IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Bhavana Ariga
2nd May 2024

Outline

Executive Summary

- Summary of methodologies
  - ➢ Data Collection through API
  - ➢ Data Collection with Web Scraping
  - ➢ Data Wrangling
  - ➢ Exploratory Data Analysis with SQL
  - ➢ Exploratory Data Analysis with Data Visualization
  - ➢ Interactive Visual Analytics with Folium
  - ➢ Machine Learning Prediction
- Summary of all results
  - ➢ Exploratory Data Analysis result
  - ➢ Interactive analytics in screenshots
  - ➢ Predictive Analytics result

Introduction

- Project background and context

  Space X 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 Space X 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers
  - ➢ What factors determine if the rocket will land successfully?
  - ➢ The interaction amongst various features that determine the success rate of a successful landing.
  - ➢ What operating conditions needs to be in place to ensure a successful landing program.

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 was 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

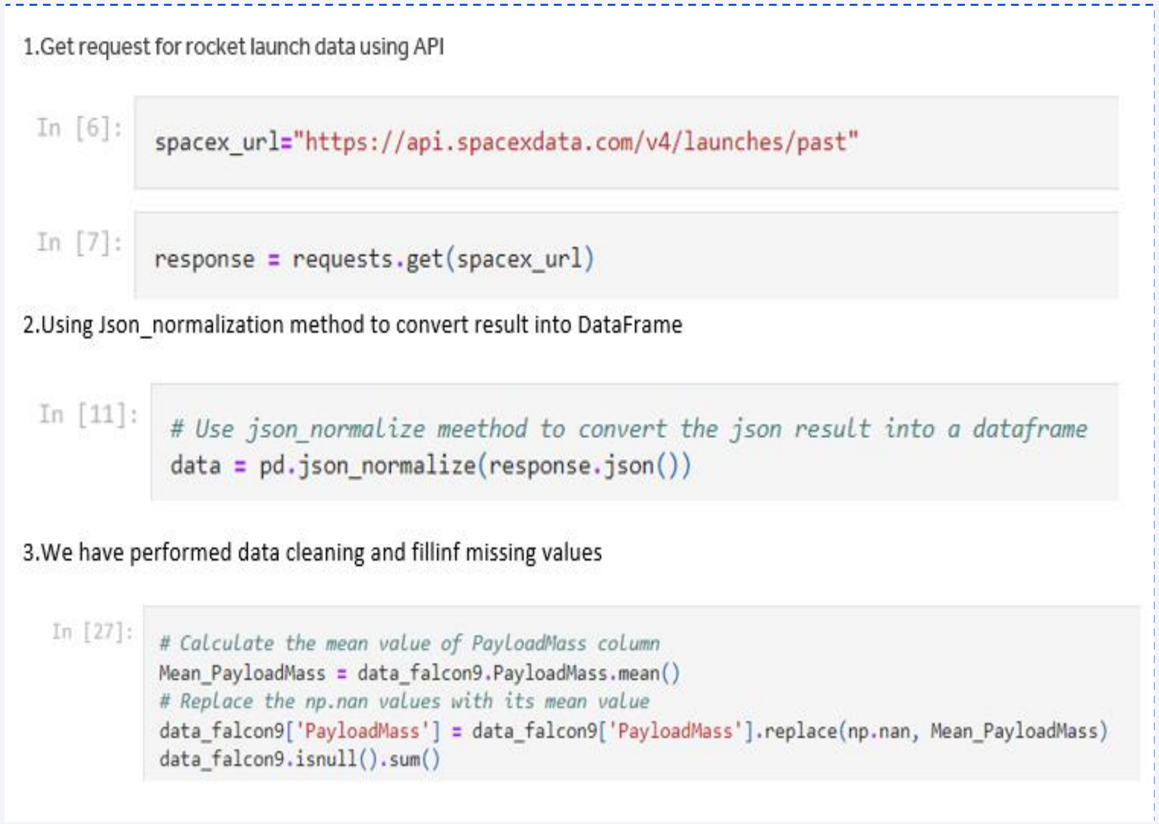
Data Collection

The data was collected using various methods

- Data collection was done using get request to the SpaceX API.
- Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
- We then cleaned the data, checked for missing values and fill in missing values where necessary.
- In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
- The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

- I have used get request tot eh SpaceX API to collect the data, clean the requested data and did some basic data wrangling and formatting.
- The GitHub link to the notebook is https://github.com/arigabhavana/FalconSpaceRace-DataScience/blob/main/Jupyter-Labs-spacex-data-collection-api.ipynb



1.Get request for rocket launch data using API

```
In [6]:  spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
In [7]:  response = requests.get(spacex_url)
```

2.Using Json_normalization method to convert result into DataFrame

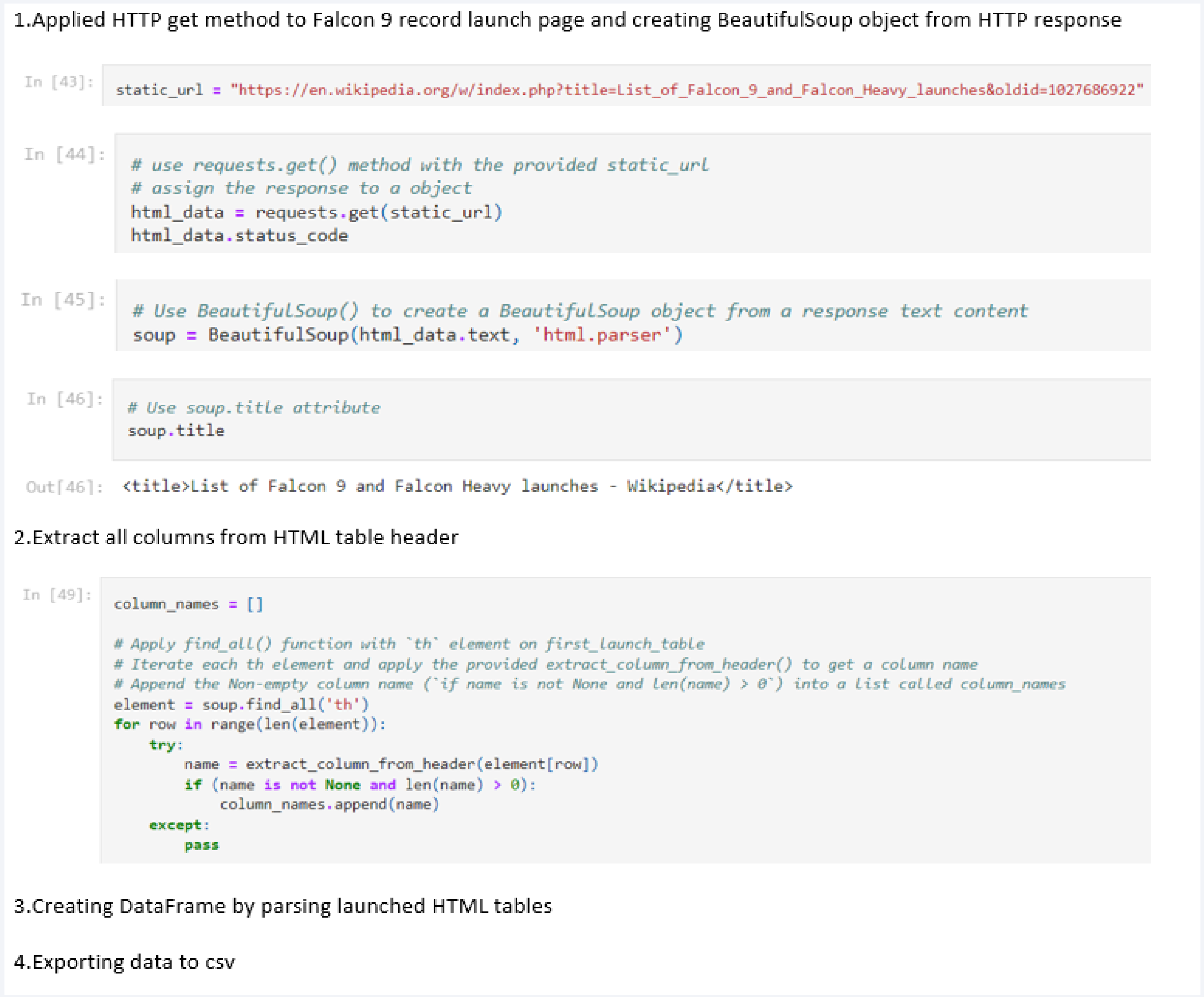
```
In [11]:  # Use json_normalize meethod to convert the json result into a dataframe
          data = pd.json_normalize(response.json())
```

3.We have performed data cleaning and fillinf missing values

```
In [27]:  # Calculate the mean value of PayloadMass column
          Mean_PayloadMass = data_falcon9.PayloadMass.mean()
          # Replace the np.nan values with its mean value
          data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan, Mean_PayloadMass)
          data_falcon9.isnull().sum()
```

Data Collection - Scraping

- Web scrapping is applied to Falcon 9 launch records with BeautifulSoup. We parsed the table and converted it into a pandas dataframe. Exported data into csv.
- The link to the notebook is https://github.com/arigabhavana/FalconSpaceRace-DataScience/blob/main/Jupyter-Labs-webscraping.ipynb



1.Applied HTTP get method to Falcon 9 record launch page and creating BeautifulSoup object from HTTP response

```
In [43]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

In [44]: # use requests.get() method with the provided static_url
         # assign the response to a object
         html_data = requests.get(static_url)
         html_data.status_code

In [45]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
         soup = BeautifulSoup(html_data.text, 'html.parser')

In [46]: # Use soup.title attribute
         soup.title

Out[46]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

2.Extract all columns from HTML table header

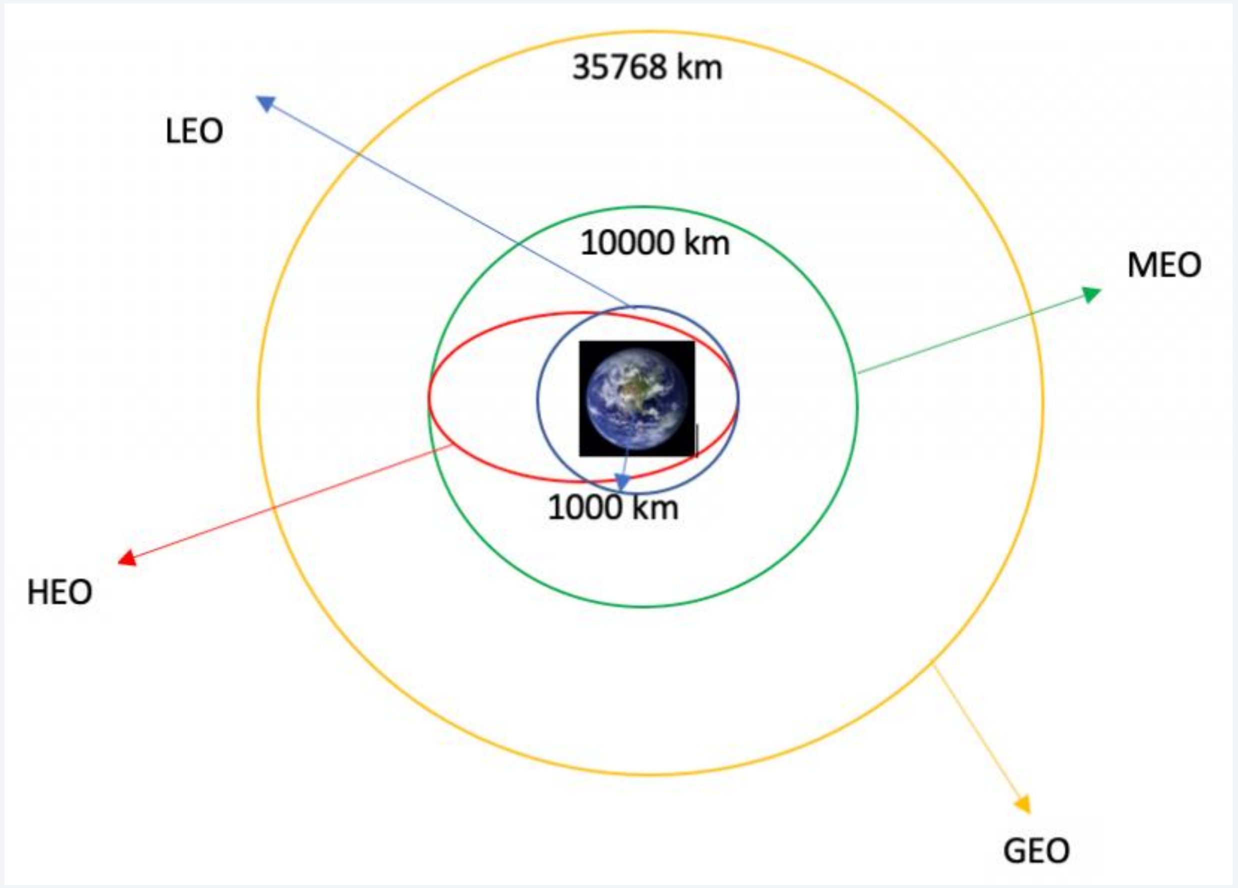
```
In [49]: column_names = []

         # Apply find_all() function with `th` element on first_launch_table
         # Iterate each th element and apply the provided extract_column_from_header() to get a column name
         # Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
         element = soup.find_all('th')
         for row in range(len(element)):
             try:
                 name = extract_column_from_header(element[row])
                 if (name is not None and len(name) > 0):
                     column_names.append(name)
             except:
                 pass
```

3.Creating DataFrame by parsing launched HTML tables

4.Exporting data to csv

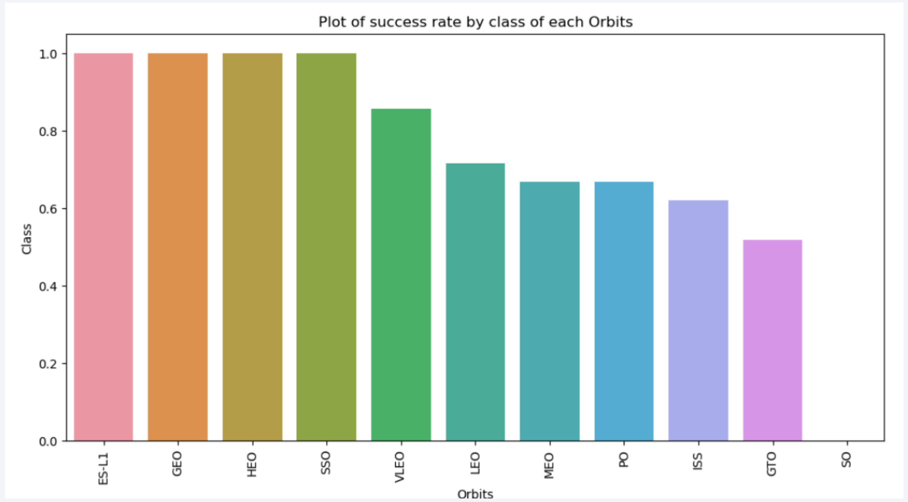
Data Wrangling

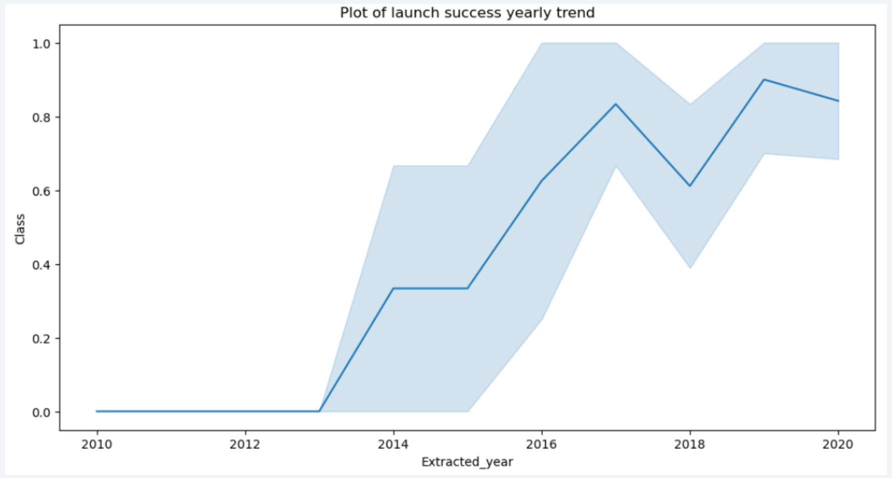
- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is https://github.com/arigabhavana/FalconSpaceRace-DataScience/blob/main/Jupyter-Labs-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- We have explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.
- The link to the Notebook is https://github.com/arigabhavana/FalconSpaceRace-DataScience/blob/main/EDA%20data%20visualization.ipynb





EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
    - ➢ The names of unique launch sites in the space mission and displaying 5 launch records where sites begin with the string ‘CCA’.
    - ➢ The total payload mass carried by boosters launched by NASA (CRS)
    - ➢ The average payload mass carried by booster version F9 v1.1
    - ➢ The total number of successful and failure mission outcomes
    - ➢ 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
    - ➢ The failed landing outcomes in drone ship, their booster version and launch site names.
    - ➢ 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.
- The link to the notebook is https://github.com/arigabhavana/FalconSpaceRace-DataScience/blob/main/EDA%20Sql.ipynb

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
  - ➢ Are launch sites near railways, highways and coastlines.
  - ➢ Do launch sites keep certain distance away from cities.
- The link to the notebook is https://github.com/arigabhavana/FalconSpaceRace-DataScience/blob/main/Map%20with%20Folium.ipynb

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.
- The link to the notebook is https://github.com/arigabhavana/FalconSpaceRace-DataScience/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- The link to the notebook is https://github.com/arigabhavana/FalconSpaceRace-DataScience/blob/main/SpaceX_Machine%20Learning%20Prediction.ipynb

Results

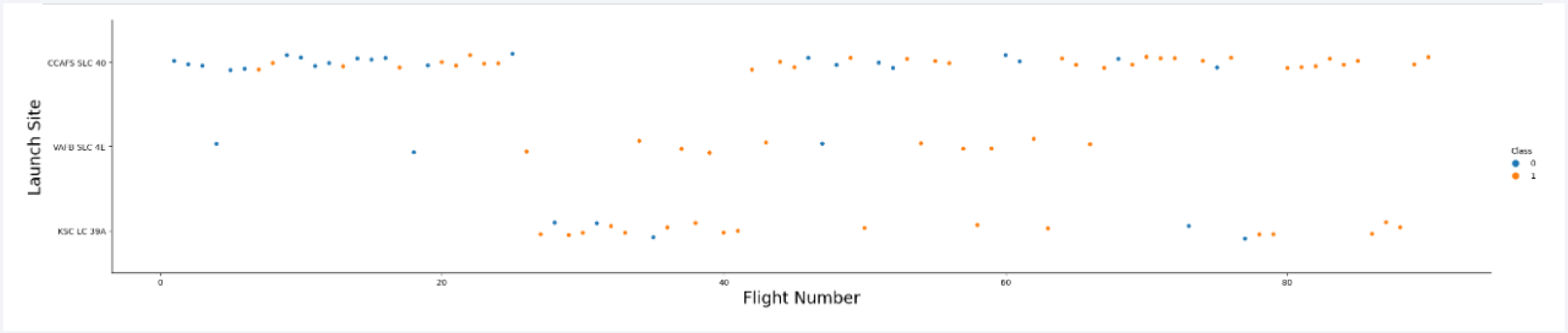
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

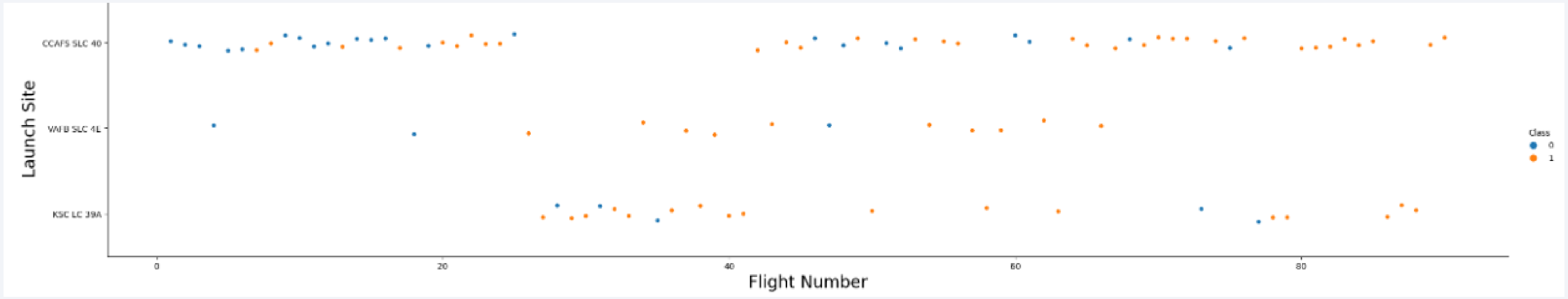
Flight Number vs. Launch Site

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



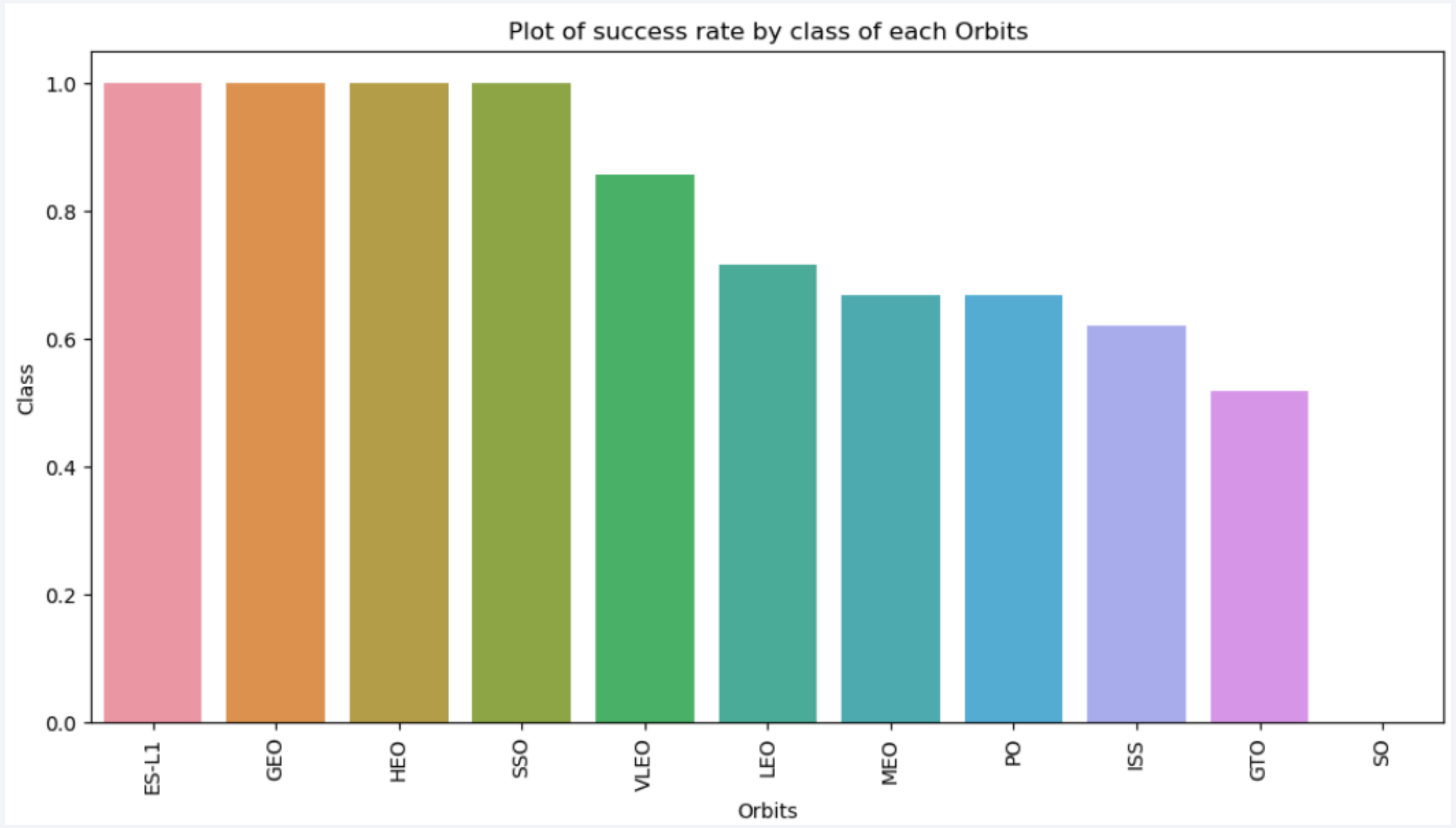
Payload vs. Launch Site

- The greatest payload mass for launch site CCAFS SLC 40 is the higher the success rate for the rocket.



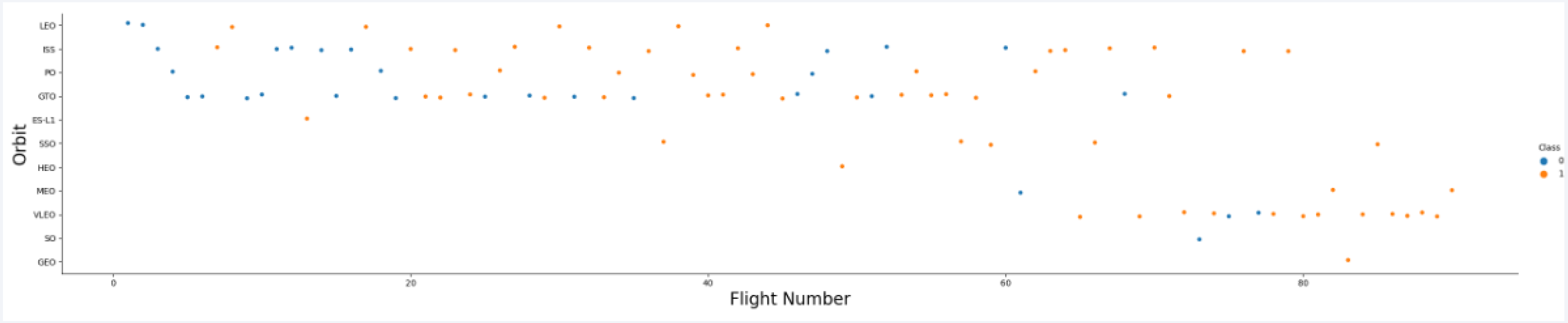
Success Rate vs. Orbit Type

The most success rate is for ES-L1,GEO,HEO,SSO orbits



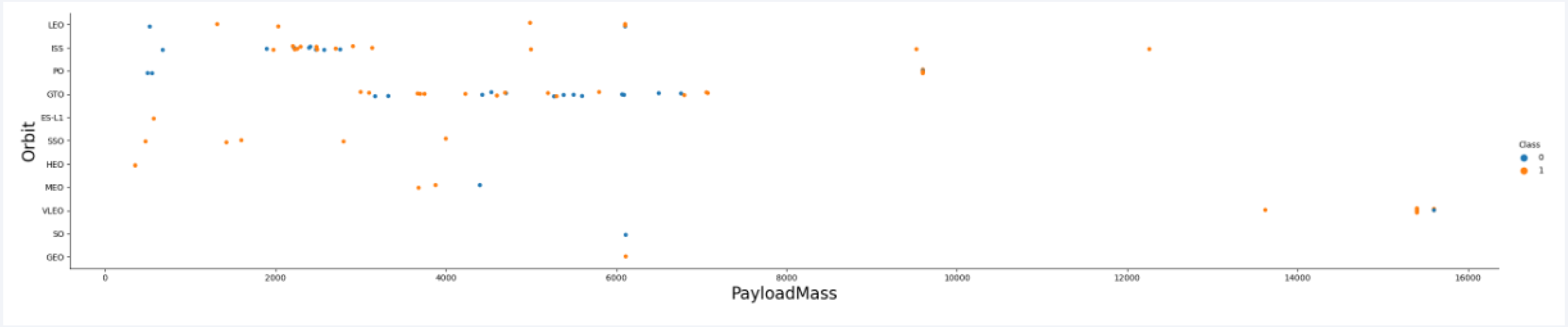
Flight Number vs. Orbit Type

- From below plot, 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.



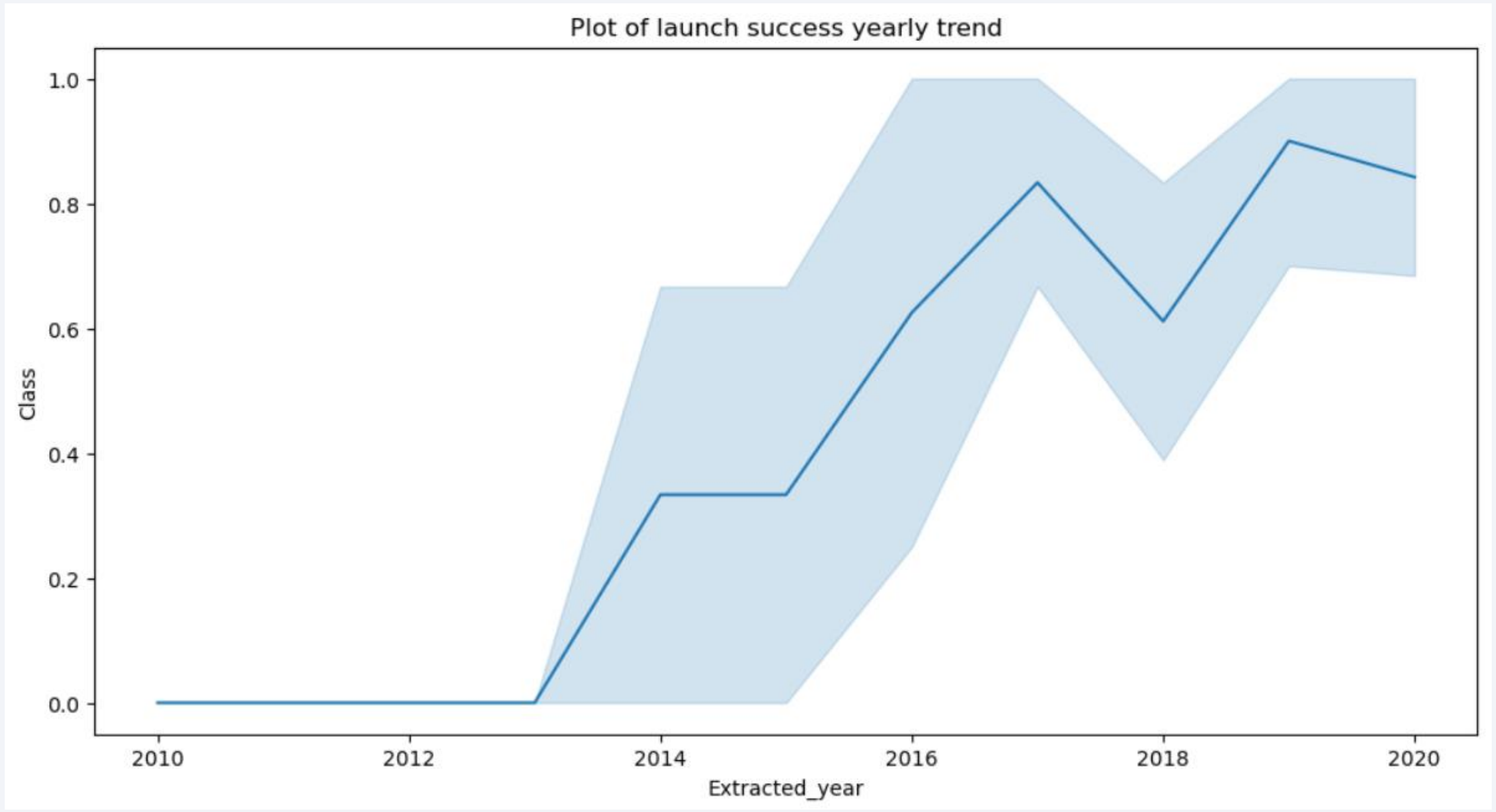
Payload vs. Orbit Type

- From below plot, we observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits



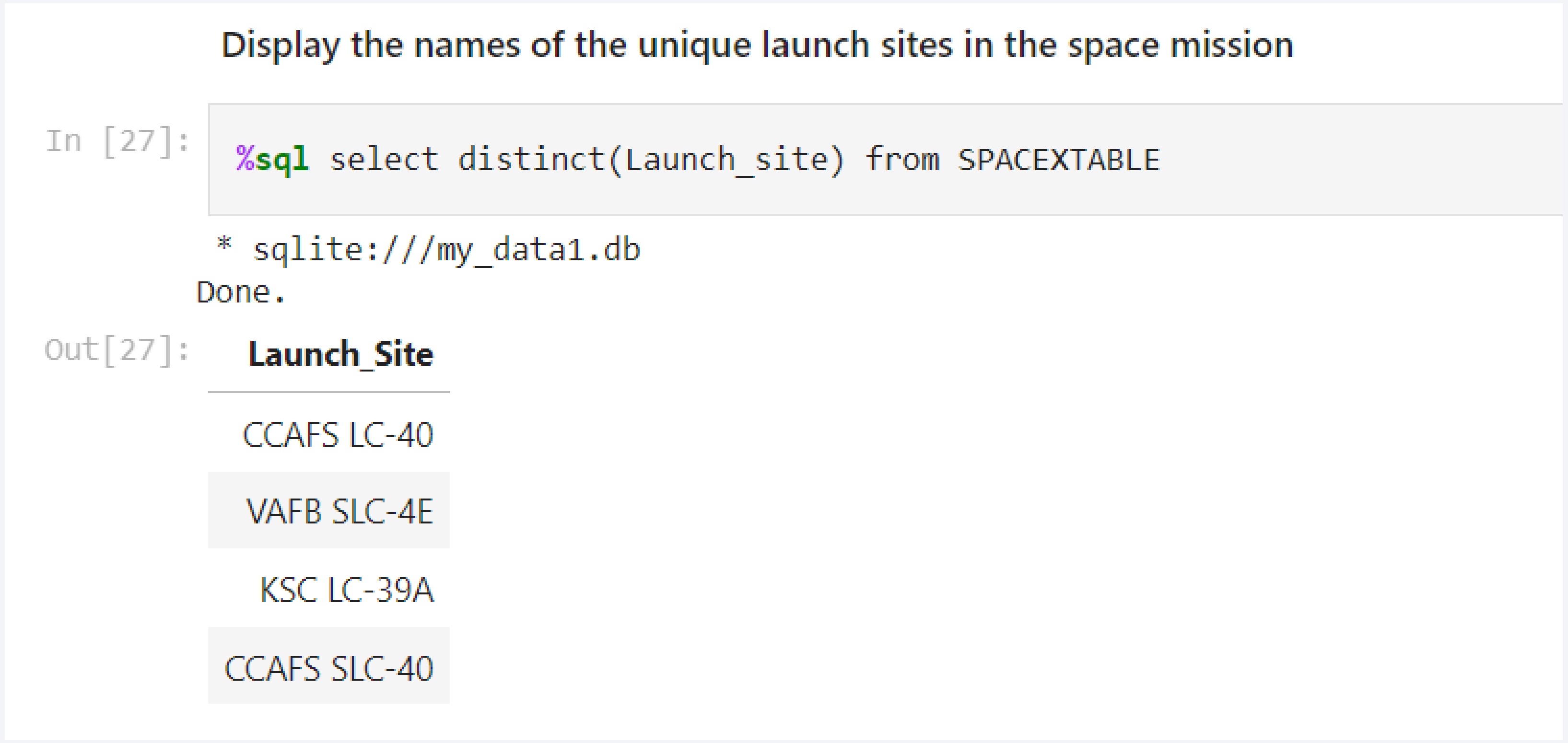
Launch Success Yearly Trend

- From the below, we can observe that success rate is increasing from 2013 till 2020



All Launch Site Names

- We have used DISTINCT to show unique launch sites from SpaceX data



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

```
In [27]: %sql select distinct(Launch_site) from SPACEXTABLE
```

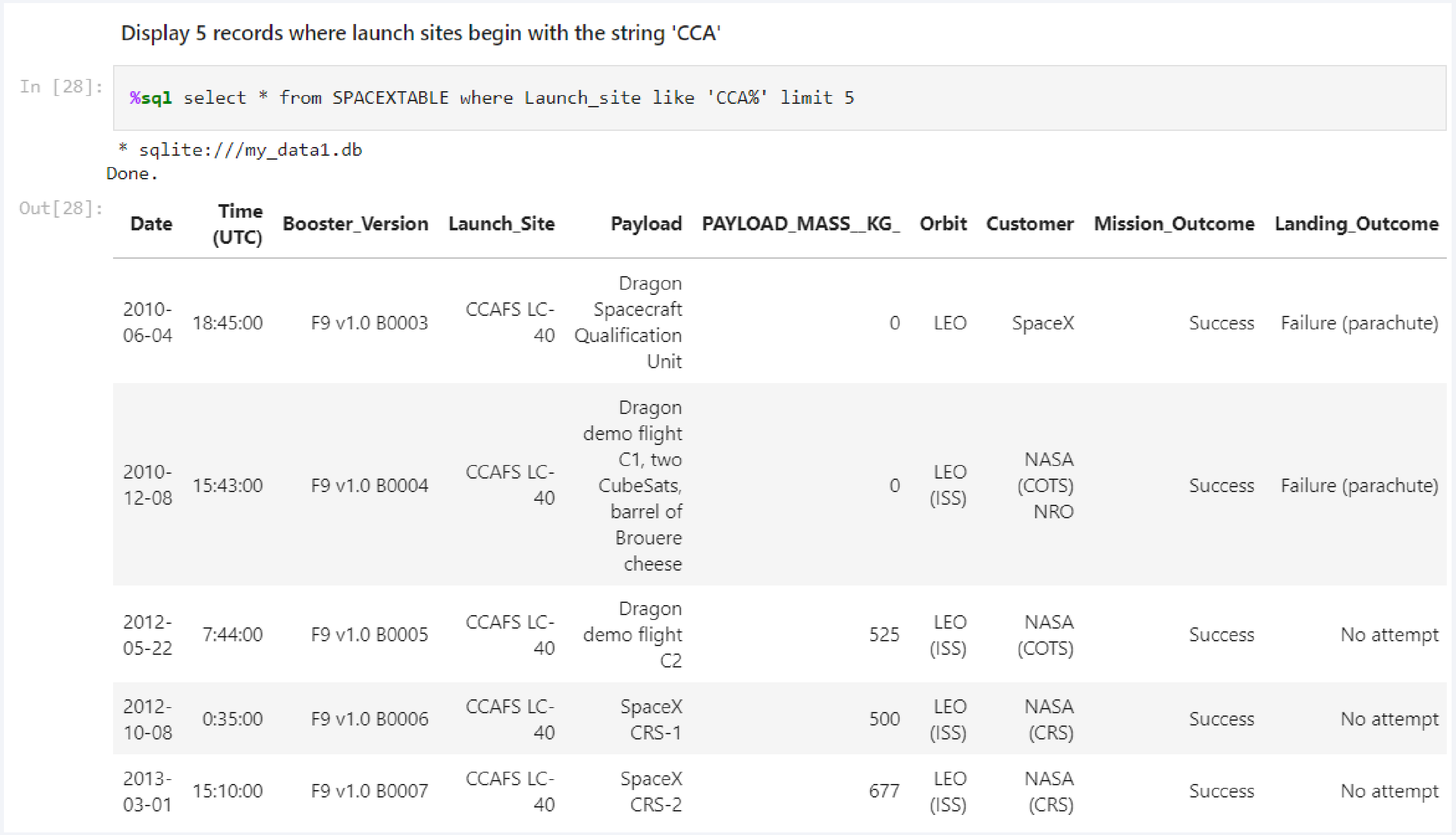
```
 * sqlite:///my_data1.db
Done.
```

Out[27]:

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

Launch Site Names Begin with 'CCA'

- We have used below query to display 5 records where launch sites begin with `CCA`



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

```
In [28]: %sql select * from SPACEXTABLE where Launch_site like 'CCA%' limit 5
```

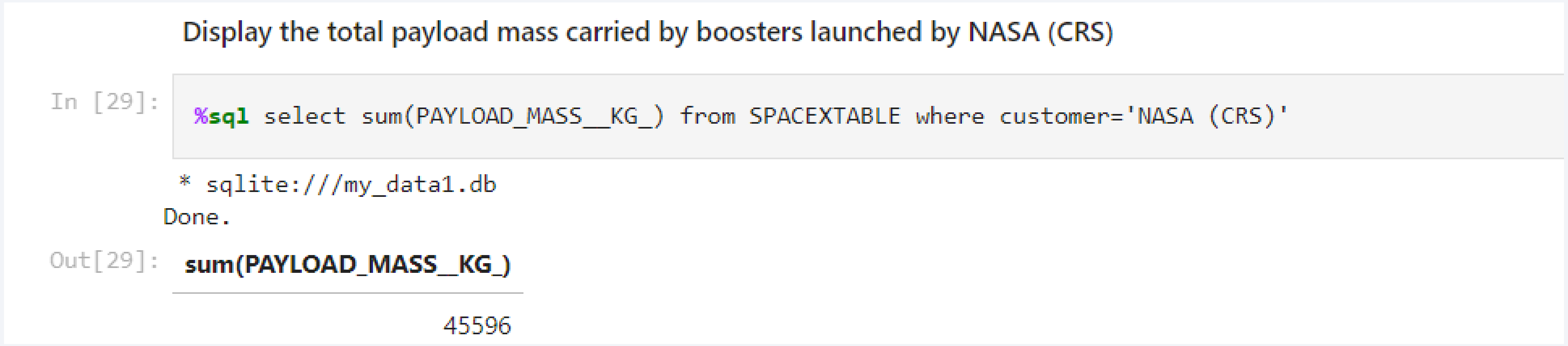
```
 * sqlite:///my_data1.db
Done.
```

Out[28]:

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

- We calculated the total payload carried by boosters from NASA as 45596 using the below query

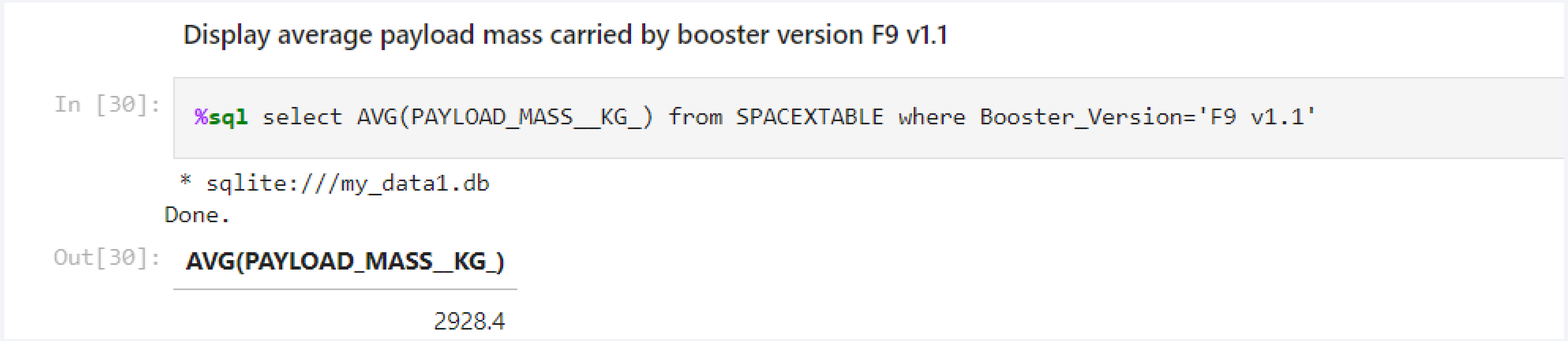


Display the total payload mass carried by boosters launched by NASA (CRS)

```
In [29]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where customer='NASA (CRS)'

 * sqlite:///my_data1.db
Done.

Out[29]: sum(PAYLOAD_MASS__KG_)
                         45596
```

Average Payload Mass by F9 v1.1

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



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

In [30]:
```
%sql select AVG(PAYLOAD_MASS__KG_) from SPACEXTABLE where Booster_Version='F9 v1.1'
```

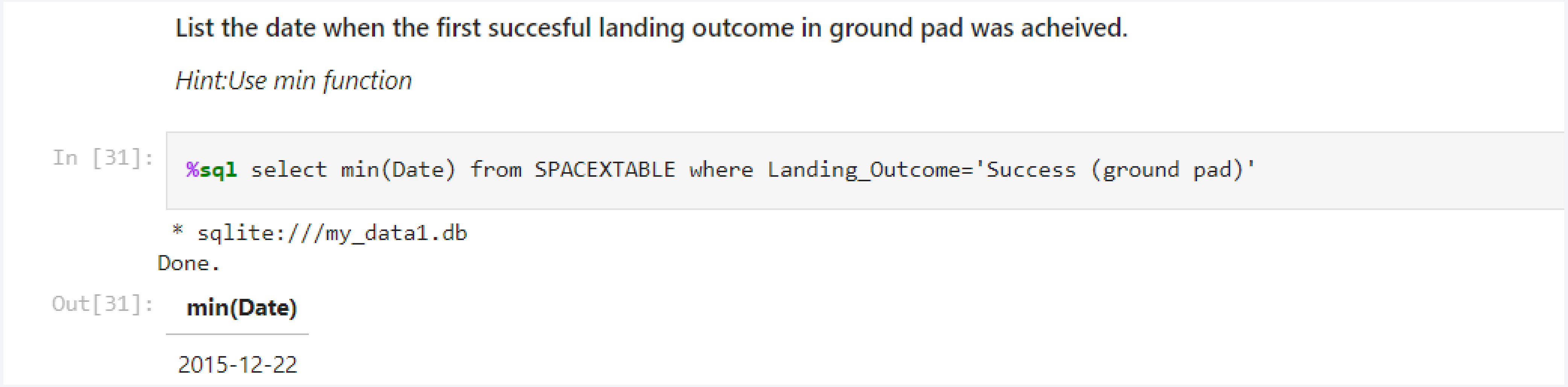
```
 * sqlite:///my_data1.db
Done.
```

Out[30]:

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- We observe that first successful landing outcome on ground pad is 22nd Dec 2015



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

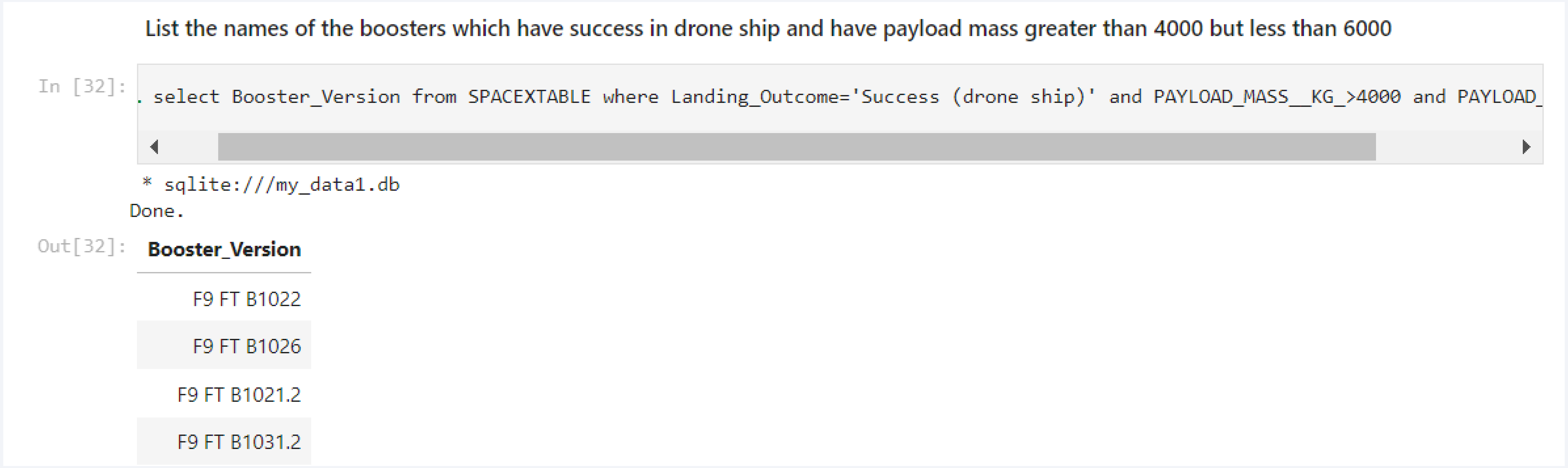
Hint:Use min function

```
In [31]: %sql select min(Date) from SPACEXTABLE where Landing_Outcome='Success (ground pad)'

 * sqlite:///my_data1.db
Done.

Out[31]:
```

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



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

```
In [32]: . select Booster_Version from SPACEXTABLE where Landing_Outcome='Success (drone ship)' and PAYLOAD_MASS__KG_>4000 and PAYLOAD_
```

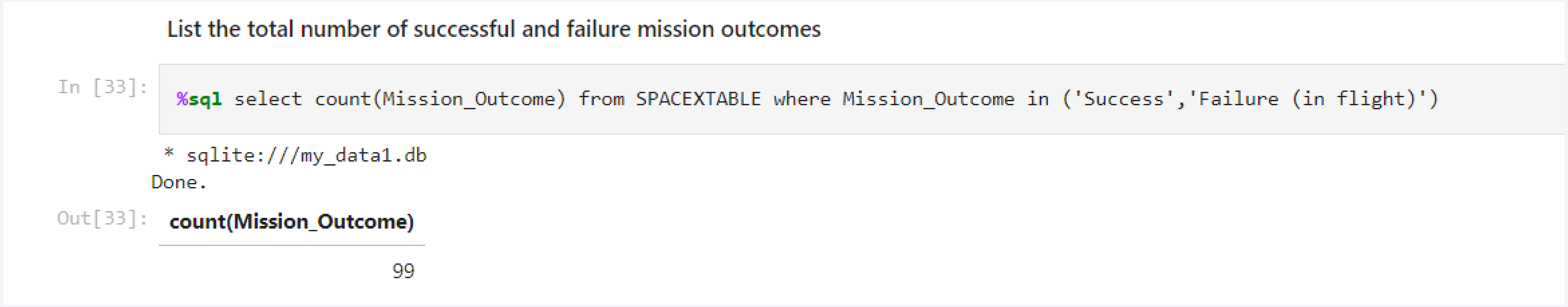
```
 * sqlite:///my_data1.db
Done.
```

```
Out[32]:
```

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 calculated the total number of successful and failure mission outcomes as 99 using below query



List the total number of successful and failure mission outcomes

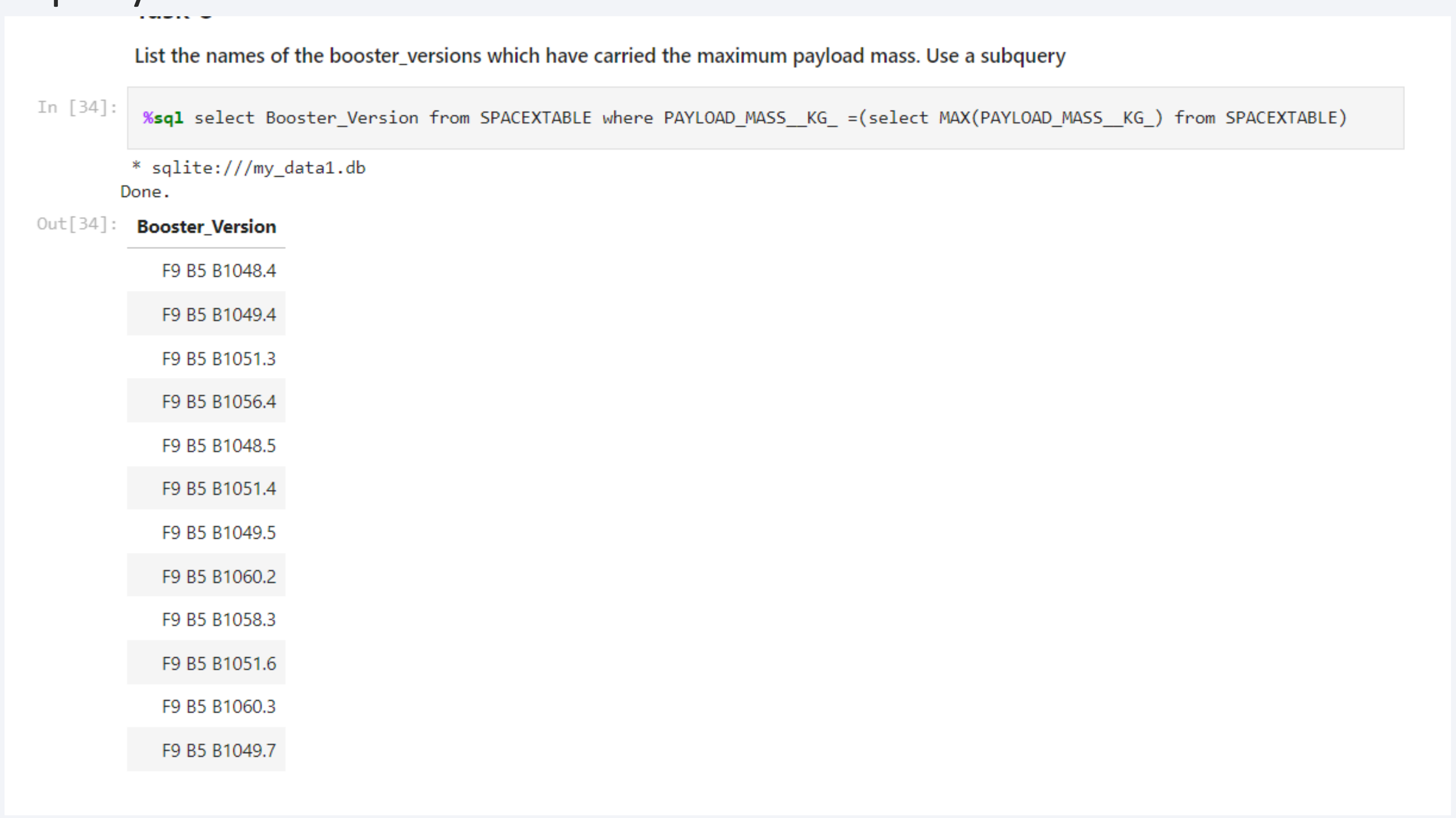
```
In [33]: %sql select count(Mission_Outcome) from SPACEXTABLE where Mission_Outcome in ('Success','Failure (in flight)')
```

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

```
Out[33]: count(Mission_Outcome)
                             99
```

Boosters Carried Maximum Payload

- We have listed the names of the booster which have carried the maximum payload mass using below query



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

```
In [34]: %sql select Booster_Version from SPACEXTABLE where PAYLOAD_MASS__KG_ =(select MAX(PAYLOAD_MASS__KG_) from SPACEXTABLE)
```

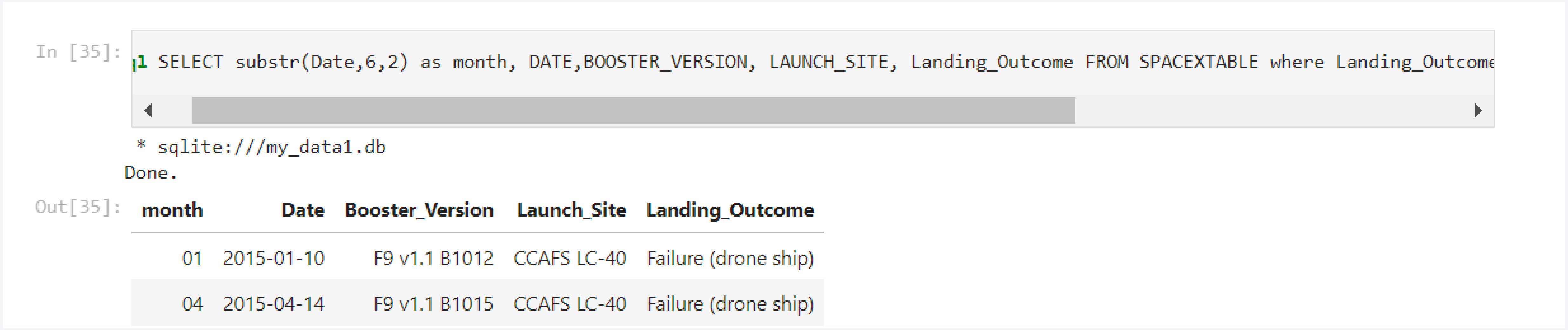
```
 * sqlite:///my_data1.db
Done.
```

Out[34]:

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



```
In [35]: l SELECT substr(Date,6,2) as month, DATE,BOOSTER_VERSION, LAUNCH_SITE, Landing_Outcome FROM SPACEXTABLE where Landing_Outcome
```

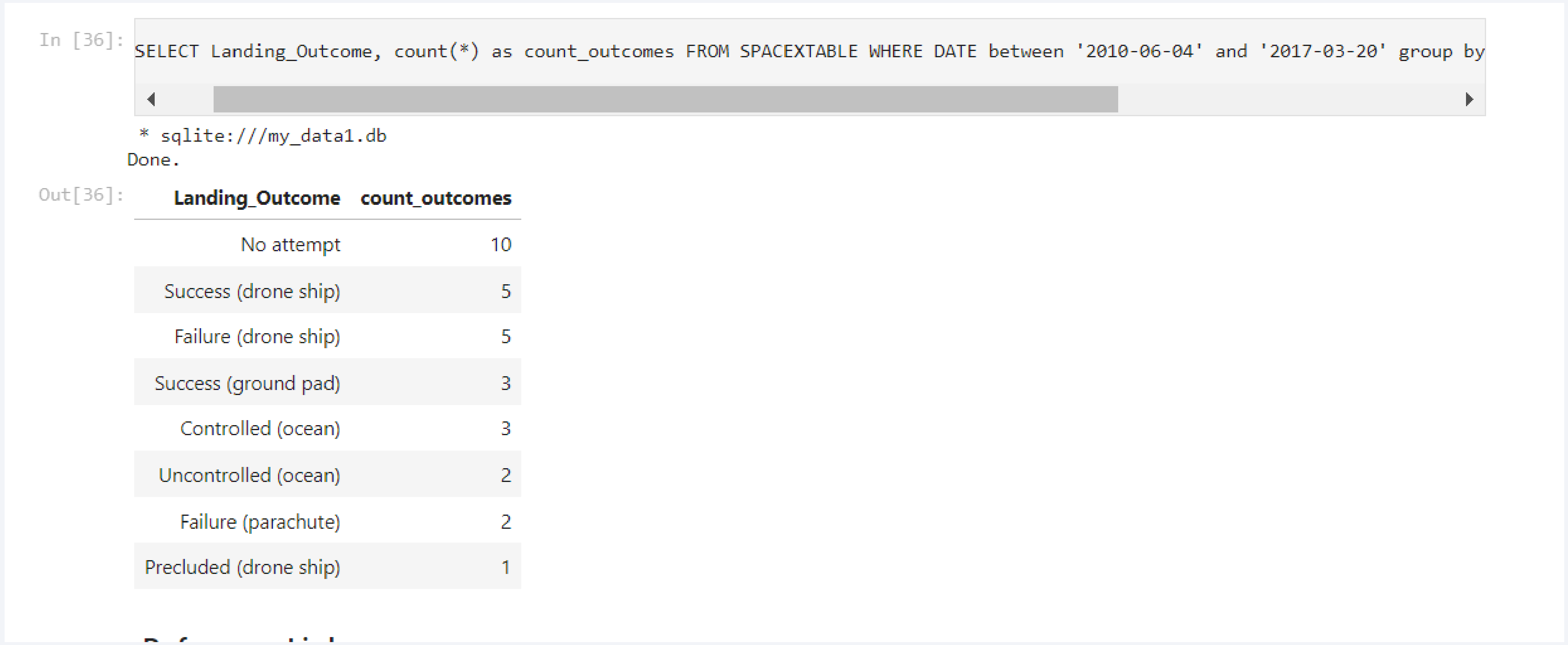
```
 * sqlite:///my_data1.db
Done.
```

Out[35]:

month	Date	Booster_Version	Launch_Site	Landing_Outcome
01	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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. and 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.



```
In [36]: SELECT Landing_Outcome, count(*) as count_outcomes FROM SPACEXTABLE WHERE DATE between '2010-06-04' and '2017-03-20' group by
```

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

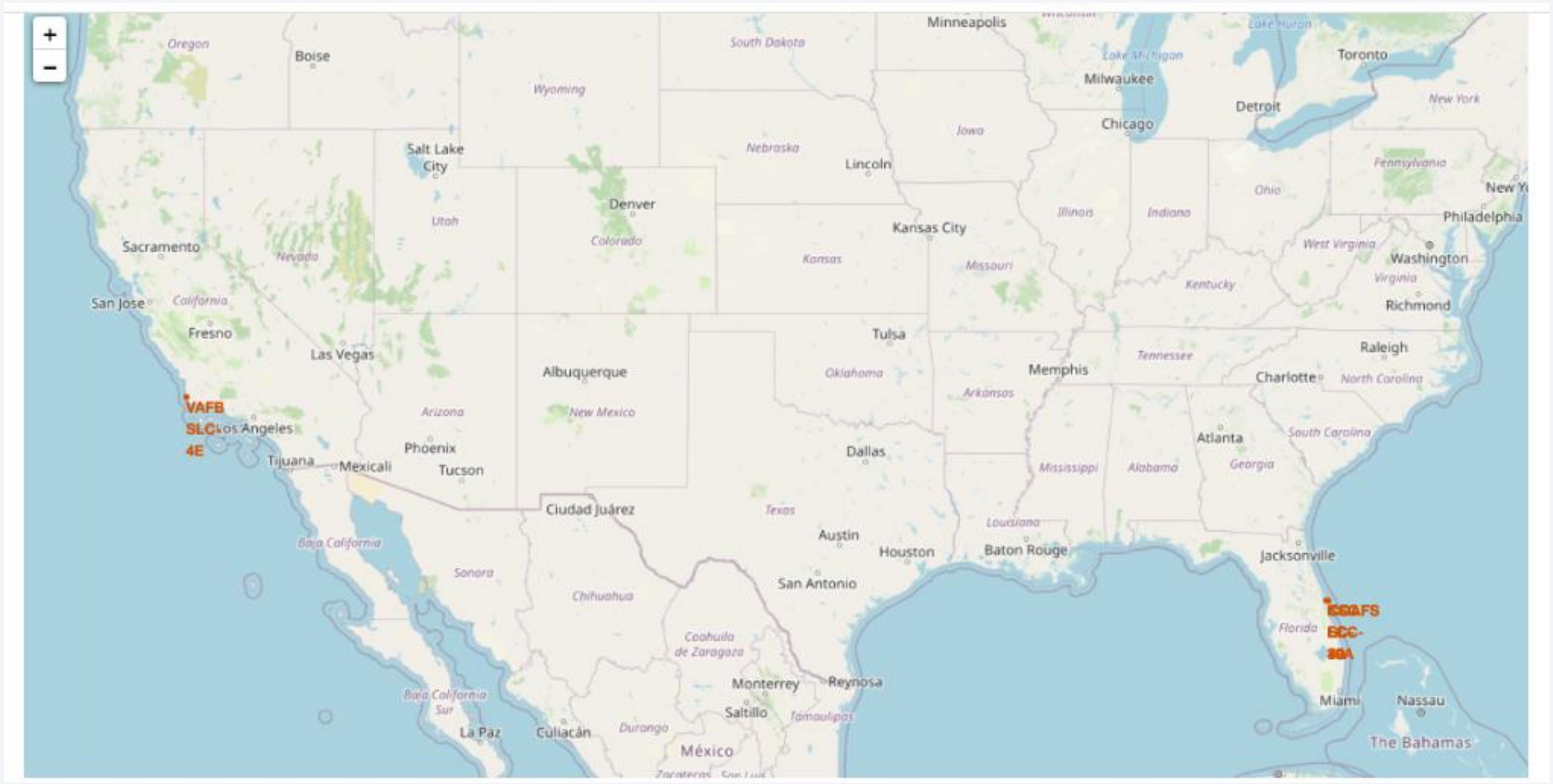
Out[36]:

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

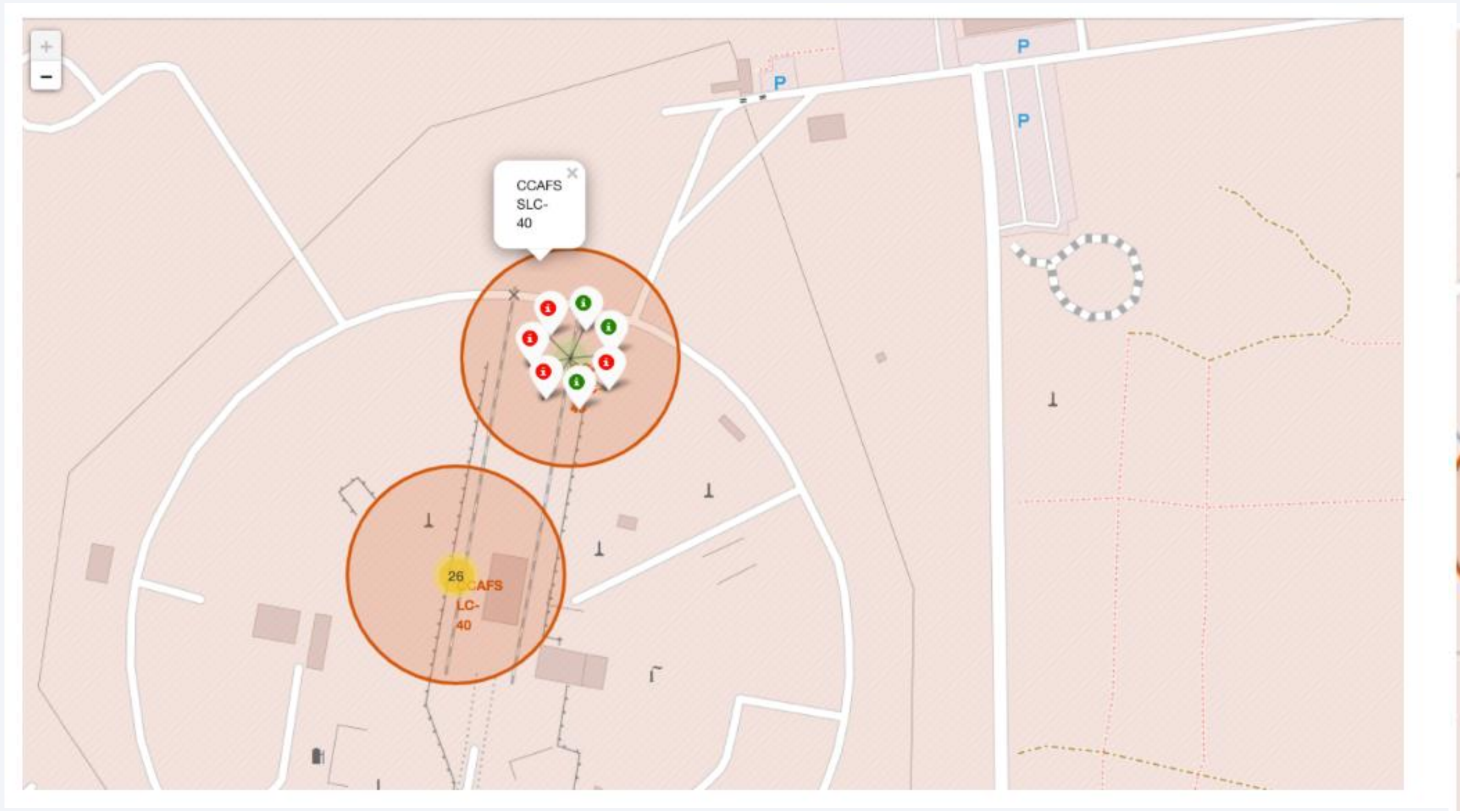
Section 3

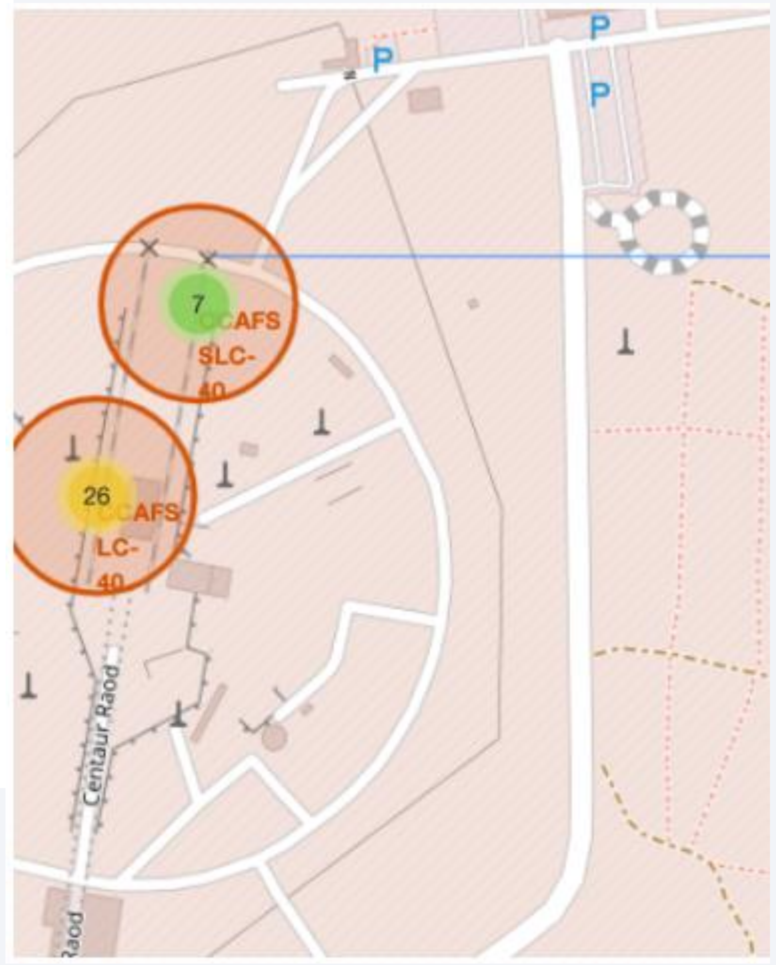
Launch Sites Proximities Analysis

All Launch sites on Map

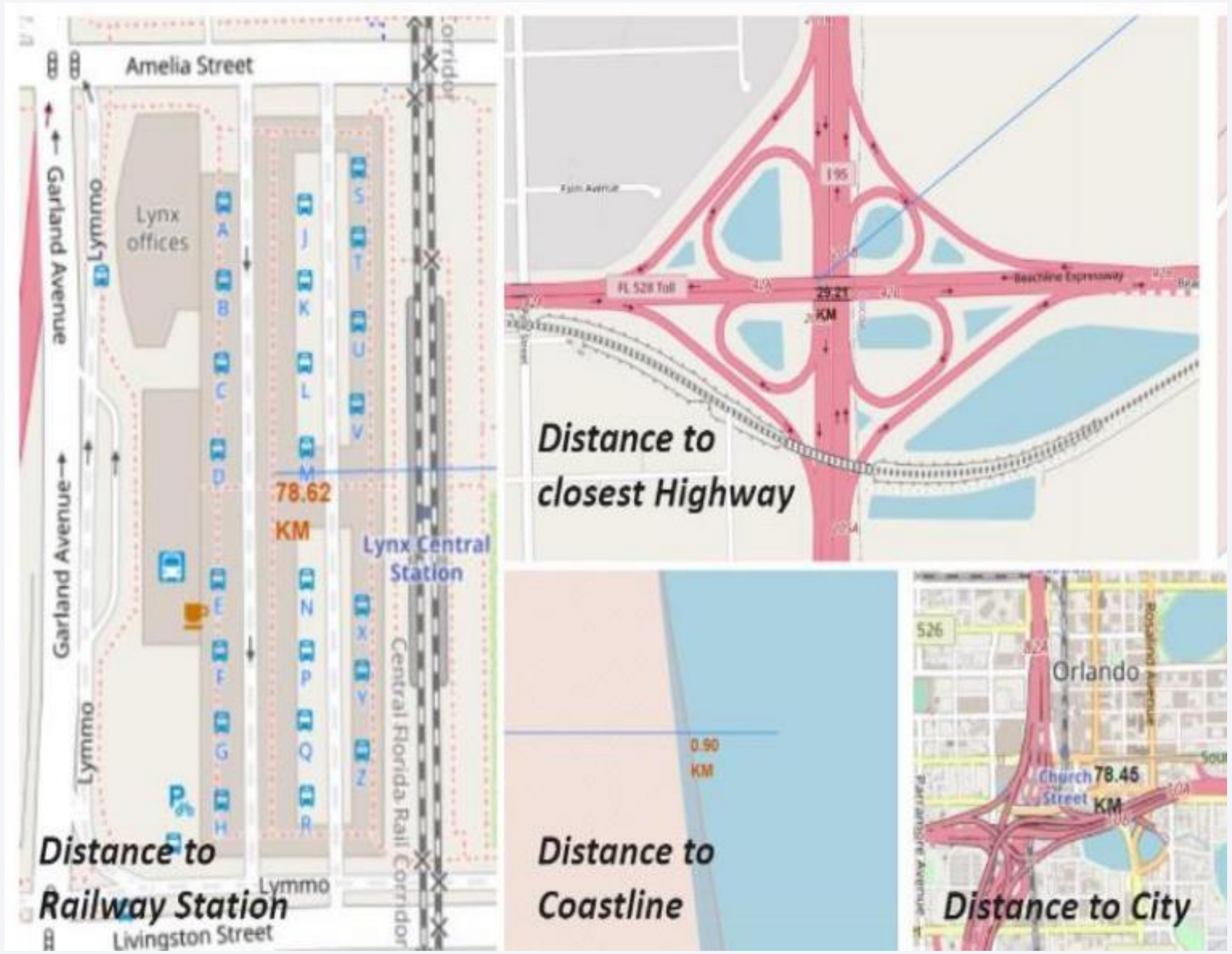


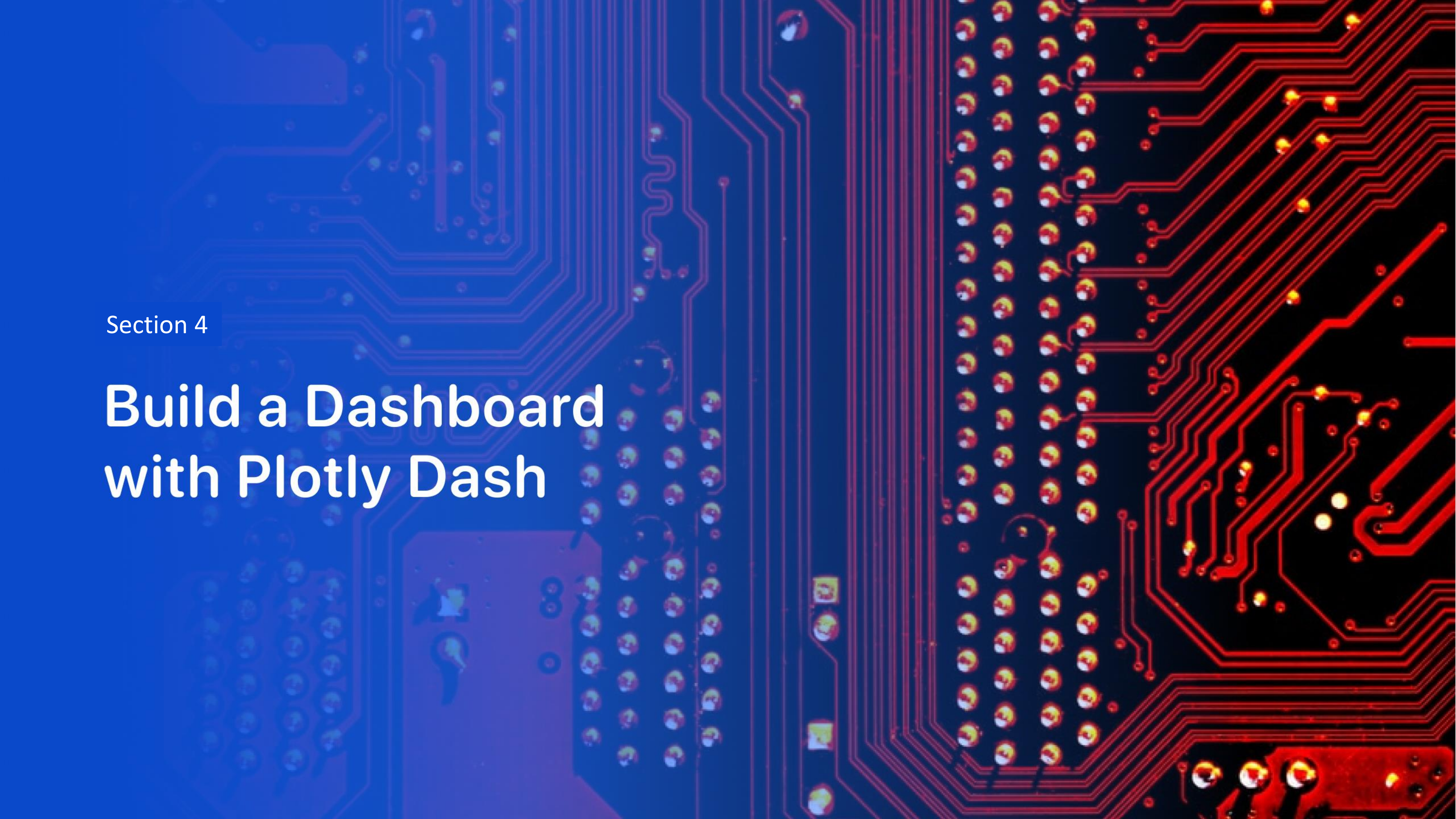
Markers showing launch sites with color labels





Launch site distance to landmarks



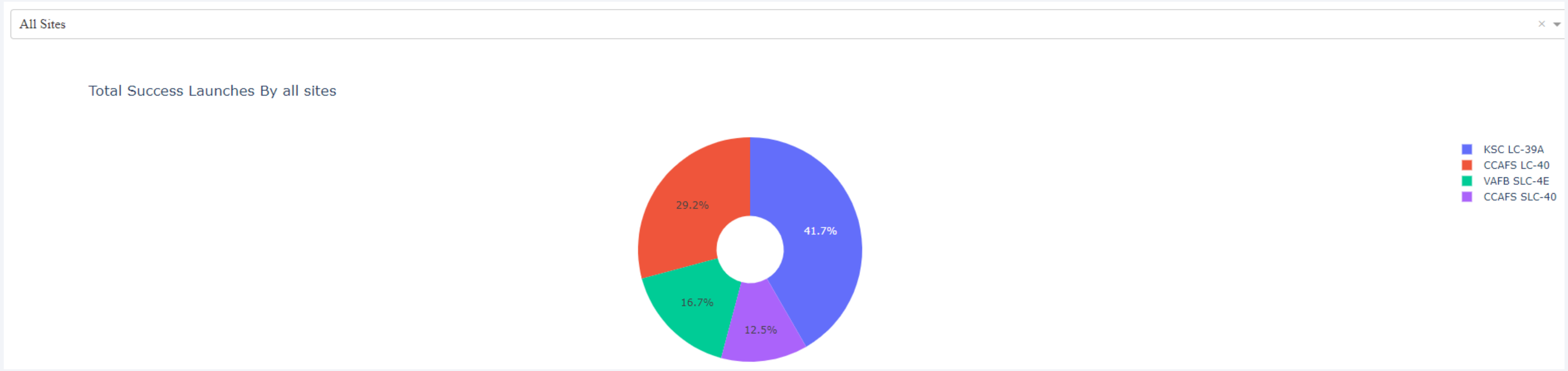


Section 4

Build a Dashboard with Plotly Dash

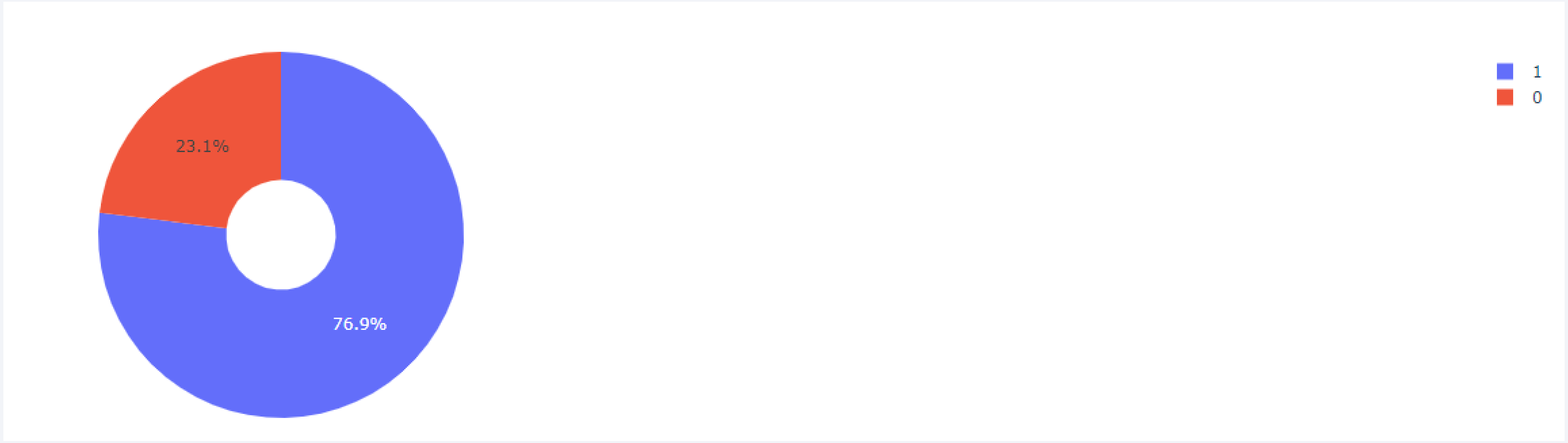
Pie chart showing the success percentage achieved by each launch site

- We can observe the at KSC LC-39A is most successful

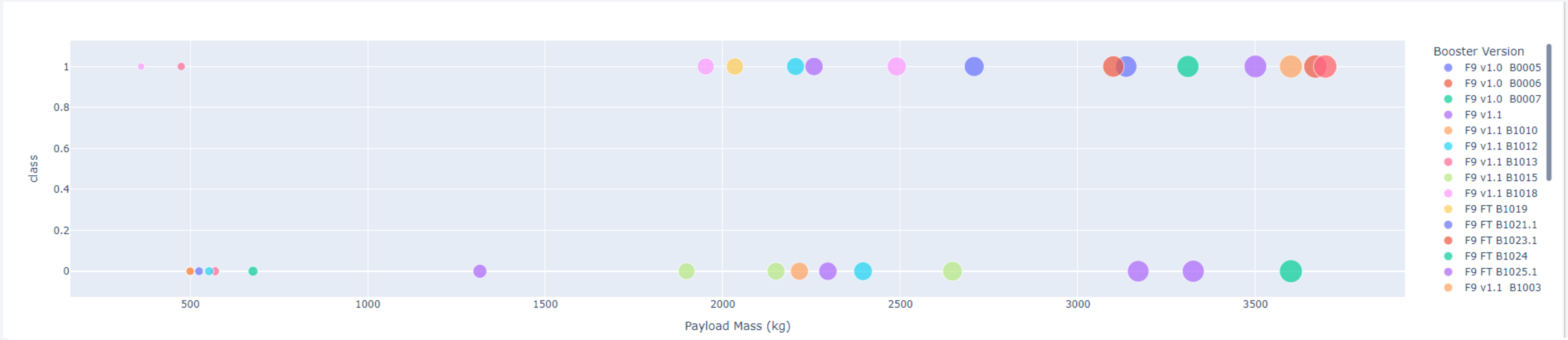


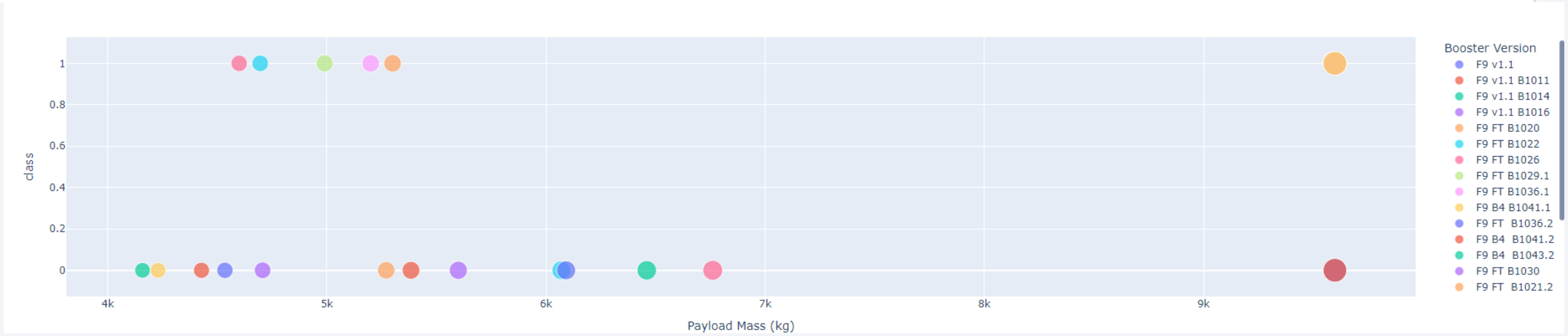
Pie chart showing the Launch site with the highest launch success ratio

- KSC LC-39A launch site have 76.9% success rate and 23.1% failure rate



Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider





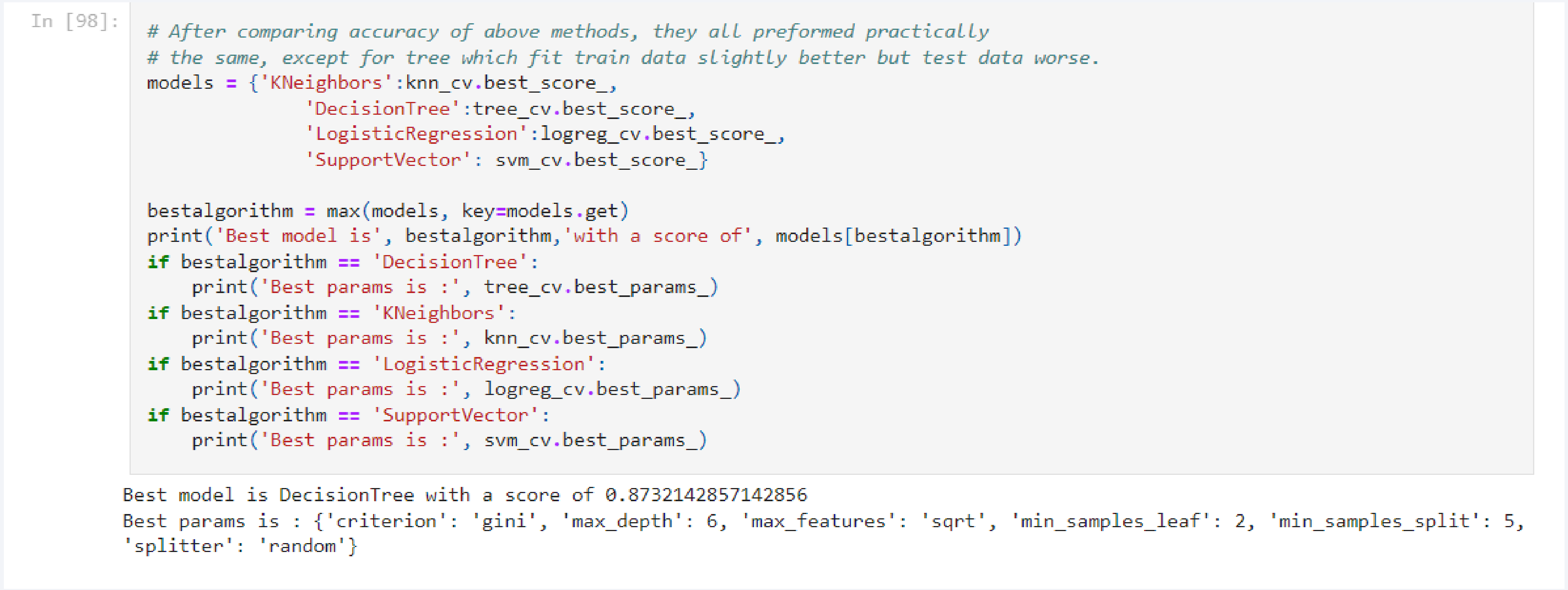
We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

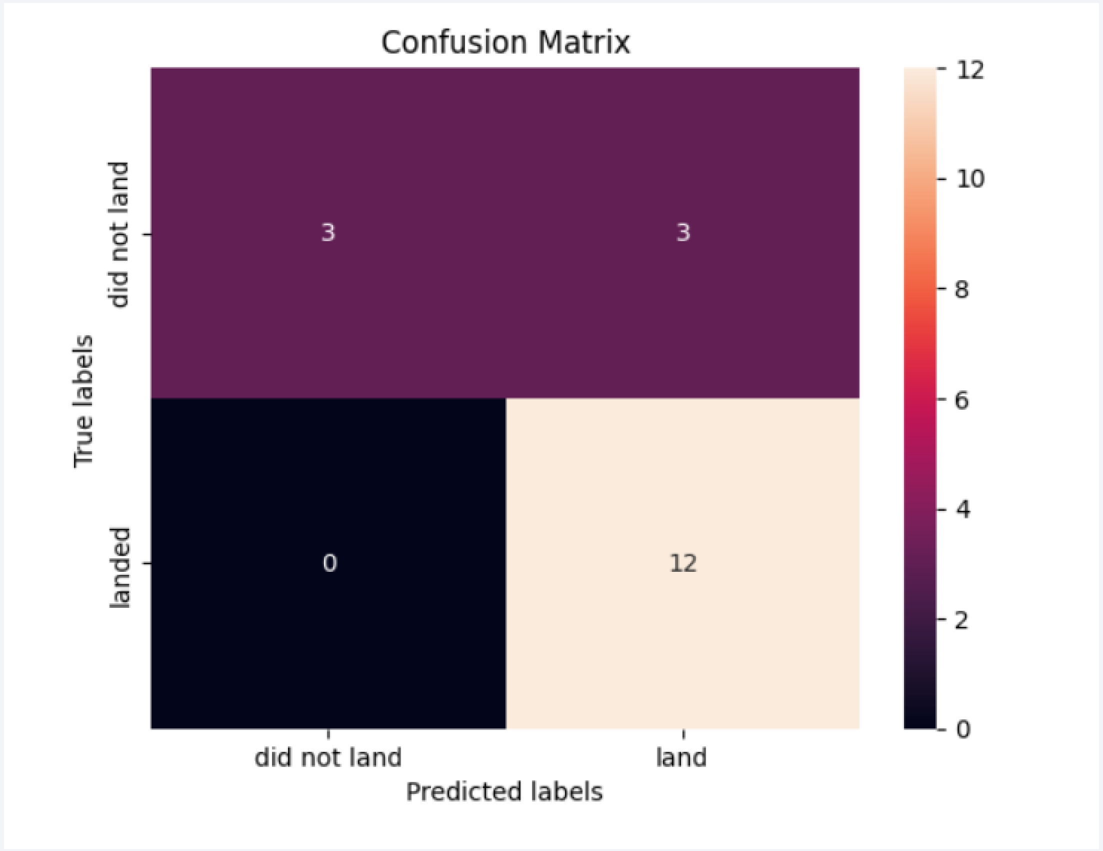
Decision Tree classifier model have high classification accuracy



```
In [98]: # After comparing accuracy of above methods, they all preformed practically
         # the same, except for tree which fit train data slightly better but test data worse.
         models = {'KNeighbors':knn_cv.best_score_,
                   'DecisionTree':tree_cv.best_score_,
                   'LogisticRegression':logreg_cv.best_score_,
                   'SupportVector': svm_cv.best_score_}

         bestalgorithm = max(models, key=models.get)
         print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
         if bestalgorithm == 'DecisionTree':
             print('Best params is :', tree_cv.best_params_)
         if bestalgorithm == 'KNeighbors':
             print('Best params is :', knn_cv.best_params_)
         if bestalgorithm == 'LogisticRegression':
             print('Best params is :', logreg_cv.best_params_)
         if bestalgorithm == 'SupportVector':
             print('Best params is :', svm_cv.best_params_)

Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5,
'splitter': 'random'}
```

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

We can conclude that:

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO 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

- 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!