IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Hugo Gomez

3/17/2023

Outline

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

Executive Summary

- We want to predict if the flacon 9 first stage will land successfully
  - If we can predict that a falcon 9 launch will be successful SpaceX will be able to reuse the launch and hence save money
  - This same information can be used to aid another company that would like to rival SpaceX launches economically
- Summary of all results
  - Found that there is a difference in success rate of landing based on location of launch
  - Some launch sites had higher success rates than others based on the payload
  - While Orbits like ES-L1, GEO, HEO, SSO, VLEO had higher end of the spectrum success rate ratio then other orbits
  - Orbit vs Flight number was not a linear relationship in the success rate of launches

Introduction

- Goal is to determine the cost of a launch
  - using the SpaceX's Flacon 9 launch information on Wikipedia
- To accomplish this goal, it is important that we first parse the information using Web Scraping, data wrangling and formatting, and requests to the SpaceX API
- After parsing the data in a way that we can use Exploratory Data Analysis and Visualization to grab data that would be used for machine learning predictions
- Folium is then used to display the different environment/locations where launches take place which help finding the most optimal location for a launch to take place
- A dashboard is created using Plotly Dash to be able to present the data in an interactive application where filtering and other features are available

Section 1

Methodology

Methodology

Executive Summary

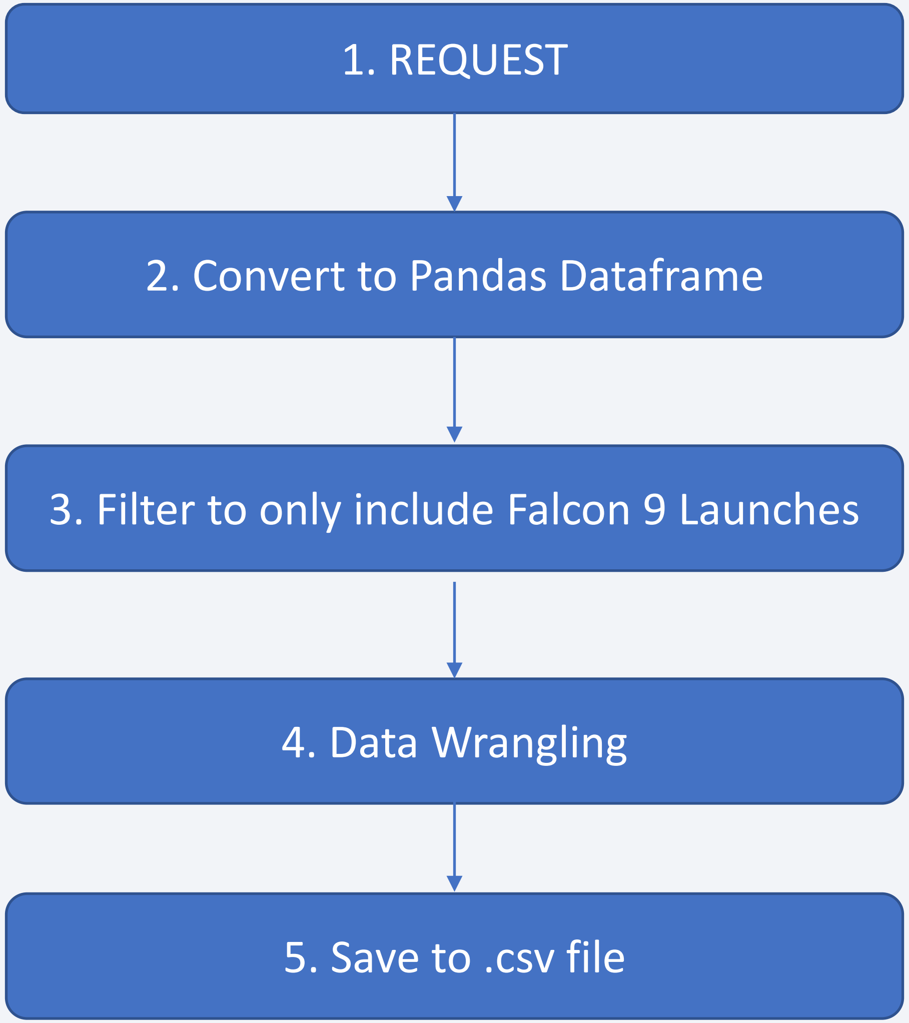
- Data collection methodology:
  - Request to the SpaceX API & Clean the Requested Data
- Perform data wrangling
  - Extract Falcon 9 launch record HTML table from Wikipedia
  - Parse the table and convert it to pandas data frame
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Find best Hyperparameter for SVM, Classification Trees and Logistic Regression

Data Collection

- Using Data Wrangling and formatting we are able to gain access to important data that would give us insight to predicting the success of a launch
- We request and parse the SpaceX launch data using the GET request to only include Falcon 9 launch information
- It is important to fill the missing data with that of the mean of the column data to get the most accurate results possible
- Data wrangling on the dataframe to get the most accurate information for future uses

Data Collection – SpaceX API

- Request to the SpaceX API
- Normalize the returned json content to convert to Pandas dataframe
- Obtain rocket, payload, launchpad, and many other useful information using API
- Store this data into lists to create readable dataframe
- Filter Falcon 9 launches only
- Data wrangle to remove missing values
- Github Link - SpaceX API



1. REQUEST

2. Convert to Pandas Dataframe

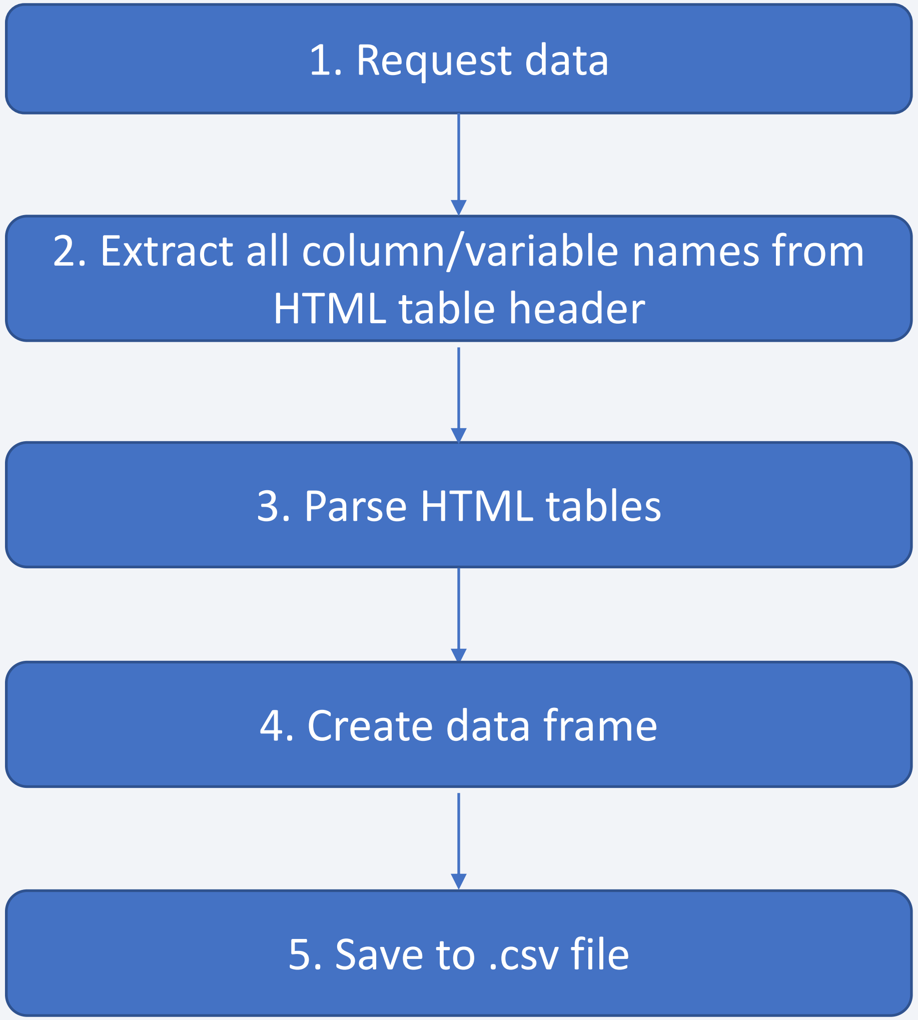
3. Filter to only include Falcon 9 Launches

4. Data Wrangling

5. Save to .csv file

Data Collection - Scraping

- Using HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response
- Create Beautiful Soup Object
- Extract all important information
- Parse HTML tables into dataframe
- Github Link - Scraping



1. Request data

2. Extract all column/variable names from HTML table header

3. Parse HTML tables

4. Create data frame

5. Save to .csv file

Data Wrangling

- Goal is to remove any missing values in order to ensure that the data is accurate
- Determine the proper data labels that would be used for training supervised models
- Calculate the number of launches on each site
- Calculate the number and occurrence of each orbit
- Calculate the number and occurrence of mission outcome per orbit
- Create a landing outcome label from Outcome column
- Github Link - Data Wrangling

EDA with Data Visualization

- Visualize the relationship between Flight Number and Launch Site with a scatter point chart
- Visualize the relationship between Payload and Launch Site with a scatter point chart
- Visualize the relationship between success rate of each orbit type with a bar chart
- Visualized the relationship between Flight Number and Orbit type with a scatter point chart
- Visualize the relationship between Payload and Orbit type
- Visualize the launch success yearly trend with a line chart
- Github Link - EDA with Data Visualization

EDA with SQL

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first 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
- 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 Link - EDA with SQL

Build an Interactive Map with Folium

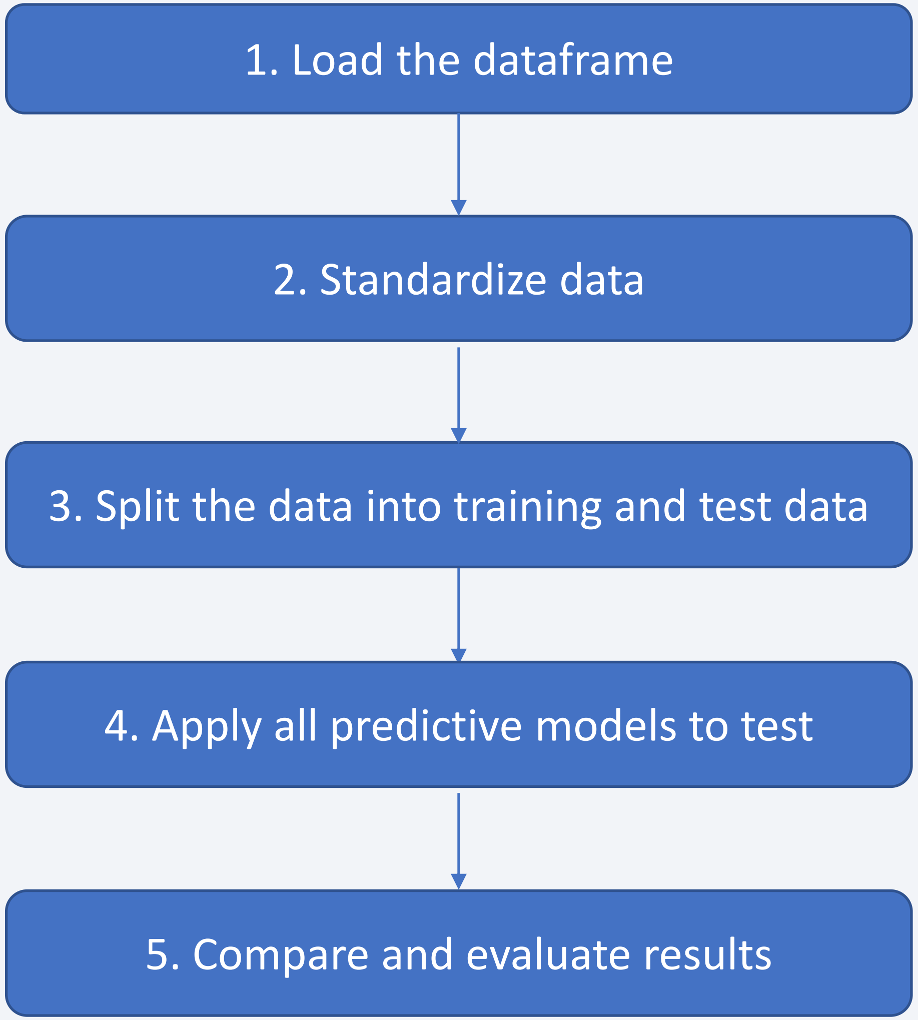
- Mark all launch sites on a map
- Mark the success/failed launches for each site on the map
- Calculate the distances between a launch site to its proximities
- All of these geographical patterns contribute to finding the most optimal launching site
- Github Link - Interactive Map with Folium

Build a Dashboard with Plotly Dash

- Add a Launch Site Drop-down Input Component
  - Allow use to select from multiple launch sites
- Add a callback function to render success-pie-chart based on selected site dropdown
  - Enable the dashboard to change to the pie chart to the currently selected launch site
- Add a Range Slider to Select Payload
  - Include a range of payload to view the scatter plot at the selected range
- Add a callback function to render the success-payload-scatter-chart scatter plot
  - Attach the range slider to function properly with the success-payload-scatter-chart scatter plot
- Github Link - Dashboard with Plotly Dash

Predictive Analysis (Classification)

- Create a column for the class
- Standardize the data
- Split into training and test data
- Use the different predictive models to test and then compare their results
- Github Link - Predict Analysis (Classification)



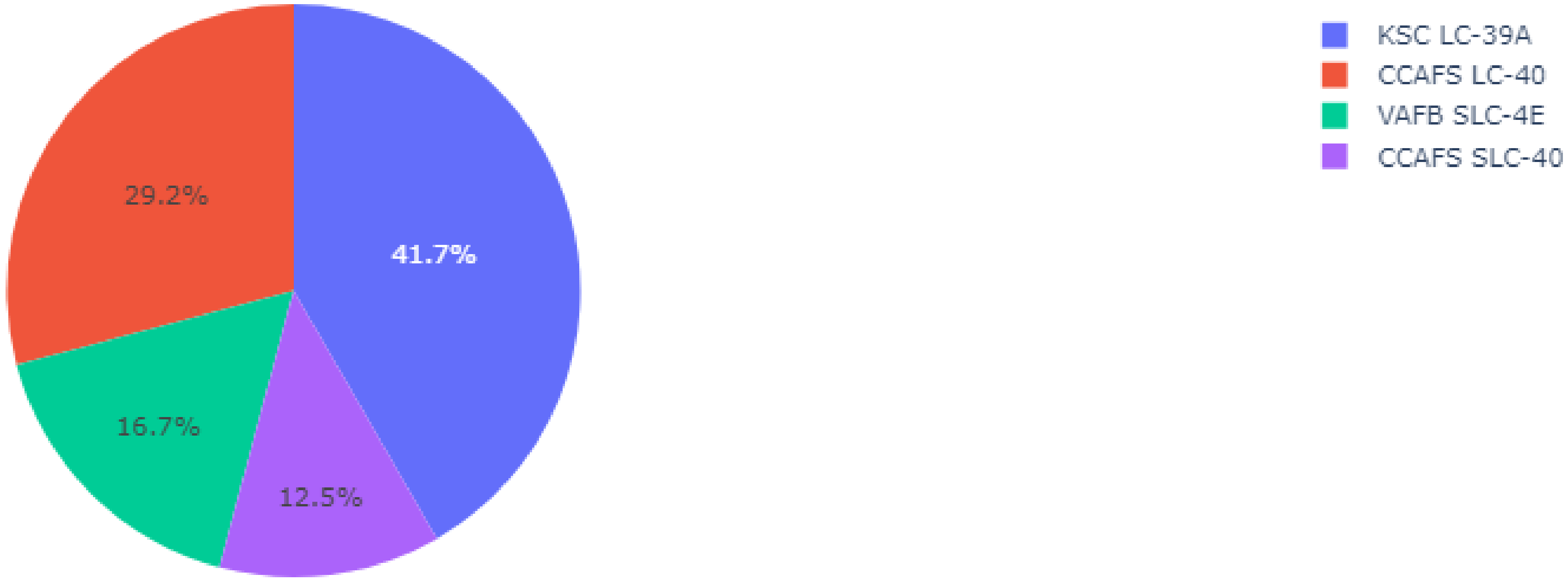
1. Load the dataframe

2. Standardize data

3. Split the data into training and test data

4. Apply all predictive models to test

5. Compare and evaluate results



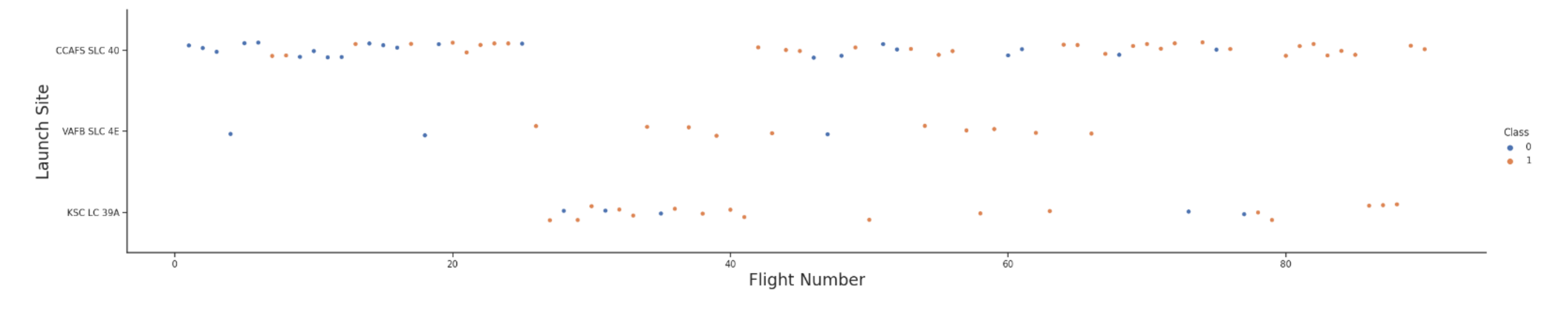
Results

- Exploratory data analysis results
  - We see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%. s
  - Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).
- Predictive analysis results
  - From the different models used for predictions of launches the most accurate was the Decision Tree with 88% accuracy compared to the others which had 83% accuracy with the given parameters

Section 2

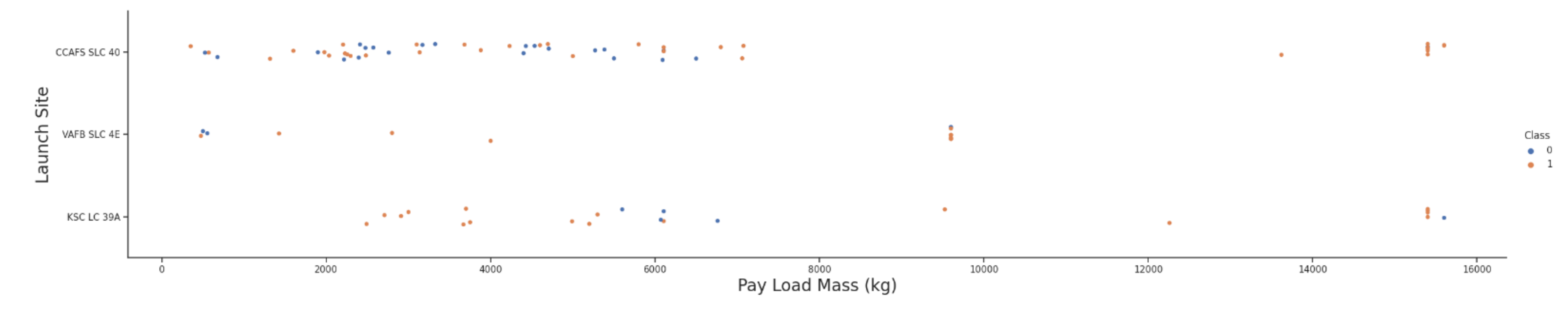
Insights drawn from EDA

Flight Number vs. Launch Site



- This is a scatter plot of Flight Number vs. Launch Site
- From this screenshot we know that from the increasing number of flights there is an increase in the number of successful launches on the first stage which would influence SpaceX to do more launches

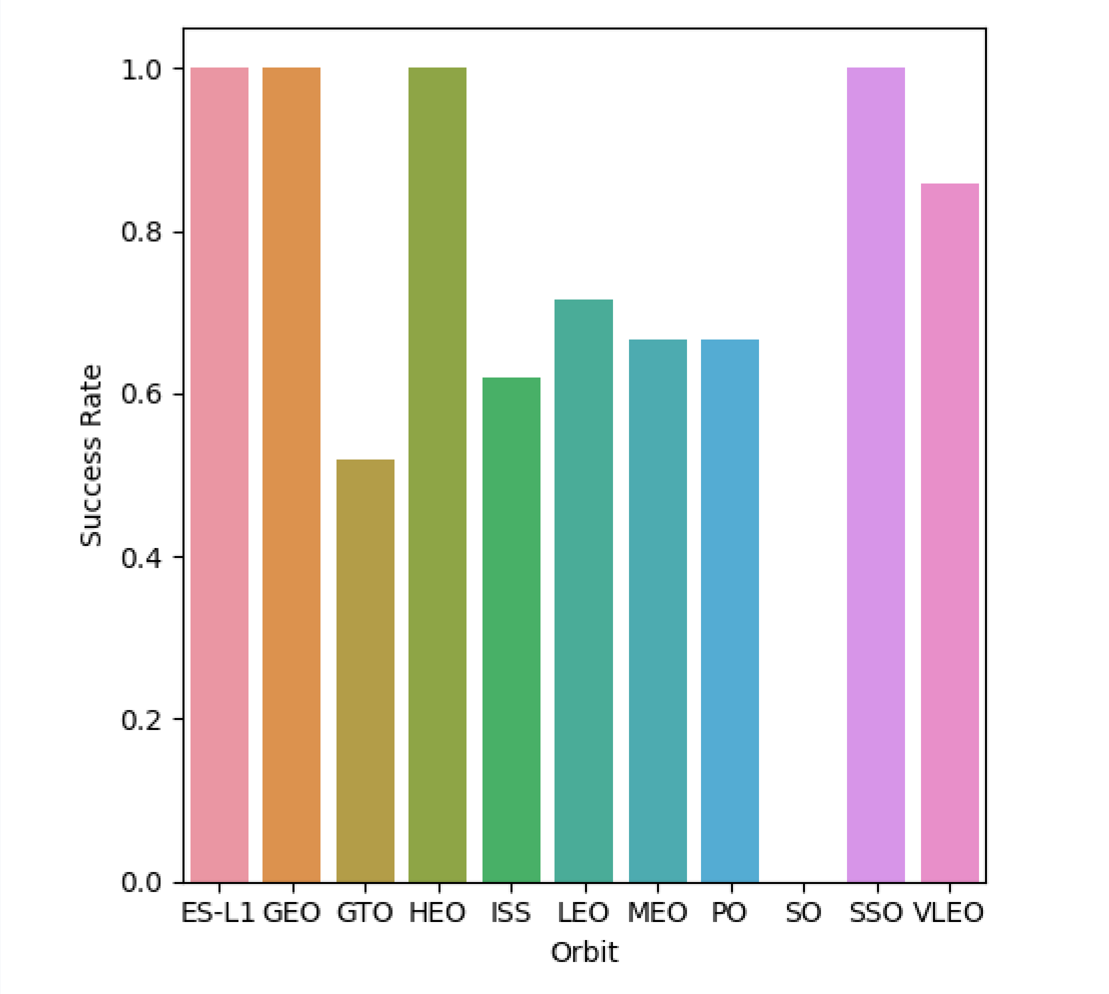
Payload vs. Launch Site



- Scatter plot of Payload vs. Launch Site
- The CCAFS SLC 40 launch site as well as the KSC LC 39A both reached max payload mass with a high success rate
- CCAFS SLC 40 Launch site experienced most all of its failure before the 80,000 pay load mass mark
- VAFB SLC 4E did not have many launches in general in comparison with the other two sites

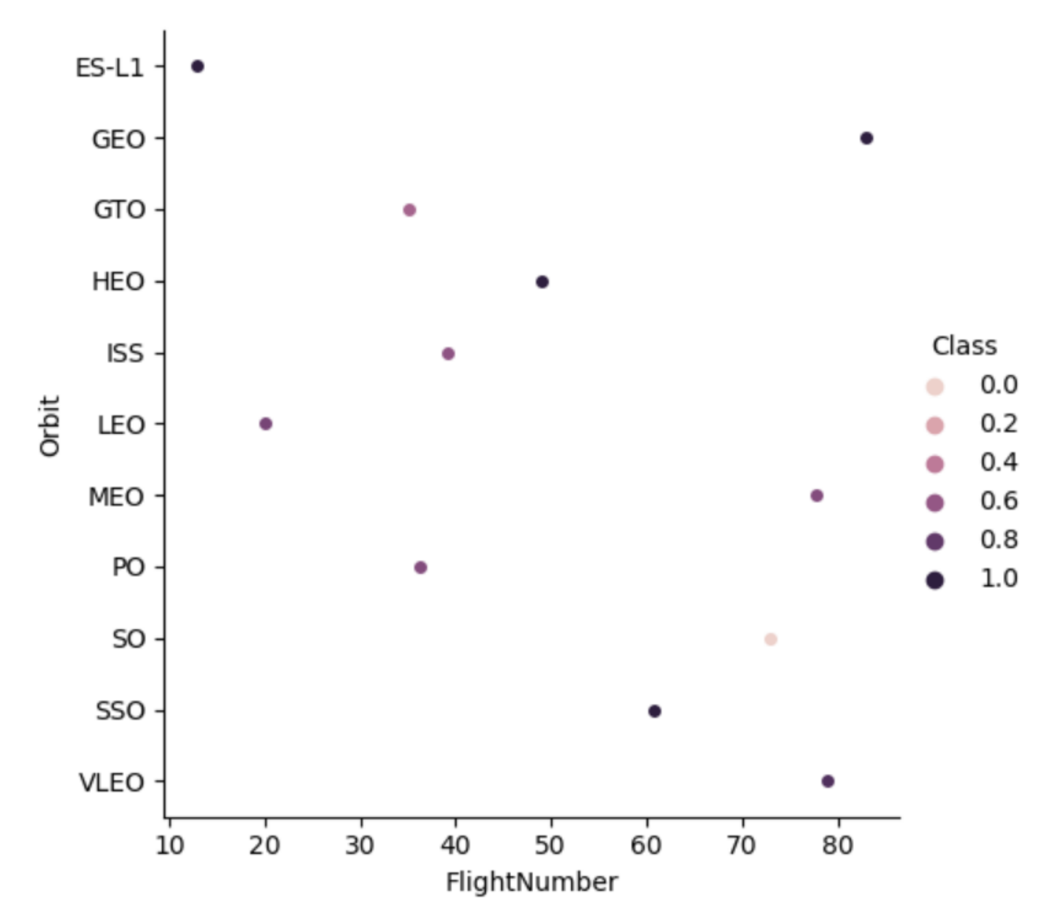
Success Rate vs. Orbit Type

- In the plot to the right, we can see that Orbits like Sun-synchronous orbit have a 0% success rate
- While Orbits like ES-L1, GEO, HEO, SSO, VLEO have all above 80% success rate with launches



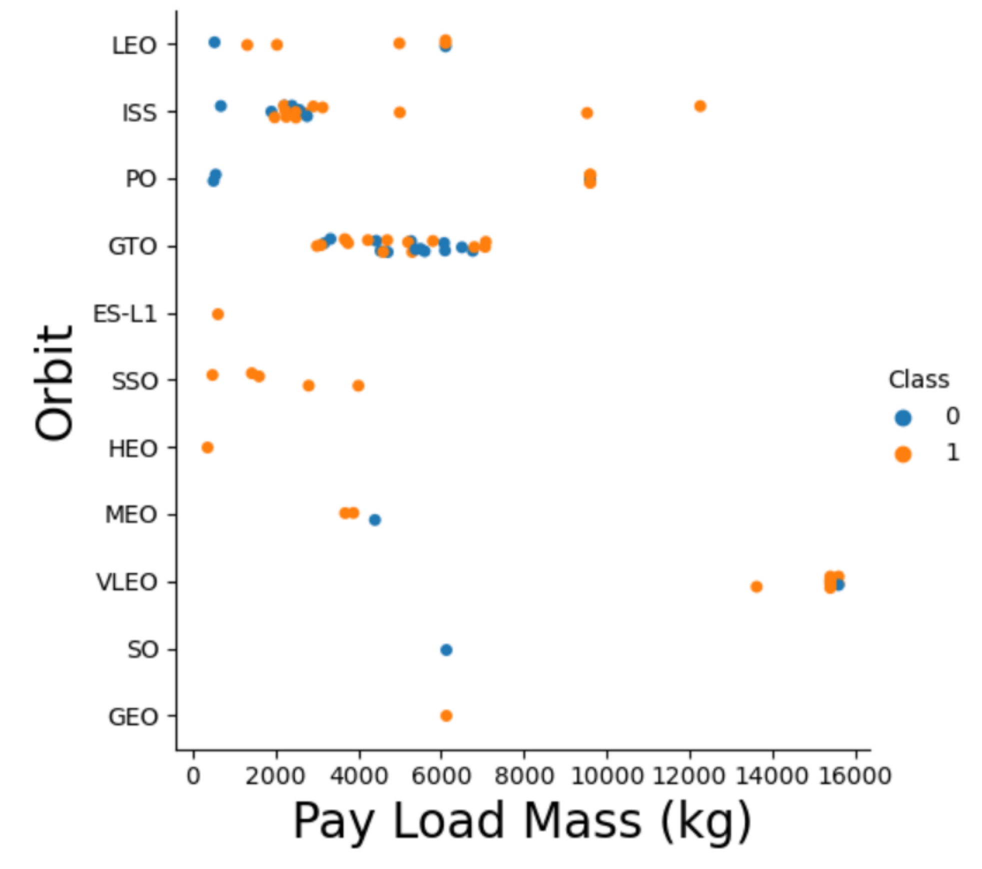
Flight Number vs. Orbit Type

- In this graph we also notice that orbits with higher flights does not guarantee high success but that the orbits plays a role in success rate
- ES-L1 with the lowest flight number has a high success rate due to orbit playing a role in success rate



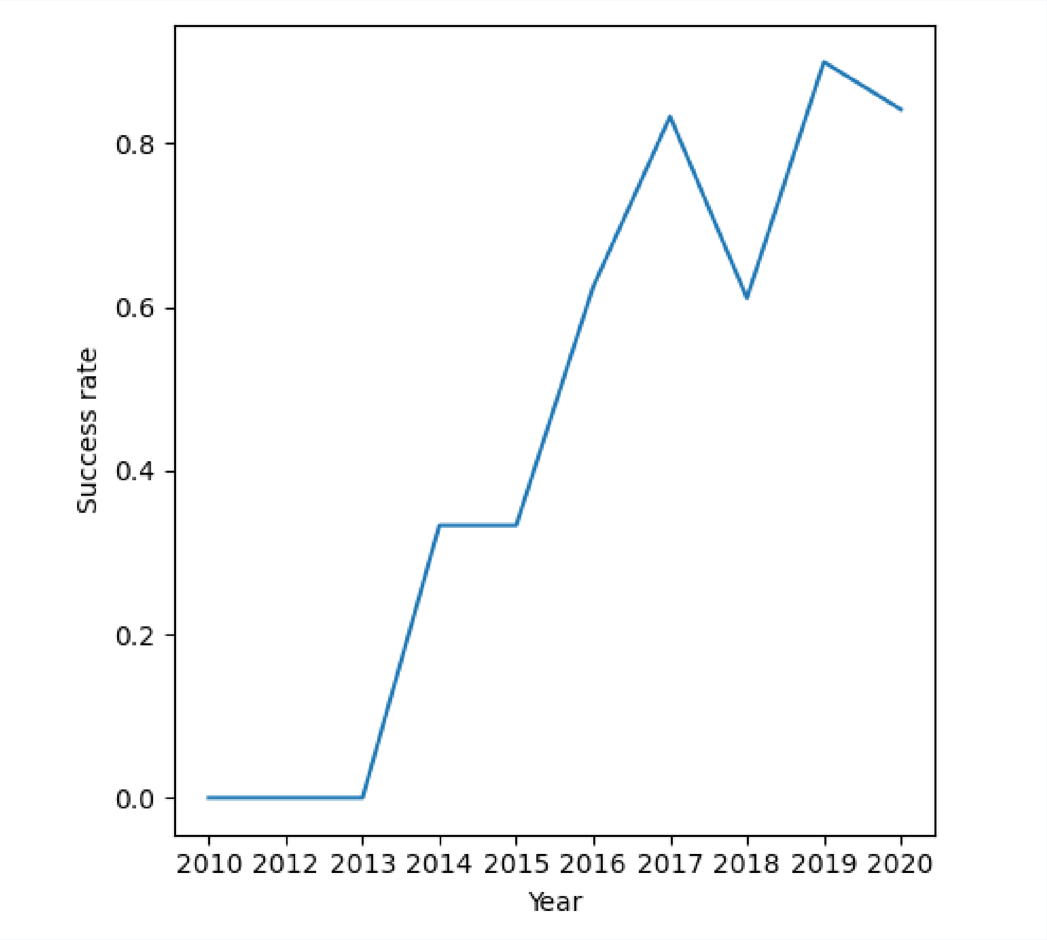
Payload vs. Orbit Type

- In the plot we notice that regarding pay load mass and success rate ES-L1, SSO, HLEO have high success rates
- Other orbits like GTO and ISS are not necessarily at a high success rate of launches
- Note that most launches are done with less than 80,000 pay load mass



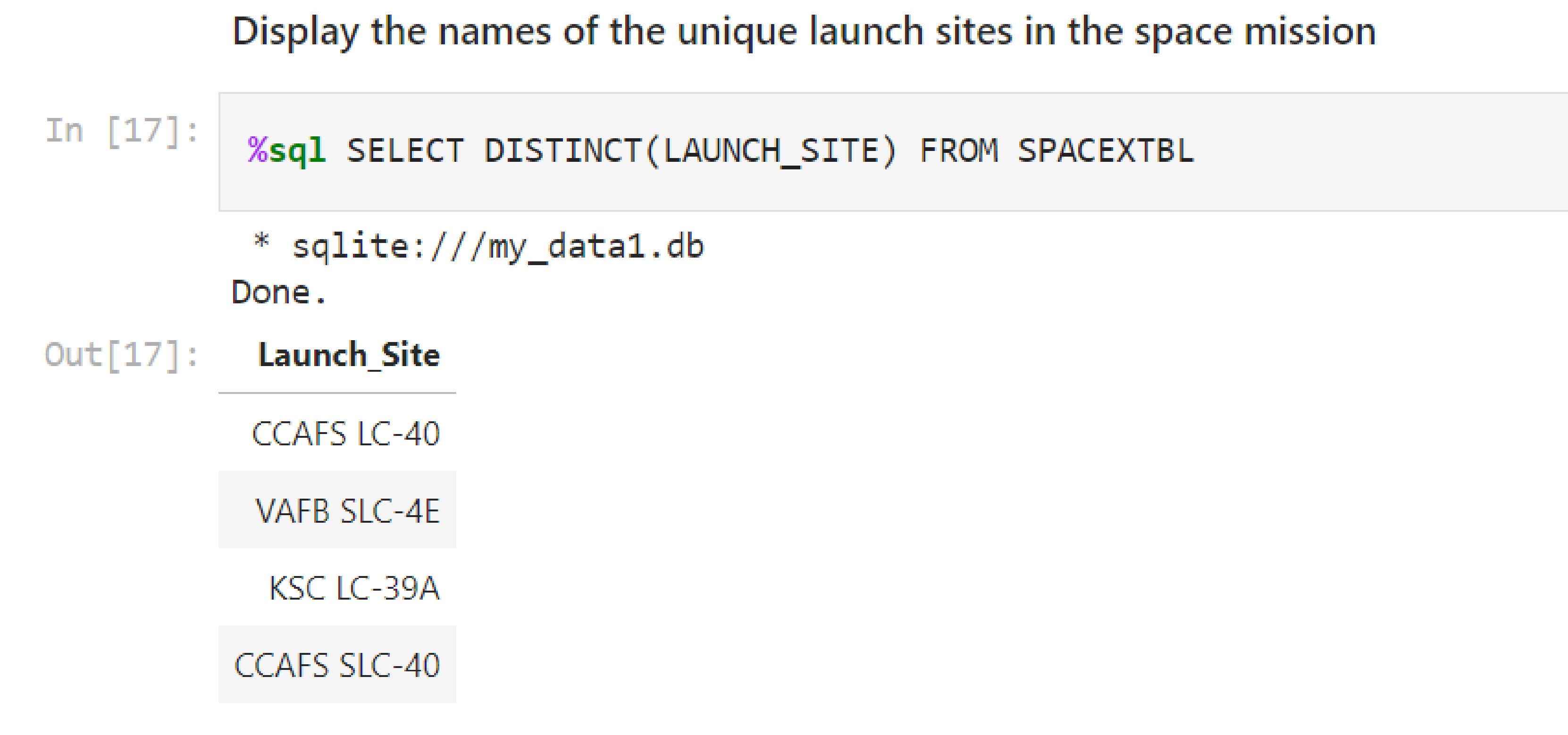
Launch Success Yearly Trend

- There is a great increase of launching success rate over the years
- This plot is telling of SpaceX ability to adapt and improve on their launches which in the long run will be more economically efficient



All Launch Site Names

- Using DISTINCT function and Launch_site as the column in the SPACEXTBL we can obtain the unique launch sites



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

In [17]:

```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL
```

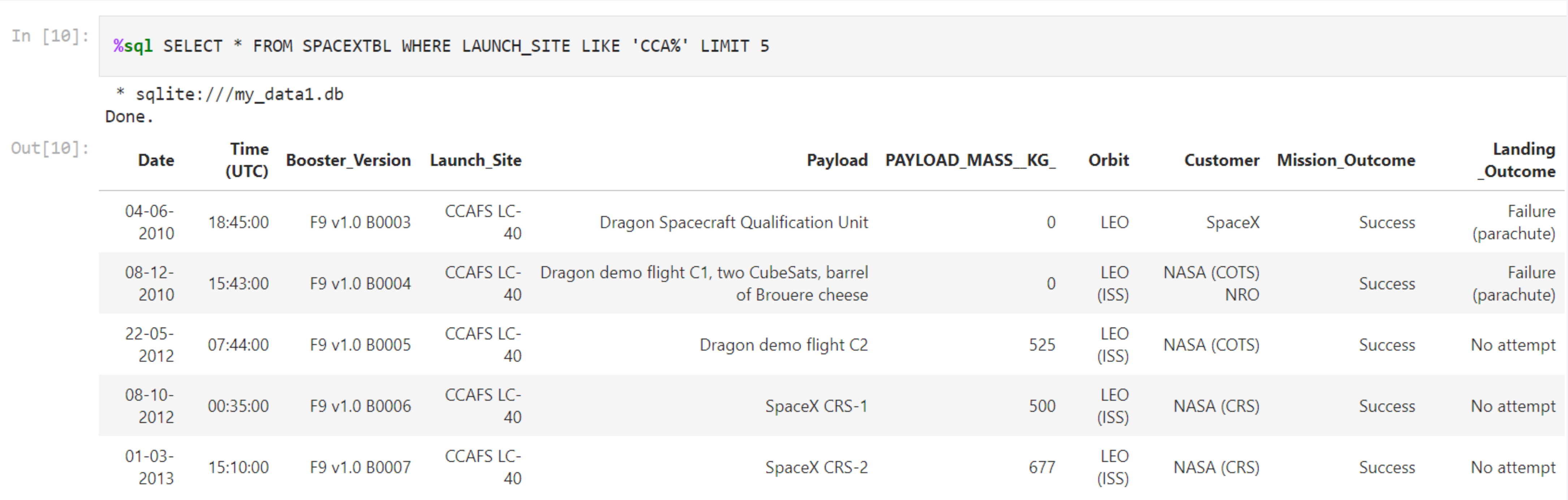
```
 * sqlite:///my_data1.db
Done.
```

Out[17]:

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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
  - Using the * to select all from the SPACEXTBL we use condition where Launch_site is like ‘CCA%’ with the beginning and limit to 5 results



```
In [10]:  %sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

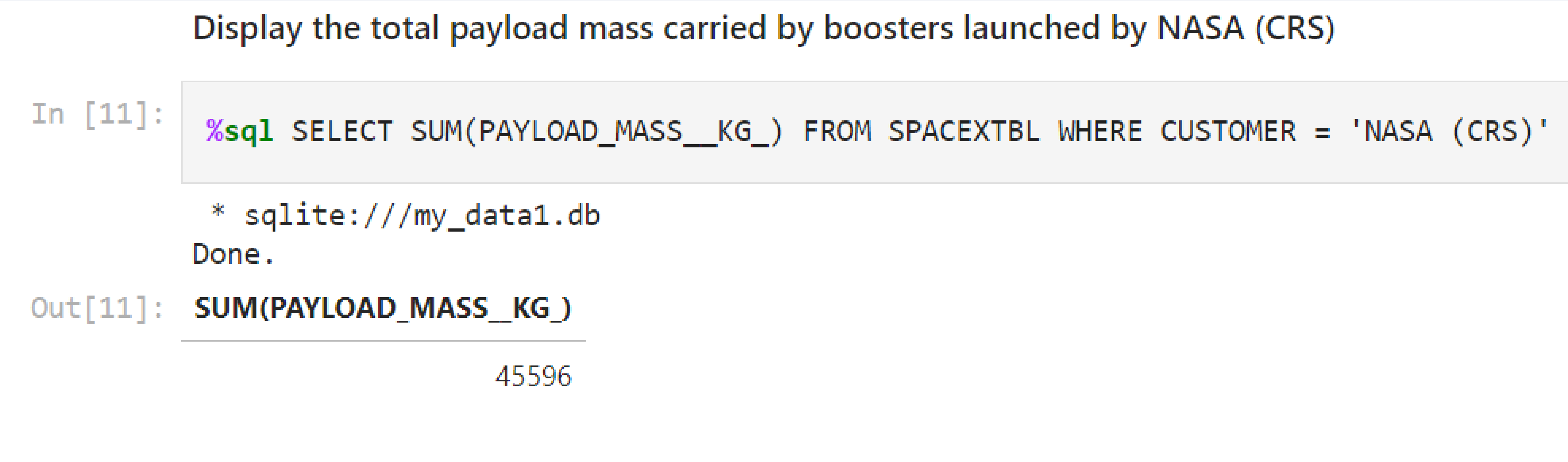
```
 * sqlite:///my_data1.db
Done.
```

Out[10]:

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 SUM function of Paylod_Mass_Kg from SPACEXTBL we use condition where Customer = ‘NASA CRS’ to obtain the total payload mass carried by boosters launched by NASA (CRS)



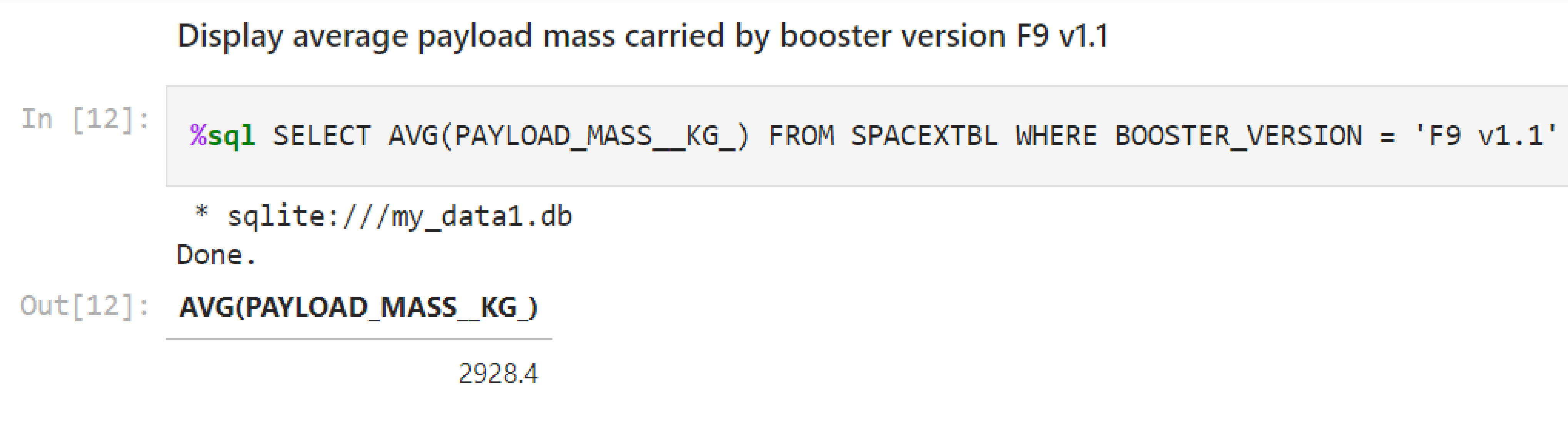
Display the total payload mass carried by boosters launched by NASA (CRS)

```
In [11]: %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)'
```

```
 * sqlite:///my_data1.db
Done.
Out[11]: SUM(PAYLOAD_MASS__KG_)
                         45596
```

Average Payload Mass by F9 v1.1

- Using the AVG function for Payload_Mass_Kg_ from the SPACEXTBL we use condition where Booster_version = ‘F9 v1.1’ to obtain the average payload mass carried by booster version F9 v1.1



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

```
In [12]:  %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1'
```

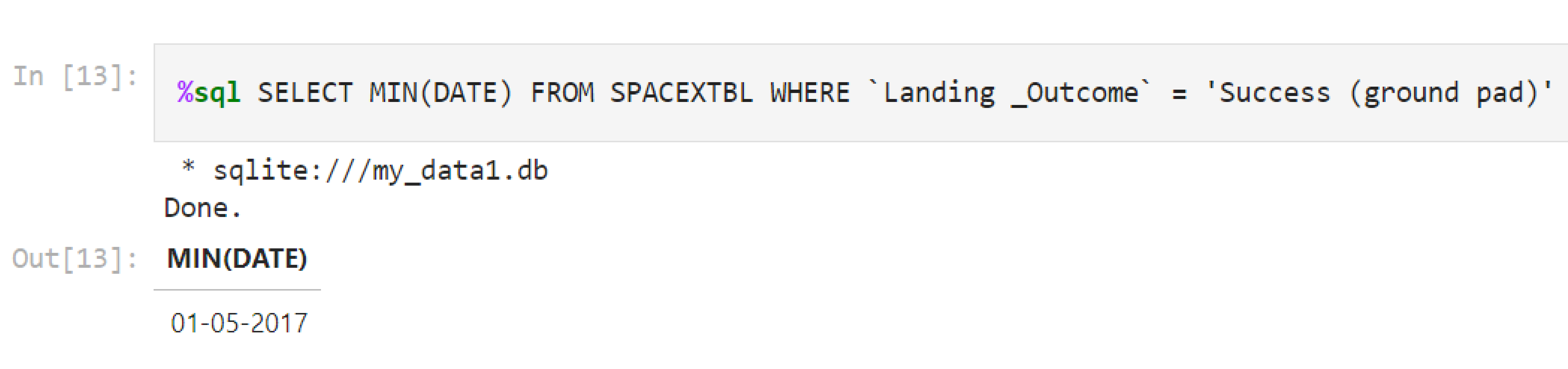
```
 * sqlite:///my_data1.db
Done.
```

Out[12]:

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- Goal is to: Find the dates of the first successful landing outcome on ground pad
- To do this using the Min function for Date from SPACEXTBL we use condition where ‘Landing_Outcome’ = ‘Success (ground_pad)’



```
In [13]:  %sql SELECT MIN(DATE) FROM SPACEXTBL WHERE `Landing _Outcome` = 'Success (ground pad)'
```

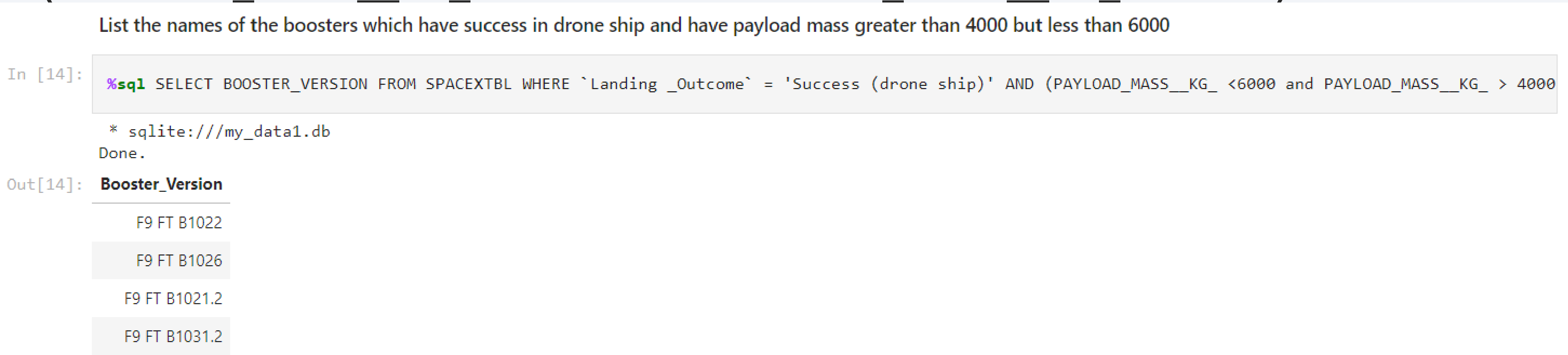
```
 * sqlite:///my_data1.db
Done.
```

Out[13]:

MIN(DATE)
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

- First we select Booster_version from SPACEXTBL and now list the conditions to be met
- The conditions are ‘Landing_Outcome’ = ‘Success (drone ship)’ and (PAYLOAD_MASS__KG_ <6000 and PAYLOAD_MASS__KG_ > 4000)



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

```
In [14]: %sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE `Landing _Outcome` = 'Success (drone ship)' AND (PAYLOAD_MASS__KG_ <6000 and PAYLOAD_MASS__KG_ > 4000
```

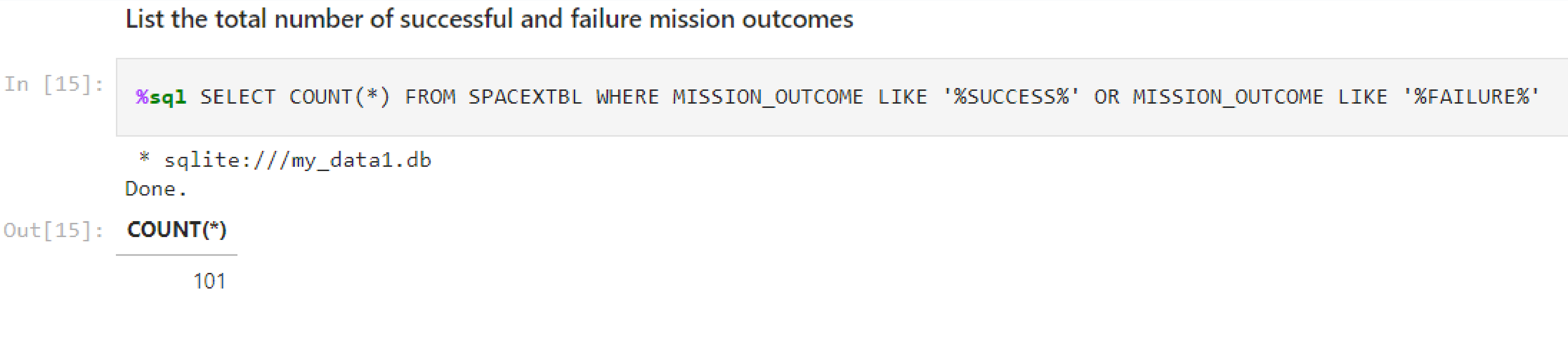
```
 * sqlite:///my_data1.db
Done.
```

Out[14]:

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

Total Number of Successful and Failure Mission Outcomes

- Using Count function for the SPACEXTBL we then use where clause to meet the condition of ‘%Success%’ or ‘%Failure%’ for Mission_Outcome using wild card %



List the total number of successful and failure mission outcomes

In [15]:

```
%sql SELECT COUNT(*) FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE '%SUCCESS%' OR MISSION_OUTCOME LIKE '%FAILURE%'
```

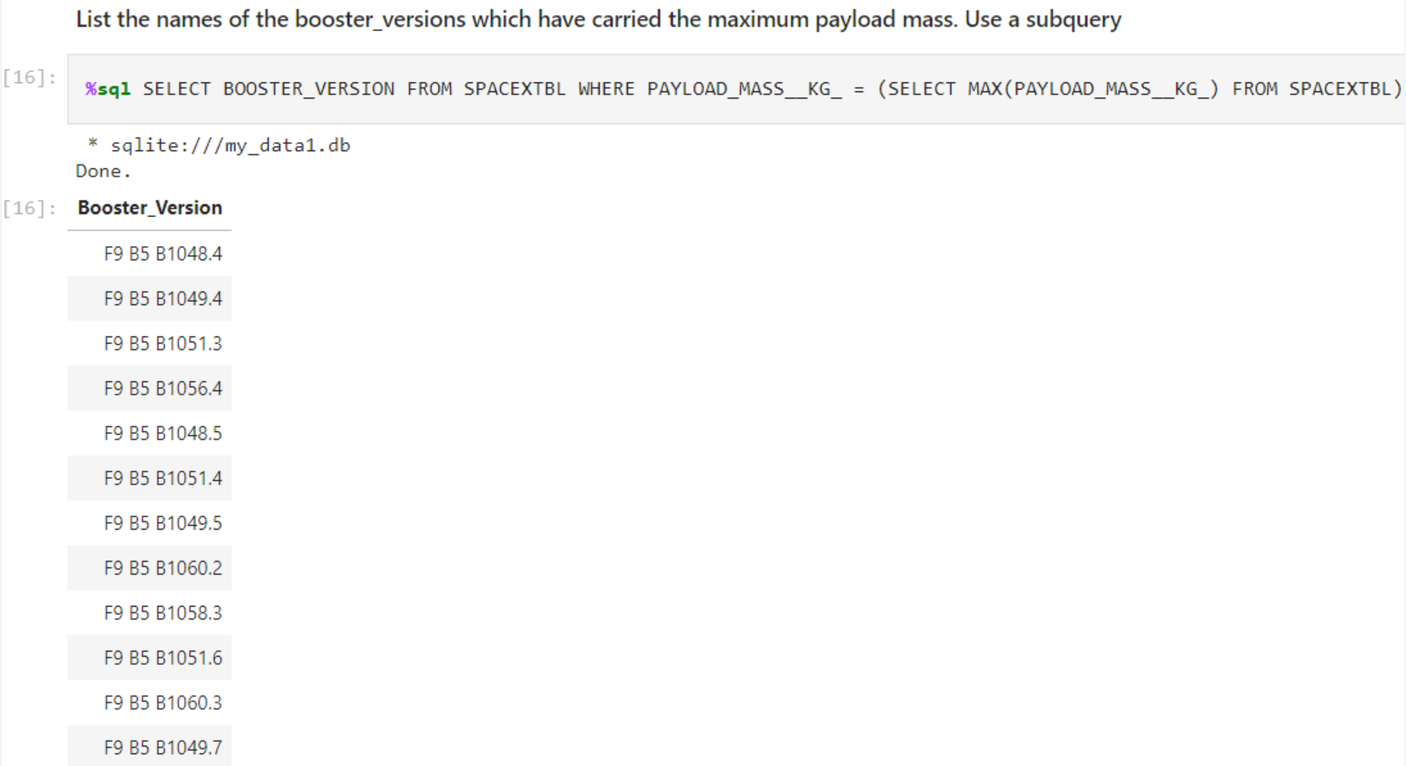
```
 * sqlite:///my_data1.db
Done.
```

Out[15]:

COUNT(*)
101

Boosters Carried Maximum Payload

- Using column Booster_version we use the SpaceXTBL and use the where clause to find the max payload mass with use of subqueries



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

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

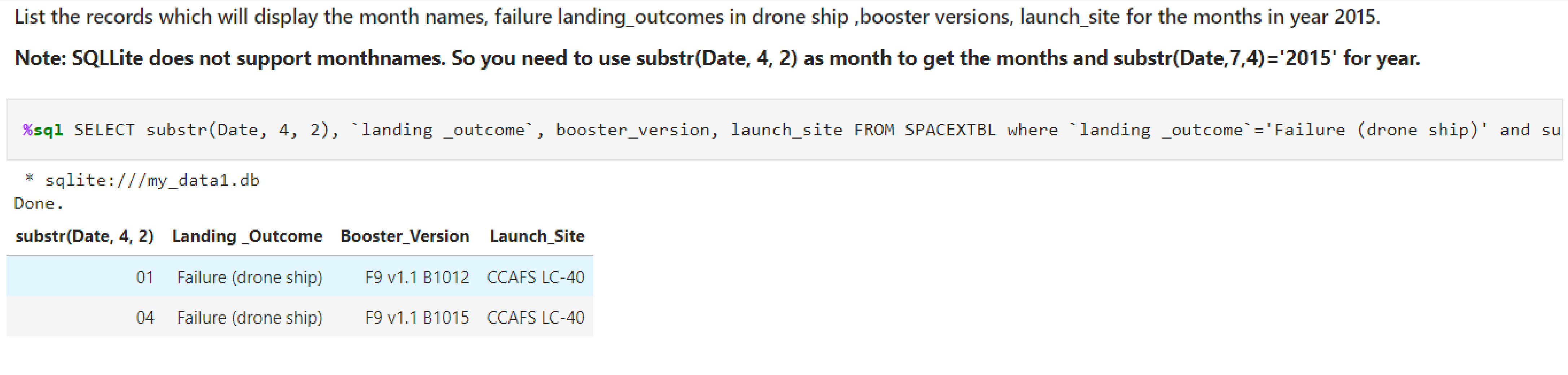
```
 * sqlite:///my_data1.db
Done.
```

[16]:

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

- With substr function and column names it is easy to select the columns we want
- Next, we list the outcomes to find the time we can to use with failures associated



List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

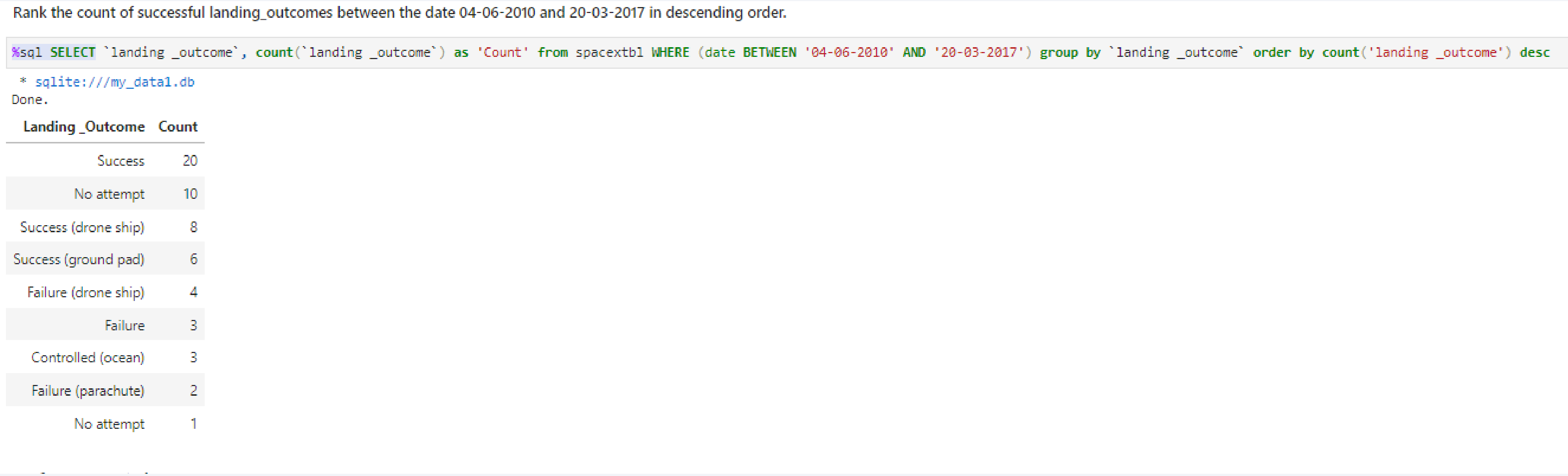
```
%sql SELECT substr(Date, 4, 2), `landing _outcome`, booster_version, launch_site FROM SPACEXTBL where `landing _outcome`='Failure (drone ship)' and su
```

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

substr(Date, 4, 2)	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

- Using Group by on landing outcome column allows for aggregation function count to function properly which is followed with a condition to choose dates between 2010-06-04 and 2017-03-20 in descending order



Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT `landing _outcome`, count(`landing _outcome`) as 'Count' from spacextbl WHERE (date BETWEEN '04-06-2010' AND '20-03-2017') group by `landing _outcome` order by count('landing _outcome') desc
```

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

Landing _Outcome	Count
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

Section 3

Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

- Replace <Folium map screenshot 1> title with an appropriate title

- Explore the generated folium map and make a proper screenshot to include all launch sites’ location markers on a global map

- Explain the important elements and findings on the screenshot

<Folium Map Screenshot 2>

- Replace <Folium map screenshot 2> title with an appropriate title

- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map

- Explain the important elements and findings on the screenshot

<Folium Map Screenshot 3>

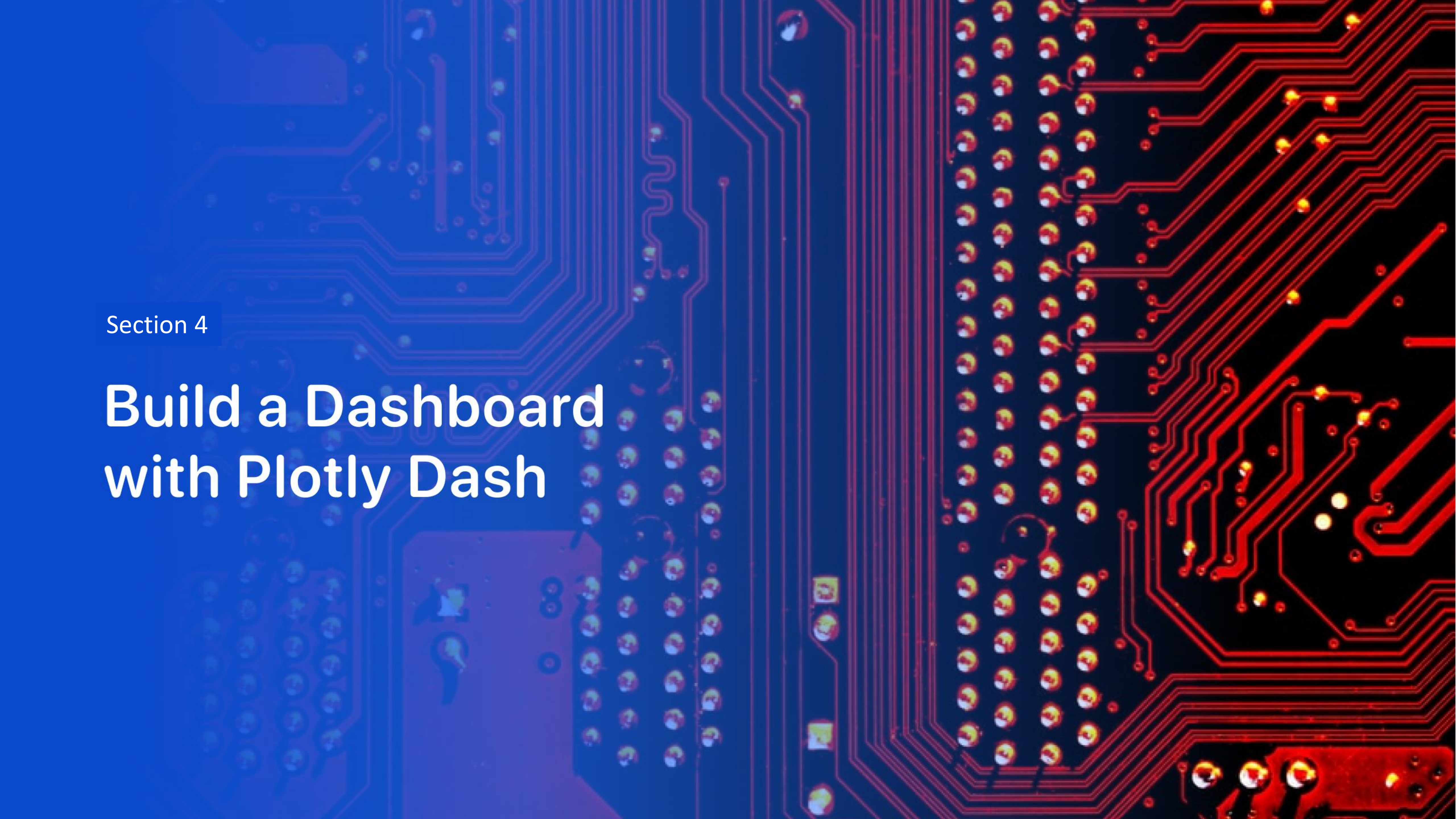
- Replace <Folium map screenshot 3> title with an appropriate title

- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed

- Explain the important elements and findings on the screenshot

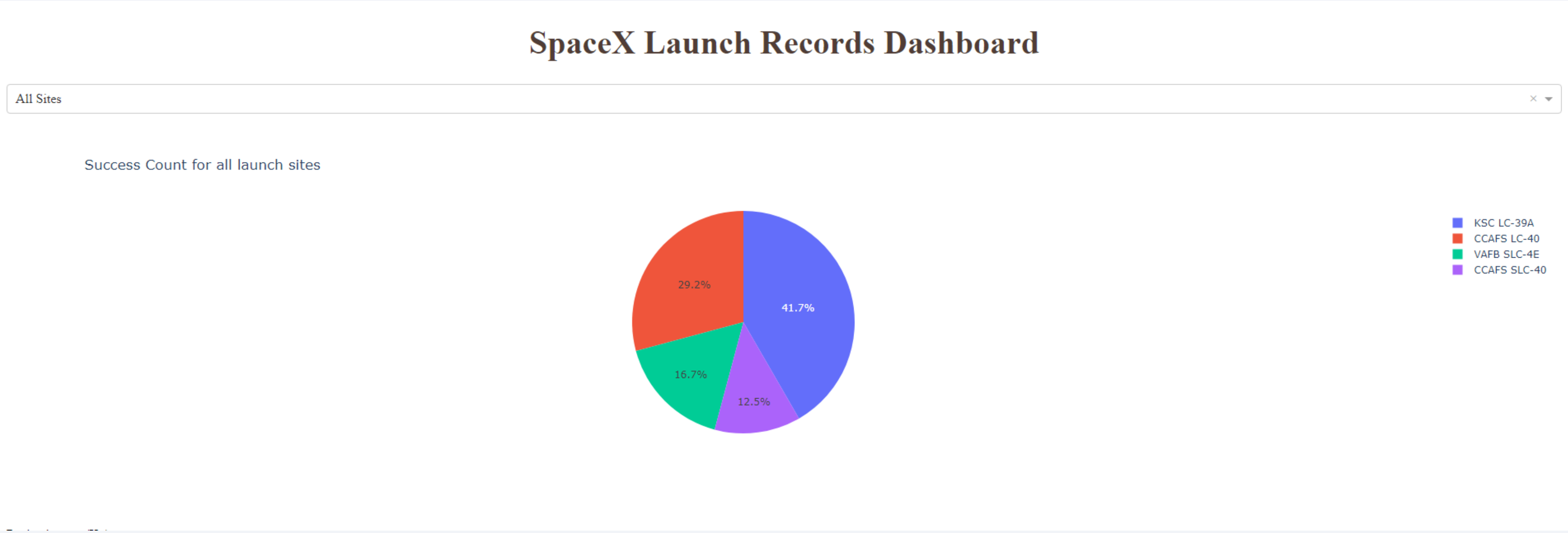
Section 4

Build a Dashboard with Plotly Dash



SpaceX Launch Records of All Sites Dashboard

- Below is a screenshot of the pie chart using Plotly Dash
- This pie chart represents the success rates of all sites where we can see that KSC LC-39A launch site had the highest success rate



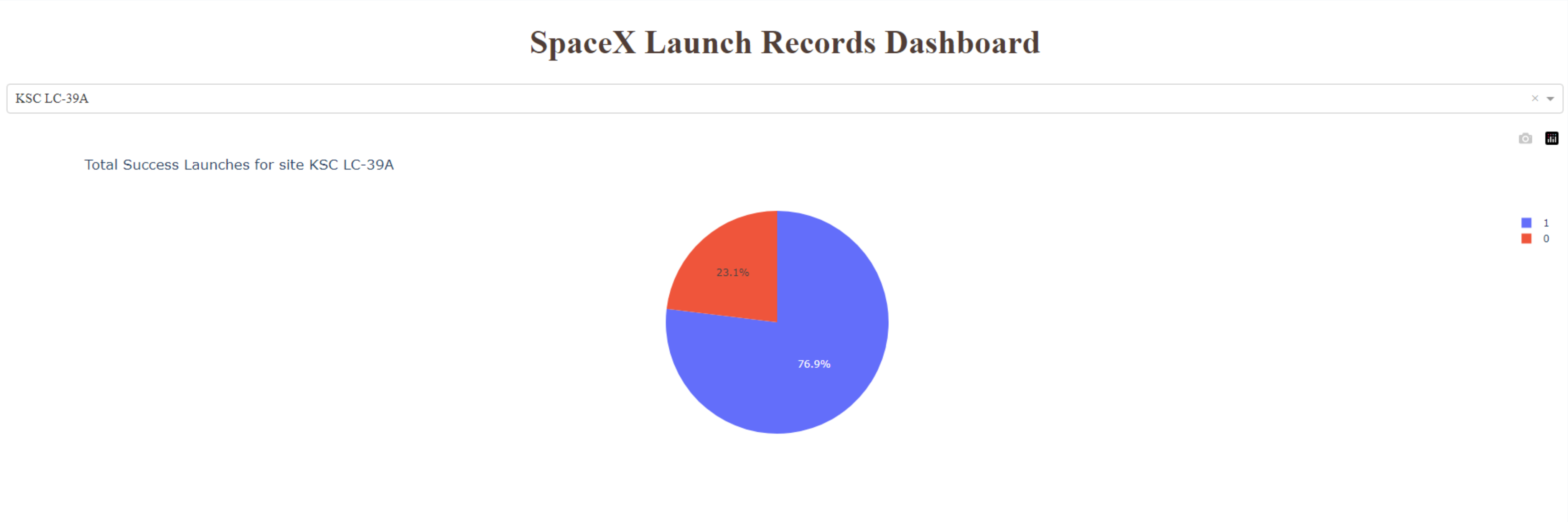
SpaceX Launch Records Dashboard

All Sites

Success Count for all launch sites

SpaceX Launch Records of Highest Launch Success Ratio

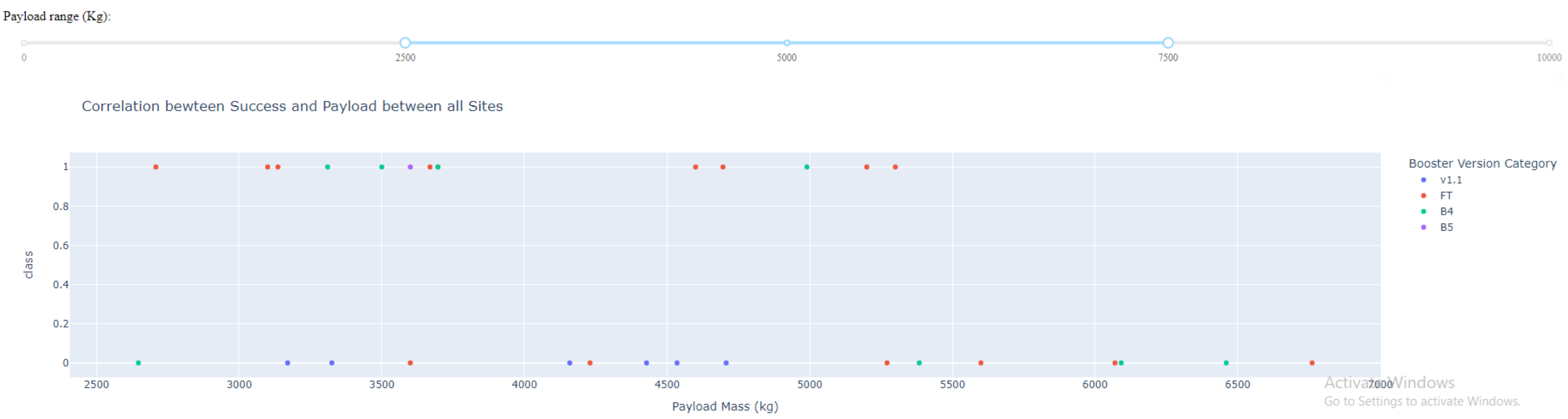
- In the below pie chart KSC LC-39A has the highest launch success ratio
- The legend on the right of the pie chart indicates 0 as failed launch and 1 as successful launch



SpaceX Launch Records Dashboard

KSC LC-39A

Total Success Launches for site KSC LC-39A



Payload vs. Launch Outcome Scatter Plot for All Sites

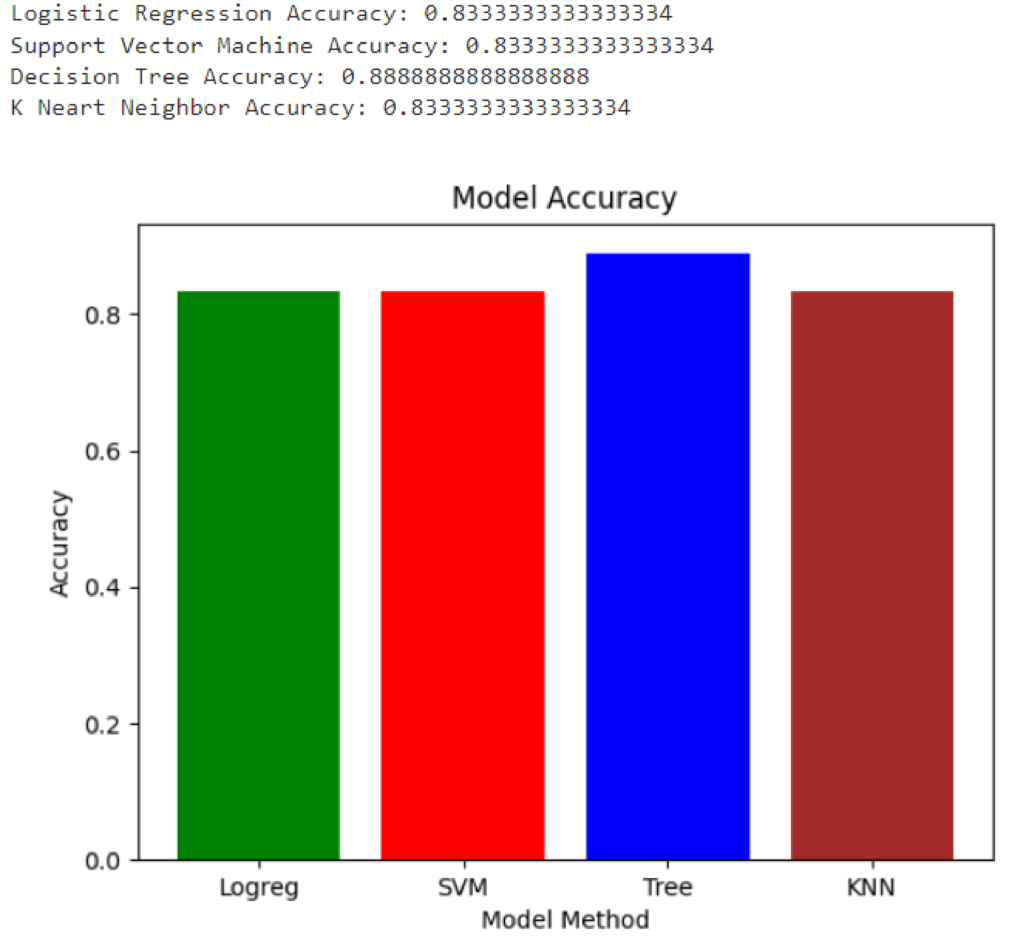
- Using a different payload range from 2500 – 7500 we can see that FT booster version category is more successful than any other booster version
- This as well as the v1.1 booster fails in with a success rate of 0

Section 5

Predictive Analysis (Classification)

Classification Accuracy

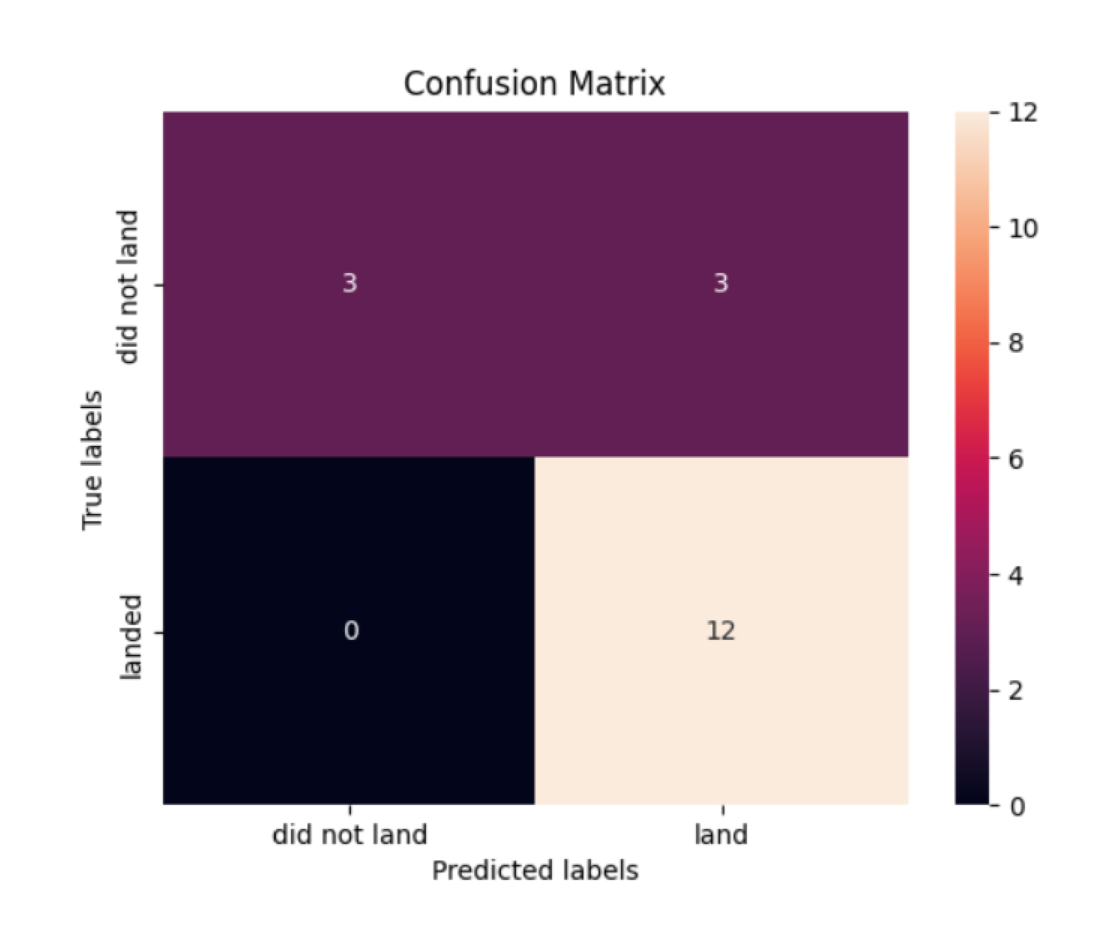
- Using all the different model methods they all generally had the same accuracy
- Running the program multiple times, the Decision Tree modeling method slightly did better than the other modeling methods in terms of accuracy



```
Logistic Regression Accuracy: 0.8333333333333334
Support Vector Machine Accuracy: 0.8333333333333334
Decision Tree Accuracy: 0.8888888888888888
K Neart Neighbor Accuracy: 0.8333333333333334
```

Confusion Matrix – Decision Tree

- To the right there is a confusion matrix to the decision tree method
- Looking at the confusion matrix it is clear that the predicted outcome labels were for the most part most like that of the actual labels



Conclusions

- Point 1
- Point 2
- Point 3
- Point 4
- …

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!