IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Chua Han Xian
30 Dec 24

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection through API.
  - Data wrangling.
  - EDA with SQLite.
  - EDA with data visualization.
  - Interactive Dashboard with Folium.
  - ML prediction.
- Summary of all results
  - EDA results.
  - Interactive Screenshots
  - Predictive results.

Introduction

- Project background and context

  Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers

  - What factors determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - New feature creation ('Class') and feature encoding for categorical features.
- 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, hyperparameter tuning, evaluate classification models.

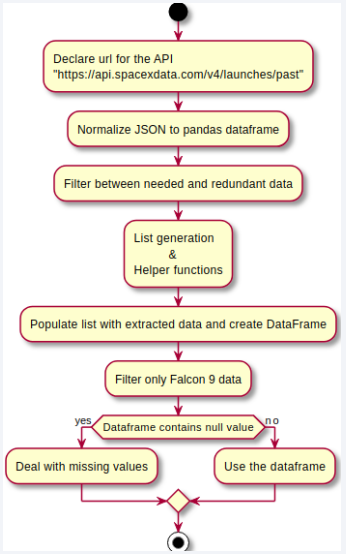
Data Collection

- The data was collected using various methods
  - Data retrieval involves sending GET request to the SpaceX API.
  - The API response was decoded as JSON using .JSON() function, then transformed into pandas dataframe using .json_normalize().
  - Subsequently, the data was cleaned by checking for missing values and addressing where necessary.
  - Additionally, web scraping was conducted using BeautifulSoup to extract Falcon 9 launch records from wikipedia, converting the parsed HTML table into pandas dataframe.

https://github.com/Pipbytes/IBM-Data-Science-Capstone-SpaceX/blob/main/SpaceX%20%20classification%20prediction/01a-spacex-data-collection-api-v2.ipynb

Data Collection – SpaceX API

- A REST API call using a static JSON url.
- Fetching and decoding JSON data into pandas dataframe.
- Status code check and process logic.
- Diplay dataframe.



Task 1: Request and parse the SpaceX launch data using the GET request

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

```
[ ]  1 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

```
[ ]  1 response = requests.get(static_json_url)
     2 print(response.status_code)
     3 print(response.content)
```

```
200
b'[{"fairings": {"reused": false, "recovery_attempt": false, "recovered": false, "ships": []}, "links": {"patch": {"small": "htt
```

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

```
1 if response.status_code == 200:
2     print('Data fetched successfully.')
3     data = response.json()
4     data = pd.json_normalize(data) # Converts nested JSON objects into a flat Pandas DataFrame.
5     print('Data loaded into dataframe.')
6 else:
7     print(f'Error fetching data. Status code {response.status_code}')
```

```
Data fetched successfully.
Data loaded into dataframe.
```

+ Code + Text

Using the dataframe `data` print the first 5 rows

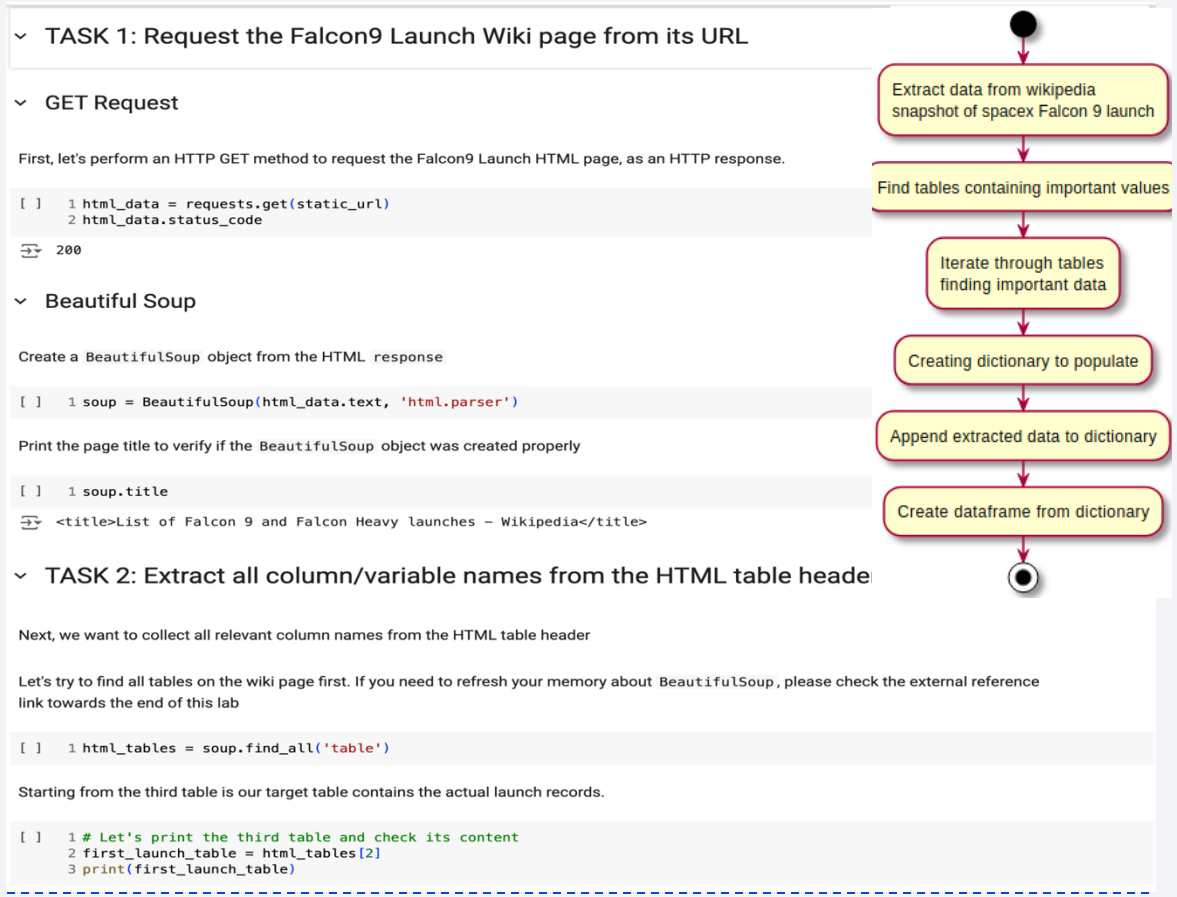
```
[ ]  1 data.head()
```

	static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	crew	ships	capsule
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Engine failure at 33 seconds and loss of vehicle	[]	[]	

Data Collection - Scraping

- Making a GET requrest to retrieve Falcon9 launch wikipidia page.
- Using beautiful soup to parse HTML response.
- Extracting and indentifying tables in HTML content.
- Print and verify target table for data extraction.

https://github.com/Pipbytes/IBM-Data-Science-Capstone-SpaceX/blob/main/SpaceX%20%20classification%20prediction/01b-spacex-data-collection-webscraping.ipynb



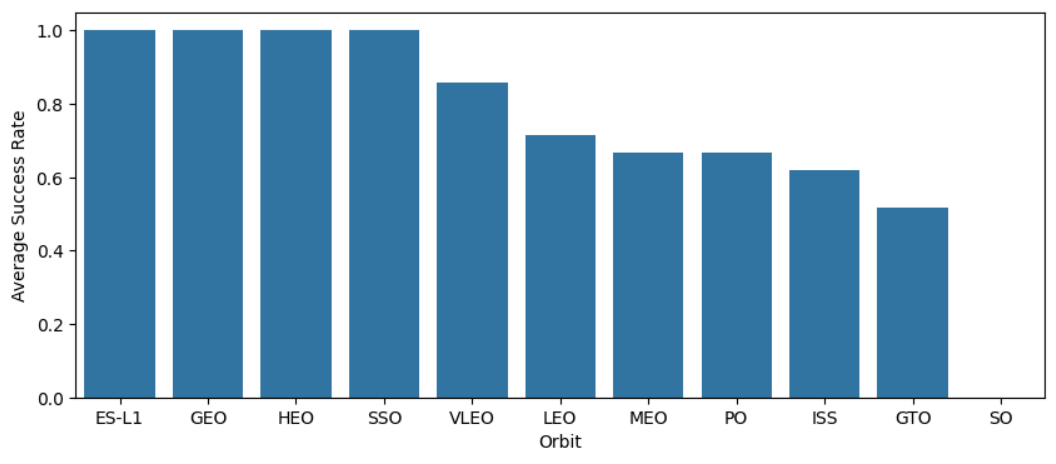
Data Wrangling

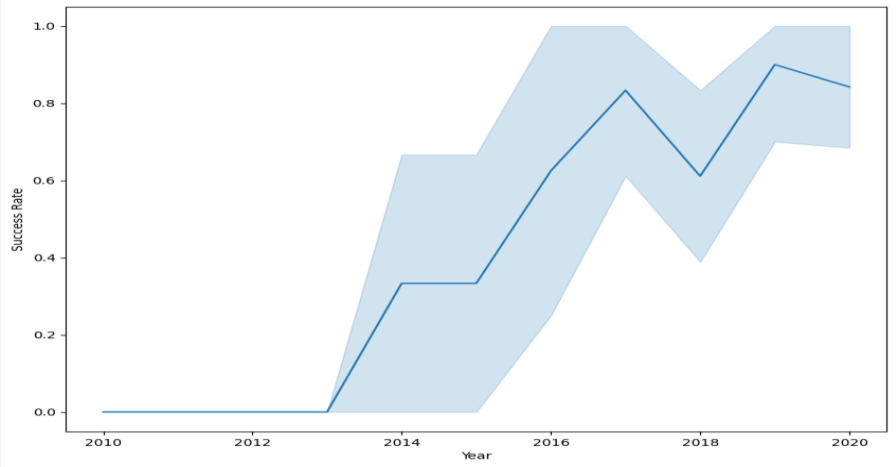
- Identified the percentage of missing values for each features.
- Identified features which are numerical and categorical.
- Calculated the number of launches for each site, and occurrence of each orbit type.
- Create a new landing outcome label from the outcome column.
- Exported the results to csv.

https://github.com/Pipbytes/IBM-Data-Science-Capstone-SpaceX/blob/main/SpaceX%20%20classification%20prediction/02-spacex-Data%20wrangling-v2.ipynb

EDA with Data Visualization

- Explored the data by visualizing the relationship between flight numbers, payload mass, launch site, and orbit type using scatterplots.
- Use bar chart to rank the success rate of each orbit type, and line chart to plot the yearly success trends.





https://github.com/Pipbytes/IBM-Data-Science-Capstone-SpaceX/blob/main/SpaceX%20%20classification%20prediction/03b-spacex-eda-dataviz-v2.ipynb

EDA with SQL

- Loaded SpaceX dataset into SQL lite3 database directly from jupyter notebook environment.
- Use SQL query for EDA, and derived valuable insights such as:
  - Identify the unique launch site names used in space missions.
  - Calculate the total payload mass carried by NASA (CRS) boosters.
  - Determine the average payload mass for booster version F9 v1.1.
  - Count the total number of successful and failed mission outcomes.
  - Listing the failed landing outcomes on drone ships, including the associated booster version and launch site names.

https://github.com/Pipbytes/IBM-Data-Science-Capstone-SpaceX/blob/main/SpaceX%20%20classification%20prediction/03a-spacex-eda-sqllite.ipynb

Build an Interactive Map with Folium

- We identified and marked all launch sites on the map and incorporated elements like markers, circles, and lines to indicate the success or failure of launches at each site using Folium library.
- Launch outcomes were categorized as class 0 for failures and class 1 for successes to enable analysis.
- By enabling color-coded analysis marker clusters, we are able to determine which launch sites exhibited higher success rate.
- We measure the distance from each launch site to nearby features and addressed the questions such as:
  - Are the launch sites located to railways, highways, and coastlines?
  - Do launch sites maintain a specific distance from urban areas?

Build a Dashboard with Plotly Dash

- Created an interactive dashboard using Plotly for dynamic data visualization.
- Displayed pie charts to illustrate the total number of launches at each site.
- Visualized the relationship between launch outcomes and payload mass (kg) across different booster versions using scatterplots.

https://github.com/Pipbytes/IBM-Data-Science-Capstone-SpaceX/blob/main/SpaceX%20%20classification%20prediction/04b-spacex-dashboard-ploty-v2.ipynb

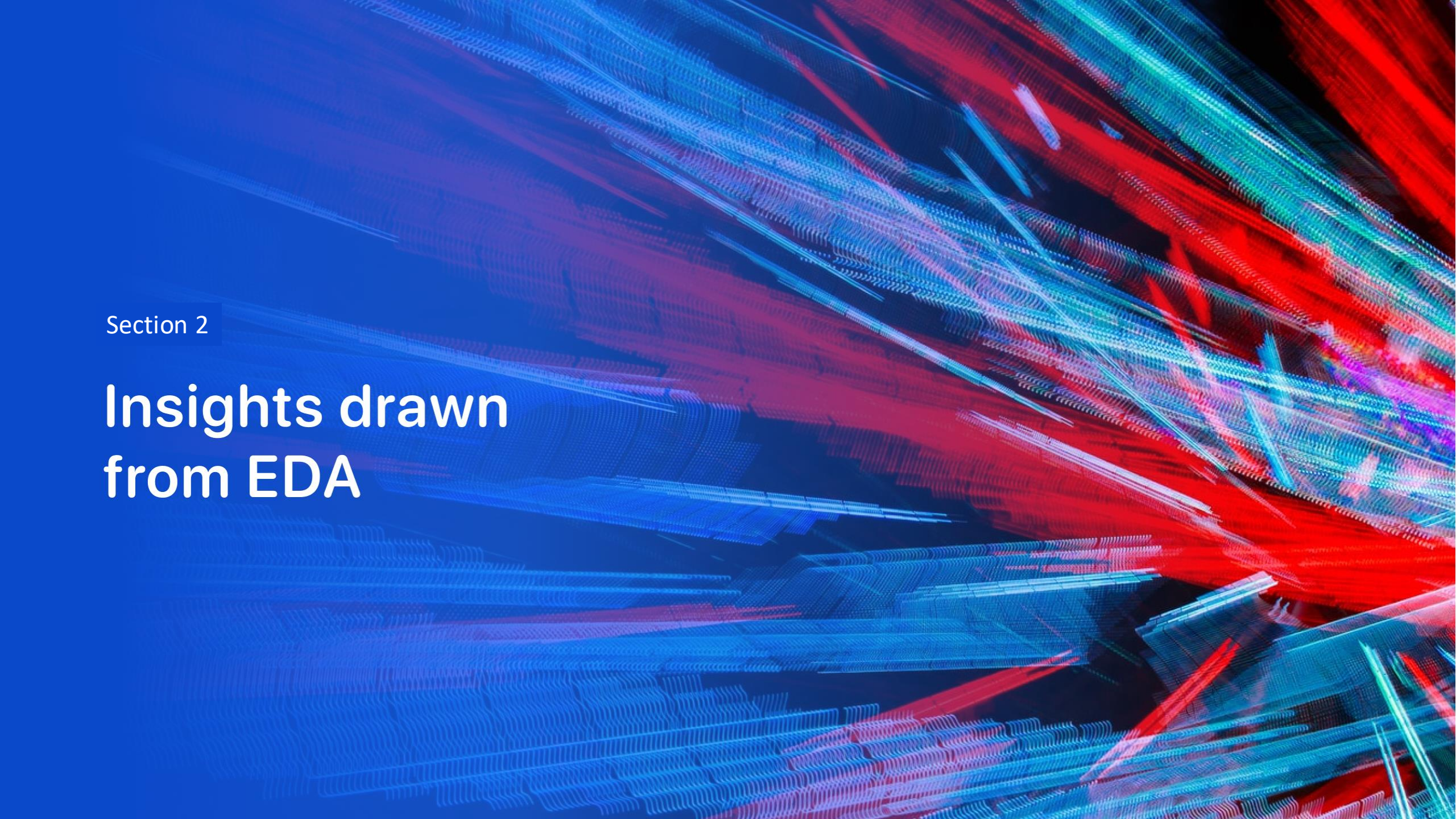
Predictive Analysis (Classification)

- Loaded and transformed data using numpy and pandas, then split into training and testing datasets.
- Developed multiple machine learning models and optimized their performance using GridSearchCV for hyperparameter tuning.
- Evaluated model performance using accuracy as the primary metric and enhanced results through feature engineering and algorithm tuning.
- Identified the best performing classification model.

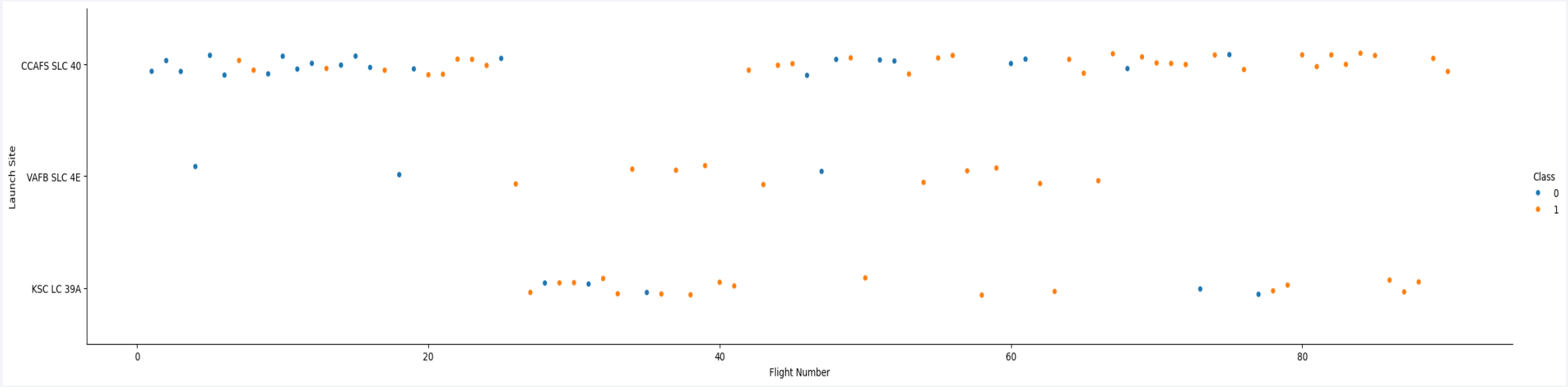
https://github.com/Pipbytes/IBM-Data-Science-Capstone-SpaceX/blob/main/SpaceX%20%20classification%20prediction/05-spacex-Machine-Learning-Prediction-v1.ipynb

Results

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

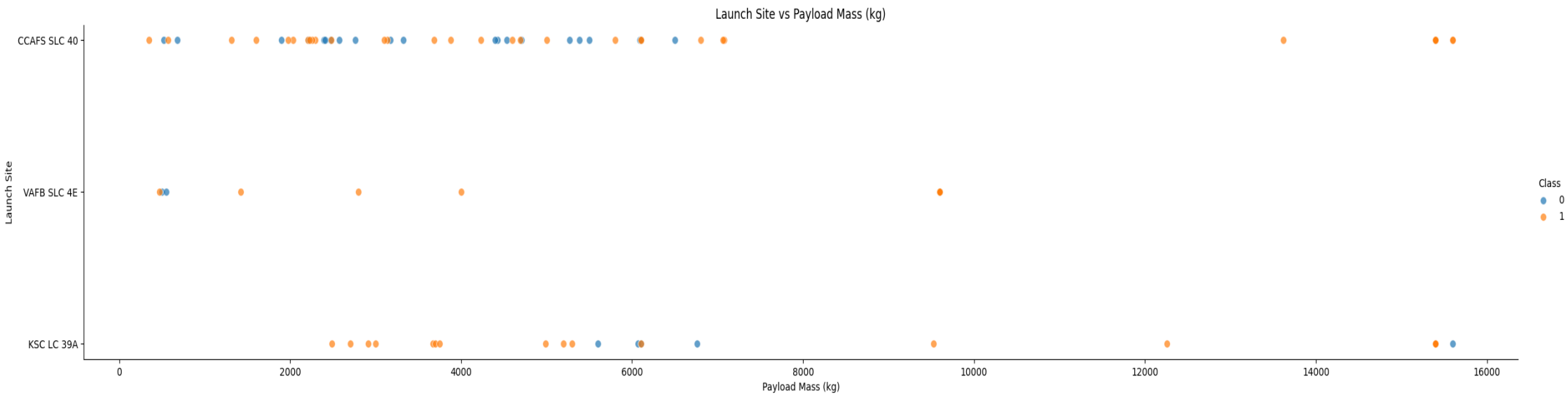

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- The overall success rate appears to improve as flight number increases, indicating learning or operational improvements over time.
- Different launch sites have varying levels of reliability, with CCAFS SLC-40 and KSC LC-39A showing better performance trends compared to VAFB SLC-4E.

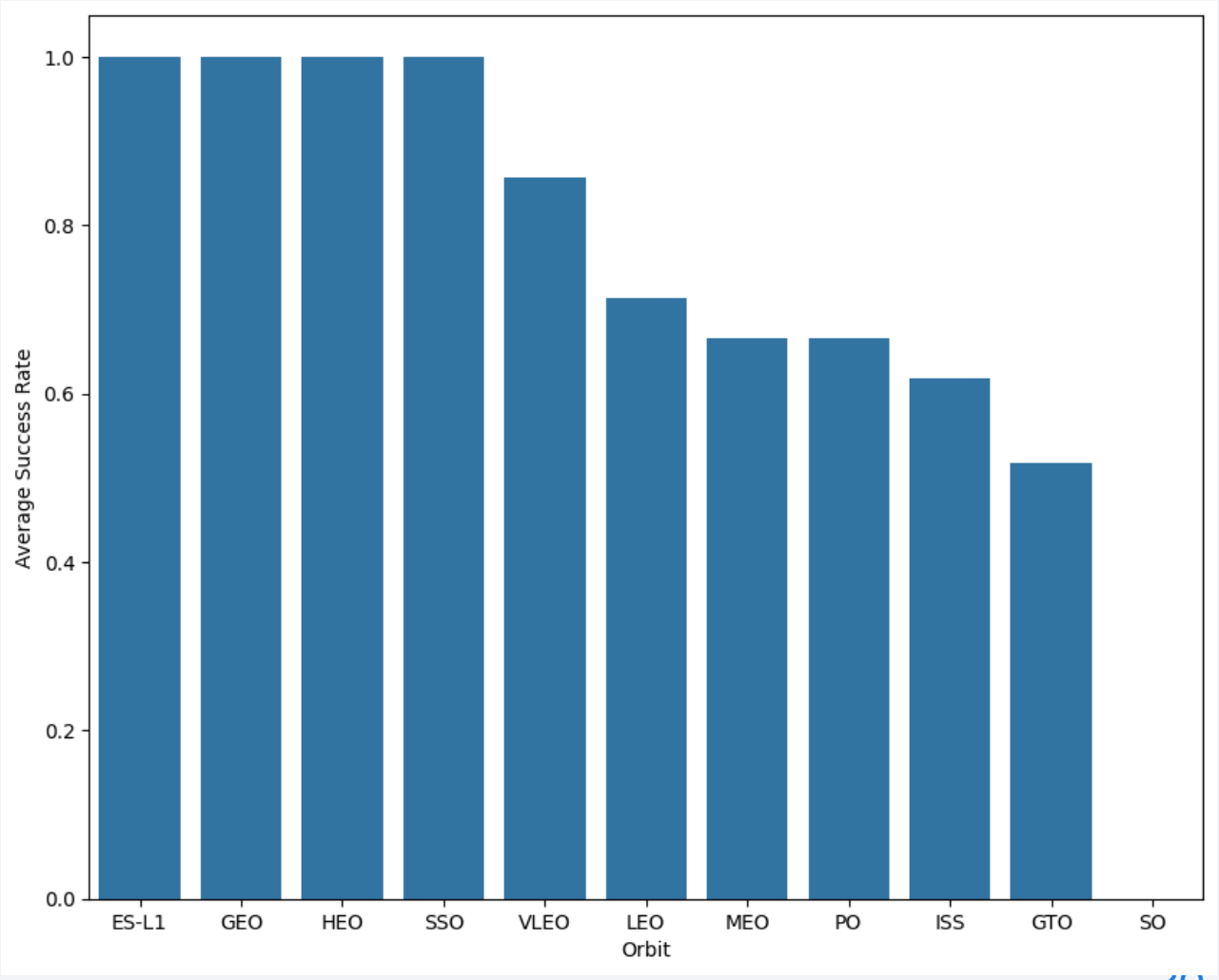
Payload vs. Launch Site



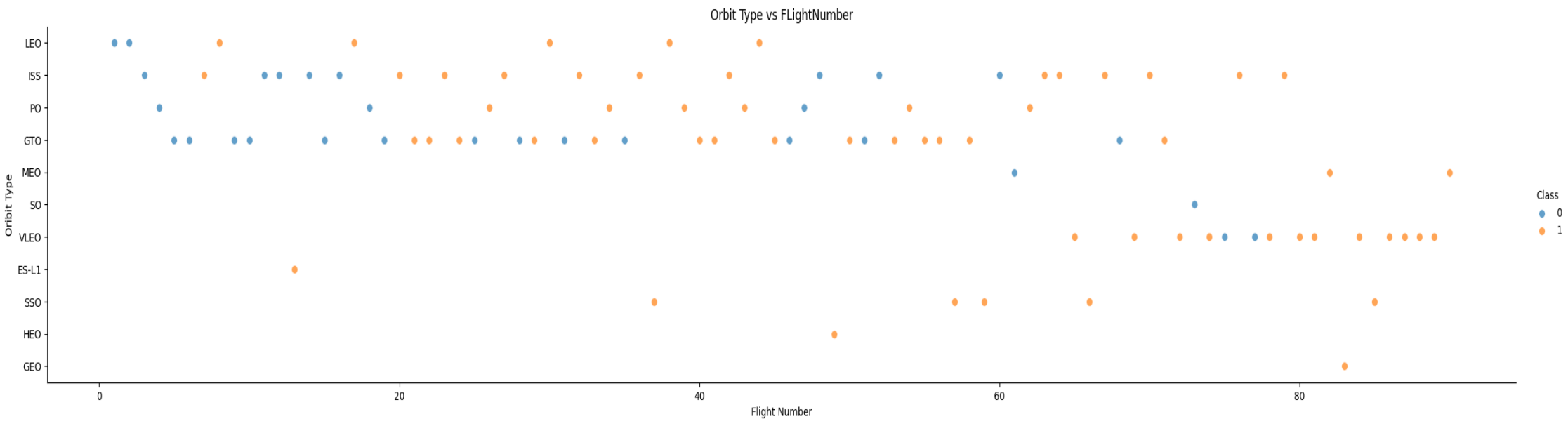
- Higher payloads are handled successfully, especially at CCFAS SLC40, suggesting that the site's infrastructure and technology are optimized for heavier payloads.
- In contrast, there are more failures with lower payload, suggesting handling or site-specific constraints.

Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, and VLEO had the most success rate.

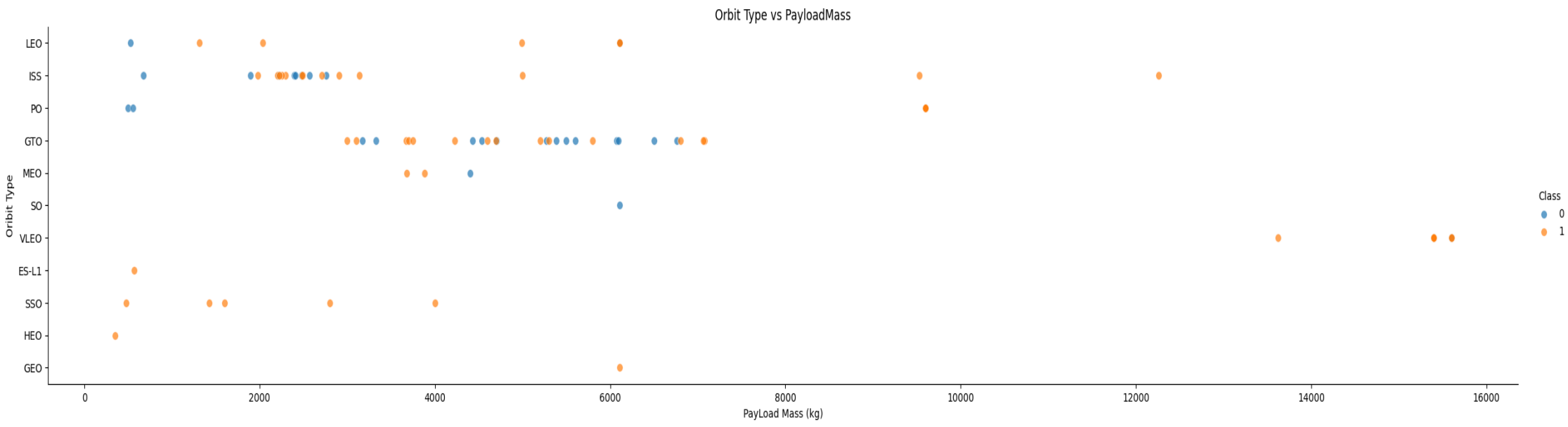


Flight Number vs. Orbit Type



- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

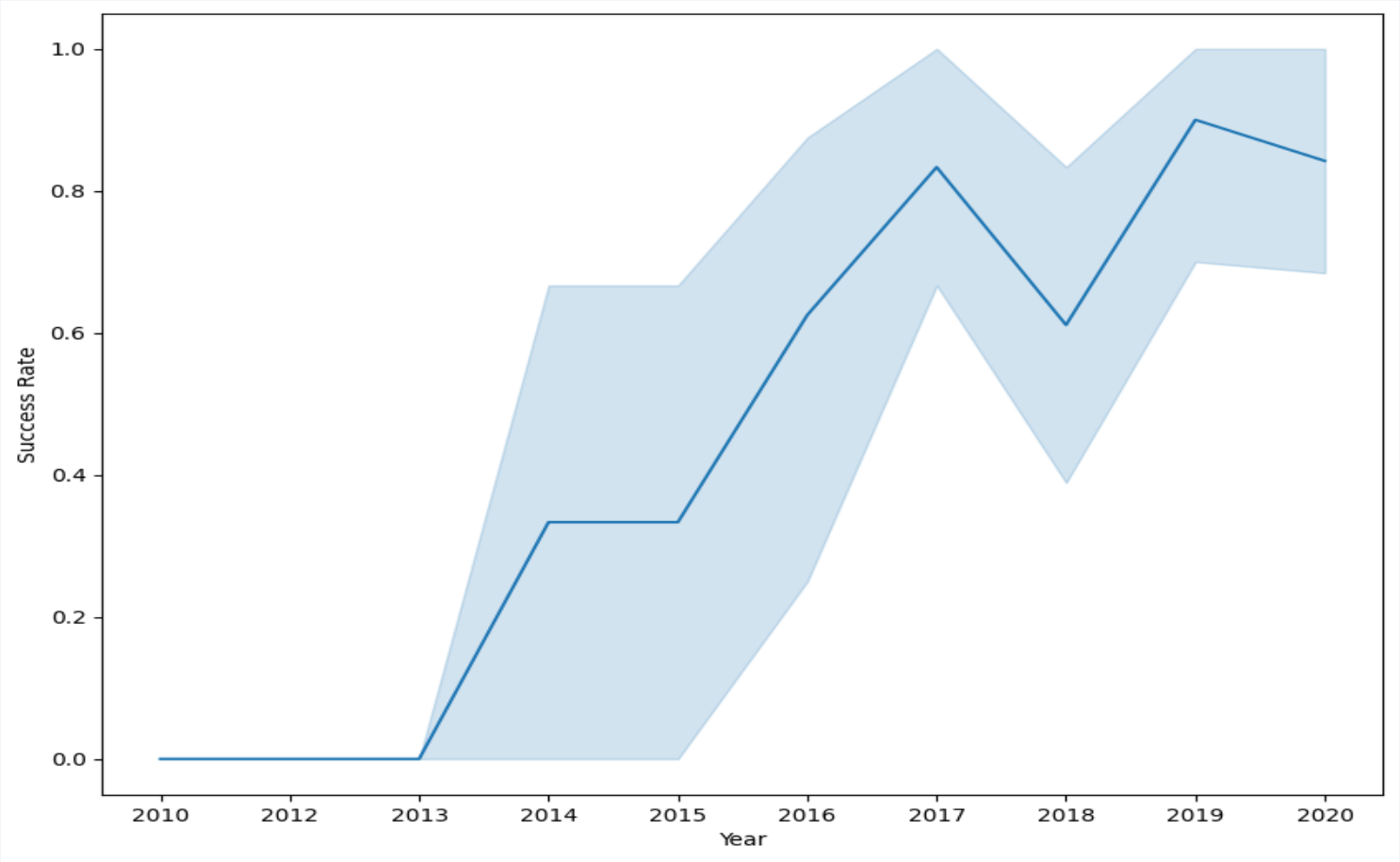
Payload vs. Orbit Type



- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.

Launch Success Yearly Trend

- You can observe that the success rate since 2013 ke increasing till 2017 (stable 2014) and after 2015 it started increasing.



All Launch Site Names

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

```
[ ]   1 %sql SELECT DISTINCT Launch_Site FROM SPACEXTABLE;
```

```
 * sqlite:///my_data1.db
Done.
```

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Launch Site Names Begin with 'CCA'

```
[ ]  1 %sql SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

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

```
[ ]  1 %sql SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)';
```

```
 * sqlite:///my_data1.db
Done.
```

Total_Payload_Mass
45596

Average Payload Mass by F9 v1.1

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

```
[ ]  1 %sql SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1';
```

```
 * sqlite:///my_data1.db
Done.
```

Average_Payload_Mass
2928.4

First Successful Ground Landing Date

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

Hint:Use min function

```
[ ]  1 %sql SELECT MIN(DATE) AS First_Successful_Landing_Date FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)';
```

```
 * sqlite:///my_data1.db
Done.
```

First_Successful_Landing_Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
[ ]  1 %sql SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' \
     2           AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * sqlite:///my_data1.db
Done.
```

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
[ ]  1 %sql SELECT Mission_Outcome, COUNT(*) AS Total_Count FROM SPACEXTABLE GROUP BY Mission_Outcome;
```

```
 * sqlite:///my_data1.db
Done.
```

Mission_Outcome	Total_Count
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

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

```
[ ]  1 %sql SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
```

```
 * sqlite:///my_data1.db
Done.
```

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

2015 Launch Records

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
[ ]  1 %sql SELECT substr(Date, 6, 2) AS Month, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTABLE \
     2 WHERE substr(Date, 0, 5) = '2015' AND Landing_Outcome = 'Failure (drone ship)';
```

```
 * sqlite:///my_data1.db
Done.
```

Month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

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.

```
[ ]  1 %sql SELECT Landing_Outcome, COUNT(*) AS Outcome_Count FROM SPACEXTABLE WHERE Date \
     2   BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY Outcome_Count DESC;
```

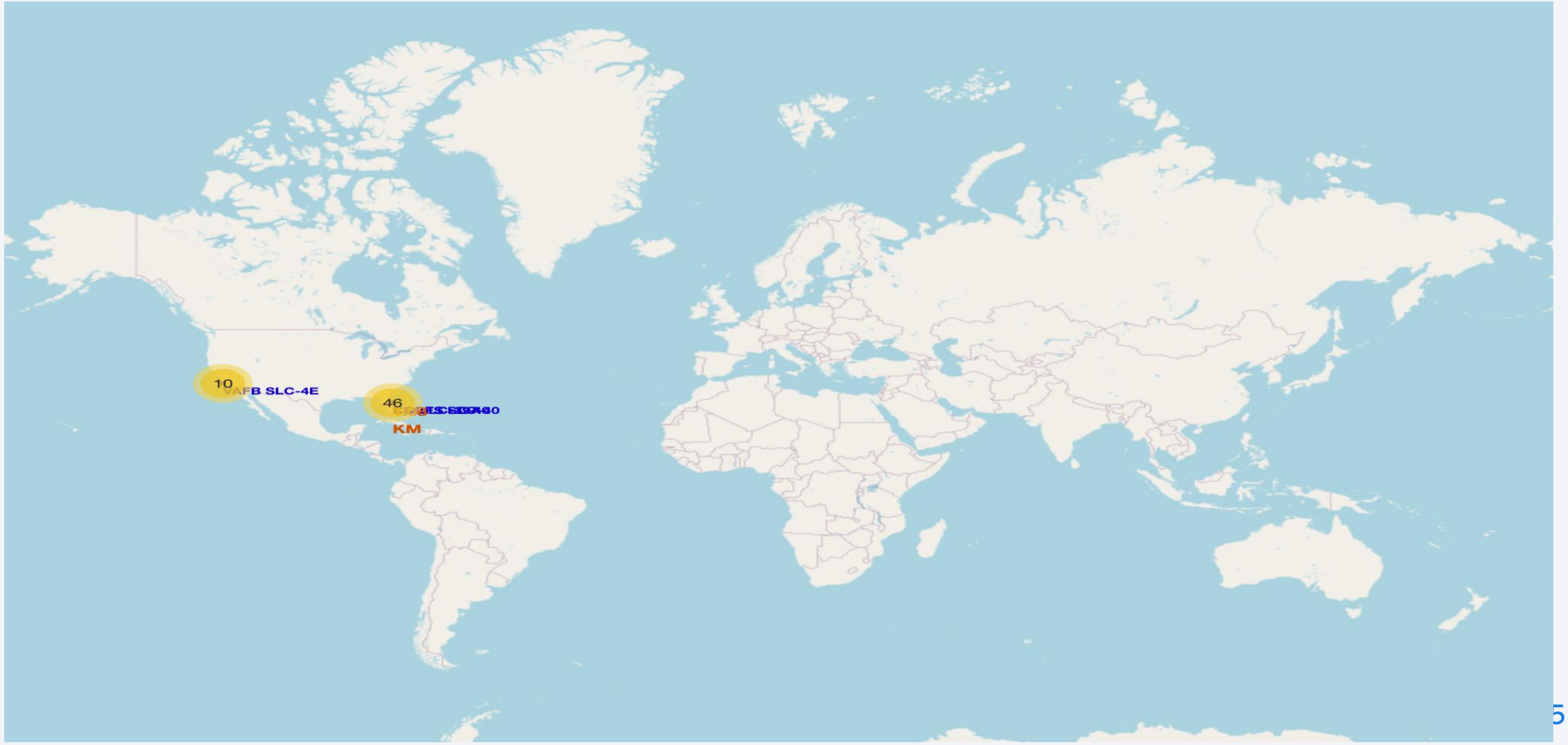
```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome	Outcome_Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

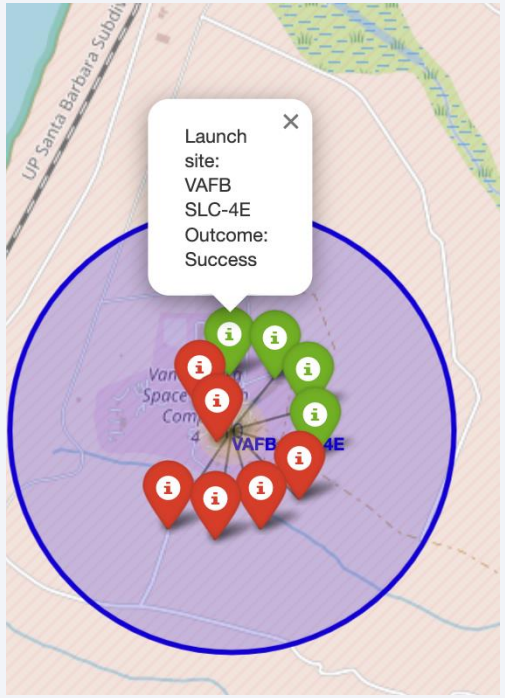
Section 3

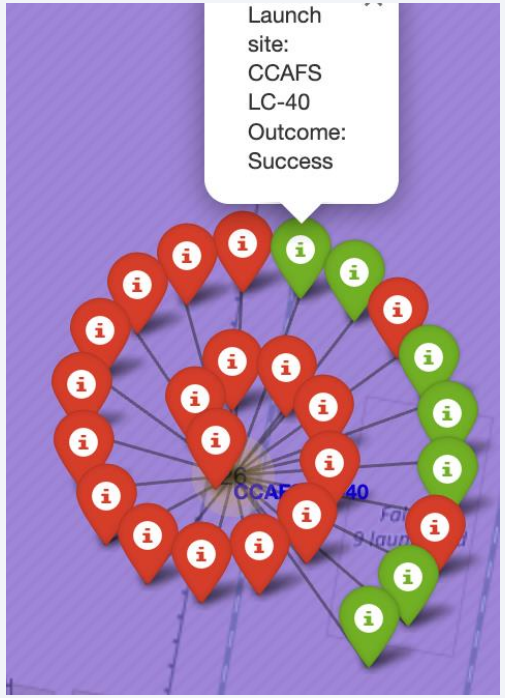
Launch Sites Proximities Analysis

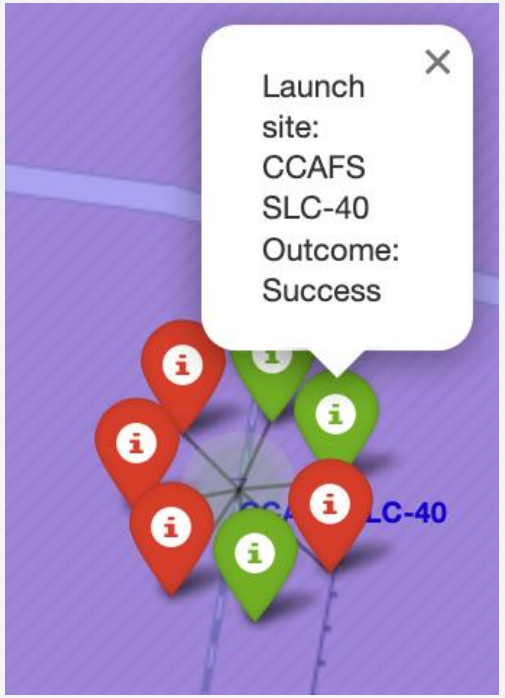
All launch sites global map markers

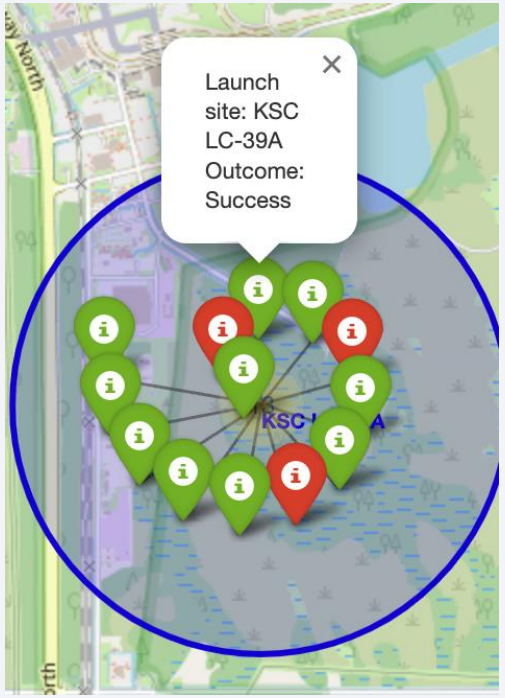


Markers showing launch sites with color labels



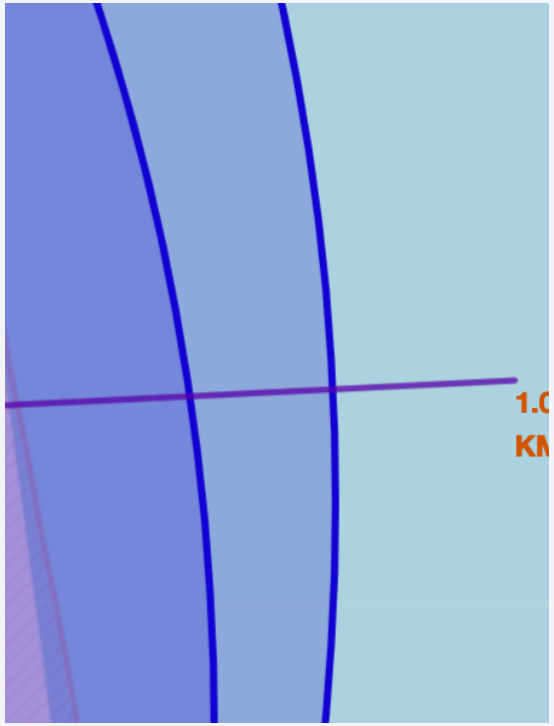


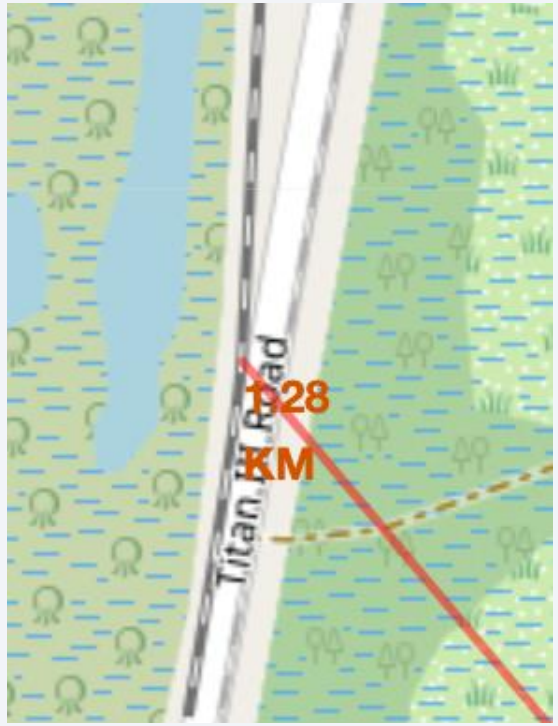




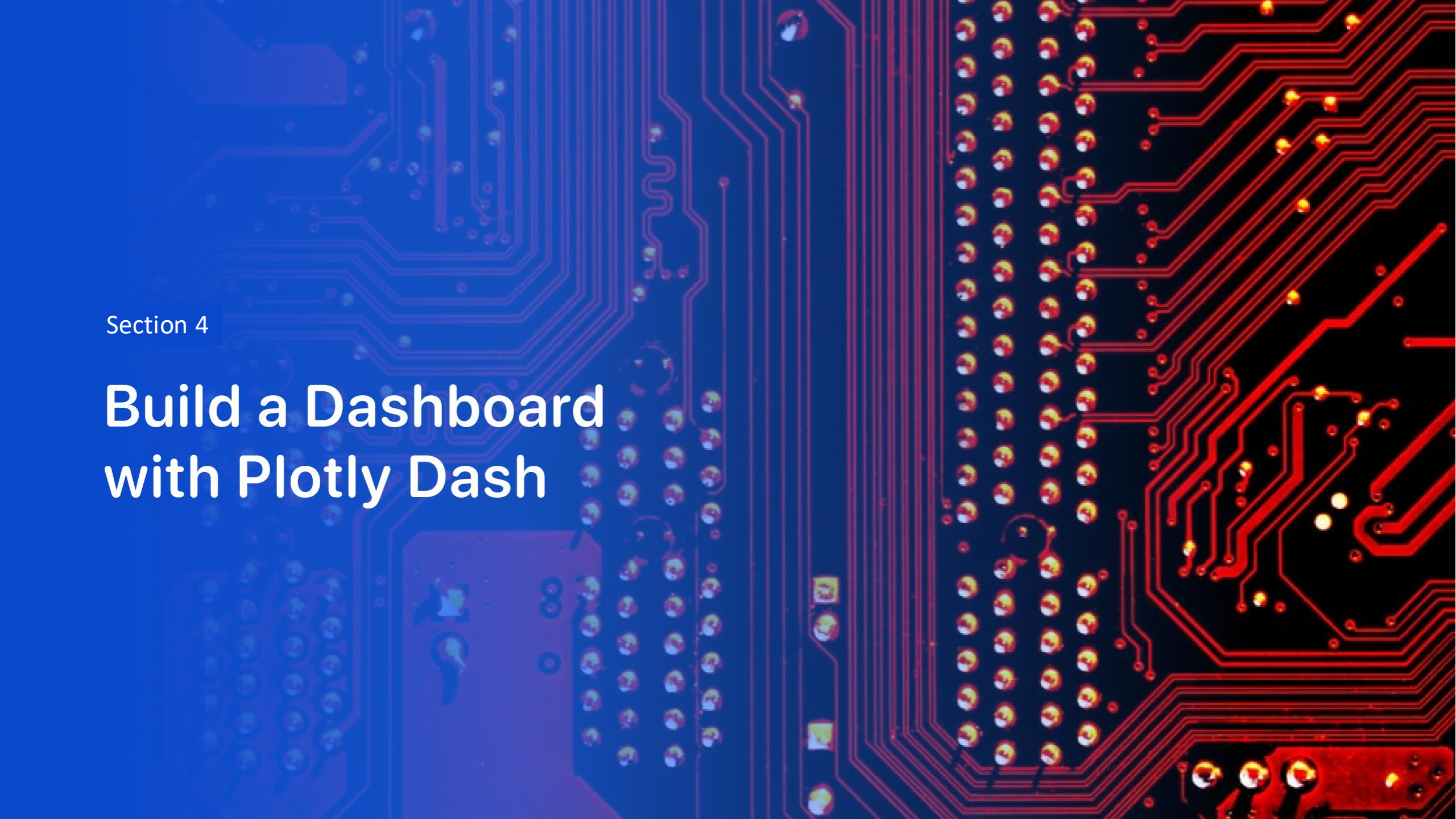
- Green markers indicate successful launches and Red markers show failure.

Markers showing launch sites with color labels

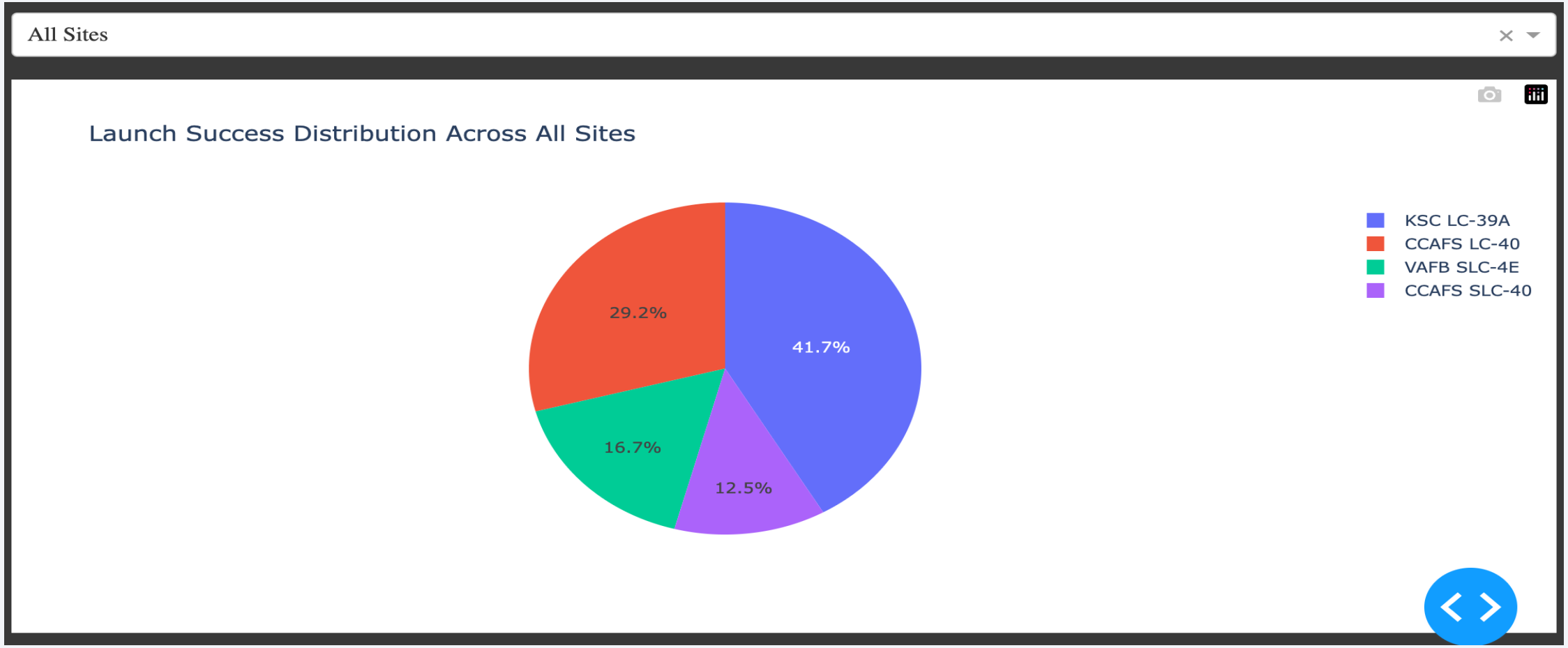




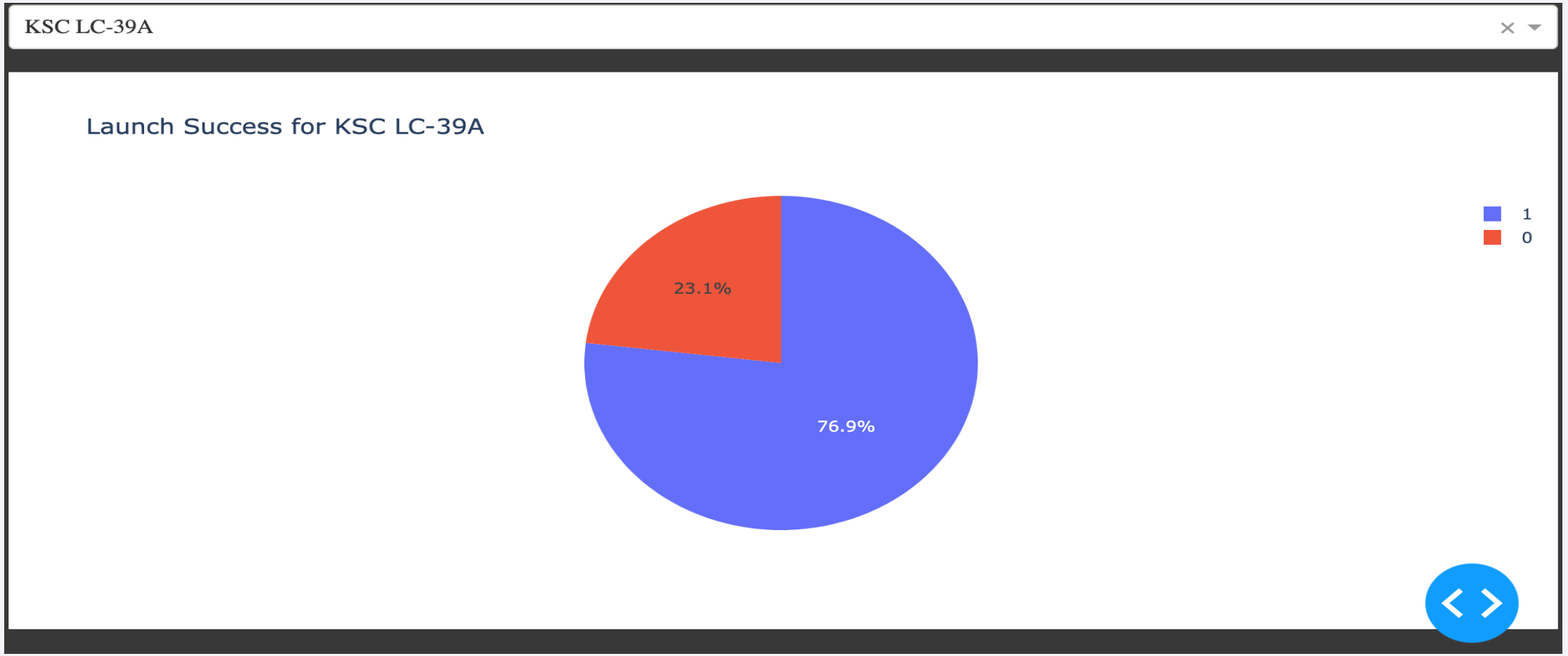
- Distances to Coastline (1.09km), City (19.3km), Railway (1.28km), Highway (0.59km)


Section 4
Build a Dashboard with Plotly Dash

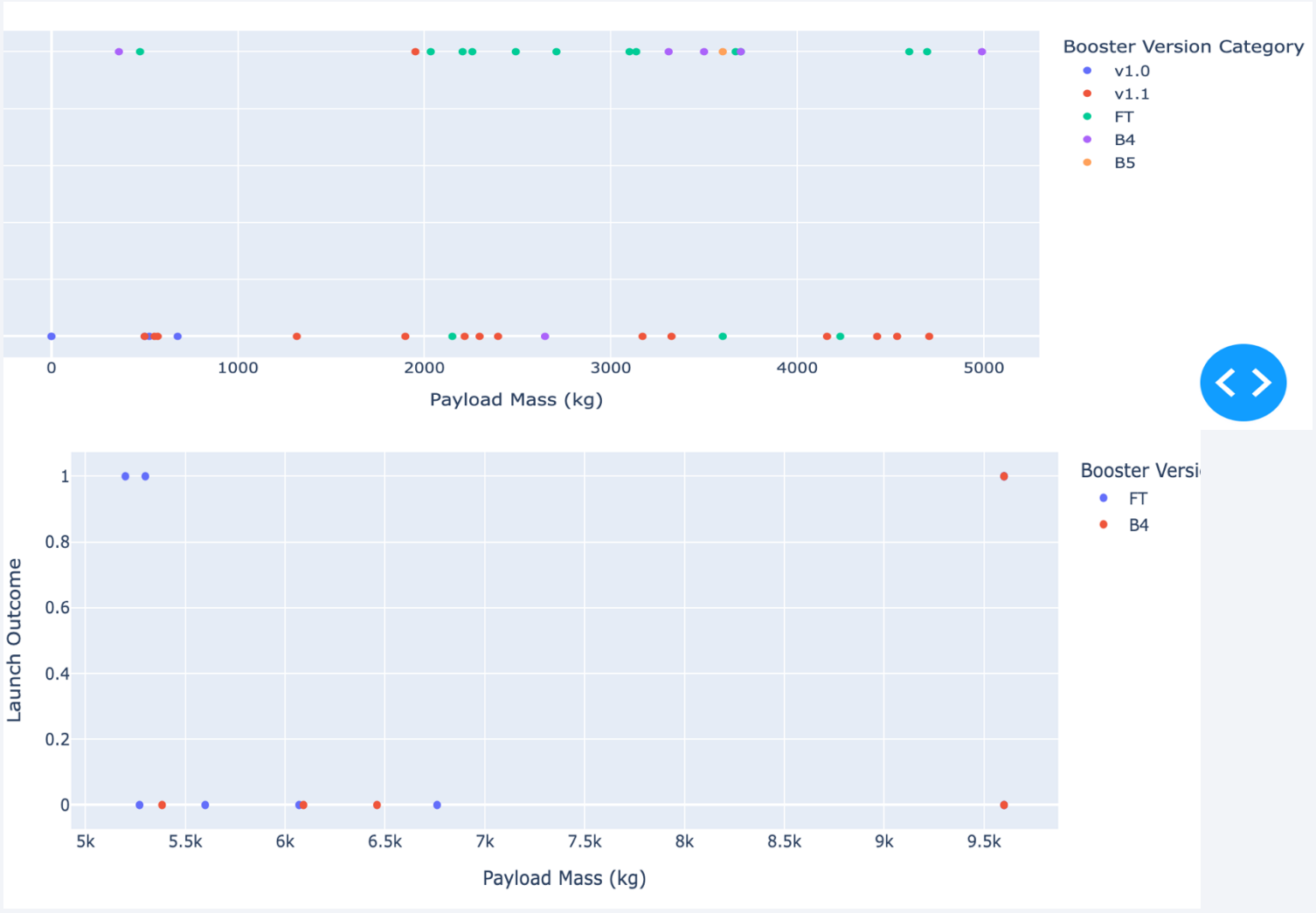
Pie chart of successful launches by each launch sites



Pie chart of launch site with highest successful rate



Pie chart of launch site with highest successful rate



Low weighted payload

(<=4000kg)

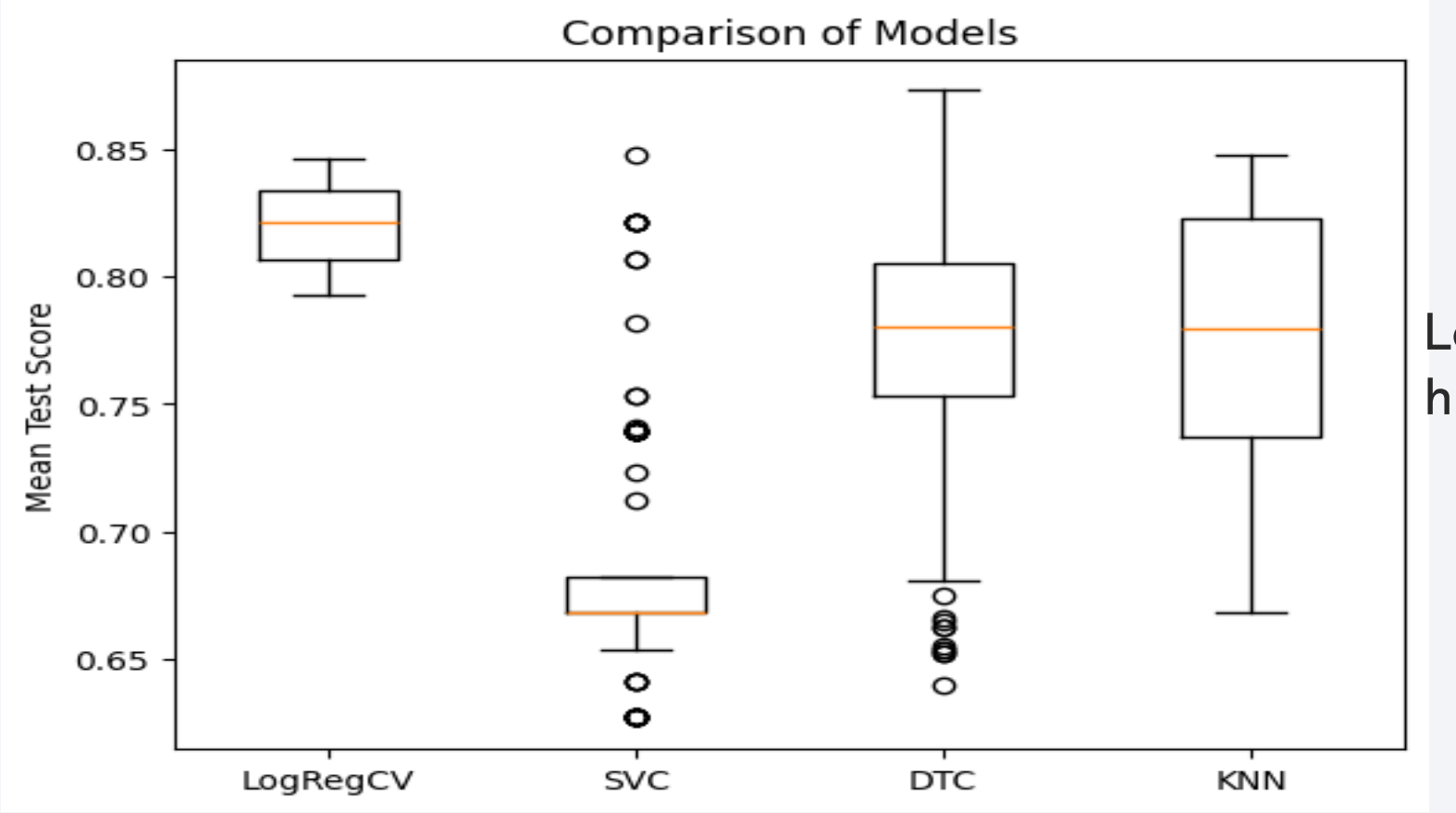
Heavy weighted payload

4000 < x <= 10000kg

Section 5

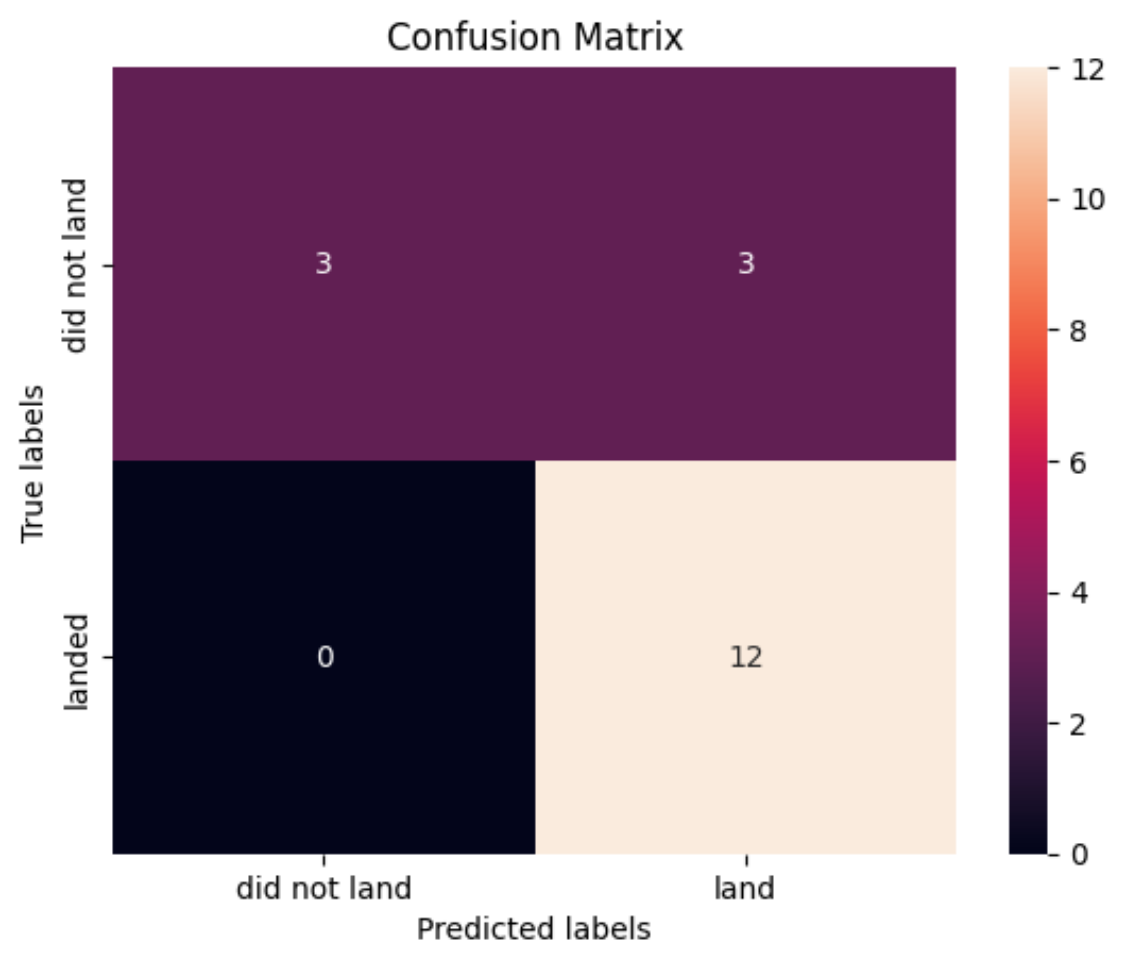
Predictive Analysis (Classification)

Classification Accuracy



Logistic Regression has the highest mean test score.

Confusion Matrix



The confusion matrix shows that the logistic regression model correctly predicted 12 "landed" outcomes and 3 "did not land" outcomes, while misclassifying 3 "did not land" cases as "landed." The overall performance indicates strong accuracy, but there is room for improvement in predicting non-landings.

Conclusions

- We can conclude that:
  - Launch success rate started to increase in 2013 till 2020.
  - Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
  - KSC LC-39A had the most successful launches of any sites.
  - The Decision tree classifier is the best machine learning algorithm for this task.
  - The larger the flight amount at a launch site, the greater the success rate at a launch site.

Thank you!