IBM Developer
SKILLS NETWORK

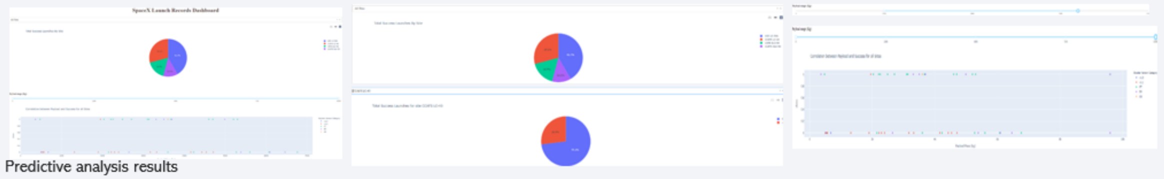
Winning Space Race with Data Science

Ravi Pohani
17-May-2024

Outline

Executive Summary

- Summary of methodologies
  - The project methodologies begin with the essential step of data collection, focusing on gathering comprehensive and relevant data from diverse sources. Following data collection, the emphasis shifts to enhancing data quality through meticulous data wrangling techniques.
  - Once the raw data is processed, the exploration phase commences, guided by our expertise in analyzing real-world datasets collaboratively. Participants will have the opportunity to hone their SQL skills by querying the data and uncovering valuable insights.
  - Further exploration involves basic statistical analysis and data visualization, enabling participants to discern relationships between variables directly. Delving deeper, the data will be segmented into groups based on categorical variables or factors, facilitating detailed examination.
  - Lastly iterative process of building, evaluating, and refining predictive models to unearth more profound insights from the data takes place.
- Summary of all results
  - Exploratory data analysis results
    - Different launch sites have different success rates
    - orbits have high sucess rate (ES, GEO, SSO, HEO)
    - LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit
    - With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.
  - Interactive analytics demo in screenshots



  - Predictive analysis results
    - Logistic Regression had the highest accuracy of ~83%

Introduction

- Predicting Falcon 9 First Stage Landing: Cost Optimization for Rocket Launch Bidding
- In this project, we aim to predict the success of Falcon 9 first stage landings, a crucial factor in optimizing the cost of rocket launches. SpaceX, renowned for its innovative approach, advertises Falcon 9 rocket launches on its website at a cost of 62 million dollars. In stark contrast, other providers charge upward of 165 million dollars for each launch. The significant cost differential arises from SpaceX's pioneering reusability of the first stage, a feature that dramatically reduces launch costs.
- Our project seeks to leverage predictive modeling techniques to determine the likelihood of a successful first stage landing. By accurately predicting whether the first stage will land successfully, we can effectively estimate the cost of a launch. This predictive insight holds immense value, particularly in scenarios where alternate companies seek to bid against SpaceX for a rocket launch contract.
- Through our analysis, we aim to provide actionable insights that enable stakeholders to make informed decisions when bidding for rocket launch contracts. Ultimately, our project aims to contribute to cost optimization strategies in the aerospace industry, fostering competition and innovation in the realm of space exploration.
- Problems you want to find answers
- Determine relationship between launch success and different variables
- Determine the cost of a launch
- Analyze launch records interactively
- Determine if the first stage of Falcon 9 will land successfully

Section 1

Methodology

Methodology

Executive Summary

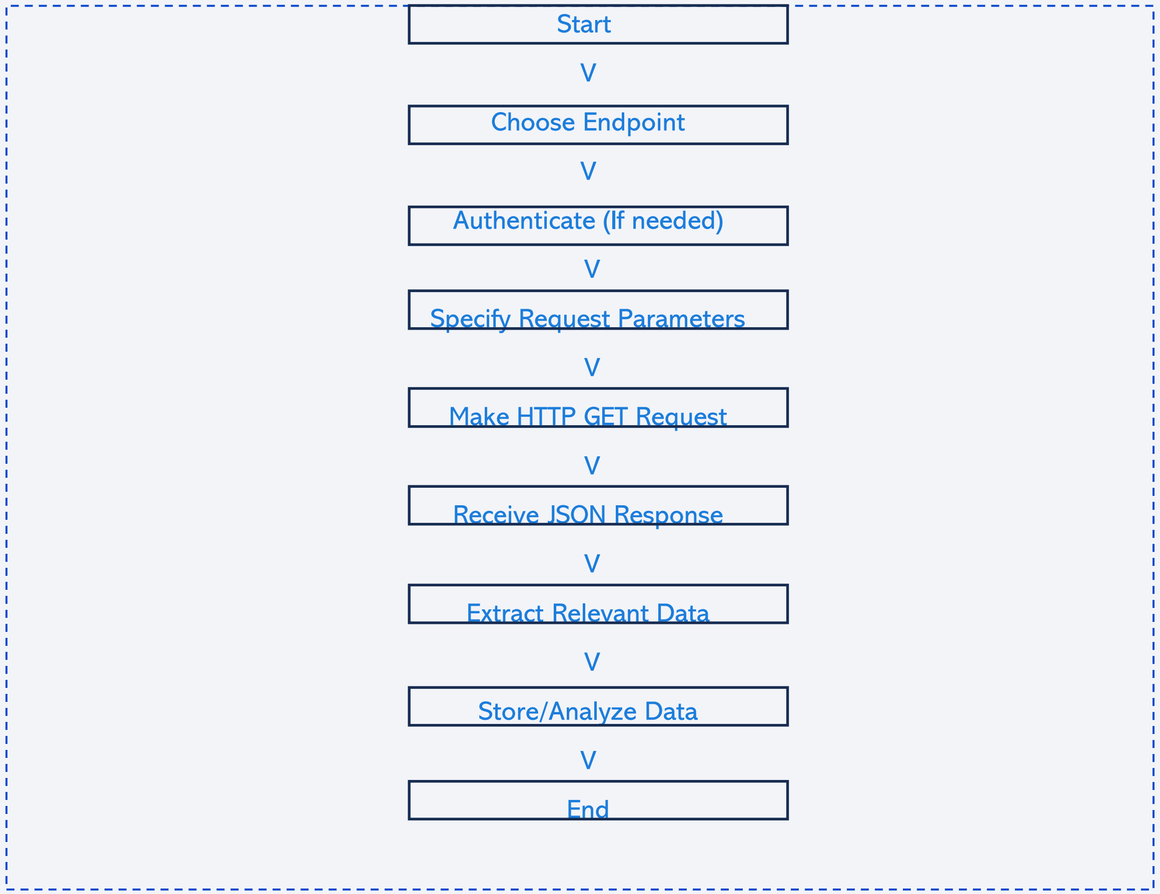
- Data collection methodology:
  - Develop Python code to manipulate data in a Pandas data frame
  - Convert a JSON file into a Create a Python Pandas data frame by converting a JSON file
- Perform data wrangling
  - Used a RESTful API and web scraping. Convert the data into a dataframe and then perform some data wrangling.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Use machine learning to determine if the first stage of Falcon 9 will land successfully. Split your data into training data and test data to find the best Hyperparameter for SVM, Classification Trees, and Logistic Regression. Then find the method that performs best using test data.

Data Collection

- **Import Libraries and Define Auxiliary Functions**:
  - Start by importing necessary libraries and defining auxiliary functions for data collection and processing.
- **Request Rocket Launch Data from SpaceX API**:
  - Utilize the SpaceX API to request rocket launch data with a specific URL.
- **Parse and Decode Response Content**:
  - Parse the SpaceX launch data using a GET request and decode the response content as JSON.
  - Convert the JSON response into a Pandas dataframe using `.json_normalize()`.
- **Retrieve Information About Launches**:
  - Use the obtained launch IDs to retrieve detailed information about launches, focusing on rocket, payloads, launchpad, and cores.
- **Construct Dataset**:
  - Store the retrieved information in lists and create a new dataframe.
  - Combine relevant columns into a dictionary and create a Pandas dataframe from the dictionary.
- **Data Filtering and Imputation**:
  - Filter the dataframe to include only Falcon 9 launches.
  - Calculate the mean for PayloadMass and replace NaN values in the data with the calculated mean using `.mean()` and `.replace()` functions.

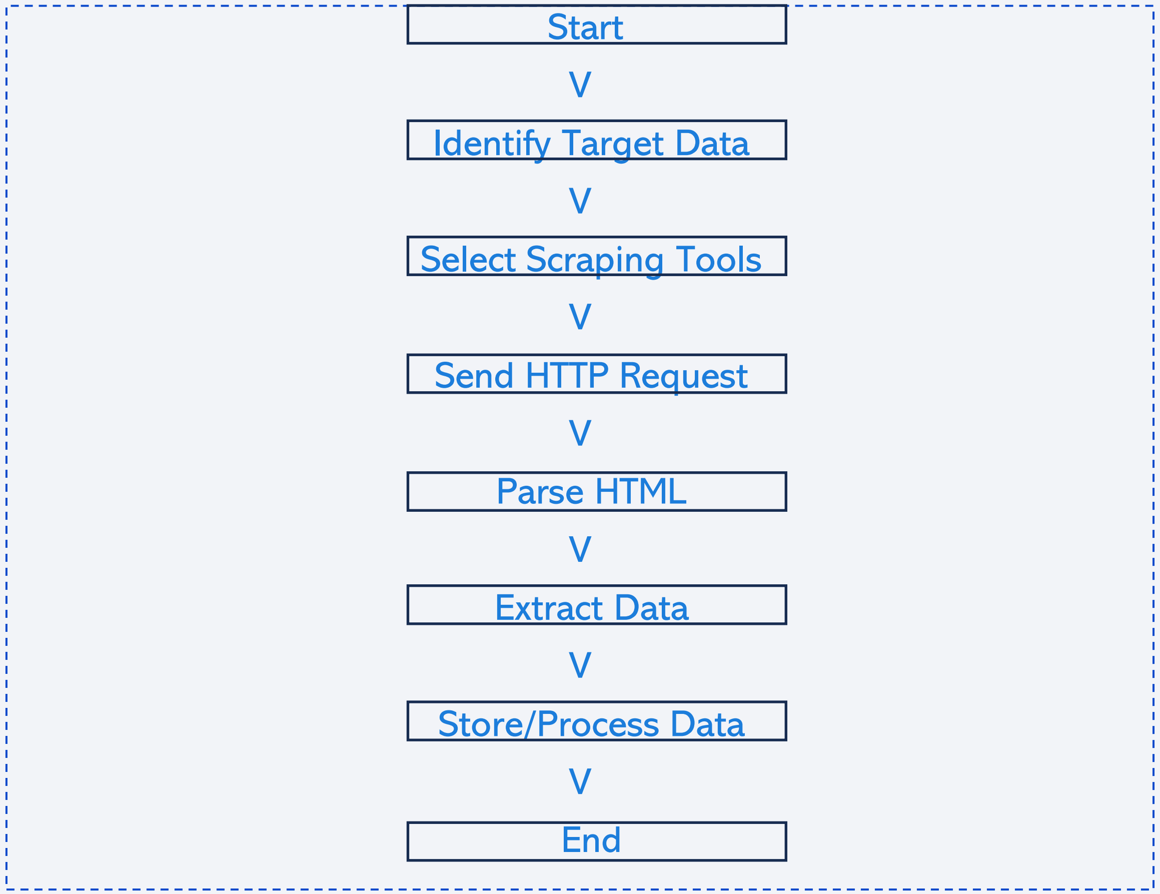
Data Collection – SpaceX API

- Endpoint Selection: Choose the appropriate SpaceX REST API endpoint based on the data you want to collect. For example:
  - /launches: Retrieve data about all launches.
  - /rockets: Get information about all rockets.
  - /payloads: Fetch details about payloads.
- Authentication: If required, authenticate your requests using an API key.
- Request Parameters: Specify any parameters such as launch year, rocket ID, etc., to filter the data.
- HTTP Request: Make a GET request to the selected endpoint with the specified parameters.
- Response Handling: Receive the JSON response from the API.
- Data Extraction: Extract relevant information from the JSON response.
- Data Storage/Analysis: Store the extracted data in a database or perform analysis as required.
- https://github.com/RaviPohani/my-submissions/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- Identify Target Data: Determine the specific data from SpaceX's website that you want to scrape, such as launch information, rocket details, etc.
- Select Scraping Tools: Choose the appropriate web scraping tools or libraries, such as BeautifulSoup in Python
- Send HTTP Request: Use the chosen tool to send an HTTP request to the Wiki page to fetch the HTML content of the page.
- Parse HTML: Parse the HTML content received from the website to extract the desired data using techniques like CSS selectors or XPath.
- Extract Data: Extract the relevant data from the parsed HTML, including launch dates, rocket names, mission details, etc.
- Store/Process Data: Store the extracted data in a suitable format such as CSV, JSON, or a database. Optionally, process the data for analysis or visualization.
- https://github.com/RaviPohani/my-submissions/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Wrangling

- Data Cleaning:
  - Handling Missing Values: Identify and handle missing values by either imputing them with appropriate values or removing them if they are insignificant.
  - Removing Duplicates: Check for and remove any duplicate records to ensure data integrity.
  - Standardizing Formats: Standardize formats of data fields such as dates, times, and categorical variables for consistency.
- Data Transformation:
  - Feature Engineering: Create new features that might provide valuable insights, such as calculating the duration between launch dates or categorizing missions based on their objectives.
  - Normalization/Scaling: Scale numerical features to a similar range to prevent certain features from dominating others during analysis.
  - One-Hot Encoding: Convert categorical variables into binary vectors to make them suitable for machine learning algorithms.
- Data Integration:
  - Combining Data: If data is collected from multiple sources, integrate them into a single dataset for comprehensive analysis.
  - Resolving Inconsistencies: Resolve any inconsistencies between datasets, such as discrepancies in naming conventions or units of measurement.
- Data Reduction:
  - Dimensionality Reduction: If the dataset contains a large number of features, apply techniques such as Principal Component Analysis (PCA) to reduce dimensionality while retaining important information.
  - Sampling: If the dataset is excessively large, consider sampling techniques to reduce computational overhead while maintaining the representativeness of the data.
- Handling Outliers:
  - Identifying Outliers: Detect outliers using statistical methods or visualization techniques.
  - Handling Outliers: Decide whether to remove outliers, cap them, or treat them separately based on their impact on the analysis.
- Data Validation:
  - Cross-Validation: Validate the quality of the data by comparing it against known standards or external sources.
  - Error Checking: Perform error checking routines to ensure data accuracy and integrity throughout the wrangling process.and flowcharts
- https://github.com/RaviPohani/my-submissions/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Plot out the FlightNumber vs. PayloadMassand overlay the outcome of the launch. We see that as the flight number increases, the first stage is more likely to land successfully. The payload mass is also important; it seems the more massive the payload, the less likely the first stage will return.

- Different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.

- Then 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’

- Payload 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).

- Plotted bar chart try to find which orbits have high sucess rate. ploted bar chart try to find which orbits have high sucess rate (ES, GEO, SSO, HEO).

- Each orbit, we want to see if there is any relationship between FlightNumber and Orbit type. LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

- Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type. With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

- Observe that the success rate since 2013 kept increasing till 2020

- https://github.com/RaviPohani/my-submissions/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite%20(1).ipynb

EDA with SQL

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was achieved
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcome
- List the names of the booster_versions which have carried the maximum payload mass.
- 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.
- 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.
- https://github.com/RaviPohani/my-submissions/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Create a folium Map object, with an initial center location to be NASA Johnson Space Center at Houston, Texas.
- Circled Huston’s Jackson Space Center
- Markers were added for all launch records. If a launch was successful (class=1), then we use a green marker and if a launch was failed, we use a red marker (class=0)
- Each launch result in spacex_df data frame, we added a folium.Marker to marker_cluster. Marked the launchsites with circles
- Marked distances on railway lines with lines
- Marked hgihway symbols with lines
- https://github.com/RaviPohani/my-submissions/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

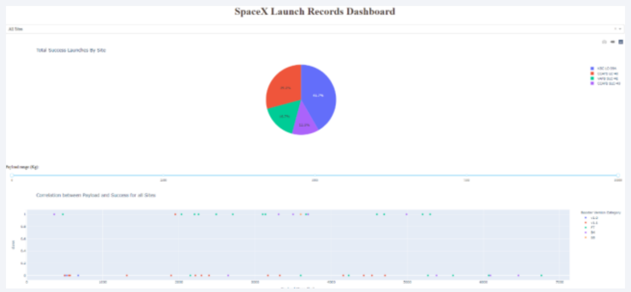
- completed and added a launch site drop-down input component
- completed and added a callback function to render the required success outcome pie chart
- completed and added a range slider to choose payload
- completed and added callback function to render the payload-outcome scatter plot
- https://github.com/RaviPohani/my-submissions/blob/main/plotlydash.py

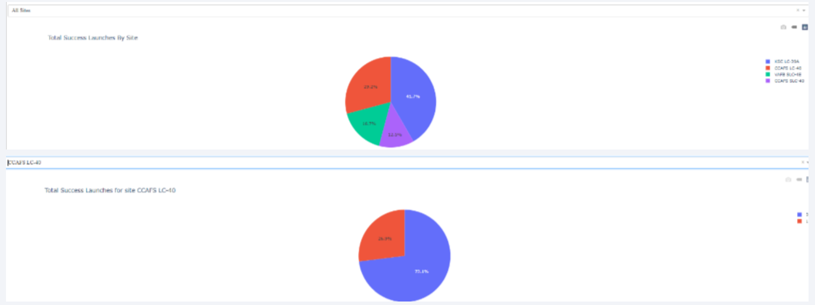
Predictive Analysis (Classification)

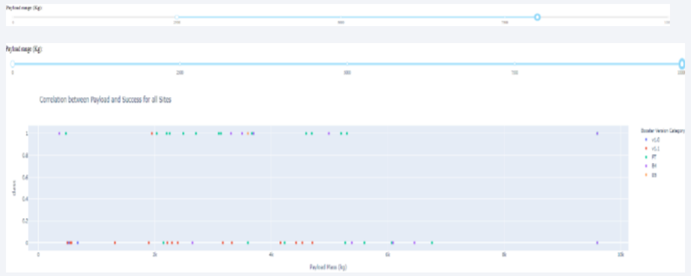
- Standardize the data
- Split the data into train and test set
- For each model, found the best hyperparameters
- Calculate the accuracy of the test data

Results

- Exploratory data analysis results
  - Different launch sites have different success rates
  - orbits have high sucess rate (ES, GEO, SSO, HEO)
  - LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit
  - With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.
- Interactive analytics demo in screenshots


SpaceX Launch Records Dashboard
Total Success Launches By Site
Correlation between Payload and Success for all Sites


Total Success Launches By Site
CCAFS LC-40
Total Success Launches for site CCAFS LC-40



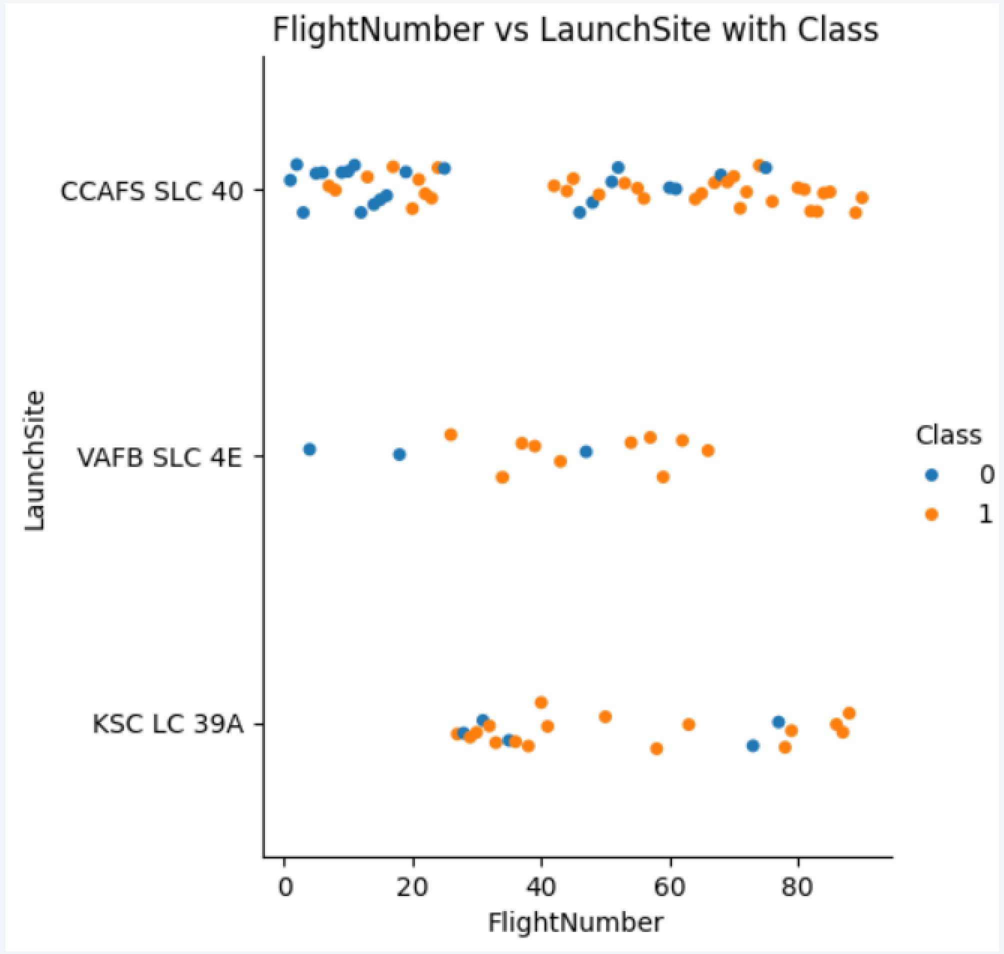
- Predictive analysis results
  - Logistic Regression had the highest accuracy of ~83%

Section 2

Insights drawn from EDA

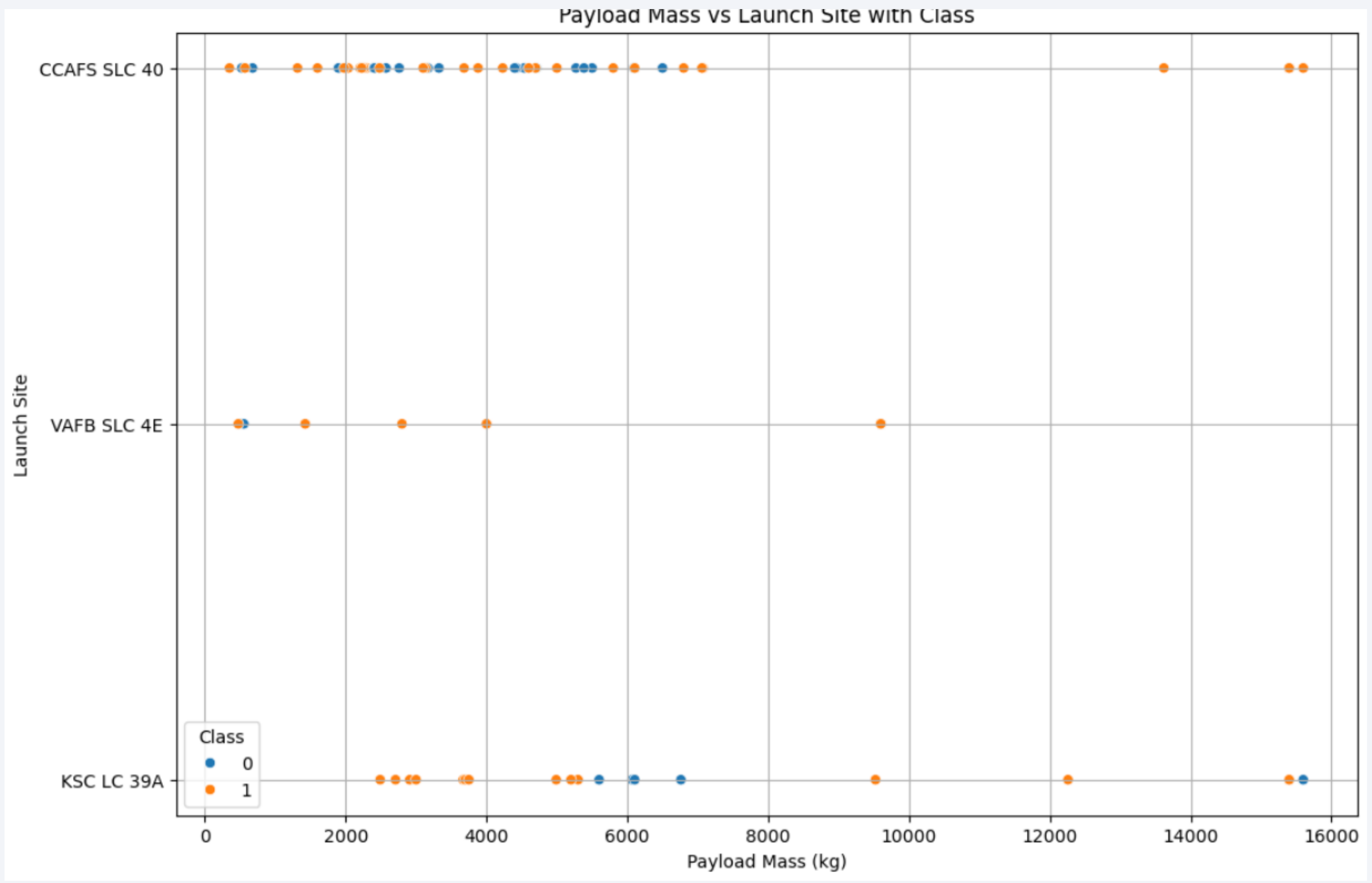
Flight Number vs. Launch Site

- The Y-axis represents the Launch Sites, which include CCAFS SLC 40, VAFB SLC 4E, and KSC LC 39A.
- The X-axis represents the Flight Number, ranging from 0 to 80.
- Data points are colored in two hues, differentiating between the classes. Class 0 is colored blue and represents failures, while Class 1 is colored orange and represents successes.
- CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.



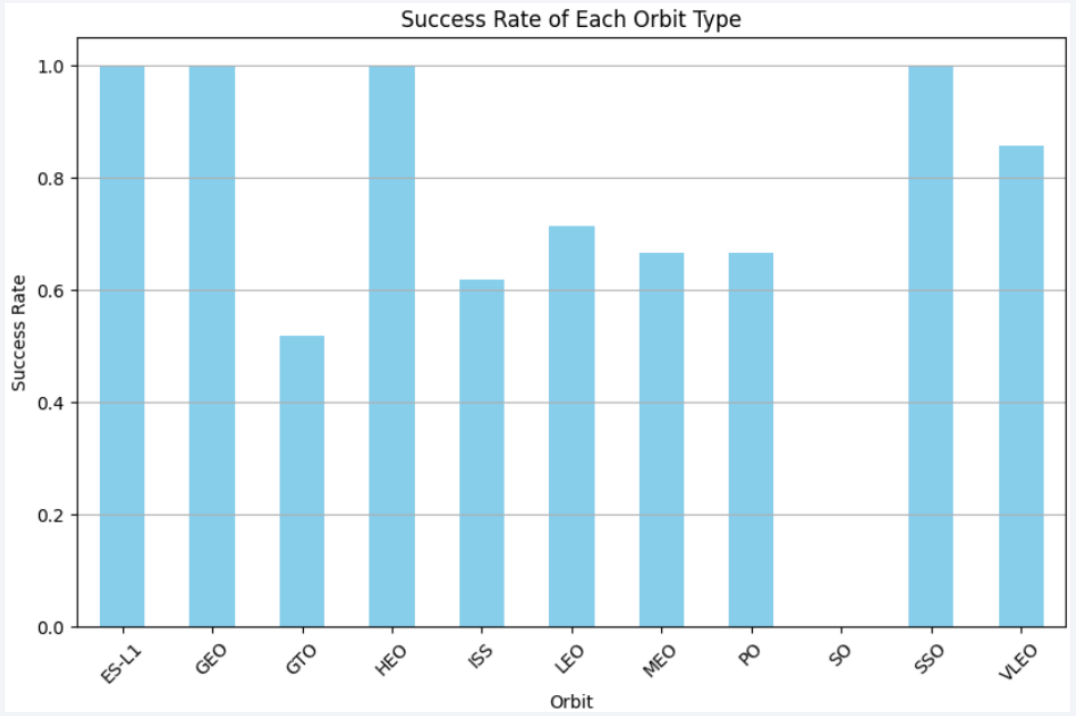
Payload vs. Launch Site

- Payload 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)
- Most sites have payloads b/w 0 to 80000



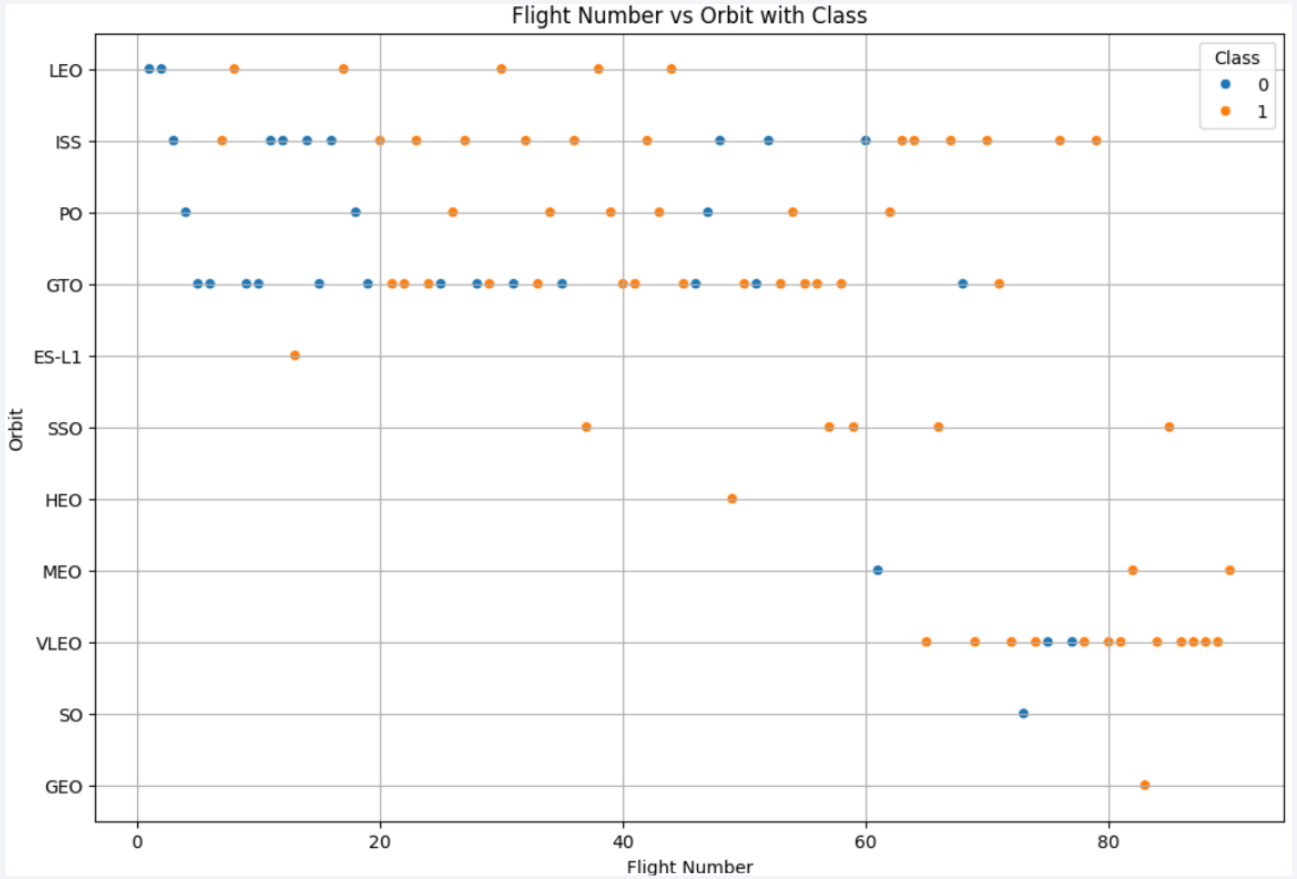
Success Rate vs. Orbit Type

- ES, GEO, HEO, SSO are orbits with top notch success rates
- While others have success rates ranging from 0.5 to 0.85



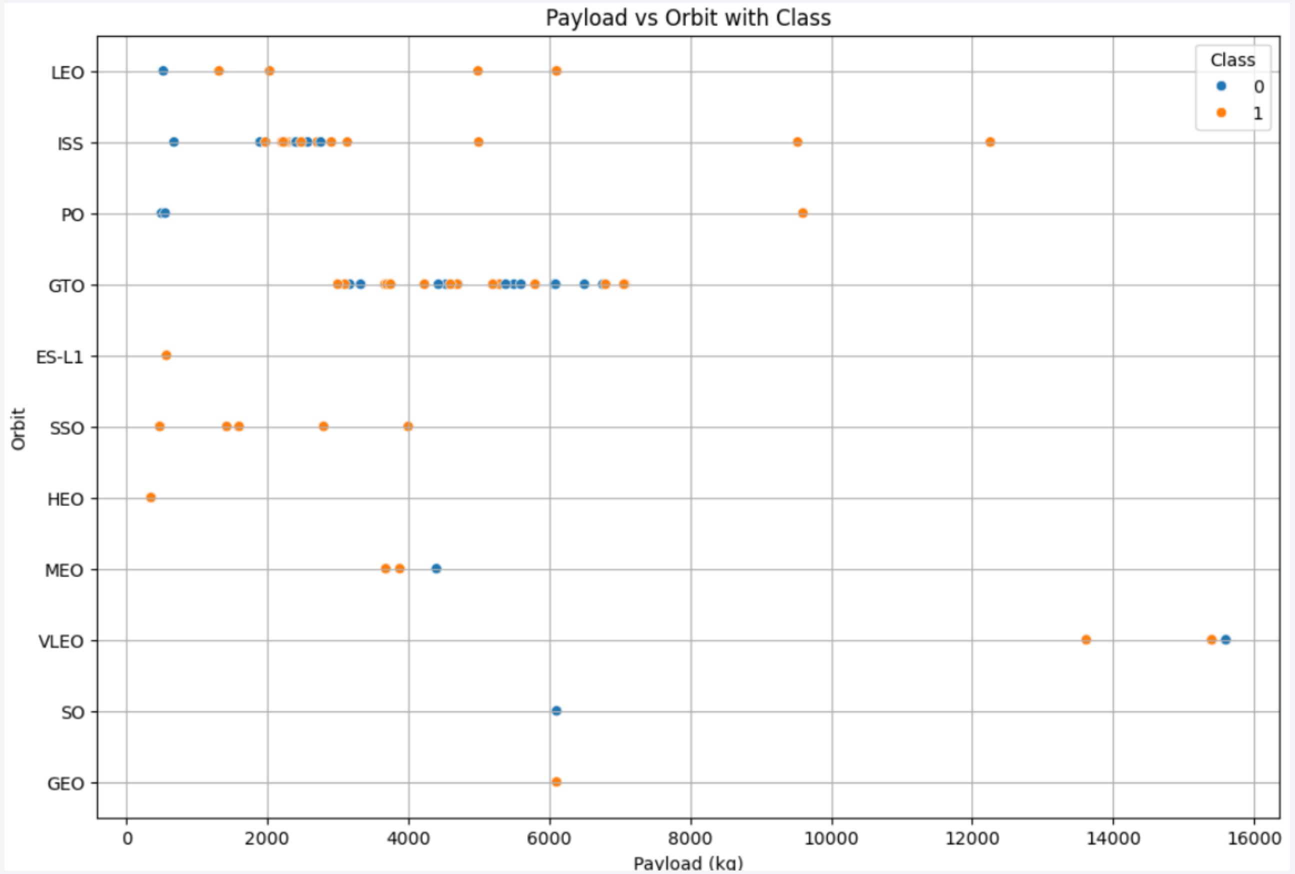
Flight Number vs. Orbit Type

- Each orbit, we want to see if there is any relationship between FlightNumber and Orbit type. LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit



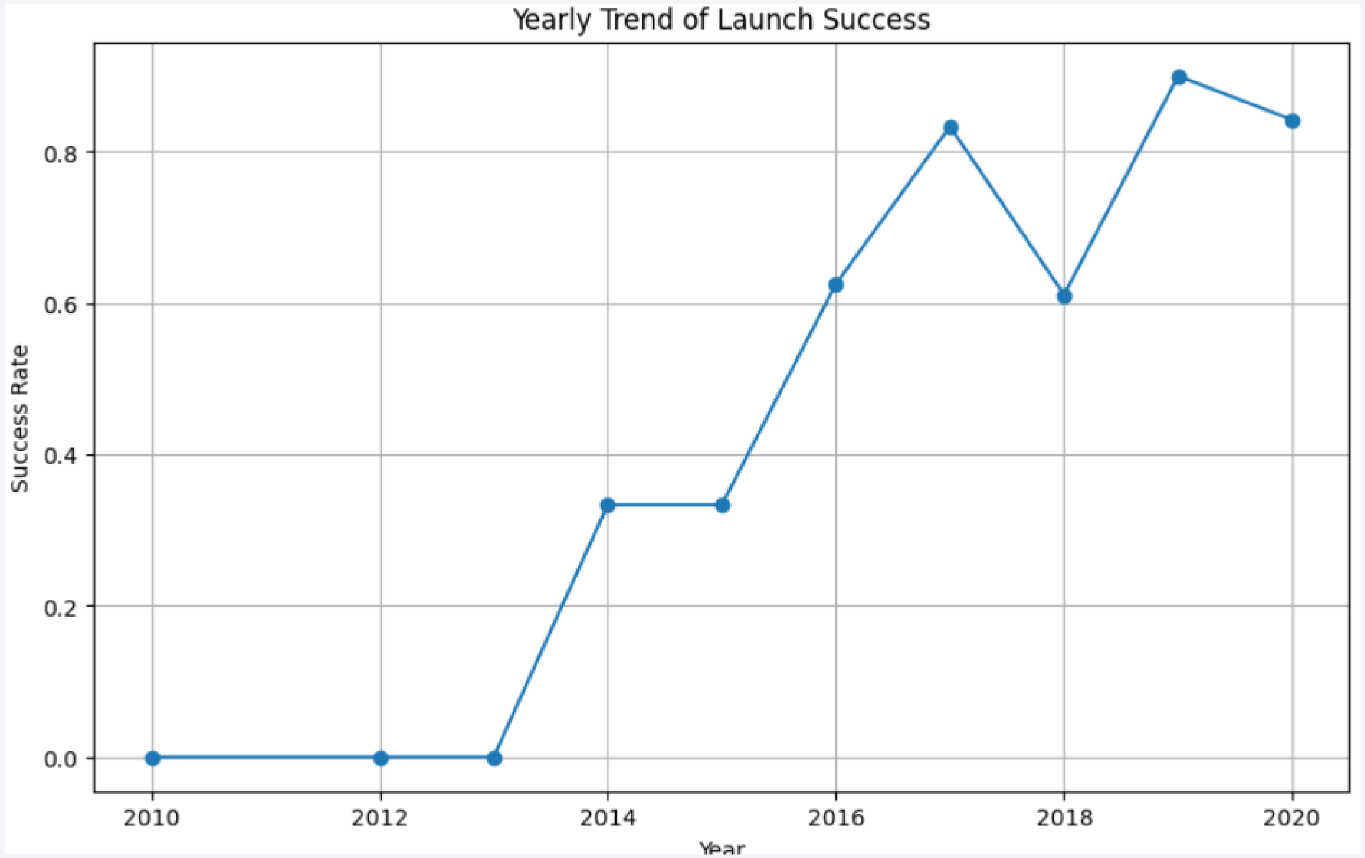
Payload vs. Orbit Type

- Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type. With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.



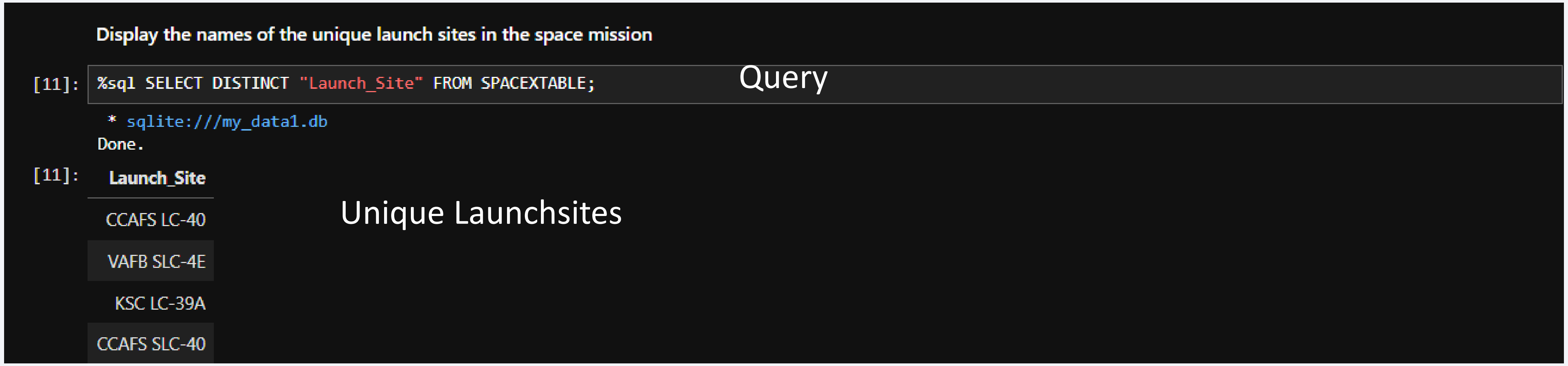
Launch Success Yearly Trend

- Observe that the success rate since 2013 kept increasing till 2020



All Launch Site Names

Use of 'Distinct' query over Launch_Sites column in SPACEXTABLE db



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

```
[11]: %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

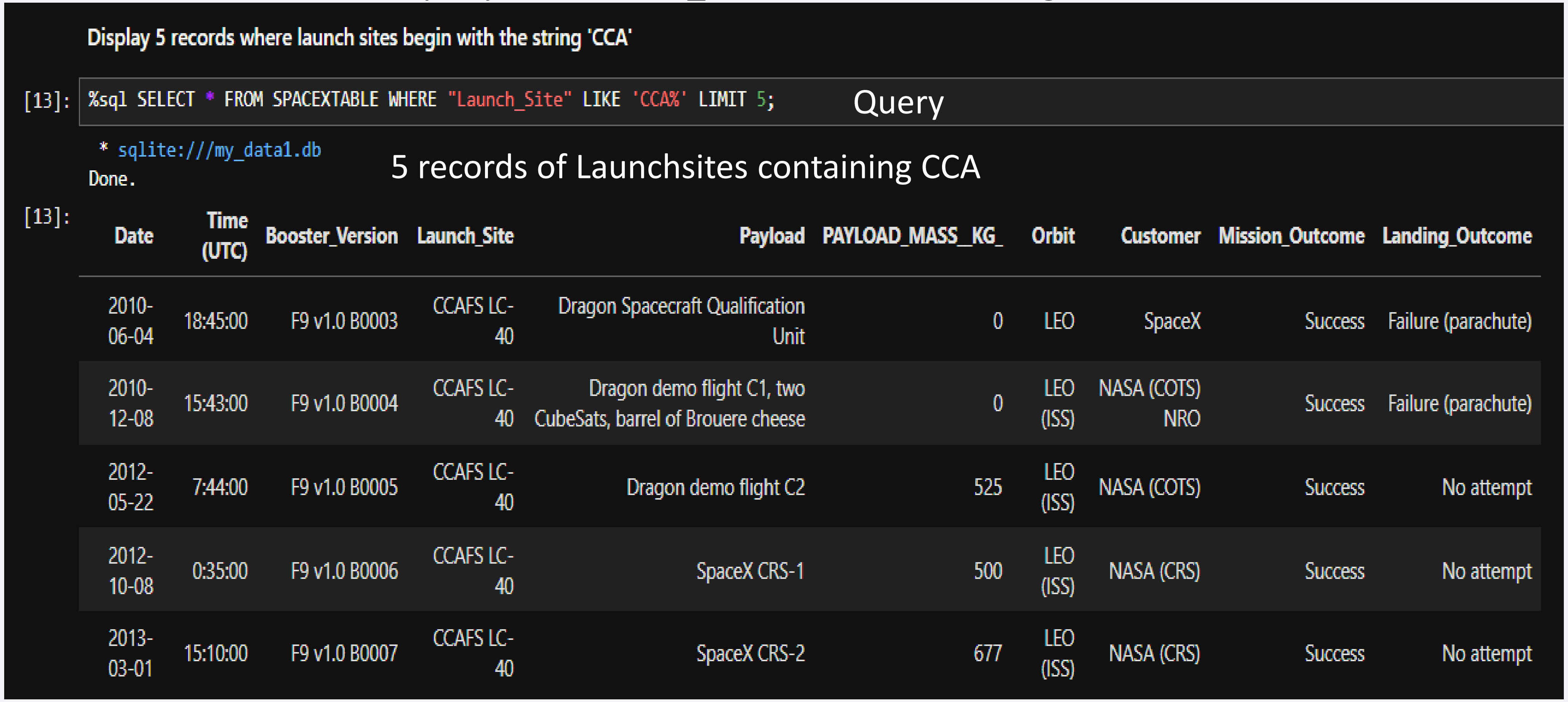
```
 * sqlite:///my_data1.db
Done.
```

[11]:

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

Launch Site Names Begin with 'CCA'

Use of ‘Where’ query over Launch_Sites column containing “CCA” in SPACEXTABLE db



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

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

Query

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

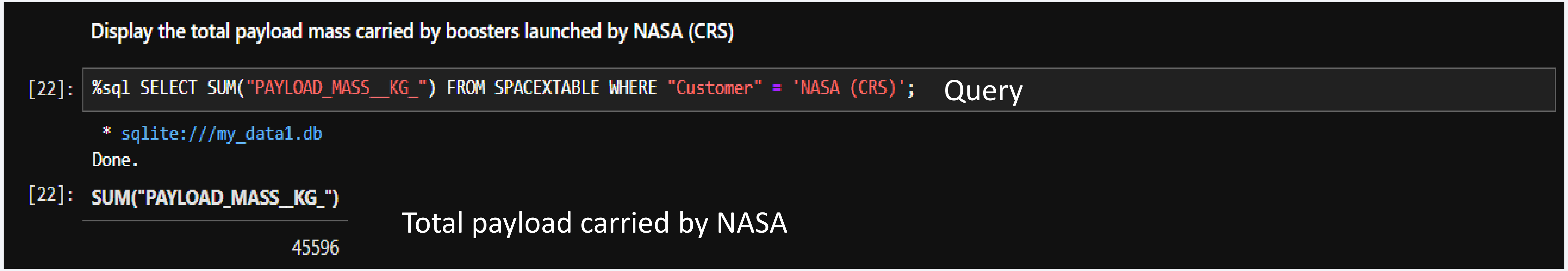
5 records of Launchsites containing CCA

[13]:

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

Use of ‘Where’ query over Customer column containing “NASA (CRS)” in SPACEXTABLE db calculated sum of “PAYLOAD_MASS_KG” column



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

```
[22]: %sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)';
```

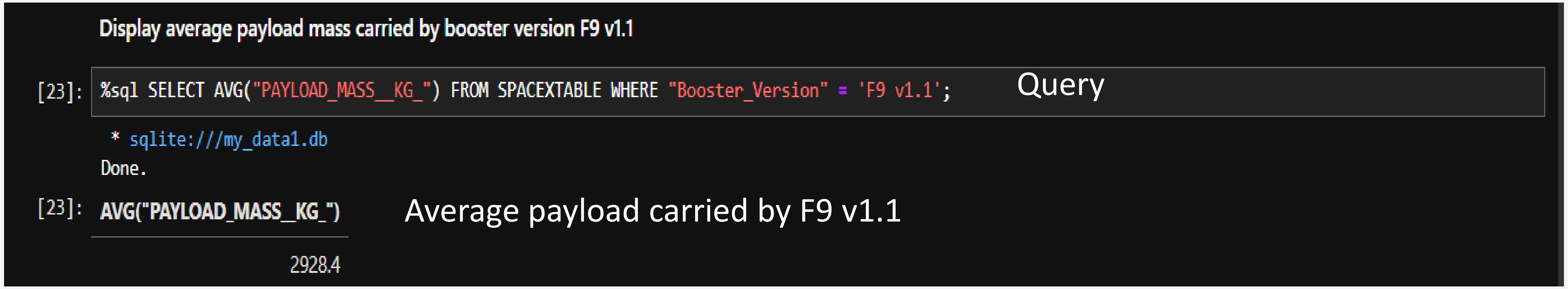
Query

```
 * sqlite:///my_data1.db
Done.
[22]: SUM("PAYLOAD_MASS_KG_")
                      45596
```

Total payload carried by NASA

Average Payload Mass by F9 v1.1

Use of ‘Where’ query over Booster_Version column containing “F9 v1.1” in SPACEXTABLE db calculated average of “PAYLOAD_MASS_KG” column



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

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

Query

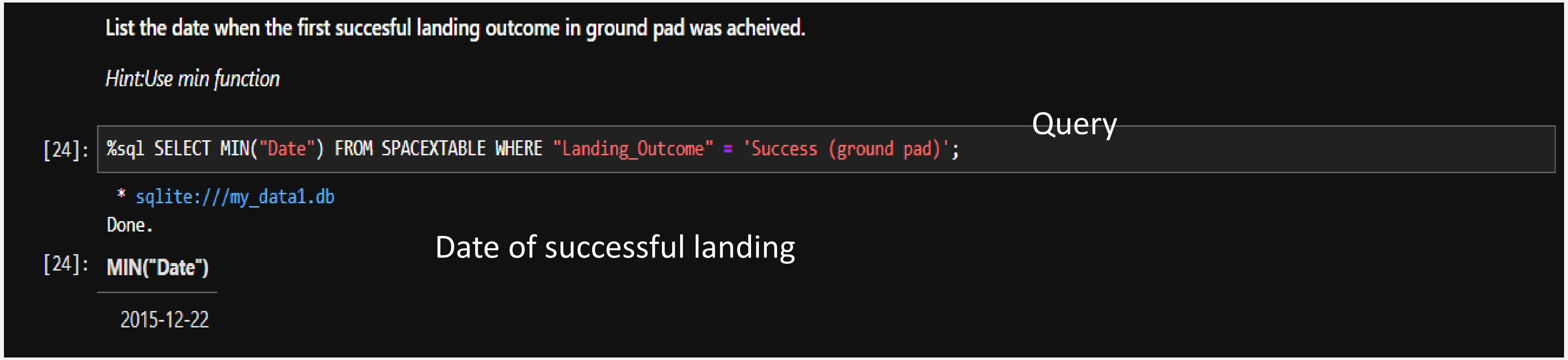
```
 * sqlite:///my_data1.db
Done.
```

```
[23]: AVG("PAYLOAD_MASS__KG_")
                        2928.4
```

Average payload carried by F9 v1.1

First Successful Ground Landing Date

Use of ‘Where’ query over Landing_Outcome column containing “Success (ground pad)” in SPACEXTABLE db calculated Minimum of “Date” column


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

Hint:Use min function

Query

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

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

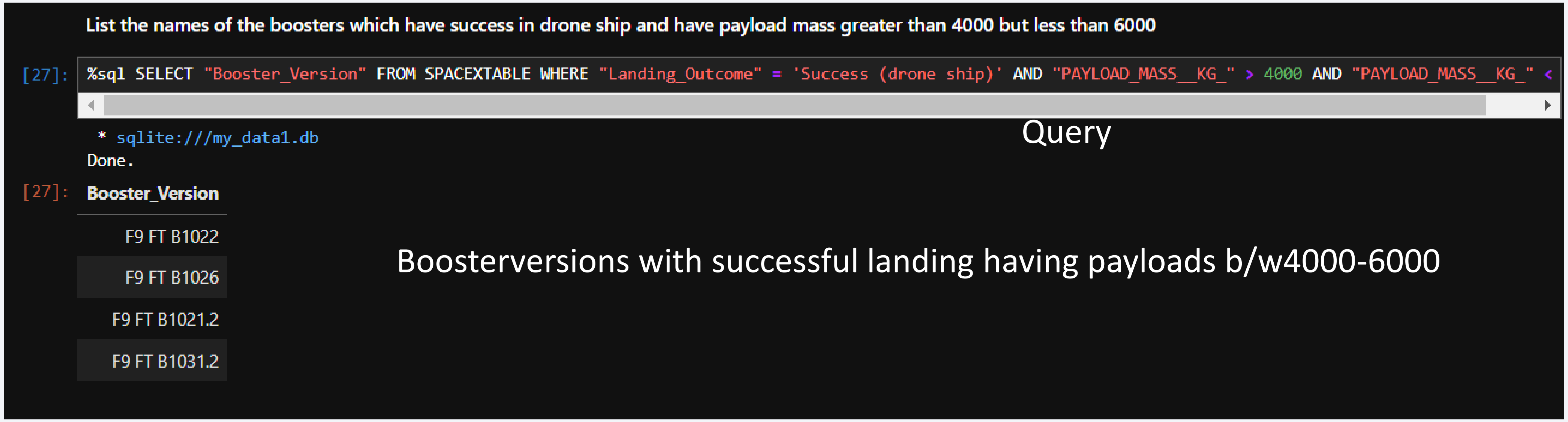
Date of successful landing

[24]:

MIN("Date")
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Use of 'Where' query over Landing_Outcome column containing "Success (drone ship)" and "PAYLOAD_MASS_KG" b/w 400
and 6000 in SPACEXTABLE db presented "Booster_Version"



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

```
[27]: %sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" <
```

Query

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

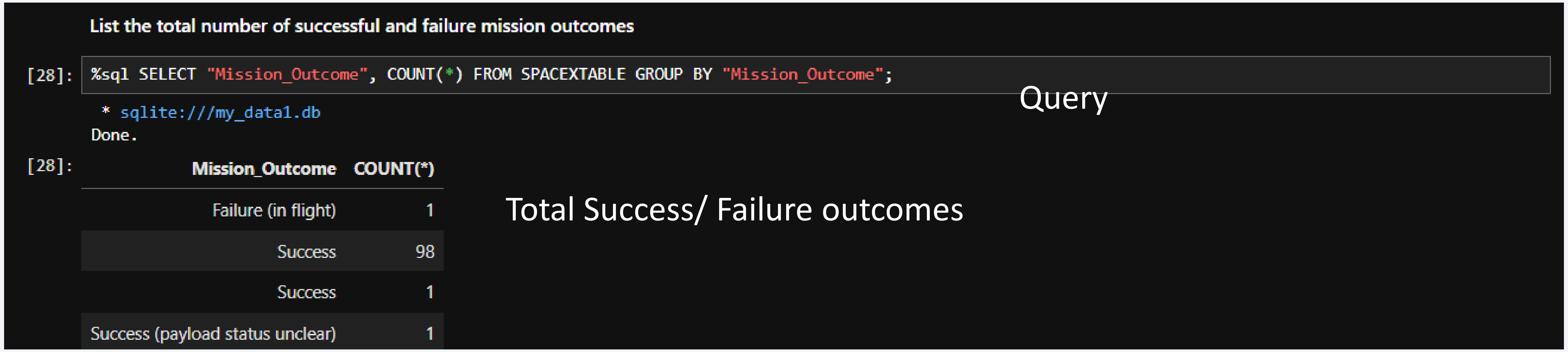
[27]:

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

Boosterversions with successful landing having payloads b/w4000-6000

Total Number of Successful and Failure Mission Outcomes

Counted Mission_Outcome column Grouped by different outcomes



List the total number of successful and failure mission outcomes

```
[28]: %sql SELECT "Mission_Outcome", COUNT(*) FROM SPACEXTABLE GROUP BY "Mission_Outcome";
 * sqlite:///my_data1.db
Done.
```

[28]:

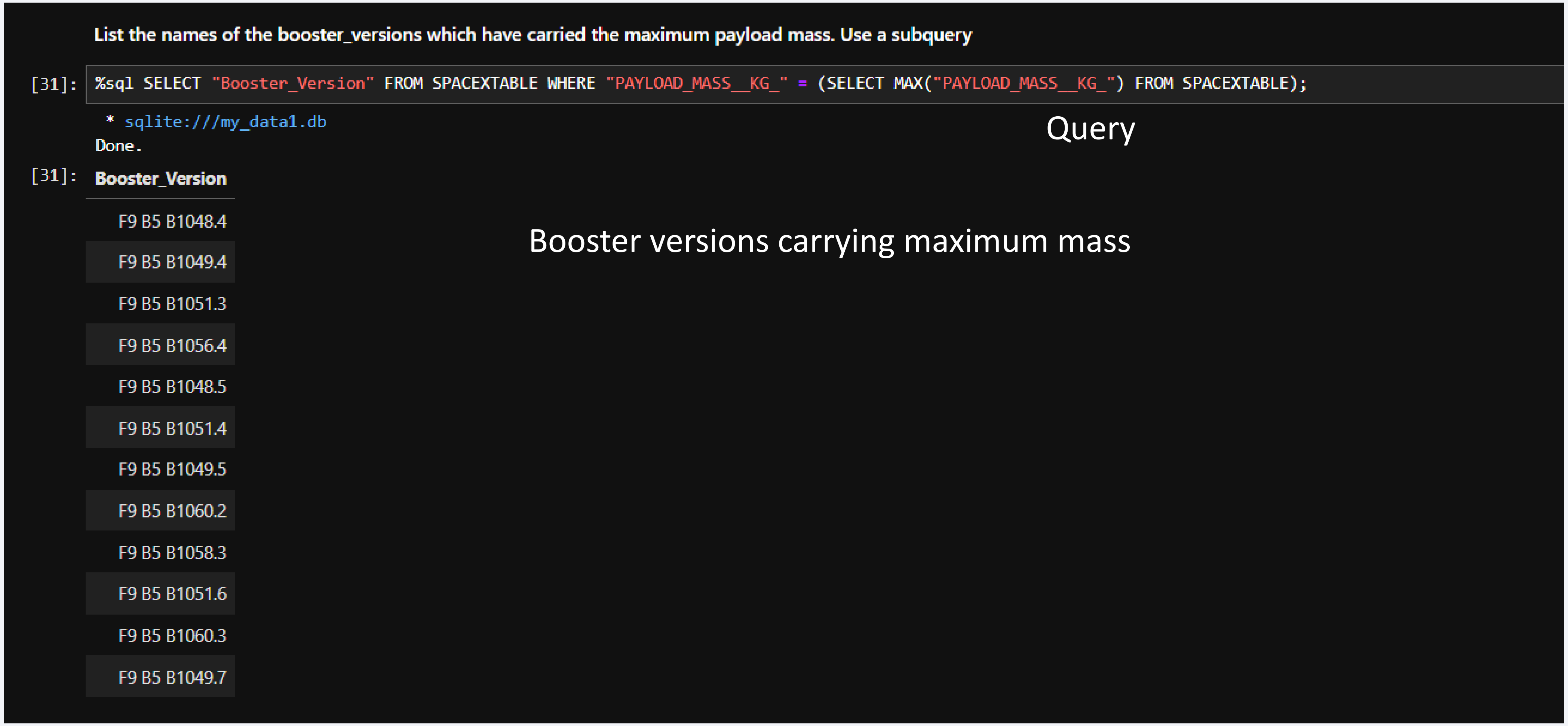
Mission_Outcome	COUNT(*)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Query

Total Success/ Failure outcomes

Boosters Carried Maximum Payload

Selected Booster version that carried maximum payload mass using subquery to identify maximum mass



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

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

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

Query

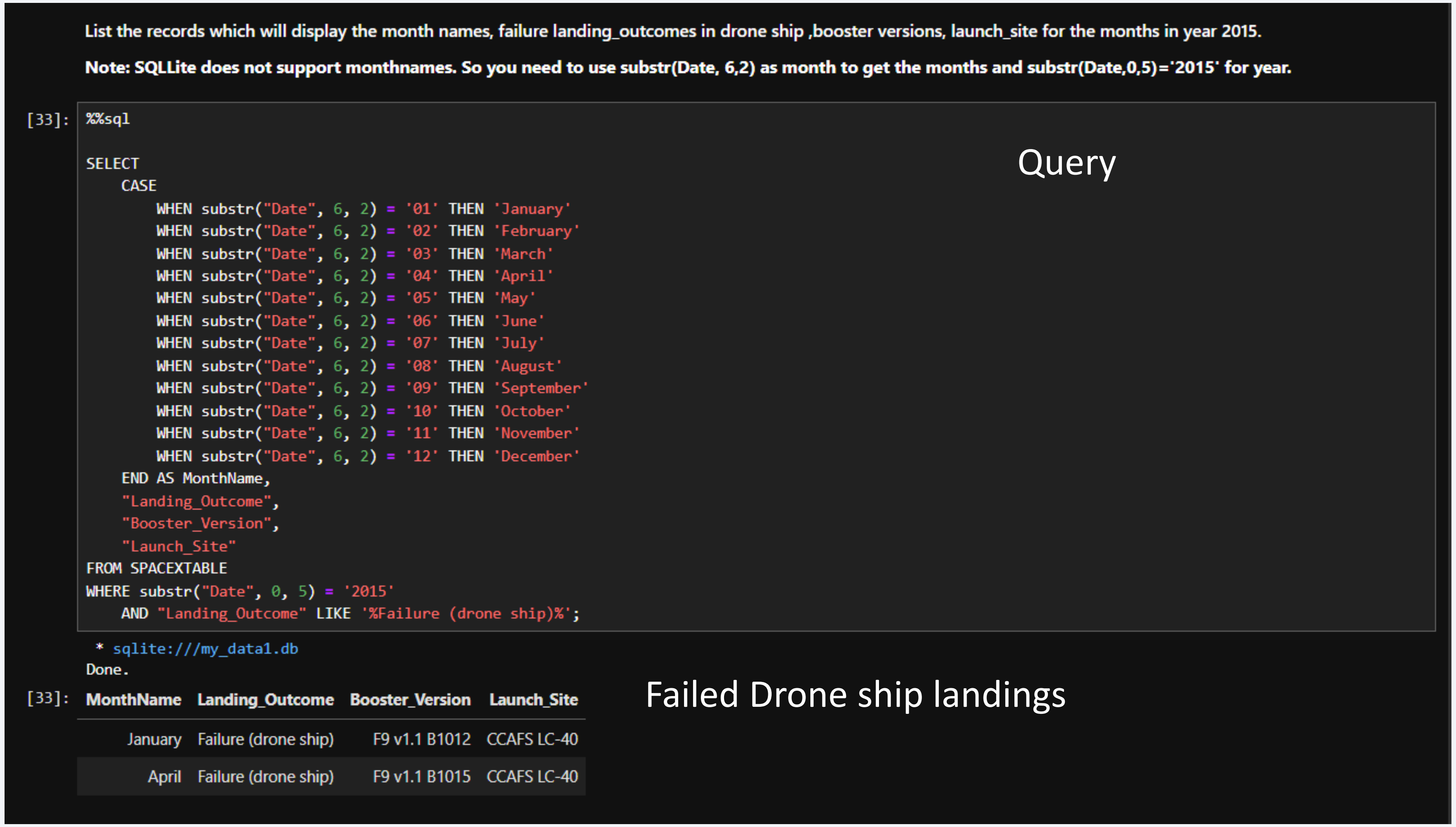
[31]:

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

Booster versions carrying maximum mass

2015 Launch Records

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. QLLite 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.



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.

```
[33]: %%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 MonthName,
    "Landing_Outcome",
    "Booster_Version",
    "Launch_Site"
FROM SPACEXTABLE
WHERE substr("Date", 0, 5) = '2015'
    AND "Landing_Outcome" LIKE '%Failure (drone ship)%';
```

Query

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

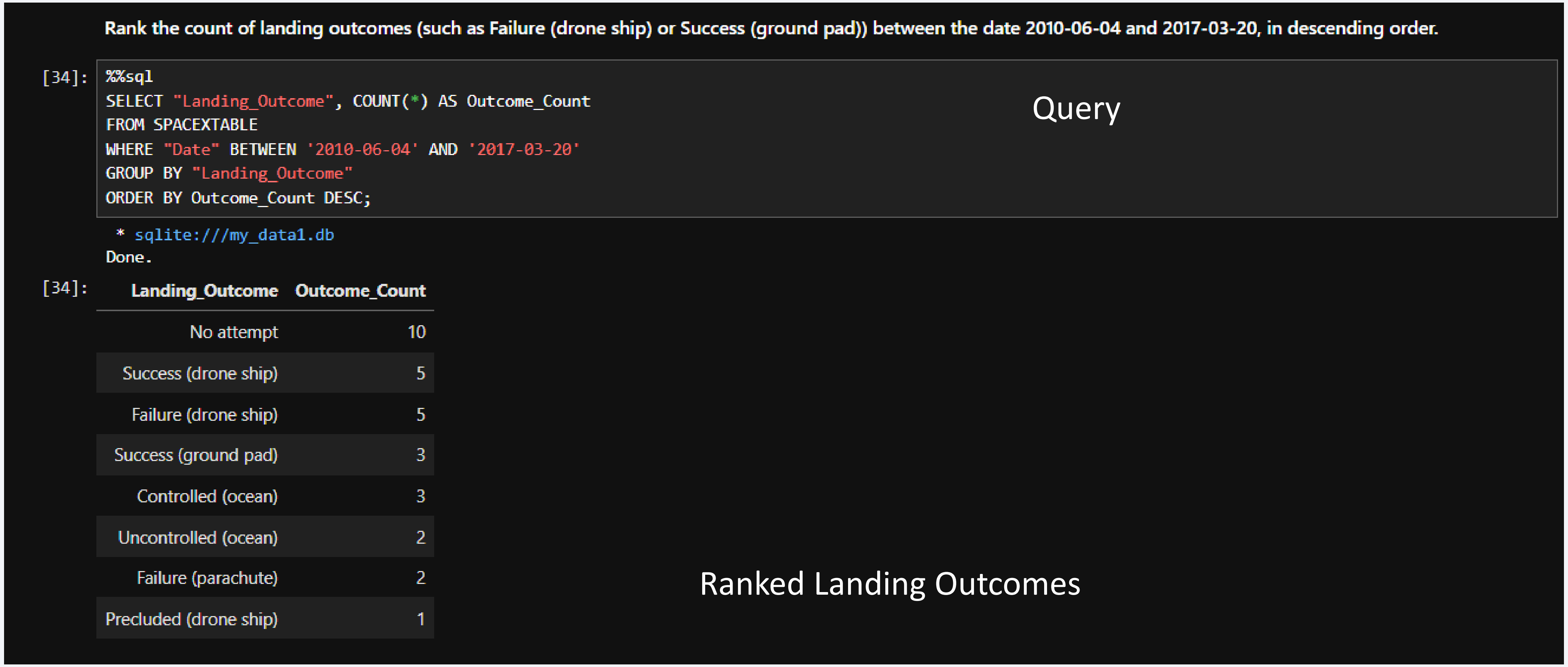
[33]:

MonthName	Landing_Outcome	Booster_Version	Launch_Site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Failed Drone ship landings

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.



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.

```
[34]: %%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;
```

Query

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

[34]:

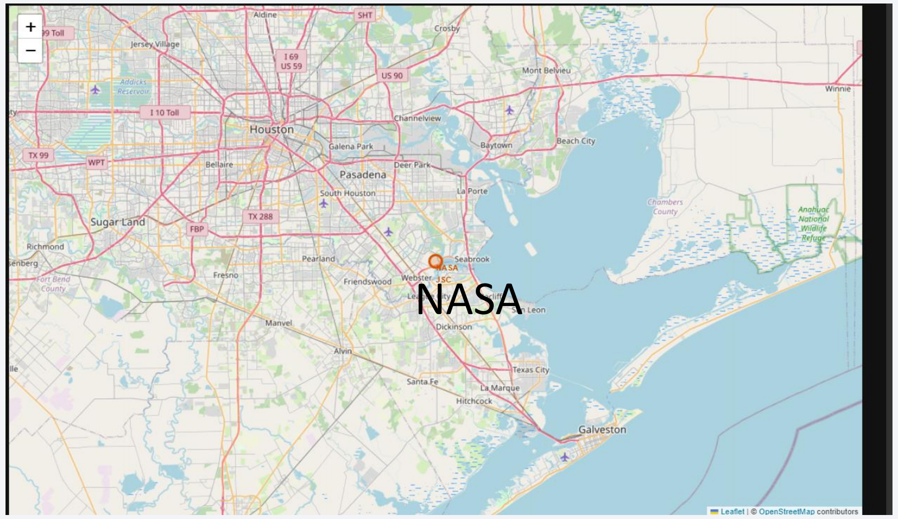
Landing_Outcome	Outcome_Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

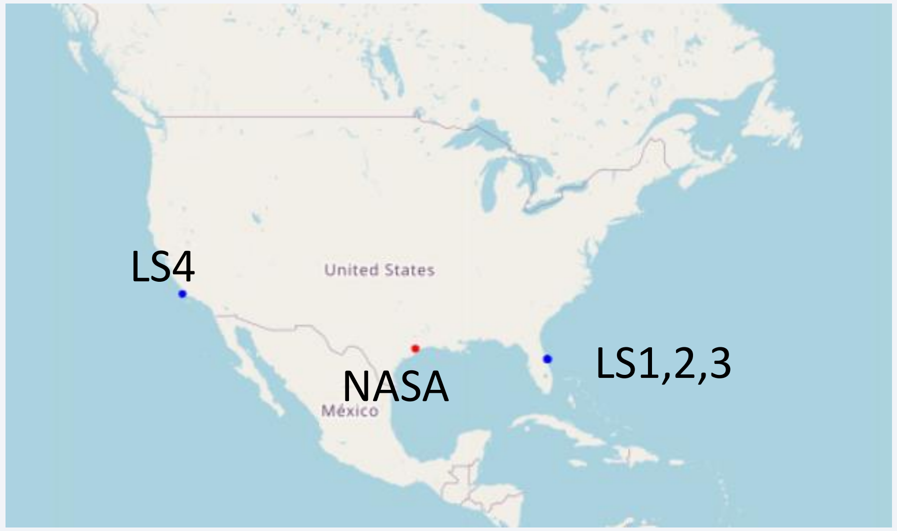
Ranked Landing Outcomes

Section 3

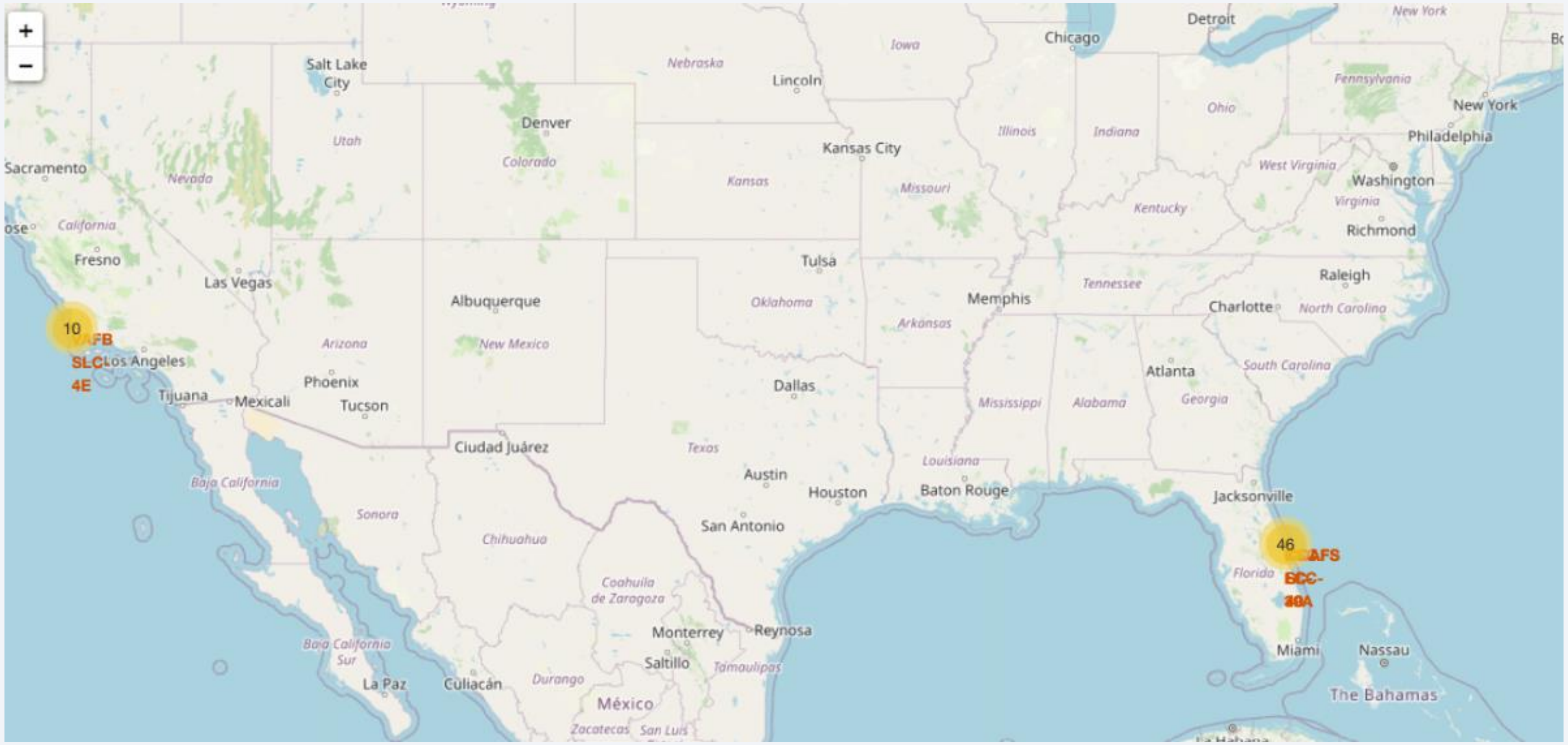
Launch Sites Proximities Analysis

NASA and Space X Launch sites

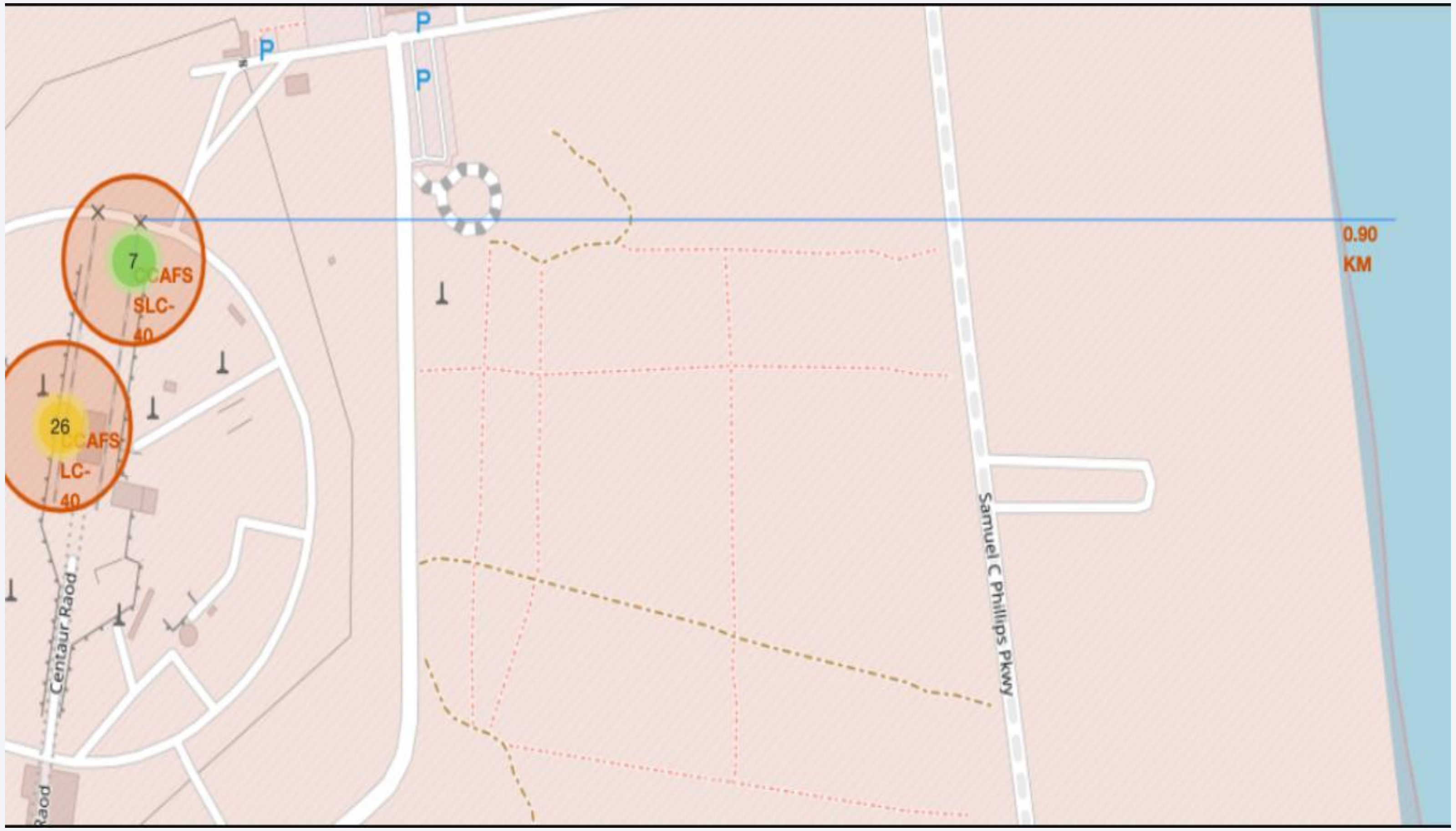




Marker Cluster of Launch Sites and



Launch Site Proximities to Transportation Systems



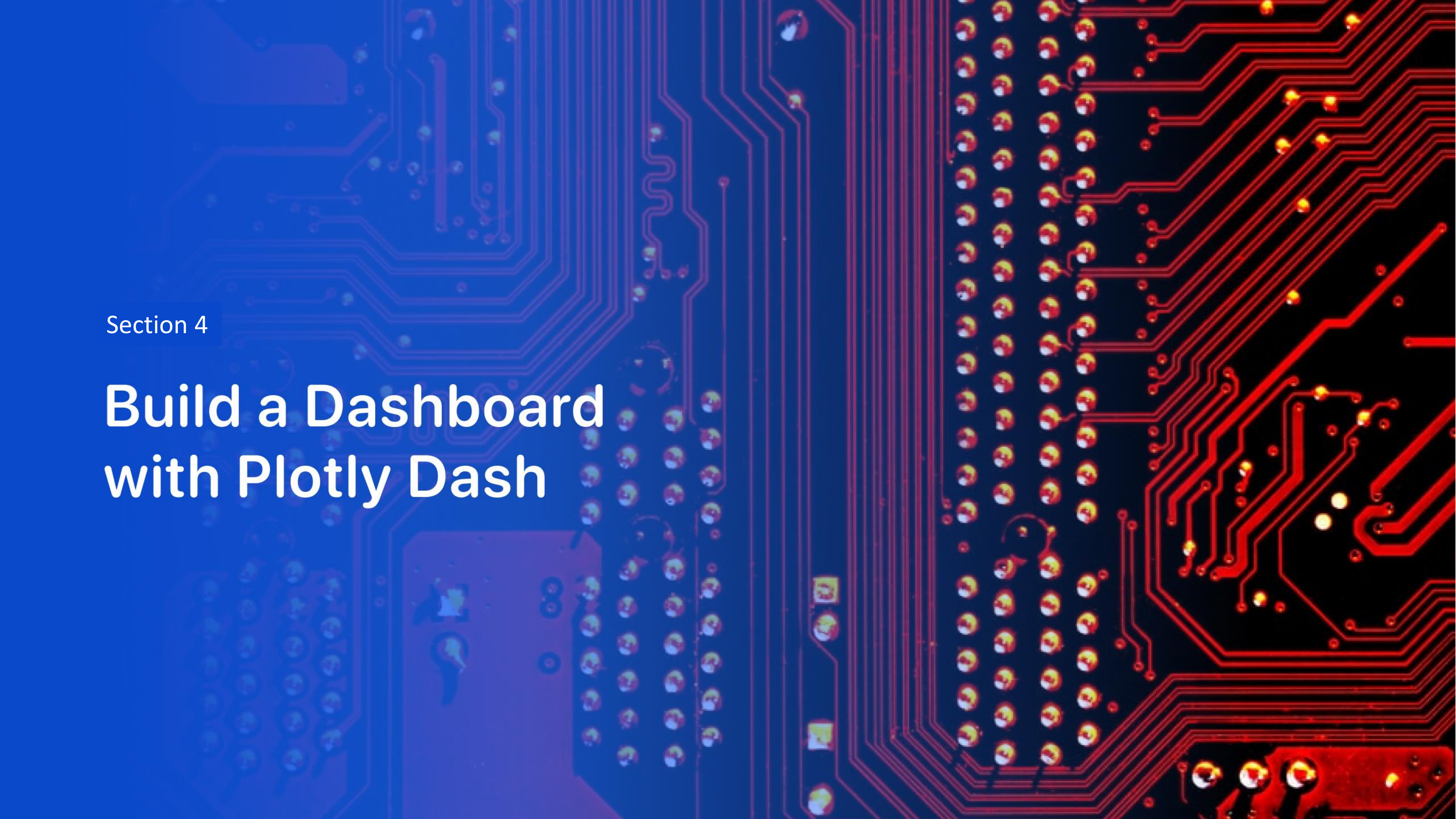
railways

roadways

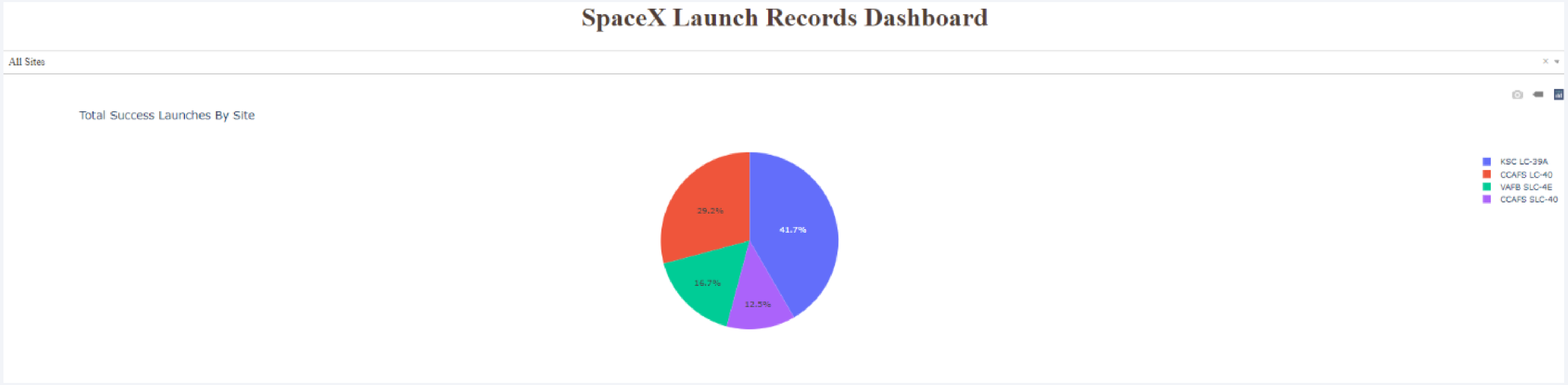
Coastlines

Section 4

Build a Dashboard with Plotly Dash



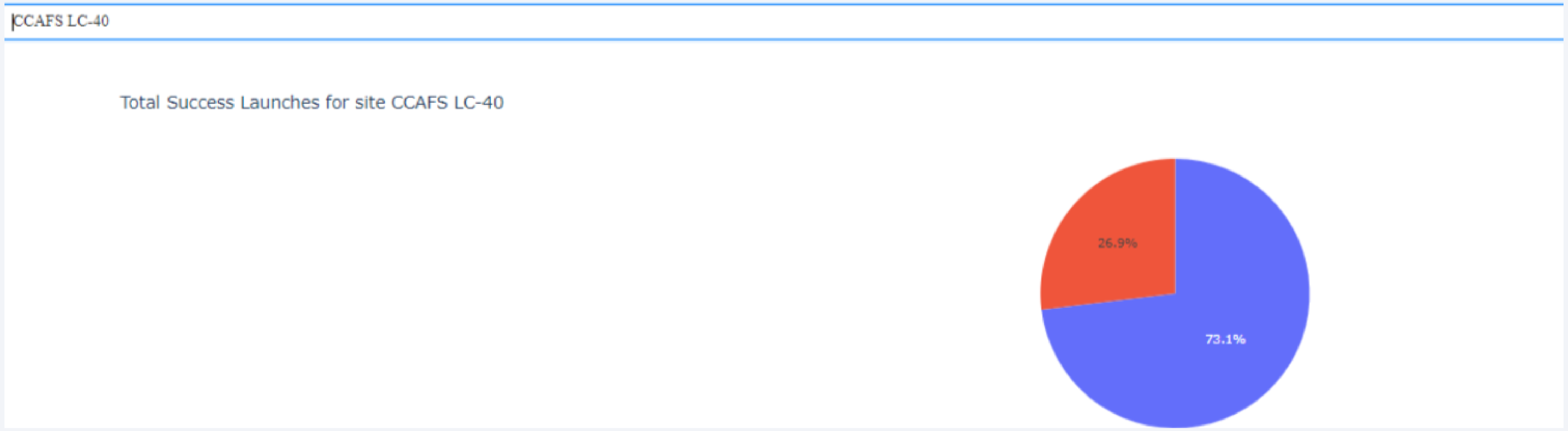
Success Count – All Sites


SpaceX Launch Records Dashboard

All Sites

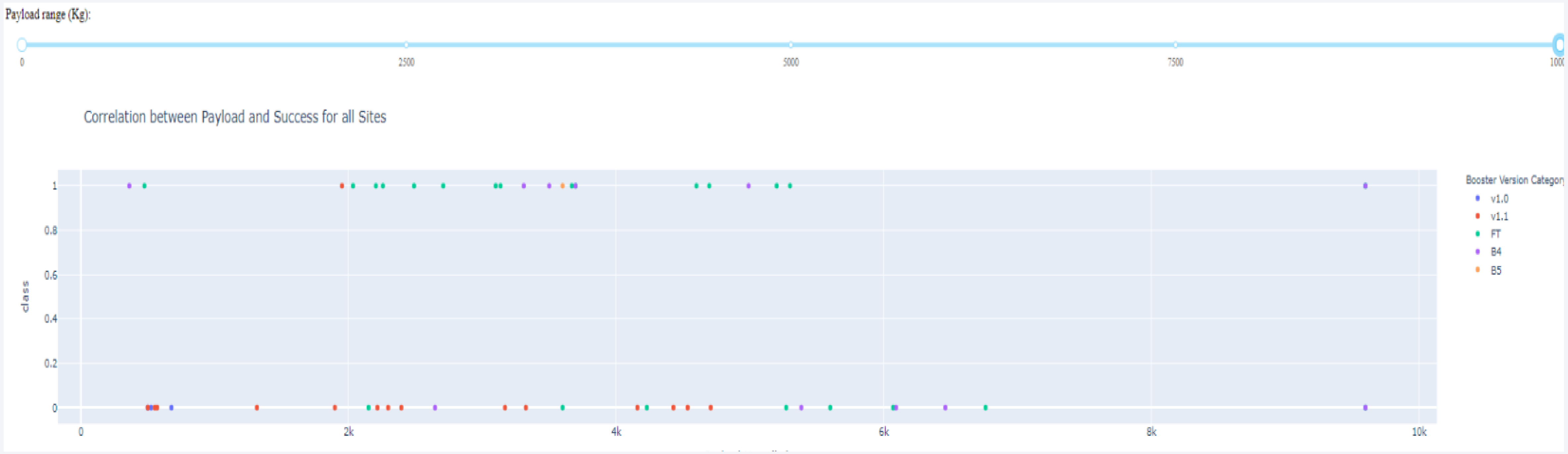
KSC has the largest share of success rate followed by CCAFS LC- 40

Highest Launch Success ratio


CCAFS LC-40

Total Success Launches for site CCAFS LC-40

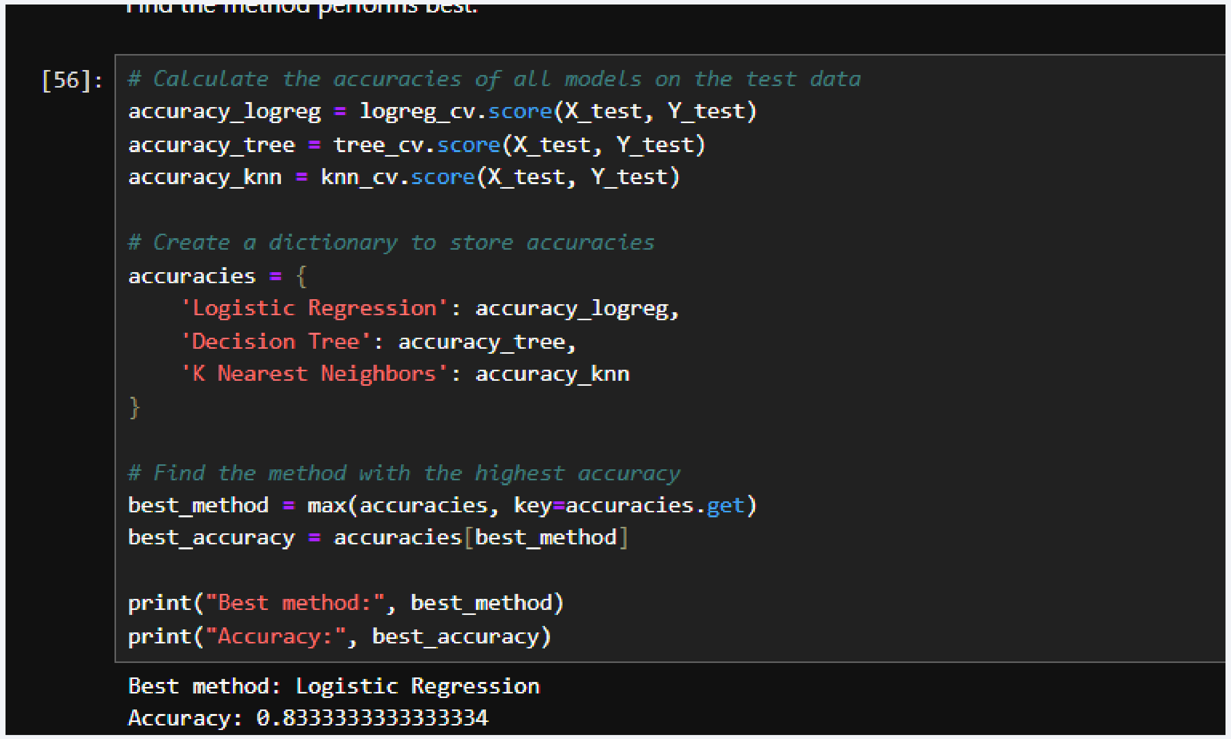
Booster Version FT has the highest success rate when carrying heavy loads



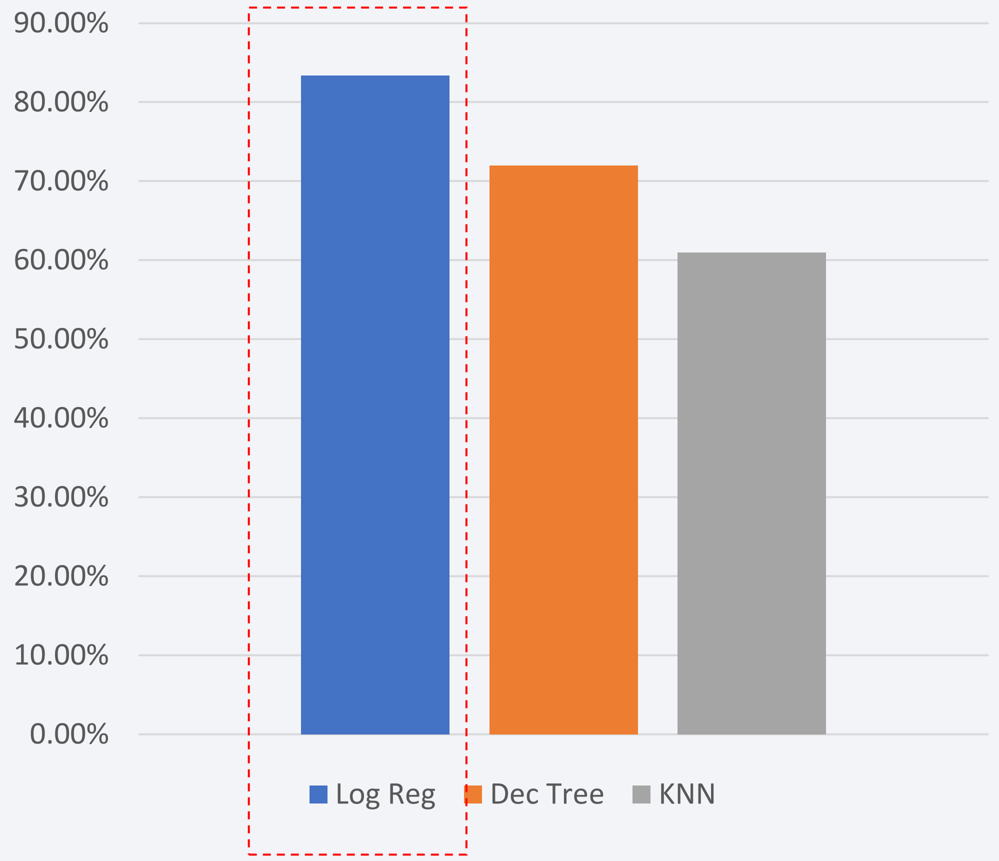
Section 5

Predictive Analysis (Classification)

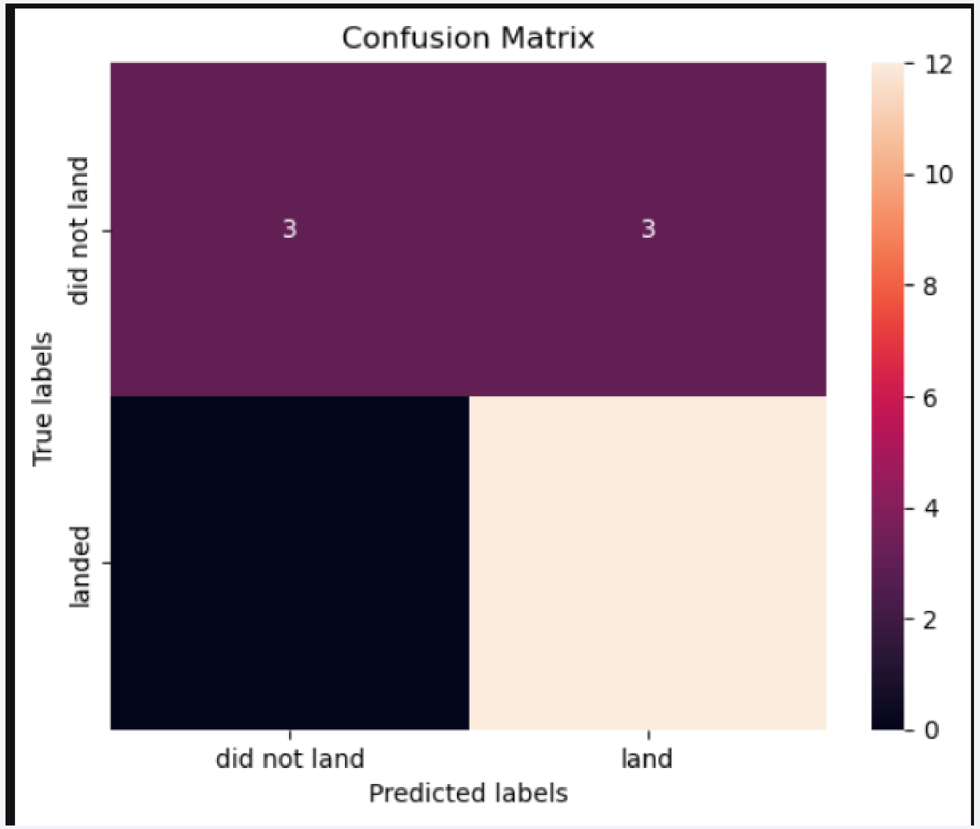
Classification Accuracy



```
[56]: # Calculate the accuracies of all models on the test data
      accuracy_logreg = logreg_cv.score(X_test, Y_test)
      accuracy_tree = tree_cv.score(X_test, Y_test)
      accuracy_knn = knn_cv.score(X_test, Y_test)

      # Create a dictionary to store accuracies
      accuracies = {
          'Logistic Regression': accuracy_logreg,
          'Decision Tree': accuracy_tree,
          'K Nearest Neighbors': accuracy_knn
      }

      # Find the method with the highest accuracy
      best_method = max(accuracies, key=accuracies.get)
      best_accuracy = accuracies[best_method]

      print("Best method:", best_method)
      print("Accuracy:", best_accuracy)

      Best method: Logistic Regression
      Accuracy: 0.8333333333333334
```



Confusion Matrix



As per confusion Matrix of Logistic Regression it is found that the model was able to predict 81% of outcomes successfully

Conclusions

- There are 50% chances of First stage of Falcon 9 will land successfully

Thank you!