

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
# Written By janmejaya Pharande
```

```
The total number of successful mission outcome is:
```

	successoutcome
0	100

```
The total number of failed mission outcome is:
```

	failureoutcome
0	1

Boosters Carried Maximum Payload

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                               SELECT MAX(PayloadMassKG)
                               FROM SpaceX
                               )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
# Written By janmejaya Pharande
```

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600

2015 Launch Records

Task 9

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
# Written By janmejaya Pharande
```

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

Task 10

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

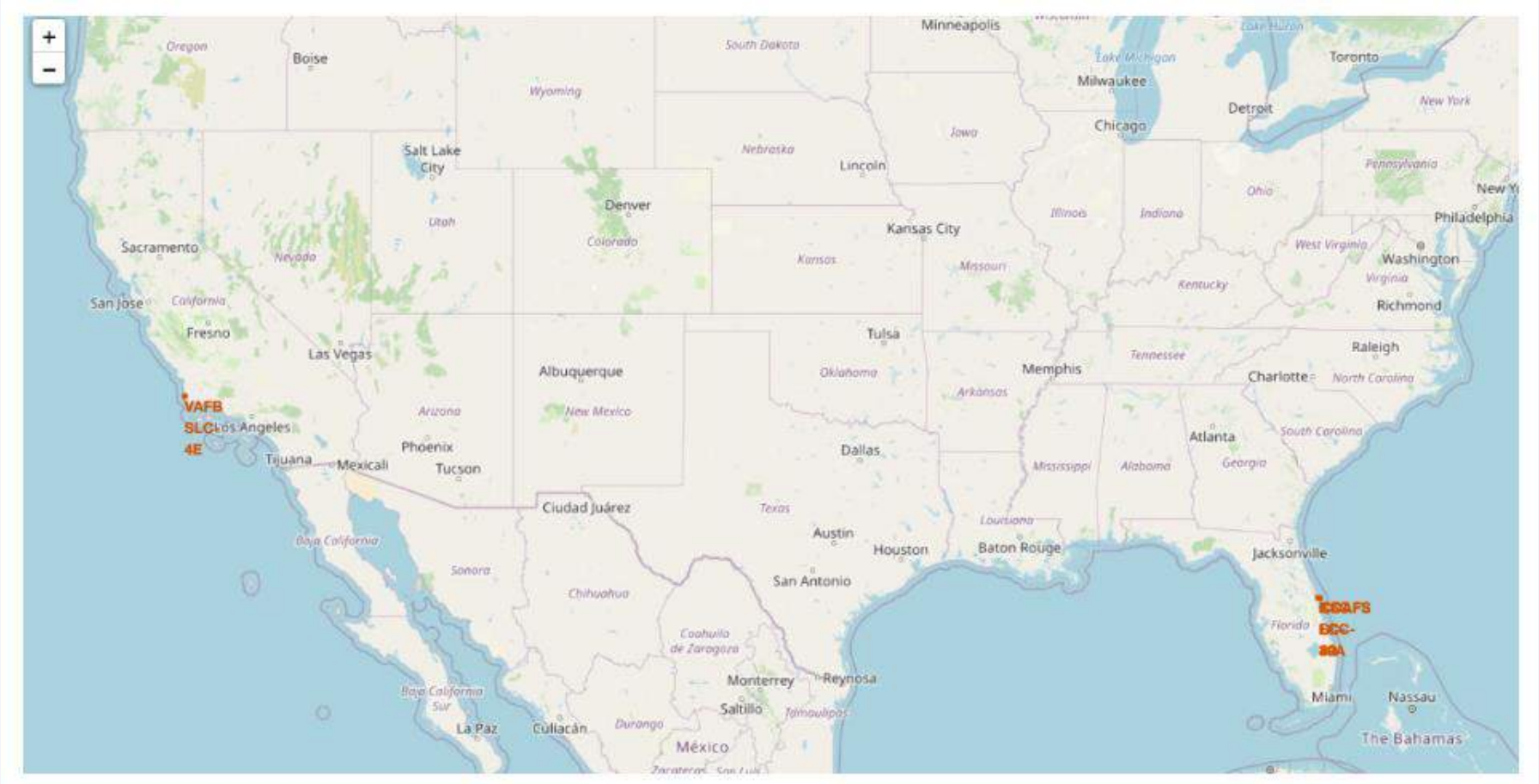
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
# Written By janmejaya Pharande
```

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Section 3

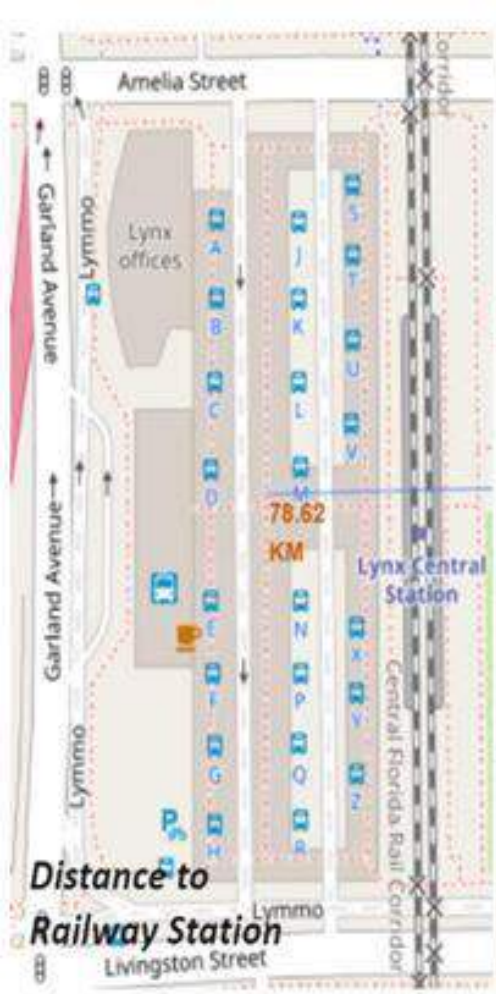
Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

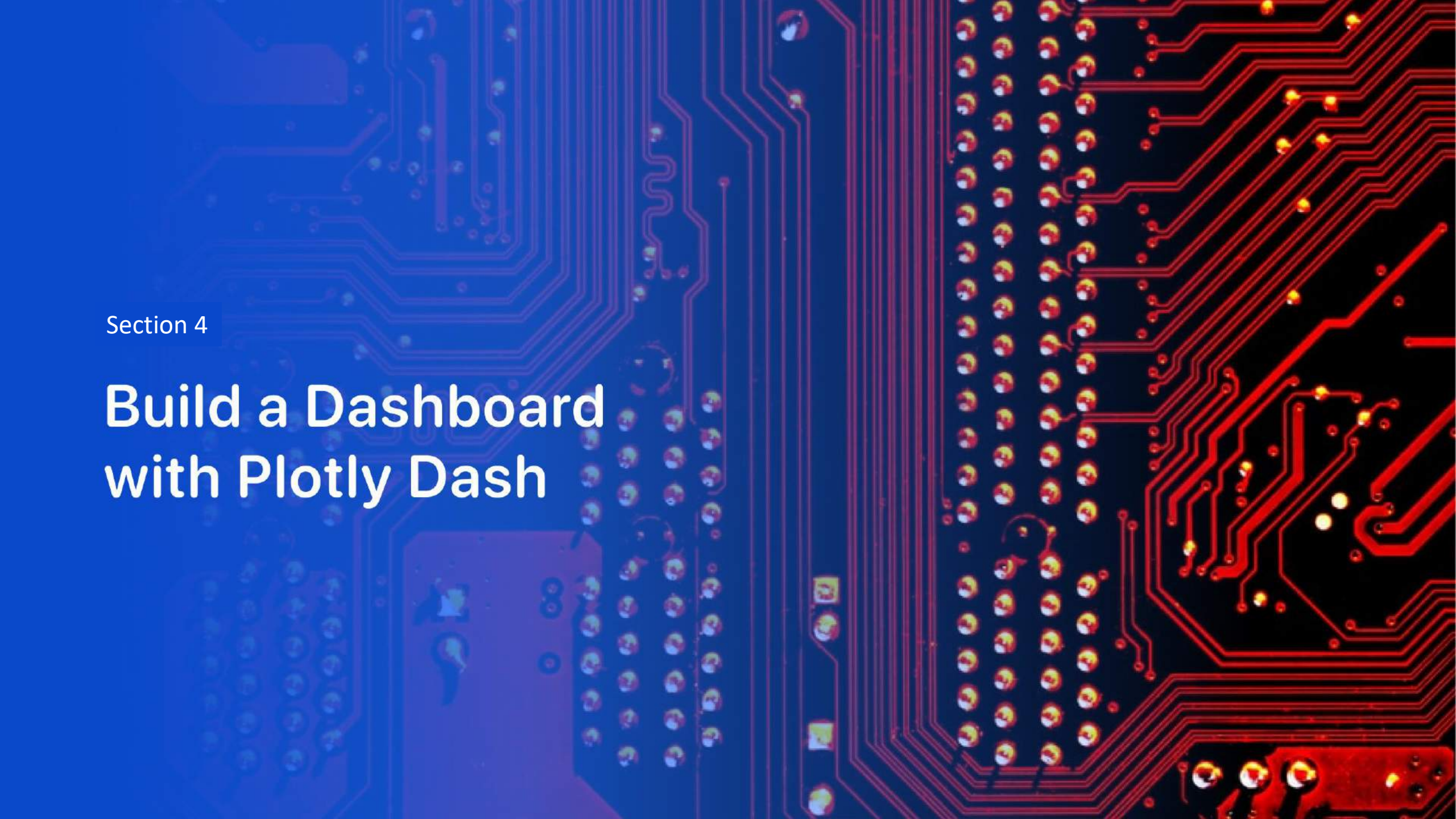


<Folium Map Screenshot 2>

<Folium Map Screenshot 3>

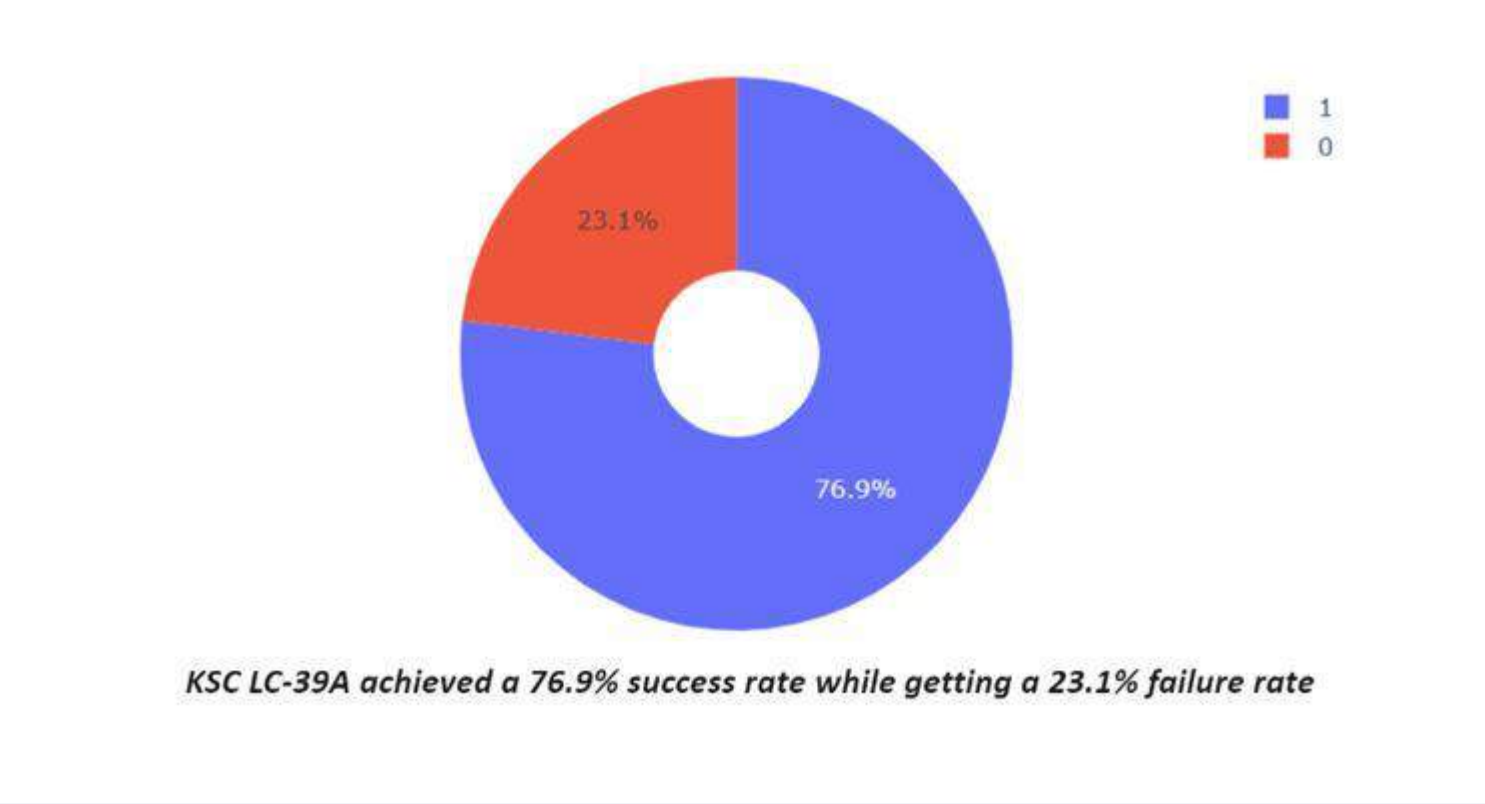


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

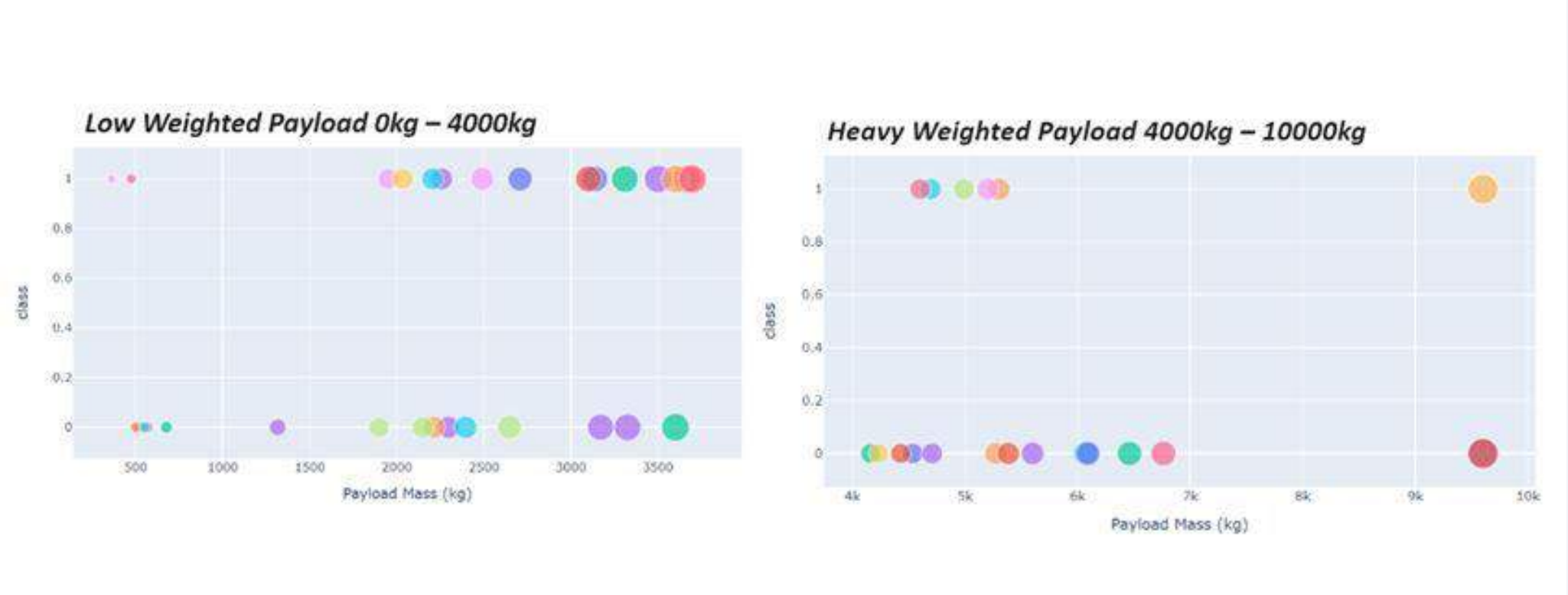

Section 4
Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>

<Dashboard Screenshot 2>



<Dashboard Screenshot 3>



Section 5

Predictive Analysis (Classification)

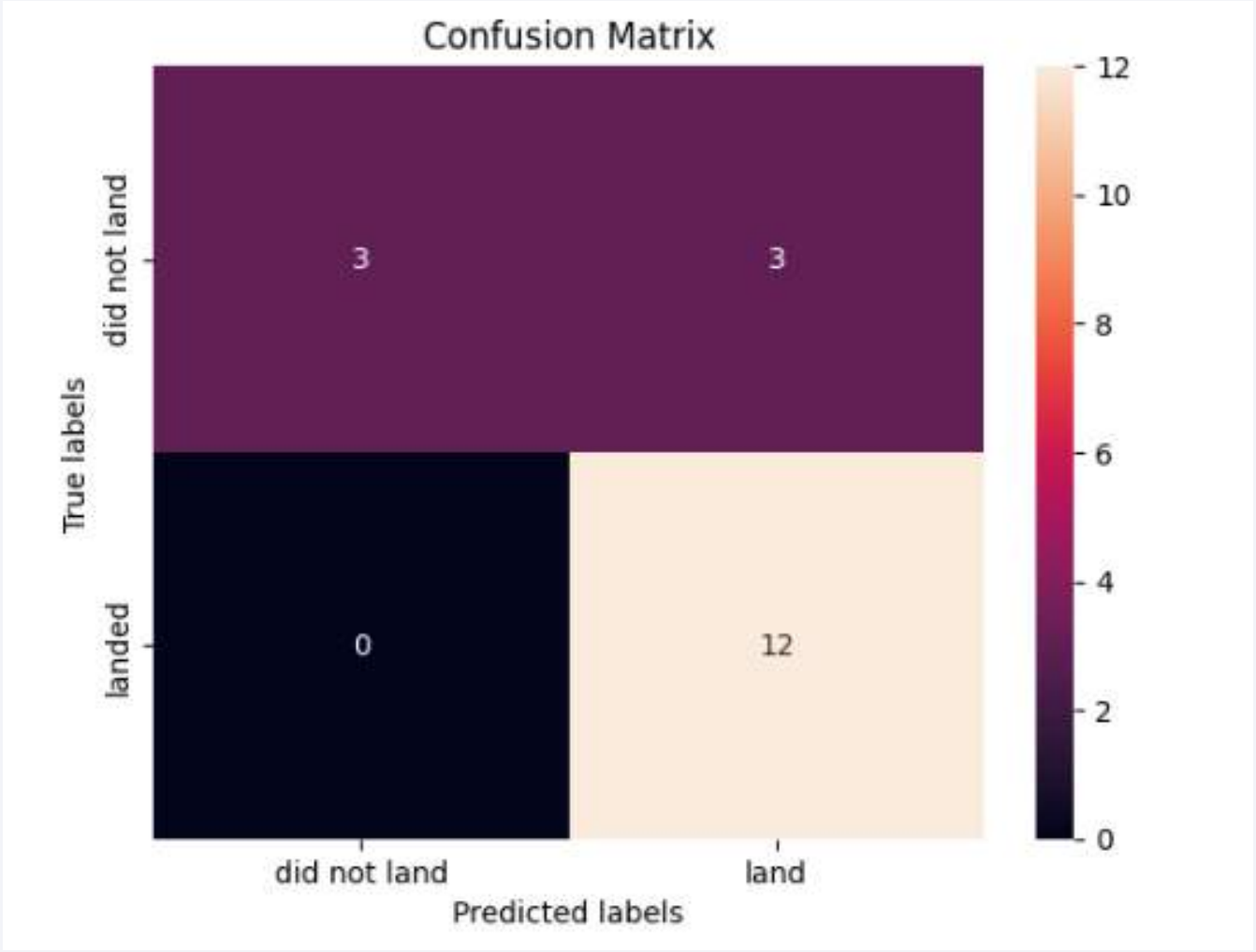
Classification Accuracy

```
# Made By Janmejaya Pharande
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix



Conclusions

1. Launch success rate appears to be positively correlated with flight amount at a launch site.
2. Launch success rates have been increasing steadily from 2013 to 2020.
3. Orbits such as ES-L1, GEO, HEO, SSO, and VLEO have had the highest success rates.
4. KSC LC-39A has had the most successful launches compared to any other site.
5. The Decision tree classifier is considered the best machine learning algorithm for this particular task.

Thank you!