IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Venkata Sai Naren Jana
04/25/2024

Outline

Executive Summary

- Summary of methodologies
  - ➢ Data Collection with API
  - ➢ Data Collection with Web Scraping
  - ➢ Data Wrangling
  - ➢ EAD with SQL
  - ➢ EAD with Visualization
  - ➢ Interactive Visualization with Folium
  - ➢ ML Models

Executive Summary(Continue)

- Summary of all results
  - ➢ EAD Results
  - ➢ Analysis Screenshots
  - ➢ Prediction Results

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.

- Problems you want to find answers

  - ➢ The Falcon 9 first stage will land successfully or not.
  - ➢ To determine the cost of a launch.
  - ➢ Location of the operating condition suitable for successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - One-Hot Encoding applied to categorical 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

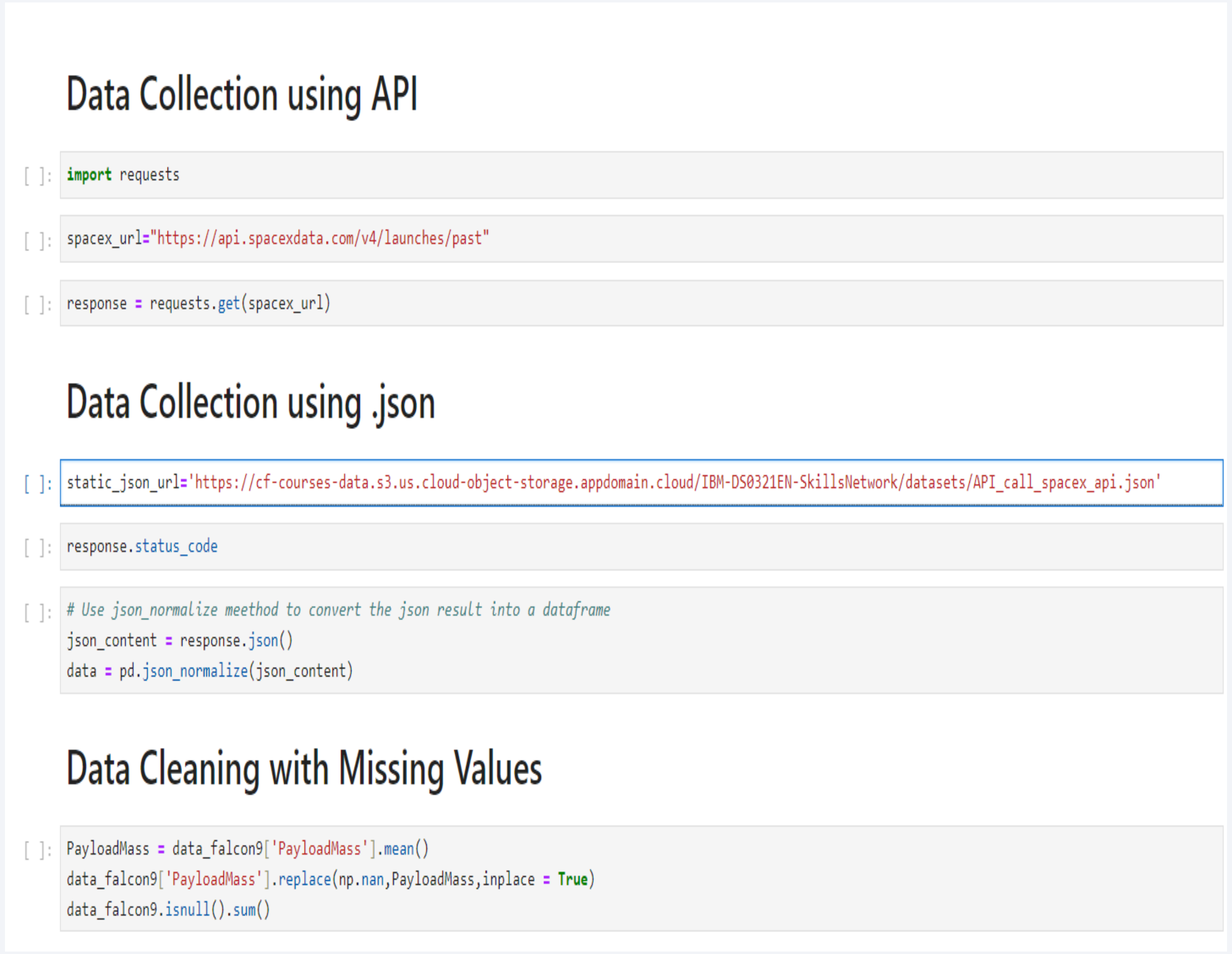
Data Collection

Data was Collected using various methods:

- Data was collected via a get call to the SpaceX API.
- We then decoded the response content as JSON using the .json() function call and converted it to a pandas data frame using .json_normalize().
- We cleaned the data, identified missing values, and filled them in as needed.
- Additionally, we used BeautifulSoup to scrape Wikipedia for Falcon 9 launch records.
- The goal was to extract launch records as an HTML table, process them, then convert them to a pandas data frame for further analysis.

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.
- GitHub URL of SpaceX API calls notebook https://github.com/sainaren41/Data-Science-Capstone-SpaceX



Data Collection using API

```
[ ]: import requests
```

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

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

Data Collection using .json

```
[ ]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

```
[ ]: response.status_code
```

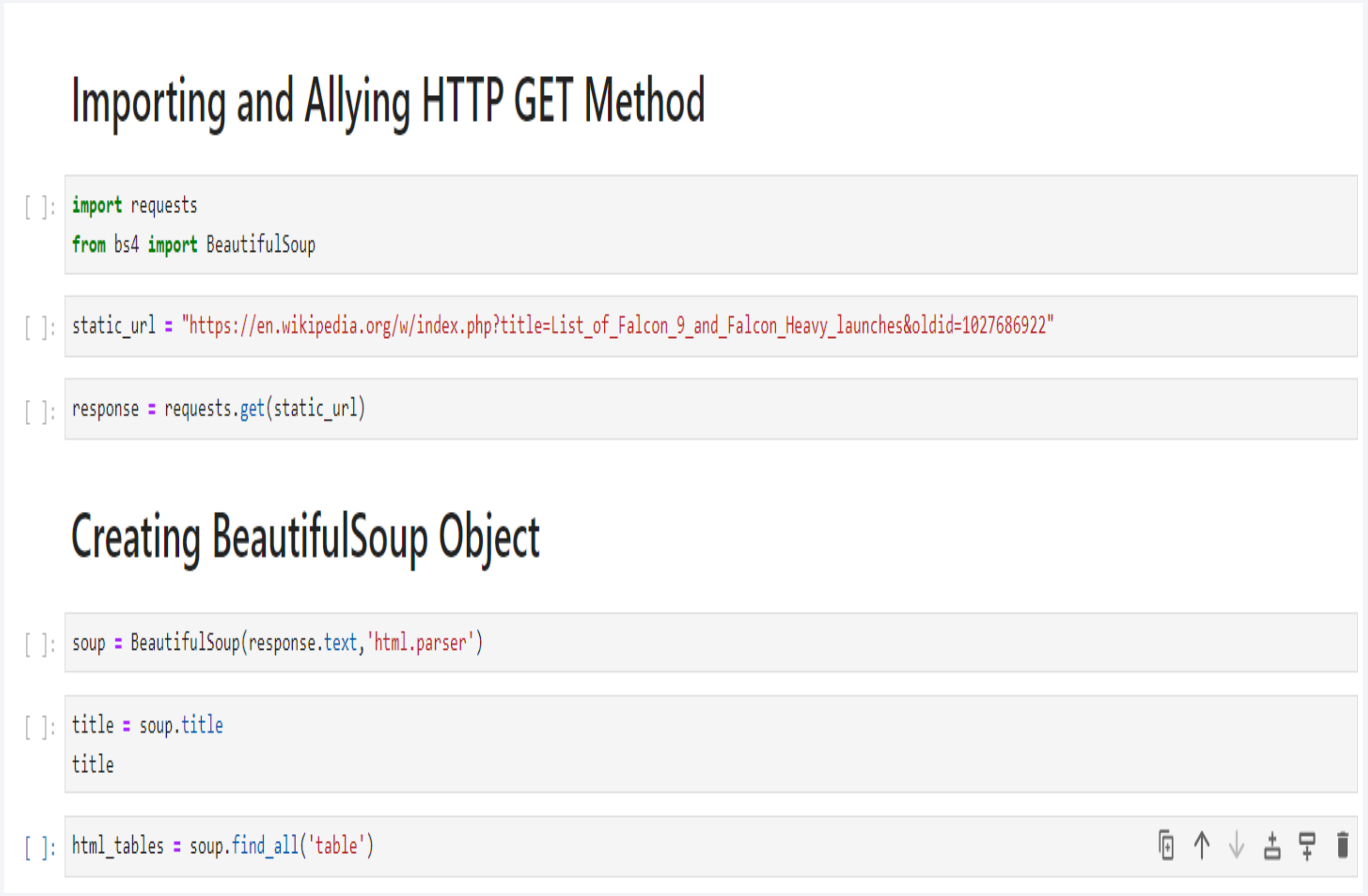
```
[ ]: # Use json_normalize meethod to convert the json result into a dataframe
     json_content = response.json()
     data = pd.json_normalize(json_content)
```

Data Cleaning with Missing Values

```
[ ]: PayloadMass = data_falcon9['PayloadMass'].mean()
     data_falcon9['PayloadMass'].replace(np.nan,PayloadMass,inplace = True)
     data_falcon9.isnull().sum()
```

Data Collection - Scraping

- We applied web scrapping to web scrap Falcon 9 launch records with BeautifulSoup.
- We parsed the table and converted it into a pandas data frame.
- GitHub URL of SpaceX API calls notebook https://github.com/sainaren41/Data-Science-Capstone-SpaceX



Importing and Allying HTTP GET Method

```
[ ]: import requests
     from bs4 import BeautifulSoup
```

```
[ ]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

```
[ ]: response = requests.get(static_url)
```

Creating BeautifulSoup Object

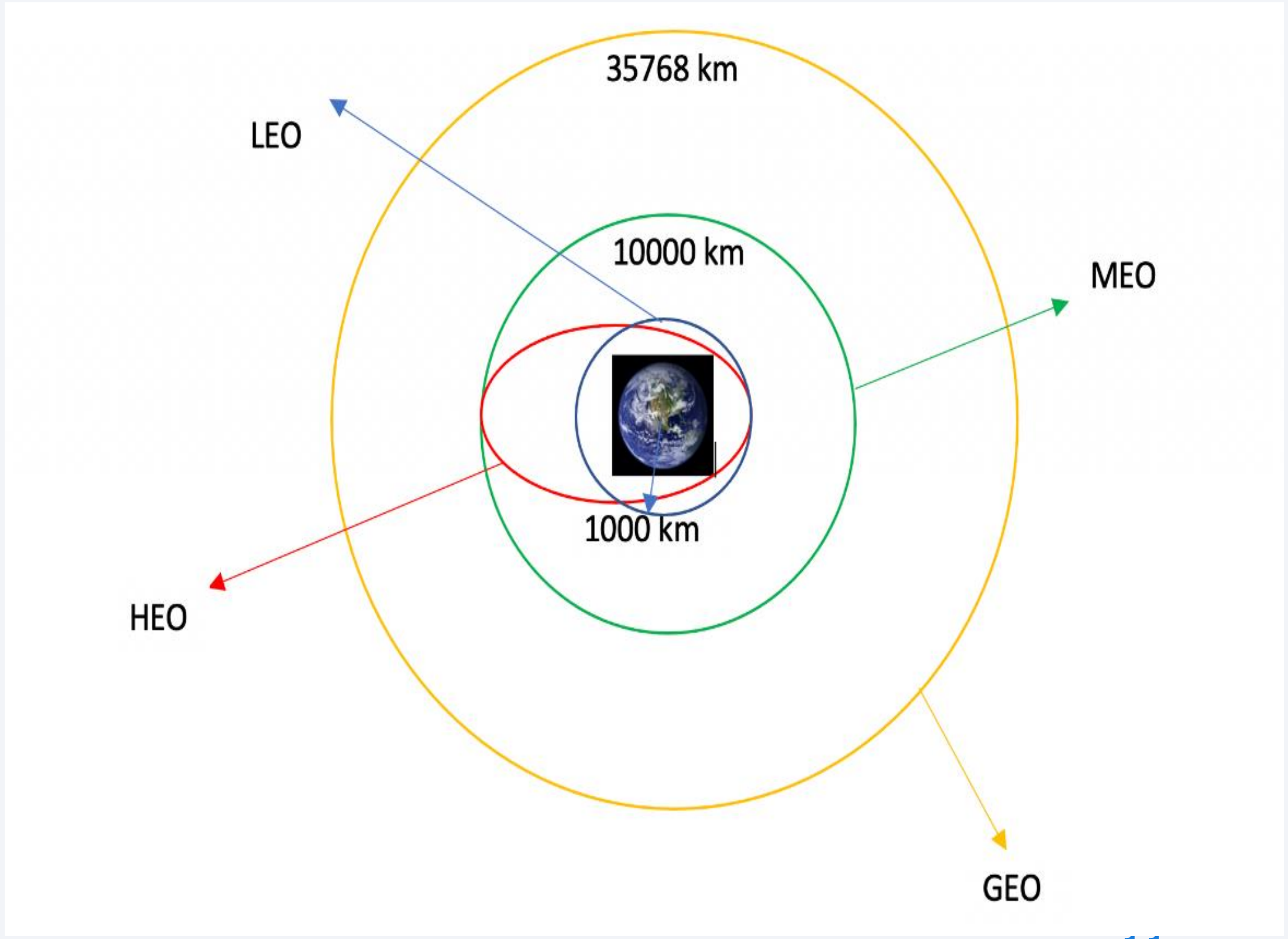
```
[ ]: soup = BeautifulSoup(response.text,'html.parser')
```

```
[ ]: title = soup.title
     title
```

```
[ ]: html_tables = soup.find_all('table')
```

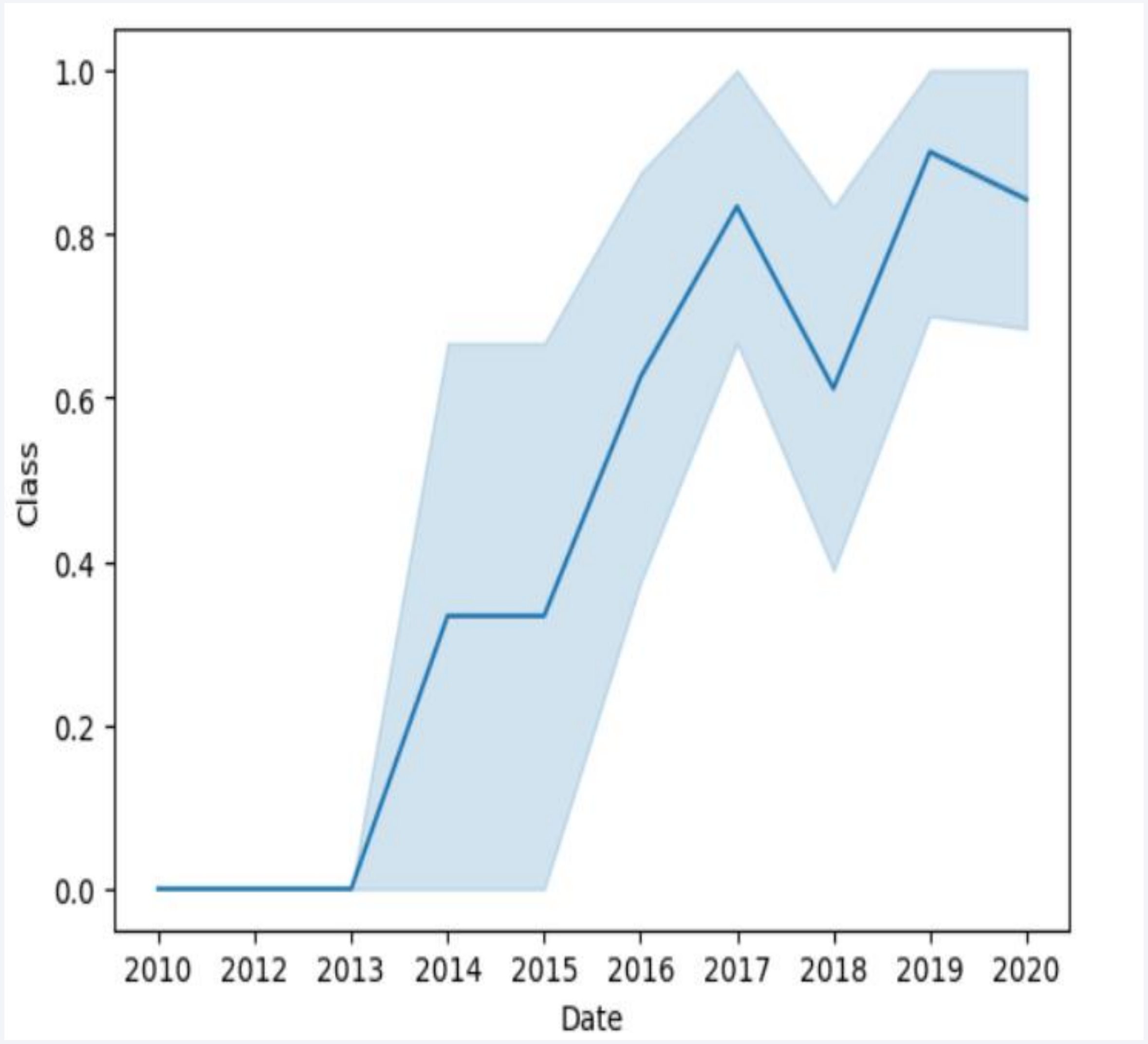
Data Wrangling

- We performed exploratory data analysis to find the training labels.
- We computed the number of launches at each site, as well as the number and frequency of each orbit.
- We generated a landing outcome label from the outcome column and exported the results to csv.
- GitHub URL of SpaceX API calls notebook https://github.com/sainaren41/Data-Science-Capstone-SpaceX



EDA with Data Visualization

- We analyzed the data by displaying the correlation between flight number and launch site, payload and launch site, success rate of each orbit type, flight number and orbit type, and annual launch success trend.
- GitHub URL of SpaceX API calls notebook https://github.com/sainaren41/Data-Science-Capstone-SpaceX



EDA with SQL

- We put the SpaceX dataset into a PostgreSQL database without leaving the Jupyter notebook. We then used EDA and SQL to gain insights from the data. We created queries to answer questions such as:
  - ➢ The names of each distinct launch point in the space mission.
  - ➢ The total payload mass of boosters launched by NASA (CRS)
  - ➢ The average payload mass transported by booster type F9 v1.1
  - ➢ Mission success and failure rates - Drone ship landing failures, including booster versions and launch sites.
- GitHub URL of SpaceX API calls notebook https://github.com/sainaren41/Data-Science-Capstone-SpaceX

Build an Interactive Map with Folium

- We marked launch sites with markers, circles, and lines to indicate success or failure on the folium map.
- We classified the feature launch outcomes (failure or success) as class 0 and 1.For example, 0 indicates failure while 1 indicates success.
- Color-labeled marker clusters helped us identify launch areas with high success rates.
- We estimated the distances between a launch site and its surroundings. We answered various questions, such as:

  Are launch sites near railways, highways, and coastlines?

  Do launch sites retain a particular distance from cities?

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash.
- We plotted pie charts showing the total launches by a certain sites.
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- GitHub URL of SpaceX API calls notebook https://github.com/sainaren41/Data-Science-Capstone-SpaceX

Predictive Analysis (Classification)

- We loaded data using numpy and pandas, modified it, then divided it into training and testing sets.
  - ➢ We created multiple machine learning models and tuned them.
- GridSearchCV is used to determine hyperparameters.
  - Our model was improved through feature engineering and algorithm optimization, with accuracy as the key metric.
- We discovered the best-performing categorization model.
- GitHub URL of SpaceX API calls notebook https://github.com/sainaren41/Data-Science-Capstone-SpaceX

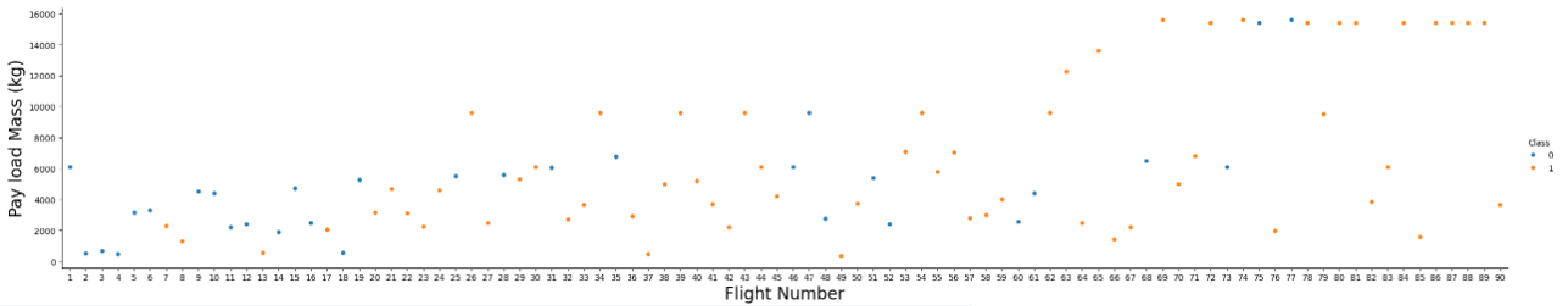
Results

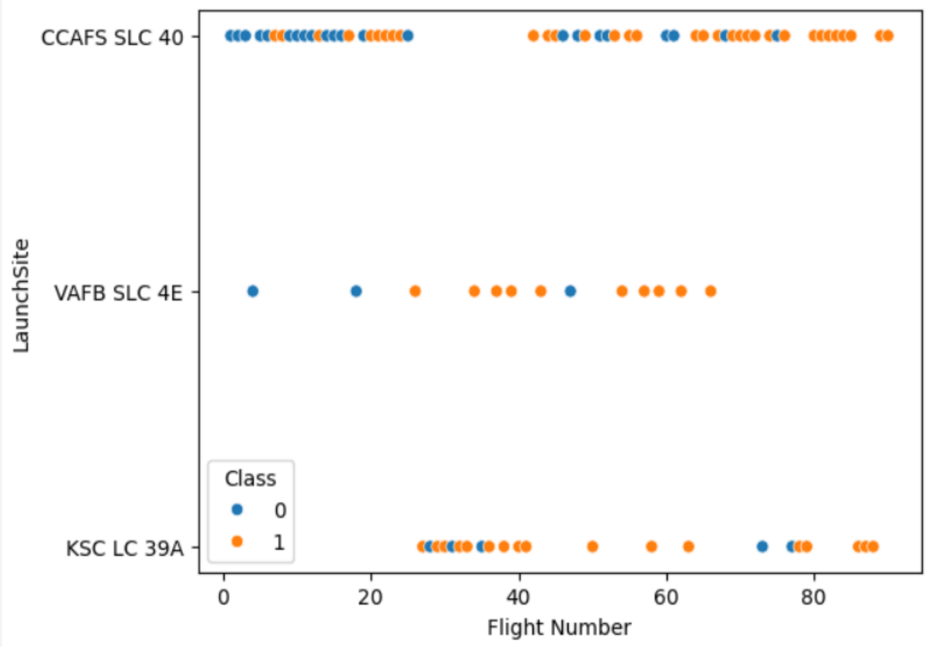
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

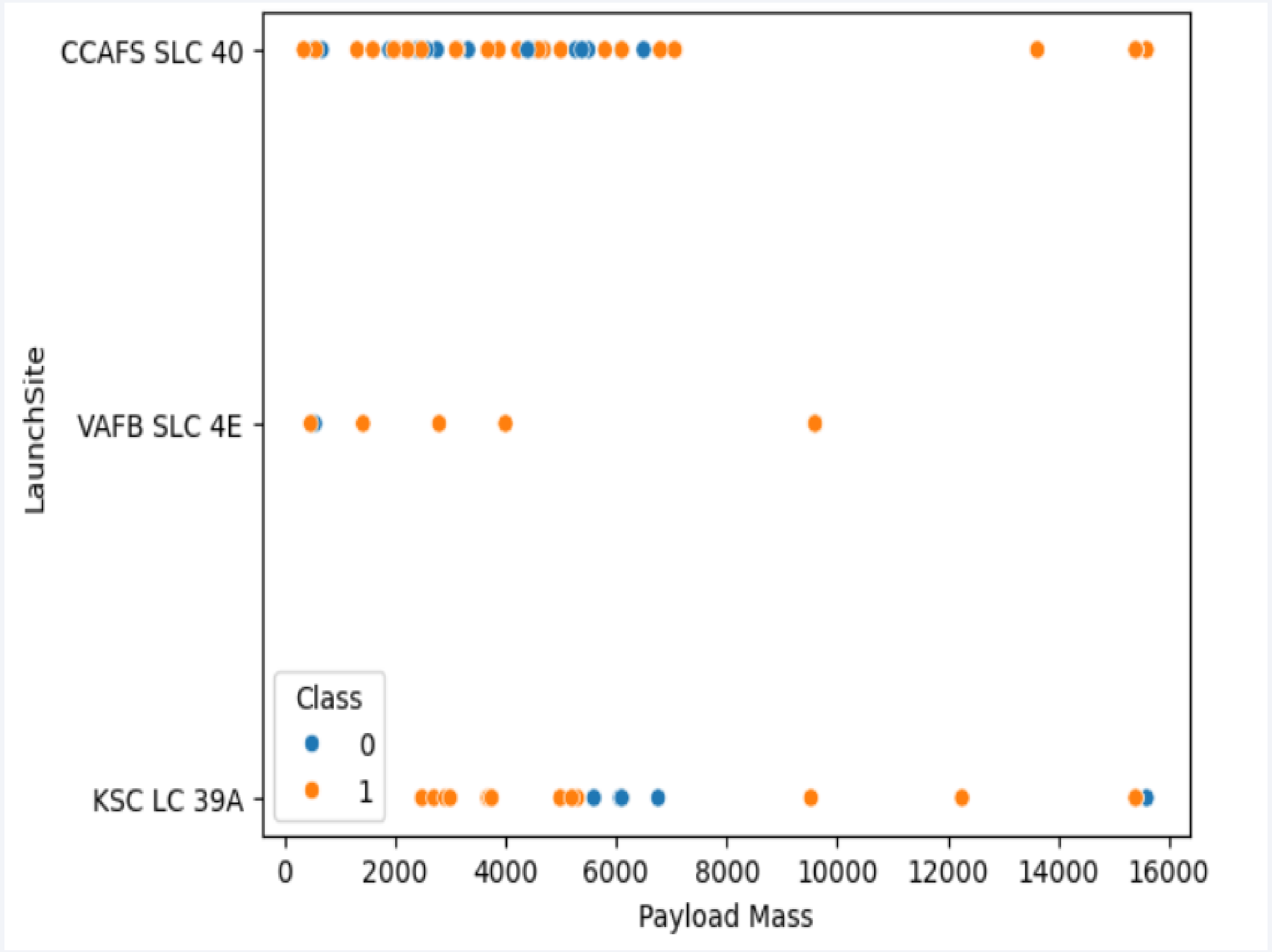




- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.

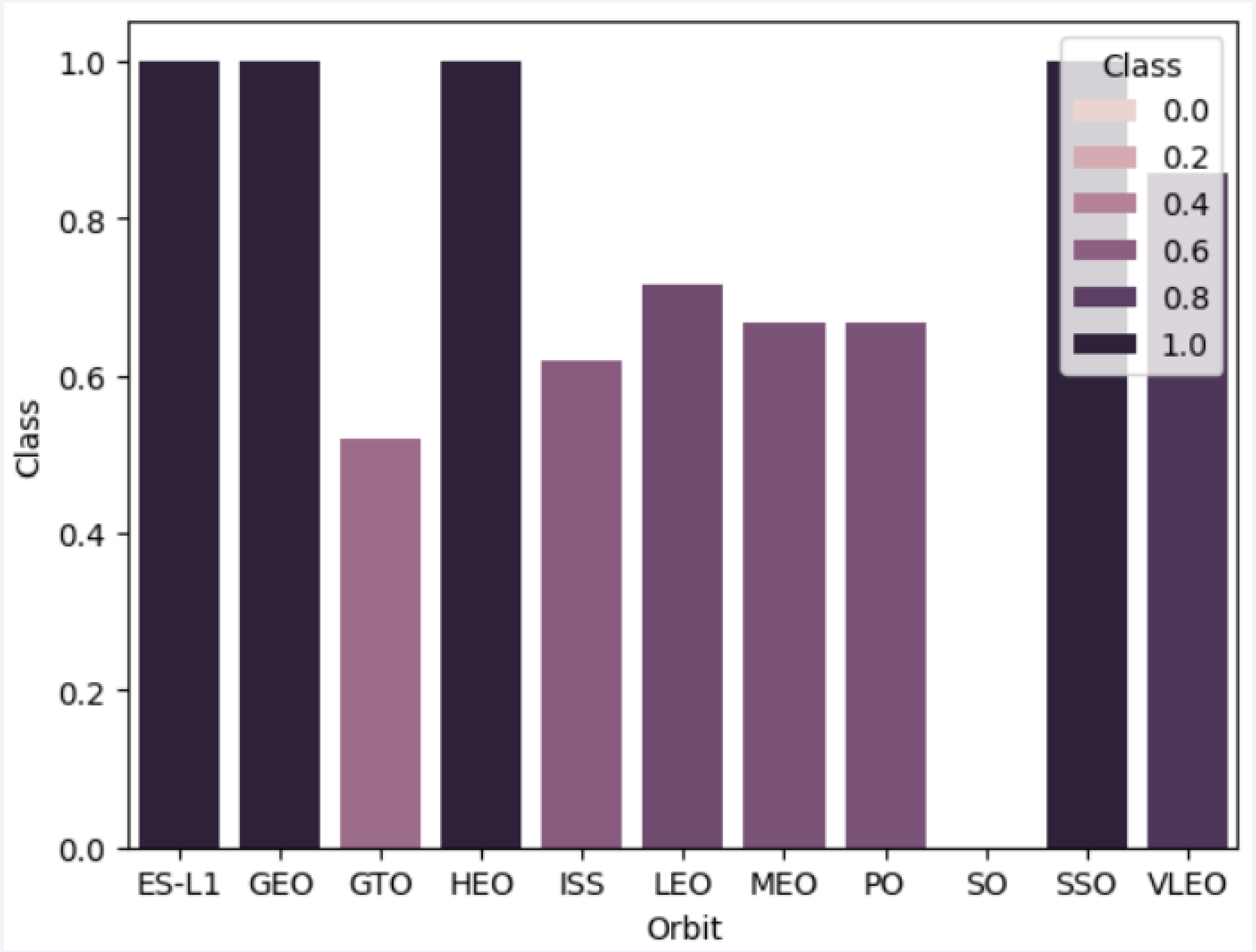
Payload vs. Launch Site

- The greater the payload launch site the greater the success rate.



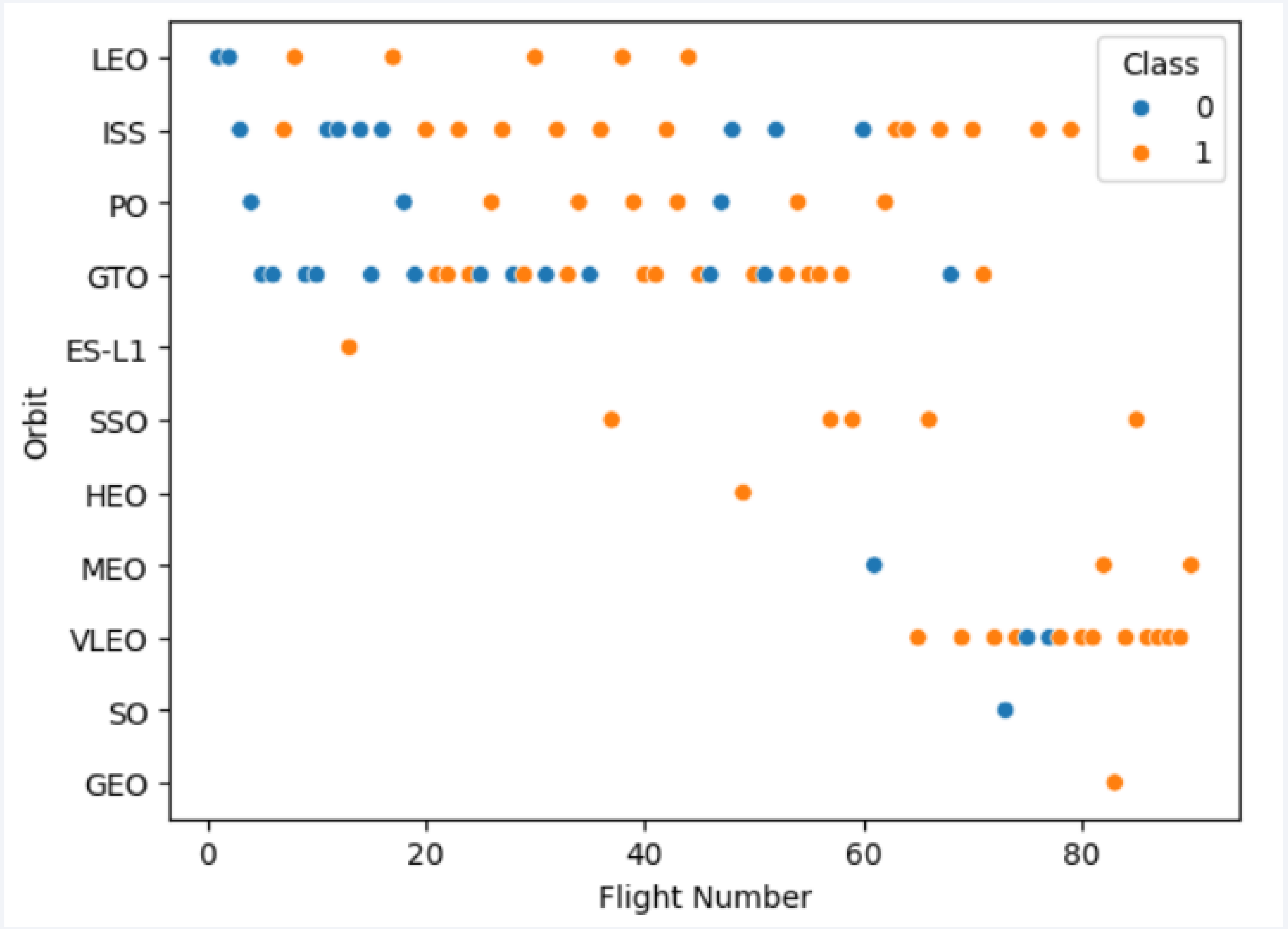
Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO had the most success rate.



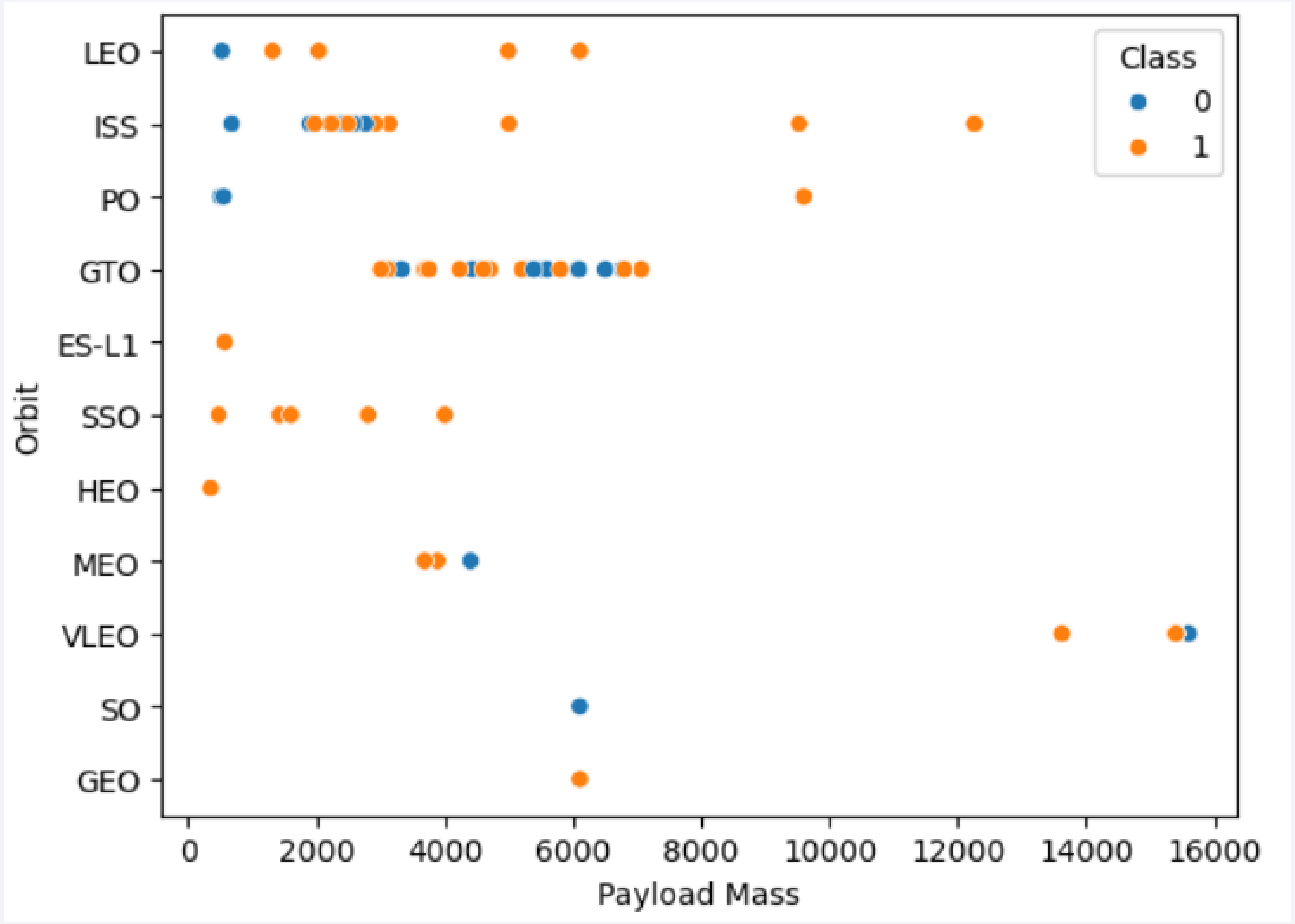
Flight Number vs. Orbit Type

- We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



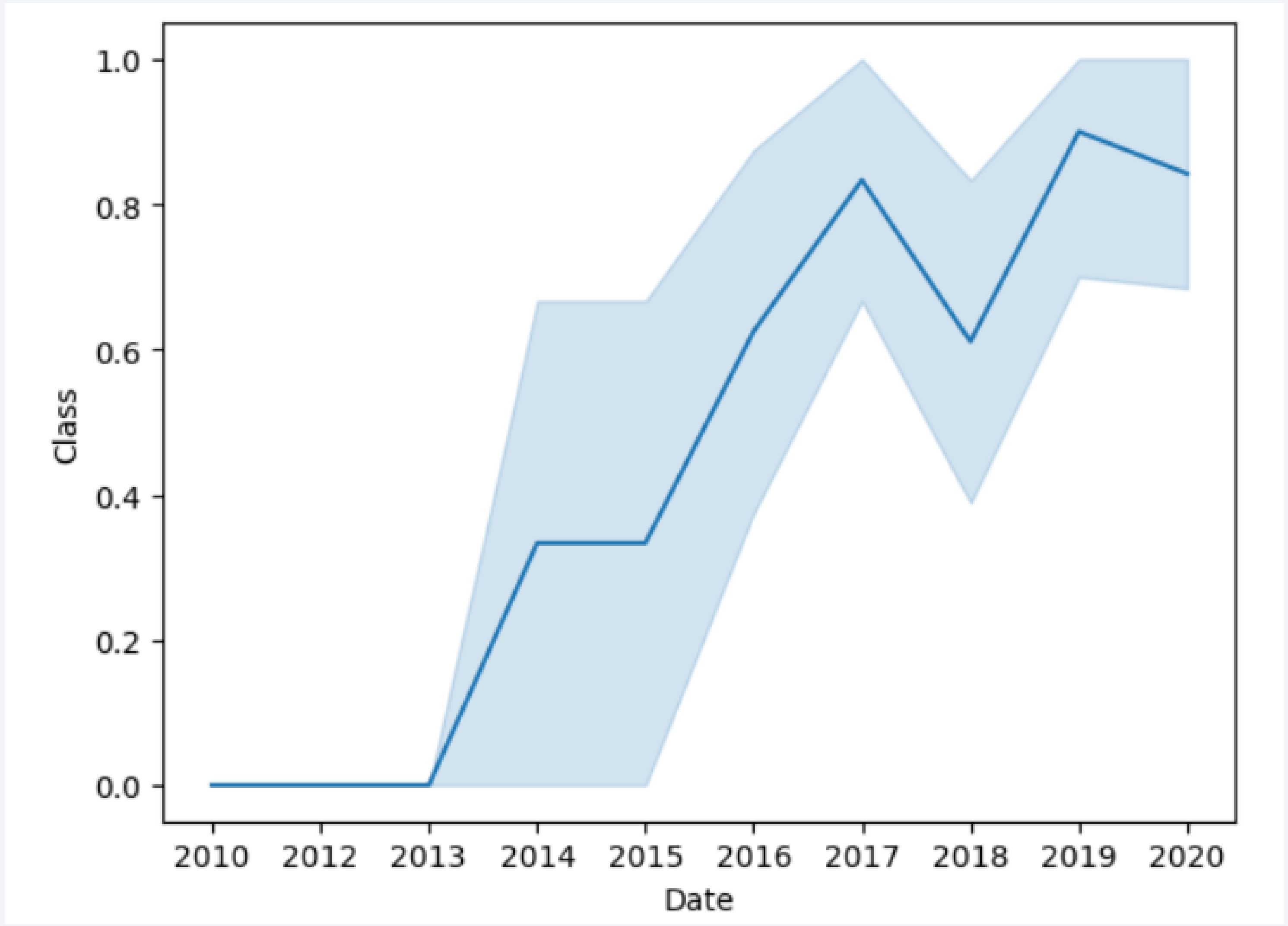
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



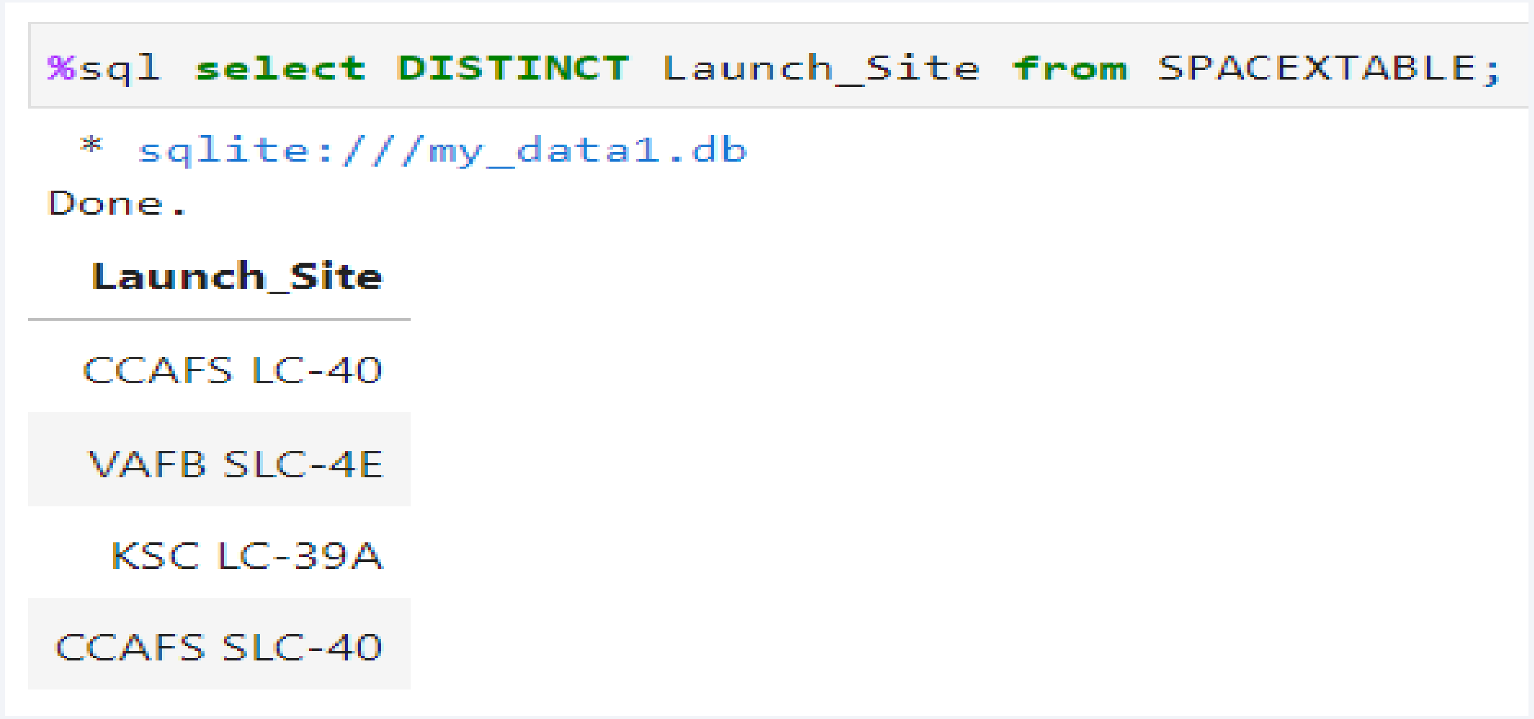
Launch Success Yearly Trend

- We can observe that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

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

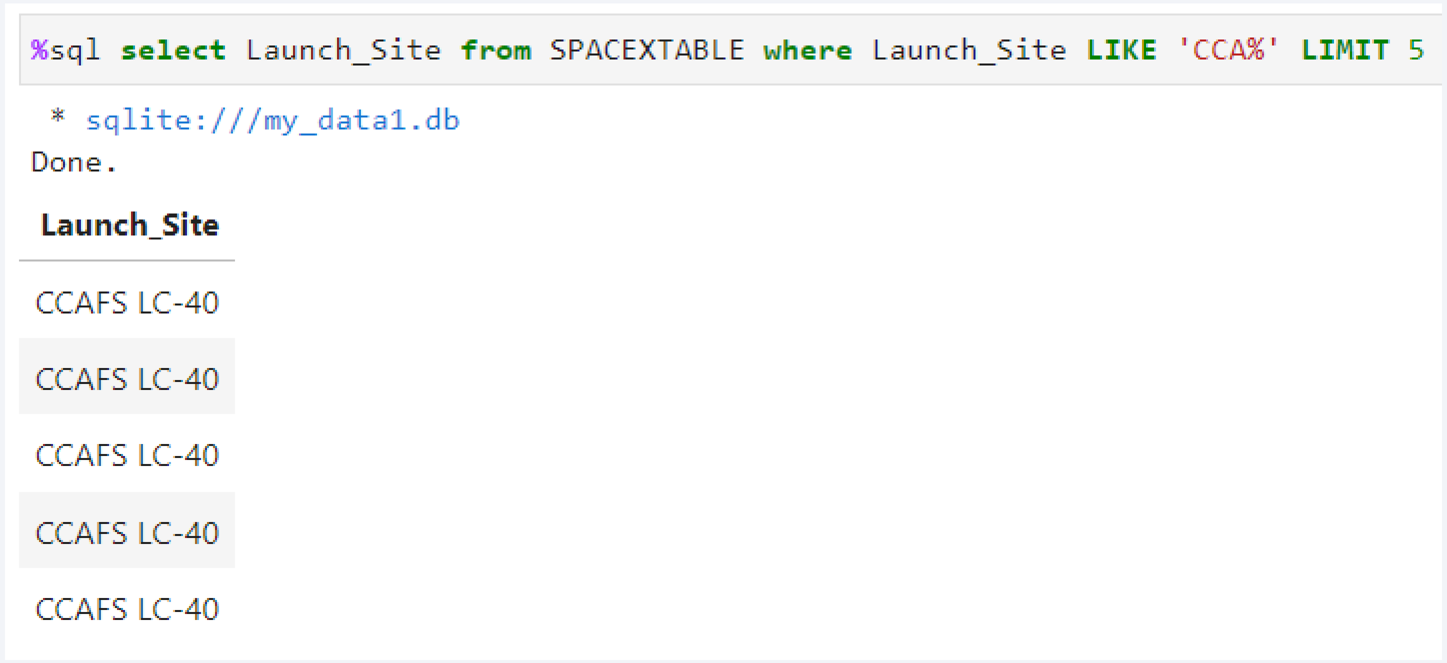


```
%sql select DISTINCT Launch_Site from SPACEXTABLE;
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

- We used the query above to display 5 records where launch sites begin with `CCA`.



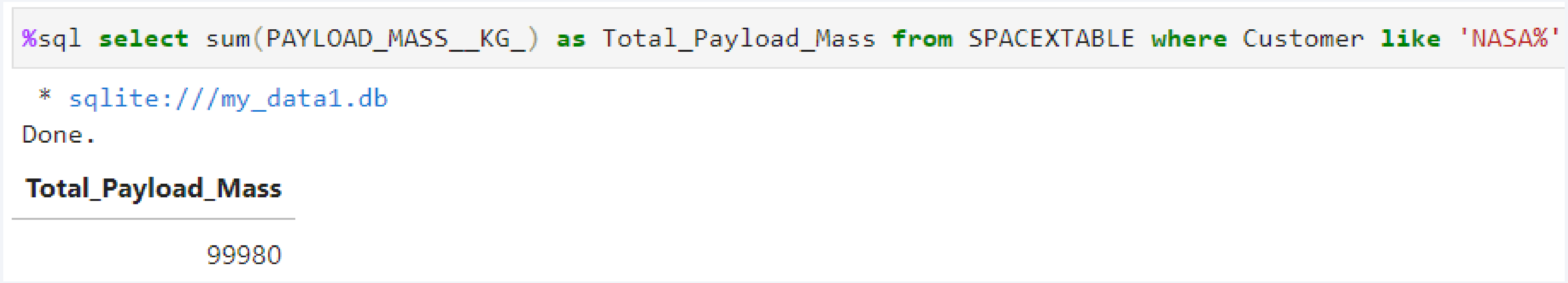
```
%sql select Launch_Site from SPACEXTABLE where Launch_Site LIKE 'CCA%' LIMIT 5
```

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

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

Total Payload Mass

- We calculated the total payload carried by boosters from NASA as 99980.



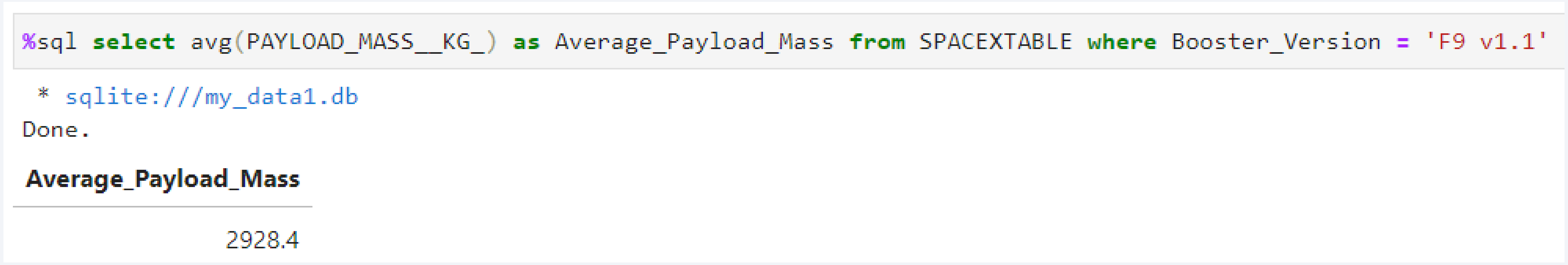
```
%sql select sum(PAYLOAD_MASS__KG_) as Total_Payload_Mass from SPACEXTABLE where Customer like 'NASA%'
```

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

Total_Payload_Mass
99980

Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4.



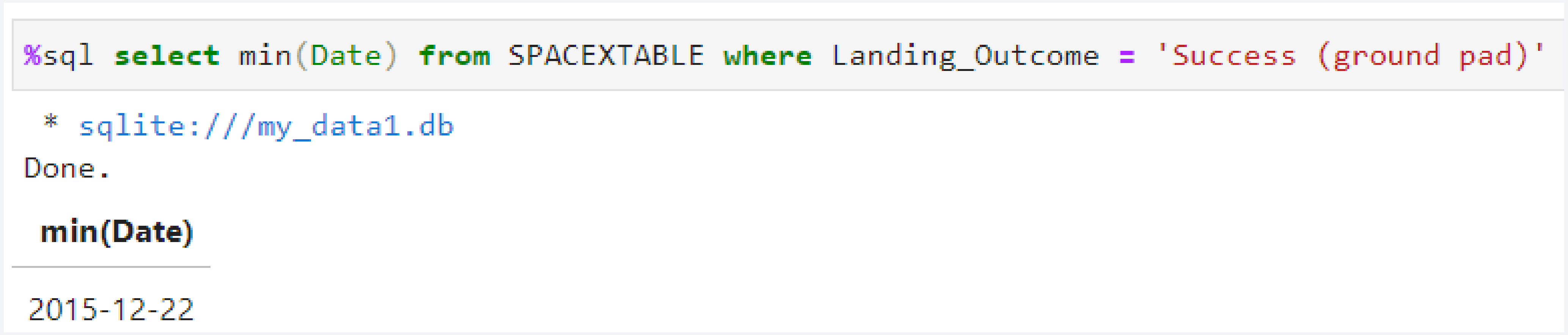
```
%sql select avg(PAYLOAD_MASS__KG_) as Average_Payload_Mass from SPACEXTABLE where Booster_Version = 'F9 v1.1'
```

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

Average_Payload_Mass
2928.4

First Successful Ground Landing Date

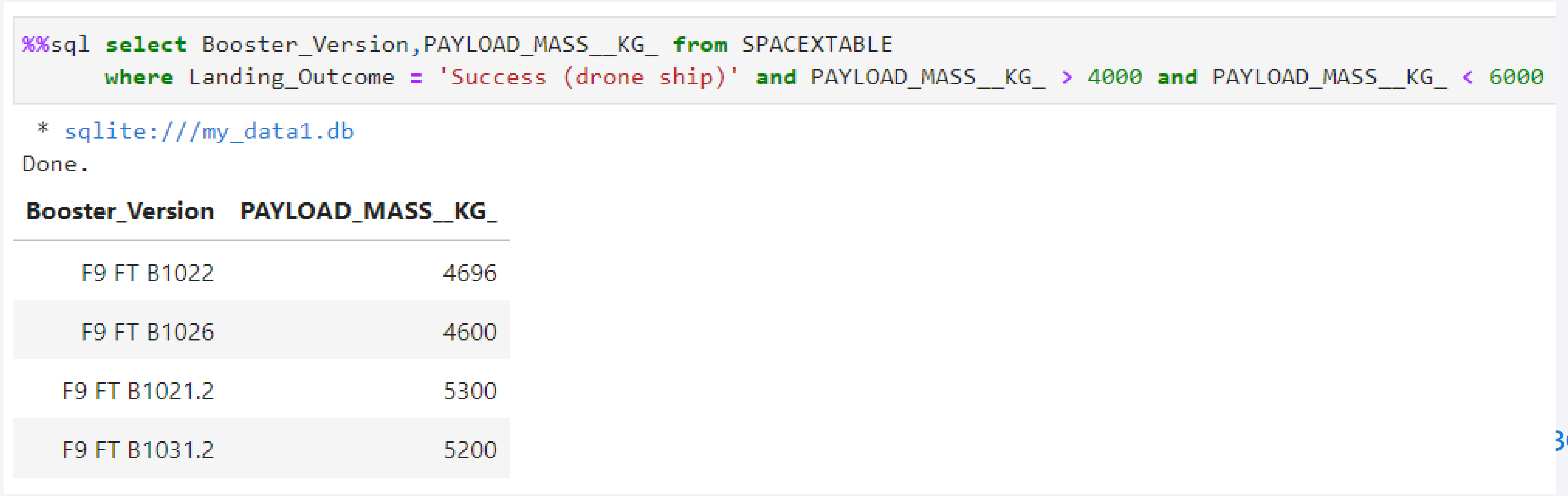
- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015.



```
%sql select min(Date) from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)'

 * sqlite:///my_data1.db
Done.

 min(Date)

2015-12-22
```

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.



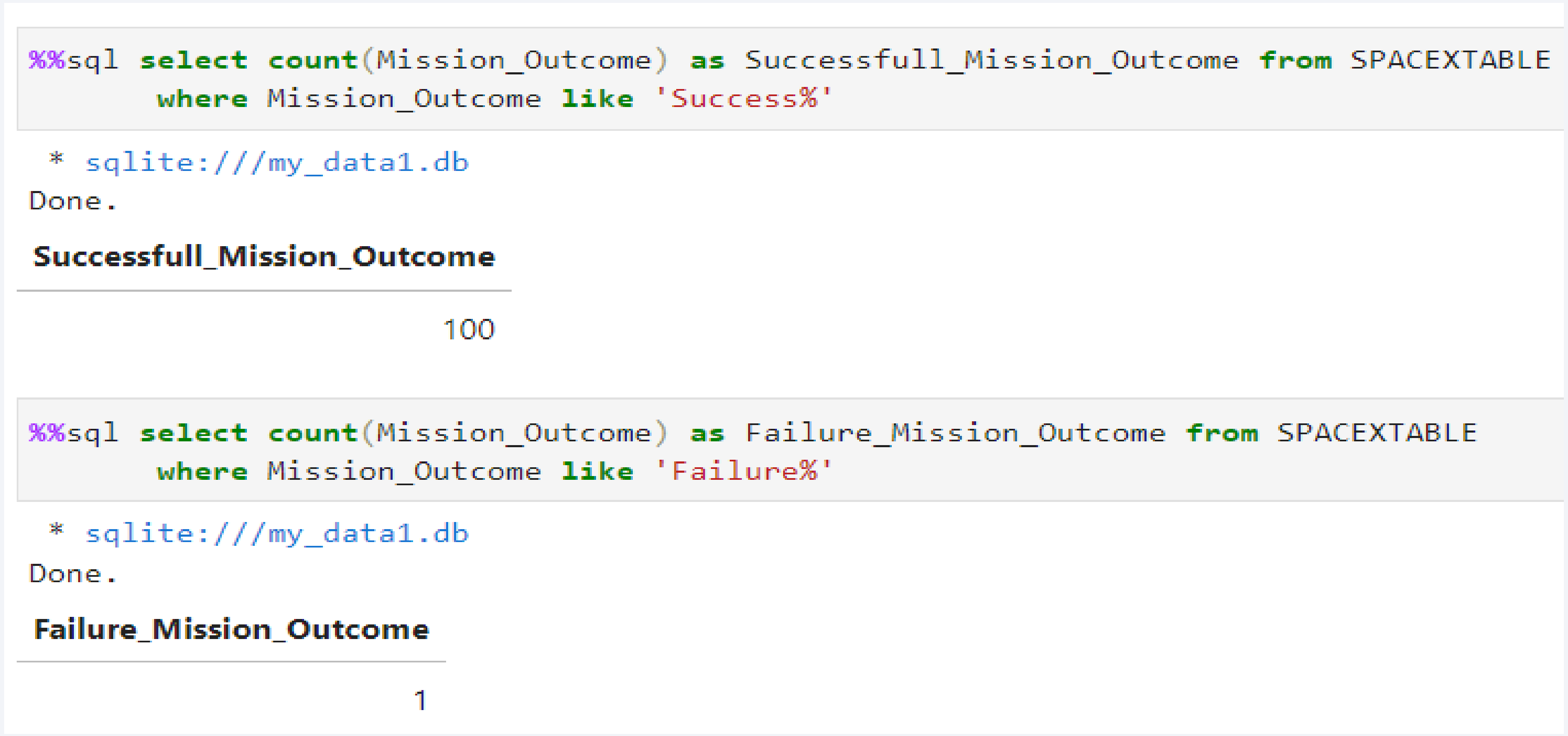
```
%%sql select Booster_Version,PAYLOAD_MASS__KG_ from SPACEXTABLE
      where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

Total Number of Successful and Failure Mission Outcomes

- We used wildcard like '%' to filter for WHERE Mission Outcome was a success or a failure.



```
%%sql select count(Mission_Outcome) as Successfull_Mission_Outcome from SPACEXTABLE
      where Mission_Outcome like 'Success%'
```

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

Successfull_Mission_Outcome
100

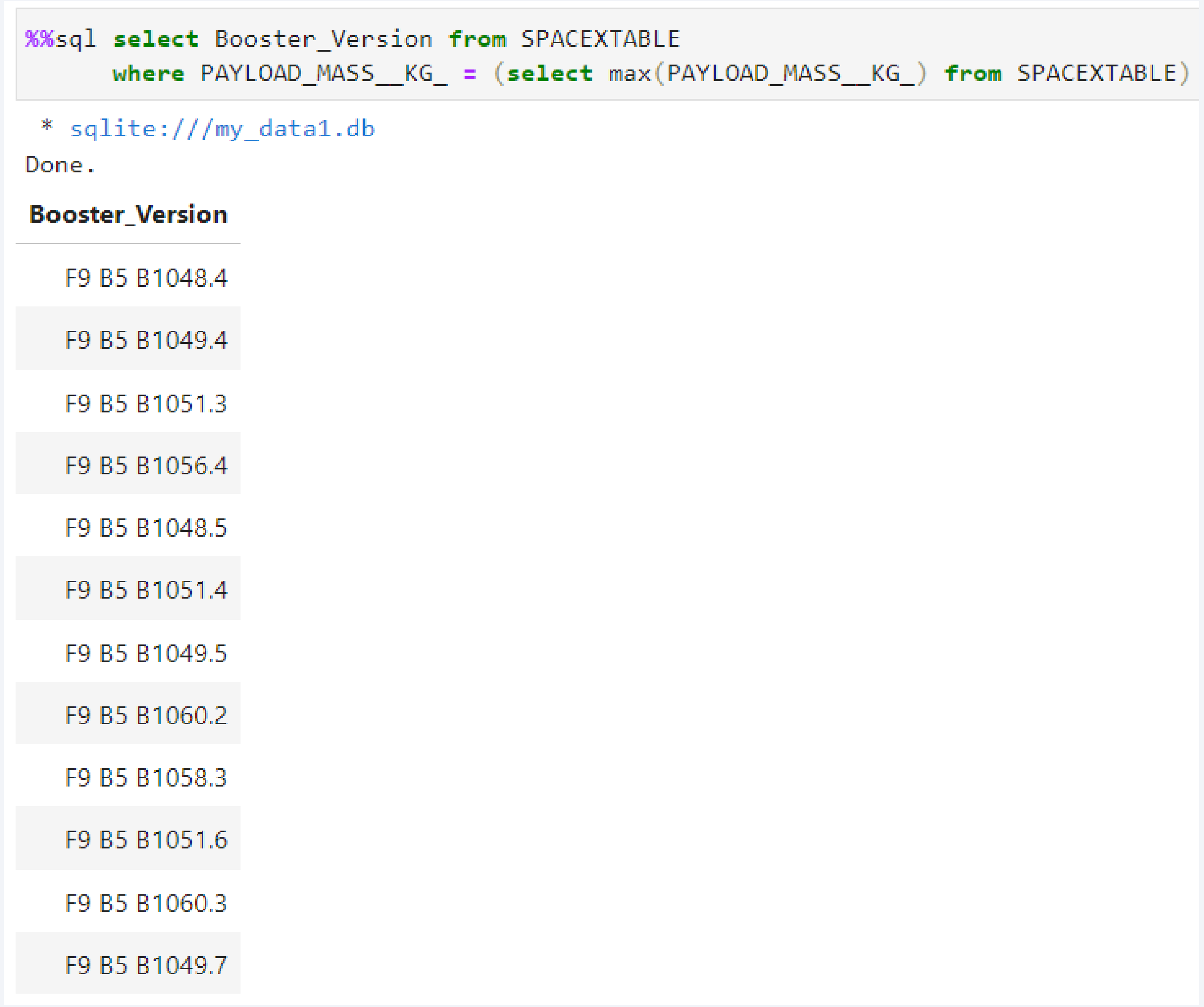
```
%%sql select count(Mission_Outcome) as Failure_Mission_Outcome from SPACEXTABLE
      where Mission_Outcome like 'Failure%'
```

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

Failure_Mission_Outcome
1

Boosters Carried Maximum Payload

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



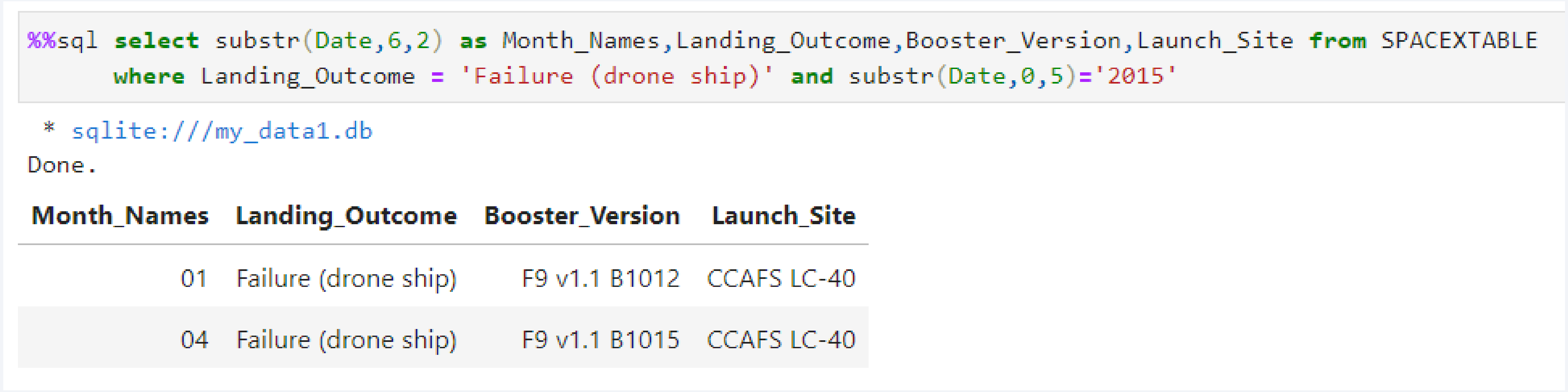
```
%%sql select Booster_Version from SPACEXTABLE
     where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTABLE)
```

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

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, AND conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015.



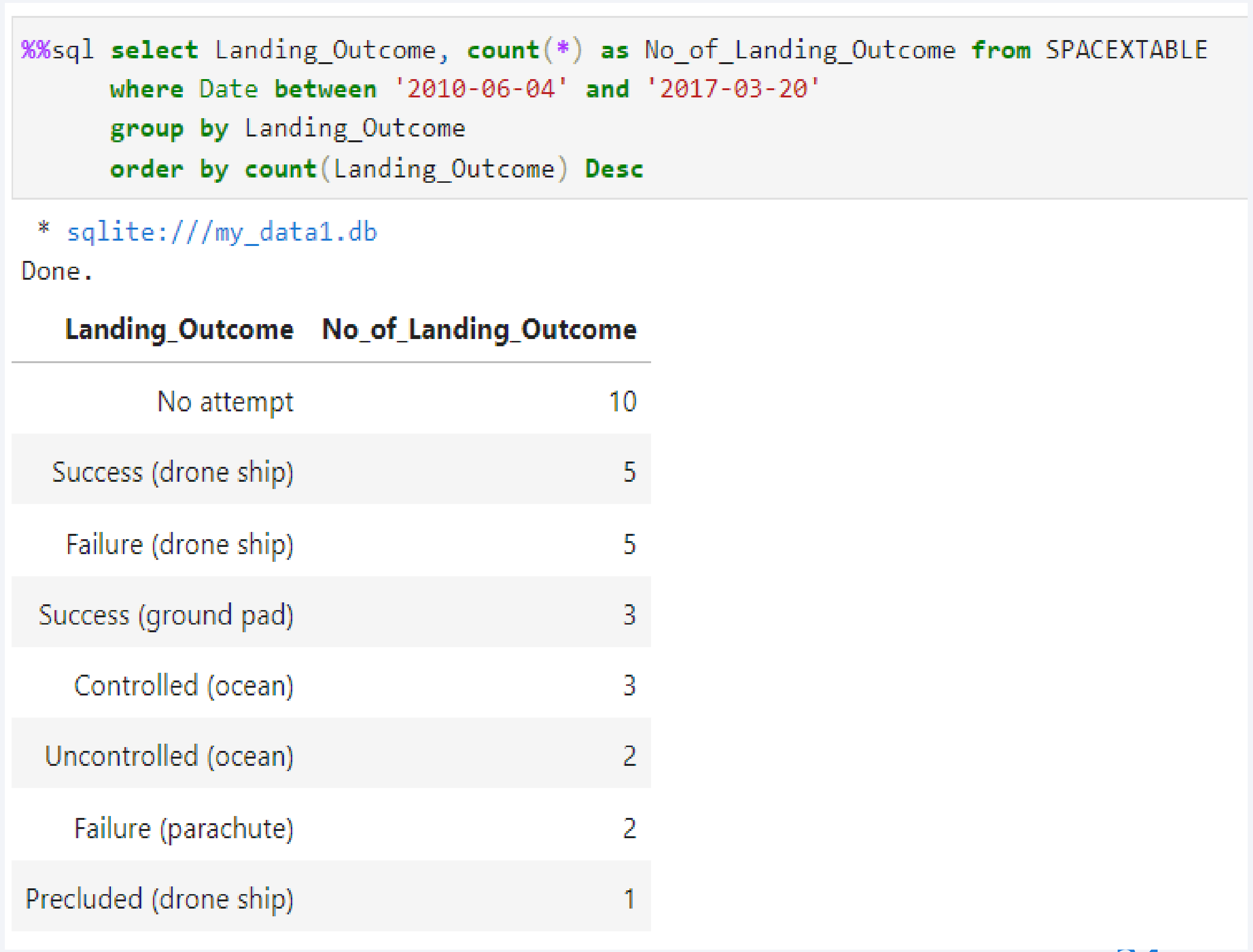
```
%%sql select substr(Date,6,2) as Month_Names,Landing_Outcome,Booster_Version,Launch_Site from SPACEXTABLE
      where Landing_Outcome = 'Failure (drone ship)' and substr(Date,0,5)='2015'
```

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

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

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

- We 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 to 2010-03-20.
- We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order.



```
%%sql select Landing_Outcome, count(*) as No_of_Landing_Outcome from SPACEXTABLE
      where Date between '2010-06-04' and '2017-03-20'
      group by Landing_Outcome
      order by count(Landing_Outcome) Desc
```

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

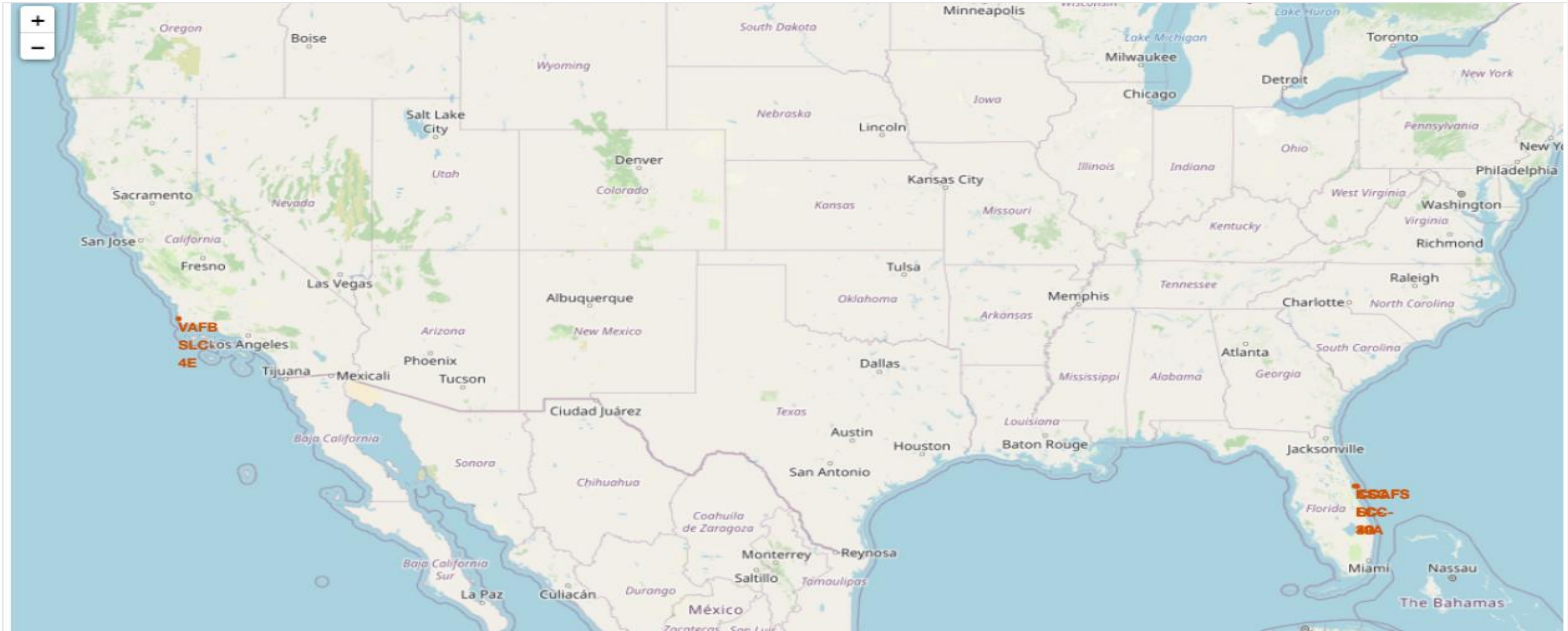
Landing_Outcome	No_of_Landing_Outcome
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

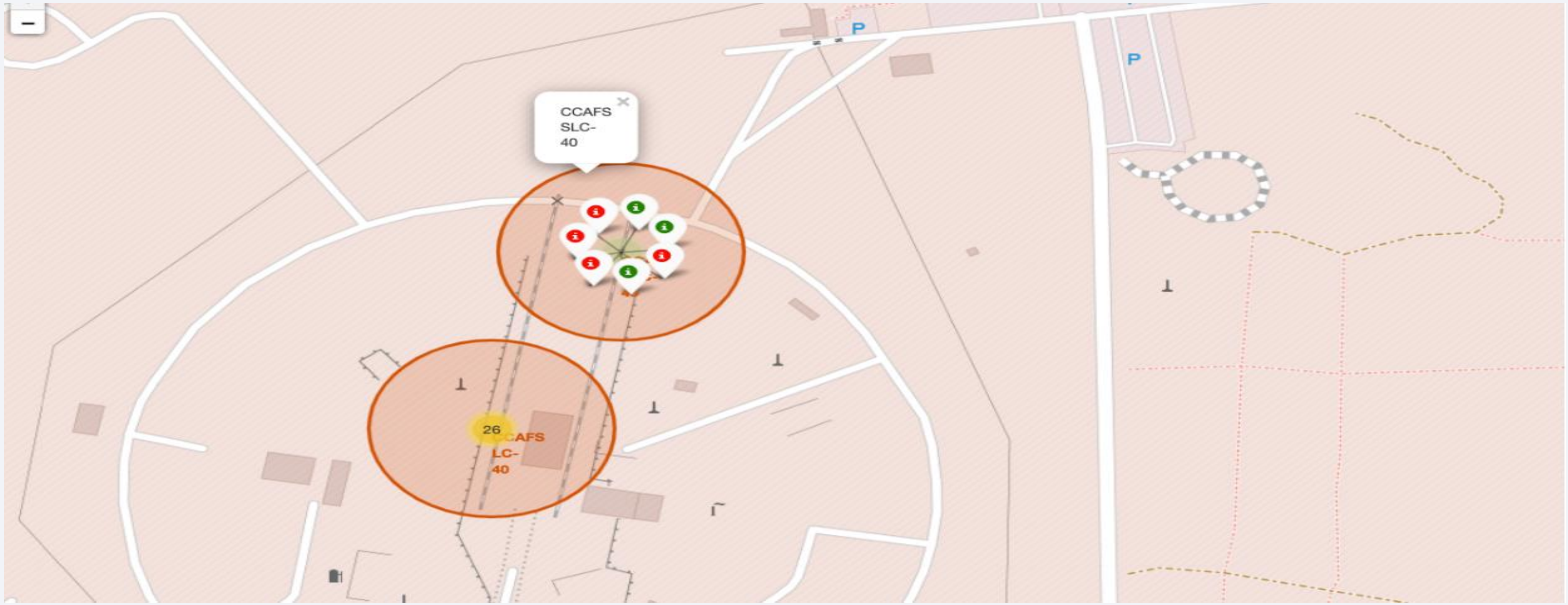
Launch Sites in USA

- All the launch sites are in USA in costal region of Florida and California.



Markers showing launch sites with color labels

- Green Markers shows Success missions and Red Markers shows Failure missions at Launch site of CCAFS SLC-40.



Launch Site Distance near more population areas

- We can see that launch sites are near to costal lines.

Amelia Street
Garland Avenue
Lymmo
Lynx offices
78.62 KM
Lynx Central Station
Central Florida Rail Corridor
Livingston Street
Distance to Railway Station

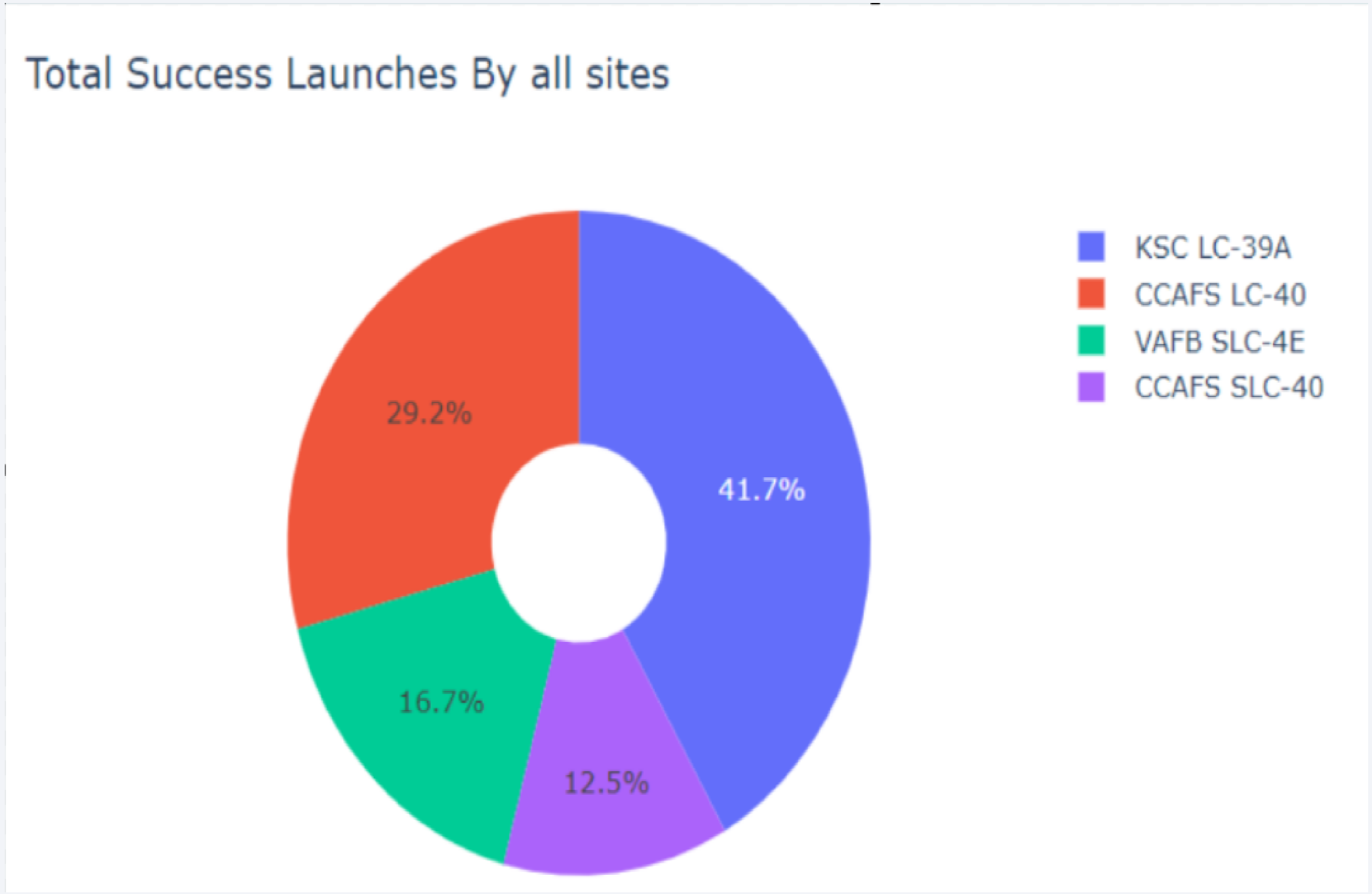
526
Orlando
Rosalind Avenue
Church Street
78.45 KM
Parramore Avenue
Distance to City

7
26
Falcon Launchpad
Centaur Road
Samuel C Phillips Parkway

Section 4

Build a Dashboard with Plotly Dash

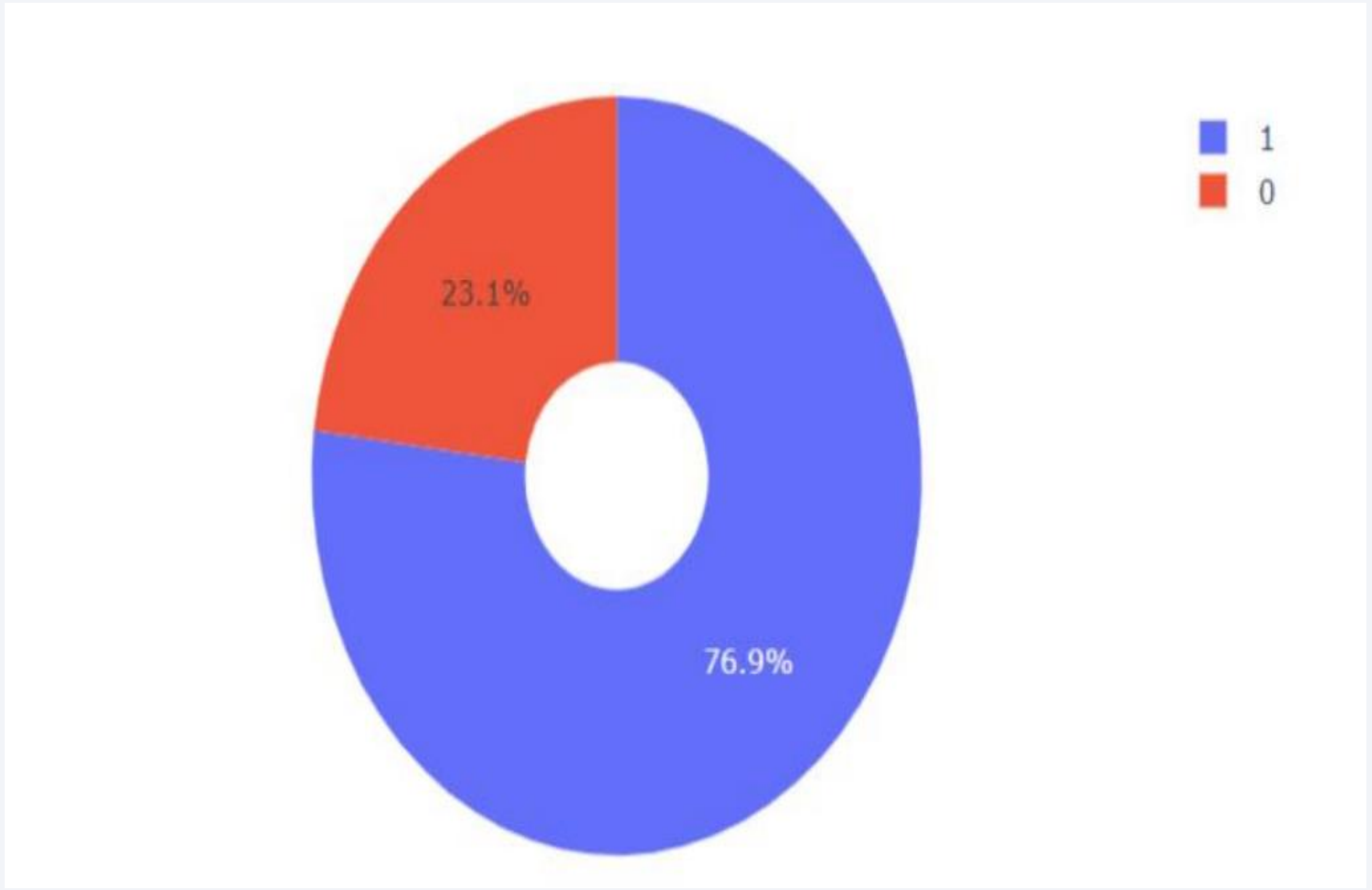
Successfull Launch Sites



- We can see that KSC LC-39A has most successful launch site.

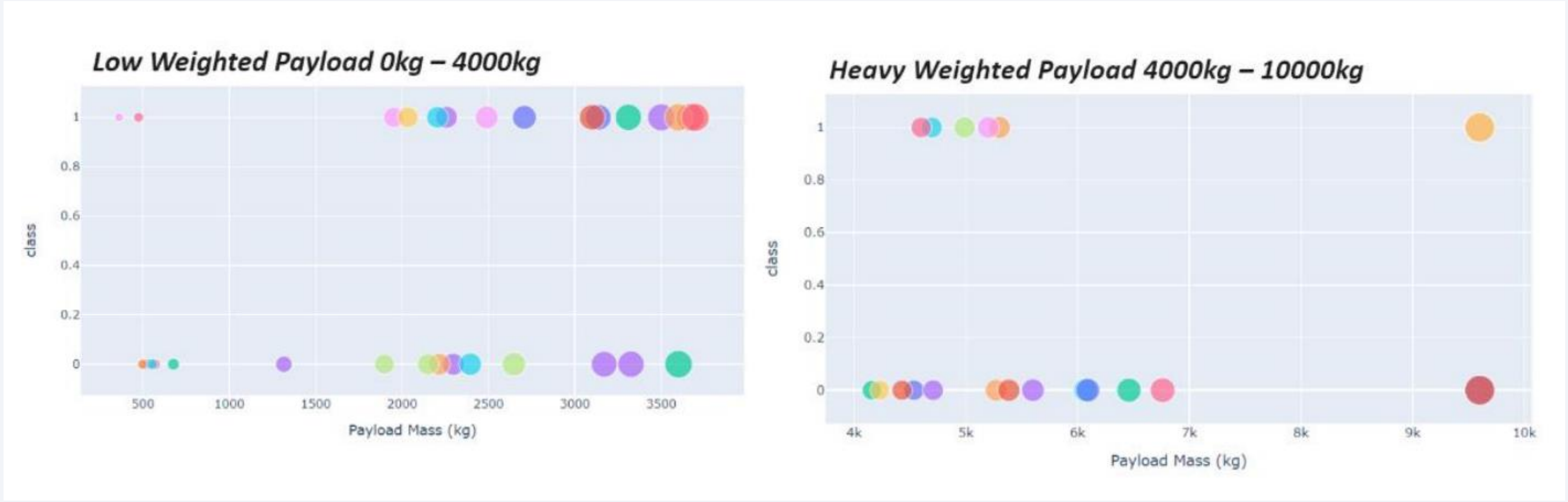
KSC LC-39A Success Rate

- KSC LC-39A has achieved most success rate of 76.9% and with a failure rate of 23.1%



Scatter plot of Payload vs Launch Outcome for all sites

- We can see that low weighted payload has more success rate than heavy weighted payload.

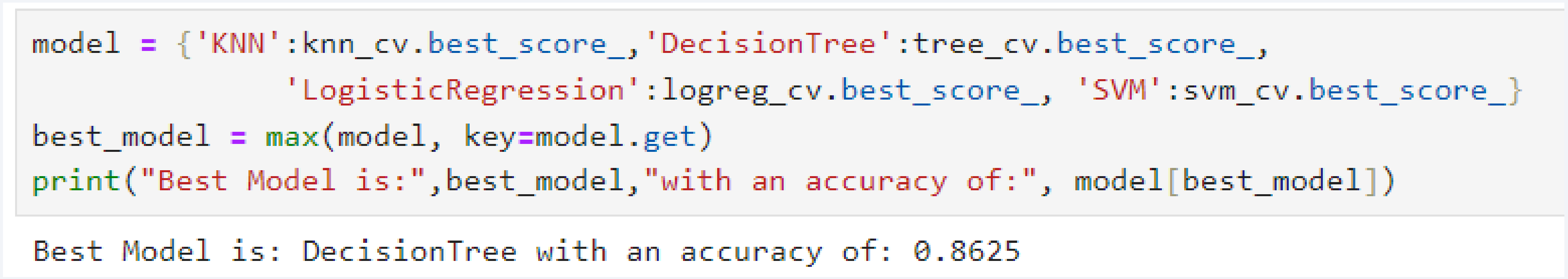


Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The decision tree classifier is the model with the highest classification accuracy.

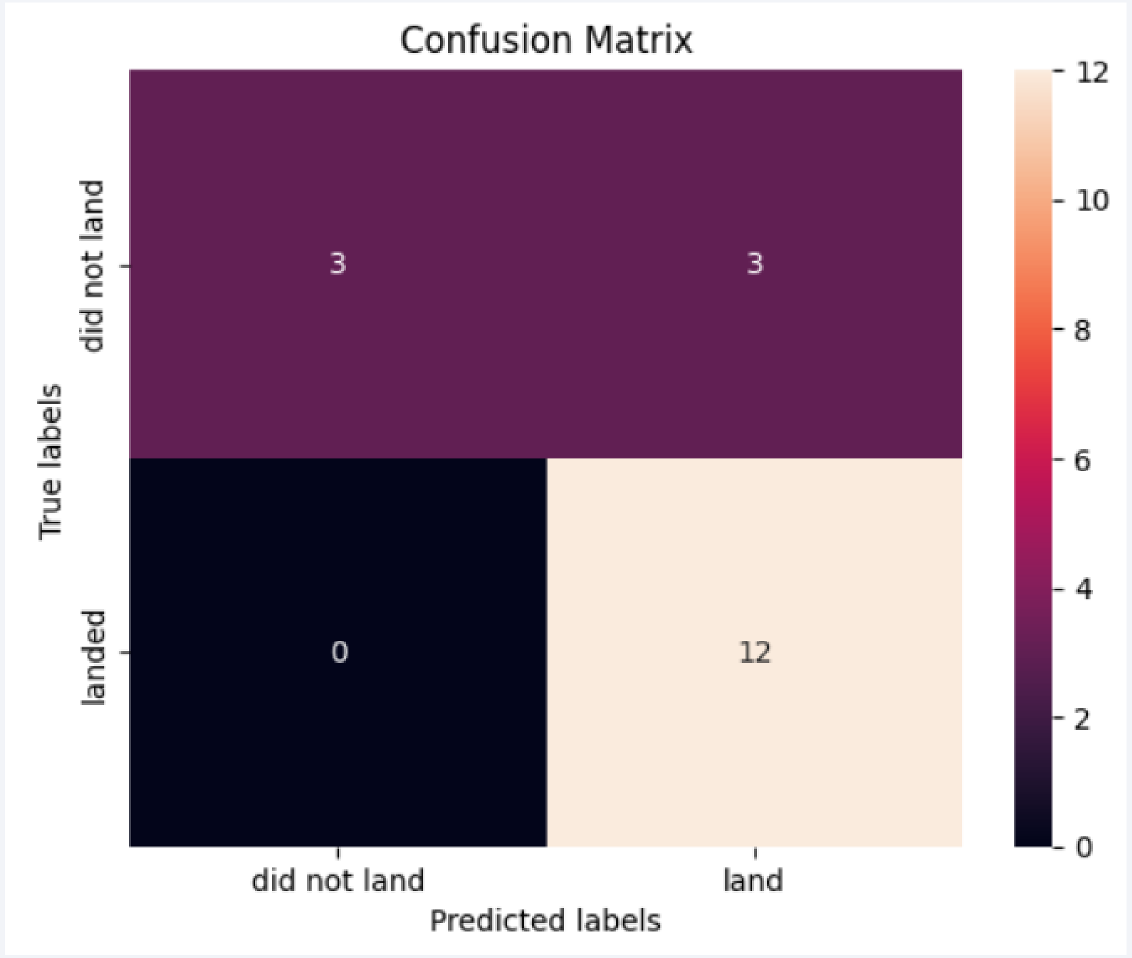


```
model = {'KNN':knn_cv.best_score_,'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_, 'SVM':svm_cv.best_score_}
best_model = max(model, key=model.get)
print("Best Model is:",best_model,"with an accuracy of:", model[best_model])
```

```
Best Model is: DecisionTree with an accuracy of: 0.8625
```

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Orbits ES-L1, GEO, HEO, SSO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Appendix

- GitHub URL of SpaceX API calls notebook https://github.com/sainaren41/Data-Science-Capstone-SpaceX

Thank you!