IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Anh Bui Ngoc Tuan
31 Oct 2025

Outline

Navigate through our complete analytical journey from data collection to actionable insights.

01

Executive Summary

Key findings and methodology overview

02

Introduction

Project background and research questions

03

Methodology

Data collection and analytical approaches

04

Results

Key findings and visual analytics

05

Conclusion

Actionable recommendations and insights

06

Appendix

Technical documentation and references

Executive Summary

This analysis synthesizes multiple data collection methodologies and presents comprehensive results across all analytical dimensions.

Methodological Overview

- Multi-source data collection from SpaceX APIs
- Advanced web scraping techniques
- Rigorous data cleaning and transformation
- Interactive visual analytics implementation

Key Results

- Complete dataset integration across sources
- Processed and validated records
- Predictive classification models deployed
- Actionable business intelligence generated

Introduction

Rockets are similar to cars in the domain of space travel, but they have one unique quality: they can be reused! SpaceX has devised a method for returning the Falcon 9 rocket's first stage to Earth after it has completed its mission. This is crucial because reusing the rocket results in significant financial savings, lowering the cost of space travel.

Introduction

Research Questions

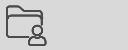
- What insights emerge from multi-source data integration?
- How can we identify meaningful patterns through exploratory analysis?
- Which classification models provide the best outcome when predicted?
- Which factors influence the successful landing of the Falcon 9 first stage?
- How the insight will help determine the cost of a launch?

Section 1

Methodology

Methodology

My comprehensive approach integrates multiple analytical techniques to extract maximum value from diverse data sources.



Data Collection

Multi-source API and web extraction

Data Wrangling

Processing and cleaning operations

SQL, EDA & Visualization

Pattern discovery and insights

Interactive Analytics

Folium and Plotly Dash deployment

Predictive Modeling

Evaluate Classification models and tuning

Data Collection

Multiple integrated approaches ensure comprehensive data acquisition while maintaining quality standards and methodological consistency.

Collection Methodologies

API Integration

Systematic REST API calls retrieve structured data from authoritative sources with consistent formatting and validation.

Web Scraping

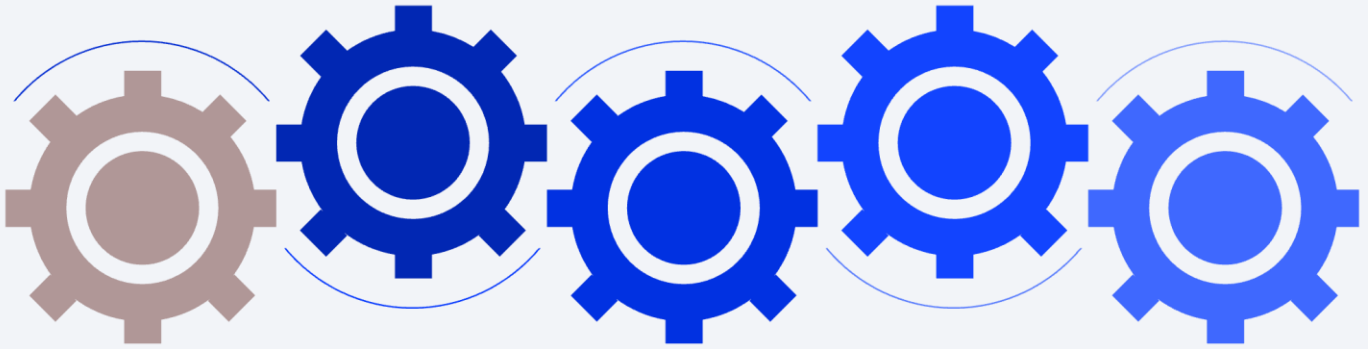
Targeted scraping techniques extract unstructured data from web pages using automated parsing and data extraction.

Data Validation

Quality checks and verification protocols ensure data integrity and consistency across all collection sources.

Data Collection – SpaceX API

REST-based API calls systematically retrieve comprehensive SpaceX operational data for advanced analytics, with a focus on detailed implementation steps.



Define Helper Functions

Create reusable code blocks for common tasks.

Make GET Request

Fetch data from the target API endpoint.

Parse JSON Response

Convert the raw JSON string into a usable object.

Extract Launch Data

Isolate the specific launch information from the response.

Convert to DataFrame

Structure the data into a tabular format for analysis.

Data Collection – SpaceX API

Main Implementation for Notebook Flowchart

1 Helper Functions & Initial Requests

Helper functions were defined to facilitate API usage, extracting launch data via identification numbers and making initial GET requests to the SpaceX API URL.

2 Executing Rocket Launch Data Requests

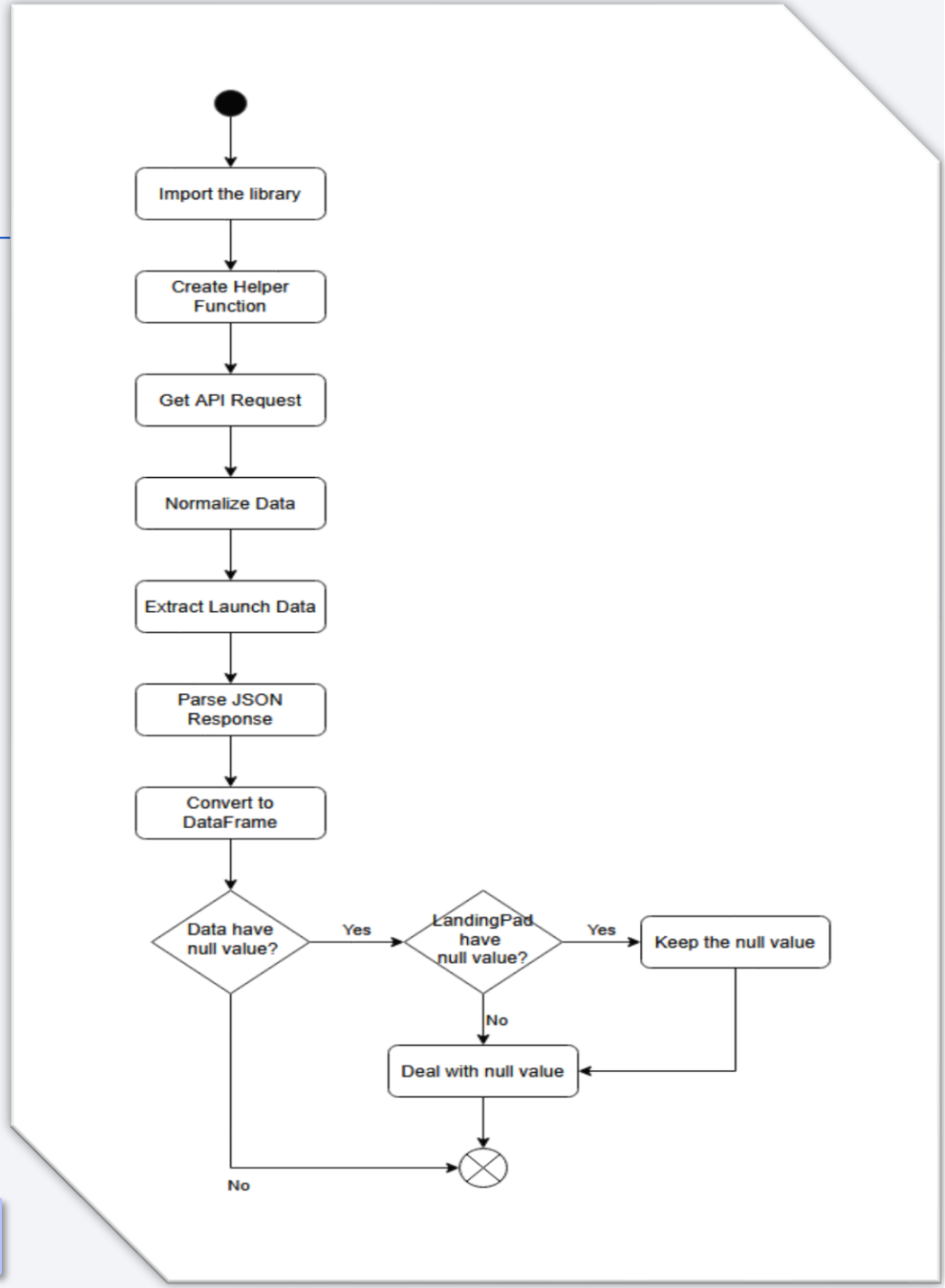
Rocket launch data was systematically retrieved from the SpaceX API URL by executing GET requests to obtain the comprehensive dataset.

3 JSON Parsing and DataFrame Conversion

The JSON responses from the GET requests were decoded and parsed, then converted into a consistent Pandas DataFrame for analytical processing and uniformity.

4 Dealing with Missing Value

Check the data for missing values and suggest ways to handle them. 'LandingPad' can accept null data.



GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/1%20-%20Collecting%20the%20data.ipynb*

Data Collection – SpaceX API

Data was collected using a Get request to SpaceX API and decoded the return content as a JSON, which was then turned into a Pandas data frame.

Now let's start requesting rocket launch data from SpaceX API with the following URL:

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

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

Check the content of the response

In [8]:
```
print(response.content)
```

```
b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://imag
                                                                                    o.png","large":"https://images2.imgbox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"la
                                                                                    ":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.
```

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:

In [9]:
```
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

In [10]:
```
response=requests.get(static_json_url)
```

In [11]:
```
response.status_code
```

Out[11]: 200

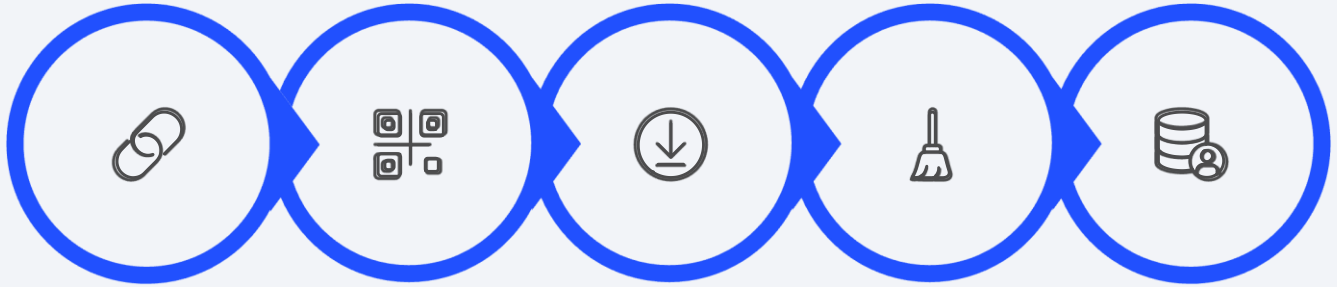
Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

In [13]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data=pd.json_normalize(response.json())
```


GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/1%20-%20Collecting%20the%20data.ipynb*

Data Collection - Scraping

Utilizing automated web scraping techniques, Falcon 9 historical launch records were systematically extracted from Wikipedia's 'List of Falcon 9 and Falcon Heavy launches' page, transforming unstructured web data into structured analytical datasets.



Target URL Identification
Identified the Wikipedia page 'List of Falcon 9 and Falcon Heavy launches' as the data source for Falcon 9 historical launch records.

HTML Parsing
Utilized BeautifulSoup and the requests library to retrieve and parse the HTML content, specifically focusing on the table containing Falcon 9 launch records.

Data Extraction
Extracted relevant data from the parsed HTML table, focusing on detailed Falcon 9 launch attributes.

Cleaning & Conversion
Cleaned the extracted data and converted it into a structured Pandas DataFrame for consistent analytical processing.

Storage
Stored the processed and structured Falcon 9 launch data, ready for further analysis and reporting.

Data Collection - Scraping

Main Implementation for Notebook Flowchart

1 Helper Functions & Initial Requests via BeautifulSoup

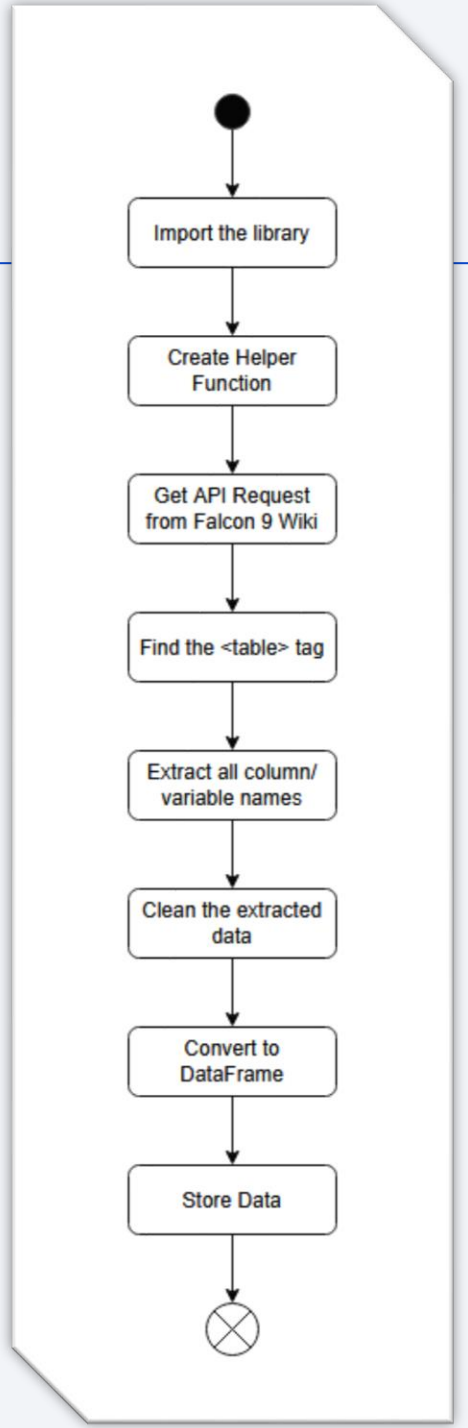
Helper functions were defined for extracting the data. Use BeautifulSoup for extracting launch data and making initial GET requests to the SpaceX Wiki URL.

2 Find the <table> tag and Extract the data

Check the <table> tag content for the Falcon 9 records. Create a dictionary for extracting the data.

3 Clean the data and DataFrame Conversion

The responses from the requests were decoded and parsed, then added to the dictionary. After some cleaning process then covert into a Pandas DataFrame for analytical processing.



GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/2%20-%20Web%20Scraping.ipynb*

Data Collection - Scraping

TASK 1: Request the Falcon9 Launch Wiki page from its URL

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

In [5]:
```
# use requests.get() method with the provided static_url and headers
# assign the response to a object
response = requests.get(static_url, headers=headers)
```

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

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content

soup = BeautifulSoup(response.content, 'html.parser')
```

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

In [8]:
```
# Use soup.title attribute

soup.title
```

Out[8]: `<title>List of Falcon 9 and Falco`

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 [10]:
```
# 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 [11]:
```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Version,<
br/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0"><a href="#cite_note-booster-11"><span class="cite-bracket">
[</span>b<span class="cite-bracket">]</span></a></sup>
</th>
```

After you have fill in the parsed launch record values into `launch_dict`, you can create a dataframe from it.

In [28]:
```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
df
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages/ipykernel_launcher.py:1: DeprecationWarning: The default dtype
for empty Series will be 'object' instead of 'float64' in a future version. Specify a dtype explicitly to silence this warnin
g.
  """Entry point for launching an IPython kernel.
```

Out[28]:

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.07B0003.18	Failure	4 June 2010	NaN
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.07B0004.18	Failure	8 December 2010	NaN
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.07B0005.18	No attempt\n	22 May 2012	NaN
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.07B0006.18	No attempt	8 October 2012	NaN
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.07B0007.18	No attempt\n	1 March 2013	NaN

Using BeautifulSoup and a request, web scraping was used to gather historical Falcon 9 launch records from a Wikipedia page. The launch HTML was then parsed to produce a data frame.

GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/2%20-%20Web%20Scraping.ipynb*

Data Wrangling

Comprehensive data processing converts raw information into clean, validated datasets ready for advanced analytical modeling.

Data Cleaning

Filtered Falcon 9 launches by 'BoosterVersion'. Managed missing values in 'LandingPad' and imputed 'PayloadMass' with the column's mean. Also, removed duplicates and standardized formats.

Transformation

Identify data patterns and define labels for supervised models. Applied feature engineering, normalization, and aggregation operations.

Validation

Quality assurance, consistency checks, statistical verification

Process Output: Analysis-ready datasets with validated structure, complete metadata documentation, and comprehensive audit trails for reproducibility and peer review.

Data Wrangling

Main Implementation for Notebook Flowchart

1 Import Library & Analysis

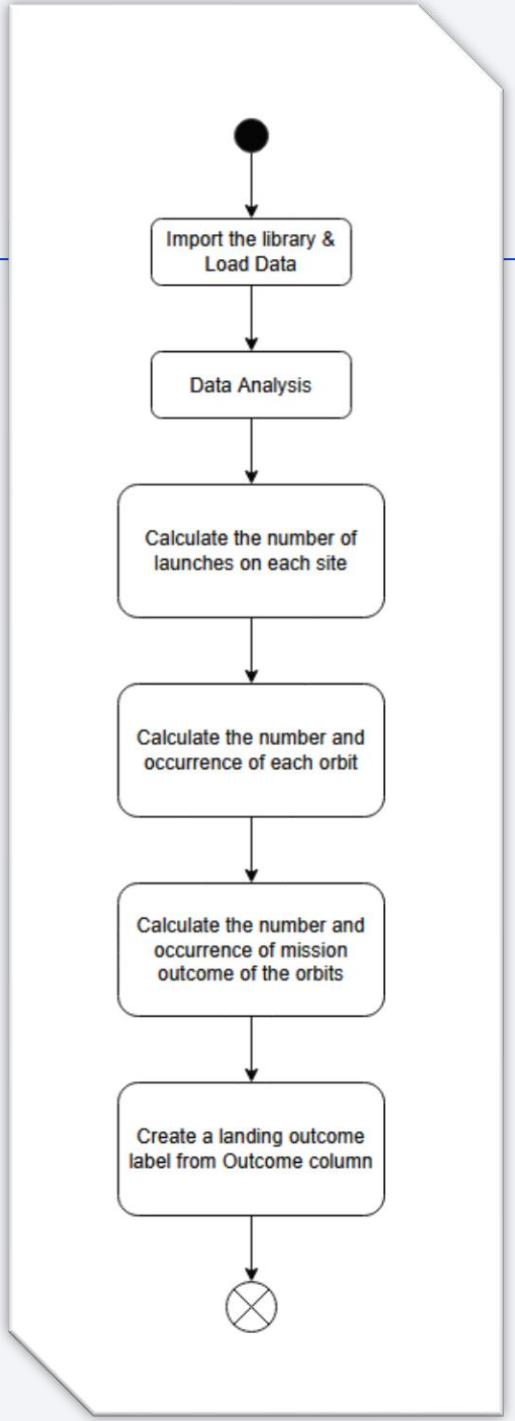
Import the library, load the data and identify which columns are numerical and categorical

2 Calculalate some statistical value

Calculate statiscal value for lauches, orbit and mission outcome

3 Create label for the Outcome

Compute and generate new labels to correspond to the 1 and 0 values for later use in the classification algorithm.



GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/3%20-%20Data%20wrangling.ipynb*

Data Wrangling

Calculate some stastical and Create landing outcome label

TASK 2: Calculate the number and occurrence of each orbit

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

Note: Do not count GTO, as it is a transfer orbit and not itself geostationary.

In [8]:
```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

Out[8]:
```
Orbit
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
HEO       1
ES-L1     1
SO        1
GEO       1
Name: count, dtype: int64
```

TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

In [28]:
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise

landing_class=[]
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)

print(landing_class.count(1))
```

```
60
```

TASK 3: Calculate the number and occurence of mission outcome of the orbits

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes`.Then assign it to a variable landing_outcomes.

In [10]:
```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

Out[10]:
```
Outcome
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: count, dtype: int64
```

`True Ocean` means the mission outcome was successfully landed to a specific region of the ocean while `False Ocean` means the mission outcome was unsuccessfully landed to a specific region of the ocean. `True RTLS` means the mission outcome was successfully landed to a ground pad `False RTLS` means the mission outcome was unsuccessfully landed to a ground pad. `True ASDS` means the mission outcome was successfully landed to a drone ship `False ASDS` means the mission outcome was unsuccessfully landed to a drone ship. `None ASDS` and `None None` these represent a failure to land.

GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/3%20-%20Data%20wrangling.ipynb*

EDA with Data Visualization

Strip plot

Purpose: Explore relationships between a categorical variable on one axis and a numerical variable on the other, such as Orbit vs. FlightNumber.

Why: Identifies distribution of individual data points, visualizing comparison of distributions across categories.

Bar Charts

Purpose: Compare categorical variables across various categories, such as rocket types or launch sites, such as launch outcomes (success or failure).

Why: Highlights patterns or trends by clearly comparing frequencies or proportions within categorical data.

Line Charts

Purpose: Keep track of changes over time, such how many Falcon 9 launches went well in different years.

Why: Shows patterns over time and helps to understand how performance has changed or stayed the same throughout certain times.

Scatter Plots

Purpose: Explore relationships between two numerical variables, such as PayloadMass vs. Launch outcome.

Why: Identifies correlations or dependencies between variables and shows how one variable varies in respect to another.

Heatmaps

Purpose: Visualize correlation matrices between multiple numerical variables.

Why: Identifies high correlations (positive or negative) between variables, aids in feature selection, and comprehending multicollinearity.

GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/5%20-%20EDA%20with%20Visualization.ipynb*

EDA with SQL

Filtering Queries

- Display 5 records where launch sites begin with 'CCA'
- List boosters with success in drone ship and payload mass between 4000-6000 kg
- Display records for failure landing outcomes in drone ship for 2015

Aggregate Queries

- Display total payload mass carried by NASA (CRS) boosters
- Display average payload mass carried by booster version F9 v1.1
- List total number of successful and failure mission outcomes
- Rank count of landing outcomes between specific dates

Subqueries

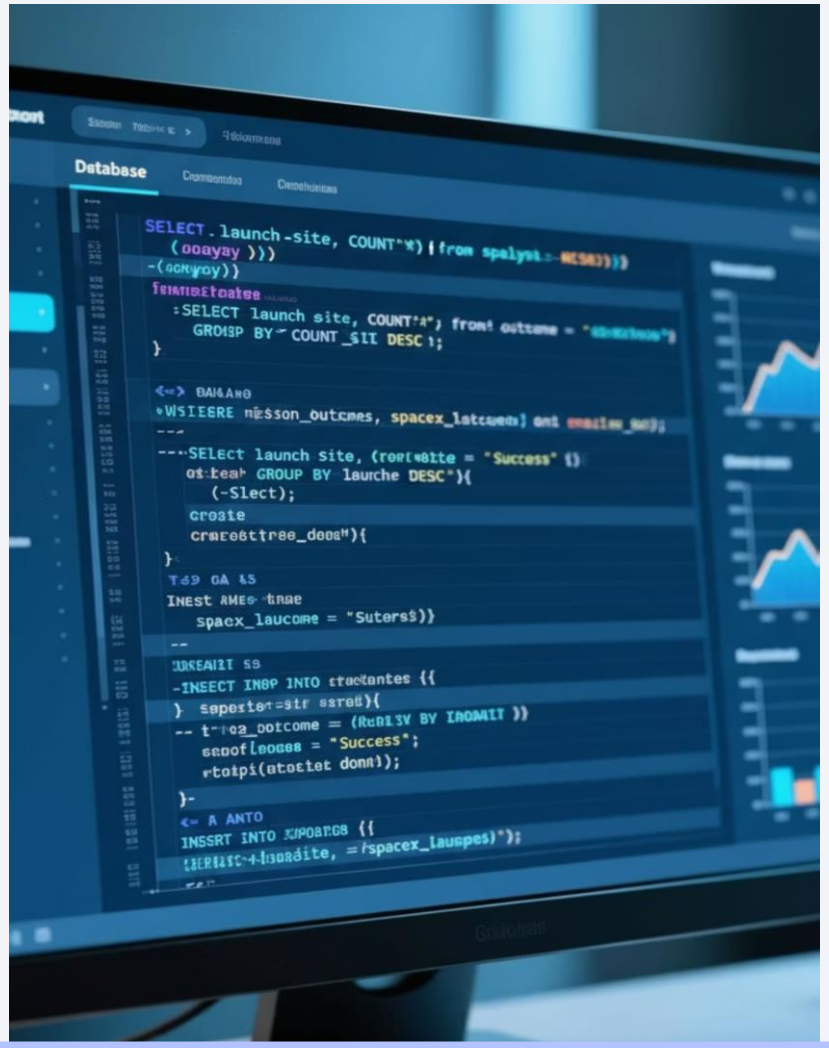
- List all booster versions that carried maximum payload mass using subquery with MAX function

Date/Time Queries

- List date of first successful landing outcome in ground pad using MIN function
- Display month names and failure outcomes for 2015 using substr functions

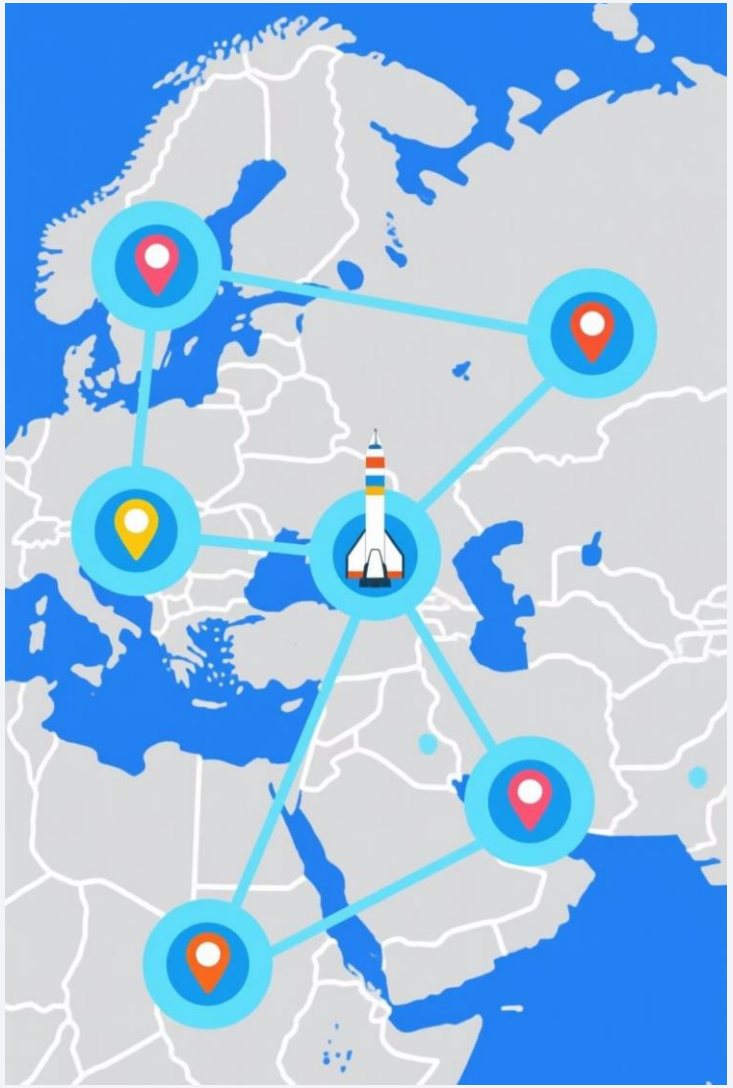
Sorting/Grouping Queries

- Display unique launch sites using DISTINCT
- Group mission outcomes by type and count totals
- Order landing outcomes by date in descending order



GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/4%20-%20EDA%20with%20SQL.ipynb*

Build an Interactive Map with Folium



Interactive maps generated using Folium provide a dynamic way to visualize and analyze SpaceX launch sites. This visualization helps in understanding geographical patterns, operational zones, and spatial relationships crucial for mission planning and analysis.

Map Objects Created

- **Markers:** Added to pinpoint exact SpaceX launch locations, each marker represents a specific geographical location where launches occurred.
- **Circles:** Added around launch sites to visualize proximity zones and operational boundaries that might influence mission planning.
- **Lines:** Drew connections between launch sites and related point like coastline.

Purpose and Benefits

- **Markers** provide spatial reference for launch locations.
- **Circles** illustrate safety perimeters and impact zones.
- **Lines** show spatial relationships and operational dependencies.
- **Interactive features** enable detailed exploration and analysis of the launch data.

GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/6%20-%20Interactive%20Visual%20Analytics%20with%20Folium.ipynb*

Build a Dashboard with Plotly Dash

Interactive dashboards created with Plotly Dash offer dynamic visualizations and user-driven exploration of complex datasets, providing immediate insights into operational metrics and performance trends.

Dashboard Visualizations

Pie Chart

- Shows how successful and unsuccessful launches are distributed
- Aids in the visualization of performance patterns and the overall success rate
- Provides immediate insight into mission reliability metrics at a glance

Payload Scatter Plot

- Illustrates the connection between launch result and "PayloadMass"
- Enables users to investigate how "PayloadMass" affects outcome results
- Reveals patterns and correlations in launch performance data across different payload categories

Interactive Controls

Launch Site Dropdown

- Allows users to choose particular launch sites for examination
- Makes geographic location-based filtering and targeted research easier
- Facilitates comparison of performance across different launch facilities

Payload Range Slider

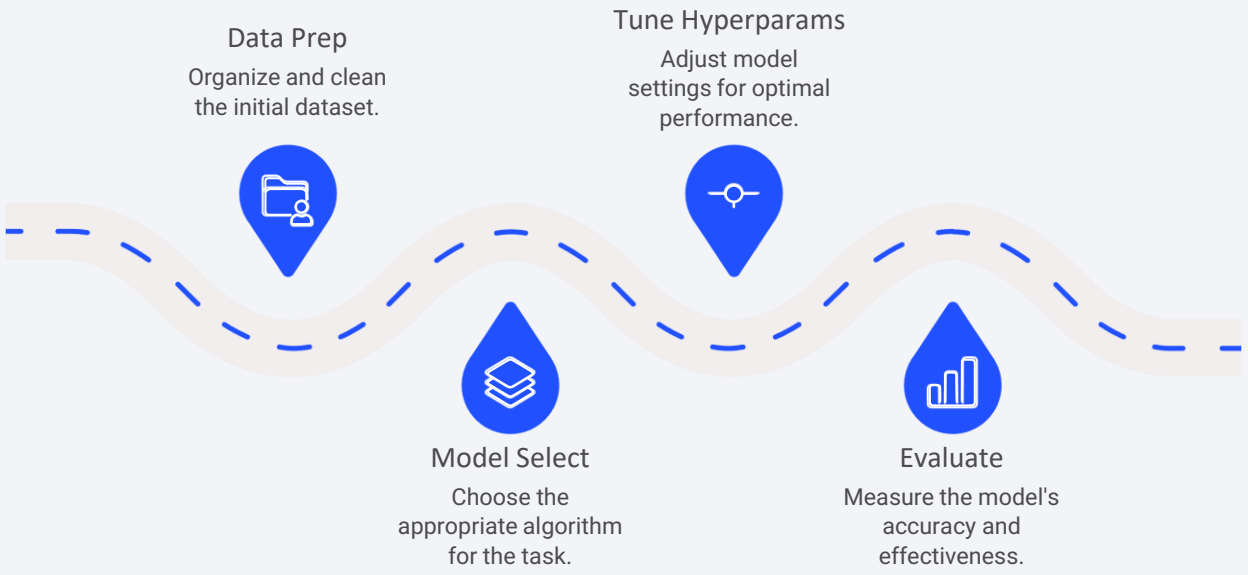
- Enables users to dynamically modify "PayloadMass" ranges
- Provides flexibility in analyzing launch success with respect to changes in "PayloadMass"
- Allows users to isolate specific payload categories for detailed analysis

Benefits and Purpose

- Interactive controls enhance user engagement and exploration capabilities
- Real-time filtering enables deeper insights into specific data segments
- Visual representations make complex data patterns immediately accessible
- Dashboard design supports both high-level overview and detailed analysis

GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/7%20-%20Interactive%20Dashboard%20with%20Plotly%20Dash%20-%20spacex-dash-app.py*

Predictive Analysis (Classification)



Data Preprocessing

- Standardized features to ensure all variables contribute equally
- Split data into training and test sets for model validation

Model Selection

- Explored multiple classification algorithms: SVM, Decision Trees, and K-Nearest Neighbors (KNN)
- Chose algorithms suitable for binary classification tasks based on project requirements

Hyperparameter Tuning

- Used GridSearchCV to systematically search for optimal hyperparameters
- Tuned parameters such as C (SVM), max_depth (Decision Trees), and n_neighbors (KNN)

Model Evaluation

- Evaluated models using cross-validation techniques to ensure robustness and generalizability
- Utilized metrics like accuracy, precision, recall, and F1-score to assess model performance
- Identified the model with the highest accuracy on the test set as the best performer

GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/8%20-%20Machine%20Learning%20Prediction.ipynb*

Predictive Analysis (Classification)

Main Implementation for Notebook Flowchart

1 Import Library & Load Data

Import the library & load the data need for prediction

2 Prepare Processing

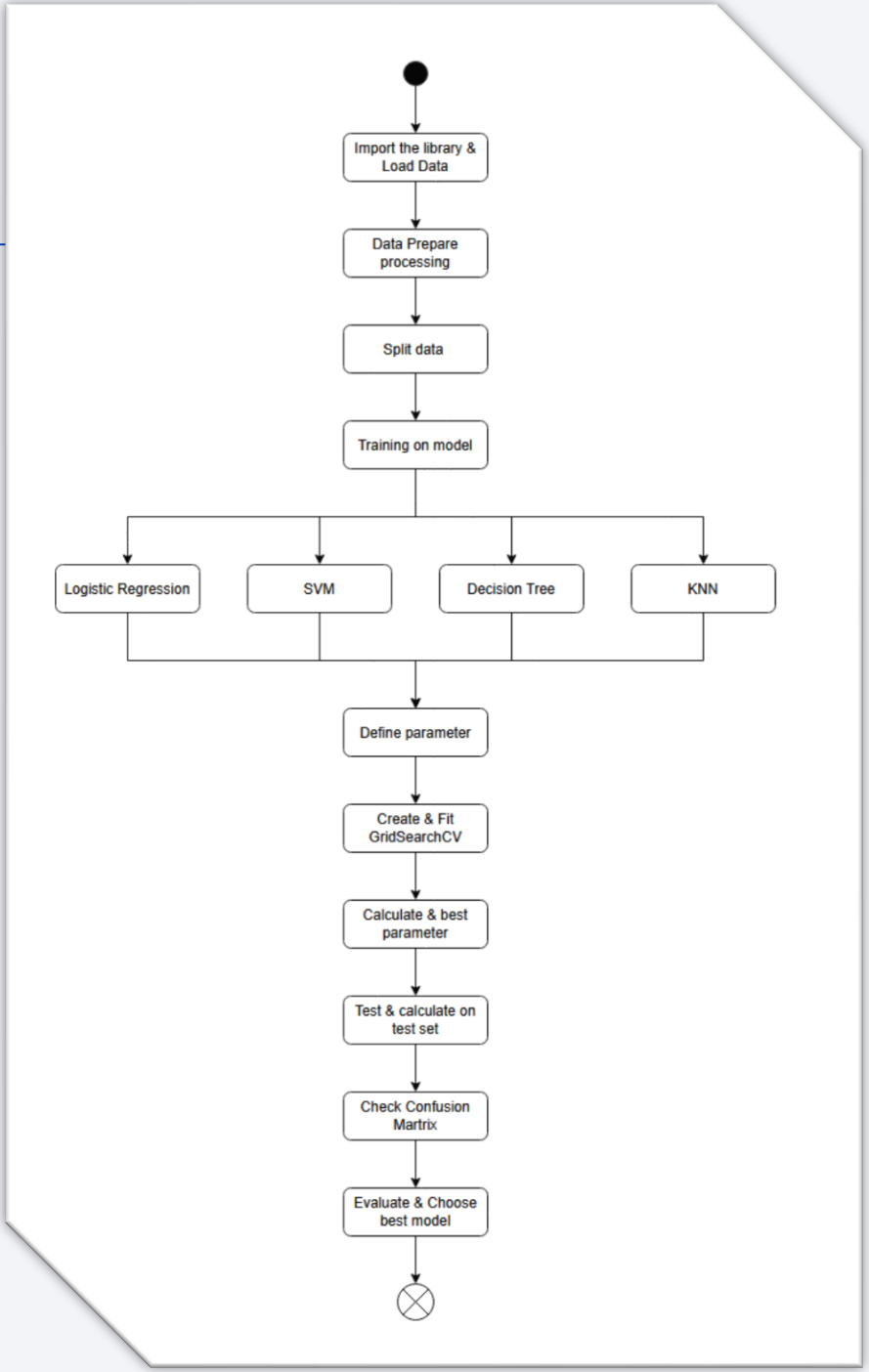
Standardize the data and split the data into train & test set.

3 Training the model

Define the parameter for each mode. Using the GridSearchCV with fold = 10 for cross-validation to find tuned hyperparameters for each model.

4 Evaluate the model

Fit and test on the test set. Calculate the accuracy for each model. Evaluate and choose the best model.



GitHub reference: *https://github.com/Anh-96/Applied-Data-Science-Capstone-Space-X-Falcon-9-First-Stage-Landing-Prediction/blob/main/8%20-%20Machine%20Learning%20Prediction.ipynb*

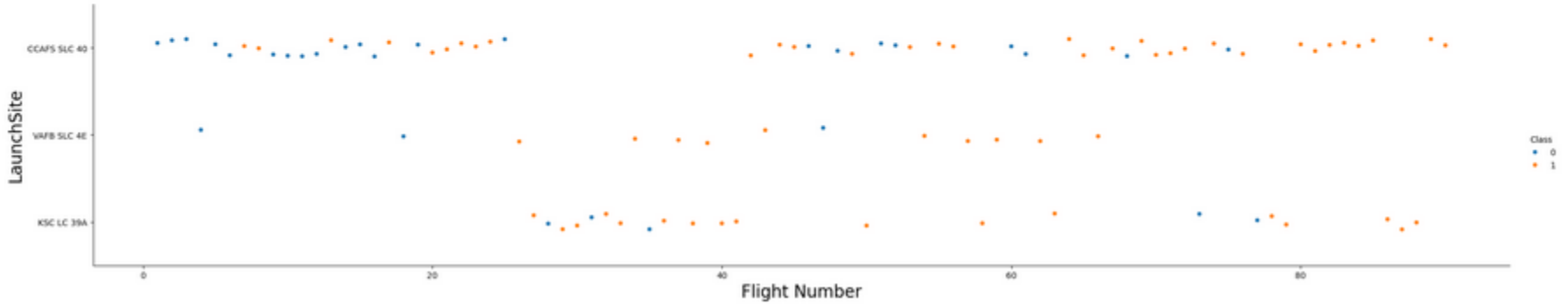
Results

TASK 1: Visualize the relationship between Flight Number and Launch Site

Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

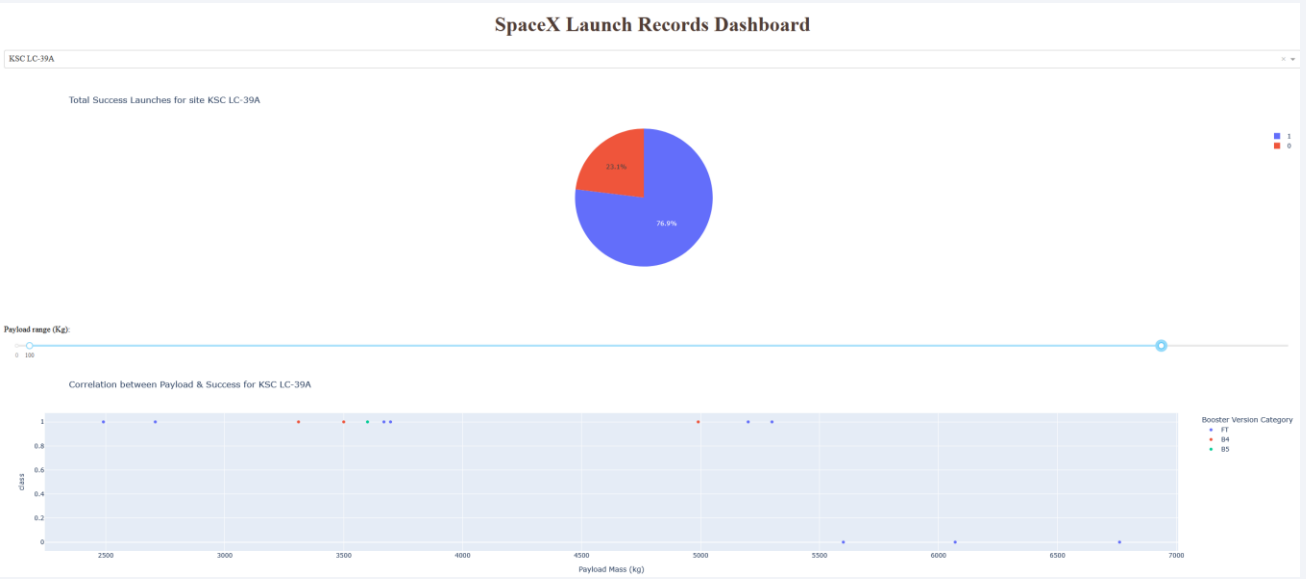
In [5]:

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class va
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



EDA results example

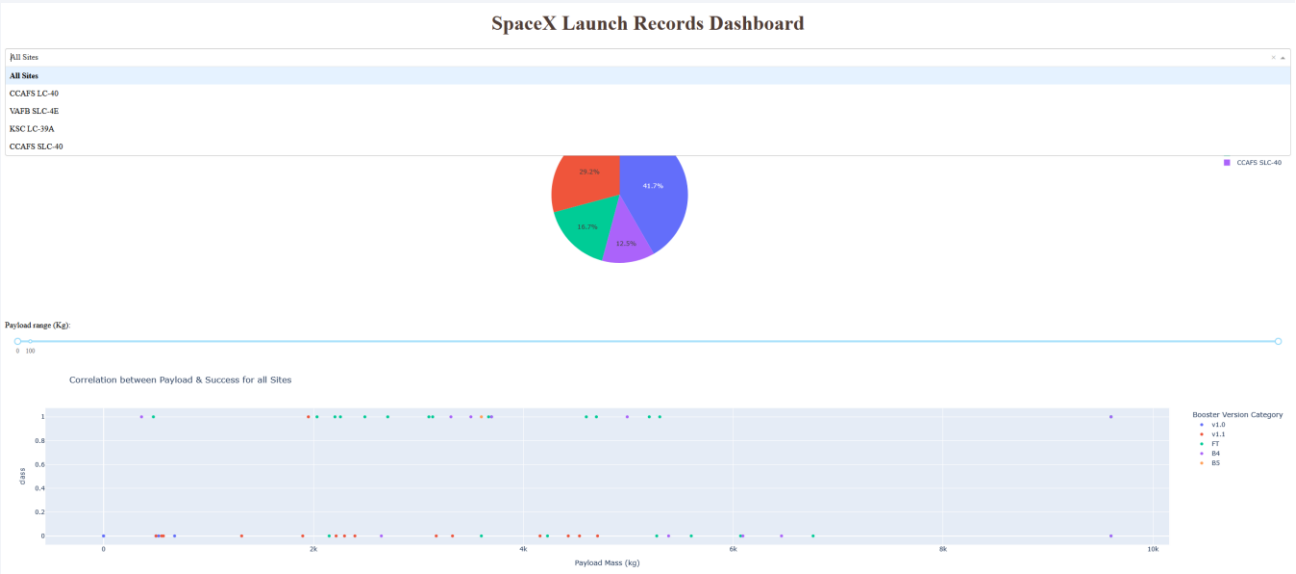
Results



Interactive analytics demo

A site is selected & A range is choosen

All Sites Select



Results

Calculate the accuracy on the test data using the method `score`:

```
[14]: logreg_score = logreg_cv.score(X_test, Y_test)
      logreg_score
```

```
[14]: 0.8333333333333334
```

Calculate the accuracy on the test data using the method `score`:

```
[19]: svm_score = svm_cv.score(X_test, Y_test)
      svm_score
```

```
[19]: 0.8333333333333334
```

Calculate the accuracy of tree_cv on the test data using the method `score`:

```
[24]: tree_score = tree_cv.score(X_test, Y_test)
      tree_score
```

```
[24]: 0.9444444444444444
```

Calculate the accuracy of knn_cv on the test data using the method `score`:

```
[29]: knn_score = knn_cv.score(X_test, Y_test)
      knn_score
```

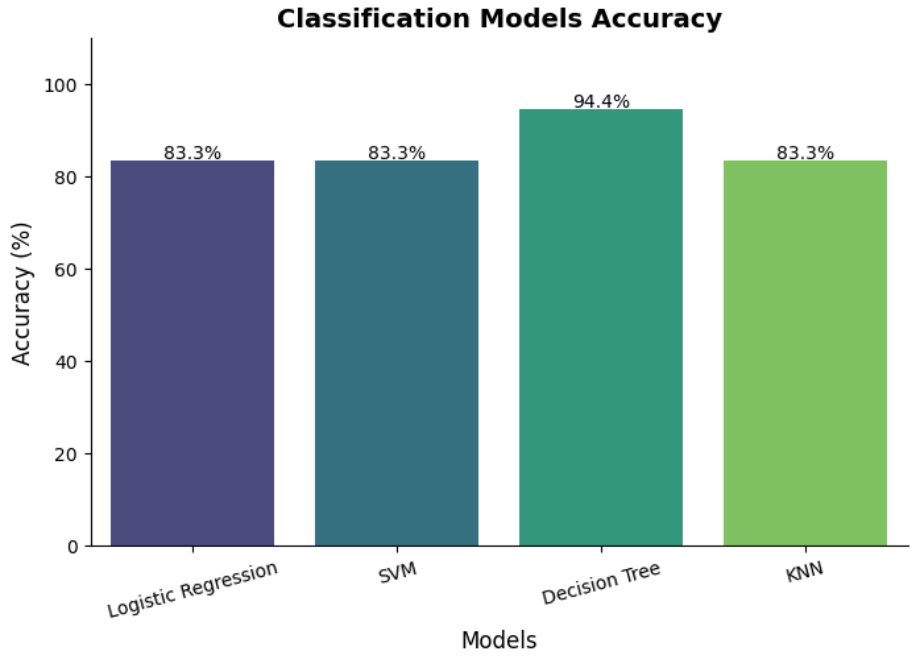
```
[29]: 0.8333333333333334
```

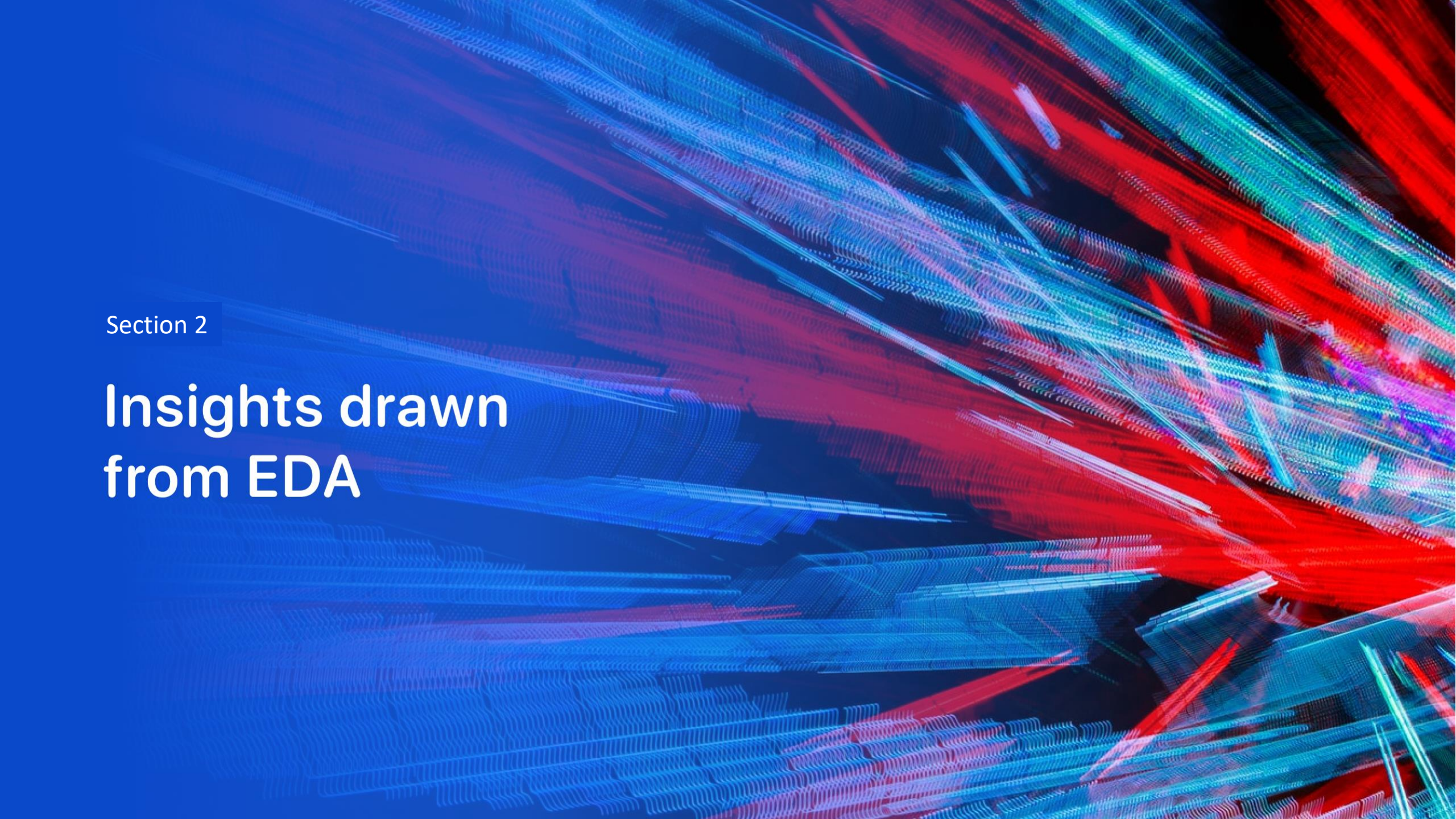
Predict Analysis Result

TASK 12

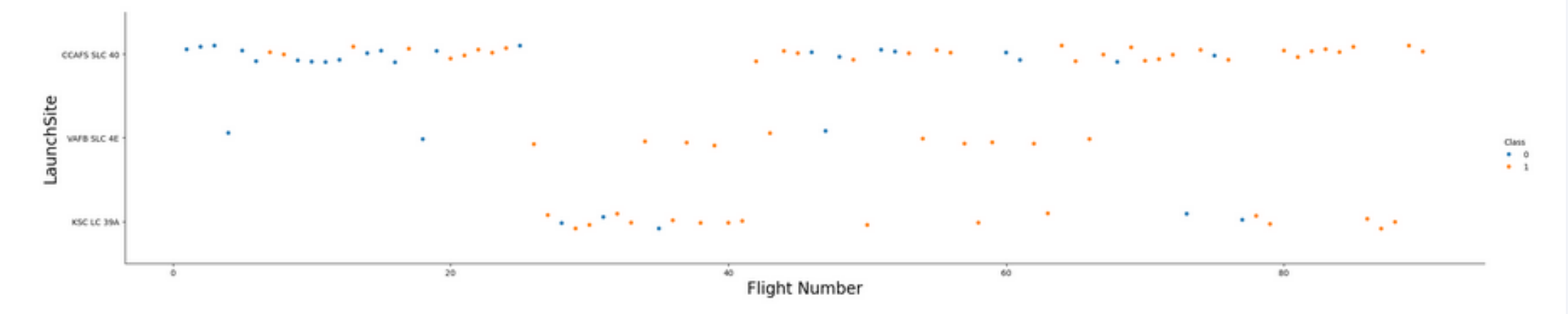
Find the method performs best:

```
[39]: acc_data = {
          'models': ['Logistic Regression', 'SVM', 'Decision Tree', 'KNN'],
          'accuracy': [logreg_score, svm_score, tree_score, knn_score]
      }

      df_acc = pd.DataFrame(acc_data)
      df_acc['accuracy'] = df_acc['accuracy'] * 100

      plt.figure(figsize=(8, 5))
      ax = sns.barplot(data=df_acc, x='models', y='accuracy', hue='models', palette='viridis')

      for i, v in enumerate(df_acc['accuracy']):
          ax.text(i, v + 0.5, f"{v:.1f}%", ha='center', fontsize=10, color='black')

      plt.title('Classification Models Accuracy', fontsize=14, fontweight='bold')
      plt.xlabel('Models', fontsize=12)
      plt.ylabel('Accuracy (%)', fontsize=12)
      plt.ylim(0, 110)
      plt.xticks(rotation=15)
      sns.despine()
      plt.show()
```



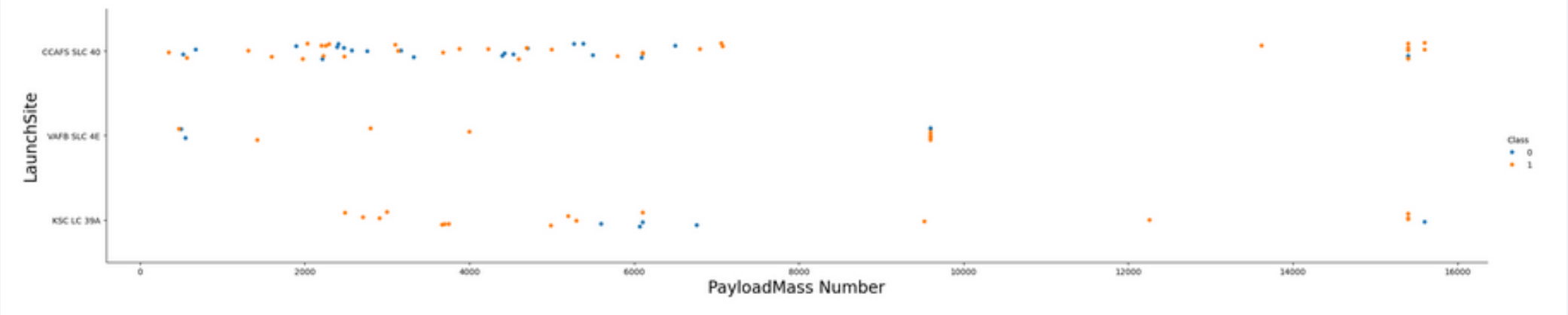

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



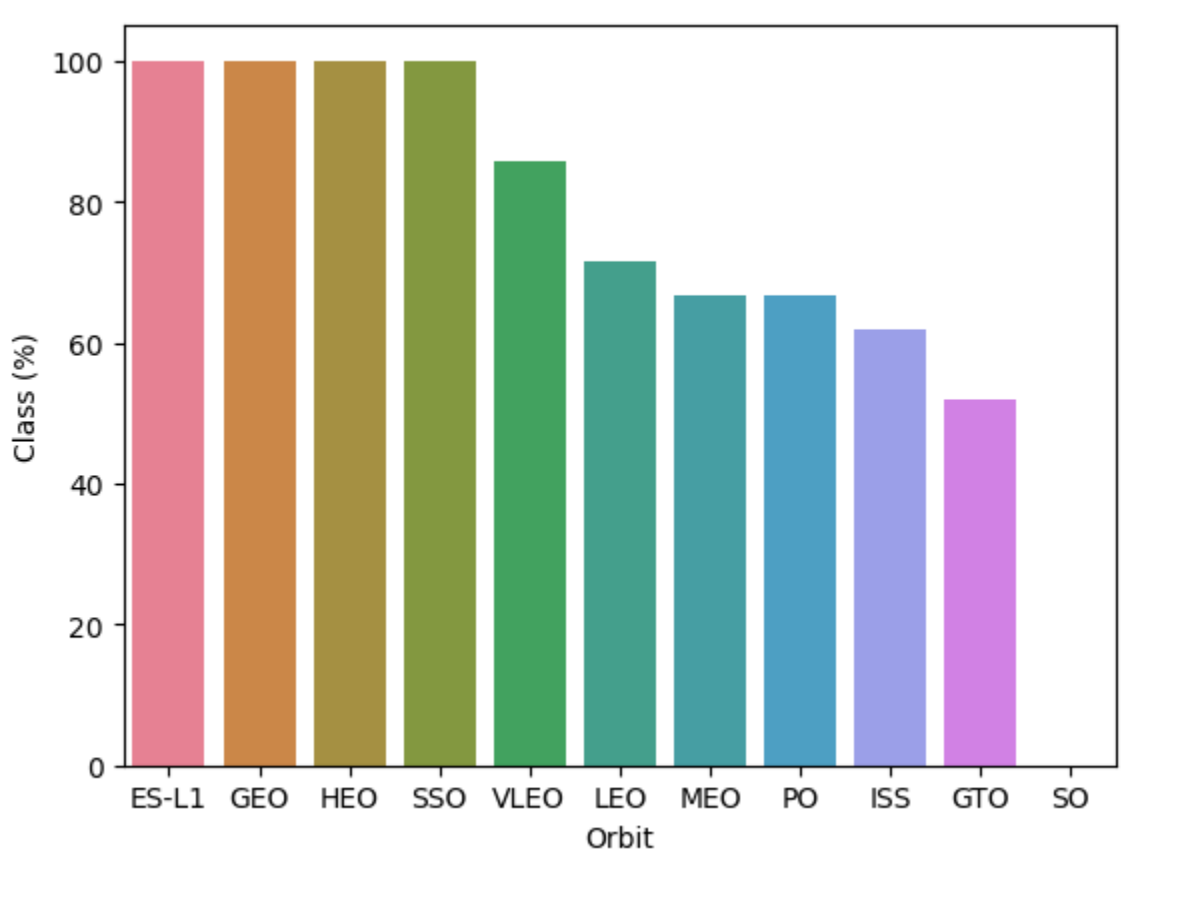
- Every launch site has a combination of successful (1 - orange) and unsuccessful (2 - blue) landings, suggesting that landing success may be influenced by variables other than the launch site itself
- Launches occur at all sites across a broad range of flight numbers, indicating consistent activity over time without a discernible trend of rising or falling landing success.

Payload vs. Launch Site



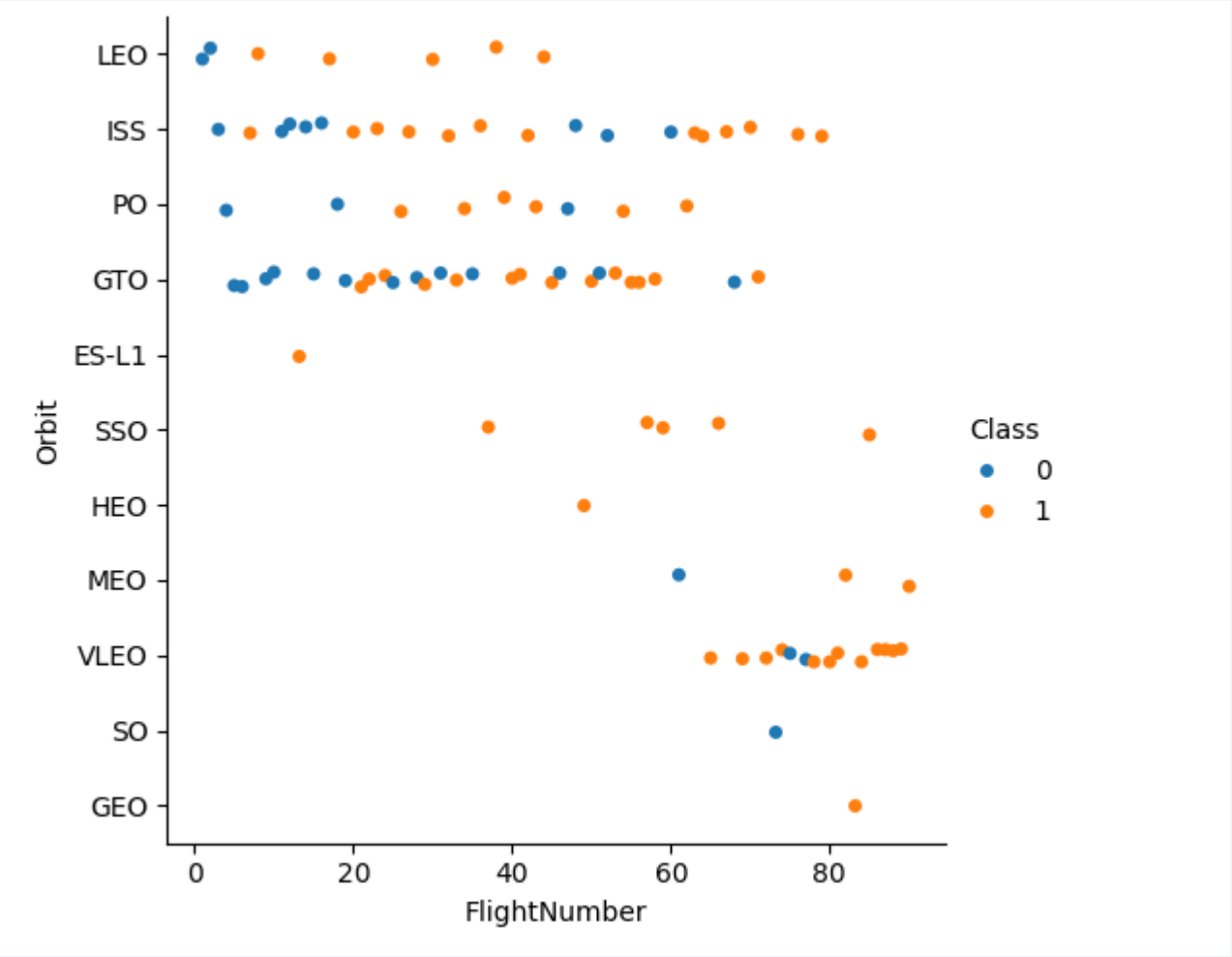
- While the VAFB SLC 4E and KSC LC 39A locations offer a greater variety of payload masses, indicating different mission profiles, the majority of launches from the CCAFS SLC 40 facility handle payloads under 10,000 kg.
- The KSC LC 39A launch pad is often used to launch larger payloads and has even been used to launch up to 15,000 kg, showing its suitability for high capacity missions.

Success Rate vs. Orbit Type



- The 100% success record of missions to ES-L1, GEO, HEO, and SSO orbits shows how dependable these orbits are for first-stage landing success.
- Compared to other orbit types, the GTO orbit type has a much lower success rate, indicating that missions to this orbit might be more difficult or complex.

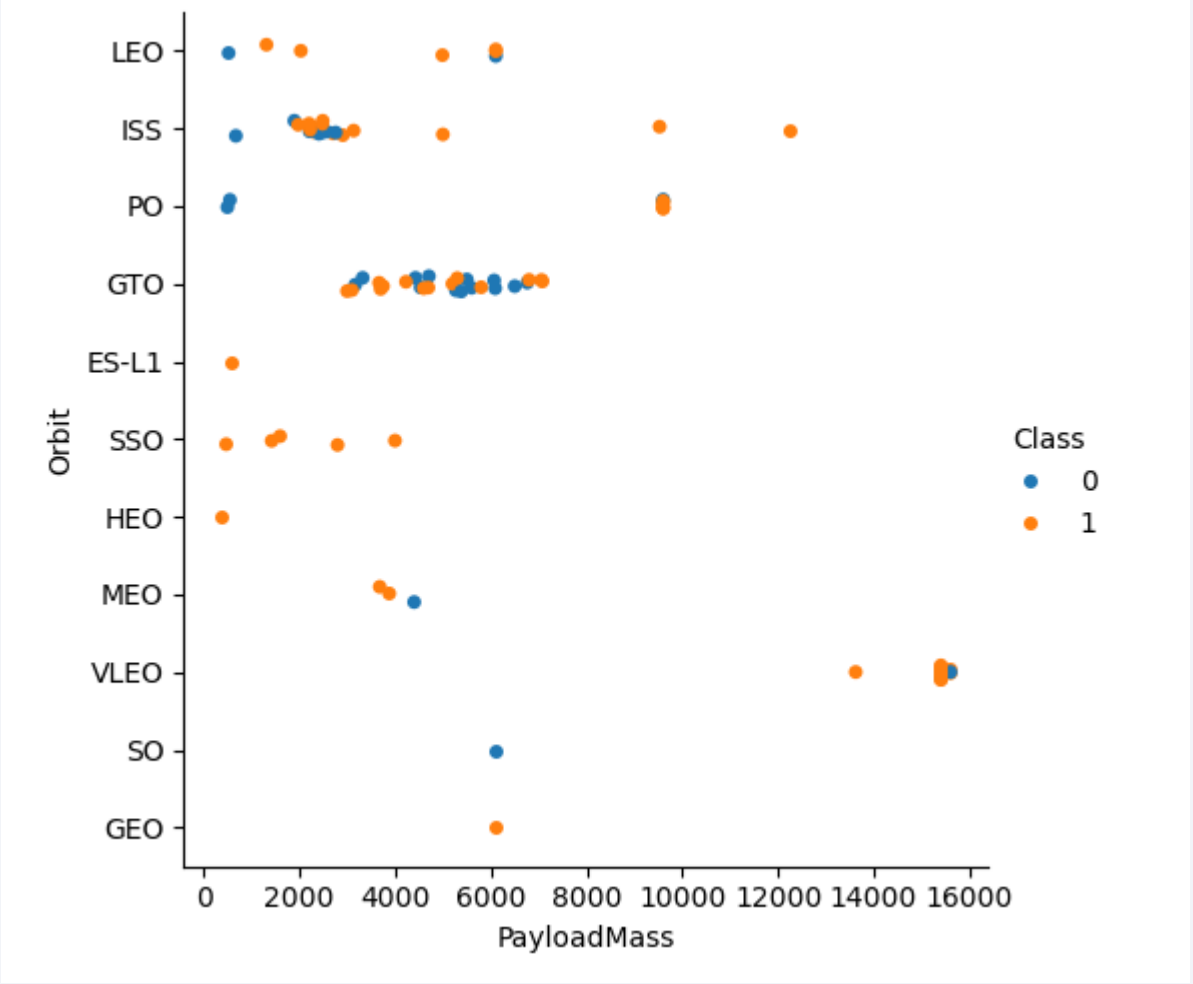
Flight Number vs. Orbit Type



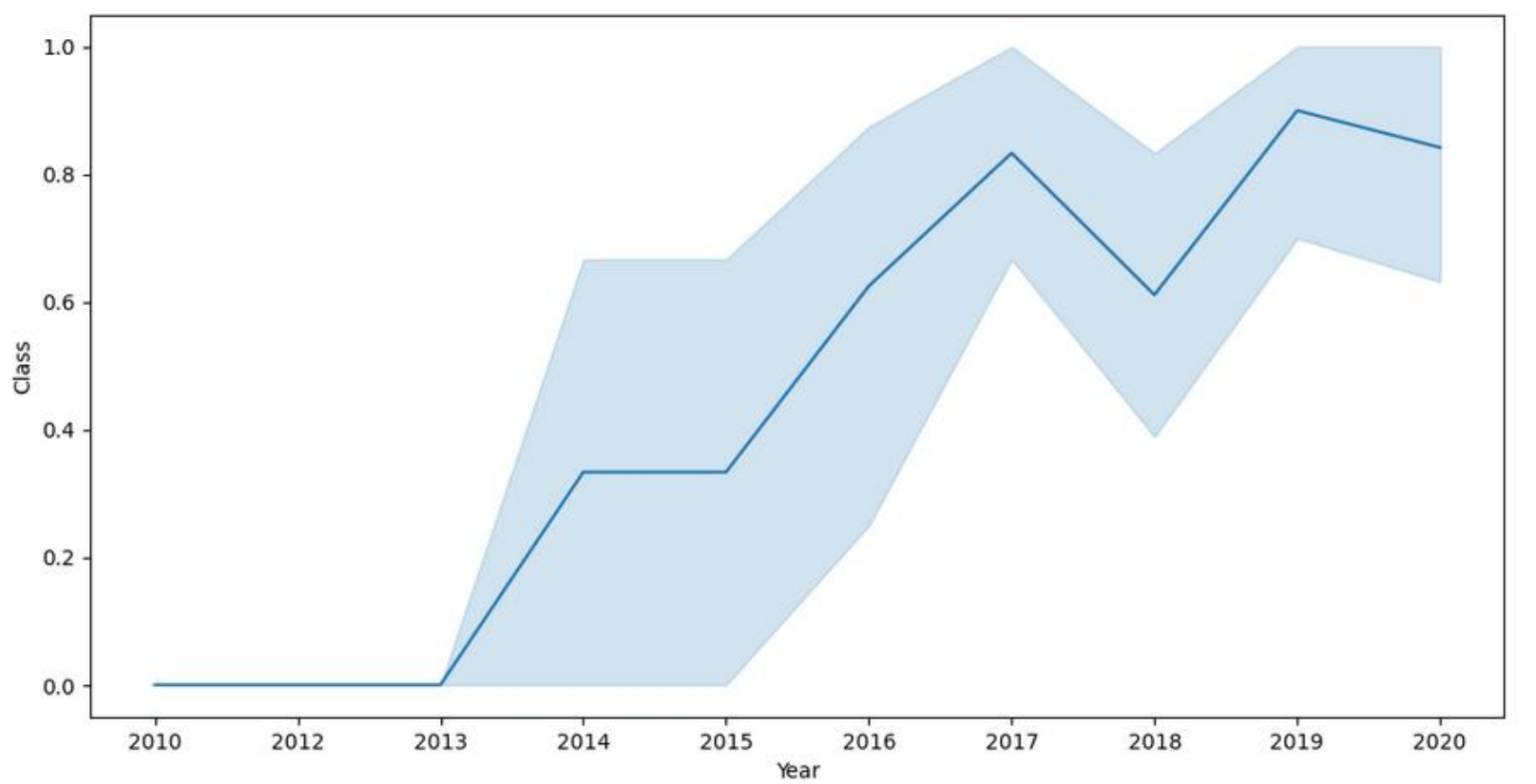
- Higher flight numbers considerably increase the Launch success percentage, suggesting that experience and iterative improvements lead to greater results.
- Although the results of early flights to GTO, ISS, LEO & PO orbits were inconsistent, more recent missions to these orbits have demonstrated a higher success rate, which is indicative of improvements in mission planning and execution.

Payload vs. Orbit Type

- Across all orbit types, successful landings are more common, particularly for cargoes weighing less than 5000 kg.
- A combination of triumphs and failures are shown with payload masses over 6,000 kg, suggesting that bigger payloads are more challenging.



Launch Success Yearly Trend



- Since 2013, the annual launch success rate has improved significantly, reaching over 0.8 (80%) by 2020.
- The general trend shows growing dependability and success in Falcon 9 launches over time, notwithstanding a decline in 2018.

All Launch Site Names

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

```
In [11]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
 * sqlite:///my_data1.db
Done.
```

Out[11]:

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

- There are 4 Launch Sites:
  - CCAFS LC-40
  - VAFB SLC-4E
  - KSC LC-39A
  - CCAFS SLC-40

Launch Site Names Begin with 'CCA'

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

In [17]:
```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

Out[17]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outc
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (paracl
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 (paracl
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No att
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No att
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No att

- It shows 5 Launch Sites start with 'CCA' and all of it is from CCAFS LC-40

Total Payload Mass

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

In [18]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

Out[18]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

- The total Payload Mass(Kgs) carried by Boosters Lauched by NASA (CRS) is 45596 kg

Average Payload Mass by F9 v1.1

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

In [20]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Average Payload Mass", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version = 'F9 v1.1';
```

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

Out[20]:

Average Payload Mass	Customer	Booster_Version
2928.4	SES	F9 v1.1

- The average Payload Mass by F9 v1.1 is 2928.4

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

In [24]:
```
%sql SELECT MIN(Date), Landing_Outcome FROM 'SPACEXTBL' WHERE Landing_Outcome = 'Success (ground pad)' ;
```

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

Out[24]:

MIN(Date)	Landing_Outcome
2015-12-22	Success (ground pad)

- 2025-12-22 is the day when first successful landing outcome in ground pad

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

In [27]:
```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_, Landing_Outcome  FROM 'SPACEXTBL' WHERE PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000 AND Landing_Outcome = 'Success (drone ship)' ;
```

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

Out[27]:

Booster_Version	PAYLOAD_MASS__KG_	Landing_Outcome
F9 FT B1022	4696	Success (drone ship)
F9 FT B1026	4600	Success (drone ship)
F9 FT B1021.2	5300	Success (drone ship)
F9 FT B1031.2	5200	Success (drone ship)

- There are 4 Booster Version with Payload between 4000 and 6000:
  - F9 FT B1022 with 4696 kg Payload Mass
  - F9 FT B1026 with 4600 kg Payload Mass
  - F9 FT B1021.2 with 5300 kg Payload Mass
  - F9 FT B1031.2 with 5200 kg Payload Mass

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

In [43]:
```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) AS Total FROM 'SPACEXTBL' GROUP BY Mission_Outcome;
```

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

Out[43]:

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

- Total number of Success is 98 + 1 + 1 = 100 and Failure is 1

Boosters Carried Maximum Payload

List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.

In [45]:
```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM 'SPACEXTBL' WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM 'SPACEXTBL');
```

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

Out[45]:

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

- Most the Booster Version come from F9 B5 series and there Maximum Playload Mass is 15600

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.

In [51]:
```
%sql SELECT substr(Date, 6, 2) AS 'Month', substr(Date, 0, 5) AS 'Year', Booster_Version, Launch_Site, Landing_Outcome FROM SPACEXTBL WHERE substr(Date,0,5)='2015' AND Landing_Outcome =  'Failure (drone ship)';
```

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

Out[51]:

Month	Year	Booster_Version	Launch_Site	Landing_Outcome
01	2015	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- There are 2 Failure Landing Outcome in drone ship in 2015

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.

In [65]:
```
%sql SELECT Landing_Outcome, COUNT(*) AS 'Total' FROM SPACEXTBL WHERE Date > '2010-06-04' AND Date < '2017-03-20' AND (Landing_Outcome = 'Failure (drone ship)' OR Landing_Outcome = 'Success (ground pad)') GROUP BY Landing_Outcome ORDER BY Date DESC;
```

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

Out[65]:

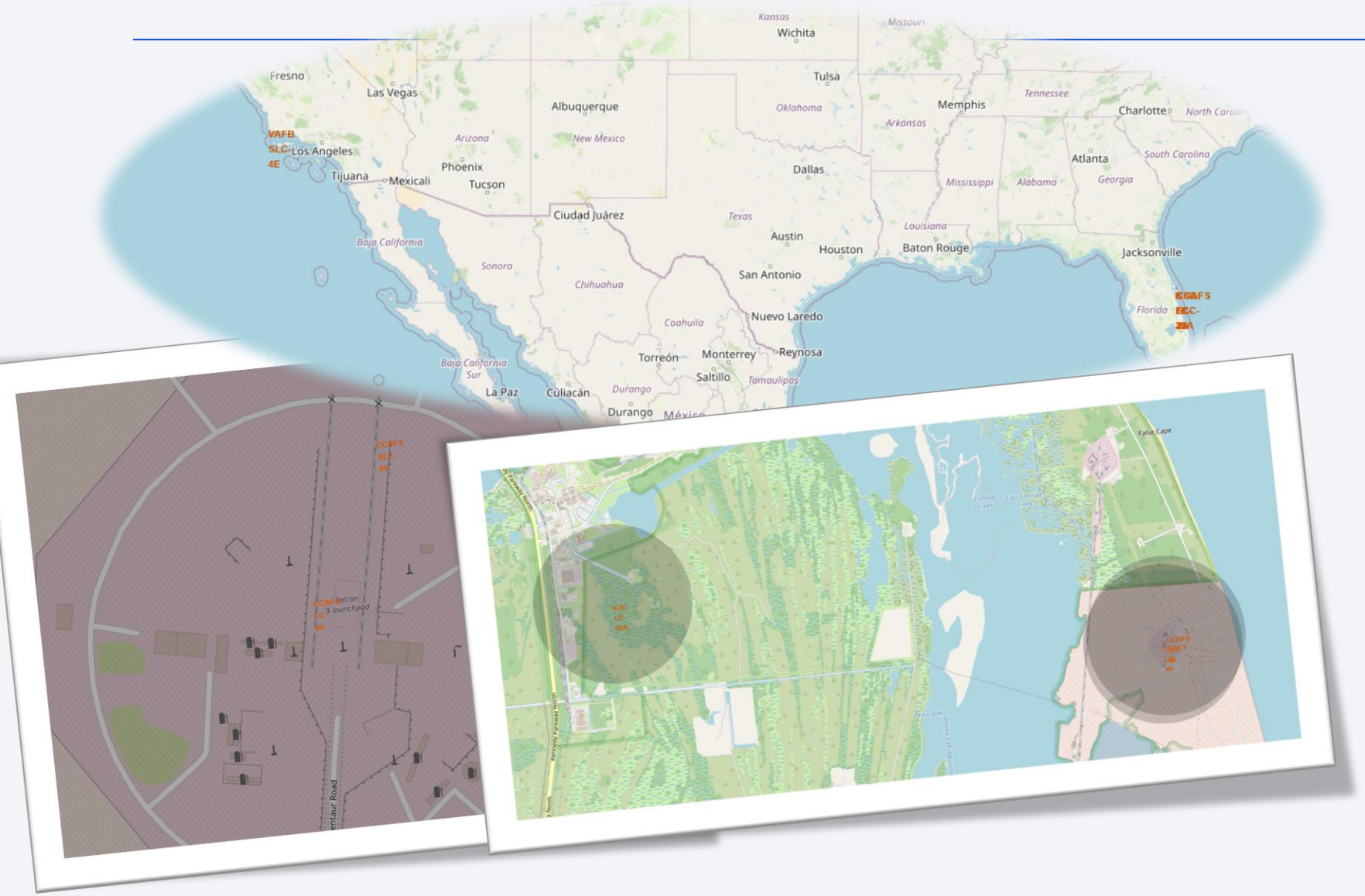
Landing_Outcome	Total
Success (ground pad)	3
Failure (drone ship)	5

- There are 3 Success when landing in ground pad and 5 Failure when landing on drone ship between 2010-06-04 and 2017-03-20

Section 3

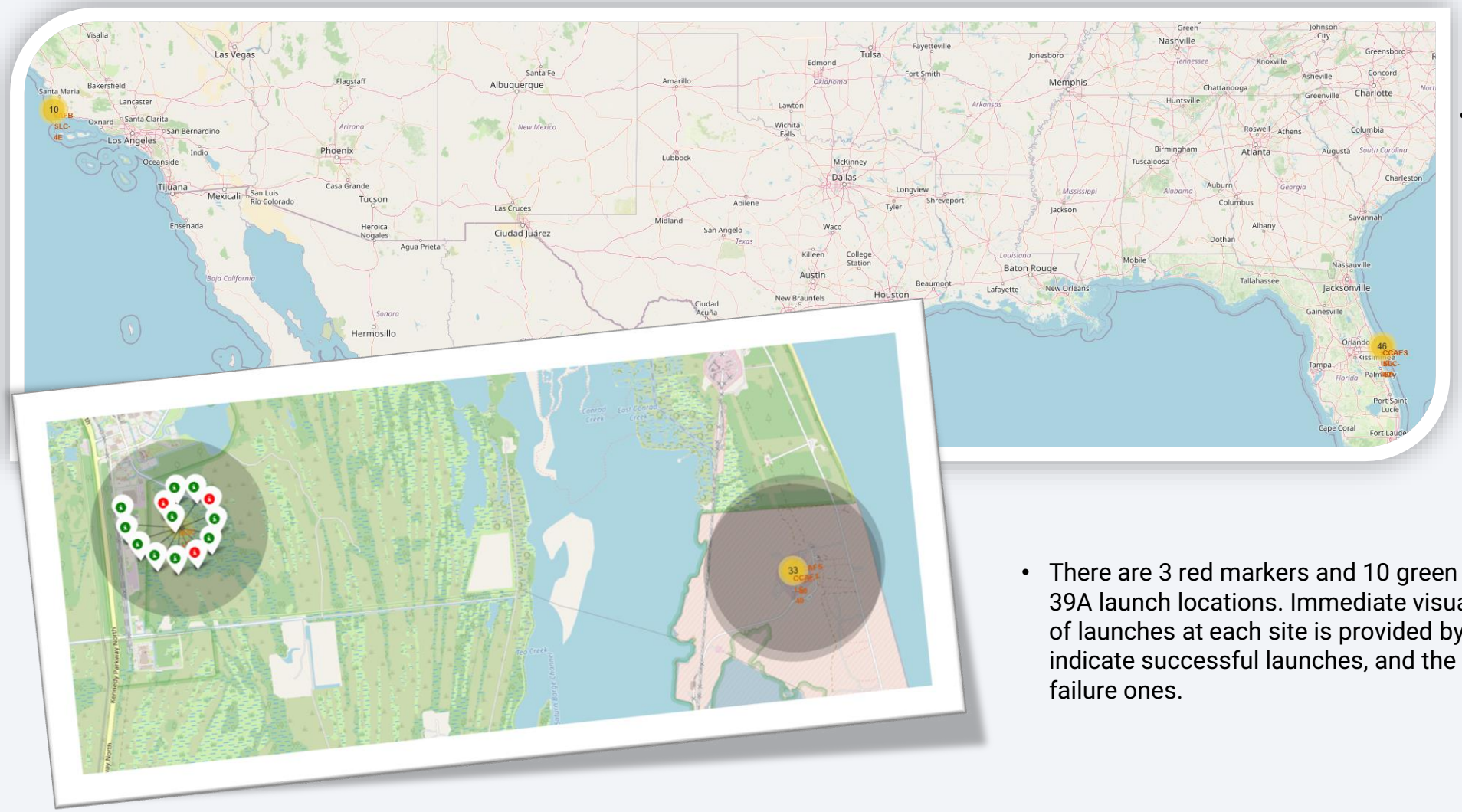
Launch Sites Proximities Analysis

Mark all launch sites on a map



- There are 4 Launch Site on map and all of it quite close to the coast.
- VAFB SLC-4E is near coast of California
- CCAFS SLC-40 & CCAFS LC-40 is very close to each other
- CCAFS SLC-40, CCAFS LC-40 & KSC LC-39A is near coast of Florida

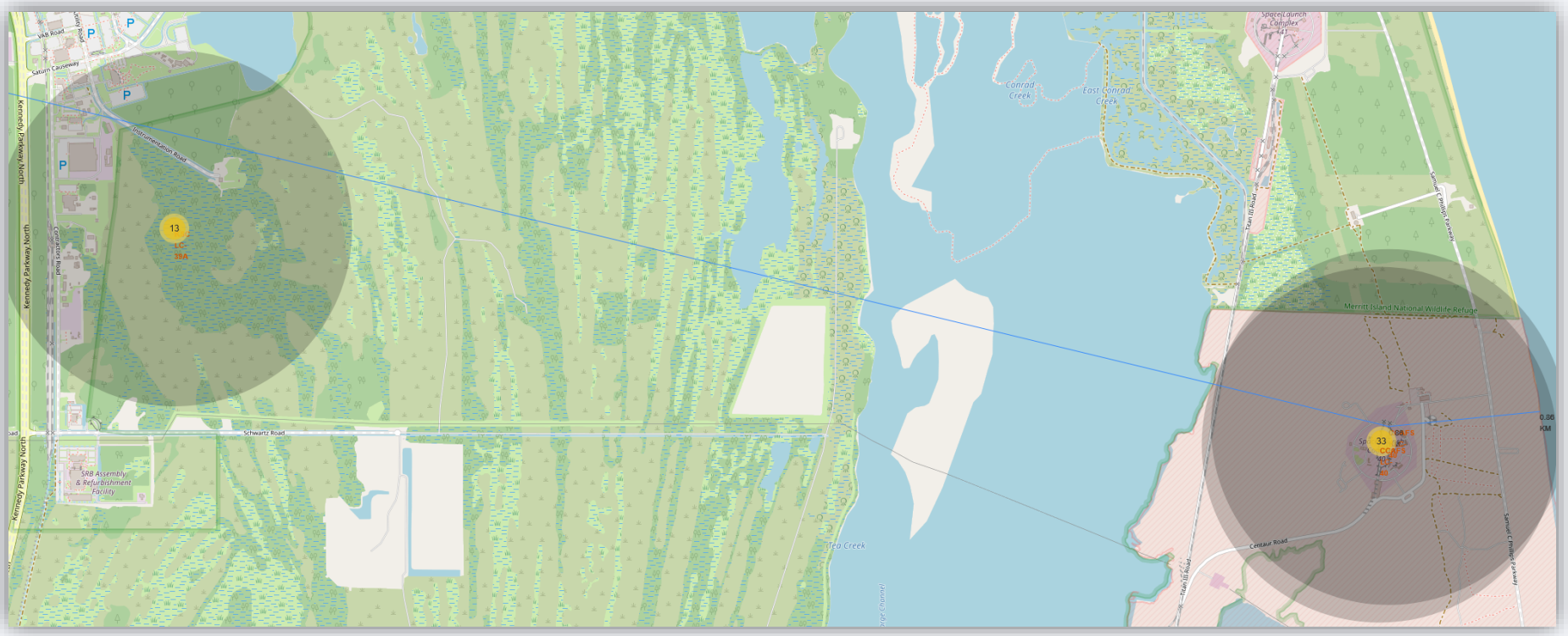
Mark the success/failed launches for each site on the map



- SpaceX launch data may be explored and analyzed more effectively thanks to this improved representation with clustered markers. It is easier to learn more about the features and distribution of SpaceX launches by looking at the marker colors and popup data.

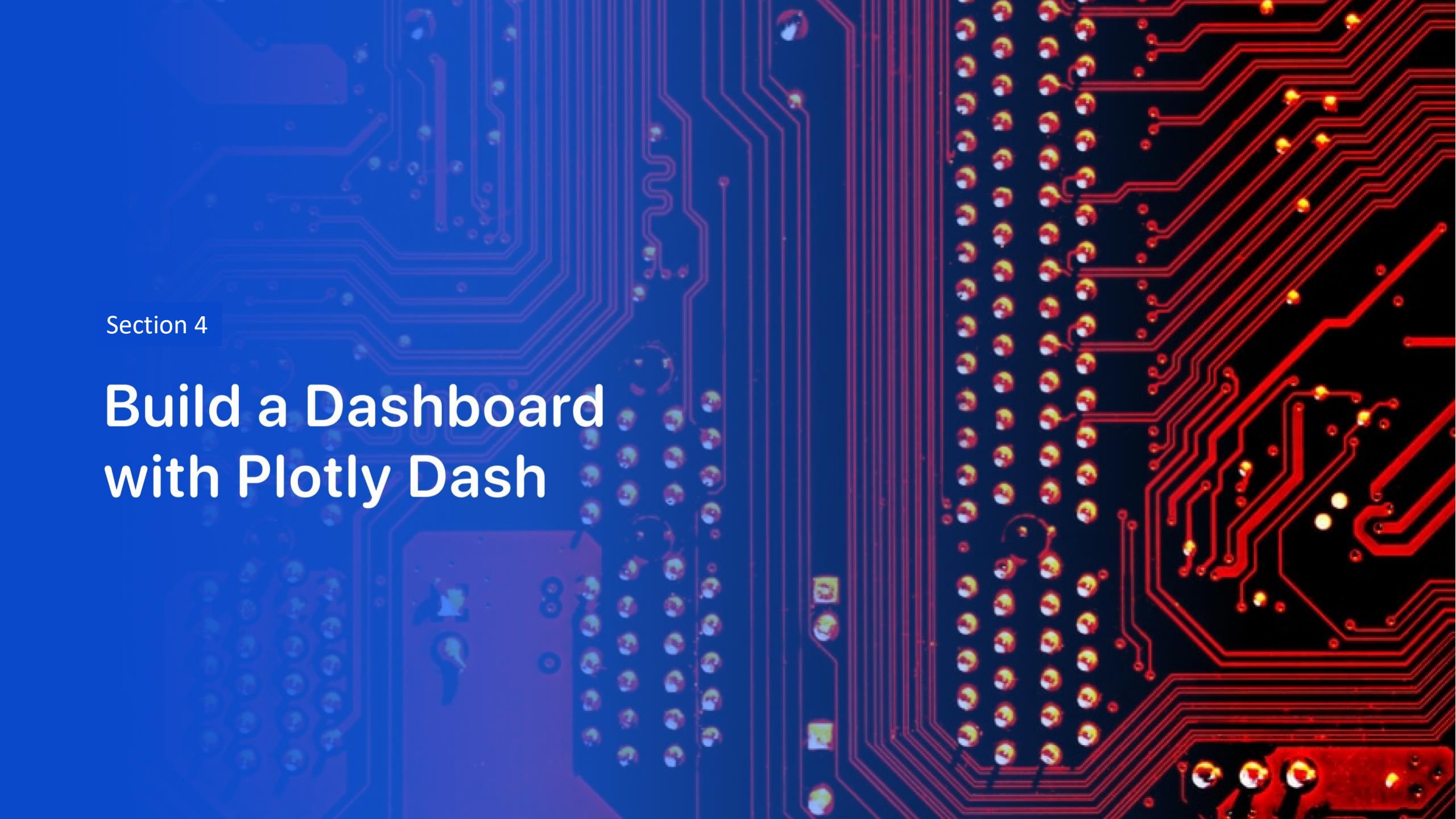
- There are 3 red markers and 10 green markers among the 13 KSC LC-39A launch locations. Immediate visual feedback on the performance of launches at each site is provided by the green markers, which indicate successful launches, and the red markers, which may signify failure ones.

Calculate the distances between a launch site to its proximities

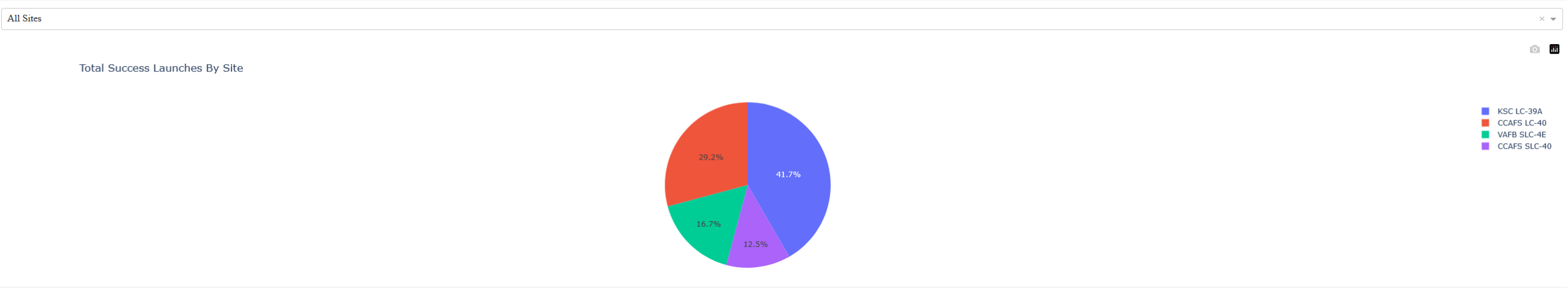


- The CCAFS SLC-40 launch site's distance from the nearest coastline is roughly 0.86 kilometers. The launch site's closeness to the coast is highlighted by the additional Polyline, which clearly displays the straight-line distance.

- Launch sites typically have this close proximity to the coast in order to enable safe recovery operations and over-water flight paths, hence minimizing risk to inhabited areas.


Section 4
Build a Dashboard with Plotly Dash

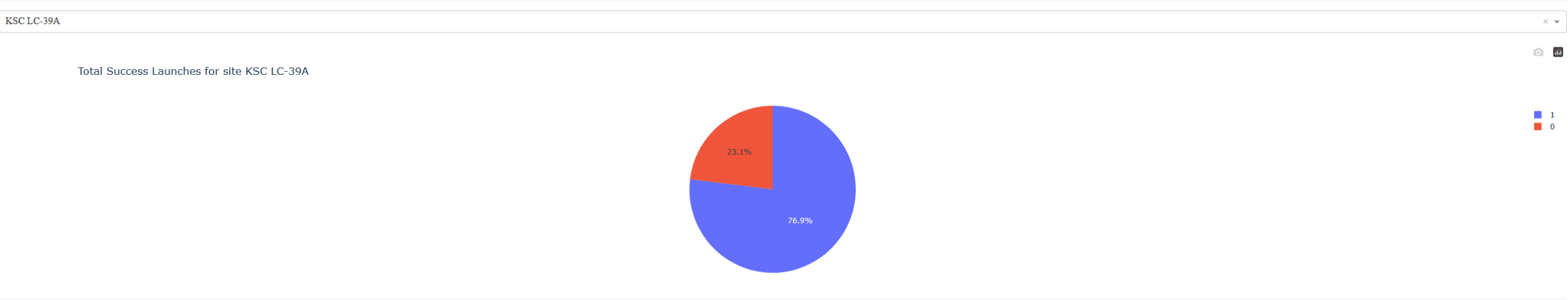
Total Success Launches By Site



- With 41.7% of all successful launches, the KSC LC-39A launch site has the most successful launches. This suggests that KSC LC-39A is a very dependable location for SpaceX launches.

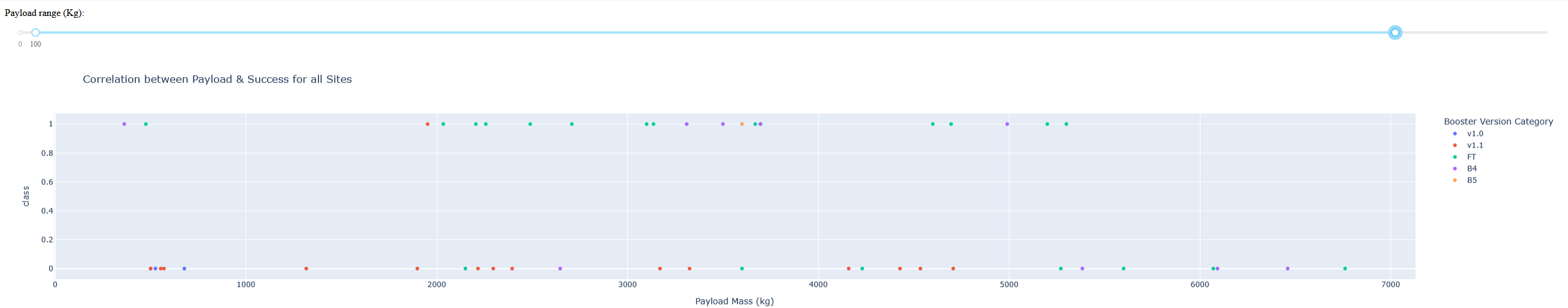
- The other:
  - CCAFS LC-40: 29.2%
  - CCAFS SLC-40: 12.5%
  - VAFB SLC-4E: 16.7%

Total Success Launches for Site KSC LC-39A



- KSC LC-39A has:
  - Successful Launch - Class 1: 76.9%
  - Failure Launch - Class 0: 23.1%
- With 76.9% successful launch, KSC LC-39A has the most reliability and effectiveness as a launch site.

Correlation between Payload & Success for all Sites



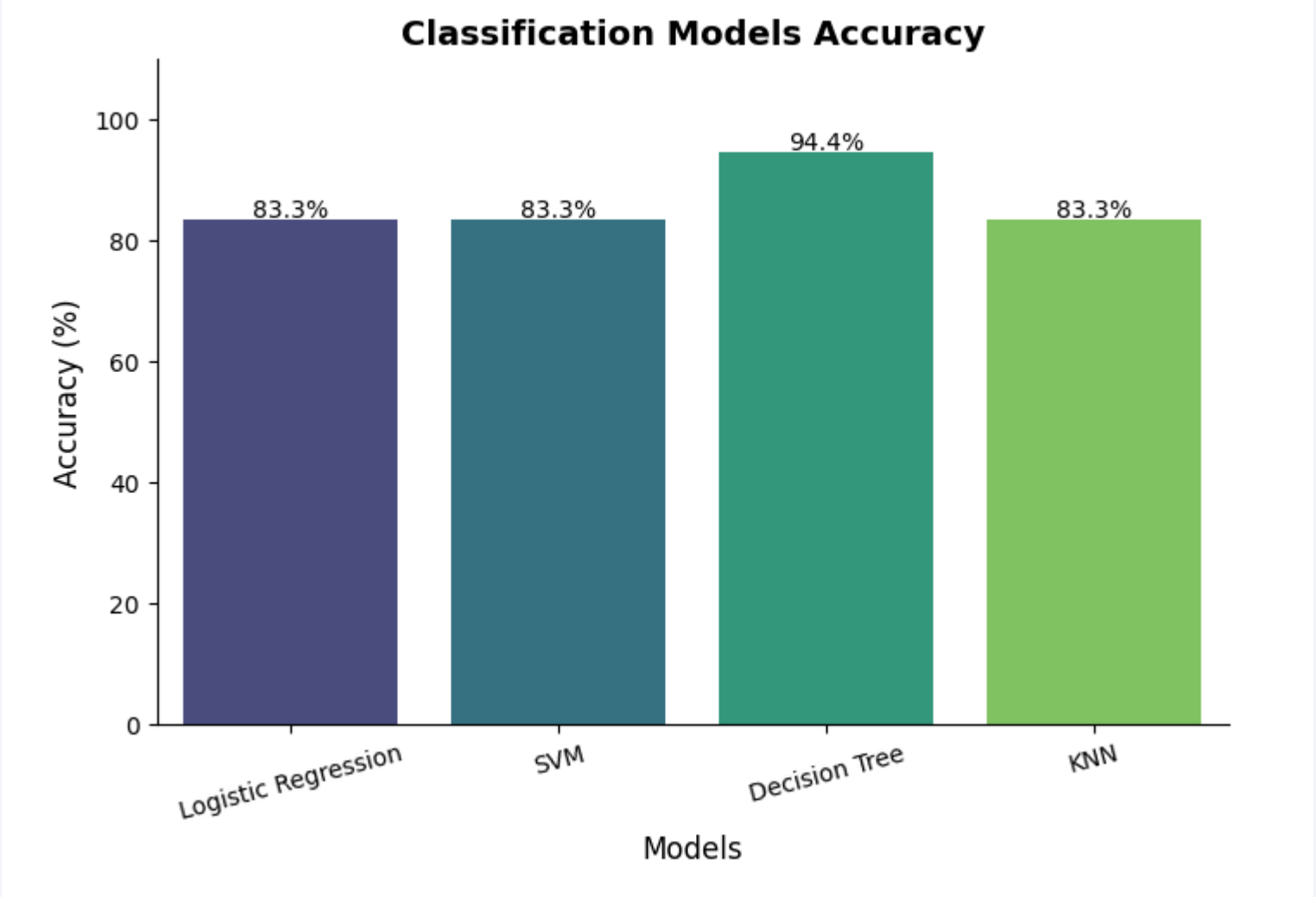
- With a high success rate over a range of payload masses, booster version FT seems to be the most popular.
- Booster version v1.0 has the fewest launches, more research may be necessary to fully comprehend its performance.
- Overall, there is no discernible pattern linking poorer success rates to larger payload masses in booster versions.

Section 5

Predictive Analysis (Classification)

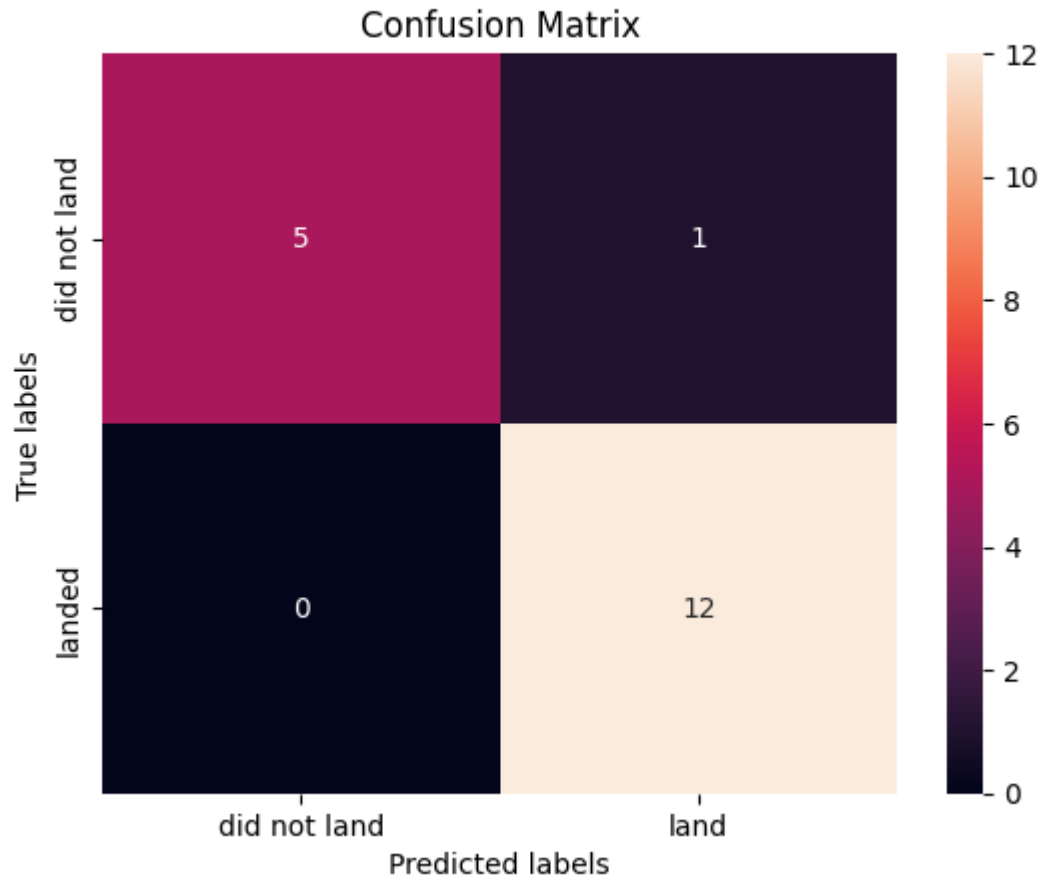
Classification Accuracy

- With 94.4% accuracy, the Decision Tree model has the best classification accuracy on the test data, according to the findings.
- In comparison to Logistic Regression, SVM, and KNN, all of which attained 83.3% accuracy, this indicates that the Decision Tree model is more appropriate for this dataset.



Confusion Matrix

```
[25]: yhat = tree_cv.predict(X_test)
      plot_confusion_matrix(Y_test,yhat)
```



- The result can be explain:
  - 5 cases of “no landing” → correct prediction is “no landing”
  - 1 case of “no landing” → wrong prediction is “landing”
  - 12 cases of “landing” → correct prediction
  - 0 cases of “landing” → wrong prediction
- The model's ability to predict Falcon 9 first stage landings was demonstrated by its high accuracy score of 94.44%, which included a sizable number of true positives and true negatives.
- The model's ability to accurately forecast successful landings is demonstrated by the lack of false negatives.
- Although there is 1 false positive, it is not as serious as false negatives in aerospace operations. It quite acceptable because over-preparation (caused by false positives) is easier to handle than under-preparation.
- With a small bias in favor of forecasting successful landings, the model exhibits balanced performance. This is in line with the actual requirements of the aerospace sector, where good landings are crucial for planning and cost estimation.

Conclusions

The analysis of SpaceX Falcon 9 launch data demonstrates the evolution of launch reliability and the success of reusable rocket technology. Through a systematic approach involving API integration, web scraping, data cleaning, visualization, and predictive modeling, several key insights were achieved:

Key Findings:

- **Improved Launch Success Over Time:**

  The yearly success rate of Falcon 9 launches has steadily increased since 2013, surpassing 80% by 2020 — evidence of SpaceX's growing operational reliability.

- **Launch Site Performance:**

  Among all sites, KSC LC-39A showed the highest success rate (over 75%), proving to be the most dependable launch facility.

- **Payload and Orbit Insights:**

  Lighter payloads (<5,000 kg) and specific orbits (ES-L1, GEO, SSO) have a higher likelihood of successful landings, while GTO orbits remain more challenging.

- **Predictive Model Accuracy:**

  The Decision Tree Classifier achieved the highest prediction accuracy (94.4%), outperforming Logistic Regression, SVM, and KNN. This model effectively predicts landing outcomes and supports mission planning.

- **Business and Engineering Impact:**

  Data-driven insights can enhance mission risk assessment, optimize booster reuse planning, and guide payload-to-orbit allocation strategies, contributing to cost reduction and operational excellence.

Overall, this project confirms that data analytics and machine learning can provide actionable intelligence for aerospace innovation, helping companies like SpaceX achieve safer, more efficient, and more sustainable space missions.

Thank you!