IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Orhan Sönmez
March 3, 2023

Outline

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

Introduction

- Project background and context

  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. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. In this project, we will try to predict if the Falcon 9 first stage will land successfully.

- Problems you want to find answers
  - Factors that determine whether the rocket will land successfully
  - The relationship between variables and their impact on a successful landing
  - What is the best combination of the conditions needed to make a successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX RESTful API and web scraping from Wikipedia.
- Perform data wrangling
  - We use one-hot encoding for transforming categorical features into numerical features.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

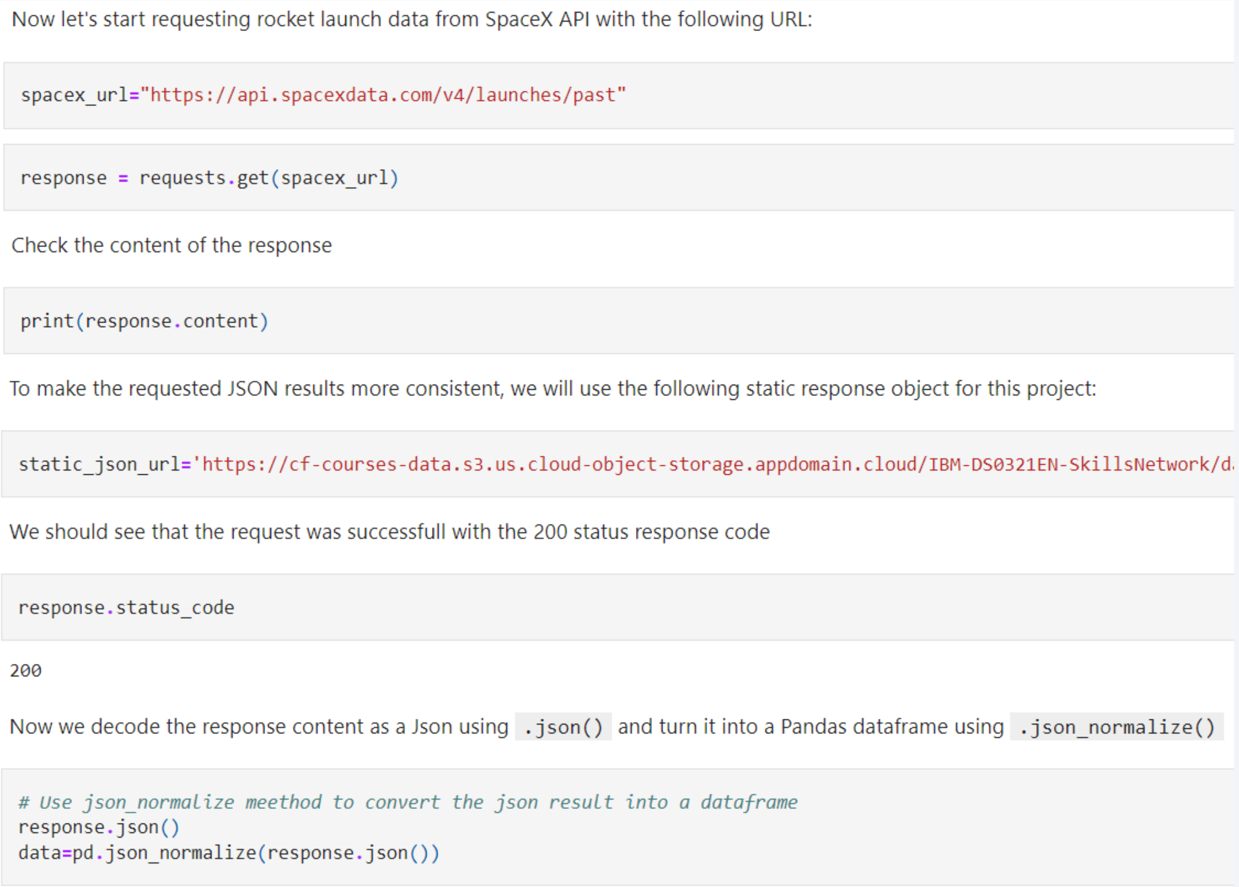
The information was gathered using a RESTful API by using the GET request method. After that, we decoded the response by utilizing the json() function and transformed it into a Pandas DataFrame via the json_normalize() function. The data was then purified, missing values were examined, and any missing values were filled in where necessary. Furthermore, we engaged in web scraping with the aid of the BeautifulSoup toolkit. The aim was to retrieve the launch records as an HTML table, parse, and turn it into a Pandas DataFrame for a more thorough study.

Data Collection – SpaceX API

- We used the GET request to collect data from the SpaceX API, cleansed the retrieved data, and performed some fundamental data wrangling techniques.

- GitHub URL:

https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/data-collection-api.ipynb



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

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

Check the content of the response

```
print(response.content)
```

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

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/d
```

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

```
response.status_code
```

```
200
```

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

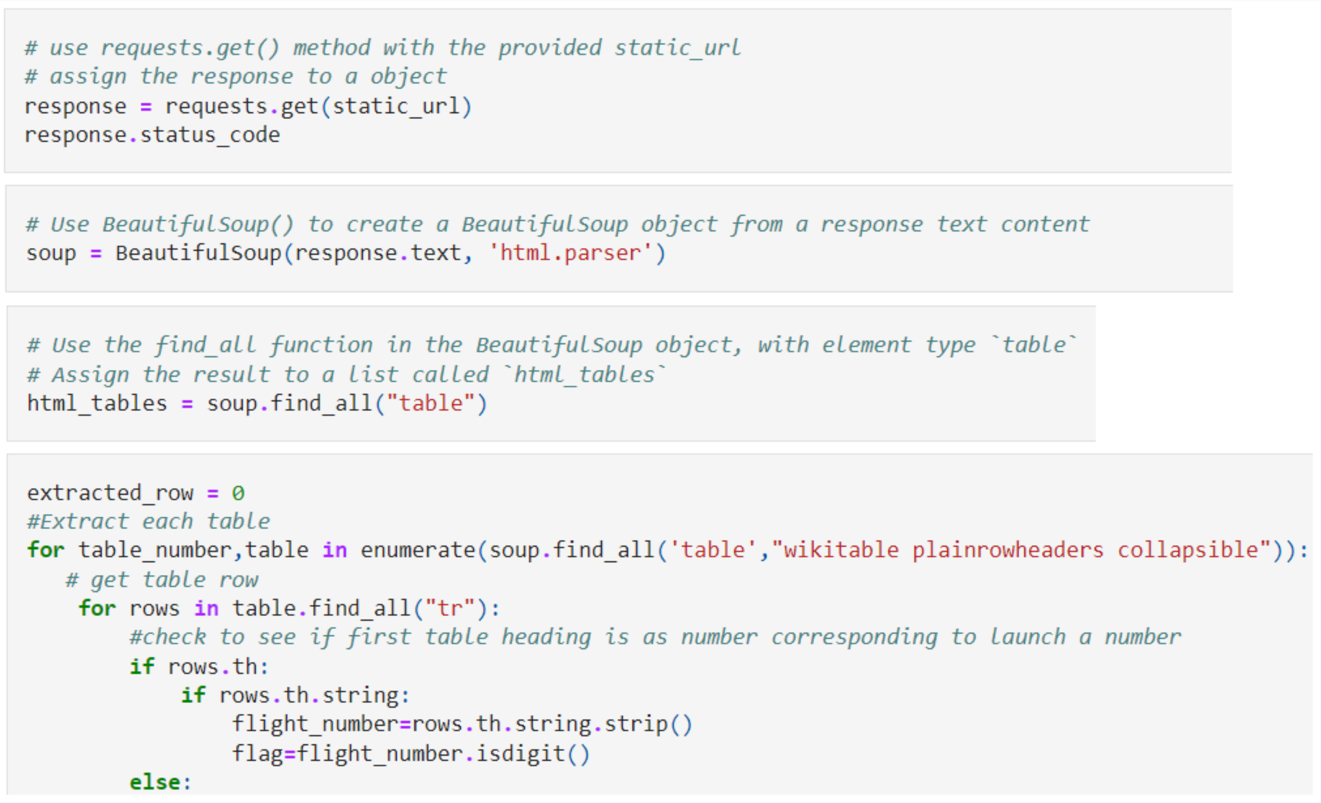
```
# Use json_normalize meethod to convert the json result into a dataframe
response.json()
data=pd.json_normalize(response.json())
```

Data Collection - Scraping

- We request the Falcon9 Launch Wikipedia page from the URL. Then, we create a BeautifulSoup from the HTML response. Finally, we extract all column names from the HTML header.

- GitHub URL:

https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/webscraping.ipynb



```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
response.status_code
```

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, 'html.parser')
```

```
# 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")
```

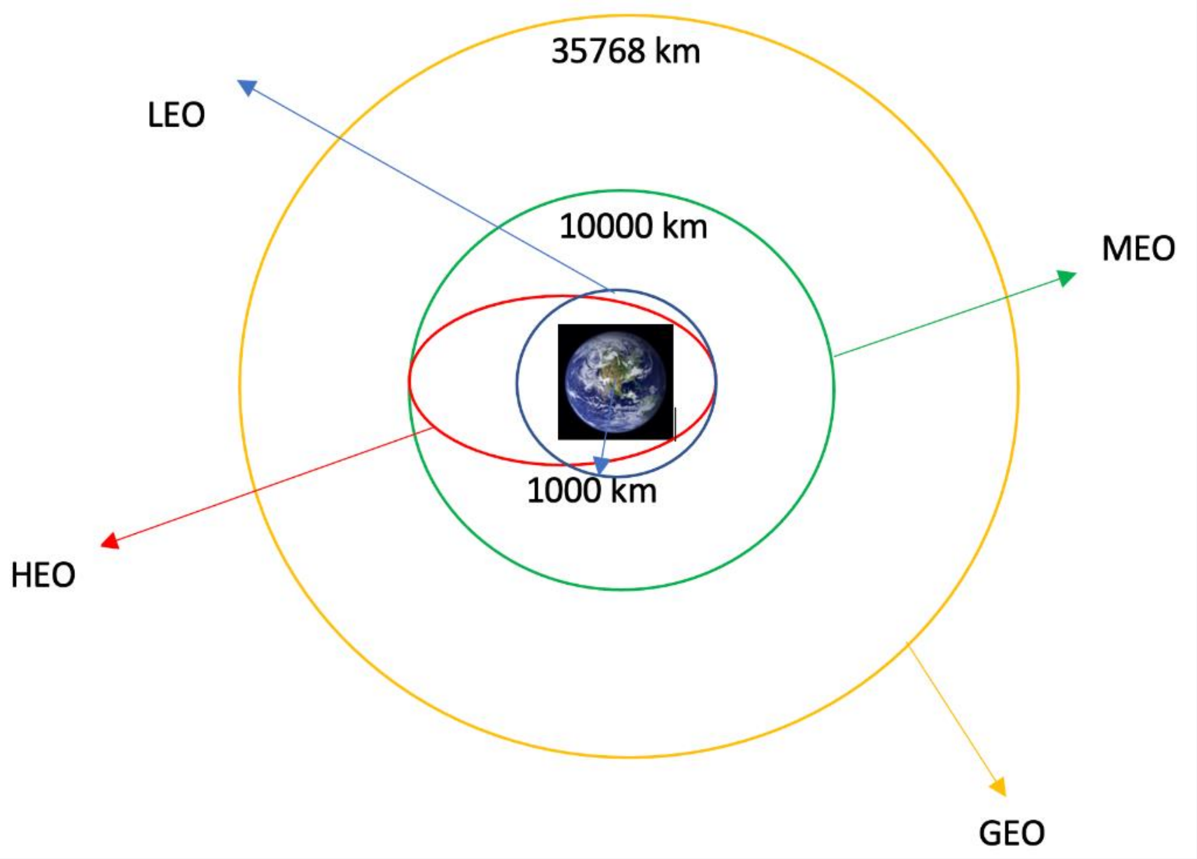
```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
 # get table row
  for rows in table.find_all("tr"):
    #check to see if first table heading is as number corresponding to launch a number
    if rows.th:
      if rows.th.string:
        flight_number=rows.th.string.strip()
        flag=flight_number.isdigit()
    else:
```

Data Wrangling

- We begin by determining the quantity of launches at each site, then evaluate the amount and prevalence of mission outcome for each orbit. Next, we establish a landing outcome column from the outcome column. This approach simplifies further investigation, representation, and prognostication. Ultimately, we will save the conclusions as a CSV document.

- GitHub URL:

https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/data_wrangling.ipynb



EDA with Data Visualization

Our initial approach was to utilize a scatter plot to identify the relation among the features such as:

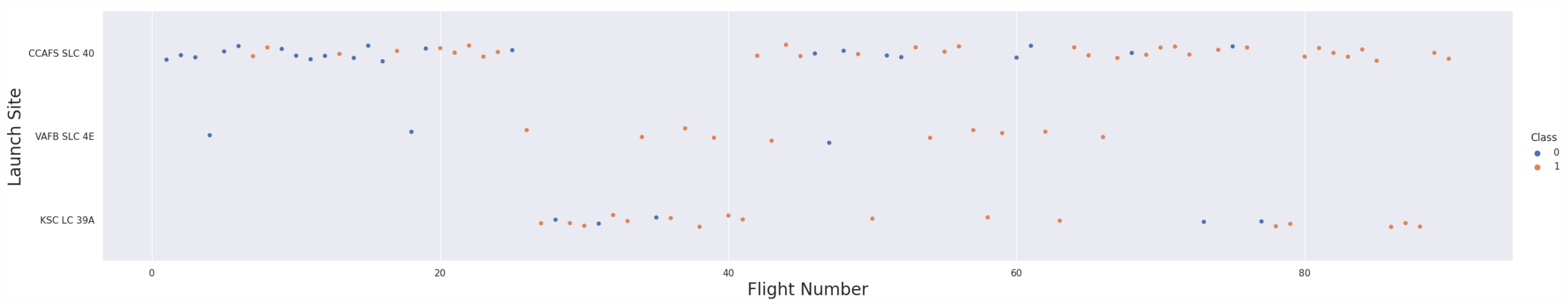
- Payload – Flight Number
- Flight Number – Launch Site
- Payload – Launch Site
- Flight Number – Orbit Type
- Payload – Orbit Type

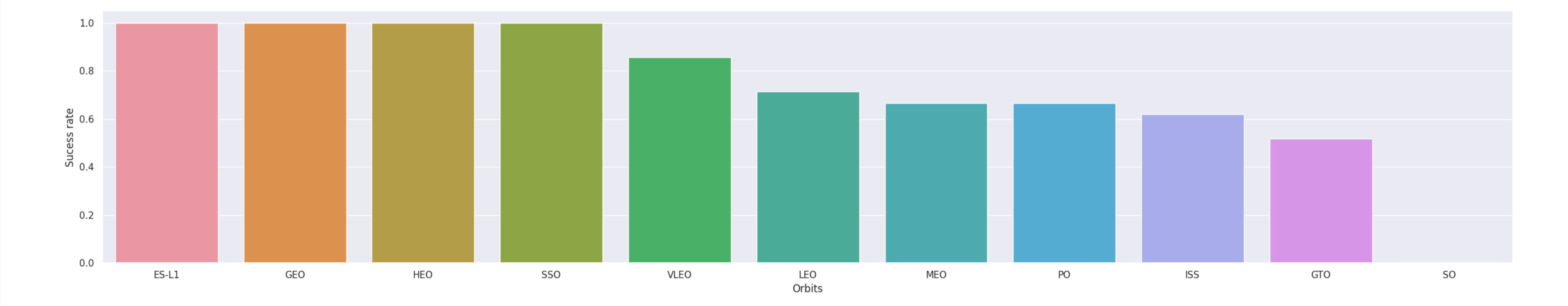
After detecting the relations using the scatter plot, we subsequently employed supplementary visual aids, such as bar graphs and line plots, for additional examination.

See the next slide for the plots.

GitHub URL: https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/eda-dataviz.ipynb

EDA with Data Visualization





GitHub URL: https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/eda-dataviz.ipynb

EDA with SQL

We executed the following queries to acquire more insight into the dataset.

- 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
- List the date when the first successful 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 outcomes
- List the names of the booster_versions which have carried the maximum payload mass using a subquery
- 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 successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

GitHub URL: https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/eda-sql.ipynb

Build an Interactive Map with Folium

In order to produce an interactive map displaying the launch data, we retrieved the latitude and longitude coordinates for every launch site and added a circular label around each one containing the site's name.

Furthermore, we classified the launch_outcomes(failure, success) dataframe into categories 0 and 1 and depicted them on the map as red and green markers via MarkerCluster().

We calculated the distance of the launch sites to various landmark to find an answer to the questions such as

- How near are the launch sites to railroads, highways, and coastlines?
- What is the distance of the launch sites to the cities?

GitHub URL: https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/IntVisAnW_Folium.ipynb

Build a Dashboard with Plotly Dash

- Using Plotly dash, we constructed an interactive dashboard.
- We generated pie charts to display the overall number of launches by certain sites.
- We produced a scatter plot illustrating the relations between Outcome and Payload Mass (Kg) for various booster versions.

GitHub URL:
https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/spacex_dash_app.py

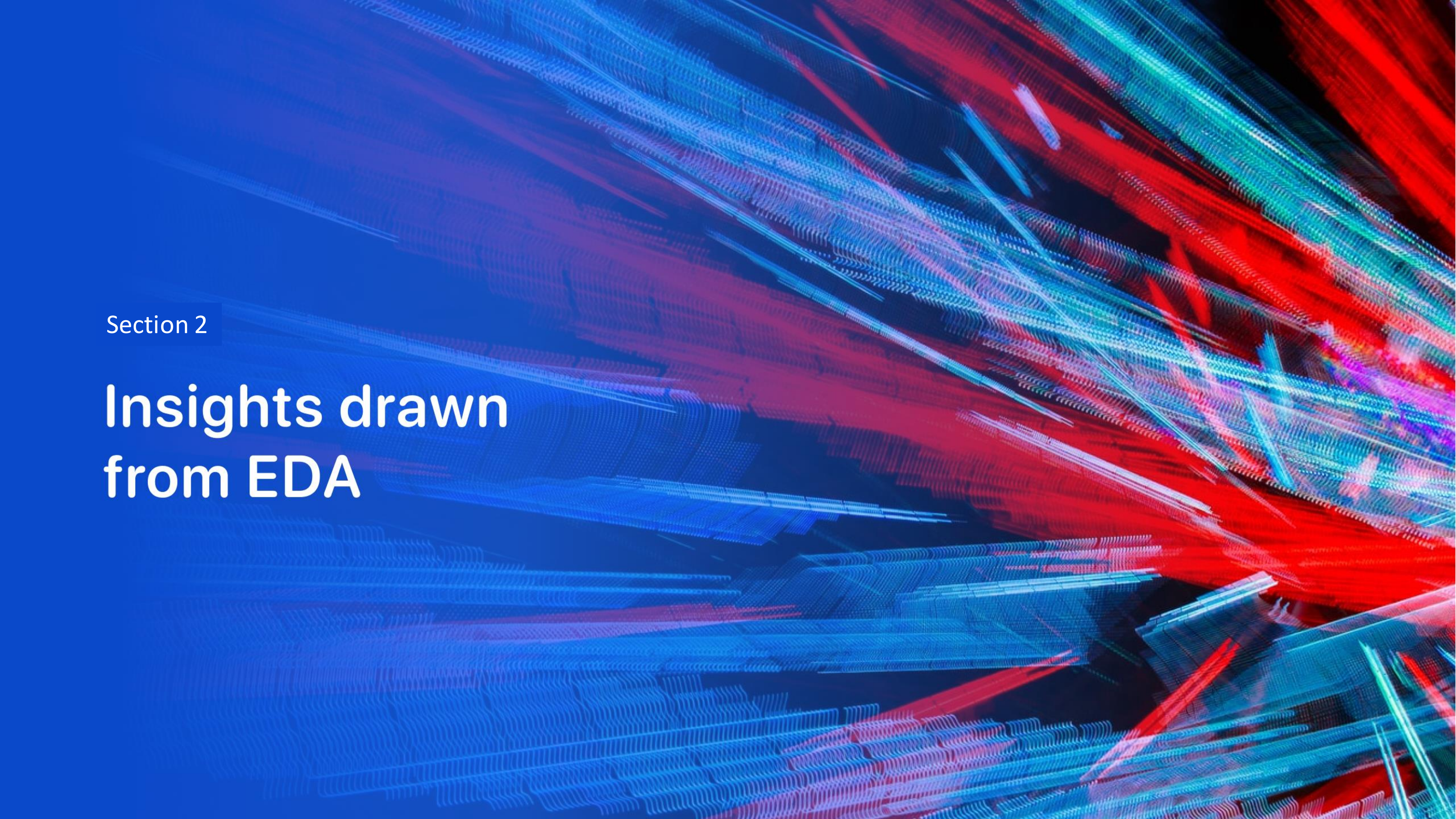
Predictive Analysis (Classification)

- Utilizing numpy and pandas, we imported the data, converted the data, and split it into training and testing subsets.
- We implemented various machine learning models and adjusted several hyperparameters using GridSearchCV.
- We employed accuracy as our model's metric and improved the model by performing feature engineering and algorithm tuning.
- We determined the top-performing classification model.

GitHub URL:
https://github.com/orhansonmeztr/IBM_Data_Science_Capstone/blob/main/ML_Prediction.ipynb

Results

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

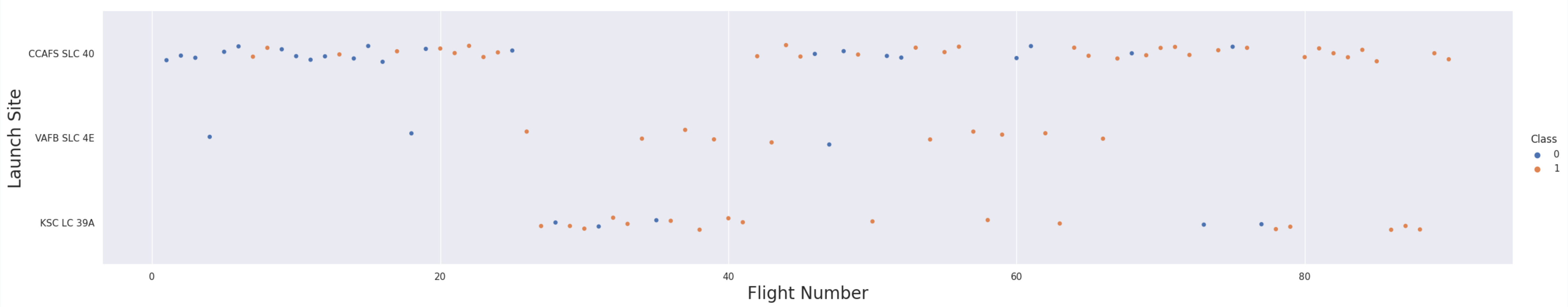


Section 2

Insights drawn from EDA

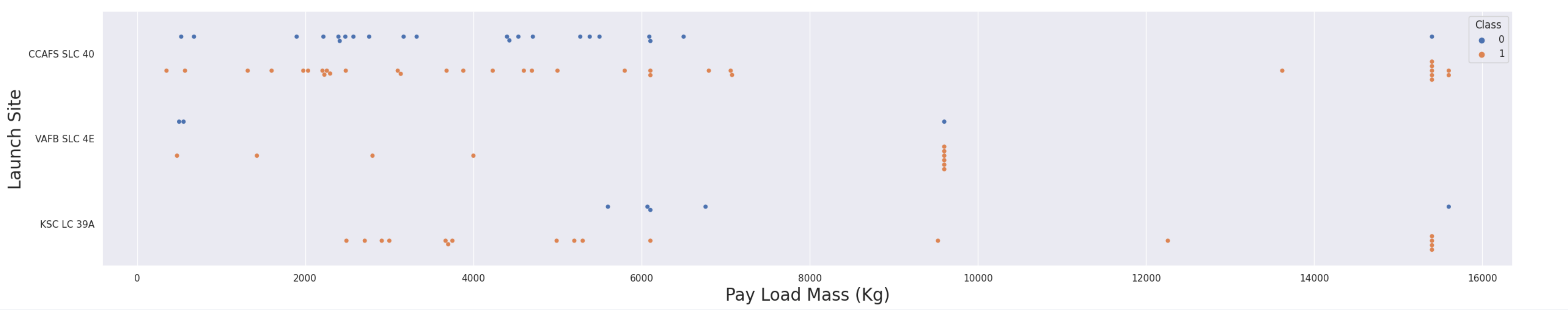
Flight Number vs. Launch Site

- This scatter plot shows that the larger the flights amount of the launch site, the greater the success rate will be.



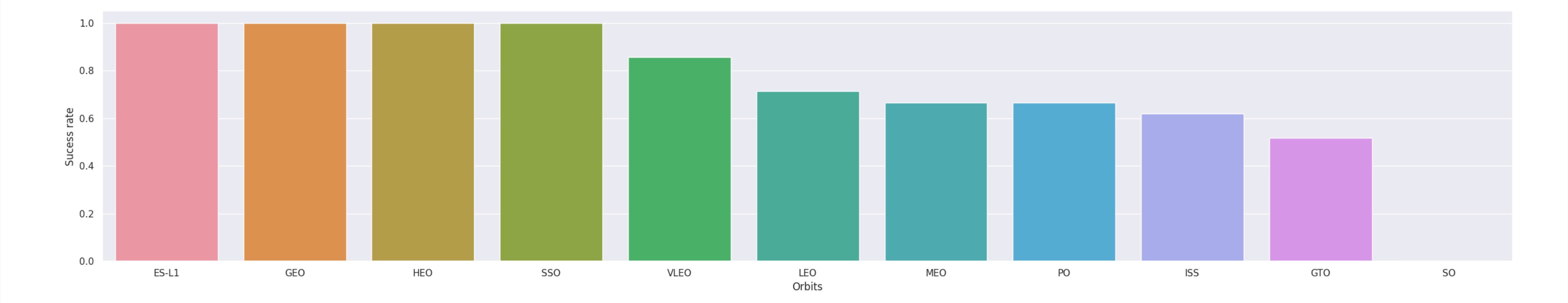
Payload vs. Launch Site

- The scatter plot indicates that an increase in success rate occurs when the payload mass exceeds 8000 kg.



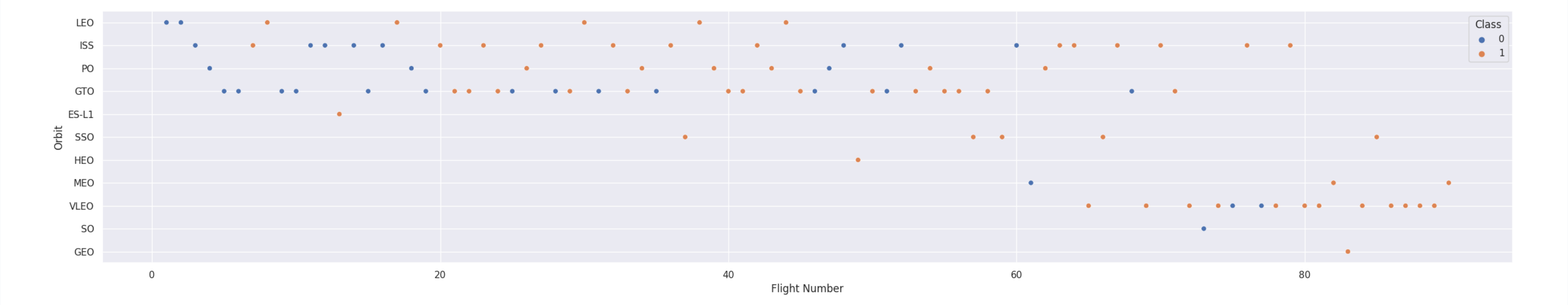
Success Rate vs. Orbit Type

- This bar chart illustrates how different orbits can affect the landing outcomes, as some orbits, such as ES-L1, GEO, HEO and SSO have a success rate of 100%, while the SO orbit has a 0% success rate.



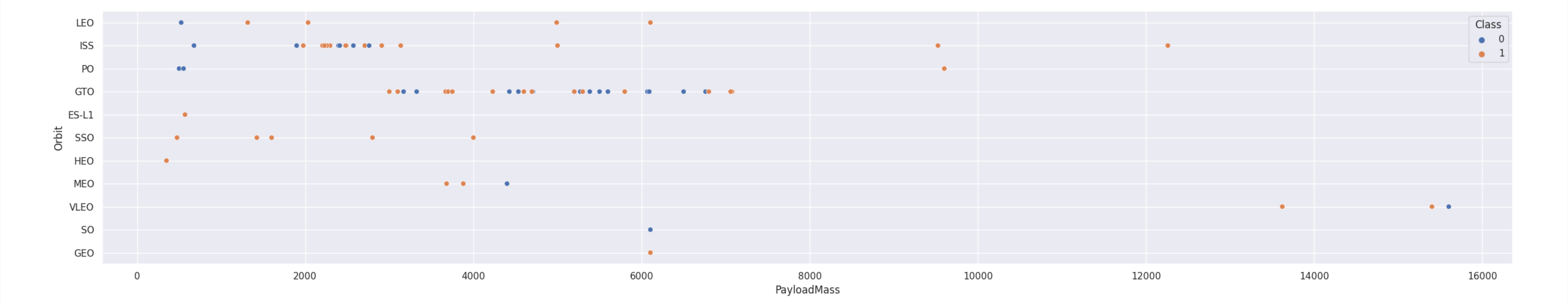
Flight Number vs. Orbit Type

- This scatter plot illustrates that, in general, as the number of flights on each orbit increases, the success rate also increases (especially for LEO orbit), except for the GTO orbit, which shows no relation between these attributes.



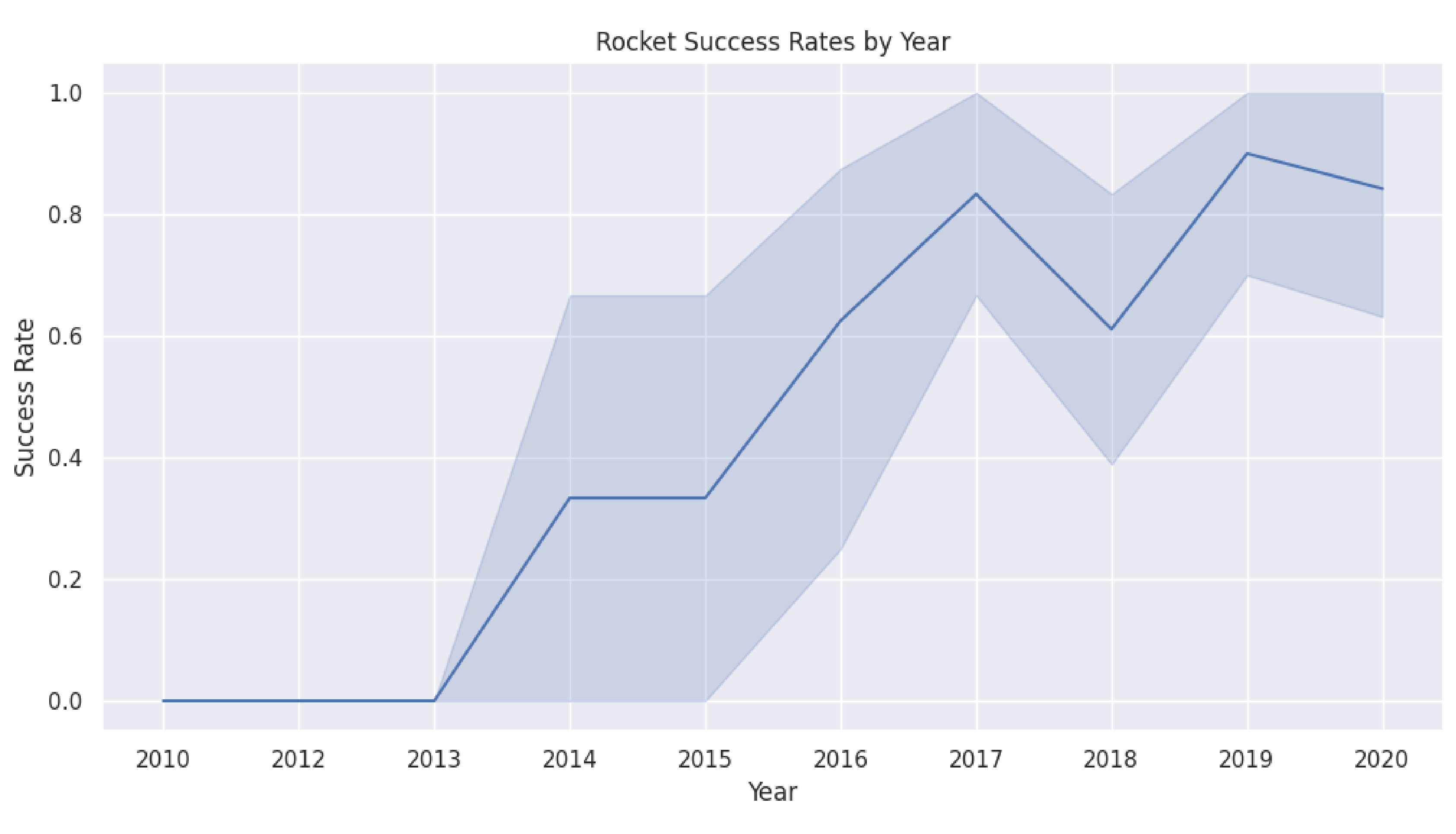
Payload vs. Orbit Type

- We can notice that massive payloads positively affect the success of landings in LEO, PO, and ISS orbits. However, they have a negative influence on VLEO and MEO orbit launches.



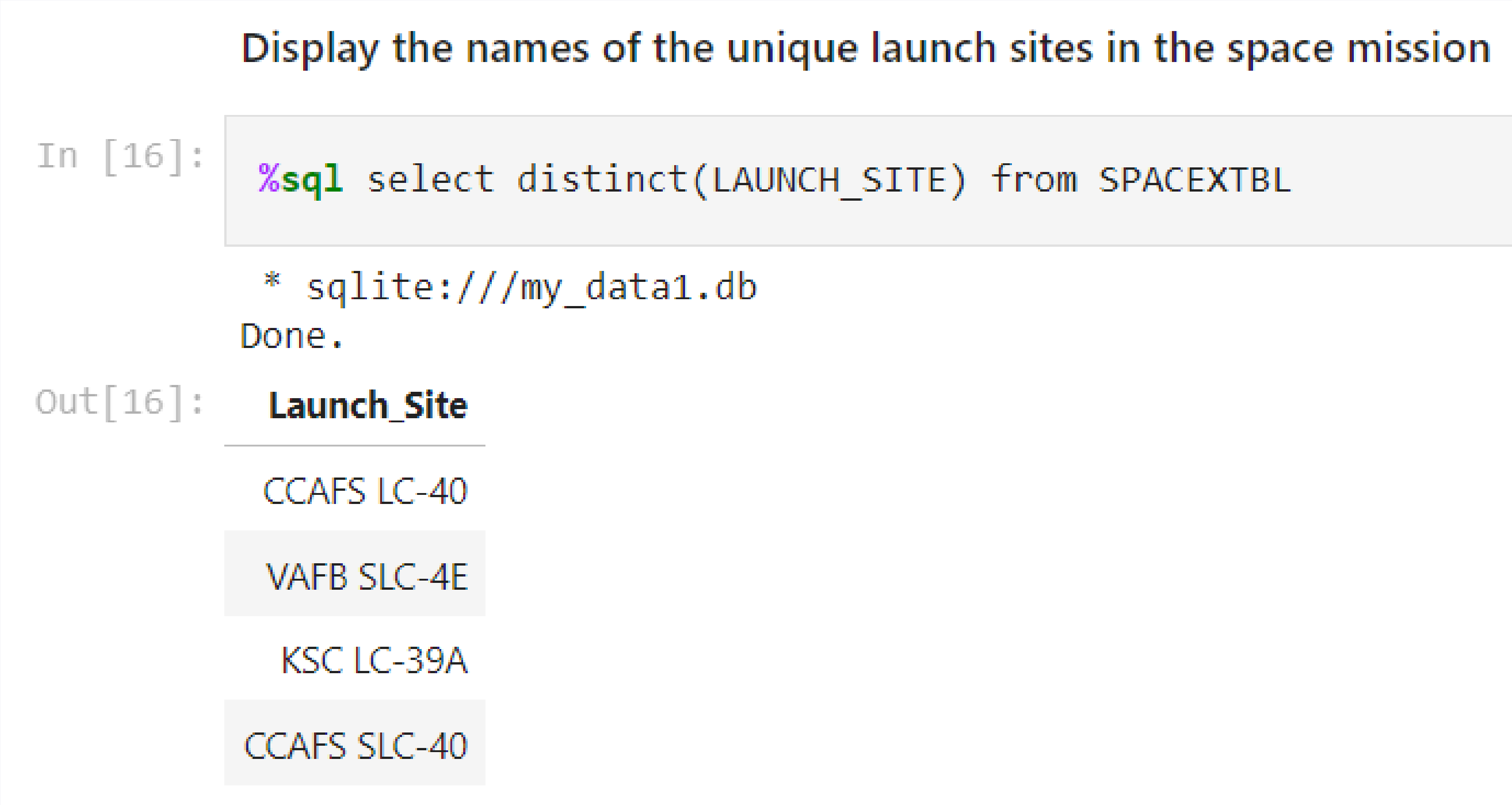
Launch Success Yearly Trend

- This plot reveals that the success rate has been steadily increasing from 2013 to 2020.



All Launch Site Names

- We used the key word distinct to show only unique launch sites from the SpaceX data.



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

In [16]:

```
%sql select distinct(LAUNCH_SITE) from SPACEXTBL
```

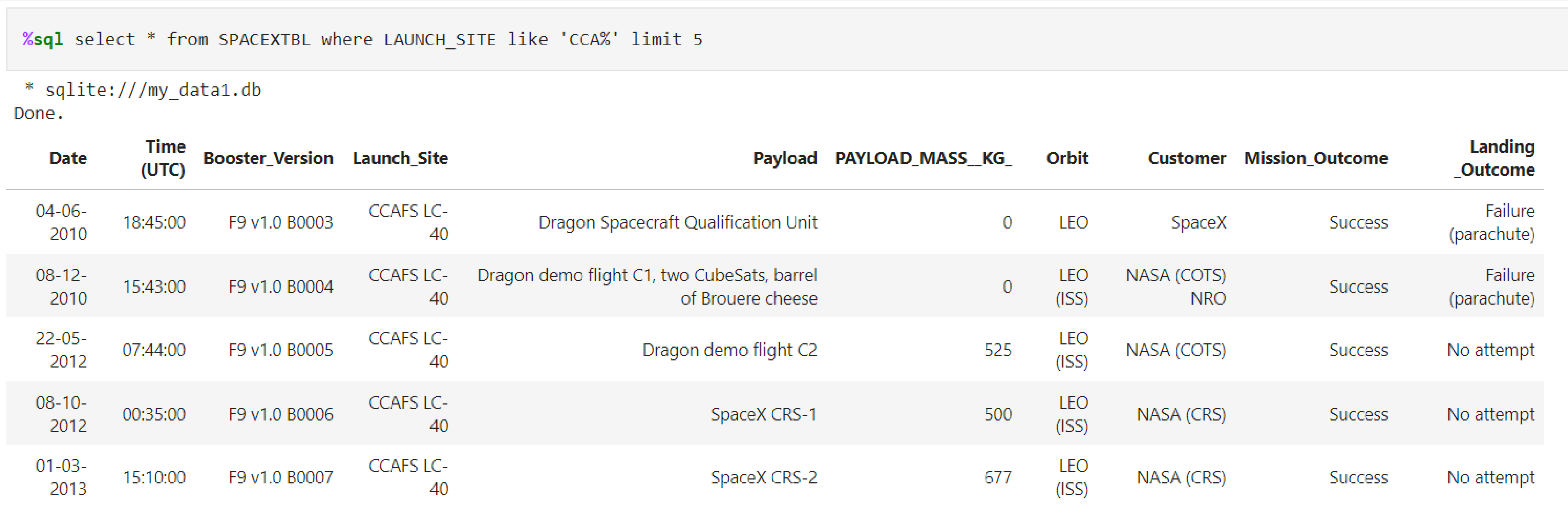
```
 * sqlite:///my_data1.db
Done.
```

Out[16]:

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

Launch Site Names Begin with 'CCA'

- We used the following query to see the 5 records where launch site starts with the string 'CCA'.



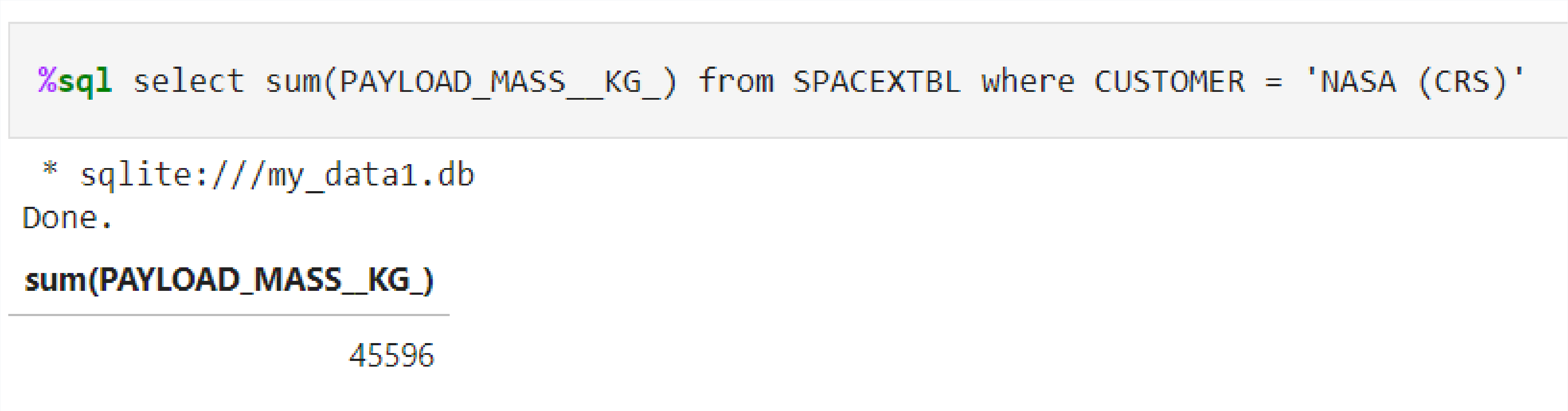
```
%sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Using the query provided below, we were able to calculate the total payload mass of boosters launched by NASA (CRS) to be 45,596



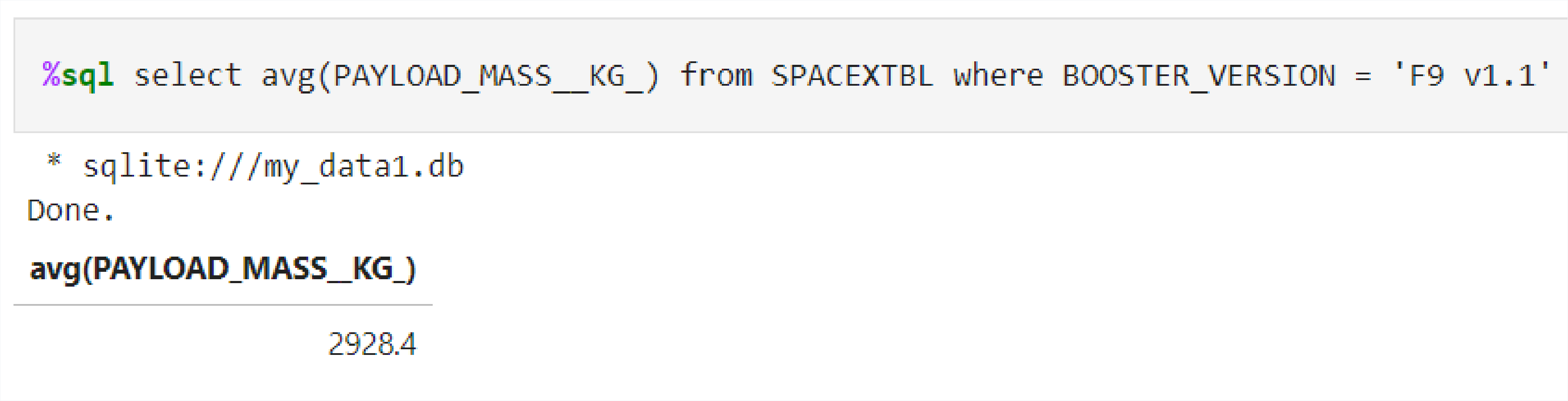
```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- Using the query provided below, we were able to calculate the average payload mass carried by booster version F9 v1.1 to be 2,928.4



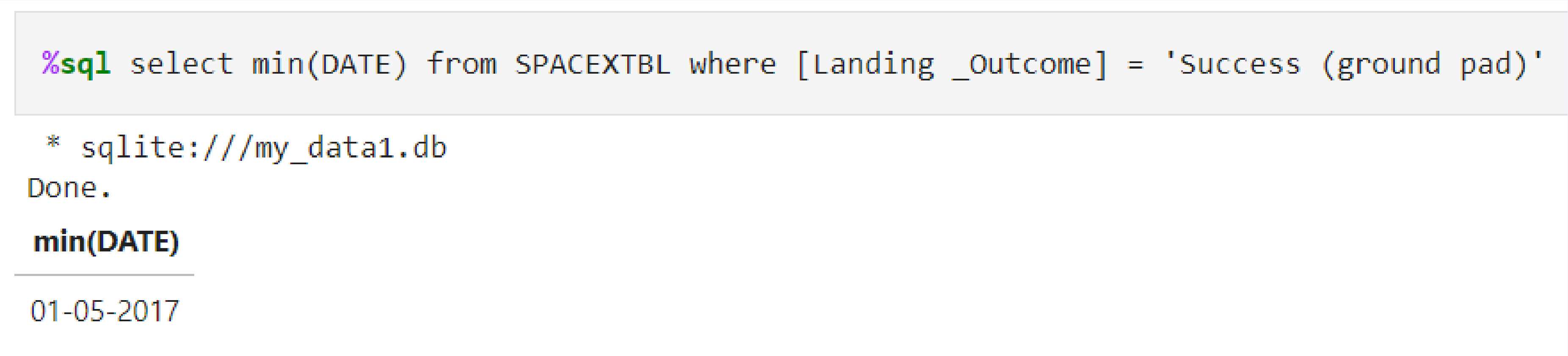
```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- We noticed that the first landing success on the ground pad occurred on May 1, 2017.



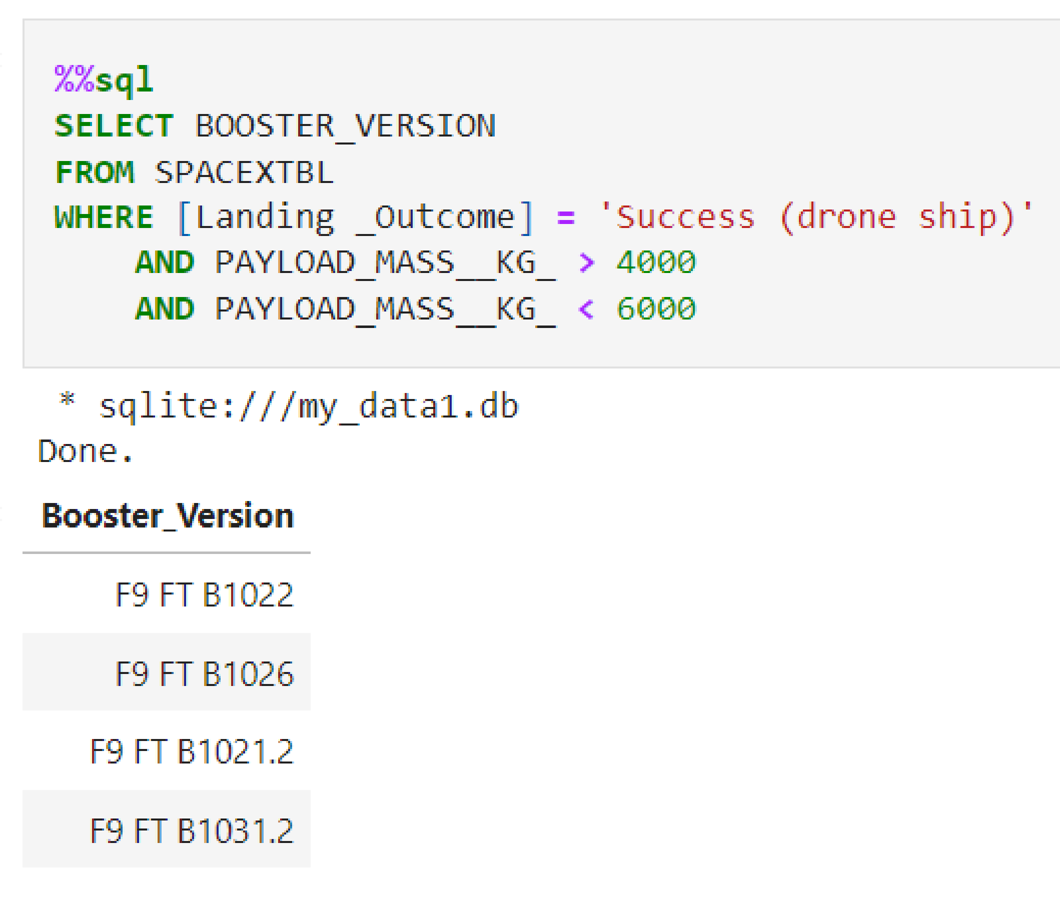
```
%sql select min(DATE) from SPACEXTBL where [Landing _Outcome] = 'Success (ground pad)'
```

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

min(DATE)
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the WHERE clause to filter for boosters which have successfully landed on drone ship and applied the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000.



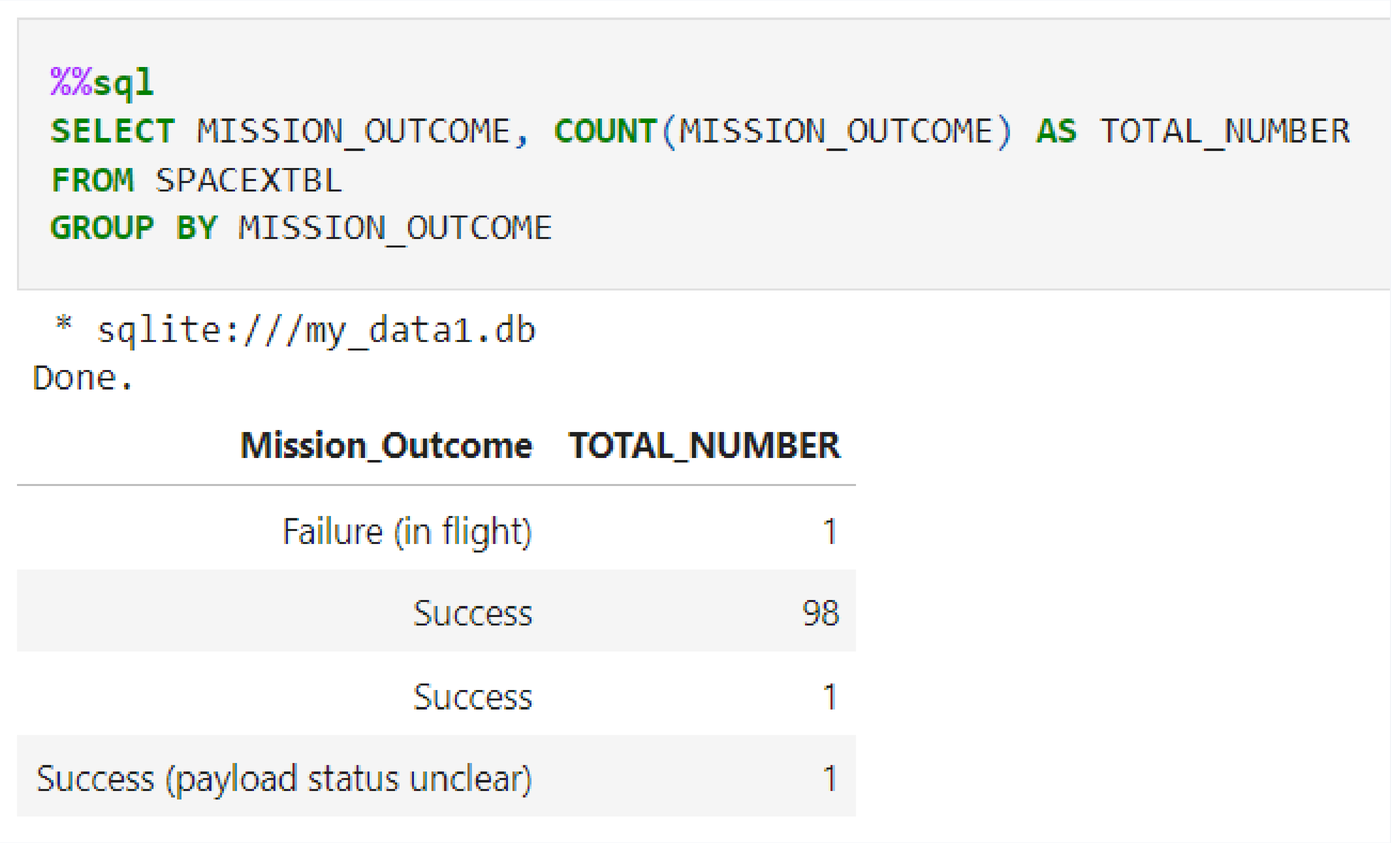
```
%%sql
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE [Landing _Outcome] = 'Success (drone ship)'
    AND PAYLOAD_MASS__KG_ > 4000
    AND PAYLOAD_MASS__KG_ < 6000
```

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

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

Total Number of Successful and Failure Mission Outcomes

- We used GROUP BY clause and deduced that the total number of successful mission outcome is 98+1+1=100 and the total number of failed mission outcome is 1.



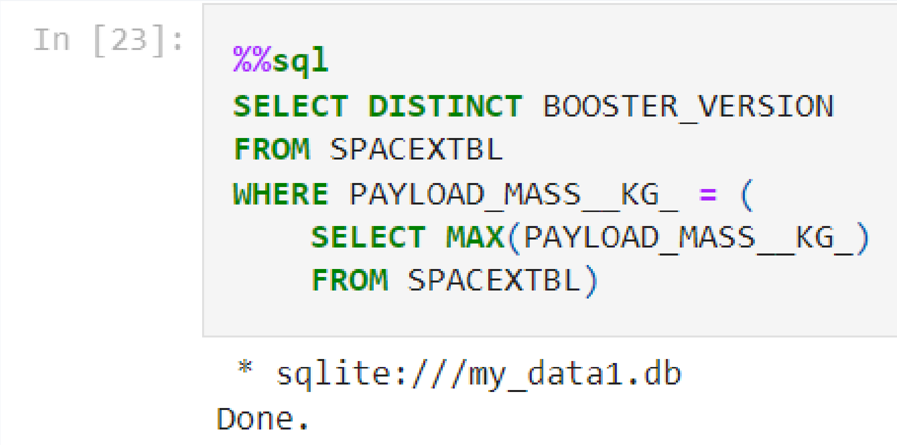
```
%%sql
SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL_NUMBER
FROM SPACEXTBL
GROUP BY MISSION_OUTCOME
```

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

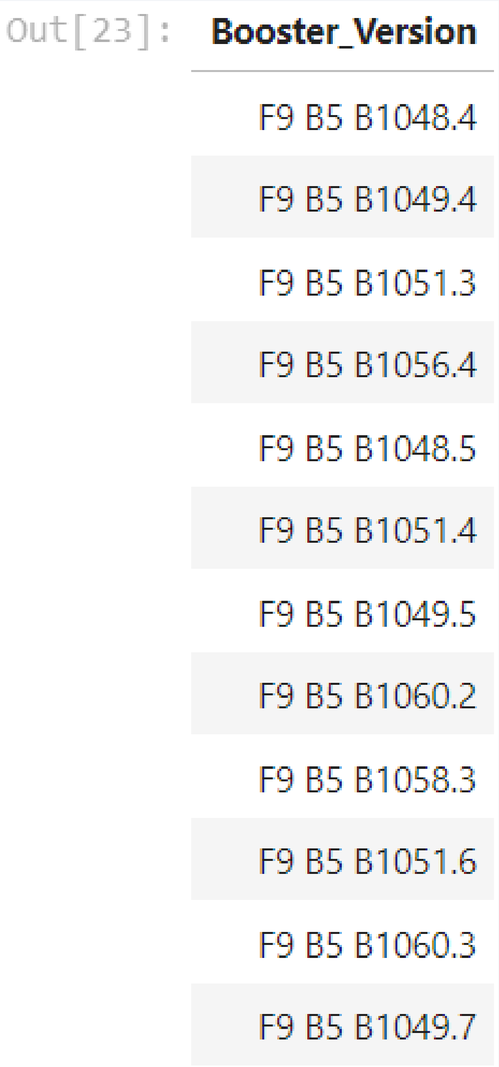
Mission_Outcome	TOTAL_NUMBER
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- We determined the booster versions that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.



```
In [23]:  %%sql
          SELECT DISTINCT BOOSTER_VERSION
          FROM SPACEXTBL
          WHERE PAYLOAD_MASS__KG_ = (
              SELECT MAX(PAYLOAD_MASS__KG_)
              FROM SPACEXTBL)

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

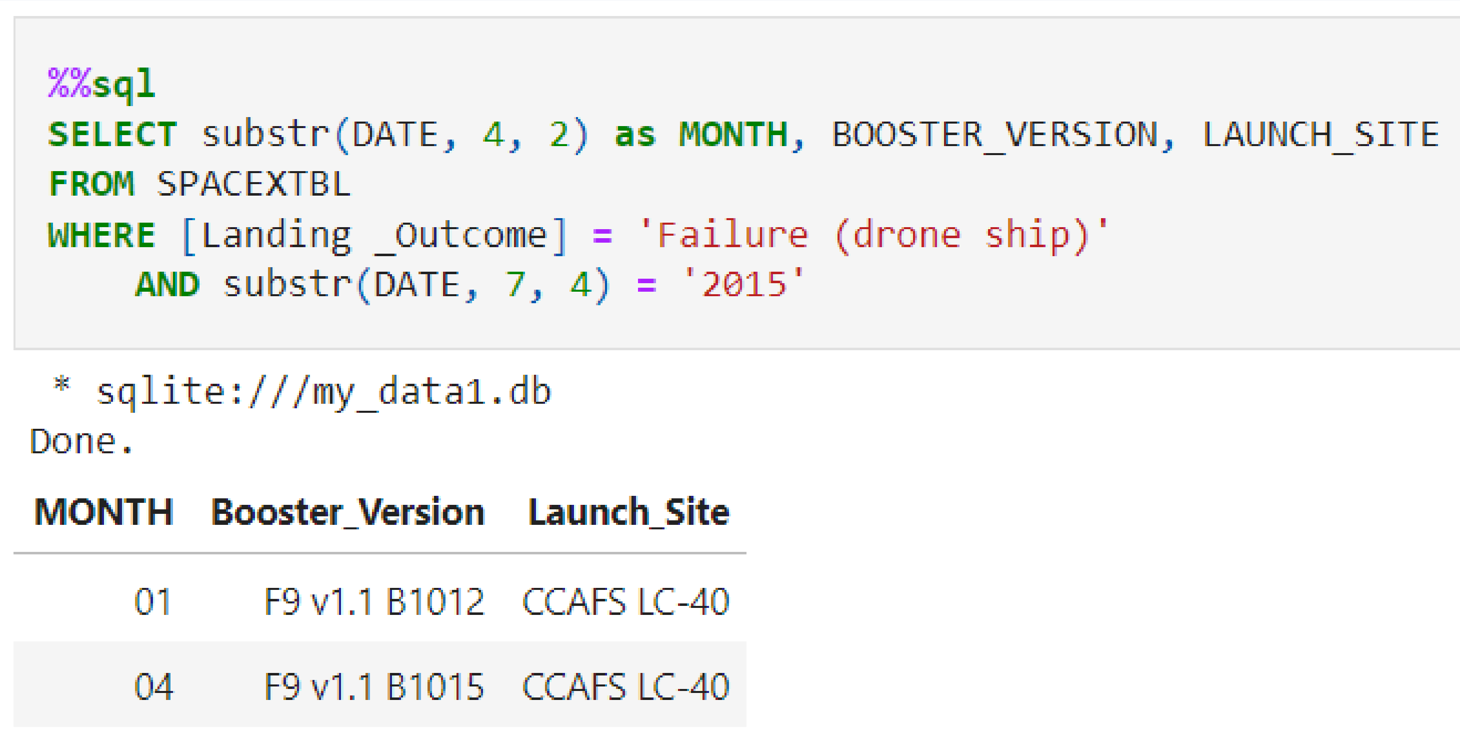


Out[23]:

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

- We used a combinations of the WHERE clause and the AND condition to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015. We also used the SUBSTR function to get the year from the DATE column.



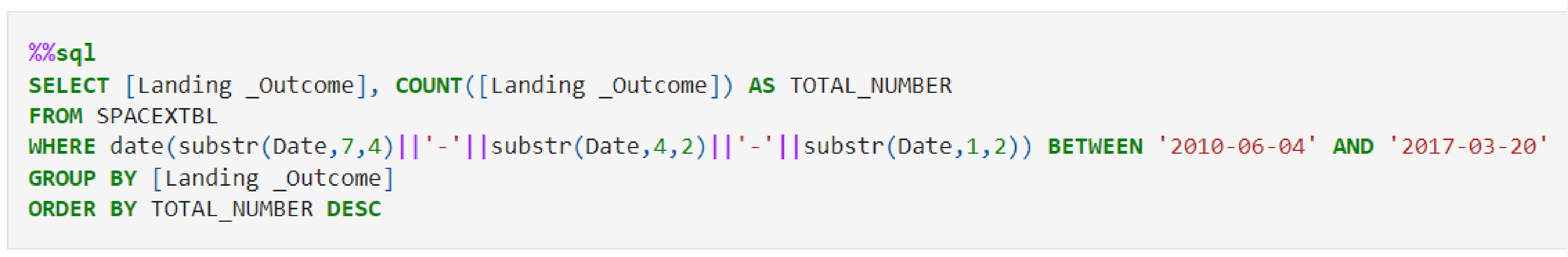
```
%%sql
SELECT substr(DATE, 4, 2) as MONTH, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE [Landing _Outcome] = 'Failure (drone ship)'
    AND substr(DATE, 7, 4) = '2015'
```

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

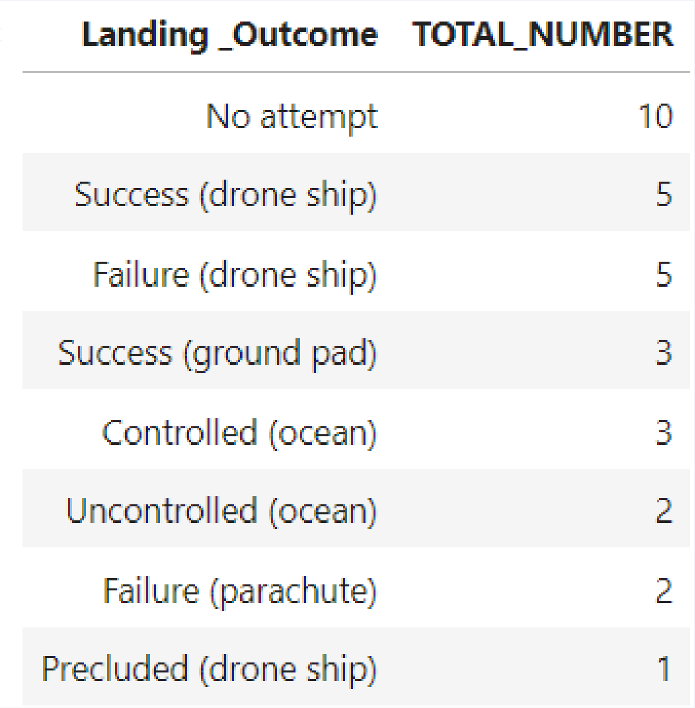
MONTH	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

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

We first used the GROUP BY clause, selected landing outcomes and the COUNT of landing outcomes from the data, and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 and 2017-03-20. We also used the SUBSTR function to get the day, month, and year from the DATE column.



```
%%sql
SELECT [Landing _Outcome], COUNT([Landing _Outcome]) AS TOTAL_NUMBER
FROM SPACEXTBL
WHERE date(substr(Date,7,4)||'-'||substr(Date,4,2)||'-'||substr(Date,1,2)) BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY [Landing _Outcome]
ORDER BY TOTAL_NUMBER DESC
```



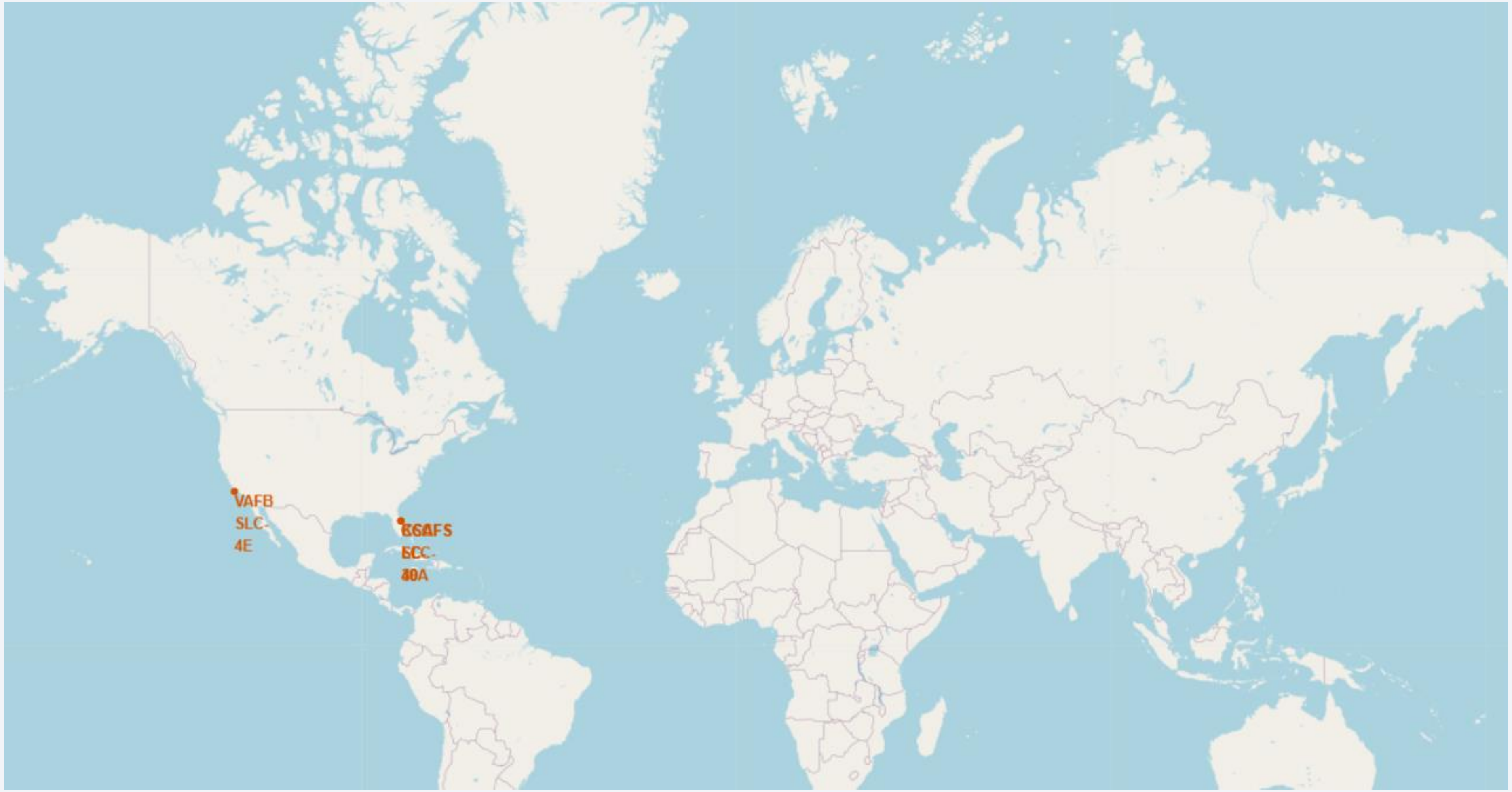
Landing _Outcome	TOTAL_NUMBER
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

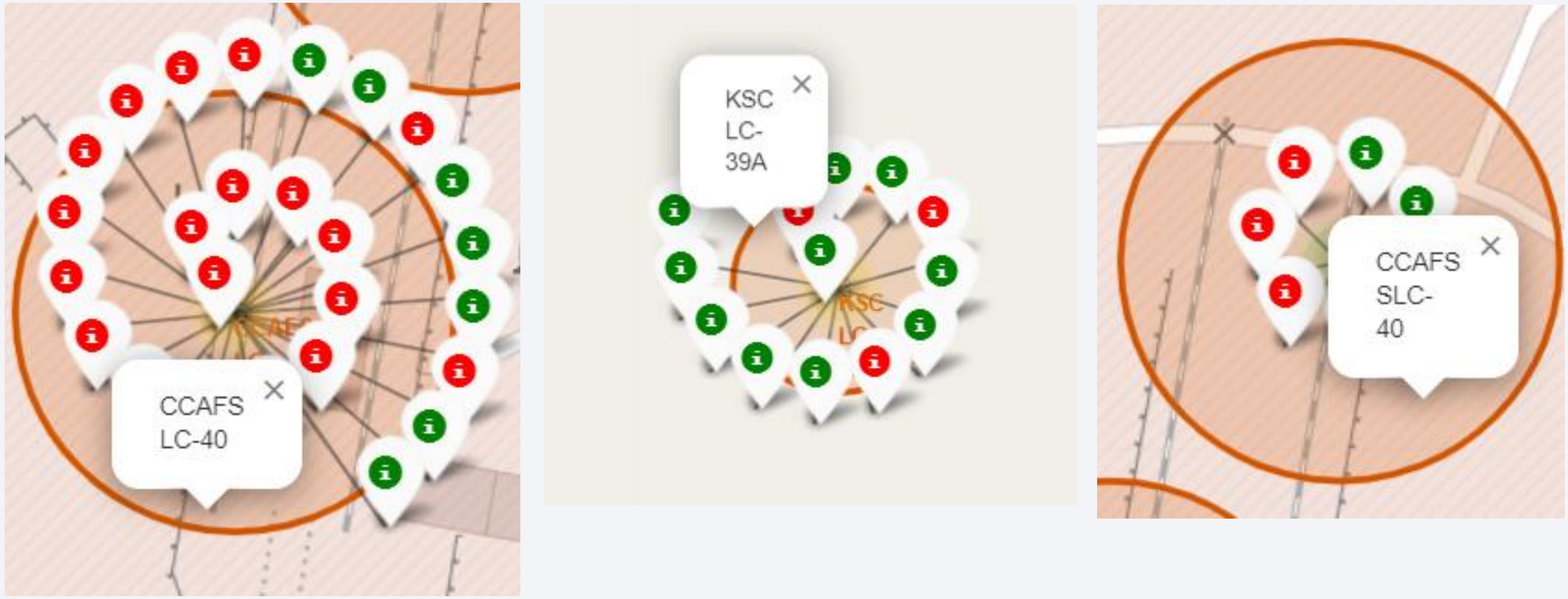
Locations of all Launch Sites

All launch sites are located in the United States.

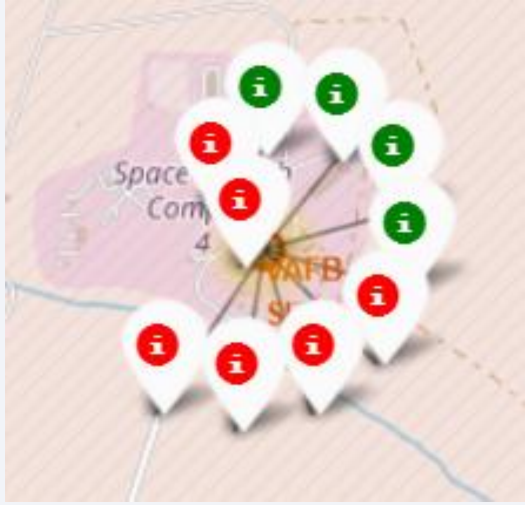


Markers showing launch sites with color labels

Florida launch sites

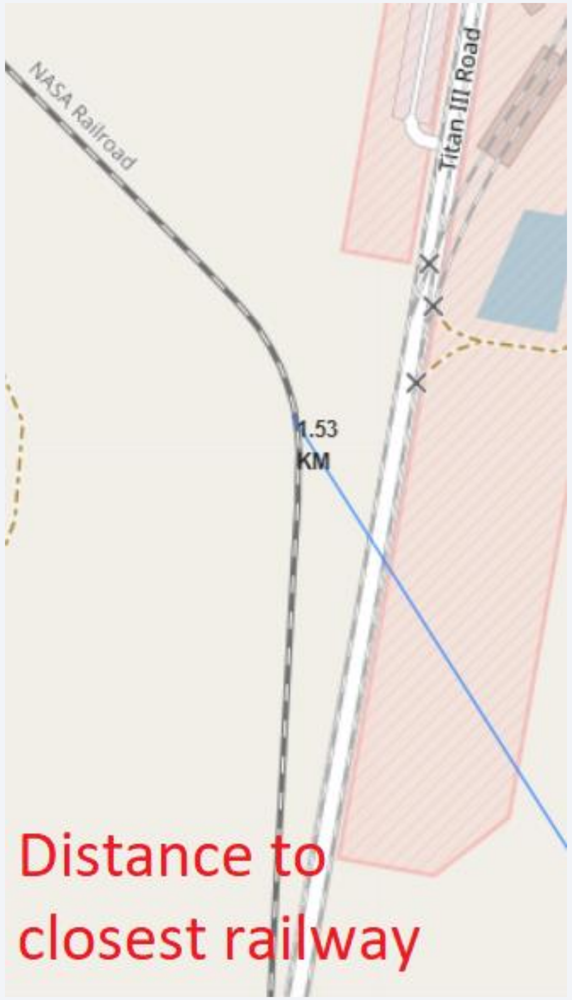


California launch site

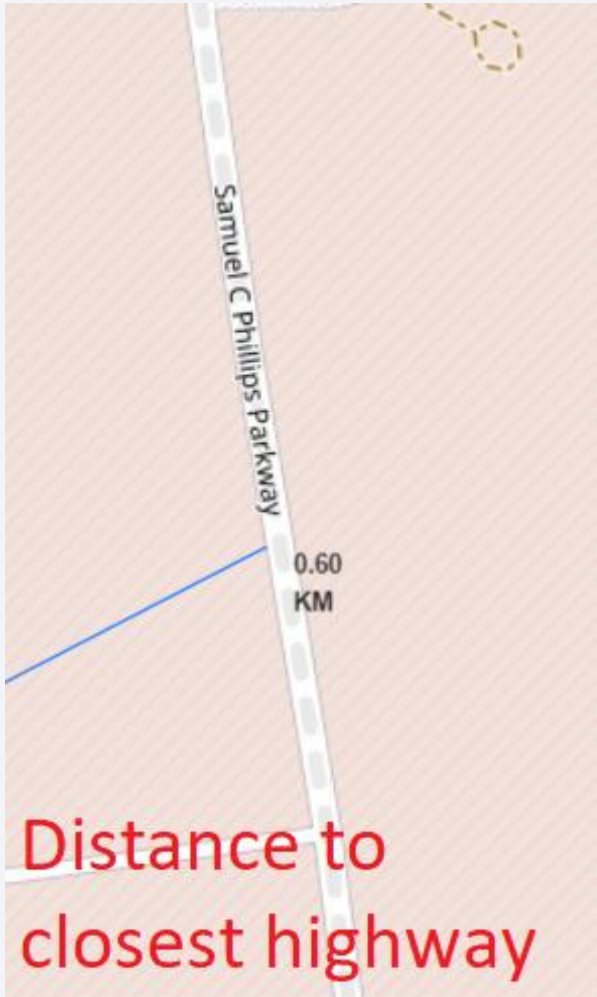


Green markers show successful launches and red markers show fails.

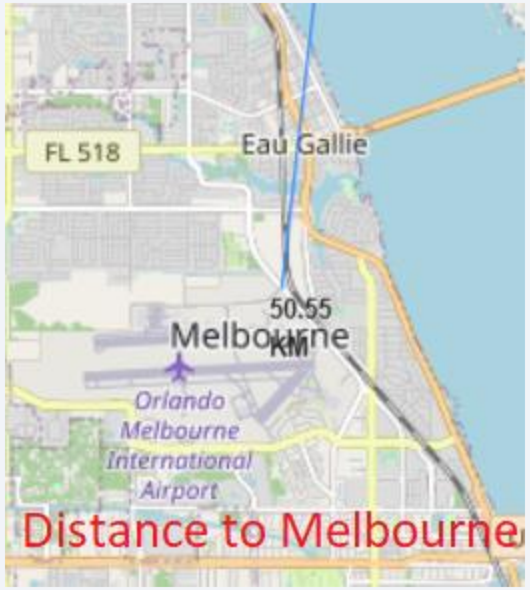
Distance to Landmarks



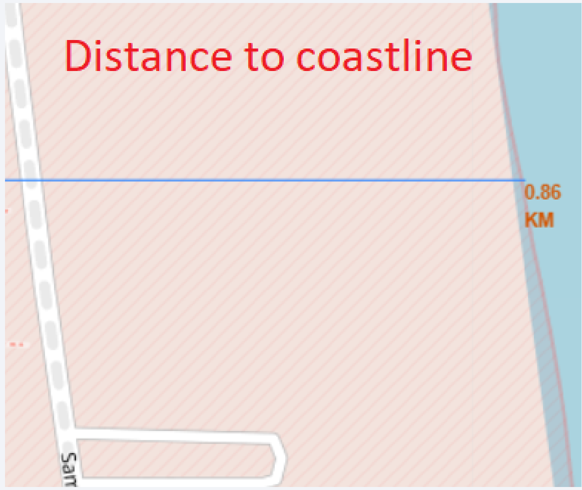
Distance to closest railway



Distance to closest highway

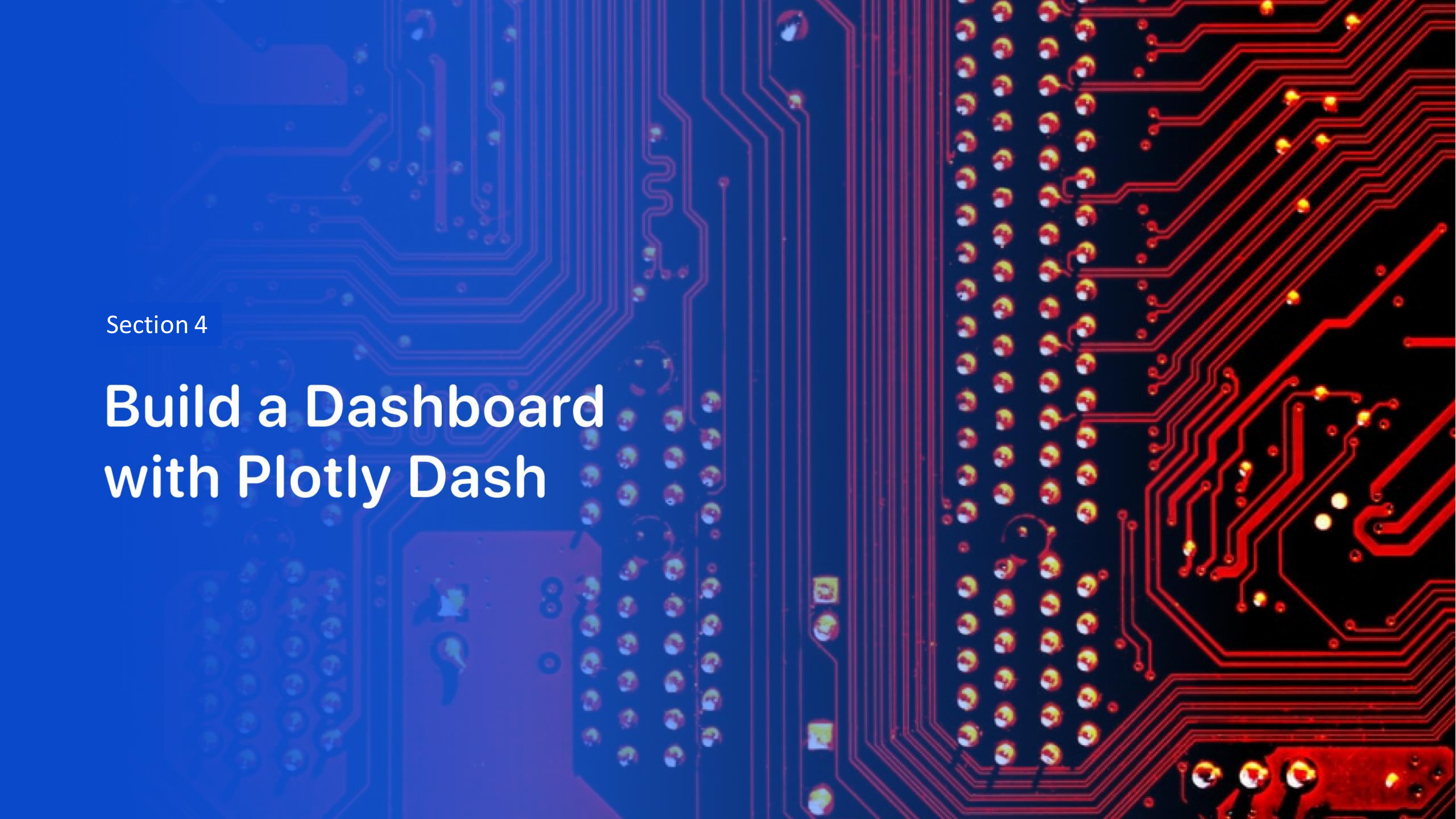


Distance to Melbourne



Distance to coastline

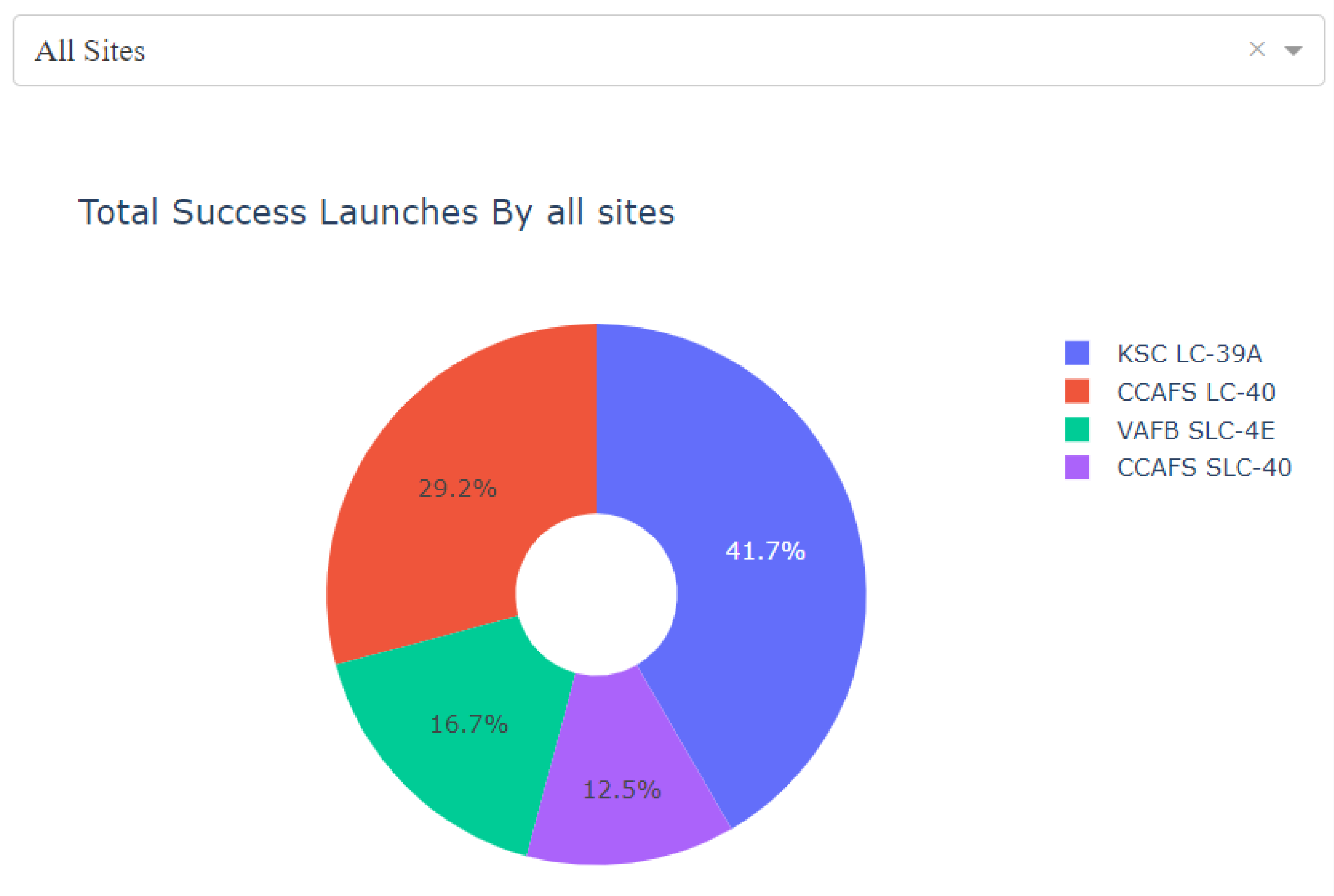
- Are launch sites in close proximity to railways? Yes
- Are launch sites in close proximity to highways? Yes
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 4

Build a Dashboard with Plotly Dash

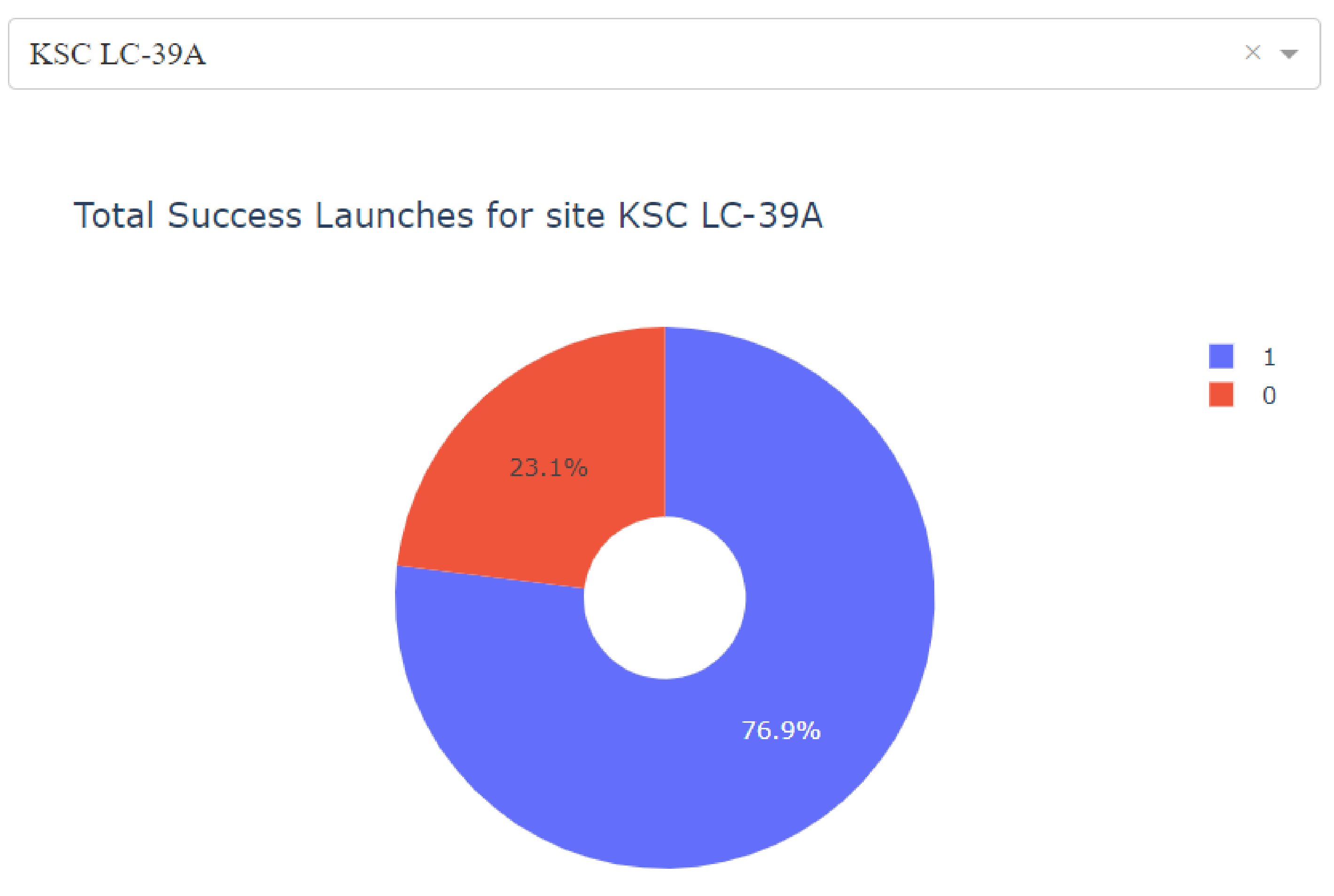
The success percentage by each sites



All Sites

Based on the data, it is evident that KSC LC-39A had the greatest number of successful launches out of all the launch sites.

The highest launch-success ratio: KSC LC-39A

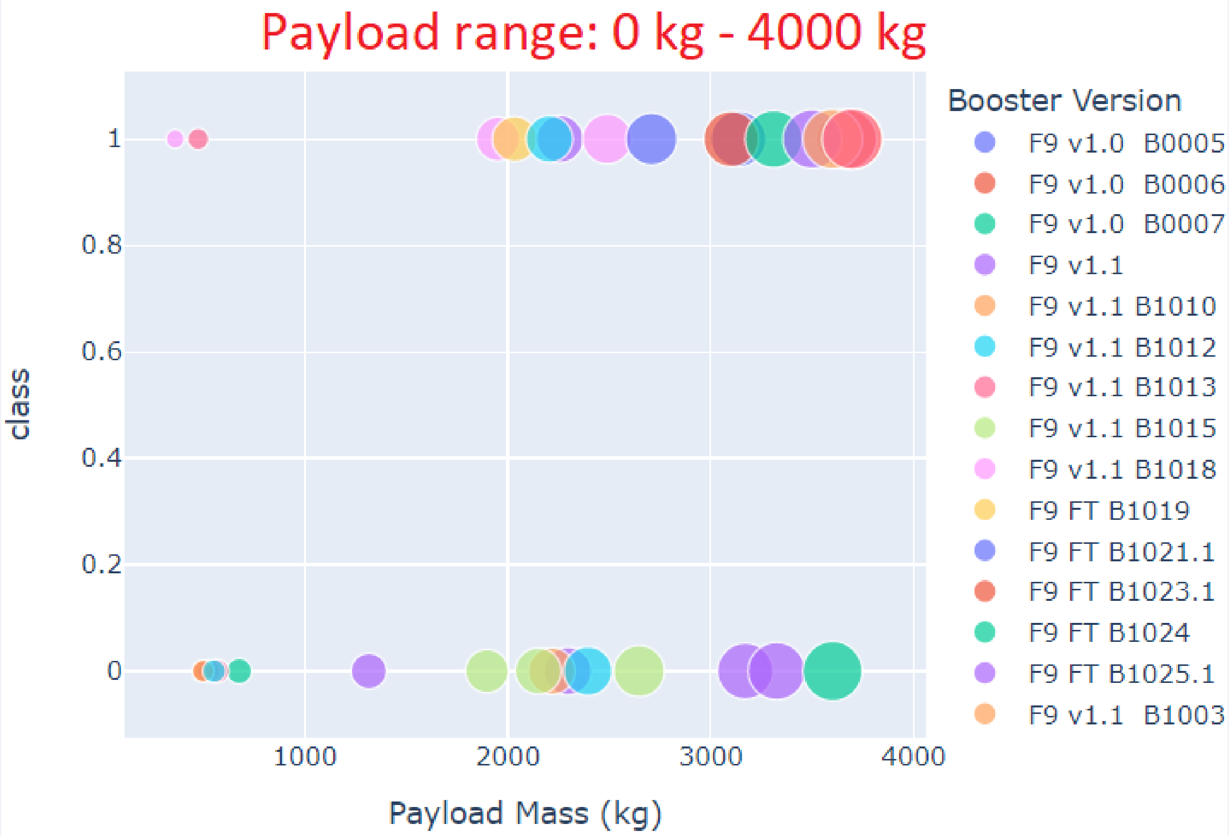


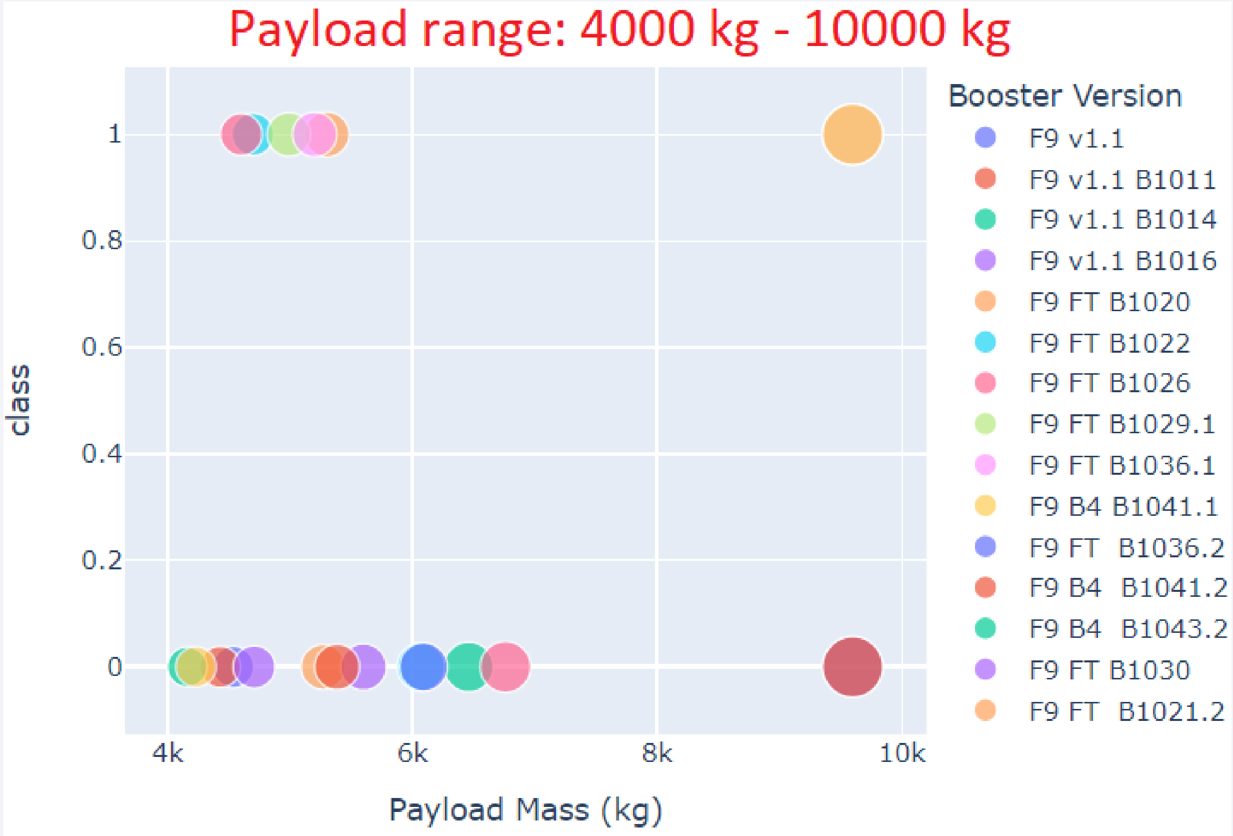
KSC LC-39A

The launch site KSC LC-39A attained a success rate of 76.9% and a failure rate of 23.1%.

Scatter plot of Payload vs Launch Outcome for all sites, with different payload

We can see that the success rates for low-weighted payloads are higher than those for heavy-weighted payloads.



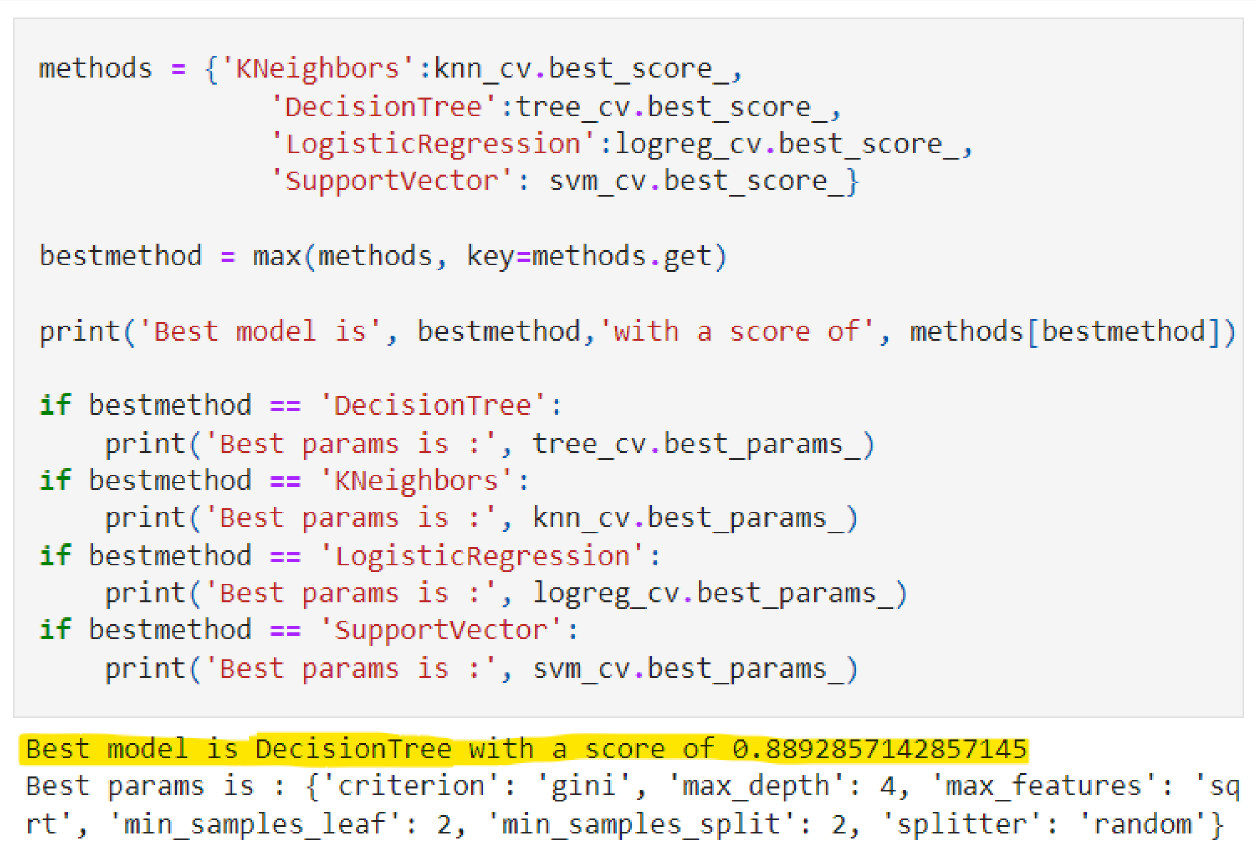


Section 5

Predictive Analysis (Classification)

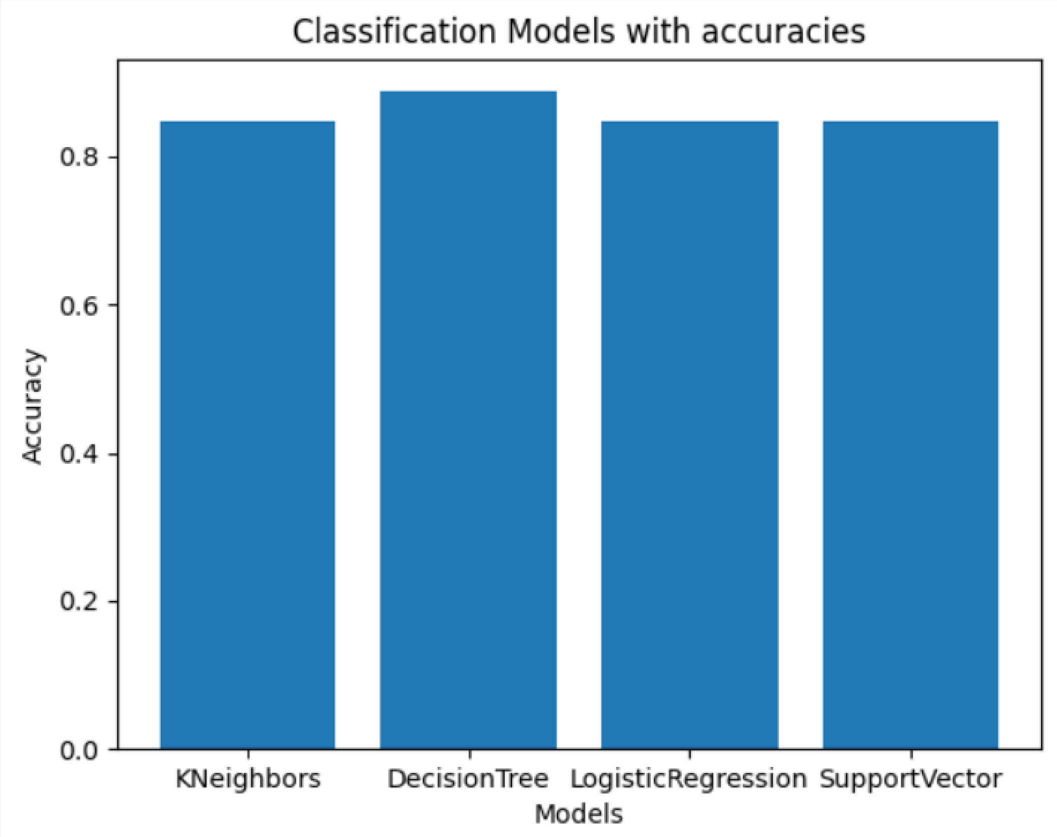
Classification Accuracy

The decision tree classifier is the model that achieves the highest accuracy.



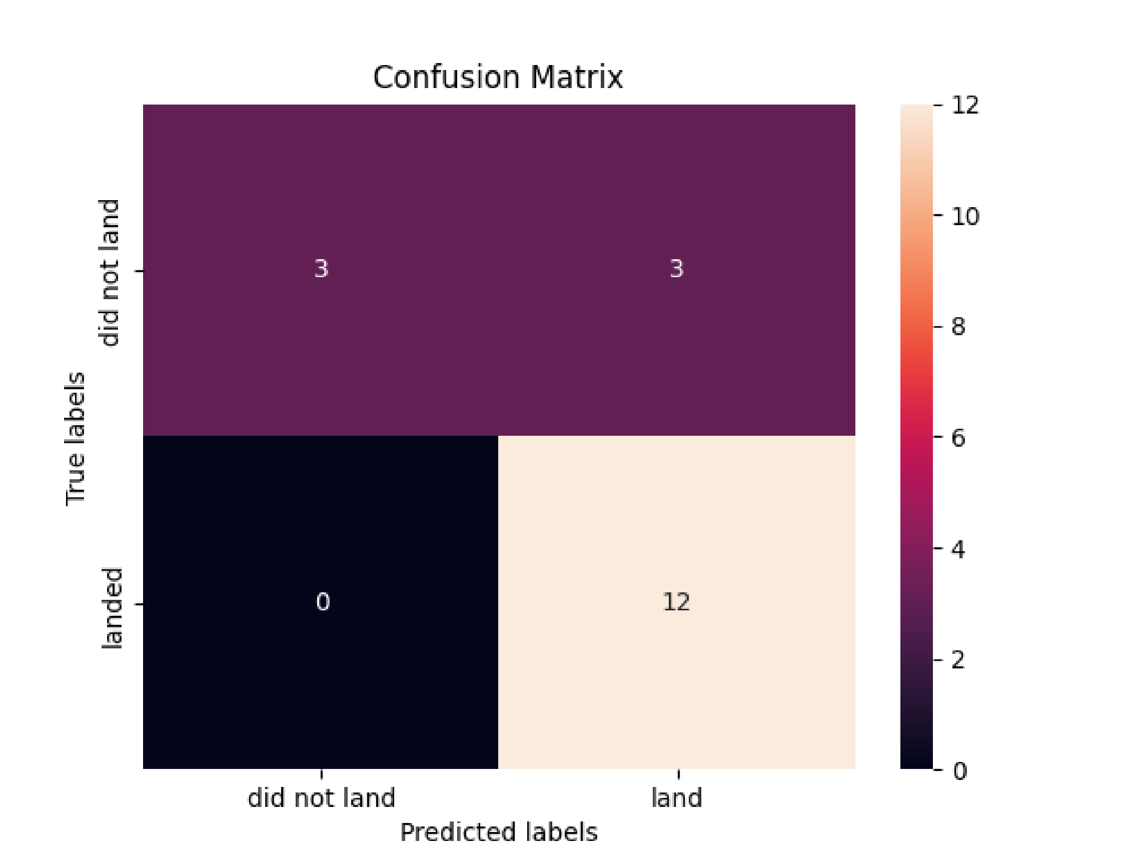
```
methods = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestmethod = max(methods, key=methods.get)

print('Best model is', bestmethod,'with a score of', methods[bestmethod])

if bestmethod == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestmethod == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestmethod == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestmethod == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8892857142857145
Best params is : {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sq
rt', 'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'random'}
```



Confusion Matrix

The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes.



Conclusions

We deduce that:

- The Decision Tree classifier algorithm is the best machine learning method for this dataset.
- The successful rate of low-weighted payloads (defined as 4000 kg and below) outperformed the heavy-weighted ones.
- Starting in 2013, SpaceX's success rate has been continuously increasing, in direct proportion to time until 2020, implying that it will eventually achieve a perfect record in the future.
- KSC LC-39A has the highest success rate among all launch sites.

Thank you!