IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<EYERUSALEM>
<DEC 06, 2024>

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

- Project background and context
- SpaceX promotes Falcon 9 rocket launches on its website for 62 million dollars, significantly less than other providers charging over 165 million dollars each, mainly due to the ability of reusing the first stage.
- Problems you want to find answers
- The project involves predicting the successful landing of the first stage of the SpaceX Falcon 9 rocket.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API and (Web Scrapping) from Wikipedia
- Perform data wrangling
  - One Hot Encoding data fields for Machine Learning and dropping irrelevant columns
- 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

- The data was gathered through different techniques.
- Data was collected by making a get request to the SpaceX API. Then, the response content was decoded as JSON using the .json() function and transformed into a pandas dataframe using .json_normalize(). The data was cleaned, missing values were identified, and necessary fill-ins were done. Additionally, web scraping was conducted on Wikipedia for Falcon 9 launch records using BeautifulSoup. The aim was to extract the launch records as an HTML table, parse the table, and convert it to a pandas dataframe for further analysis.

Data Collection – SpaceX API

- We utilized the get request to the SpaceX API for data collection, then proceeded to clean, perform basic data wrangling, and format the obtained data.
- GitHub URL of the completed SpaceX API calls notebook https://github.com/Eyerusalem-Mulugeta/-IBM-DataScience-SpaceX-Capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head()
```

static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	cap

[{'time': 33

Data Collection - Scraping

- We utilized web scraping with BeautifulSoup to extract Falcon 9 launch records. Subsequently, we parsed the table and transformed it into a pandas dataframe.

GitHub URL
https://github.com/Eyerusalem-Mulugeta/-IBM-DataScience-SpaceX-Capstone/blob/main/jupyter-labs-webscraping.ipynb

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
# use requests.get() method with the provided static_url
# assign the response to a object
page = requests.get(static_url)
page.status_code
```

```
200
```

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(page.text, 'html.parser')
```

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

```
# Use soup.title attribute
soup.title
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Data Wrangling

- We conducted exploratory data analysis to identify the training labels.
- We computed the frequency of launches at each site and the occurrence of orbits.
- We derived the landing outcome label from the outcome column and saved the results to a CSV file.
- GitHub URL https://github.com/Eyerusalem-Mulugeta/-IBM-DataScience-SpaceX-Capstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- We visually analyzed the data to examine the correlation between flight number and launch site, payload and launch site, success rates for each orbit type, flight number and orbit type, as well as the annual trend in launch success.
- GitHub URL https://github.com/Eyerusalem-Mulugeta/-IBM-DataScience-SpaceX-Capstone/blob/main/edadataviz.ipynb

EDA with SQL

- We imported the SpaceX dataset into a PostgreSQL database directly from the Jupyter notebook.
- We utilized SQL for Exploratory Data Analysis (EDA) to gain insights from the data. We constructed queries to uncover details such as:
- The unique launch site names in the space mission.
- The total payload mass of boosters launched by NASA (CRS).
- The average payload mass of booster version F9 v1.1.
- The total count of successful and failed mission outcomes.
- The details of failed landing outcomes on drone ships, including the booster version and launch site names.
- GitHub URL https://github.com/Eyerusalem-Mulugeta/-IBM-DataScience-SpaceX-Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite%20(1).ipynb

Build an Interactive Map with Folium

• We labeled all launch sites and incorporated map elements like markers, circles, and lines to indicate the success or failure of launches at each site on the folium map.

• We categorized the launch outcomes (failure or success) as class 0 and 1, with 0 representing failure and 1 representing success.

• By using color-coded marker clusters, we determined the launch sites with higher success rates.

• We computed the distances between a launch site and its surroundings to address queries such as:

- Proximity of launch sites to railways, highways, and coastlines.
- Whether launch sites maintain a certain distance from cities.
- GitHub URL https://github.com/Eyerusalem-Mulugeta/-IBM-DataScience-SpaceX-Capstone/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

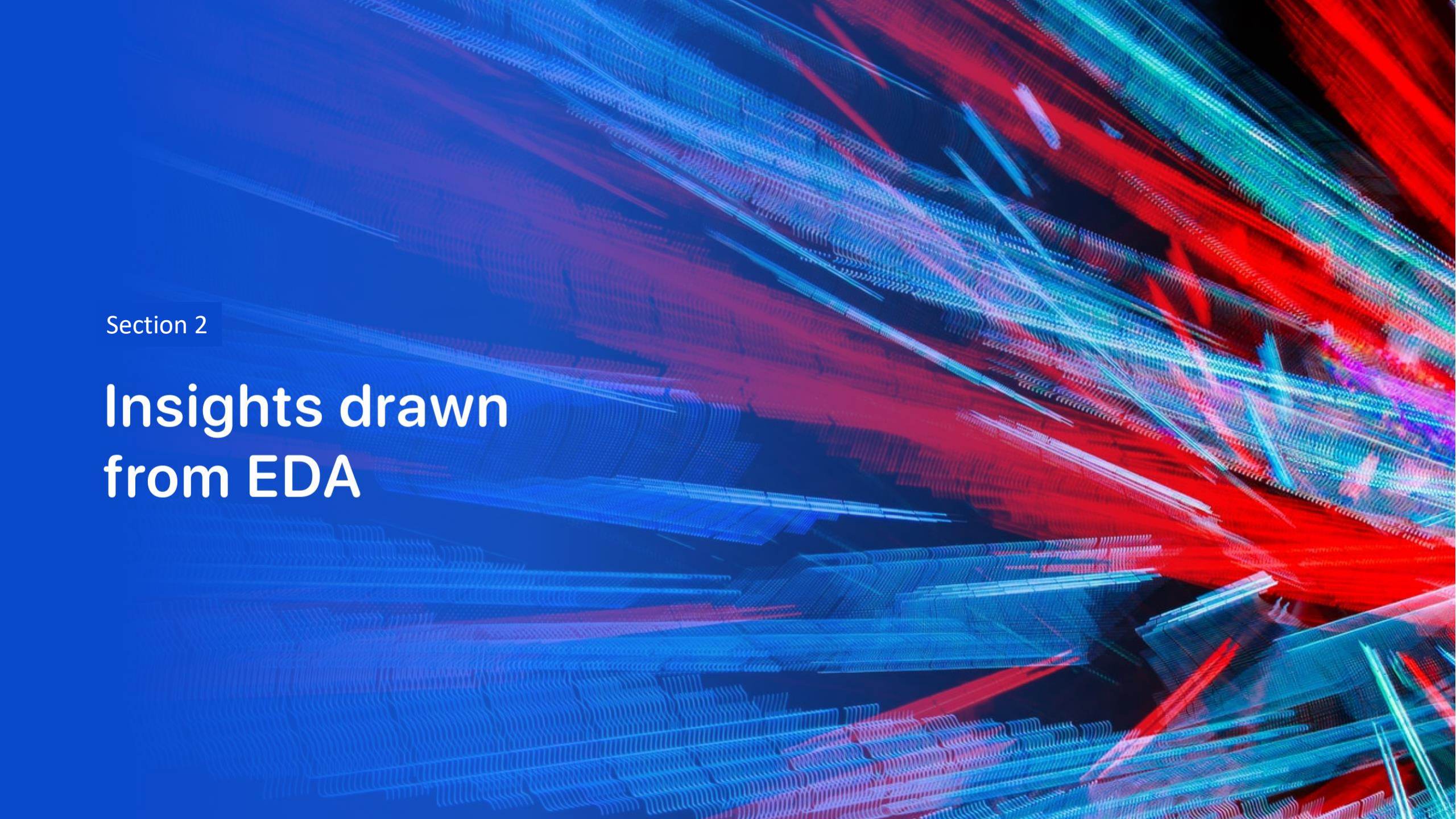
- An interactive dashboard was created using Plotly Dash.
- Pie charts were generated to display the total launches at specific sites.
- Scatter plots were created to illustrate the relationship between Outcome and Payload Mass (Kg) for various booster versions.
- GitHub URL https://github.com/Eyerusalem-Mulugeta/-IBM-DataScience-SpaceX-Capstone/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- The data was loaded and processed using numpy and pandas, then divided into training and testing sets.
- Various machine learning models were constructed and their hyperparameters were adjusted using GridSearchCV.
- Accuracy was chosen as the evaluation metric, and the model was enhanced through feature engineering and algorithm tuning.
- The classification model with the best performance was identified.
- GitHub https://github.com/Eyerusalem-Mulugeta/-IBM-DataScience-SpaceX-Capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

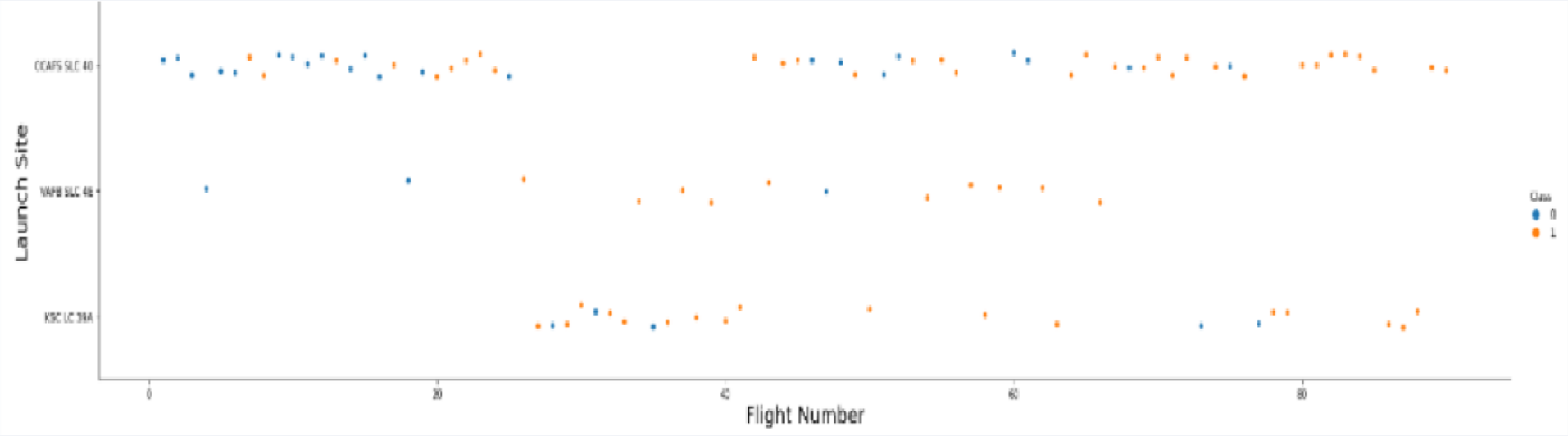
Results

- Exploratory data analysis results
- Both API and web scraping methods can effectively gather Xspace data.
- Interactive analytics demo in screenshots
- Using SQL for Exploratory Data Analysis (EDA) is efficient for filtering data. Interactive visualization during EDA offers valuable and informative insights. Plotly Dash is robust in displaying real-time data changes effectively.
- Predictive analysis results
- The Decision Tree Classifier Algorithm demonstrates the highest accuracy in prediction


Section 2
Insights drawn from EDA

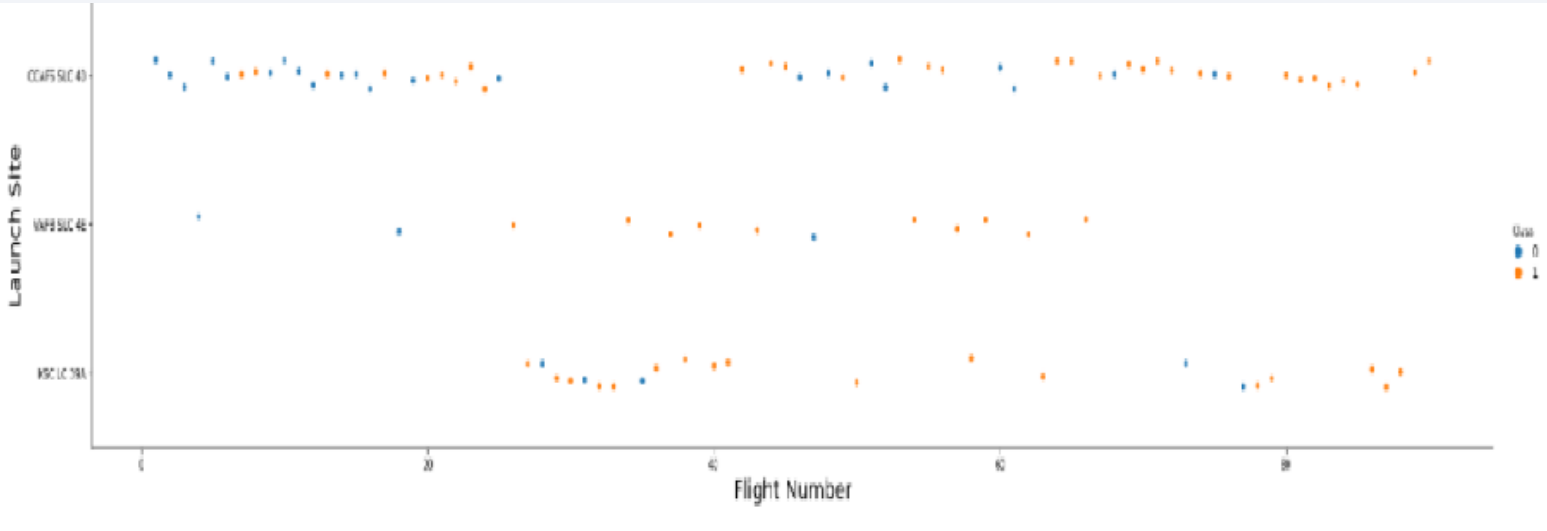
Flight Number vs. Launch Site

- Based on the plot analysis, it was observed that as the number of flights increases at a specific launch site, the success rate at that site also tends to rise. This suggests a positive correlation between the frequency of flights and the success rate, indicating that sites with higher flight volumes are associated with higher success rates in launches.



Payload vs. Launch Site

The higher the payload mass at launch site CCAFS SLC 40, the greater the success rate for the rocket.

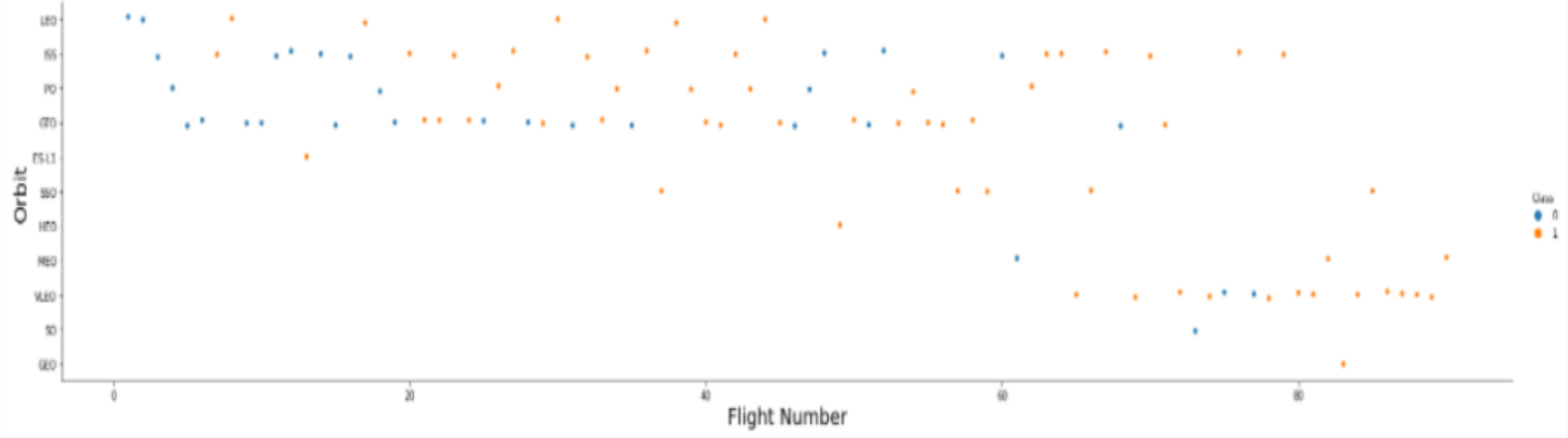


Success Rate vs. Orbit Type

- The plot illustrates that ES-L1, GEO, HEO, SSO, and VLEO orbits had the highest success rates, indicating their reliability in achieving successful launches.

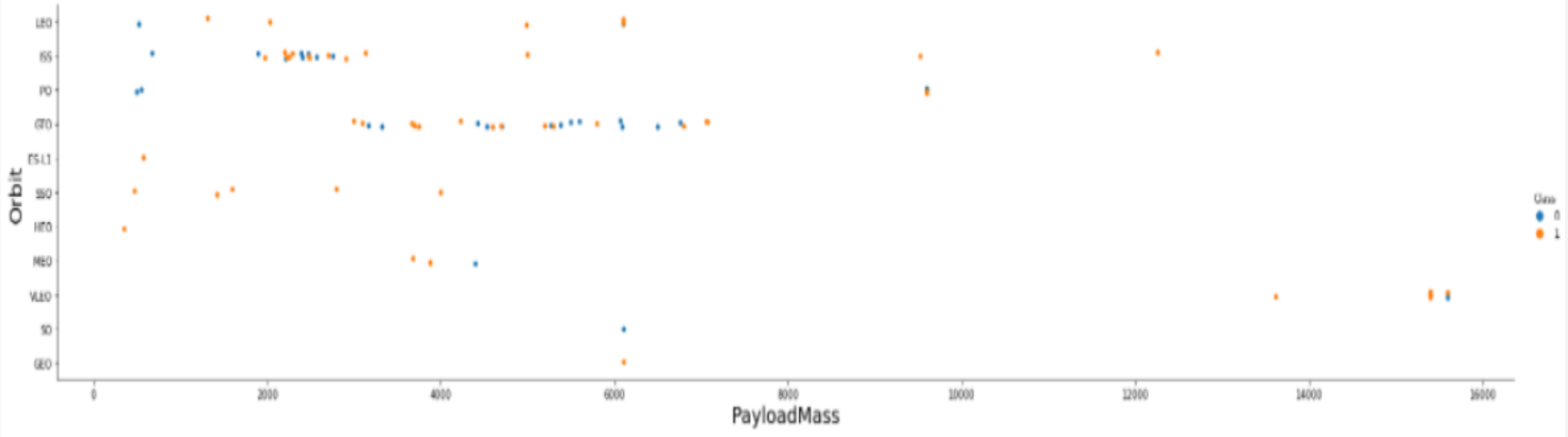
Flight Number vs. Orbit Type

- The graph depicts the correlation between Flight Number and Orbit type. It is evident that success in the LEO orbit is influenced by the number of flights, while for the GTO orbit, there is no discernible relationship between flight number and the orbit's success rate.



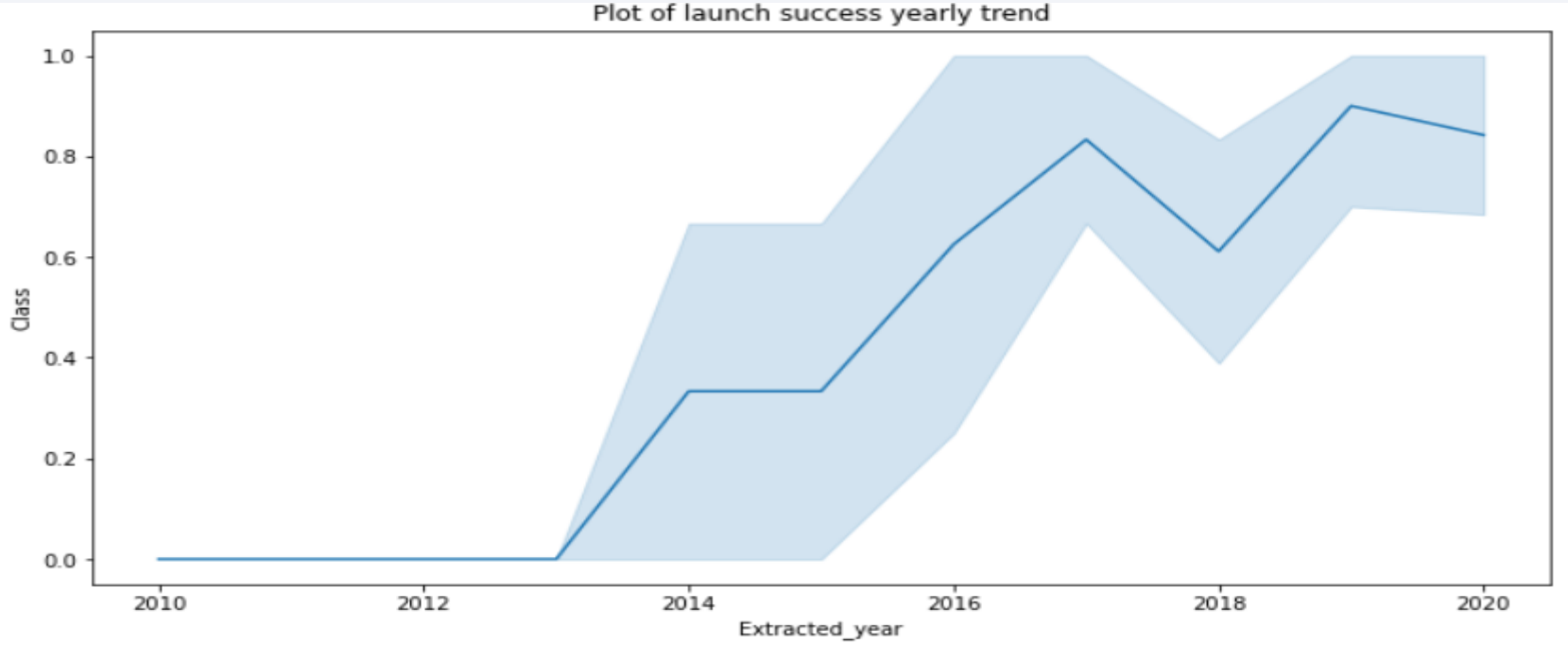
Payload vs. Orbit Type

- It is noticeable that successful landings are more frequent for Payload Orbits (PO), Low Earth Orbits (LEO), and International Space Station (ISS) orbits when carrying heavy payloads.



Launch Success Yearly Trend

- According to the plot, the success rate has shown a consistent increase from 2013 to 2020.



All Launch Site Names

- We utilized the keyword DISTINCT to display solely the unique launch sites from the SpaceX dataset.

Display the names of the unique launch sites in the space mission

```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

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

Launch Site Names Begin with 'CCA'

- We employed the aforementioned query to exhibit 5 entries where the launch sites commence with 'CCA'.

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)
```

	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

- We computed the total payload transported by NASA boosters as 45,596 using the following query.

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)
```

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

- The average payload mass transported by booster version F9 v1.1 was calculated to be 2928.4.

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)
```

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- It was noted that the date of the initial successful landing outcome on a ground pad was December 22, 2015.

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

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We utilized the WHERE clause to filter boosters that achieved successful landings on a drone ship, and then applied the AND condition to identify successful landings with a payload mass ranging between 4000 and 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)
```

	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

- We employed wildcard characters such as '%' to filter for cases where the Mission Outcome was either a success or a failure.

```
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)
```

```
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

- We used a subquery in the WHERE clause with the MAX() function to identify the booster with the highest payload capacity.

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)
```

	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
11	F9 B5 B1060.3	15600

2015 Launch Records

- We employed a combination of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes on drone ships, including their booster versions and launch site names specifically for the year 2015.

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)
```

	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

- We chose Landing outcomes and the COUNT of landing outcomes from the dataset, filtering for landing outcomes between June 4, 2010, and March 20, 2010, using the WHERE clause. The GROUP BY clause was then utilized to group the landing outcomes, and the ORDER BY clause was applied to arrange the grouped landing outcomes in descending order

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

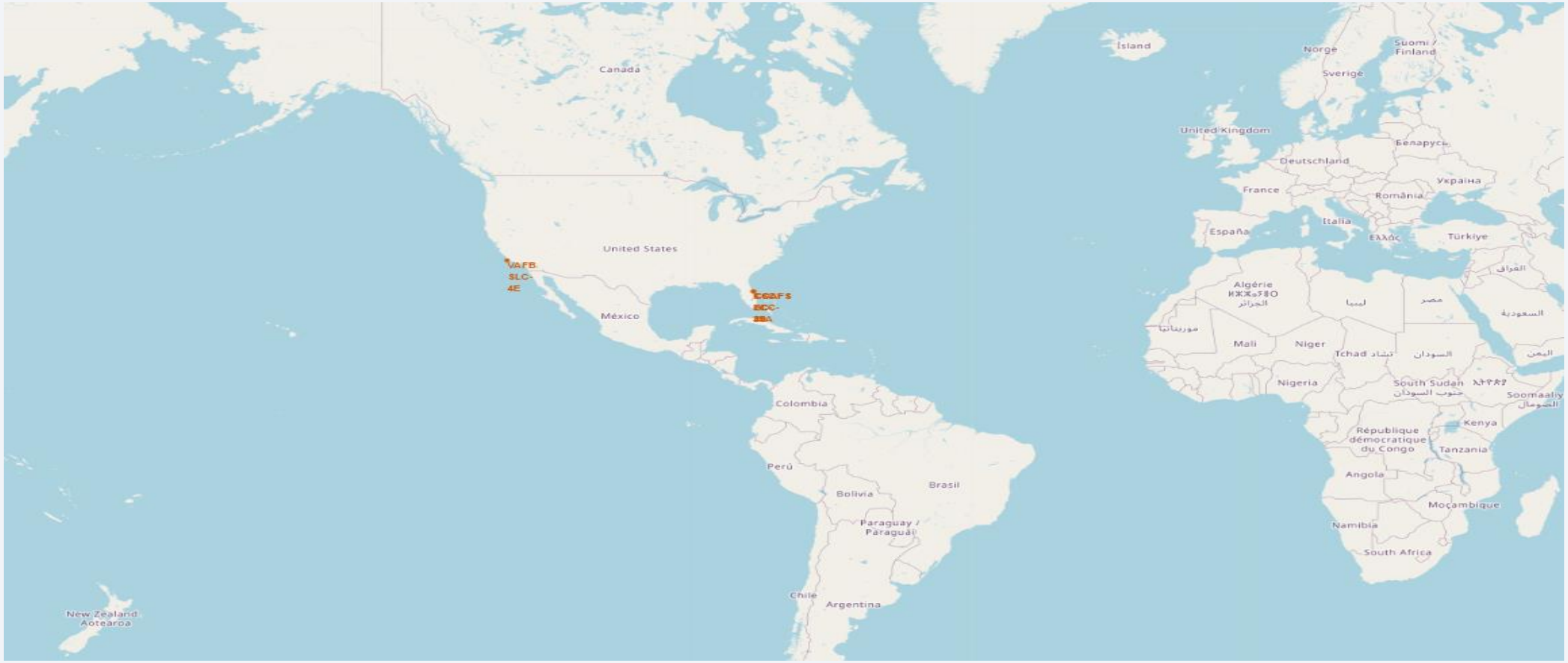
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

	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

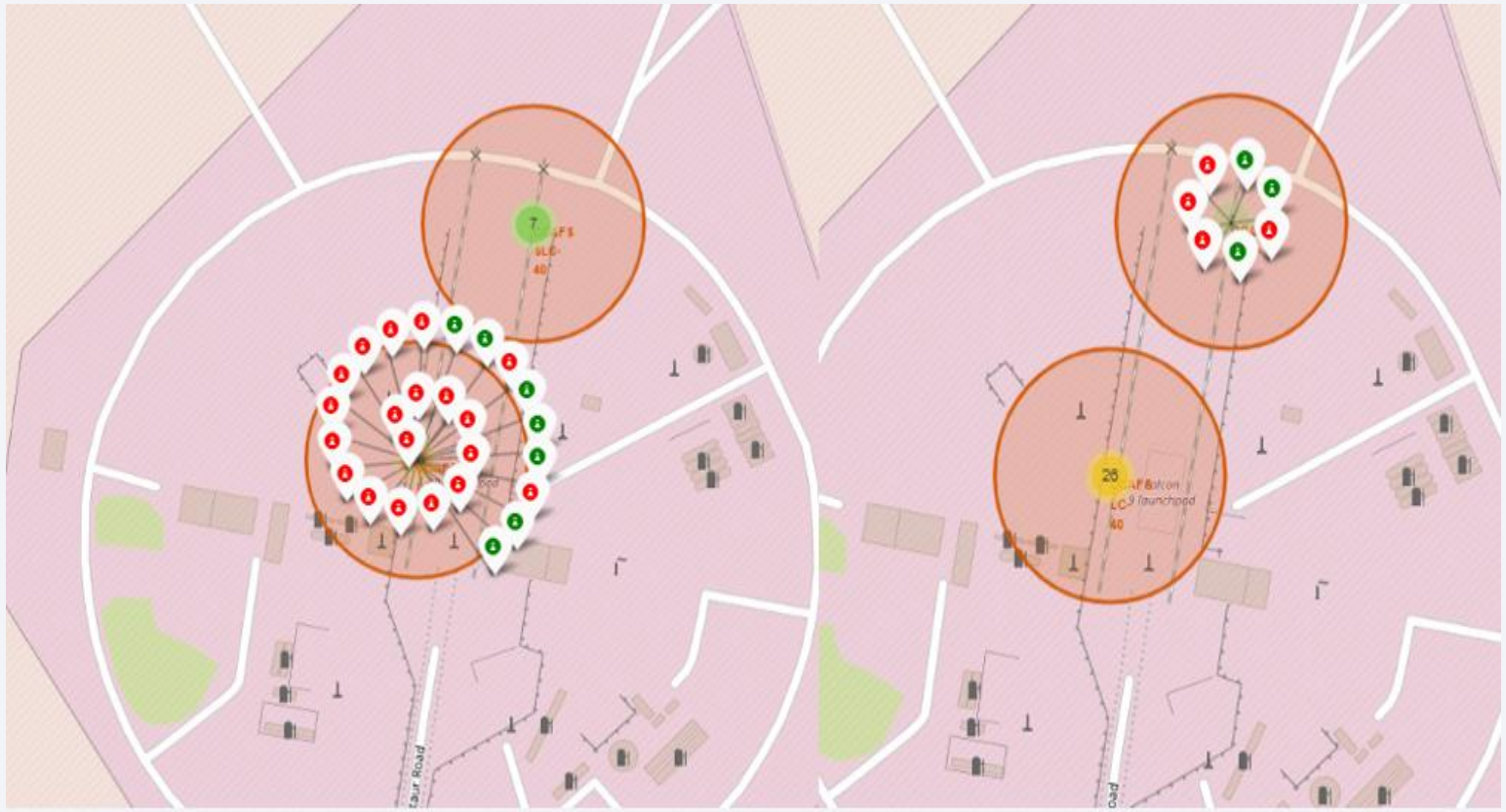
Section 3

Launch Sites Proximities Analysis

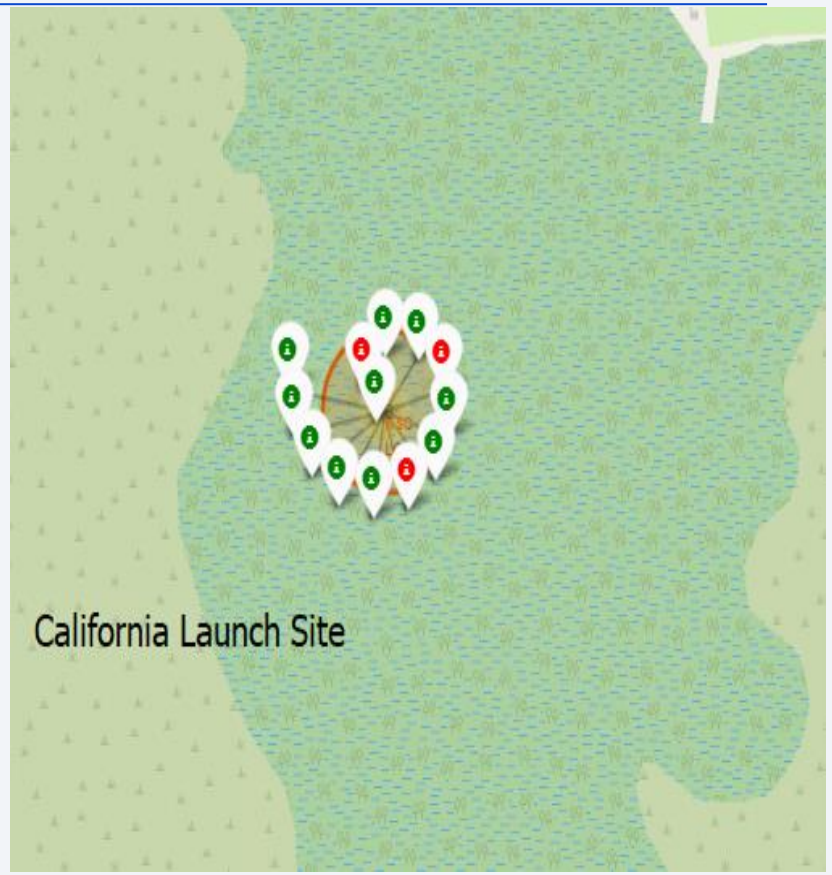
All launch sites global map markers



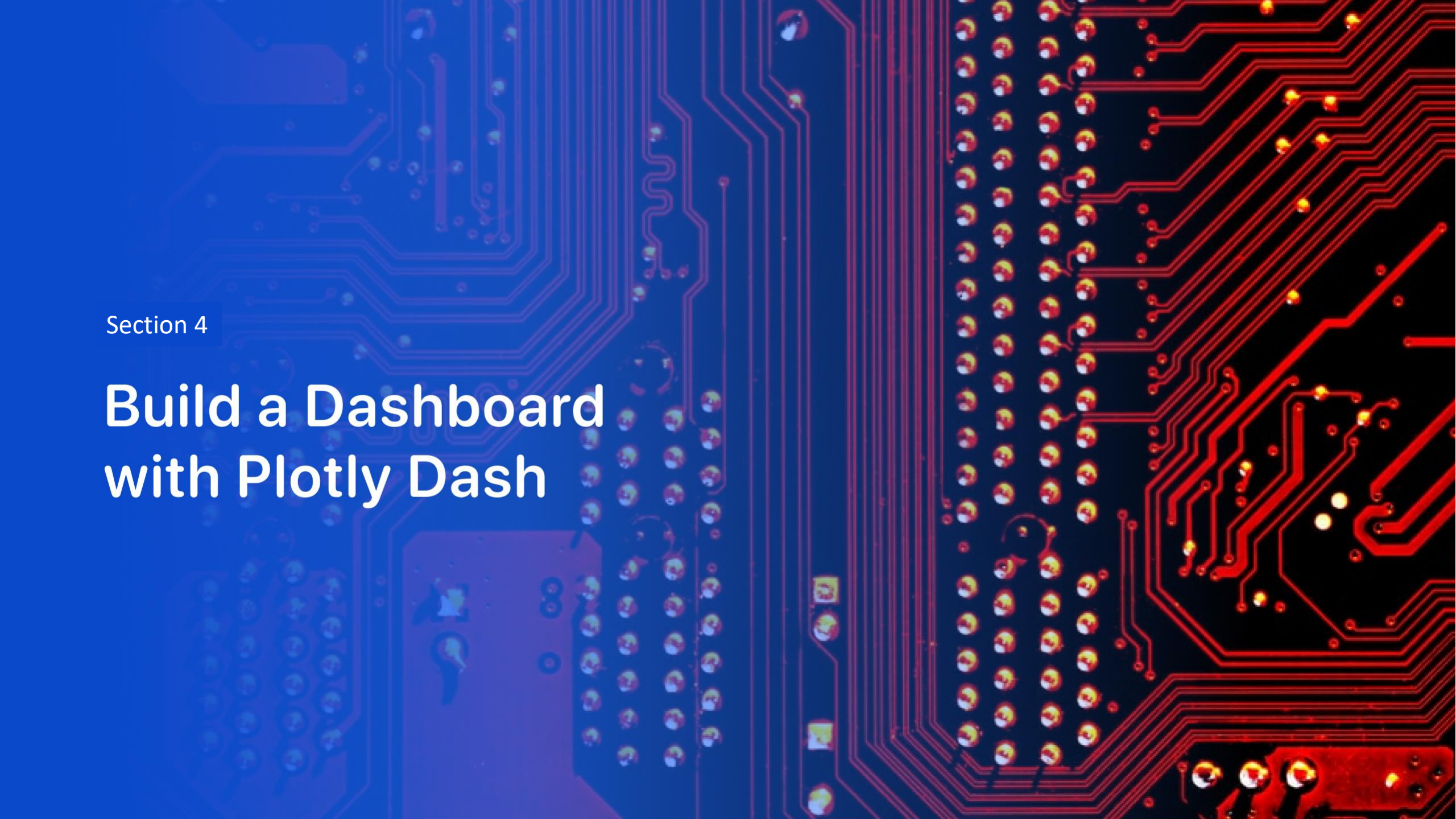
Markers displaying launch sites with coloured tags



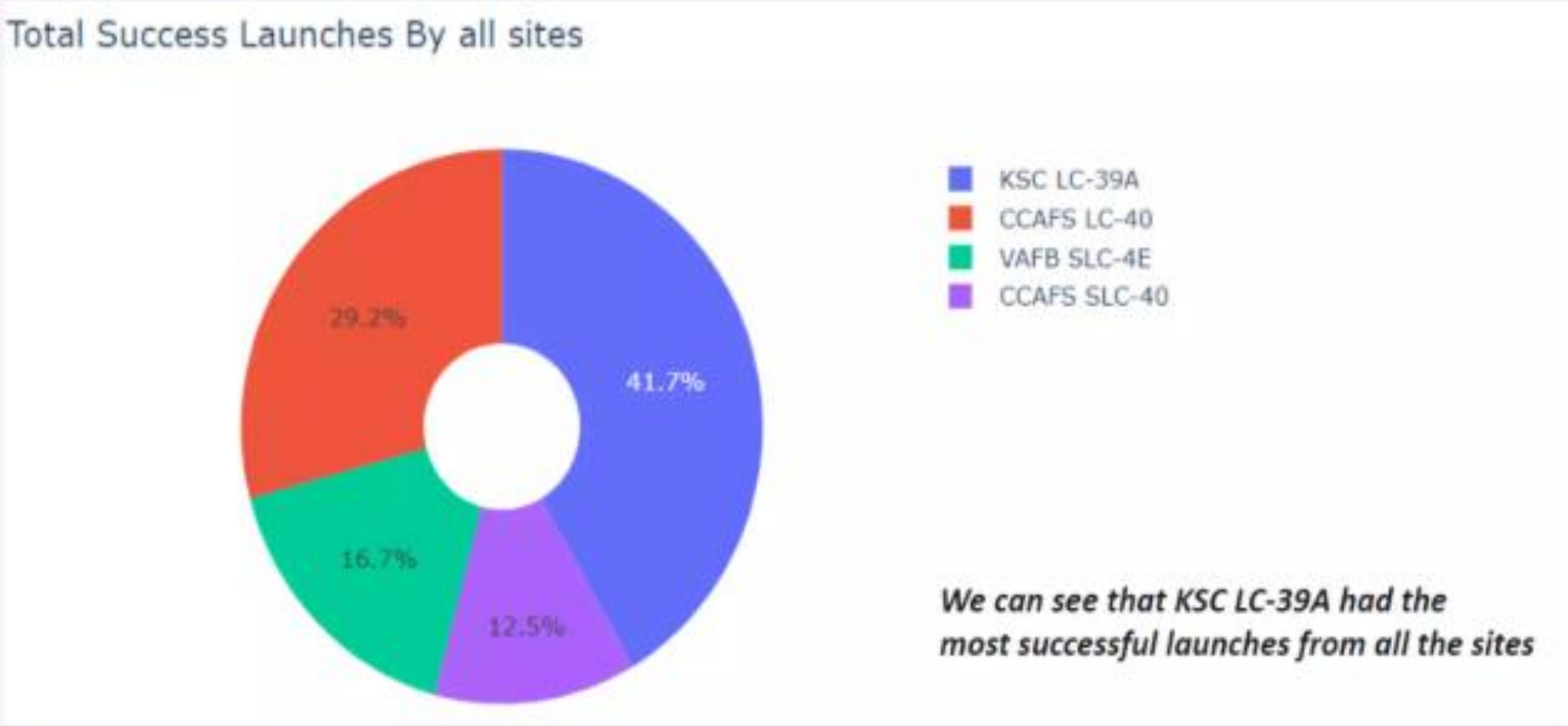
Florida Launch Sites



Distance of Launch Site to Landmarks


Section 4
Build a Dashboard with Plotly Dash

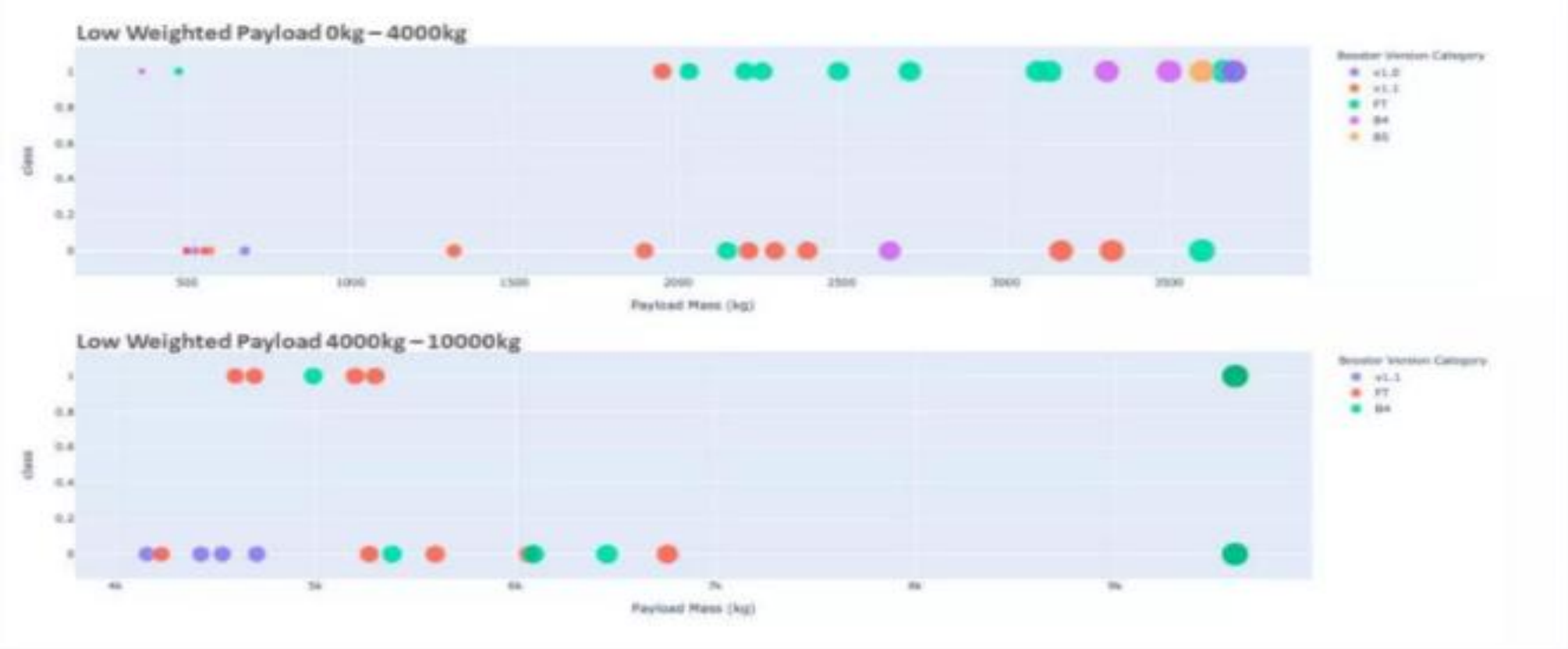
Total Success Launch By All Sites



Launch Sites With Highest Score

- We get 76.9% of successes and 23.1% failure rate

Payload Vs Launch Outcomes



Section 5

Predictive Analysis (Classification)

Classification Accuracy

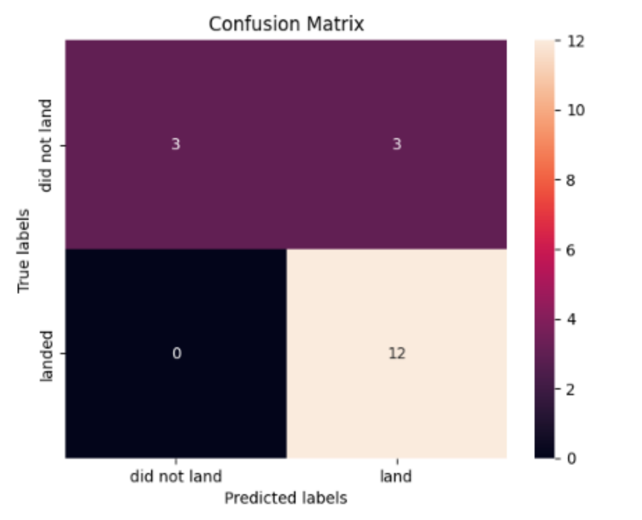
Model with the highest classification accuracy

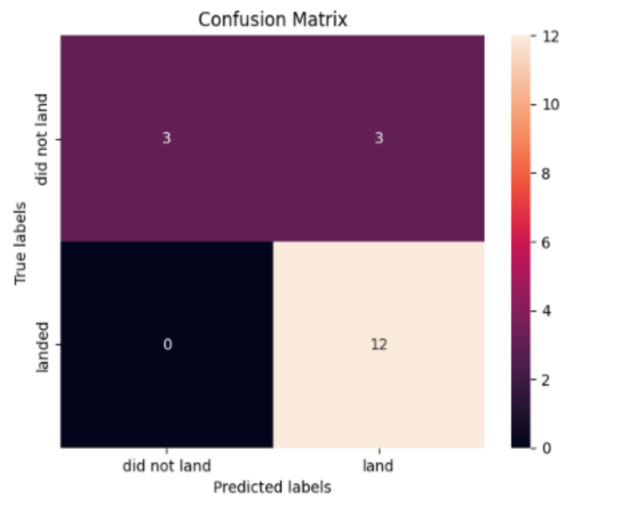
Find the method performs best:

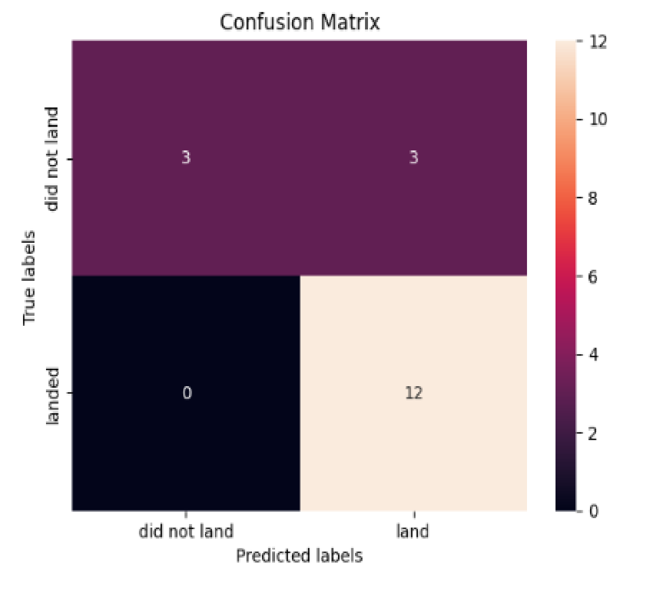
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

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_)
```

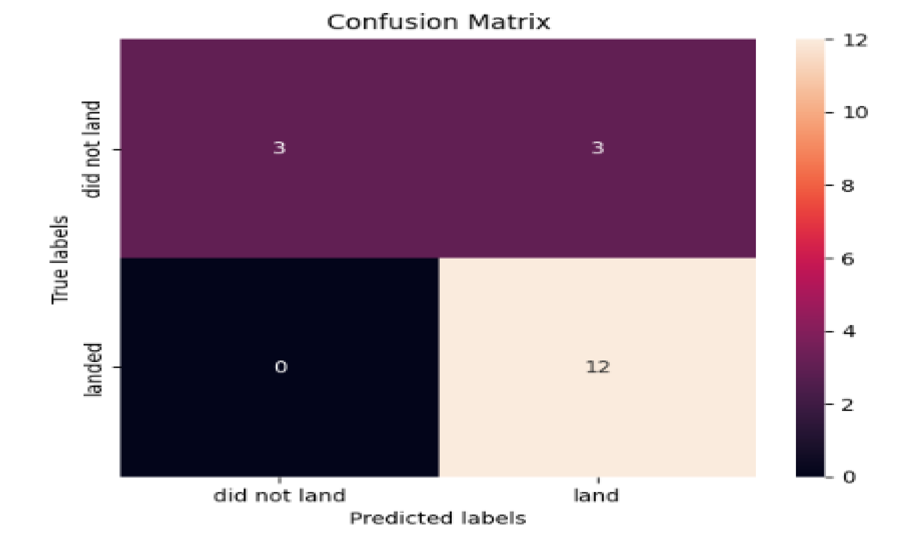
```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix









Conclusions

- A greater number of flights from a launch site is associated with an increased likelihood of success for launches from that site, indicating a positive correlation between flight frequency and success rate.
- The success rate of launches has shown a consistent upward trend from 2013 to 2020, suggesting an improvement in launch outcomes over the years.
- Orbits such as ES-L1, GEO, HEO, SSO, and VLEO have exhibited notably high success rates, indicating that missions to these specific orbits are more likely to succeed.
- KSC LC-39A stands out as the launch site with the highest number of successful launches compared to other sites.
- In terms of machine learning algorithms, the Decision Tree Classifier has been identified as the most effective choice for predicting outcomes in this context.

Thank you!