IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Samuel MICHAUD
16th Jan. 2025

Outline

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

Executive Summary

- Summary of methodologies :
  - Data Collection
  - Data Wrangling
  - Data Exploration
  - Data Analysis
  - Data Prediction

- Model performances:
  - Logistic Regression: 0.8333
  - Support Vector Machine: 0.8333
  - Decision Tree: 0.7222
  - K-Nearest Neighbors: 0.8333

- Best performing model: Logistic Regression with an accuracy of 0.8333

Introduction

- SpaceX 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 SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch.

- In this project, we will predict if the Falcon 9 first stage will land successfully, as it is crucial for the future of American Space exploration

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected with the SpaceX API :
    spacex_url="https://api.spacexdata.com/v4/launches/past"
- Perform data wrangling
  - Missing values were taken care of to clean the data file
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Using 4 methods : Log Regression, SVM, Decision Tree and K-Nearest

Data Collection

API Selection: Chose the SpaceX API as the primary data source for historical launch data.

Data Retrieval: Made GET requests to the SpaceX API endpoint for past launches: https://api.spacexdata.com/v4/launches/past.

Data Wrangling: Cleaned and formatted the retrieved data to ensure it was suitable for analysis.

Helper Functions: Defined auxiliary functions to extract specific information from the API response:

- Booster Version: Extracted the name of the rocket booster.
- Launch Site: Retrieved the name, longitude, and latitude of the launch site.
- Payload Data: Collected payload mass and orbit information.

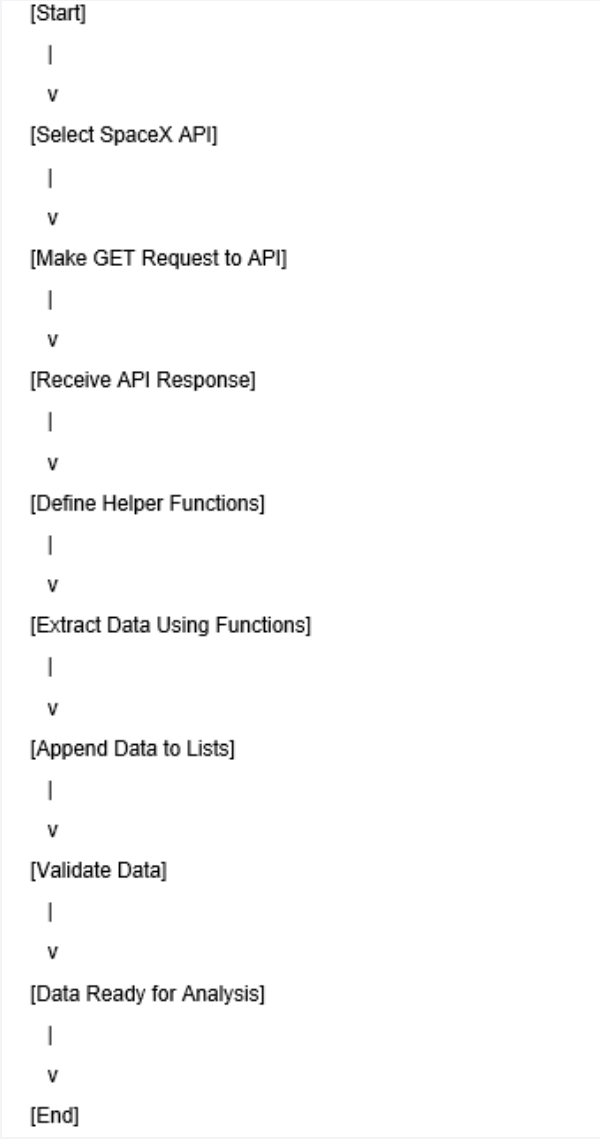
Core Data: Gathered details about the landing outcome, landing type, reuse count, and core serial number.

Data Storage: Appended the extracted data into lists for further analysis.

Data Validation: Ensured that the data collected was accurate and complete by checking for null values and handling exceptions.

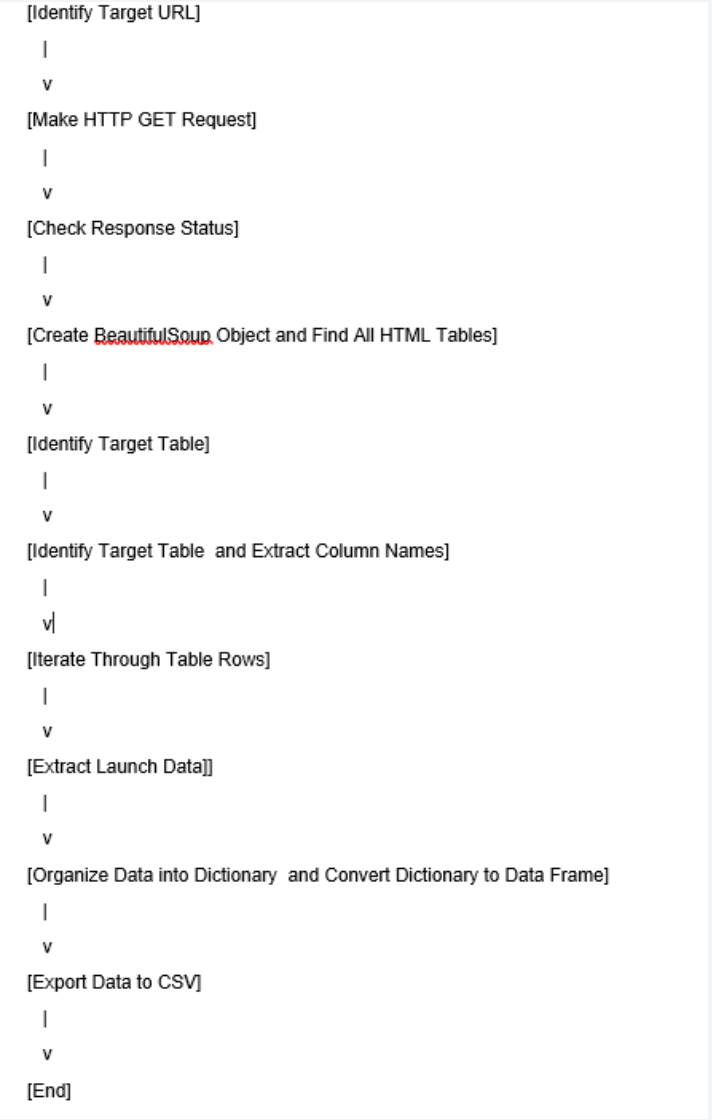
Data Collection – SpaceX API

- **Steps in the previous page**
- https://github.com/sampic78/coursera_ibm/blob/main/1jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- Identified the Wikipedia page containing the launch records: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches.
- Imported necessary libraries: requests for making HTTP requests and BeautifulSoup for parsing HTML content.
- Made a GET request to the target URL to retrieve the HTML content of the page.
- Checked the response status code to ensure the request was successful (status code 200).
- Created a BeautifulSoup object to parse the HTML content.
- Used BeautifulSoup to find all HTML tables on the page and identified the specific table containing the launch records.
- Extracted column names from the table header (<th> elements) for later use in data structuring.
- Iterated through the rows of the target table to extract relevant data for each launch,
- Organized the extracted data into a dictionary for easy conversion into a Pandas DataFrame.
- Converted the structured data dictionary into a Pandas DataFrame for analysis.
- Exported the DataFrame to a CSV file for future use.
- https://github.com/sampic78/coursera_ibm/blob/main/2jupyter-labs-webscraping.ipynb



Data Wrangling

- Loaded the SpaceX dataset from a CSV file using Pandas.
- Calculated the percentage of missing values in each attribute to assess data quality.
- Identified numerical and categorical columns in the dataset using the dtypes attribute.
- Analyzed the number of launches from each site using the value_counts() method on the LaunchSite column.
- Determined the number and occurrence of each orbit type using the value_counts() method on the Orbit column.
- Analyzed the mission outcomes by counting occurrences in the Outcome column and identifying successful and unsuccessful landings.
- Created a set of bad outcomes where the landing was unsuccessful (e.g., False ASDS, False Ocean, etc.).
- Generated a binary classification label (Class) based on the Outcome column, where 1 indicates a successful landing and 0 indicates an unsuccessful landing.
- Calculated the success rate of landings by taking the mean of the Class column.
- Exported the processed DataFrame to a new CSV file for future use.https://github.com/sampic78/coursera_ibm/blob/main/3labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- 1. Scatter Plot: Flight Number vs. Payload Mass. Purpose: To observe the relationship between the flight number (indicating the sequence of launches) and the payload mass. The hue was set to represent the landing outcome (success or failure). Insights: This chart helped identify trends indicating that as the flight number increases, the likelihood of successful landings also increases. It also showed that heavier payloads could still result in successful landings.
- 2. Scatter Plot: Flight Number vs. Launch Site. Purpose: To visualize the relationship between the flight number and the launch site, with the hue representing the landing outcome. Insights: This chart provided insights into how different launch sites performed over time, revealing patterns in successful and unsuccessful landings based on the site used.
- 3. Scatter Plot: Payload Mass vs. Launch Site. Purpose: To examine the relationship between payload mass and launch site, with the hue indicating the landing outcome. Insights: This chart illustrated that for the VAFB-SLC launch site, there were no launches with heavy payloads (greater than 10,000 kg), indicating a potential limitation of that site.
- 4. Bar Chart: Success Rate by Orbit Type. Purpose: To visualize the success rate of launches based on different orbit types. Insights: This chart allowed for a clear comparison of success rates across various orbit types, helping to identify which orbits had higher success rates and which were more challenging.
- 5. Scatter Plot: Flight Number vs. Orbit Type. Purpose: To explore the relationship between flight number and orbit type, with the hue representing the landing outcome. Insights: This chart helped to determine if there was any correlation between the number of flights and the success rate for different orbit types.
- 6. Scatter Plot: Payload Mass vs. Orbit Type. Purpose: To visualize the relationship between payload mass and orbit type, with the hue indicating the landing outcome. Insights: This chart revealed that heavier payloads had a higher success rate for certain orbits (like Polar, LEO, and ISS), while GTO showed mixed outcomes.
- 7. Line Chart: Success Rate Over the Years. Purpose: To visualize the trend of launch success rates over the years. Insights: This chart demonstrated that the success rate of launches has generally increased since 2013, indicating improvements in technology and processes over time.
- https://github.com/sampic78/coursera_ibm/blob/main/edadataviz.ipynb

EDA with SQL

- Task 1: Displayed unique launch sites in the SpaceX mission dataset.
- Task 2: Retrieved 5 records of launch sites that begin with 'CCA'.
- Task 3: Calculated the total payload mass carried by boosters launched by NASA (CRS).
- Task 4: Computed the average payload mass for the booster version F9 v1.1.
- Task 5: Listed the date of the first successful landing outcome on a ground pad.
- Task 6: Identified booster versions that successfully landed on a drone ship with payload mass between 4000 and 6000 kg.
- Task 7: Counted the total number of successful and failed mission outcomes.
- Task 8: Retrieved booster versions that carried the maximum payload mass using a subquery.
- Task 9: Listed records showing month names, failure landing outcomes on a drone ship, booster versions, and launch sites for the year 2015.
- Task 10: Ranked the count of landing outcomes between specific dates in descending order.
- https://github.com/sampic78/coursera_ibm/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

1. Markers, Description: Markers were created for each launch site and for each launch record. Purpose: For launch sites, markers indicate the geographical locations of the launch facilities. For launch records, markers represent the outcomes of each launch (successful or failed) at their respective coordinates. The color of the markers (green for successful launches and red for failed ones) provides an immediate visual cue about the success rates of launches at each site.

2. Circles, Description: Circles were drawn around each launch site. Purpose: The circles visually highlight the area around each launch site, making it easier to see the proximity of each site to other geographical features or infrastructure.
They also serve as a visual representation of the launch site's influence area, which can be important for safety and operational considerations.

3. Marker Clusters, Description: A MarkerCluster object was used to group markers that are close to each other.
Purpose:
This helps to manage the display of multiple markers that may overlap or be very close together, making the map cleaner and easier to read.
It allows users to zoom in and out without cluttering the map with too many individual markers.

4. Polylines, Description: Polylines were drawn to connect launch sites to points of interest, such as coastlines or cities.
Purpose:
These lines visually represent the distance between the launch sites and nearby features, helping to analyze the proximity of launch sites to critical infrastructure like coastlines, railways, and highways.
They provide a clear visual indication of the relationships between launch sites and their surroundings.

5. Mouse Position, Description: A MousePosition object was added to the map.
Purpose:
This feature allows users to see the latitude and longitude of their mouse cursor as they explore the map, which is useful for identifying specific points of interest and measuring distances to those points from the launch sites.

Summary of Purpose
The overall purpose of adding these objects to the Folium map is to create an interactive and informative visualization of SpaceX launch sites and their operational contexts. By using markers, circles, and lines, the map provides insights into the geographical distribution of launch sites, the success rates of launches, and the proximity of these sites to important geographical features. This visualization aids in understanding the operational landscape of space launches and can help in making data-driven decisions regarding future launch site locations and operations.

https://github.com/sampic78/coursera_ibm/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

1. Interactive Map, Description: A central feature of the dashboard would be the interactive map created using Folium, displaying launch sites, launch outcomes, and proximity to geographical features.
Purpose:
To provide a visual representation of the geographical distribution of launch sites and their success rates.
Users can interact with the map to explore different launch sites, view details about each launch, and analyze spatial relationships with nearby features.

2. Bar Charts for Success Rates, Description: Bar charts could be used to display the success rates of launches for each site.
Purpose:
To provide a quick visual comparison of the performance of different launch sites.
This helps stakeholders identify which sites have higher success rates and may warrant further investment or focus.

3. Pie Charts for Launch Outcomes, Description: Pie charts could illustrate the proportion of successful versus failed launches across all sites.
Purpose:
To give a clear visual representation of overall launch performance.
This can help in understanding the general trends in launch success and failure.

4. Line Graphs for Launch Trends Over Time, Description: Line graphs could show the number of launches over time, categorized by success and failure.
Purpose:
To analyze trends in launch activity and success rates over time.
This can help identify patterns, such as improvements in technology or operational practices leading to higher success rates.

5. Dropdowns or Filters, Description: Interactive dropdown menus or filters could allow users to select specific launch sites or time periods.
Purpose:
To enable users to customize their view of the data, focusing on specific areas of interest.
This enhances user engagement and allows for more tailored analysis.

6. Hover Tooltips, Description: Tooltips that appear when hovering over markers or data points on the map or graphs.
Purpose:
To provide additional information about specific launches or sites without cluttering the visual space.
This interaction enhances the user experience by making the data more accessible and informative.

Summary of Purpose
The inclusion of these plots, graphs, and interactions in a dashboard serves to create a comprehensive and user-friendly interface for analyzing SpaceX launch data. By combining visualizations with interactive elements, users can gain insights into launch performance, geographical relationships, and trends over time. This approach not only facilitates data exploration but also supports informed decision-making for stakeholders involved in space exploration and launch operations.

https://github.com/sampic78/coursera_ibm/blob/main/lab_jupyter_launch_site_location.ipynb

Predictive Analysis (Classification)

Model Development Process :

1. Building the Model
Data Collection: Loaded datasets containing features related to Falcon 9 launches.
Data Preprocessing:
Created a target variable Y from the Class column.
Standardized the feature set X using StandardScaler.
Split the data into training and testing sets using train_test_split.

2. Evaluating the Model
Model Selection: Implemented multiple classification algorithms:
Logistic Regression
Support Vector Machine (SVM)
Decision Tree
K-Nearest Neighbors (KNN)
Hyperparameter Tuning: Used GridSearchCV for each model to find the best hyperparameters through cross-validation.
Performance Metrics: Evaluated models based on accuracy and confusion matrix.

3. Improving the Model
Hyperparameter Optimization: Adjusted parameters for each model to enhance performance.
Model Comparison: Compared the accuracy of all models to identify strengths and weaknesses.

4. Finding the Best Performing Model
Final Evaluation: Calculated accuracy on the test set for each model.
Best Model Identification: Selected the model with the highest accuracy as the best performing model.

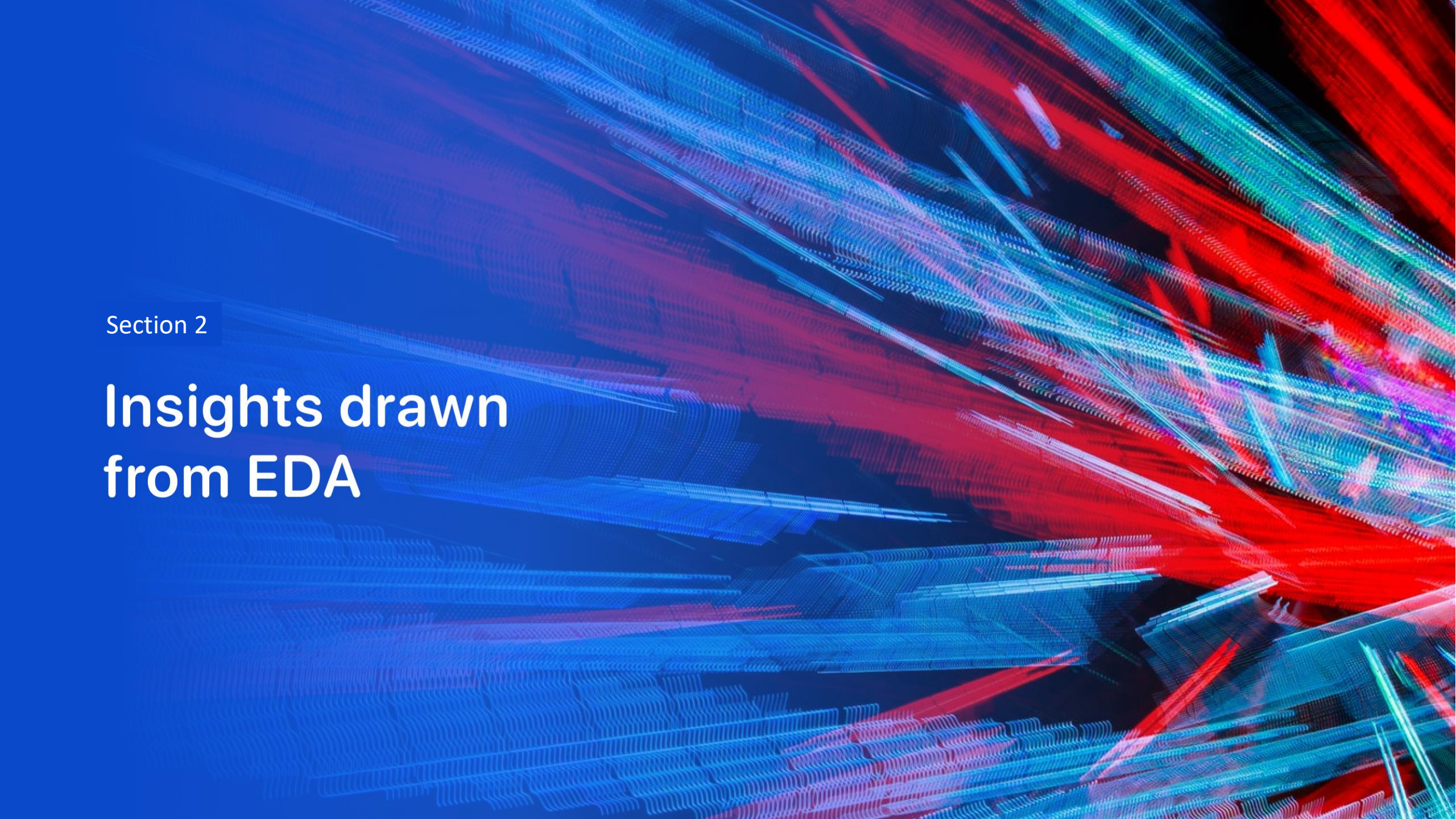
https://github.com/sampic78/coursera_ibm/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

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

Model performances:
Logistic Regression: 0.8333
Support Vector Machine: 0.8333
Decision Tree: 0.7222
K-Nearest Neighbors: 0.8333

Best performing model: Logistic Regression with an accuracy of 0.8333


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

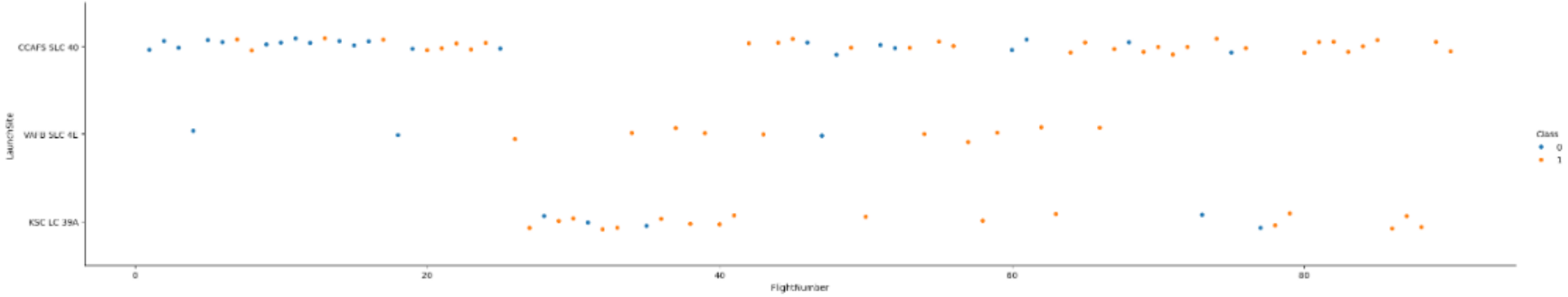
TASK 1: Visualize the relationship between Flight Number and Launch Site

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)
```

Out[5]: `<seaborn.axisgrid.FacetGrid at 0x6790370>`



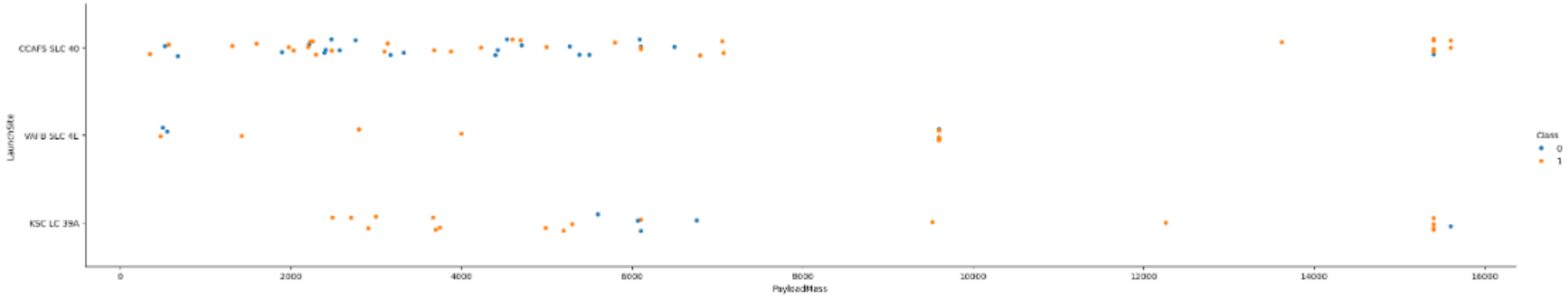
Payload vs. Launch Site

TASK 2: Visualize the relationship between Payload Mass and Launch Site

We also want to observe if there is any relationship between launch sites and their payload mass.

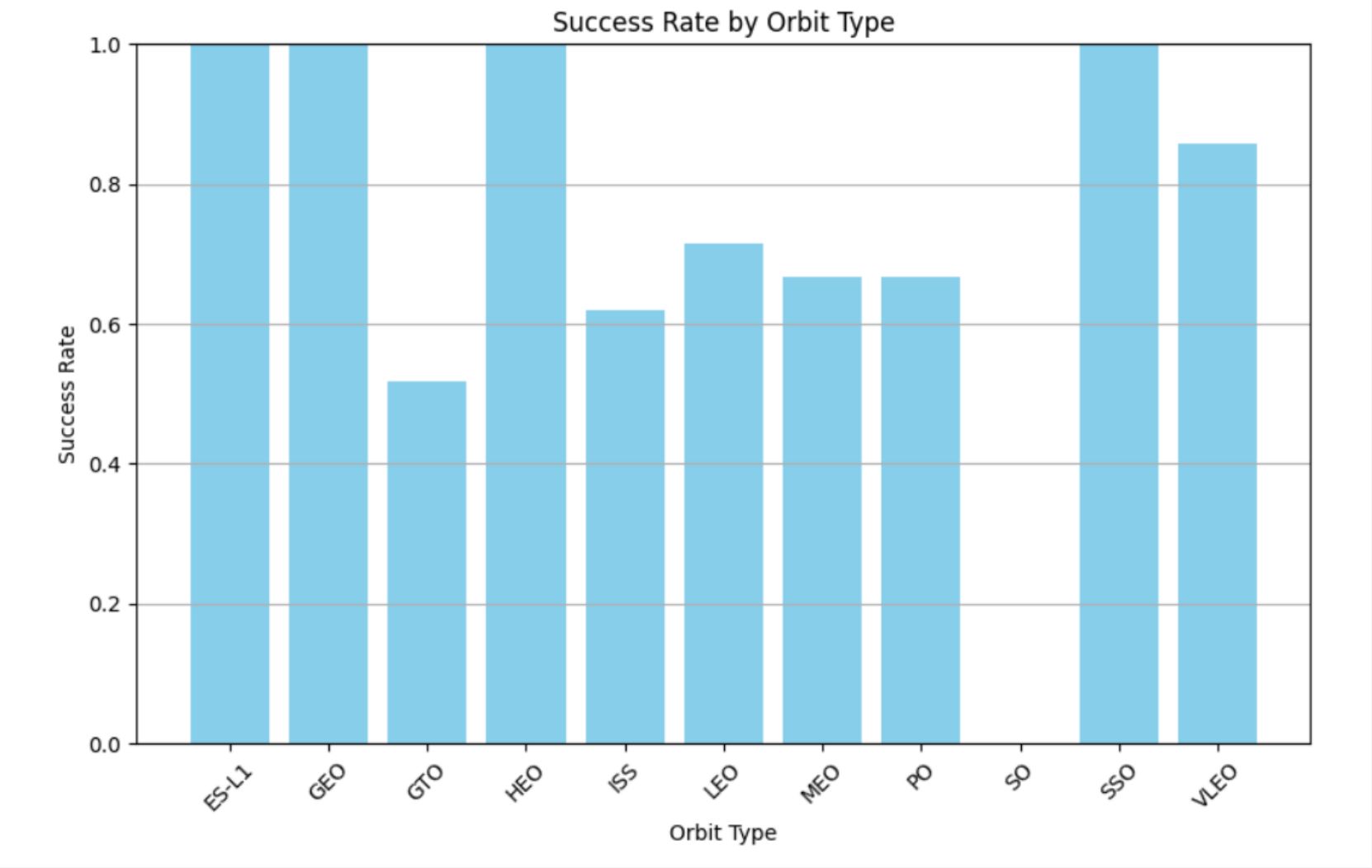
In [6]:
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the cla
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect=5)
```

Out[6]: <seaborn.axisgrid.FacetGrid at 0x7cc6d58>

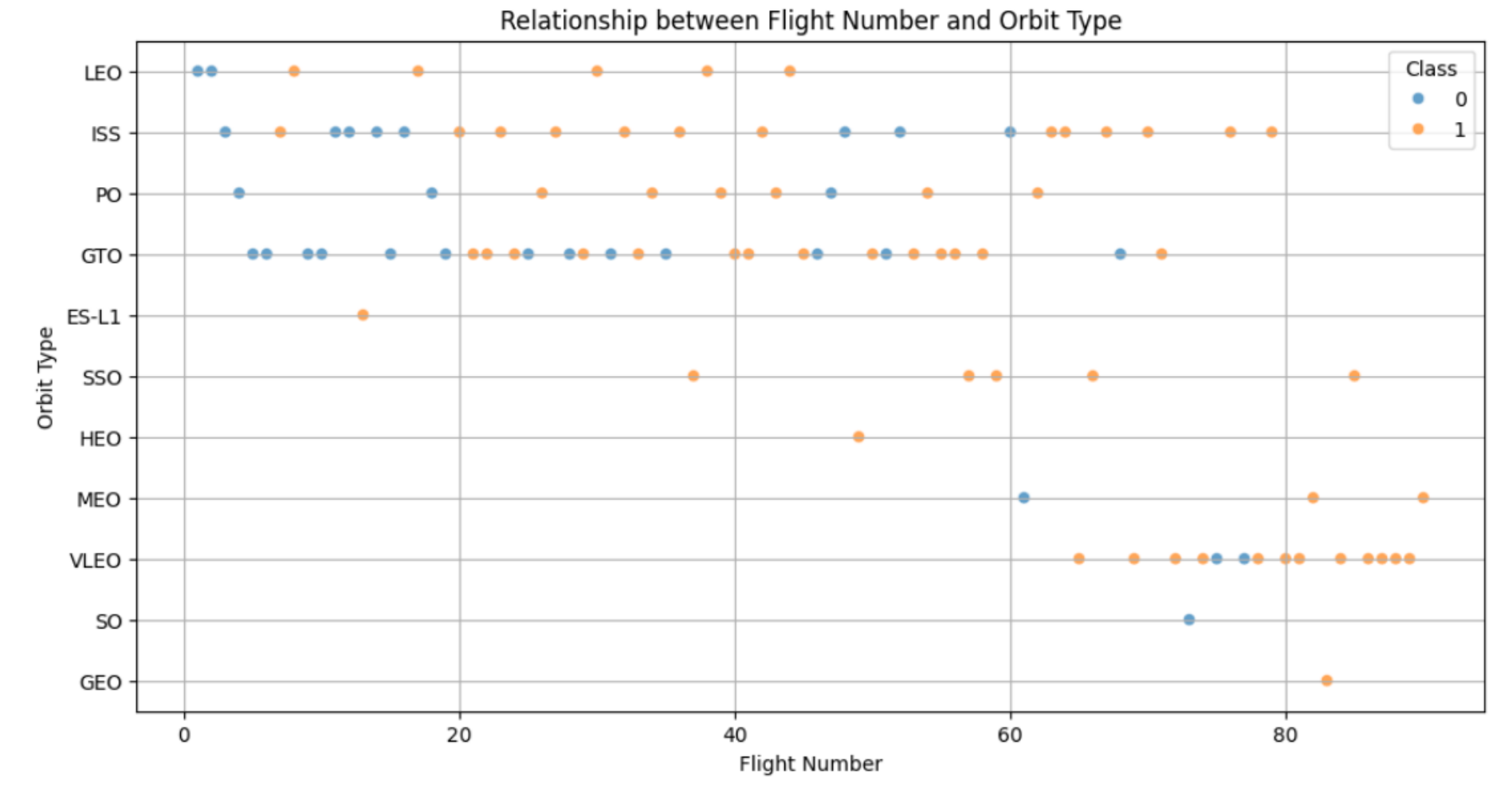


Now if you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

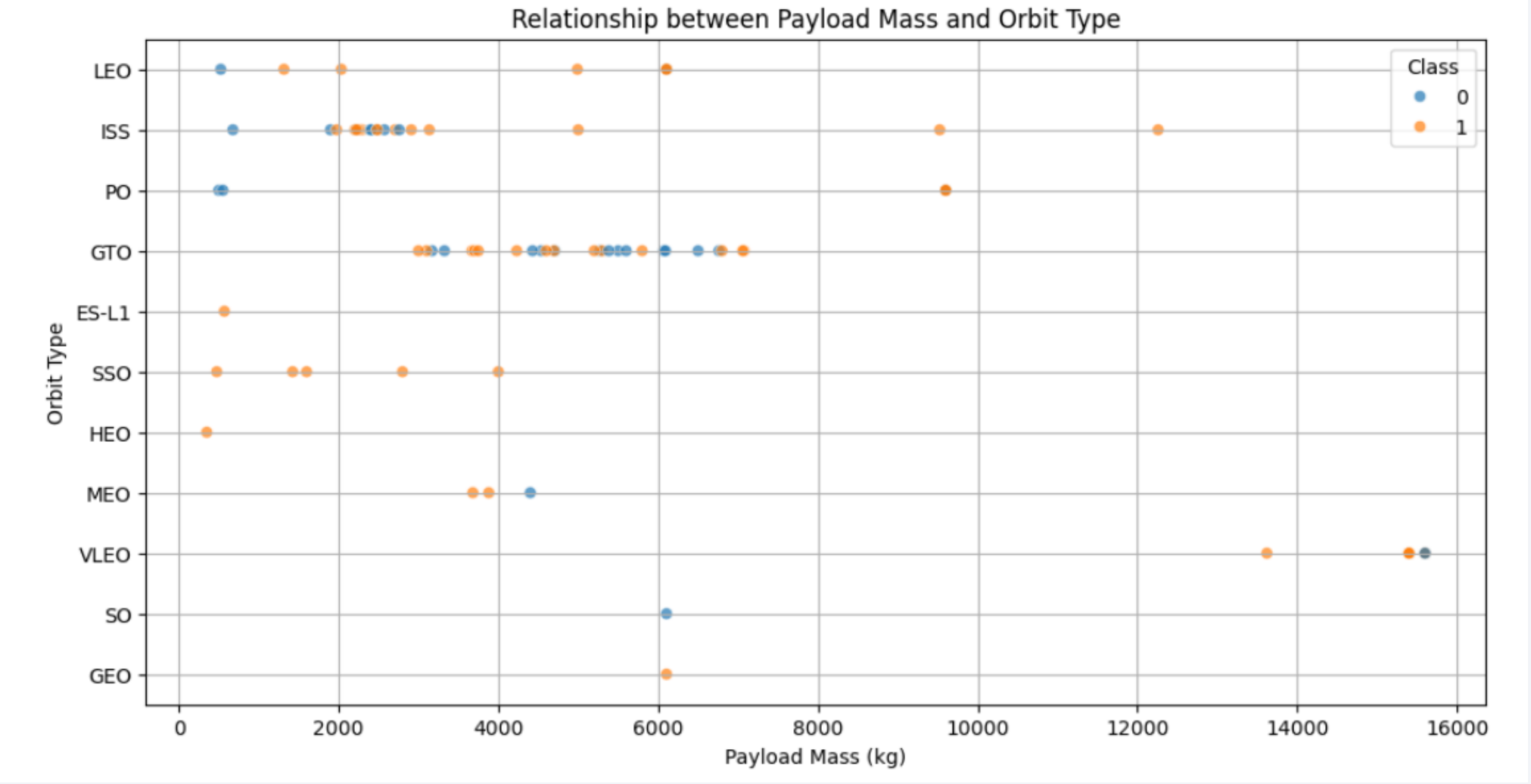
Success Rate vs. Orbit Type



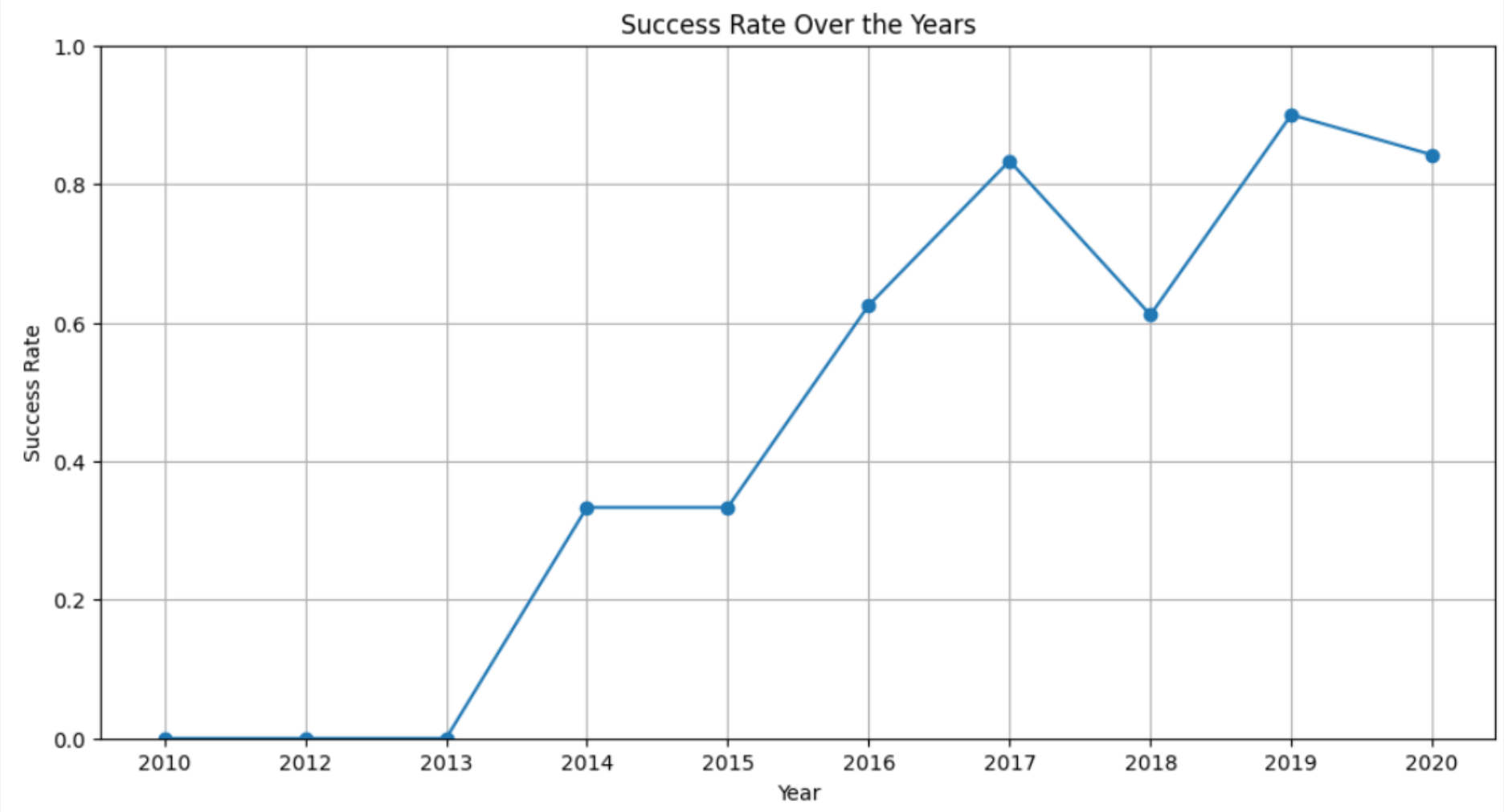
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

Task 1

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

In [10]:
```
%sql SELECT distinct Launch_Site from SPACEXTABLE
```

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

Out[10]:

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

Launch Site Names Begin with 'CCA'

Task 2

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

In [11]:
```
%sql select launch_site from SPACEXTABLE where substr(launch_site,1,3)='CCA' limit 5
```

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

Out[11]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

Task 3

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

In [12]:
```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer = 'NASA (CRS)'
```

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

Out[12]:

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

Task 4

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

In [21]:
```
%sql select distinct(Landing_Outcome) from SPACEXTABLE
```

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

Out[21]:

Landing_Outcome
Failure (parachute)
No attempt
Uncontrolled (ocean)
Controlled (ocean)
Failure (drone ship)
Precluded (drone ship)
Success (ground pad)
Success (drone ship)
Success
Failure
No attempt

In [18]:
```
%sql select AVG(PAYLOAD_MASS__KG_) AS average_payload_mass FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'
```

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

Out[18]:

average_payload_mass
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 [22]:
```
%sql SELECT Date FROM SPACEXTABLE WHERE Landing_Outcome LIKE 'Success (ground pad)' ORDER BY Date LIMIT 1;
```

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

Out[22]:

Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

In [23]:
```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome LIKE 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000
```

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

Out[23]:

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

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

In [24]:
```
%sql SELECT Mission_Outcome, COUNT(*) AS Total FROM SPACEXTABLE GROUP BY     Mission_Outcome
```

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

Out[24]:

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

Boosters Carried Maximum Payload

Task 8

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

In [25]:
```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
```

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

Out[25]:

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

Task 9

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.

[30]:
```
%%sql SELECT
    CASE
        WHEN substr(Date, 6, 2) = '01' THEN 'January'
        WHEN substr(Date, 6, 2) = '02' THEN 'February'
        WHEN substr(Date, 6, 2) = '03' THEN 'March'
        WHEN substr(Date, 6, 2) = '04' THEN 'April'
        WHEN substr(Date, 6, 2) = '05' THEN 'May'
        WHEN substr(Date, 6, 2) = '06' THEN 'June'
        WHEN substr(Date, 6, 2) = '07' THEN 'July'
        WHEN substr(Date, 6, 2) = '08' THEN 'August'
        WHEN substr(Date, 6, 2) = '09' THEN 'September'
        WHEN substr(Date, 6, 2) = '10' THEN 'October'
        WHEN substr(Date, 6, 2) = '11' THEN 'November'
        WHEN substr(Date, 6, 2) = '12' THEN 'December'
    END AS Month_Name,
    Booster_Version,
    Launch_Site
FROM
    SPACEXTABLE
WHERE
    Landing_Outcome LIKE 'False ASDS'
    AND substr(Date, 0, 5) = '2015';
```

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

[30]:

Month_Name	Booster_Version	Launch_Site

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

Task 10

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

In [31]:

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

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

Out[31]:

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

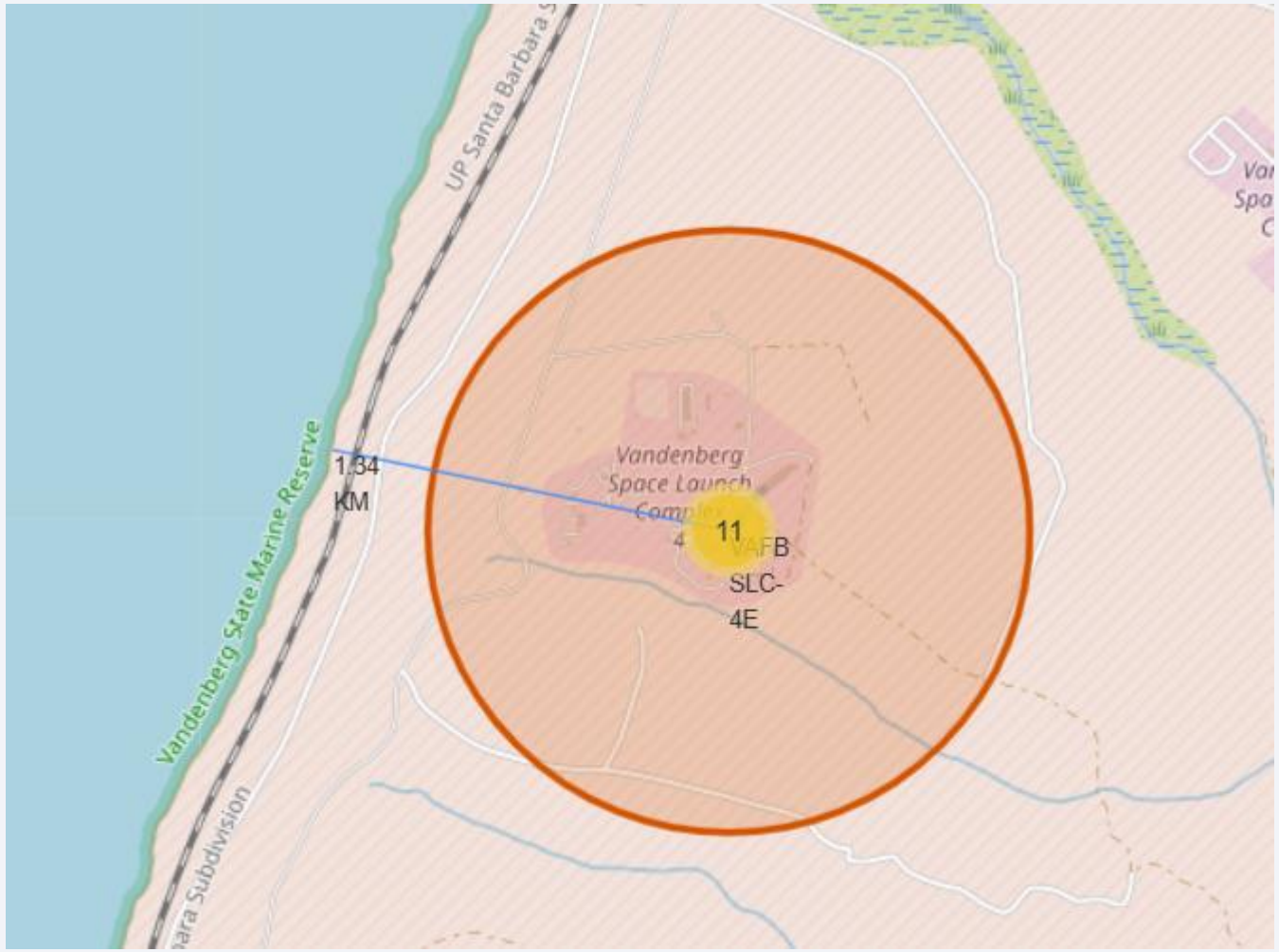
All Launch Sites

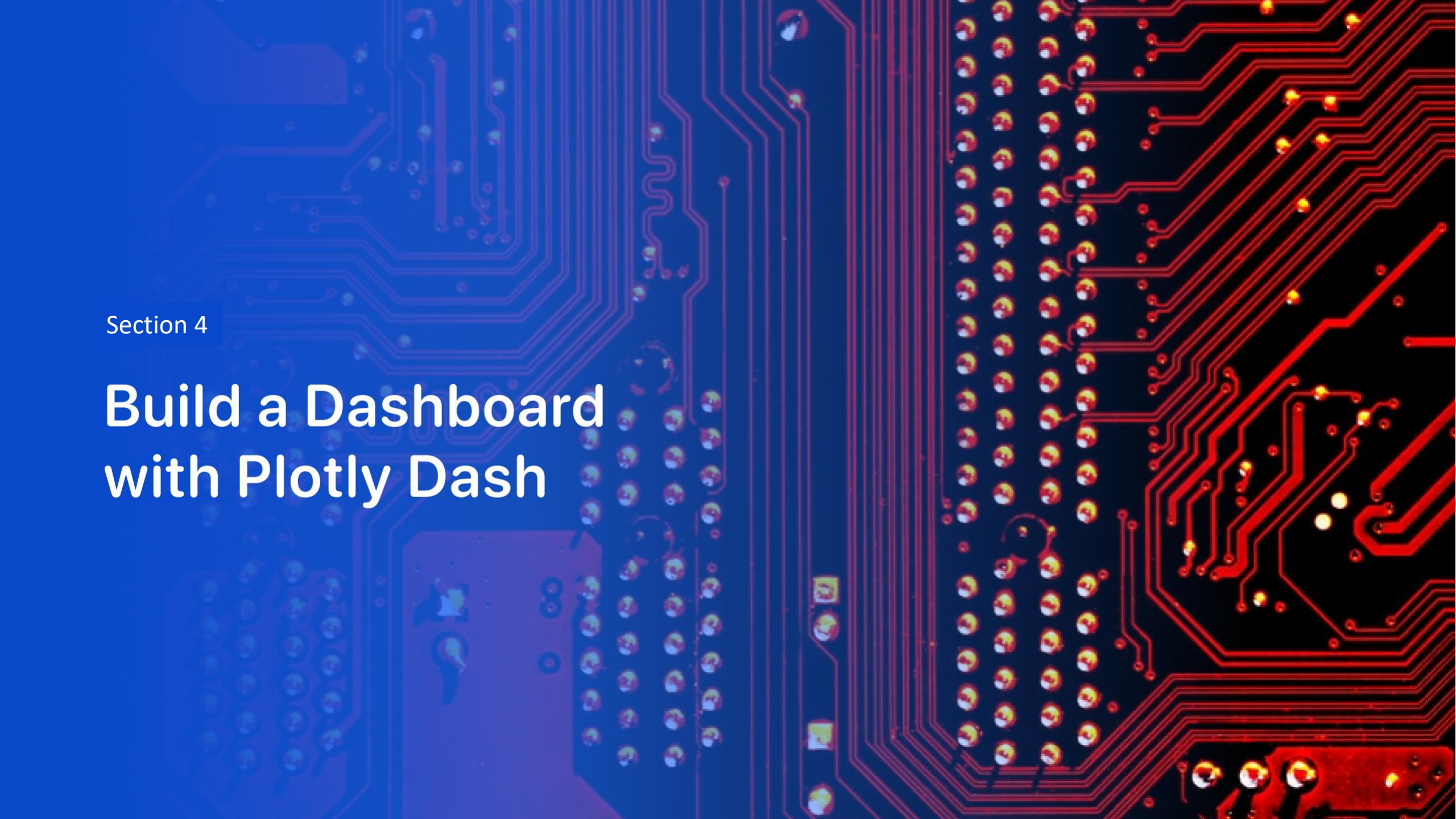
all launch sites' location markers on a global map

Launch outcomes

- Color-labeled launch outcomes on the map

Calculation of distance




Section 4
Build a Dashboard with Plotly Dash

Piechart success

- Replace <Dashboard screenshot 1> title with an appropriate title

- KSL was the most successful site

<Dashboard Screenshot 2>

- Replace <Dashboard screenshot 2> title with an appropriate title
- Show the scree launch success ratio
- Explain the imp

<Dashboard Screenshot 3>

- Replace <Dashboard screenshot 3> title with an appropriate title

- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

Section 5

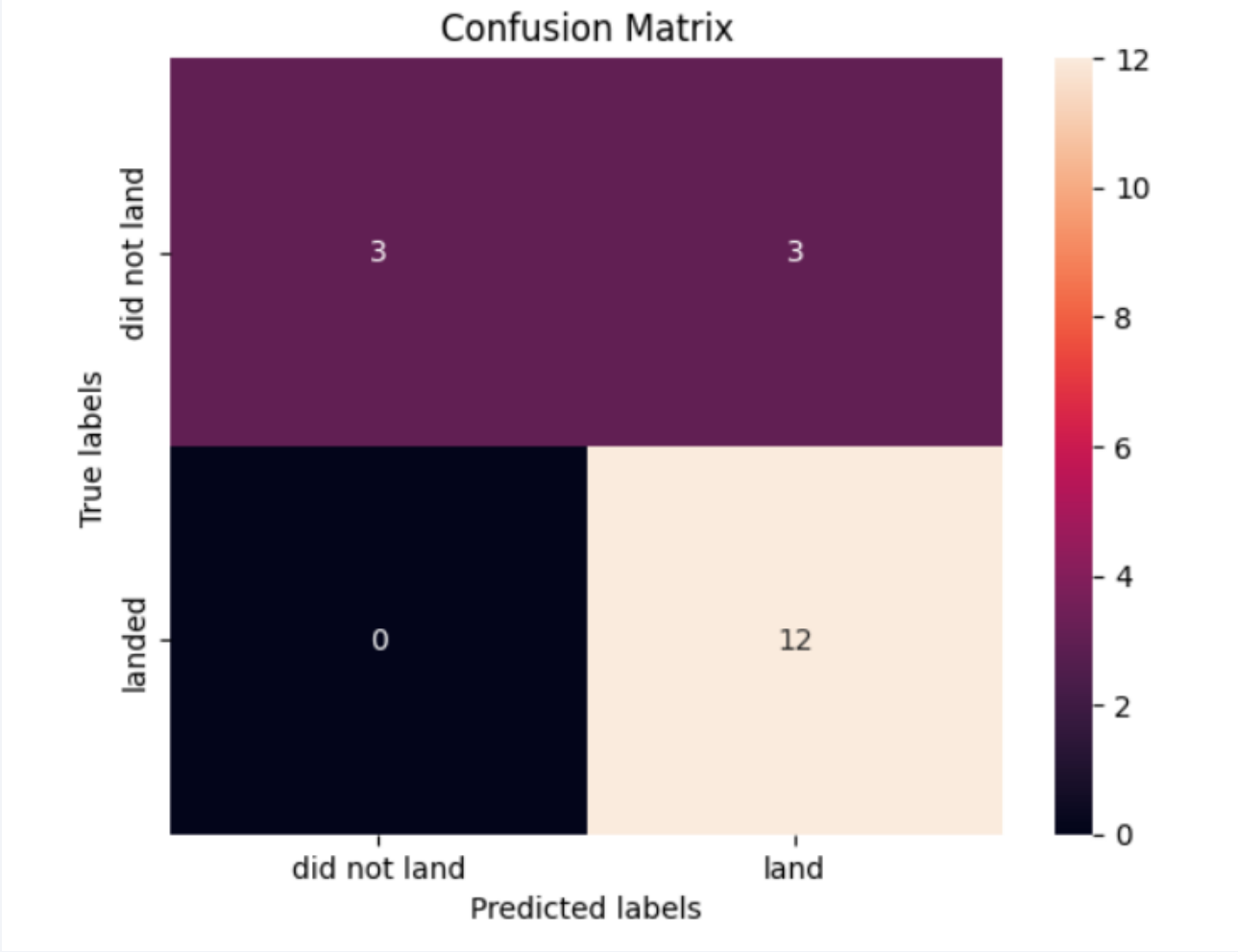
Predictive Analysis (Classification)

Classification Accuracy

- Model performances:
- Logistic Regression: 0.8333
- Support Vector Machine: 0.8333
- Decision Tree: 0.7222
- K-Nearest Neighbors: 0.8333

- Best performing model: Logistic Regression with an accuracy of 0.8333

Confusion Matrix



Conclusions

- Best performing model: Logistic Regression with an accuracy of 0.8333

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!