IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Burak Sahin Kucuk
27.02.2024

Outline

Executive Summary

- Summary of methodologies
  - Acquiring Data via API Calls
  - Extracting Data with Web Scrapping
  - Data Cleaning & Transformation (Data Wrangling)
  - EDA with SQL
  - Data Visualization
  - Folium - Interactive Visualizations
  - Prediction With ML Steps
- Summary of all results
  - EDA Results
  - Interactive Analytics
  - Prediction Analysis

Introduction

- Project background and context
- Problems you want to find answers

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
    - Acquiring data with SpaceX API and web-scraping thanks to W
- Perform data wrangling
    - Categorical features underwent one-hot encoding.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    - How to build, tune, evaluate classification models

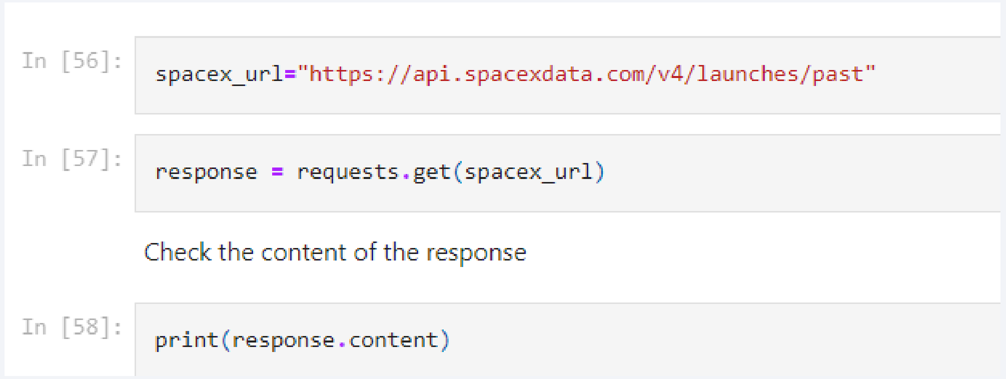
Data Collection

Data was collecting with Python Libraries for Web Scraping and get request with SpaceX API

We parsed the response content into a JSON format using the .json() function, then transformed it into a pandas dataframe with .json_normalize(). After cleaning the data and addressing missing values, we scraped Falcon 9 launch records from Wikipedia using BeautifulSoup. Our aim was to extract the launch records from an HTML table, parse it, and convert it into a pandas dataframe for further analysis.

Data Collection – SpaceX API

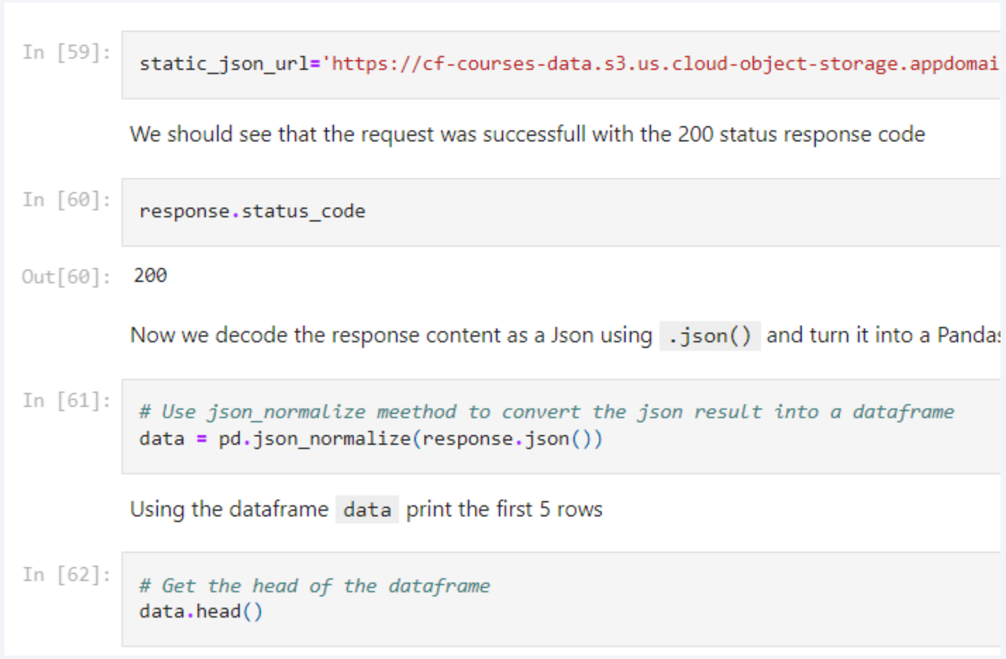
- In the project, data was collected and cleaned from the SpaceX API, ensuring its quality and consistency. This involved tasks like handling missing values and standardizing formats. The goal was to prepare the data for analysis and visualization.
- https://github.com/bupacone/IBMDS-Capstone-Final/blob/main/01a-%20SpaceX%20Data%20Collection%20API.ipynb



```
In [56]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [57]: response = requests.get(spacex_url)
```

Check the content of the response

```
In [58]: print(response.content)
```



```
In [59]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomai
```

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

```
In [60]: response.status_code

Out[60]: 200
```

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

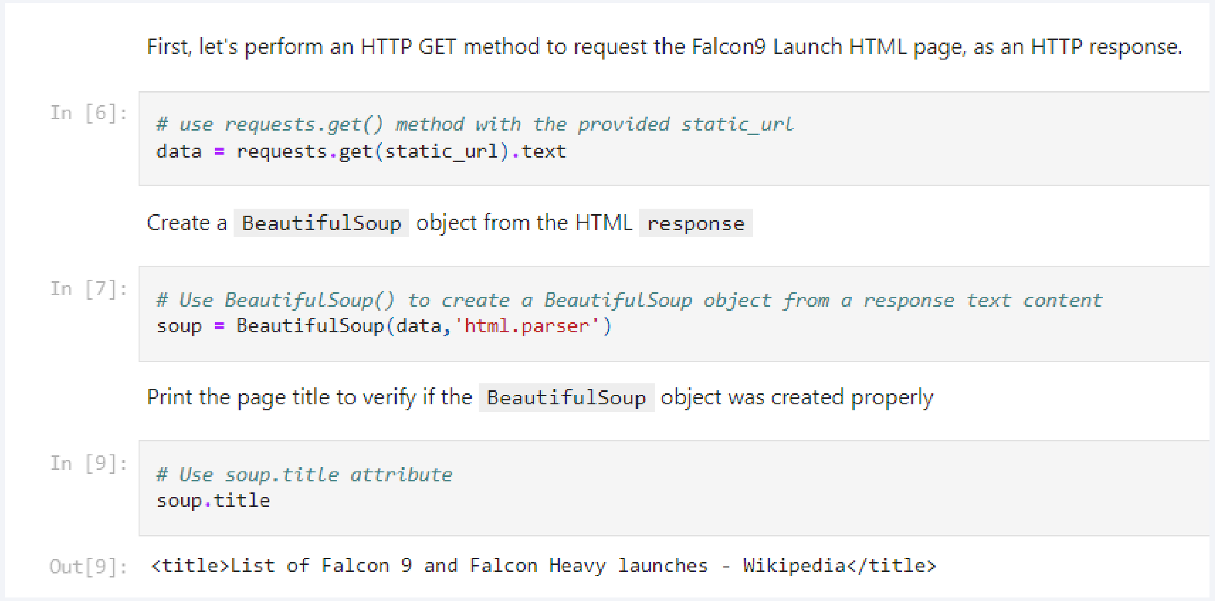
```
In [61]: # Use json_normalize meethod to convert the json result into a dataframe
         data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
In [62]: # Get the head of the dataframe
         data.head()
```

Data Collection – Web Scraping

- SpaceX data extracted from Wikipedia with WebScrapping which are created for Python such as BeautifulSoup
- https://github.com/bupacone/IBMDS-Capstone-Final/blob/main/02a-%20SpaceX%20Web%20Scraping%20Lab.ipynb



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

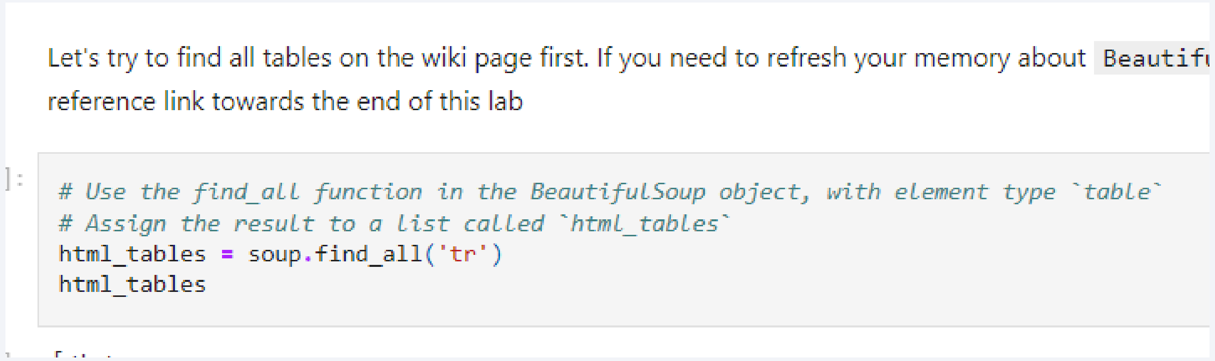
```
In [6]:  # use requests.get() method with the provided static_url
         data = requests.get(static_url).text
```

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

```
In [7]:  # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
         soup = BeautifulSoup(data,'html.parser')
```

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

```
In [9]:  # Use soup.title attribute
         soup.title

Out[9]:  <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```



Let's try to find all tables on the wiki page first. If you need to refresh your memory about `Beautifu`
reference link towards the end of this lab

```
]:  # 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('tr')
    html_tables
```

Data Wrangling

- We performed exploratory data analysis (EDA) on SpaceX launch data to identify patterns and determine the training labels for supervised models. First, we imported the necessary libraries and loaded the dataset. Then, we conducted an analysis to identify missing values, followed by categorizing columns as numerical or categorical.
- https://github.com/bupacone/IBMDS-Capstone-Final/blob/main/03a-%20SpaceX%20Data%20Wrangling.ipynb

understand launch distribution
Calculated the number of launches

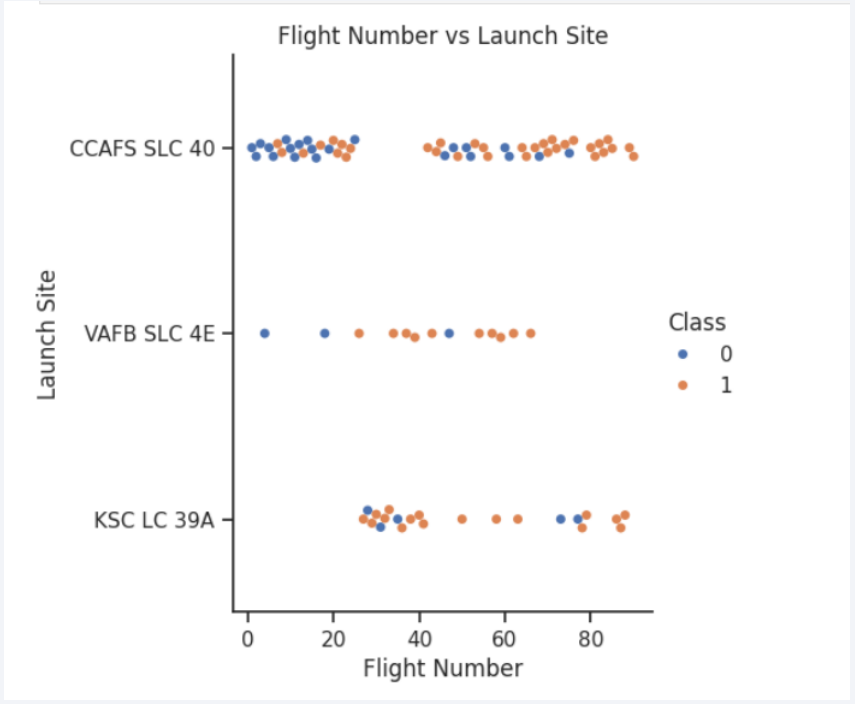
analyze mission trajectories.
Determined the number and occurrence

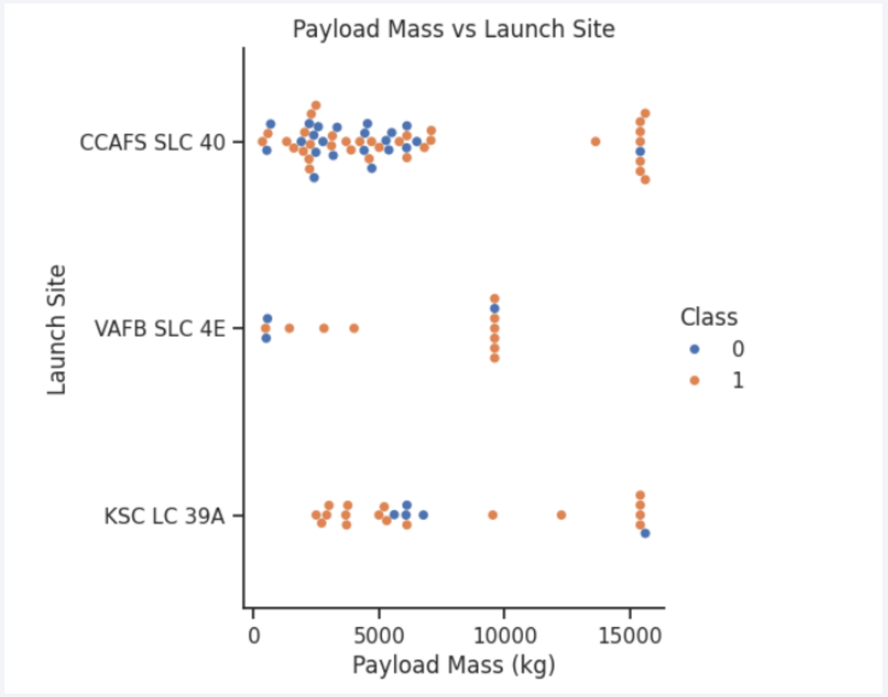
Calculated the number and occurrence of mission outcomes

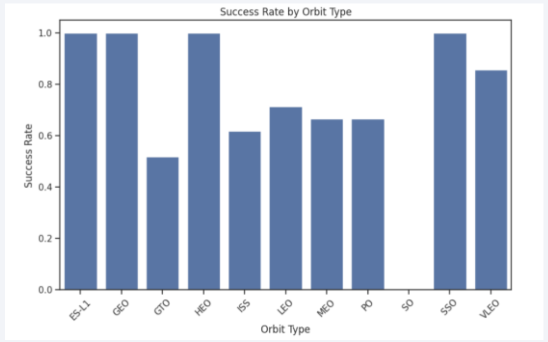
Created a landing outcome label from the Outcome column

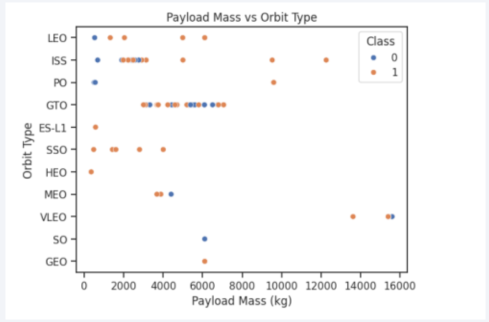
EDA with Data Visualization

- We create visualization to look our situation with Data Visualization tools like matplotlib, seaborn…
- https://github.com/bupacone/IBMDS-Capstone-Final/blob/main/05a-%20SpaceX%20EDA%20Data%20Visualization.ipynb









EDA with SQL

In this analysis, we've utilized SQL queries within Jupyter Notebook (b2) to delve into the SpaceX dataset. Here's a list of the tasks we performed:

1. Displayed unique launch sites.
2. Retrieved records where launch sites begin with 'CCA'.
3. Calculated the total payload mass carried by boosters launched by NASA (CRS).
4. Determined the average payload mass carried by booster version F9 v1.1.
5. Listed the date of the first successful landing outcome on a ground pad.
6. Identified booster names with successful landings on drone ships with payload mass between 4000 and 6000 kg.
7. Counted the total number of successful and failed mission outcomes.
8. Listed booster versions that carried the maximum payload mass using a subquery.
9. Retrieved records showing failure landing outcomes on drone ships, including booster versions and launch sites for the year 2015.
10. Ranked the count of landing outcomes between specific dates in descending order.
11. https://github.com/bupacone/IBMDS-Capstone-Final/blob/main/04a-%20SpaceX%20EDA%20with%20SQL%20Lab.ipynb

Build an Interactive Map with Folium

We utilized Folium, a Python library for interactive mapping, to analyze the locations of SpaceX launch sites and their proximities. Here's a breakdown of the tasks we completed:

- Marked all launch sites on a map: We imported a dataset containing the coordinates of each launch site and plotted them on a Folium map centered around NASA Johnson Space Center in Houston, Texas.
- Marked the success/failed launches for each site on the map: We enhanced the map by adding markers for each launch record, color-coded based on whether the launch was successful or failed. Marker clusters were used to simplify the visualization of multiple markers at the same location.
- Calculated the distances between a launch site and its proximities: We explored the proximities of launch sites by adding markers and distance lines to the closest coastline, railway, highway, and city. Distances were calculated using geographical coordinates and displayed on the map.

https://github.com/bupacone/IBMDS-Capstone-Final/blob/main/06a-%20SpaceX%20Launch%20Site%20Location%20-%20Folium.ipynb

Build a Dashboard with Plotly Dash

- We created a user-friendly dashboard using Plotly Dash, a Python library for building interactive web applications. Within the dashboard, we included pie charts to display the total number of launches from each site, providing a clear overview of launch distribution. Additionally, scatter plots were utilized to depict the correlation between launch outcomes and payload mass for different booster versions. Users can access the dashboard via the provided notebook link to explore SpaceX launch data and gain insights into mission outcomes and payload characteristics.

https://github.com/bupacone/IBMDS-Capstone-Final/blob/main/07a-%20DashApp.py

Predictive Analysis (Classification)

- Modeling: Standardized the input features to ensure that all variables have the same scale. Split the data into training and testing sets for model evaluation. Utilized four classification algorithms: Logistic Regression, Support Vector Machine (SVM), Decision Tree, and K Nearest Neighbors (KNN).
- Hyperparameter Tuning: Employed GridSearchCV to find the optimal hyperparameters for each classification algorithm. Tuned hyperparameters for each model to improve performance.
- Evaluation: Calculated the accuracy of each model on the test dataset to assess its performance. Plotted confusion matrices to visualize the performance of each model in distinguishing between successful and unsuccessful landings.
- Comparison and Selection: Compared the performance of all models based on their accuracy scores. Selected the best-performing model, which was the Decision Tree classifier, with an accuracy score of 88.93%. Identified the best hyperparameters for the selected model. Overall, we performed data preprocessing, model training, hyperparameter tuning, and evaluation to create a machine learning pipeline for predicting the outcome of SpaceX Falcon 9 rocket first stage landings.
- https://github.com/bupacone/IBMDS-Capstone-Final/blob/main/08a-%20SpaceX%20Machine%20Learning%20Prediction%20Lab.ipynb

Results

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

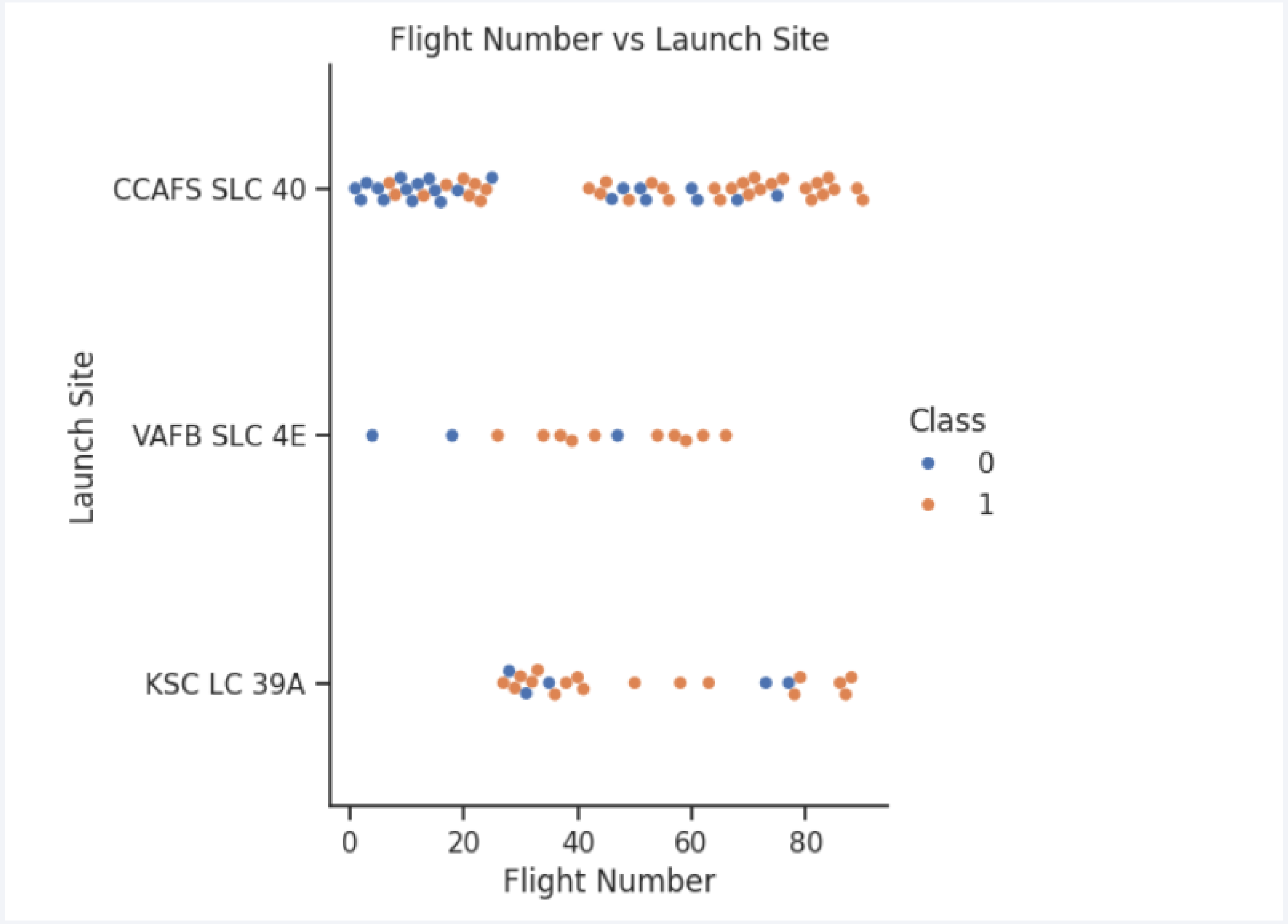
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- CCAFS SLC 40 has much more flights than VAFB SLC 4E and KSC LC39A

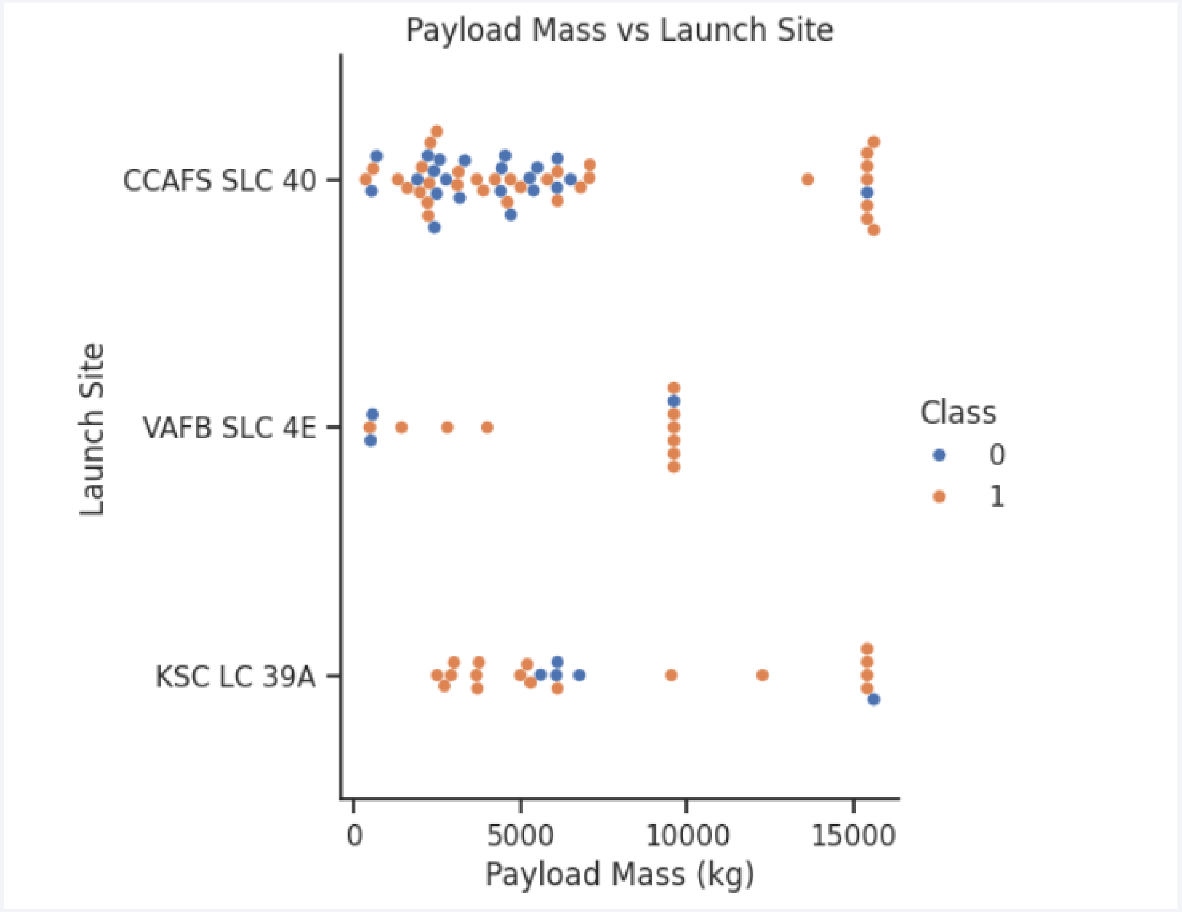
- Also, the flights usually occurred.



Payload vs. Launch Site

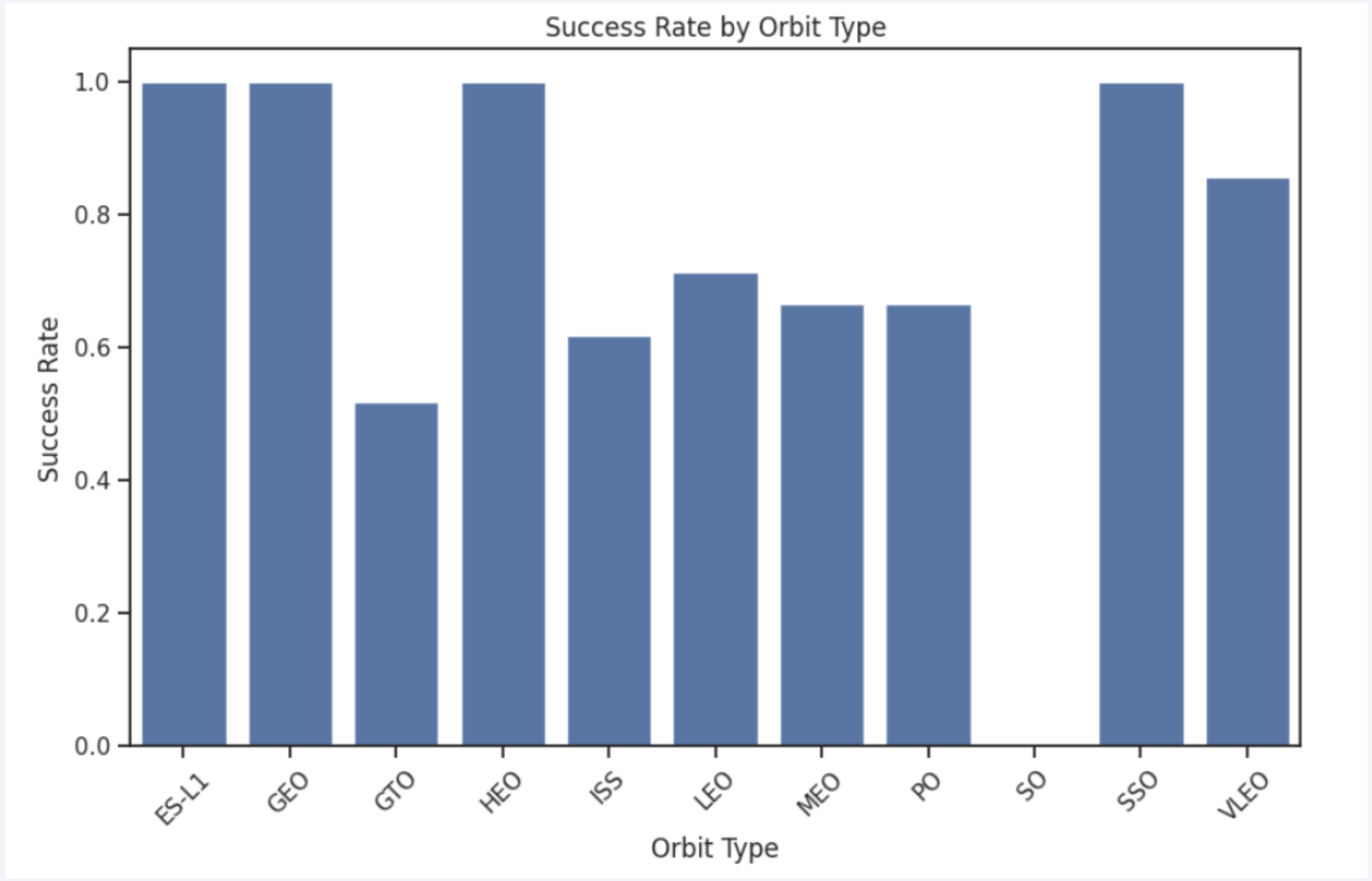
- Greater payload mass for launch site is CCAFS SLC40

- Also, we indicates that the number of highest payload mass usually occurred CCAFS SLC 40



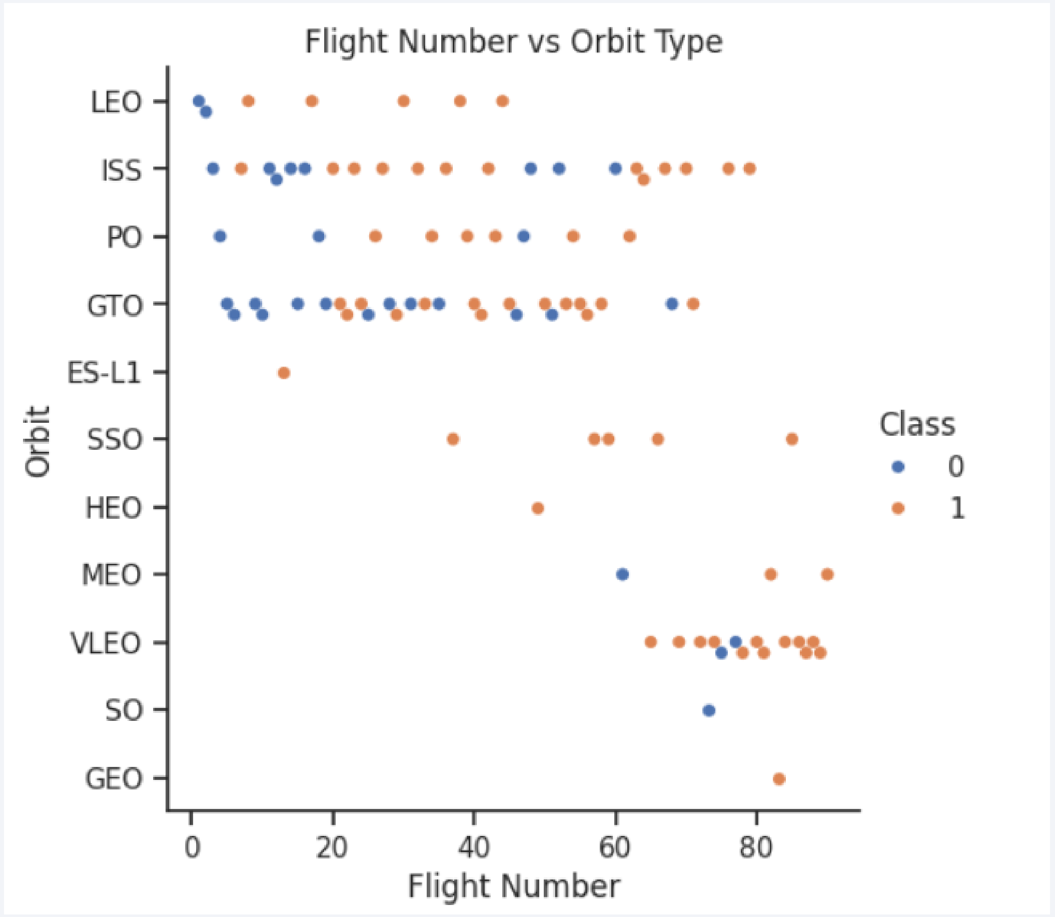
Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO had the most success rate. On the other hand, there is no success in SO.



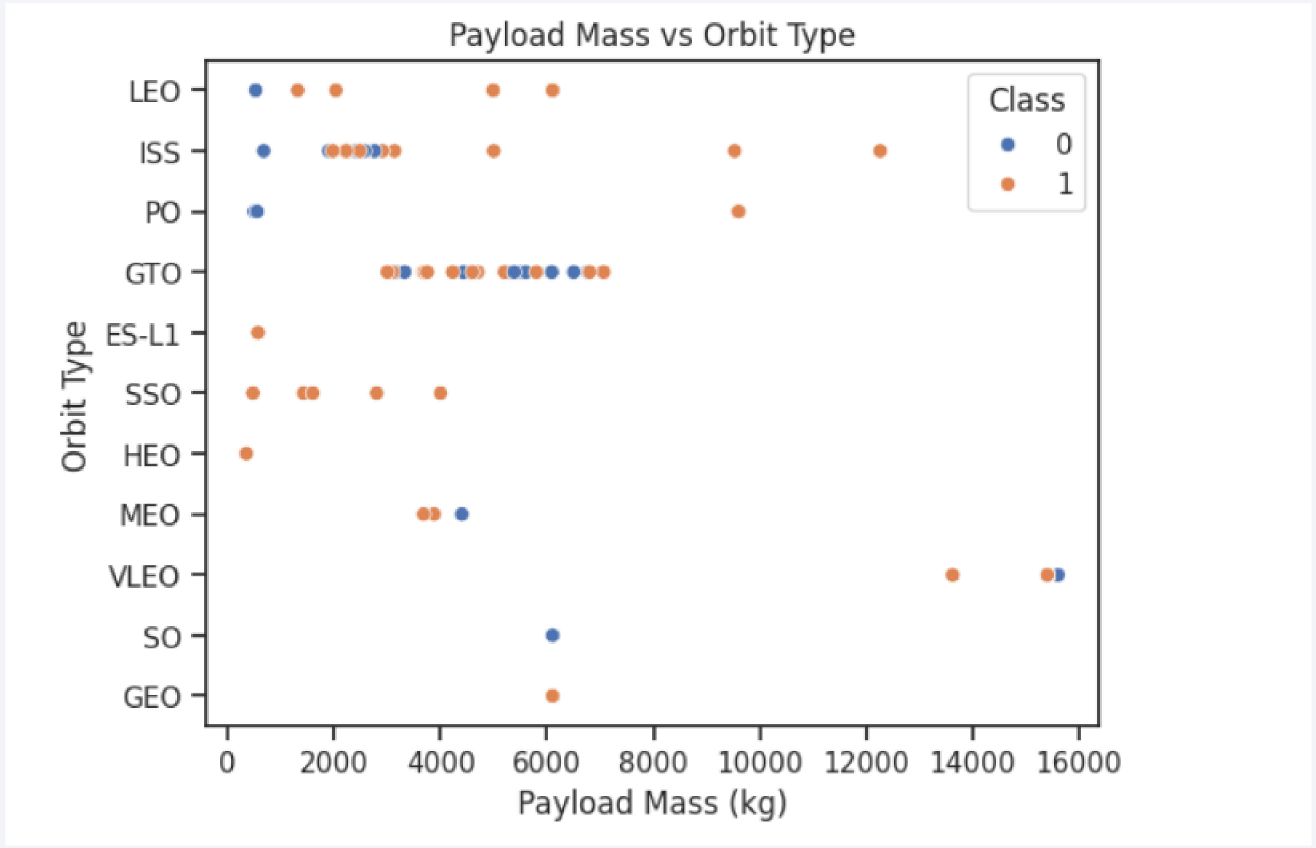
Flight Number vs. Orbit Type

- In the LEO orbit, the success rate seems to be influenced by the number of flights, whereas in the GTO orbit, there doesn't appear to be any correlation between the flight number and success.



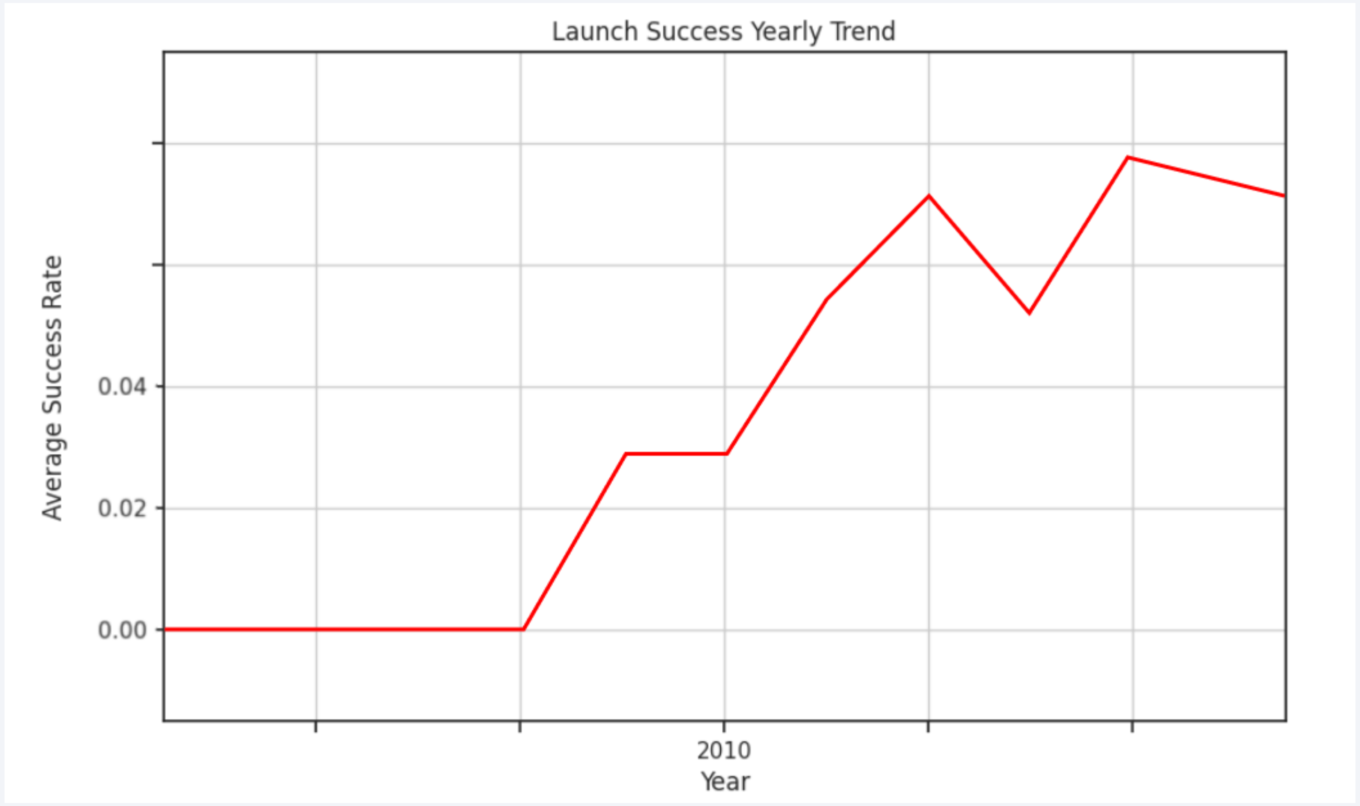
Payload vs. Orbit Type

- When dealing with heavy payloads, the likelihood of successful landings or positive outcomes appears to be higher for missions to Polar, LEO, and ISS orbits. However, distinguishing between successful and unsuccessful missions becomes more challenging for GTO orbits, as both positive and negative landing rates are observed.



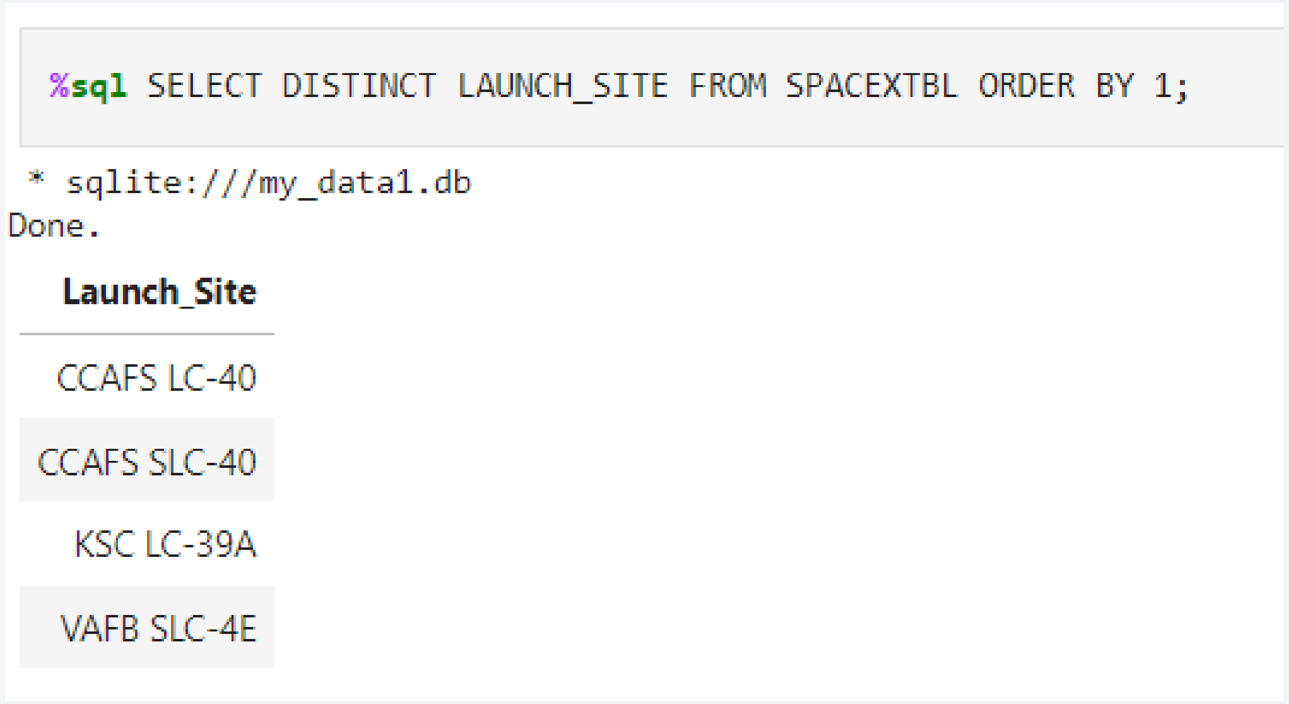
Launch Success Yearly Trend

- Success rate is increasing from 2013 and there is some fluctuate cases.



All Launch Site Names

- Launch Names are CCAFS LC-40, CCAFS SLC-40, KSC LC-39A ,and VAFB SLC 4E

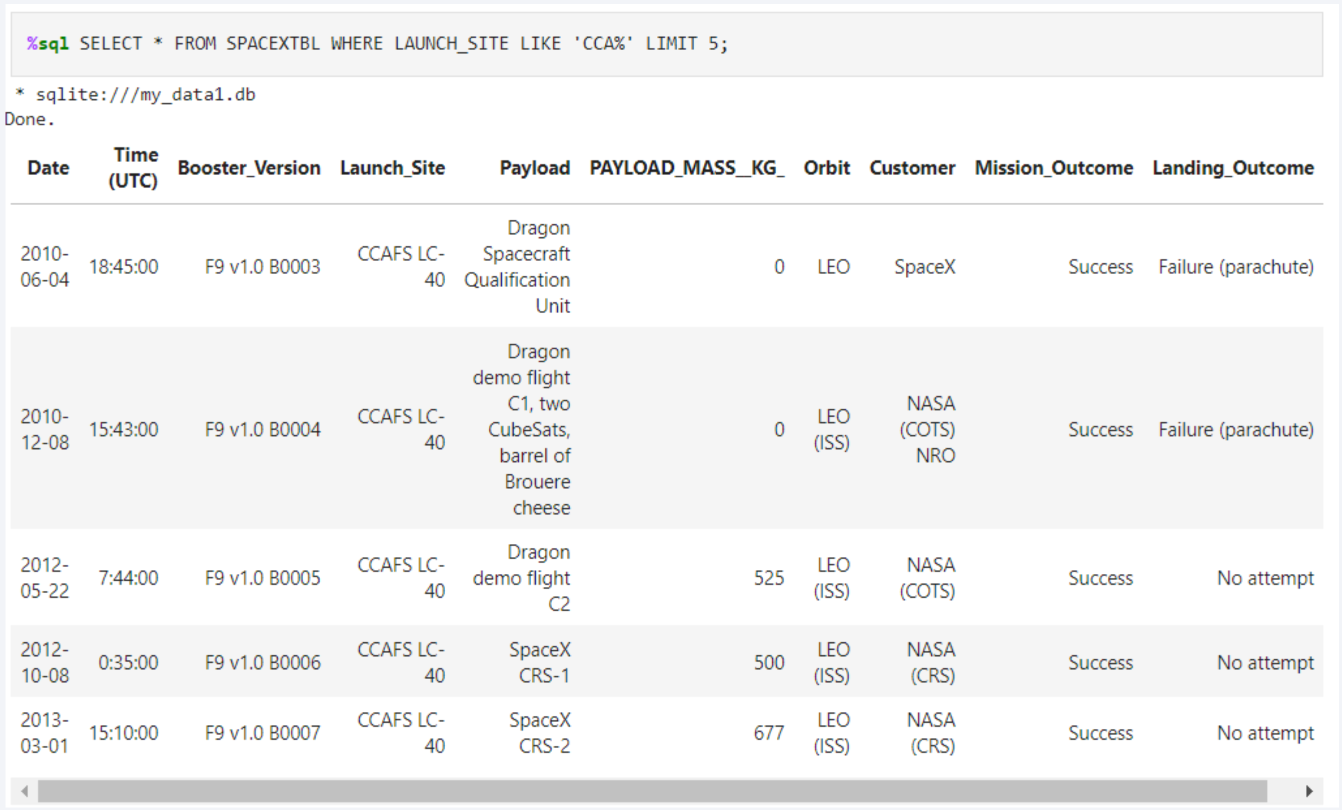


```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL ORDER BY 1;

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

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

Launch Site Names Begin with 'CCA'

- There is 5 records that launch sites name begin “CCA”



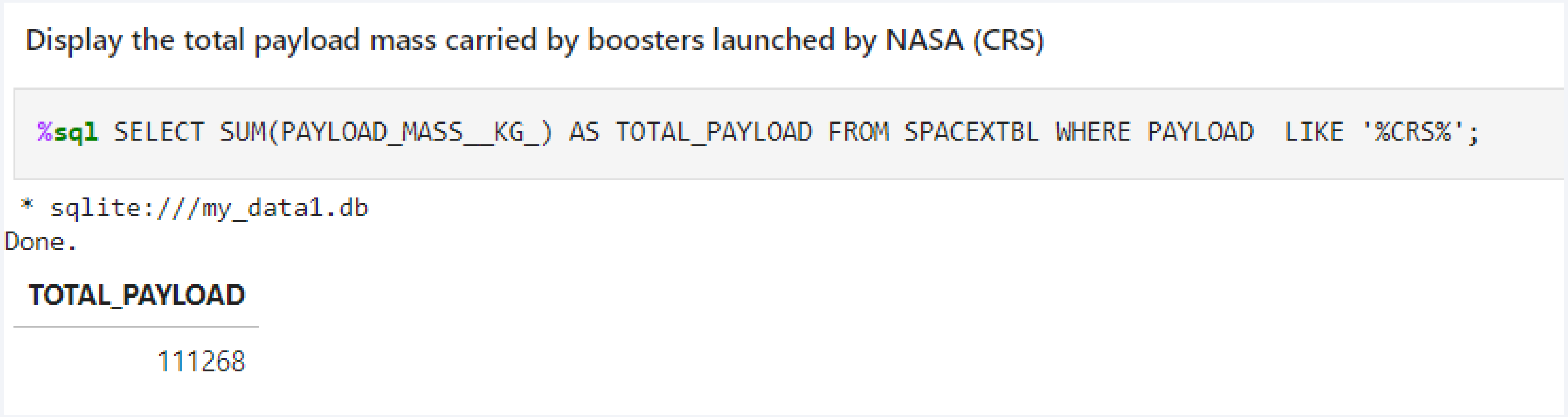
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

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

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

Total Payload Mass

- Total Payload Mass is 111268. It is founded with SQL Sum



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

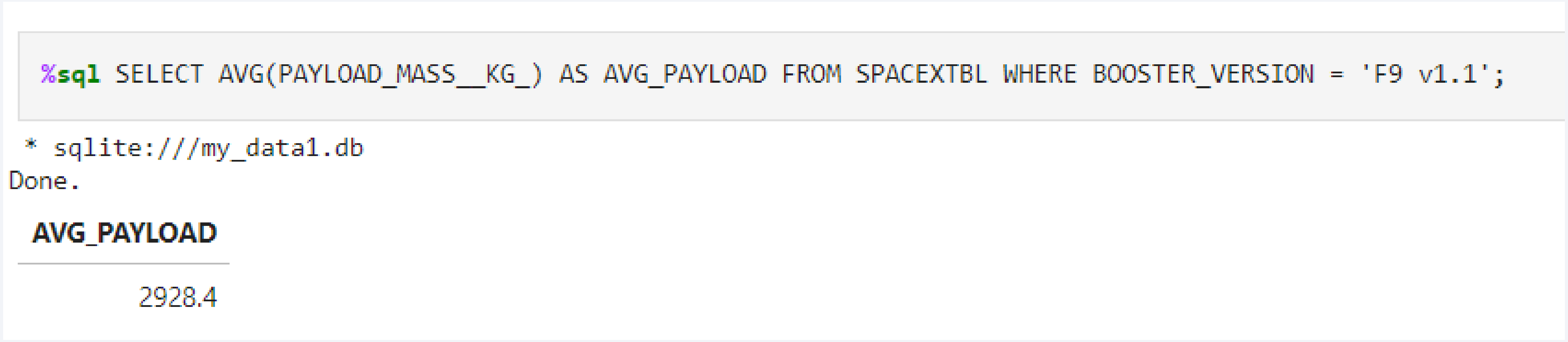
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD FROM SPACEXTBL WHERE PAYLOAD  LIKE '%CRS%';
```

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

TOTAL_PAYLOAD
111268

Average Payload Mass by F9 v1.1

- Findings illustrates that the average payload mass is 2928.4 for F9 V1.1



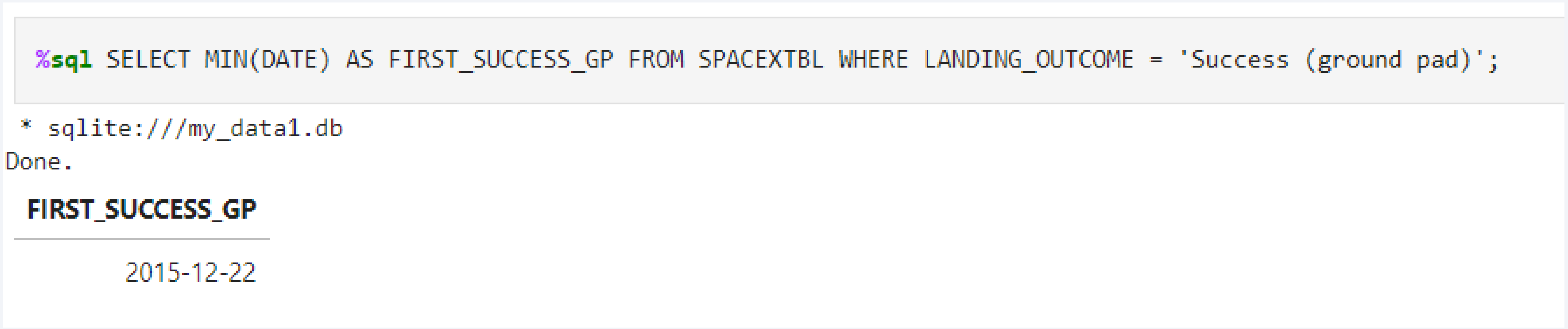
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

AVG_PAYLOAD
2928.4

First Successful Ground Landing Date

- It was observed the first successful landing was occurred December 22 2015



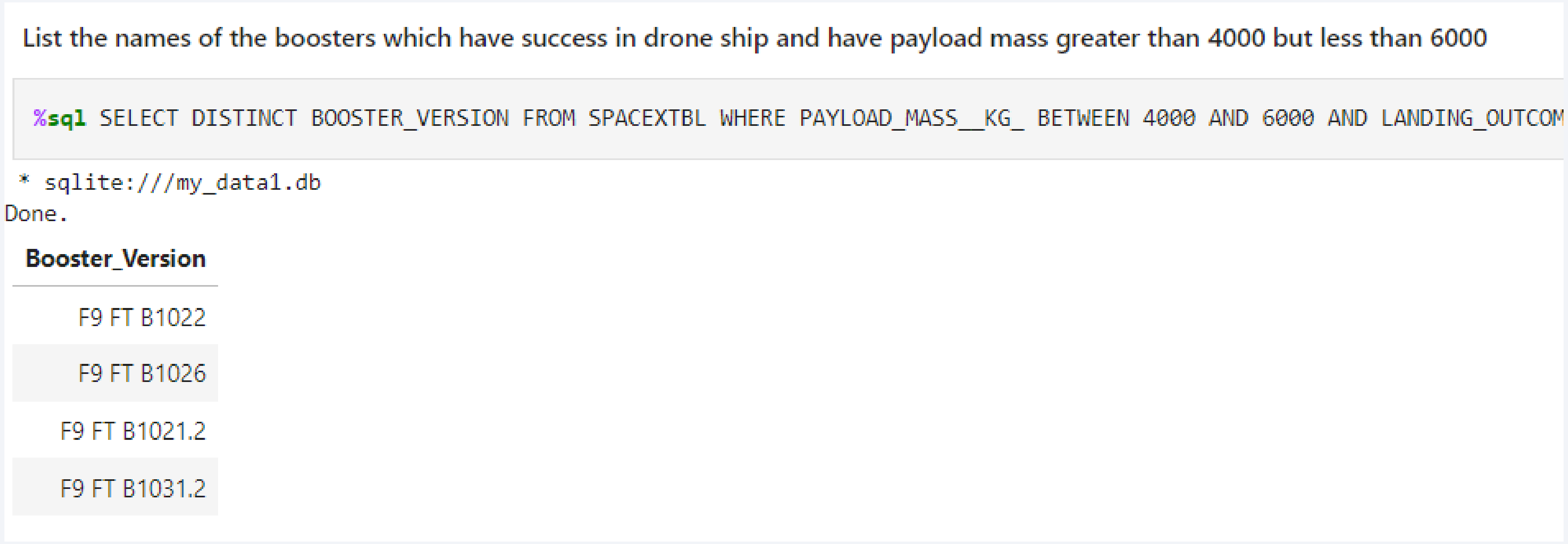
```
%sql SELECT MIN(DATE) AS FIRST_SUCCESS_GP FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (ground pad)';
```

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

FIRST_SUCCESS_GP
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- WHERE used and filtered results. Finally we found the 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

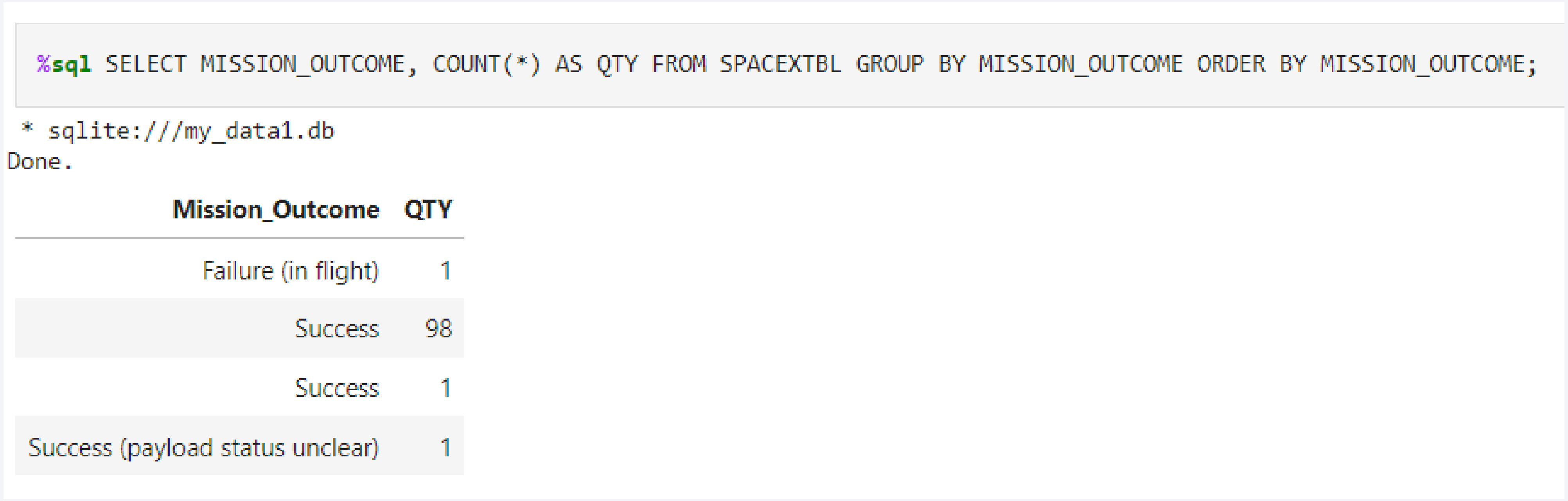
```
%sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000 AND LANDING_OUTCOM
```

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

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

Total Number of Successful and Failure Mission Outcomes

- It is separated fail or success. And there are 100 success and a failure.



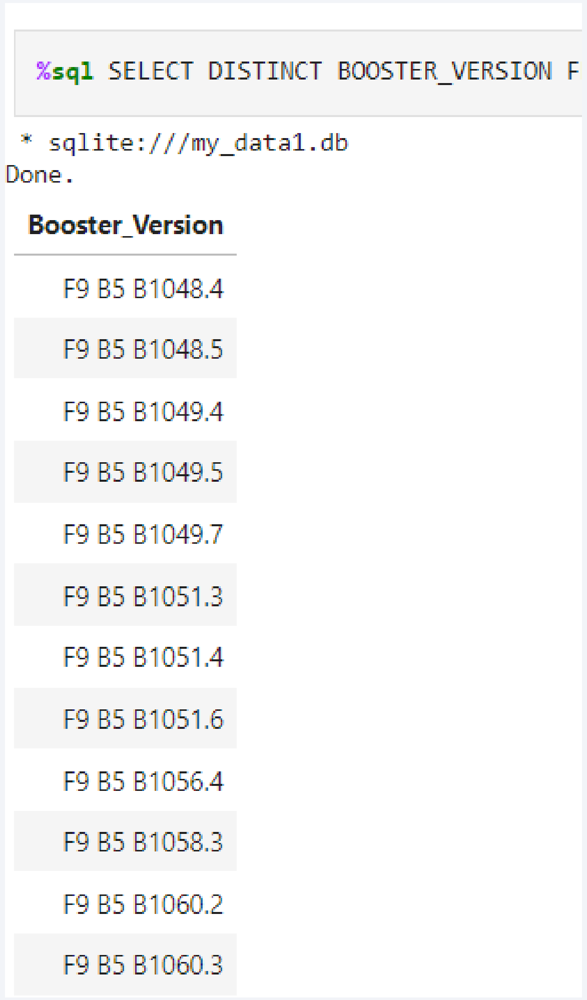
```
%sql SELECT MISSION_OUTCOME, COUNT(*) AS QTY FROM SPACEXTBL GROUP BY MISSION_OUTCOME ORDER BY MISSION_OUTCOME;
```

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

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

Boosters Carried Maximum Payload

• WHERE and MAX() was used to find maximum payload and boosters



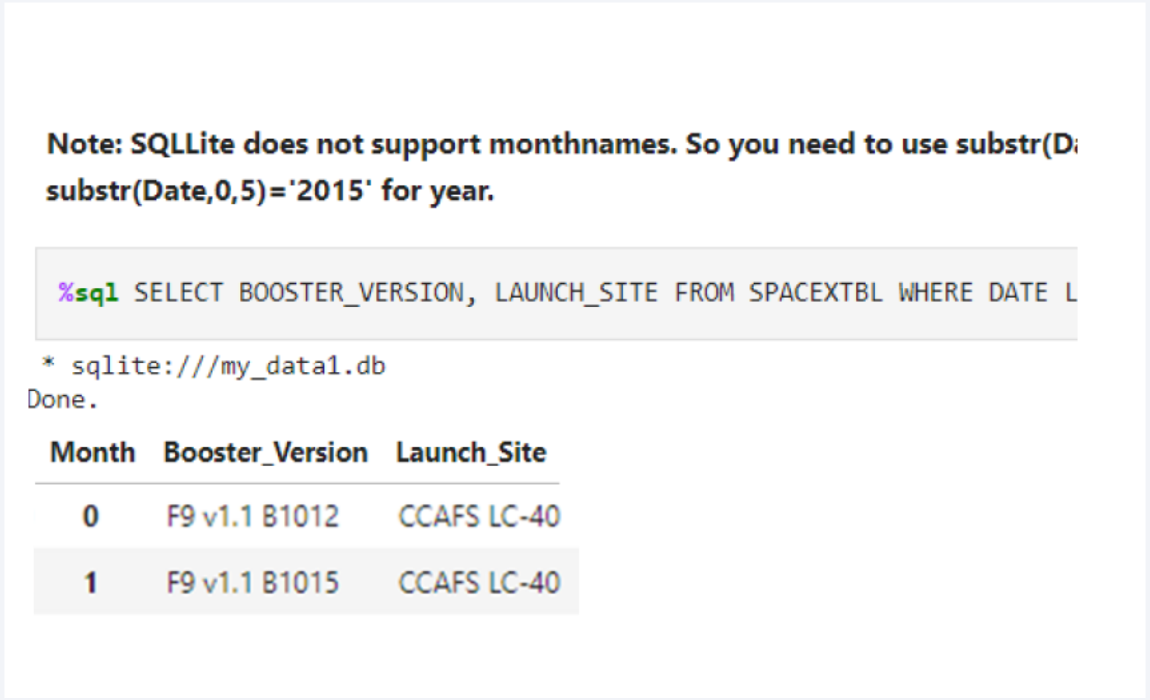
```
%sql SELECT DISTINCT BOOSTER_VERSION F
```

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

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

2015 Launch Records

- Listed the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015



Note: SQLLite does not support monthnames. So you need to use substr(D substr(Date,0,5)='2015' for year.

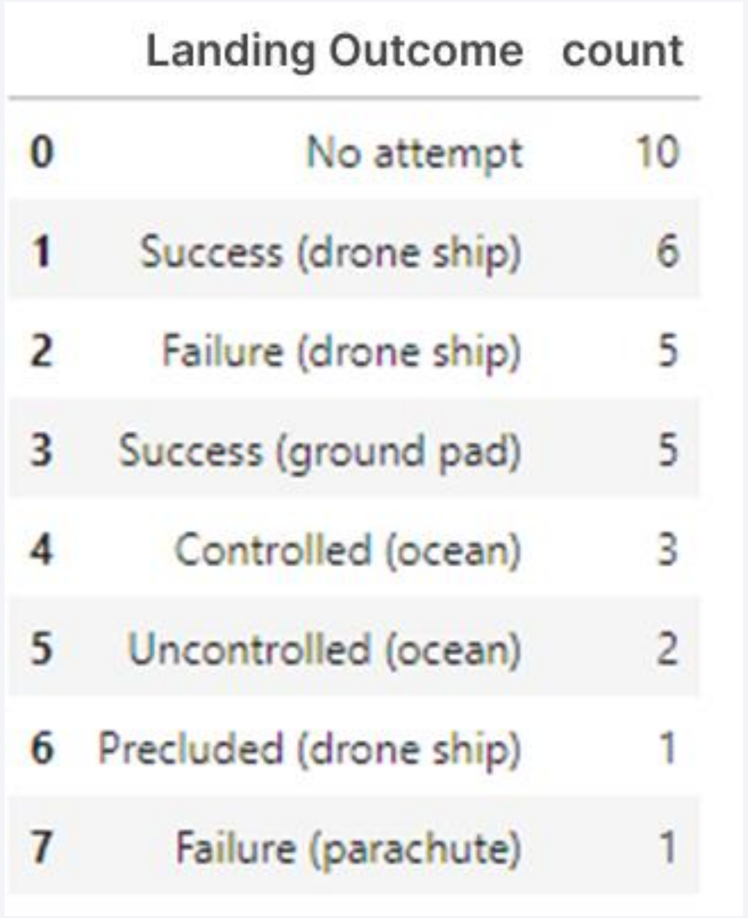
```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE DATE L
```

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

Month	Booster_Version	Launch_Site
0	F9 v1.1 B1012	CCAFS LC-40
1	F9 v1.1 B1015	CCAFS LC-40

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

- SQL query to extract and rank the count of landing outcomes, specifically focusing on successes, between the dates June 4, 2010, and March 20, 2017. We filtered the SpaceX table (SPACEXTBL) to include only entries where the landing outcome contains the term "Success" and falls within the specified date range. By grouping the results based on the landing outcome and counting the occurrences of each outcome, we obtained a count of successful landings. Finally, we arranged the results in descending order based on the count of successful landings, allowing us to observe the most frequent successful landing outcomes during the specified time frame.

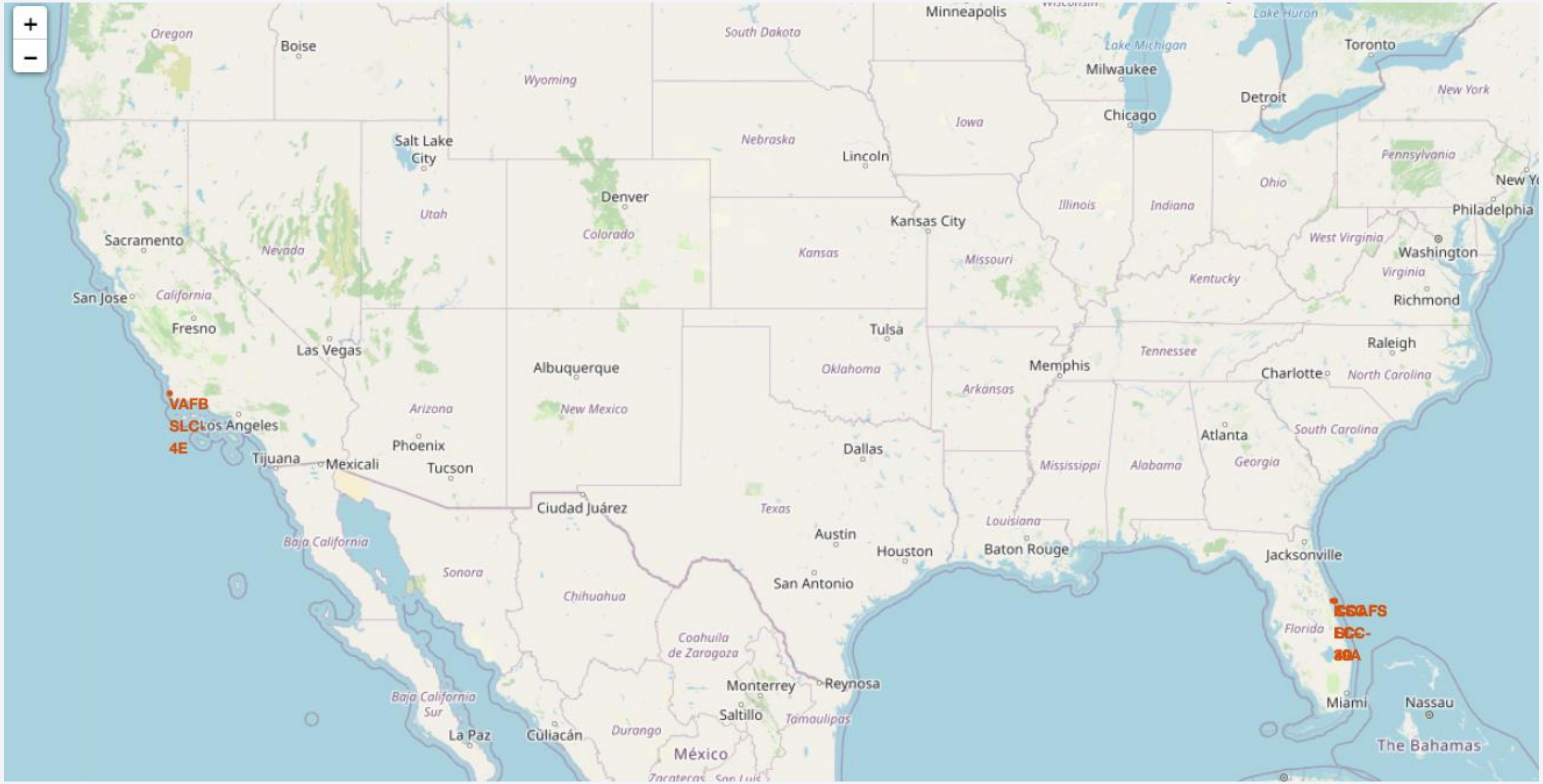


	Landing Outcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

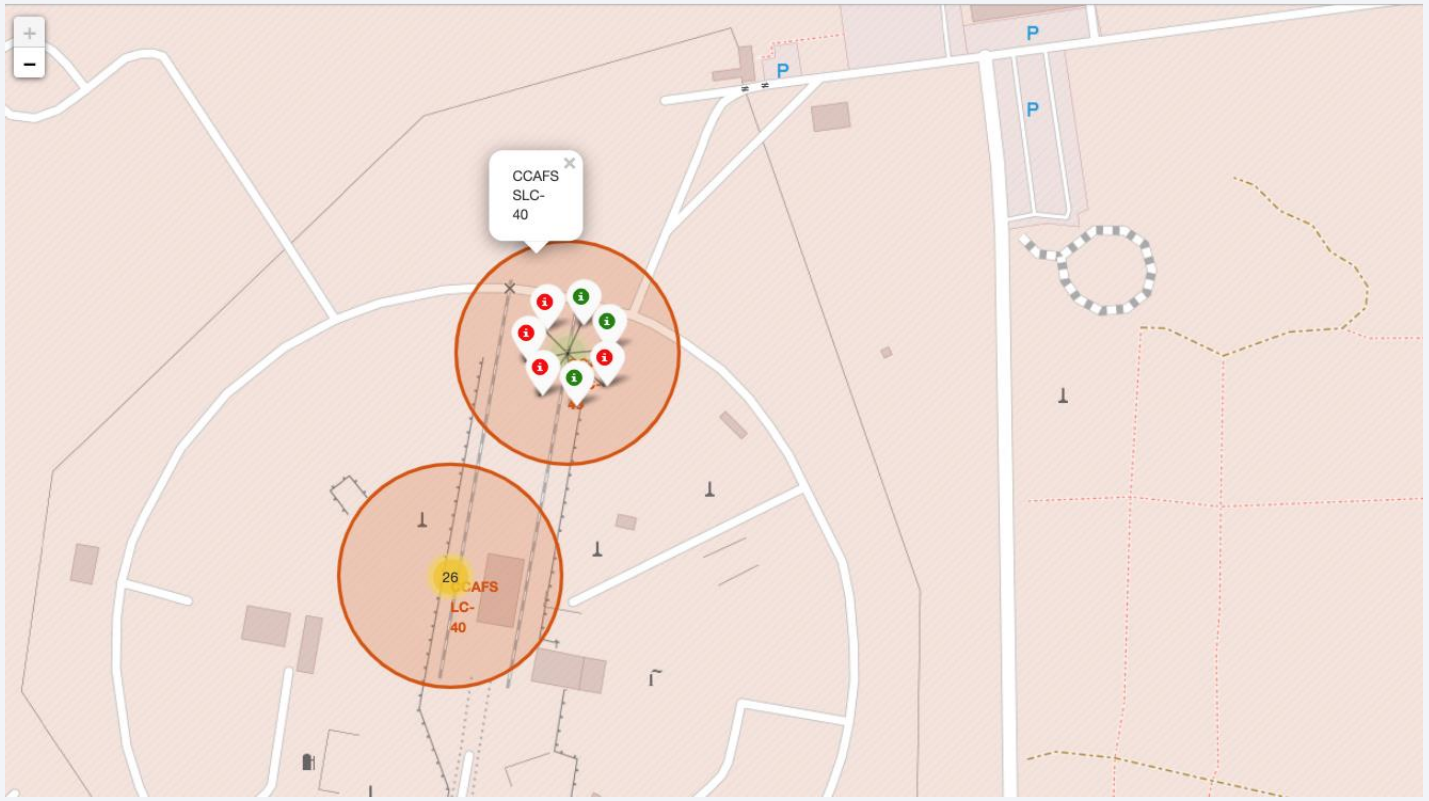
Section 3

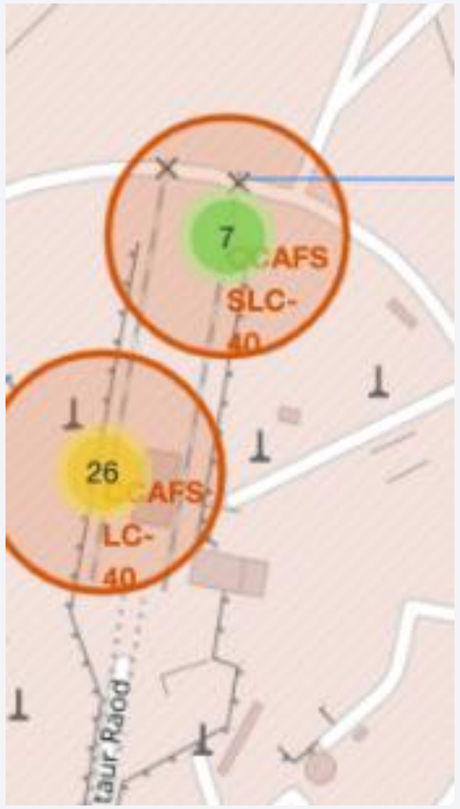
Launch Sites Proximities Analysis

Launch Areas



Looking Launch Sites with Markers

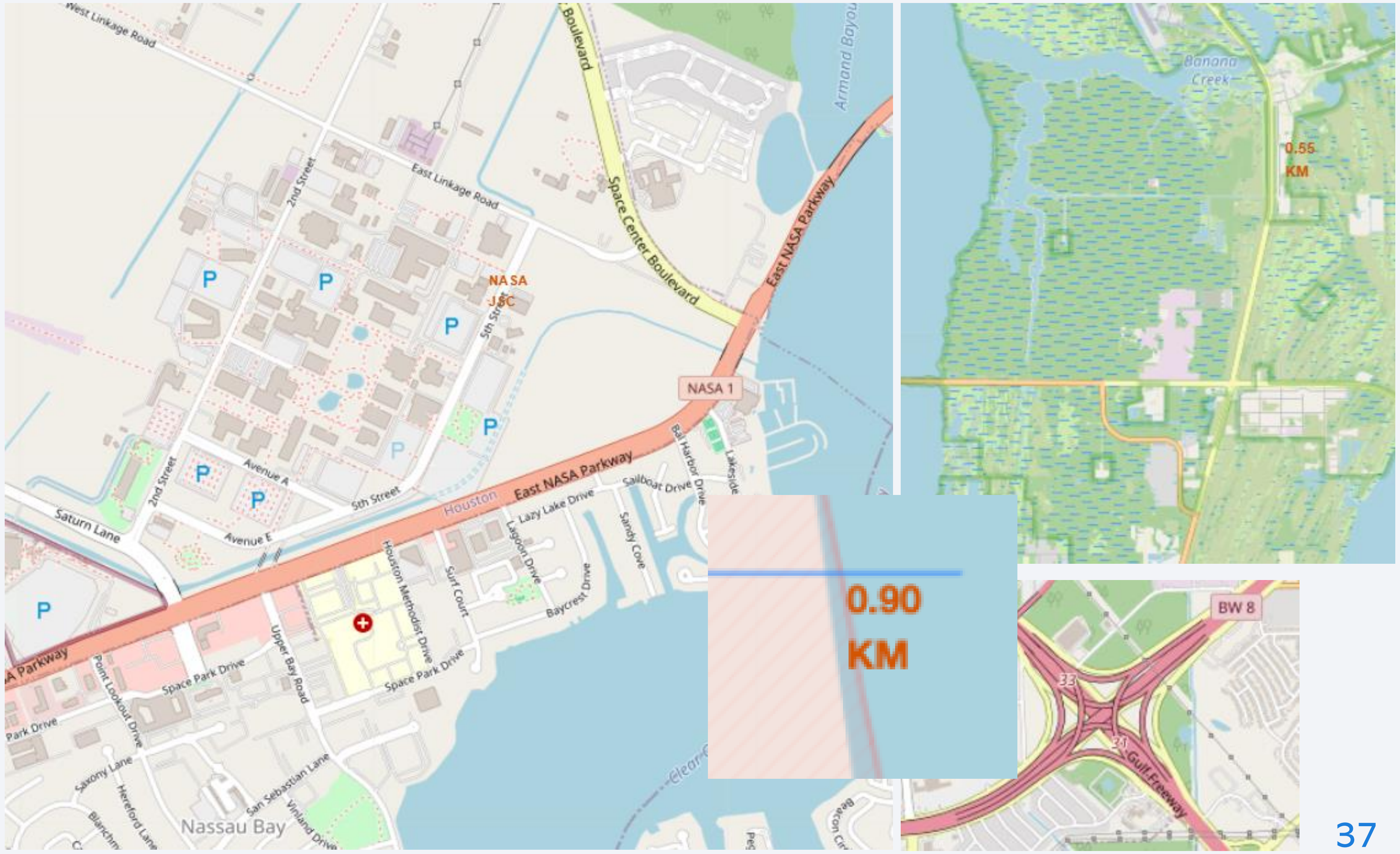




- Florida Launch Site

Distance To Landmarks

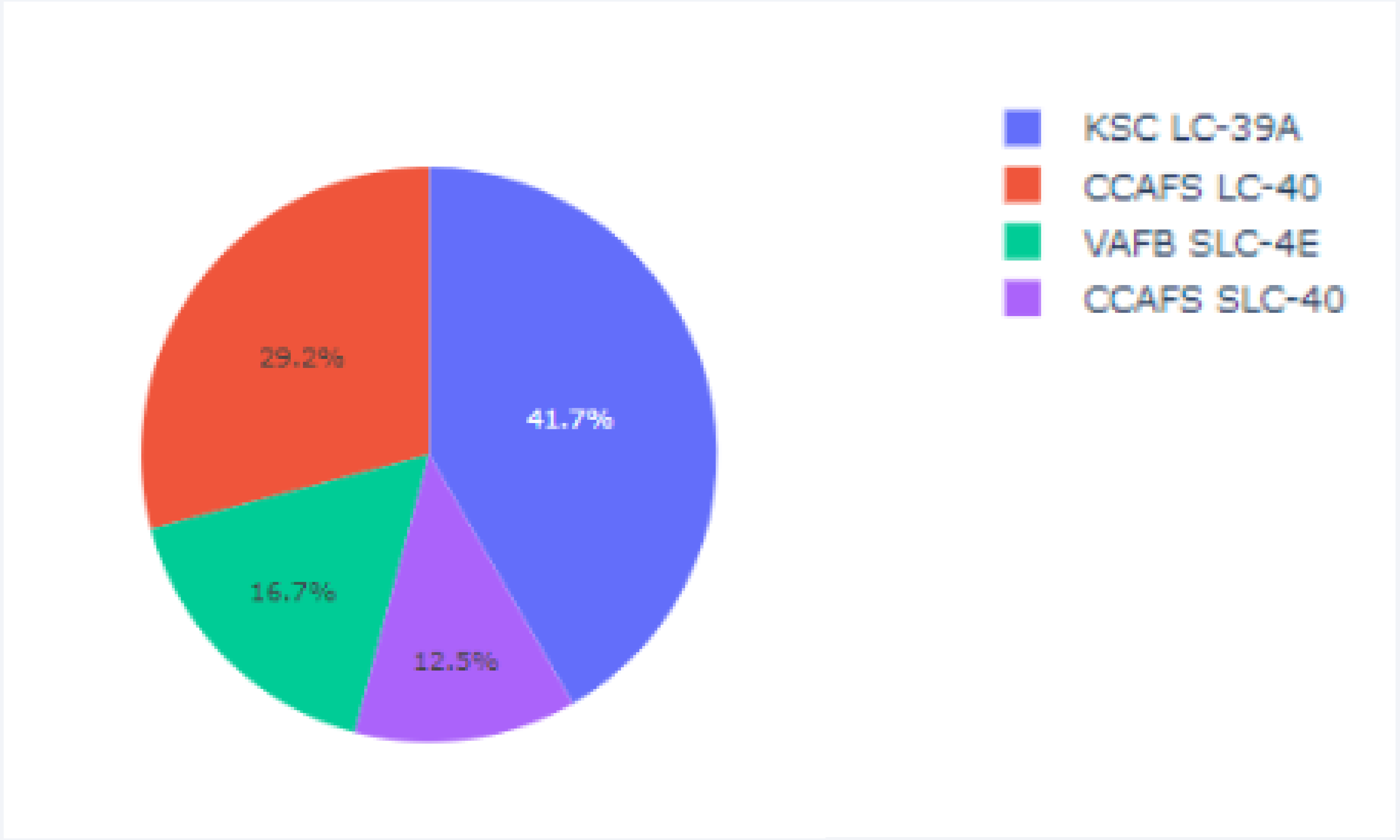
- Are launch sites in close proximity to Railways (NO)
- Are launch sites in close proximity to highwways? (NO)
- Are launch sites in close proximity to coastline? (YES)
- Do launch sites keep certain distance away from cities? (YES)



Section 4

Build a Dashboard with Plotly Dash

Pie Chart, Success Percentage



<Dashboard Screenshot 3>

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

- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

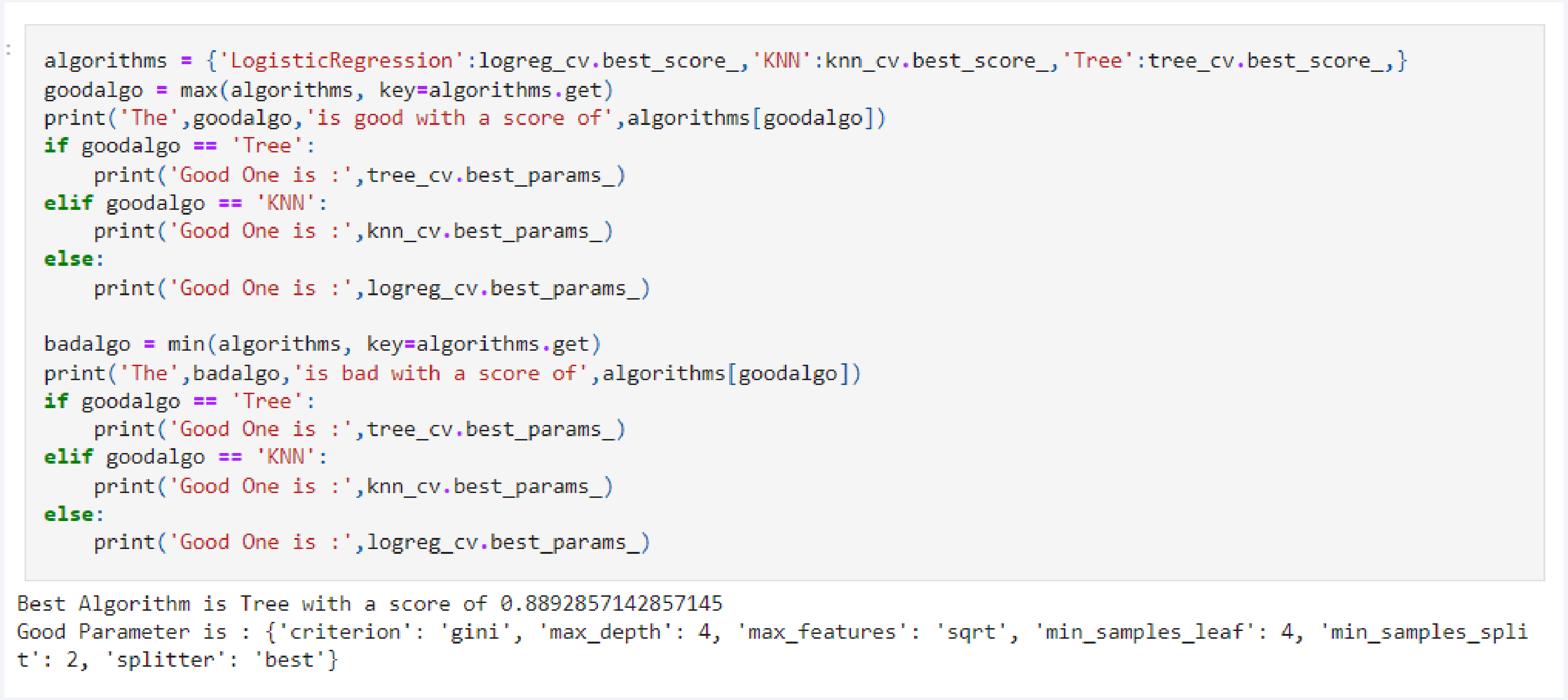
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- TREE has the highest score among three classification.



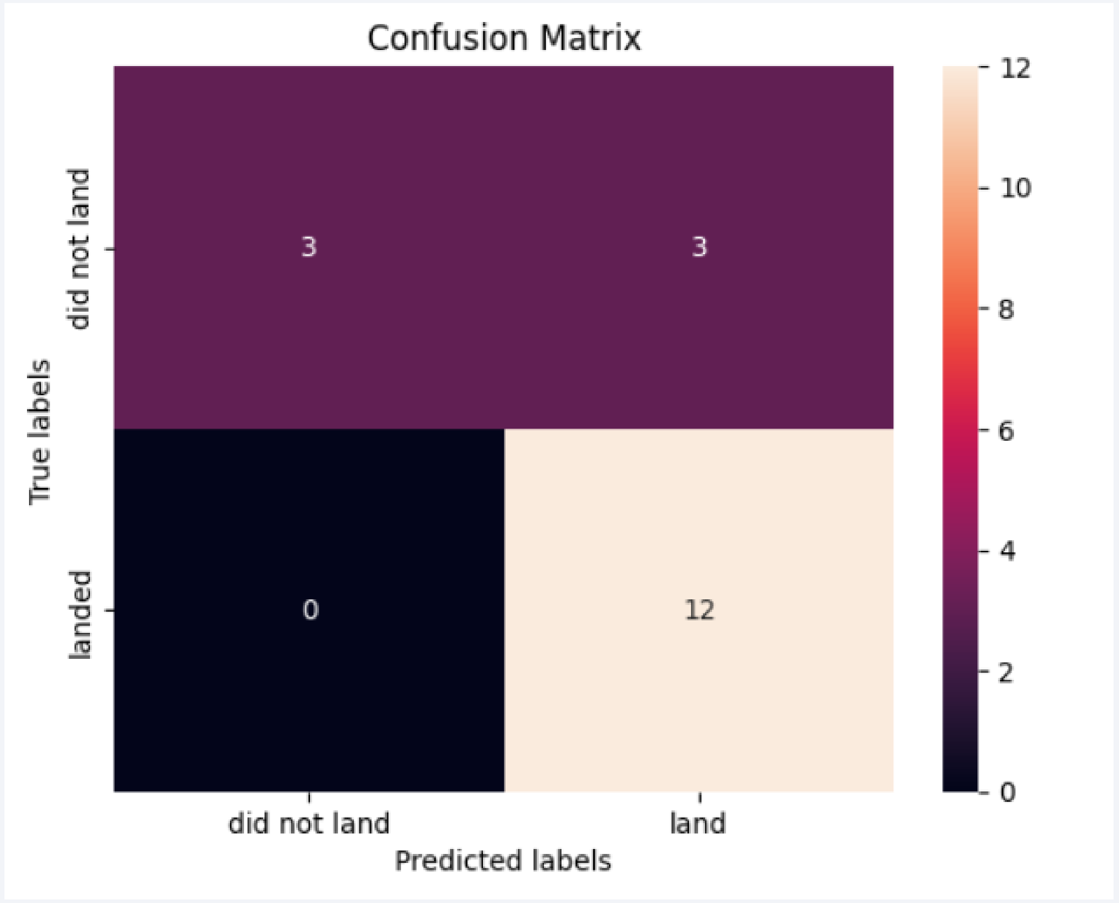
```
algorithms = {'LogisticRegression':logreg_cv.best_score_,'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,}
goodalgo = max(algorithms, key=algorithms.get)
print('The',goodalgo,'is good with a score of',algorithms[goodalgo])
if goodalgo == 'Tree':
    print('Good One is :',tree_cv.best_params_)
elif goodalgo == 'KNN':
    print('Good One is :',knn_cv.best_params_)
else:
    print('Good One is :',logreg_cv.best_params_)

badalgo = min(algorithms, key=algorithms.get)
print('The',badalgo,'is bad with a score of',algorithms[goodalgo])
if goodalgo == 'Tree':
    print('Good One is :',tree_cv.best_params_)
elif goodalgo == 'KNN':
    print('Good One is :',knn_cv.best_params_)
else:
    print('Good One is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.8892857142857145
Good Parameter is : {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_spli
t': 2, 'splitter': 'best'}
```

Confusion Matrix

The confusion matrix for the decision tree classifier indicates its ability to differentiate between various classes, suggesting proficiency in classification.

However, a notable issue lies in the occurrence of false positives, where unsuccessful landings are incorrectly Identified as successful landings by the classifier.



Conclusions

- Space X offers cost-effective Falcon 9 rocket launches due to its first stage reusability.
- Predicting successful first stage landings is crucial for launch cost estimation and competitive bidding.
- Data analysis revealed higher success rates for heavy payloads in certain orbits.
- Machine learning models, including Logistic Regression, SVM, Decision Trees, and KNN, were evaluated.
- The Decision Tree classifier achieved the highest accuracy (88.93%) but struggled with false positives.
- Other models like Logistic Regression, SVM, and KNN showed accuracies around 83-85%.
- Confusion matrices helped assess model performance.
- Hyperparameter tuning was done using GridSearchCV.
- Decision Tree emerged as the most effective model for predicting landing outcomes.
- These findings offer insights into Space X's rocket launch operations.

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!