IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Daniel Gordon
01.07.2022

# Outline

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

# Executive Summary

- Summary of methodologies
- Summary of all results

# Introduction

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

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - The data was collected from both Wikipedia tables and from SpaceX official site. One can find the data in the following links:
    - Wikipedia -https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
    - SpaceX - https://api.spacexdata.com/v4/launches/past
- Perform data wrangling
  - We then create useable data frame with the relevant information to work with using Python libraries.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Using SQL in Jupiter Lab environment we established connection between several features to successes.

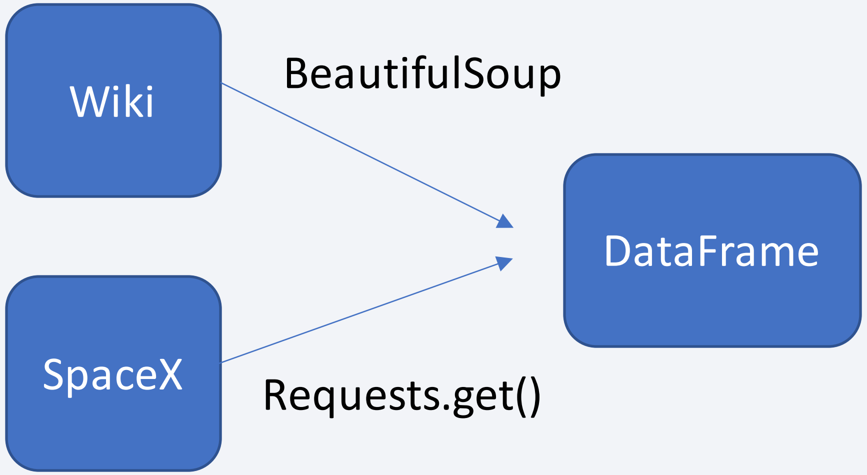
# Methodology

## Executive Summary

- Perform interactive visual analytics using Folium and Plotly Dash
  - Using Folium we give the launch site location and provide a dashboard as can be found her
- Perform predictive analysis using classification models
  - Finally, we generate a classification model using decision tree to predict mission succusses based on first stage launch with accuracy of 83% for unseen cases.

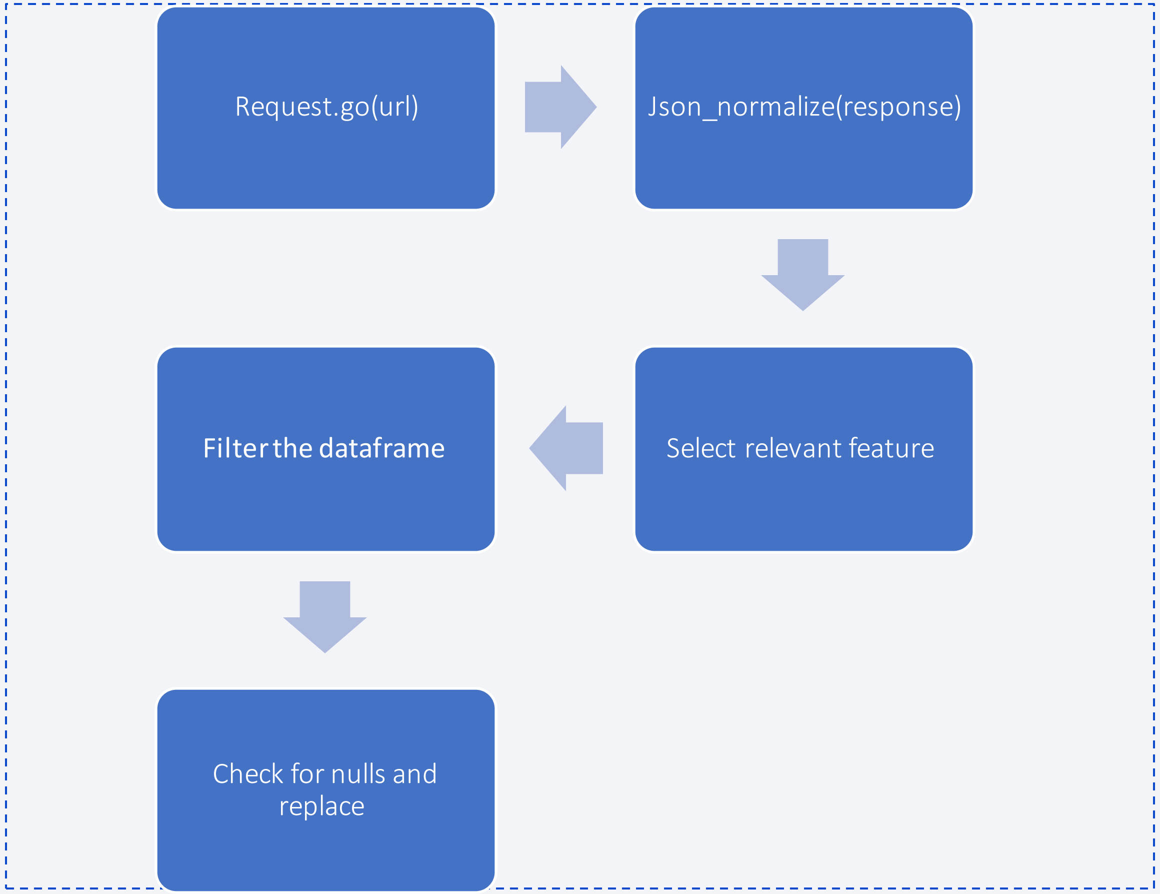
# Data Collection

- We collect the data from both Wikipedia and SpaceX API
- For the first we use BeautifulSoup methods on html to extract the relevant information from pre chosen table and generate a data frame.
- For the second, we used an API get requests.



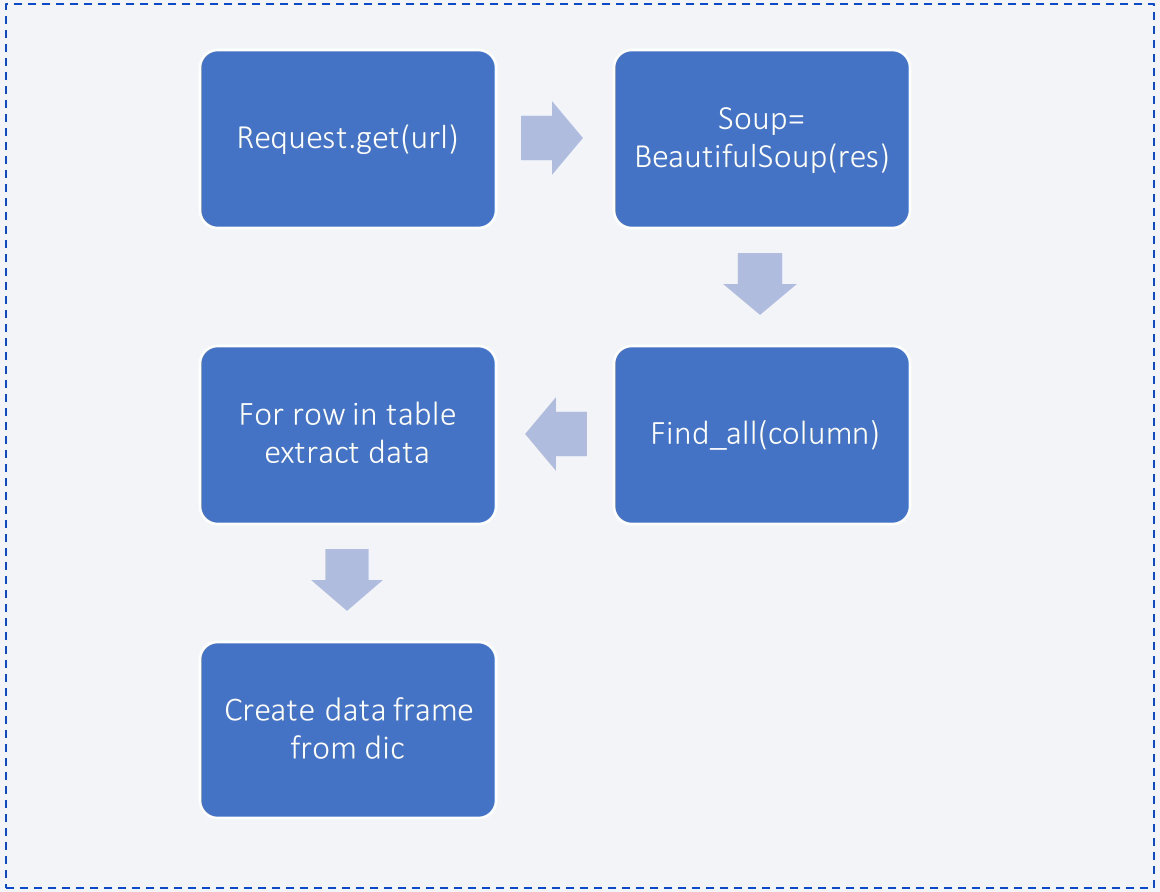
# Data Collection – SpaceX API

- Using the requests.get method we obtain the relevent Json file and normalize it.
- We then created the data frame using launch dictionary.
- Using loc[] we the filter the data frame to include only Falcon 9 launches.
- Github SpaceX API



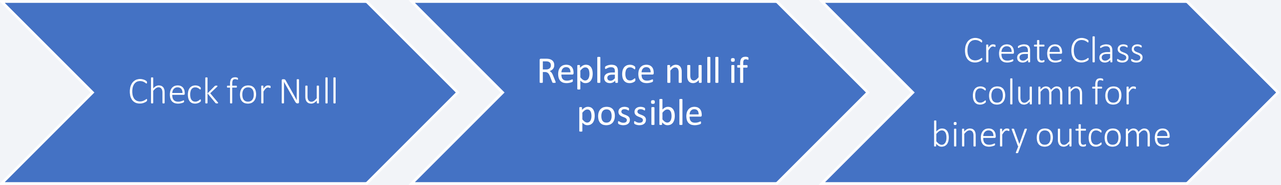
# Data Collection - Scraping

- Create a BeautifulSoup object from html
- Using find_all we extract all variable names
- Then we extract all the information to a dictionary for generating data frame.
- GitHub URL



# Data Wrangling

- In the data set, there are several different cases where the booster did not land successfully.
- we converted those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.
- GitHub URL



# EDA with Data Visualization

- In order to find connections between the features to the output class we used data visualization tools.
  - Ploting FlightNumber vs PayloadMass and overly the outcome of the launch We see that different launch sites have different success rates.
  - Bar graph for the success rate of orbits shows that different orbits have different success rates.
  - Furthermore, line plot of years vs success rate show that the sucess rate since 2013 kept increasing till 2020

- GitHub URL

# EDA with SQL

- Using Jupyter lab and %%sql we been able to run some queries and better understand the data as can be seen in the link below.
- GitHub URL

# Build an Interactive Map with Folium

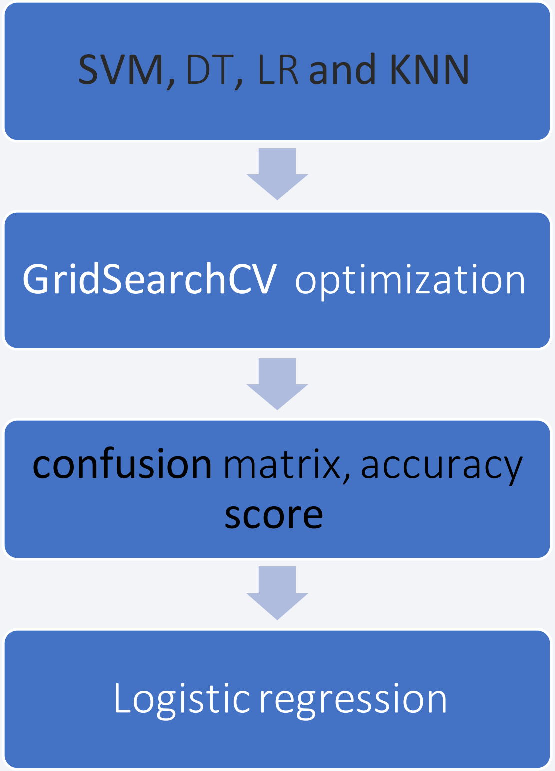
- we created Folium map with the launch sites as an objects
- We added markers, circles that shows the successes and failure for each launch
- Doing so we can get a visualization of the launch sites and successes.
- GitHub URL

# Build a Dashboard with Plotly Dash

- We present pie charts for both success ratio for specific site and success ratio of all places.
- We also plot a scatter plot of success base on payload.
- By that we can check the effect of launch sites and payload on success.
- GitHub URL

# Predictive Analysis (Classification)

- We construct SVM, decision tree, logistic regression and KNN
- Using GridSearchCV we optimize each classifier
- Using confusion matrix and accuracy score we evaluate them
- We then choose the logistic regression with performance score of 0.94 on the test set
- GitHub URL



SVM, DT, LR and KNN

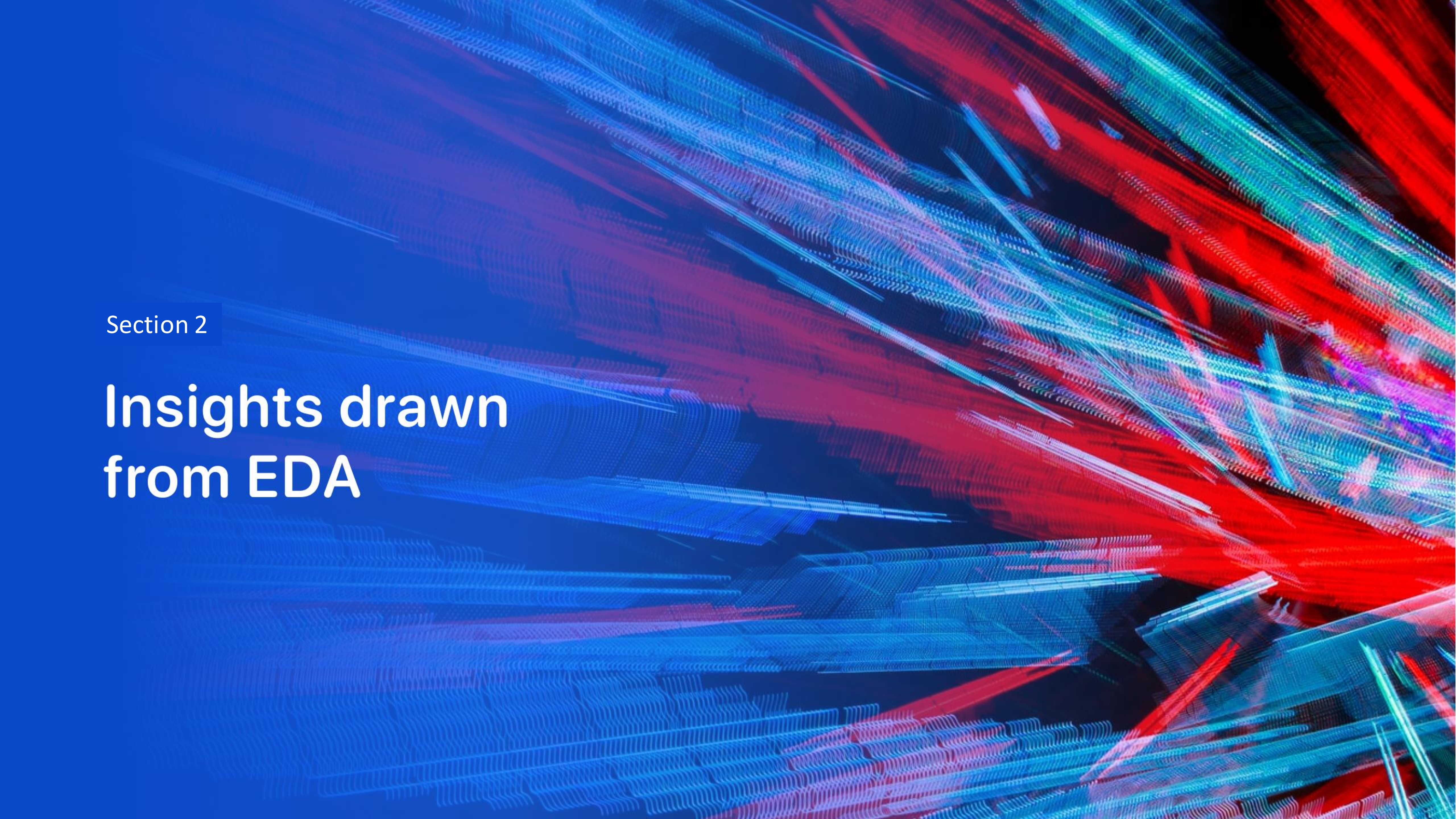
GridSearchCV optimization

confusion matrix, accuracy score

Logistic regression

# Results

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

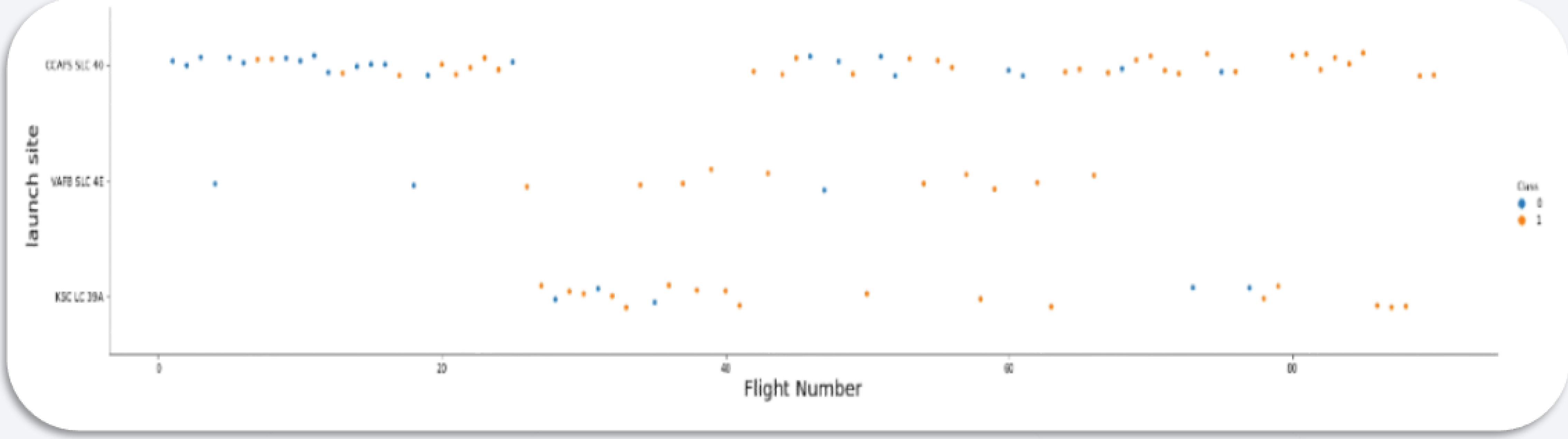


Section 2

# Insights drawn from EDA

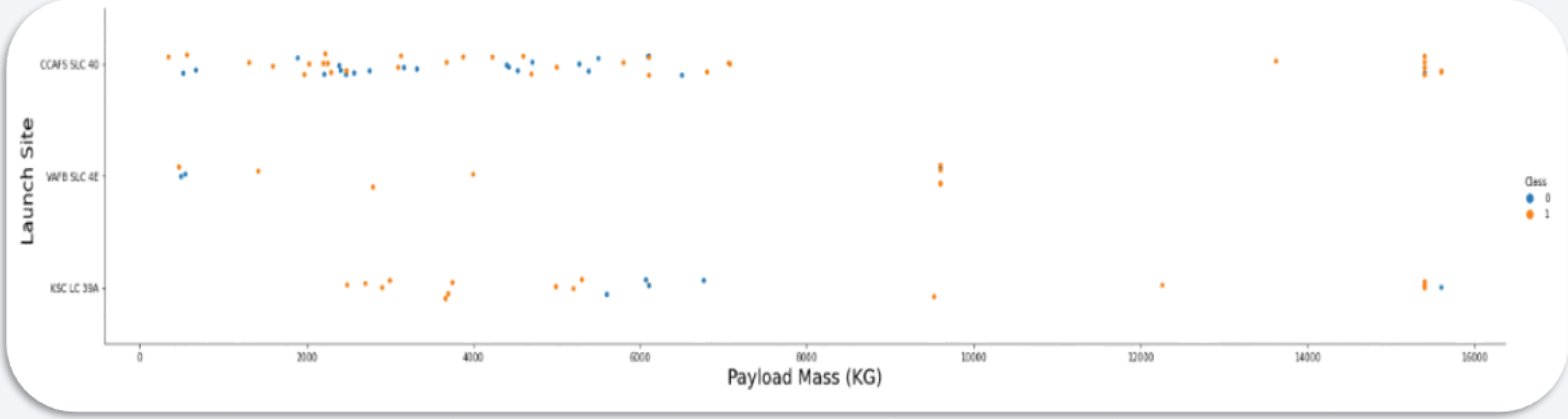
# Flight Number vs. Launch Site

- Note that:
  - As the flight number increases, the first stage is more likely to land successfully.
  - Most of our data is about CCAFS LC-40



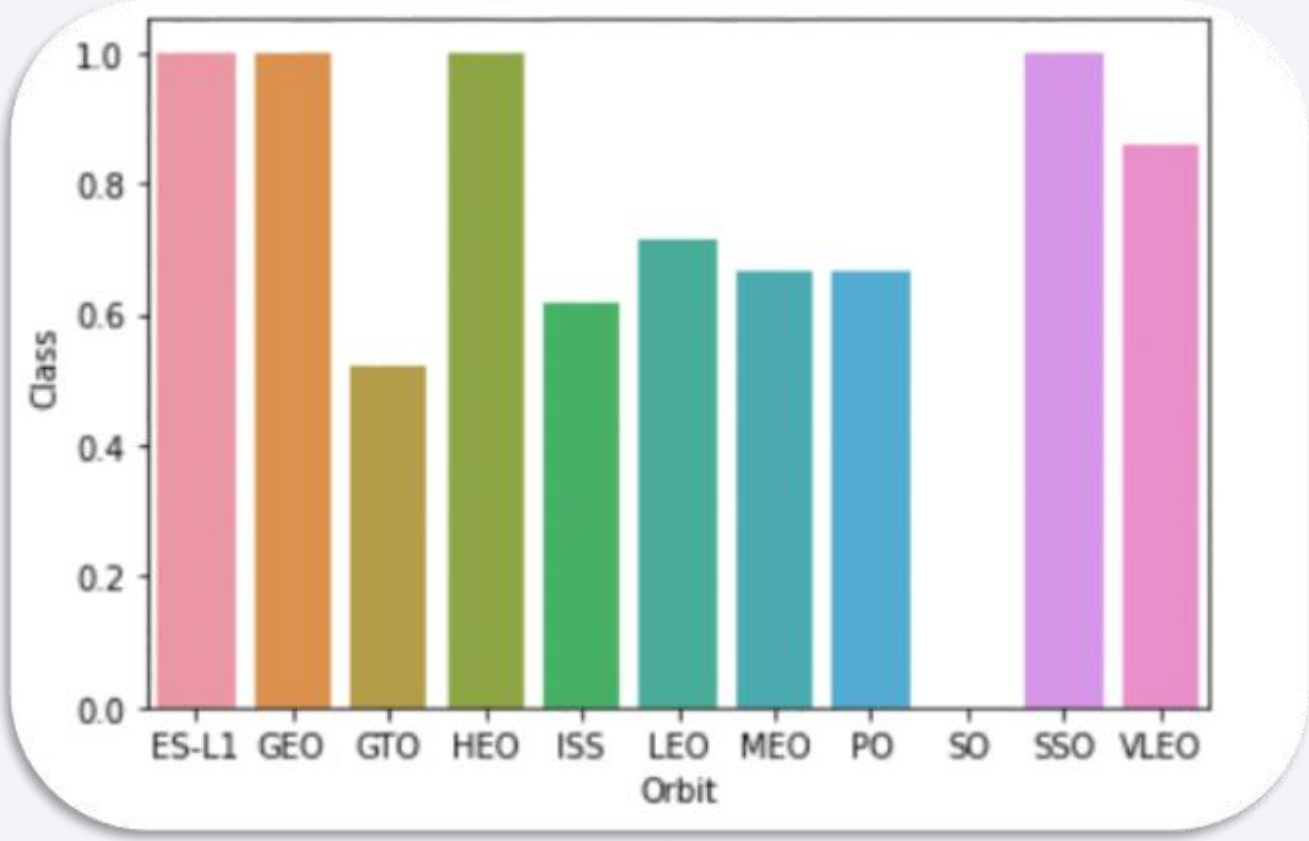
# Payload vs. Launch Site

- Note that:
  - the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).



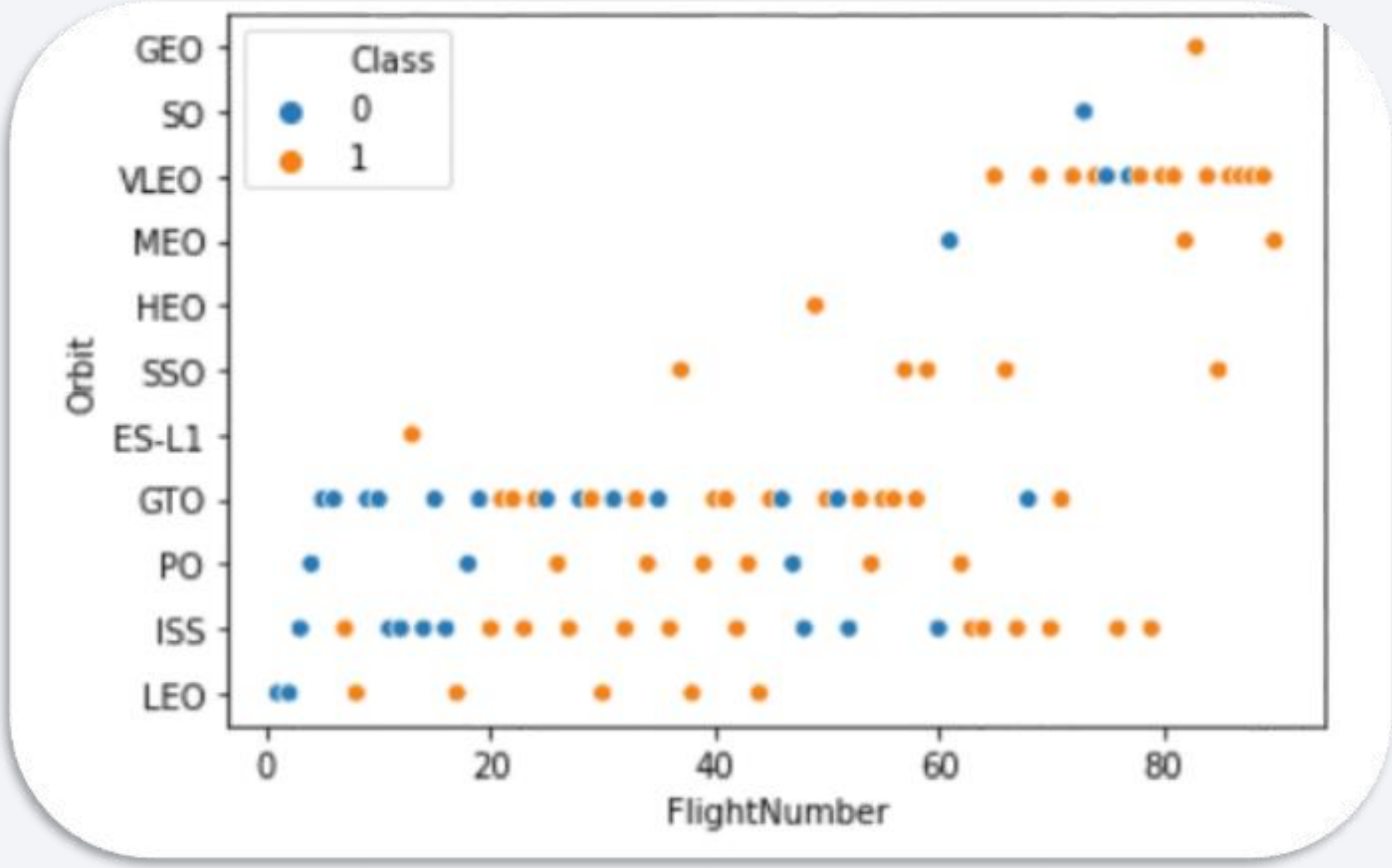
# Success Rate vs. Orbit Type

- Note that:
  - ES-L1, GEO, HEO, SSO orbits have the highest success rate.
  - GTO with the lowest



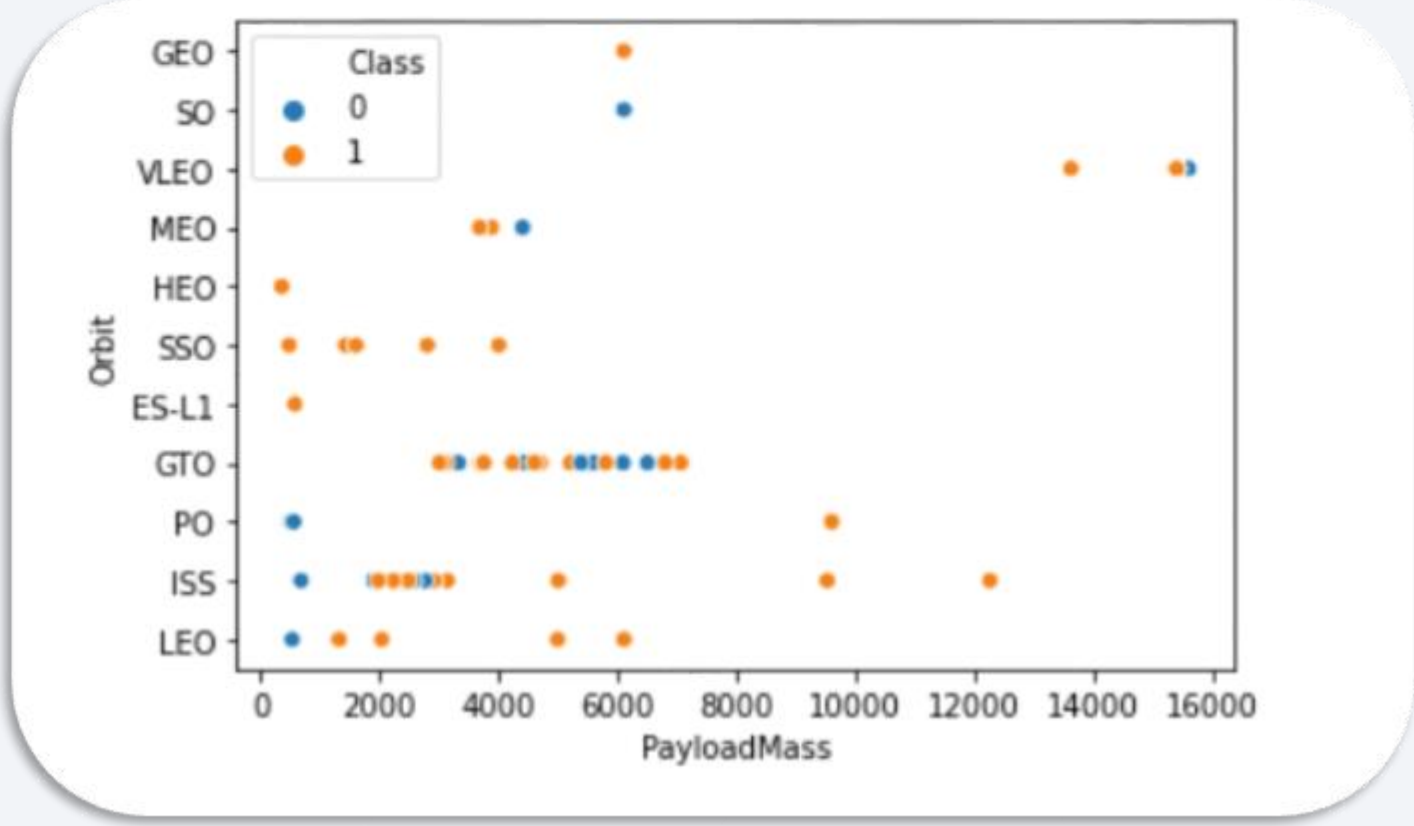
# Flight Number vs. Orbit Type

- Note that:
  - For the LEO orbit the success appears related to the number of flights
  - There seems to be no relationship between flight number when in GTO orbit.



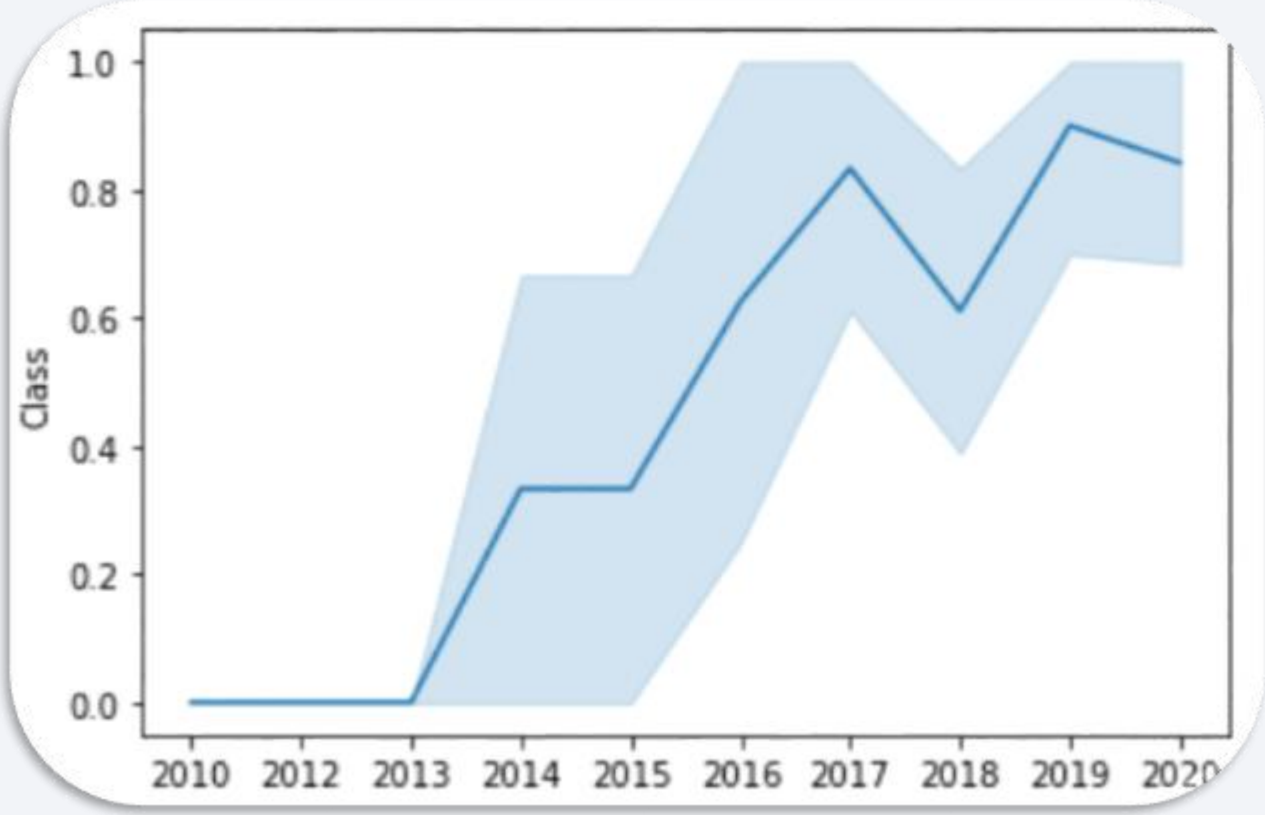
# Payload vs. Orbit Type

- Note that:
  - Heavy payloads effect the successful landing rate
  - Polar, LEO and ISS orbits effected even more
  - For GTO orbit it semes that we cannot distinguish success and failure by payload mass.



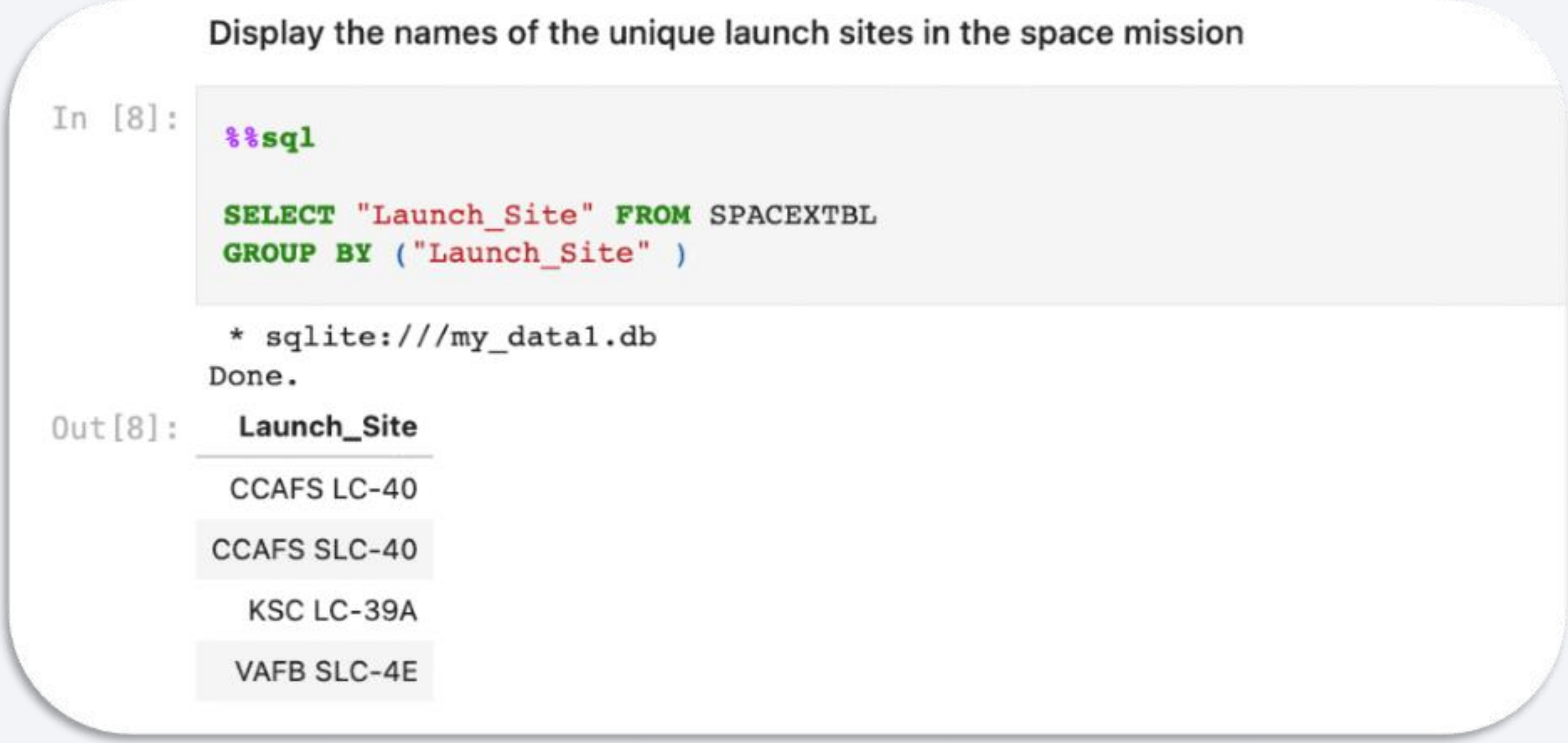
# Launch Success Yearly Trend

- Note that:
  - The sucess rate since 2013 kept increasing till 2020



# All Launch Site Names

- Find the names of the unique launch site



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

In [8]:

```
%%sql

SELECT "Launch_Site" FROM SPACEXTBL
GROUP BY ("Launch_Site" )
```

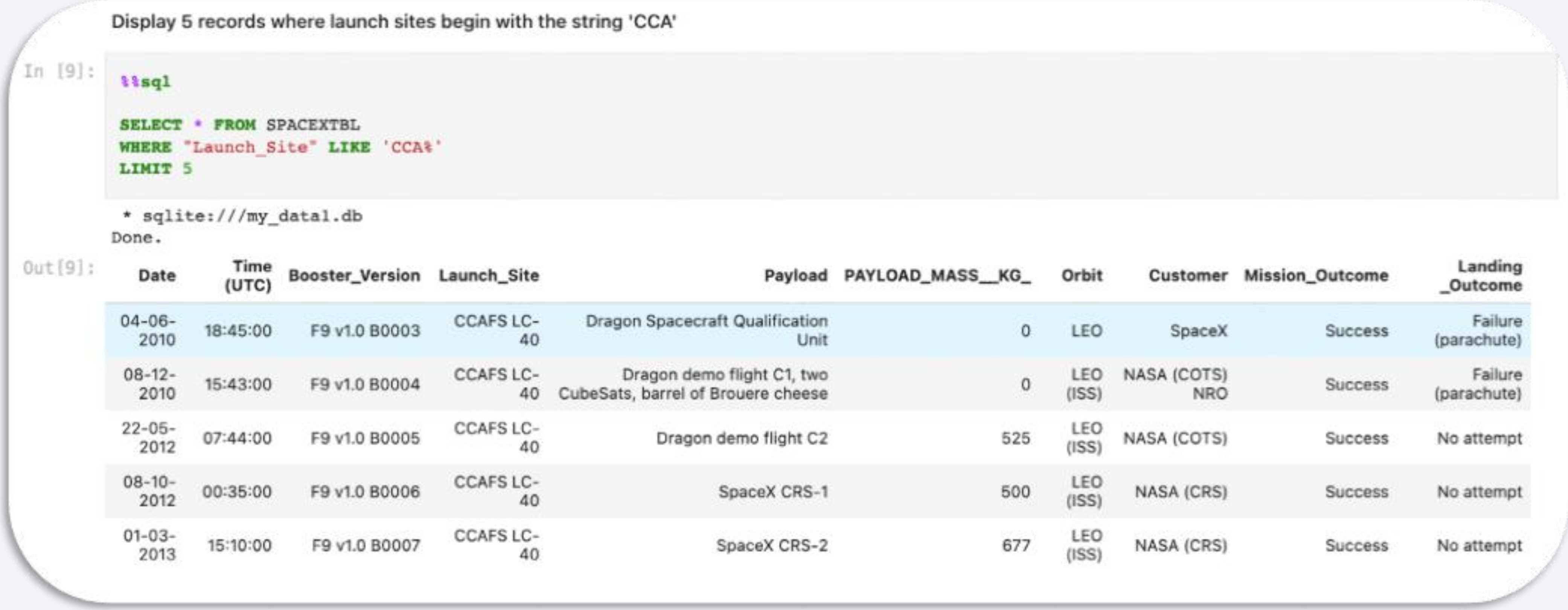
```
 * sqlite:///my_data1.db
Done.
```

Out[8]:

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

# Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`



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

In [9]:

```
%%sql

SELECT * FROM SPACEXTBL
WHERE "Launch_Site" LIKE 'CCA%'
LIMIT 5
```

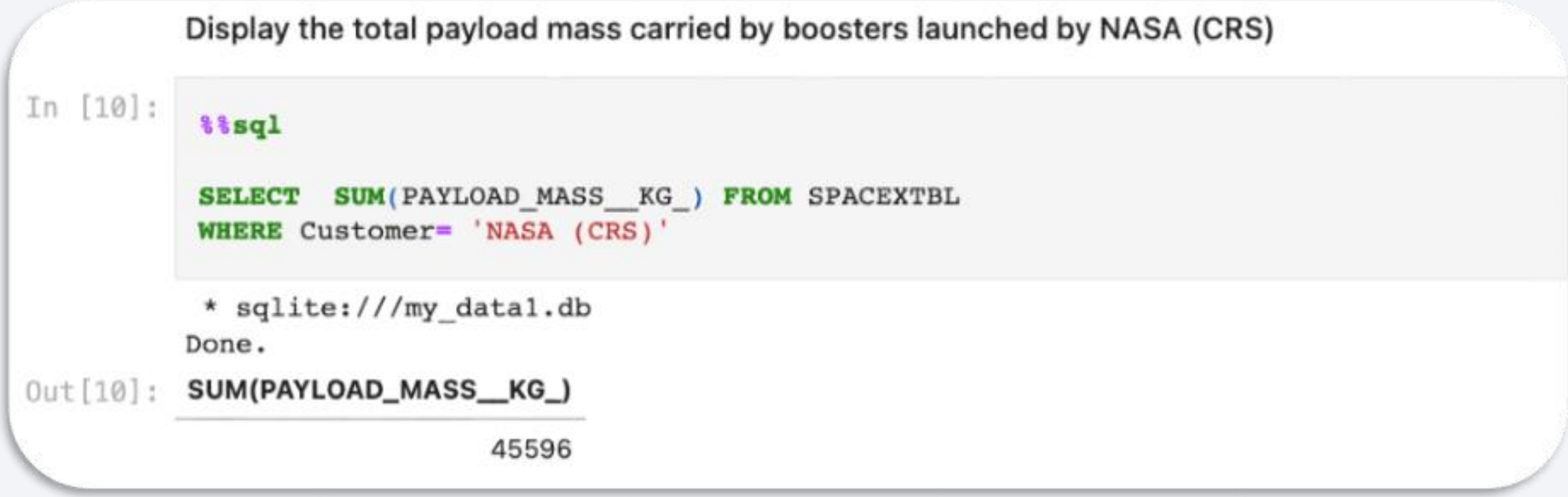
```
 * sqlite:///my_data1.db
Done.
```

Out[9]:

| 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

- Calculate the total payload carried by boosters from NASA

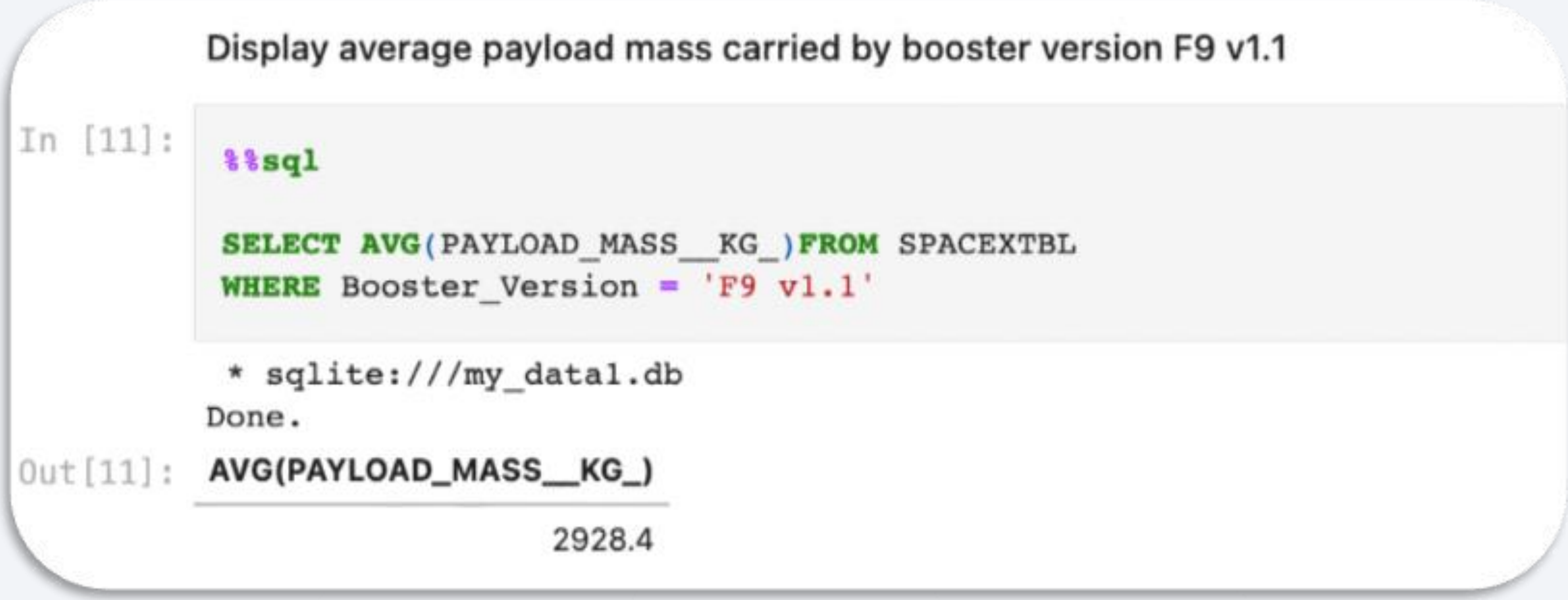


Display the total payload mass carried by boosters launched by NASA (CRS)

```
In [10]: %%sql

         SELECT  SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL
         WHERE Customer= 'NASA (CRS)'

          * sqlite:///my_data1.db
         Done.
Out[10]: SUM(PAYLOAD_MASS__KG_)
                          45596
```

# Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1



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

In [11]:

```
%%sql

SELECT AVG(PAYLOAD_MASS__KG_)FROM SPACEXTBL
WHERE Booster_Version = 'F9 v1.1'
```

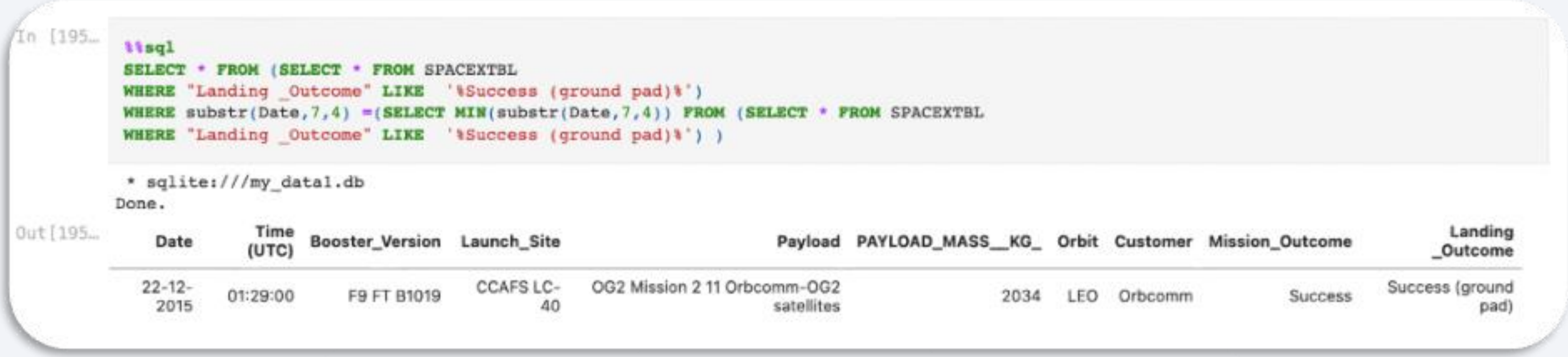
```
 * sqlite:///my_data1.db
Done.
```

Out[11]:

| AVG(PAYLOAD_MASS__KG_) |
|---|
| 2928.4 |

# First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Note that I had a problem with the Date as it find the min by dd and not all dd-mm-yyyy.



In [195…

```
%%sql
SELECT * FROM (SELECT * FROM SPACEXTBL
WHERE "Landing _Outcome" LIKE  '%Success (ground pad)%')
WHERE substr(Date,7,4) =(SELECT MIN(substr(Date,7,4)) FROM (SELECT * FROM SPACEXTBL
WHERE "Landing _Outcome" LIKE  '%Success (ground pad)%') )
```

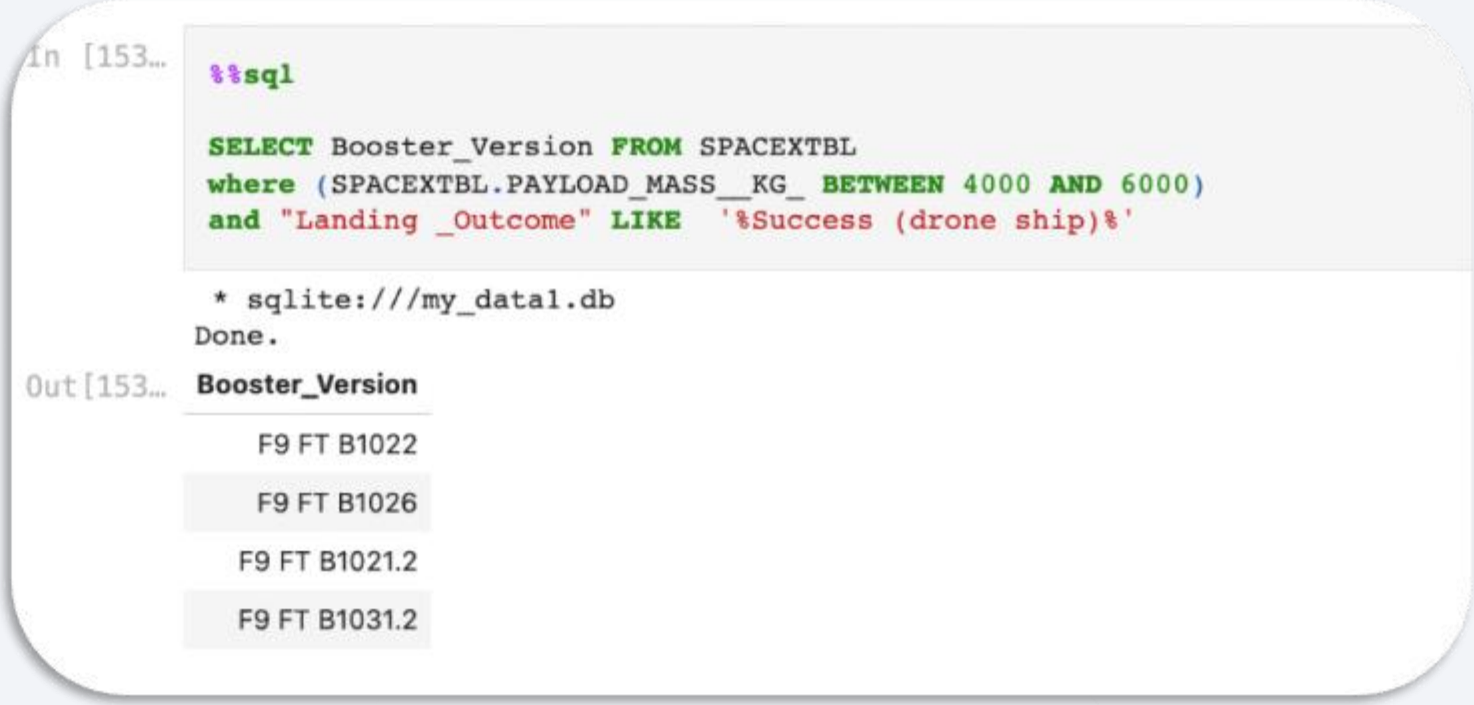
```
 * sqlite:///my_data1.db
Done.
```

Out[195…

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing _Outcome |
|---|---|---|---|---|---|---|---|---|---|
| 22-12-2015 | 01:29:00 | F9 FT B1019 | CCAFS LC-40 | OG2 Mission 2 11 Orbcomm-OG2 satellites | 2034 | LEO | Orbcomm | Success | Success (ground pad) |

# Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000



In [153...

```
%%sql

SELECT Booster_Version FROM SPACEXTBL
where (SPACEXTBL.PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000)
and "Landing _Outcome" LIKE  '%Success (drone ship)%'
```

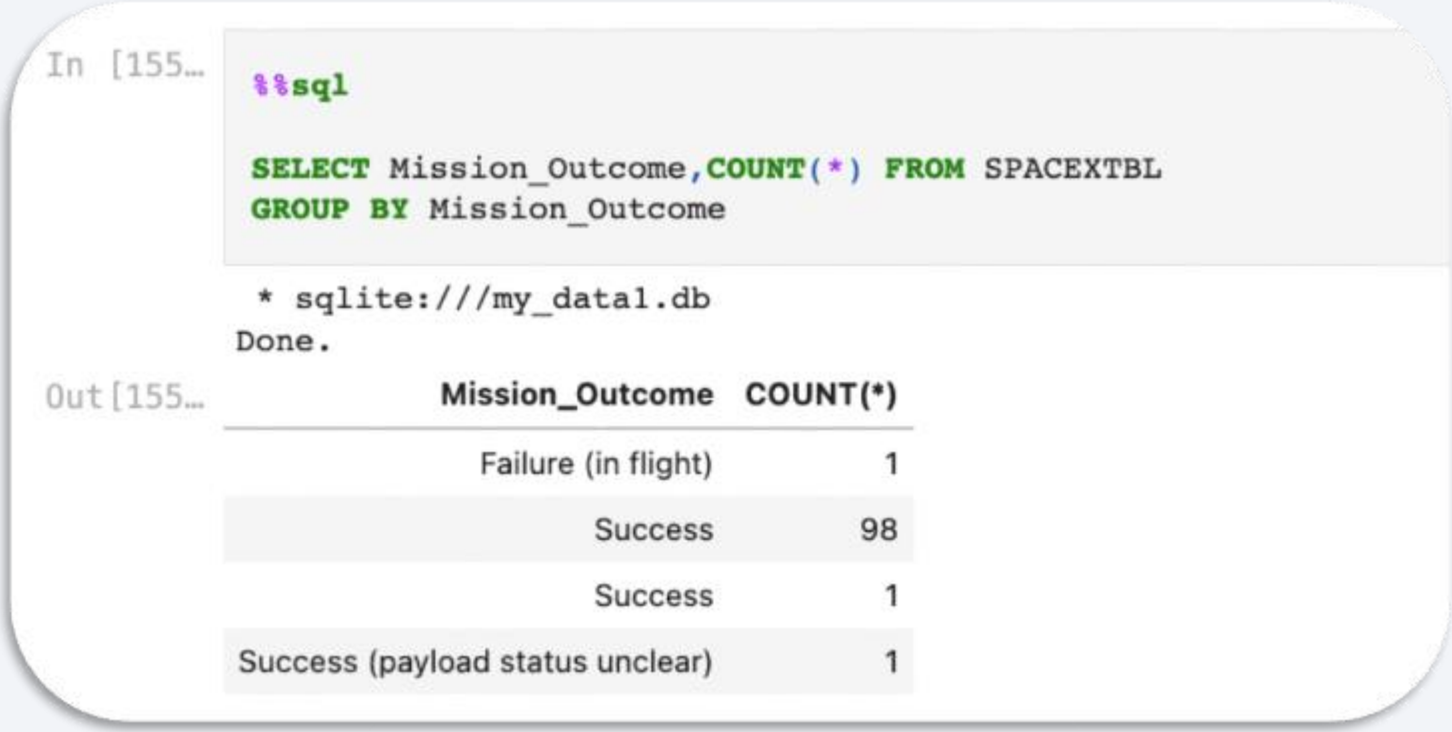
```
 * sqlite:///my_data1.db
Done.
```

Out[153...

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

# Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes



In [155…

```
%%sql

SELECT Mission_Outcome,COUNT(*) FROM SPACEXTBL
GROUP BY Mission_Outcome
```

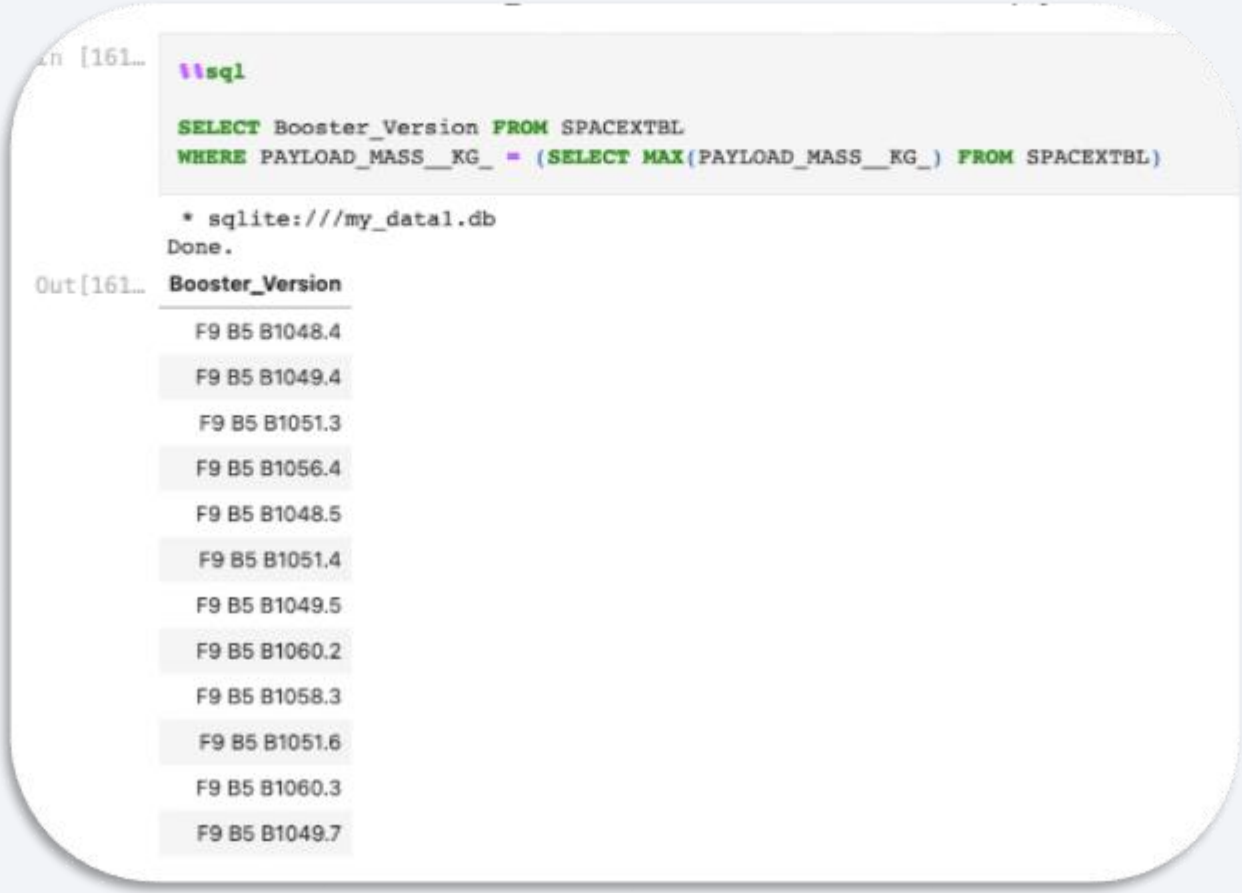
```
 * sqlite:///my_data1.db
Done.
```

Out[155…

| Mission_Outcome | COUNT(*) |
|---:|---:|
| Failure (in flight) | 1 |
| Success | 98 |
| Success | 1 |
| Success (payload status unclear) | 1 |

# Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass



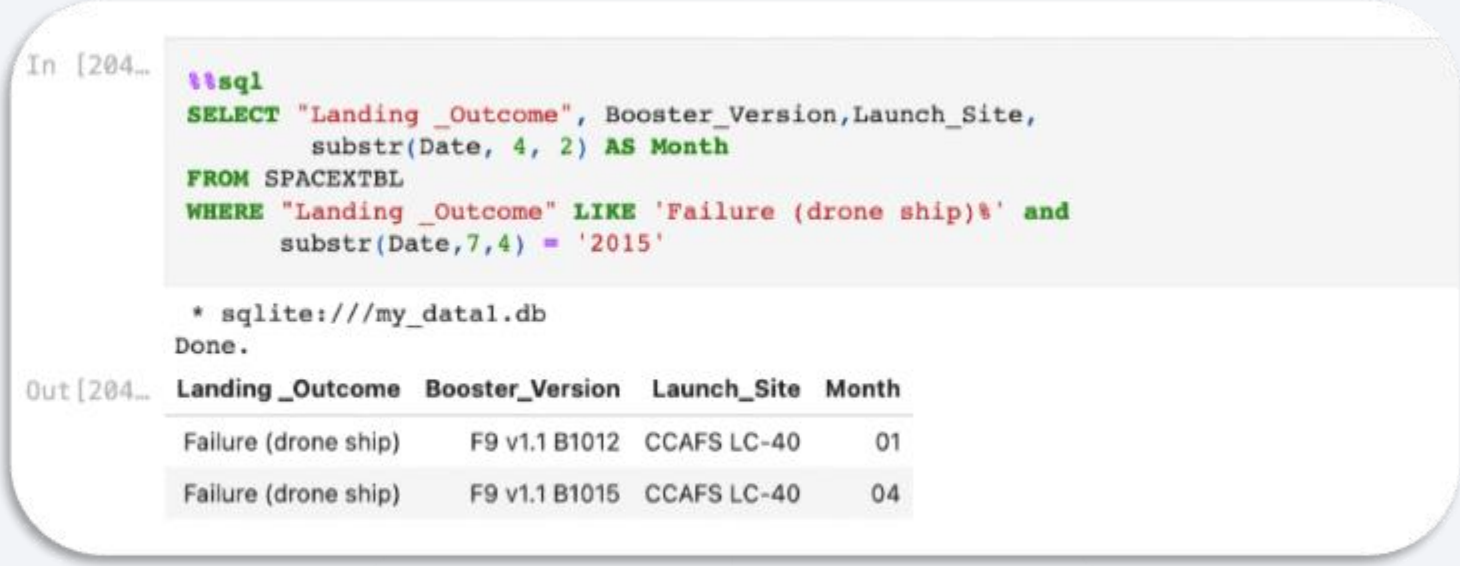
```
%%sql

SELECT Booster_Version FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

```
 * 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

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



In [204…

```
%%sql
SELECT "Landing _Outcome", Booster_Version,Launch_Site,
        substr(Date, 4, 2) AS Month
FROM SPACEXTBL
WHERE "Landing _Outcome" LIKE 'Failure (drone ship)%' and
       substr(Date,7,4) = '2015'
```

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

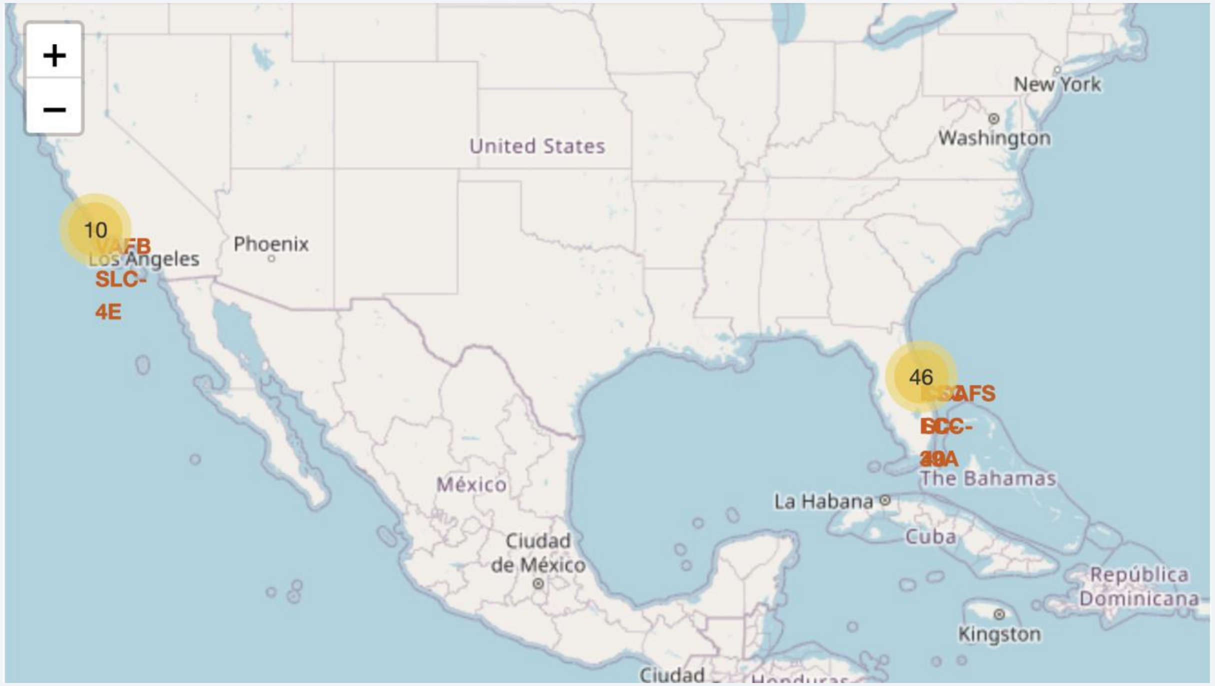
Out[204…

| Landing _Outcome | Booster_Version | Launch_Site | Month |
|---|---|---|---|
| Failure (drone ship) | F9 v1.1 B1012 | CCAFS LC-40 | 01 |
| Failure (drone ship) | F9 v1.1 B1015 | CCAFS LC-40 | 04 |

Section 3

# Launch Sites Proximities Analysis

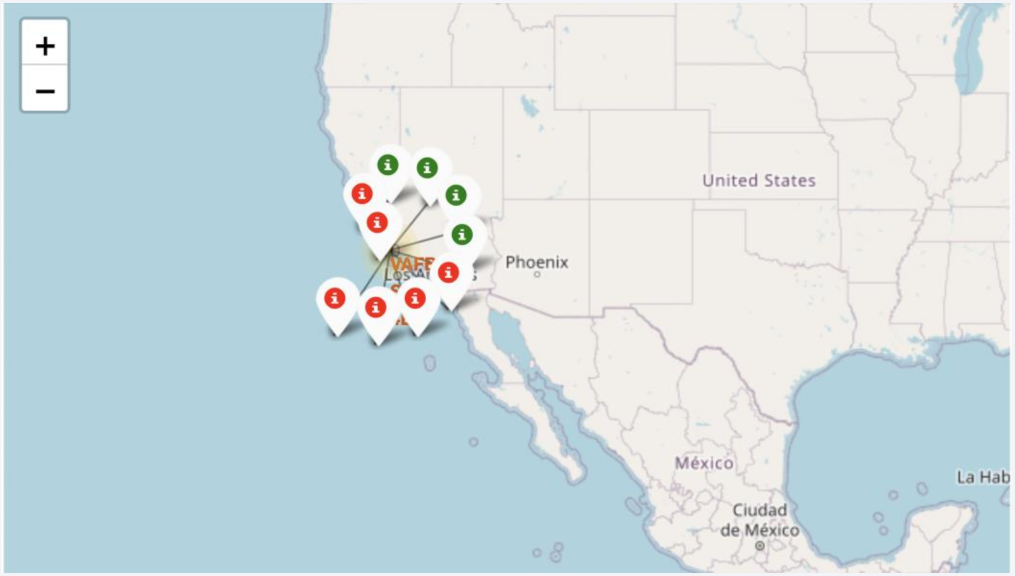
# Launch sites location map

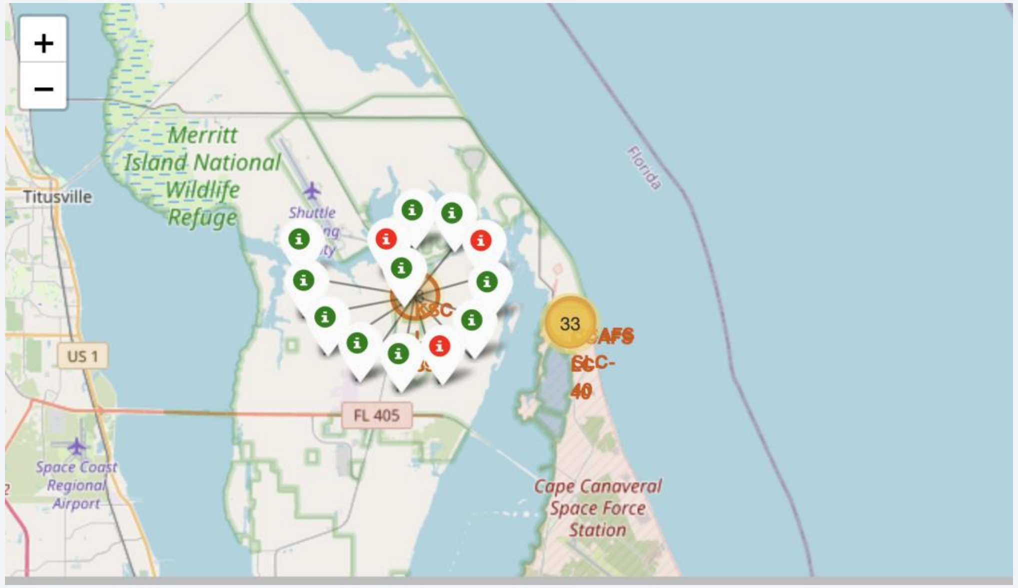


- We can see that the launch sites near the equator

# Launch sites success rates map

- In the right figure we can see the success and failure of in the east side of the previous figure
- In the right figure we can see the success and failure of in the west side of the previous figure




+
−
Merritt Island National Wildlife Refuge
Titusville
US 1
FL 405
33
AFS
CCSFS SLC-40
Cape Canaveral Space Force Station
Space Coast Regional Airport
Florida

# <Folium Map Screenshot 3>

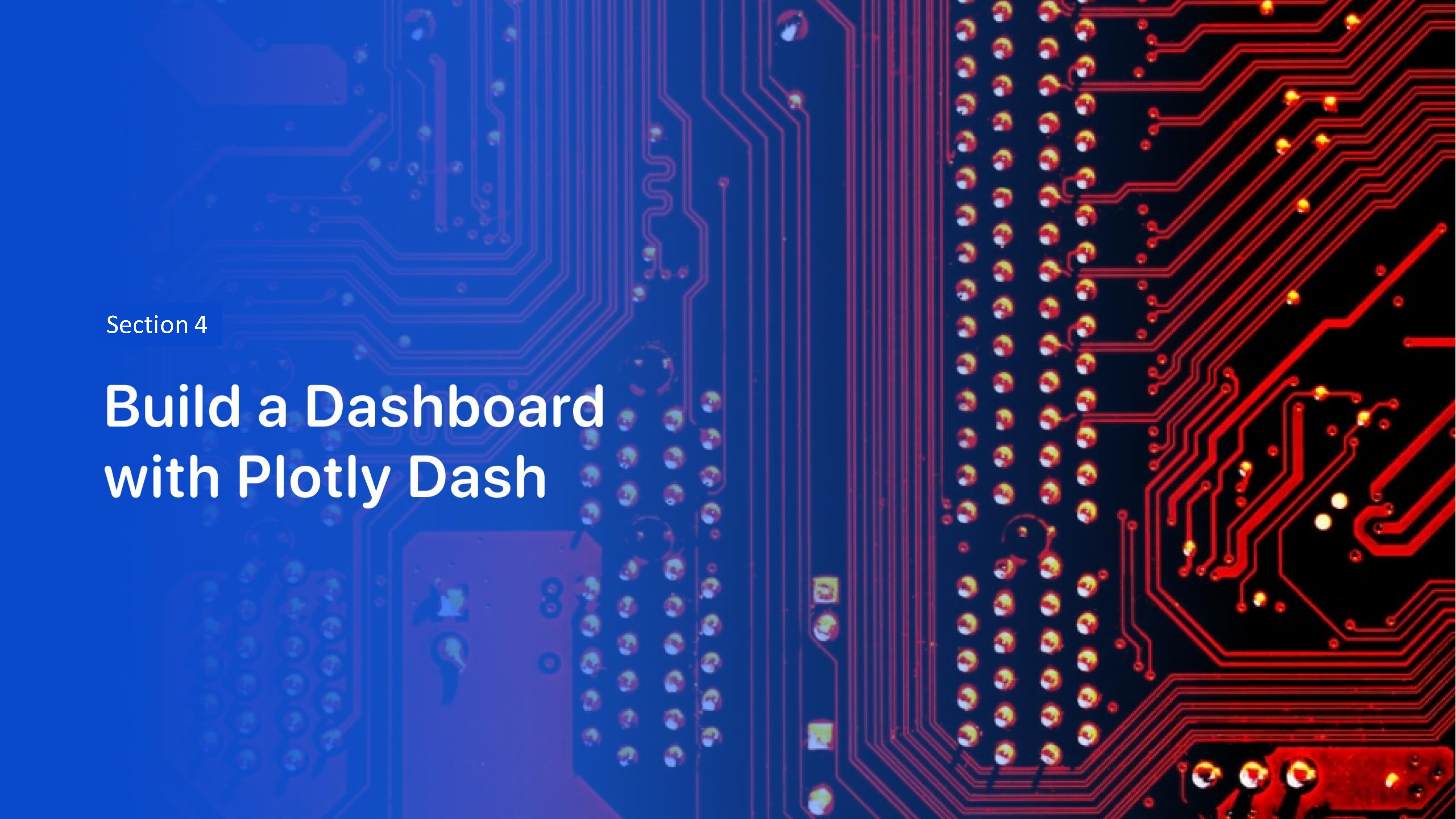
- Replace <Folium map screenshot 3> title with an appropriate title

- 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

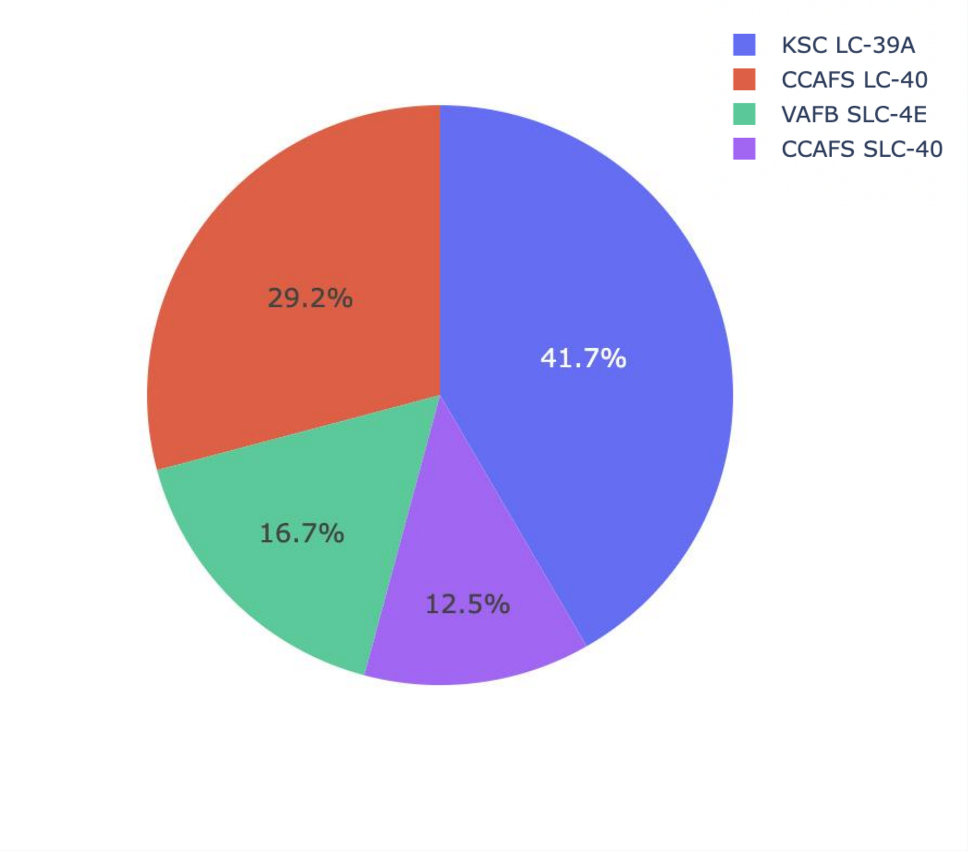
Section 4

# Build a Dashboard with Plotly Dash



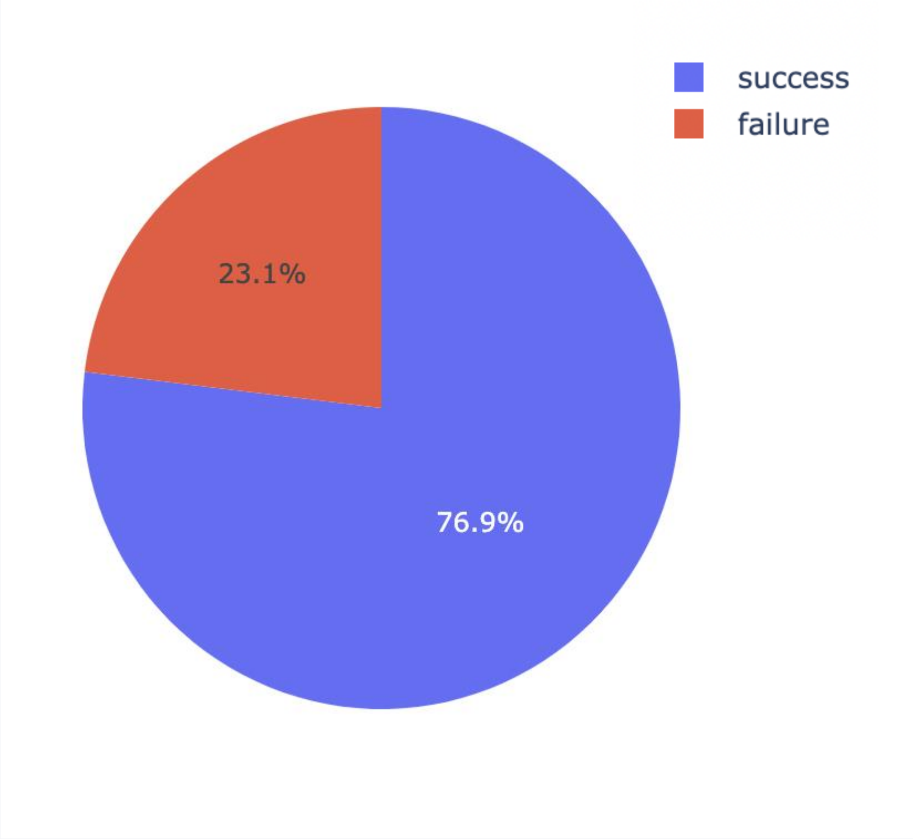
# Successful launches for all sites

- We can see that most of the successful launches are from KSC LC-39A

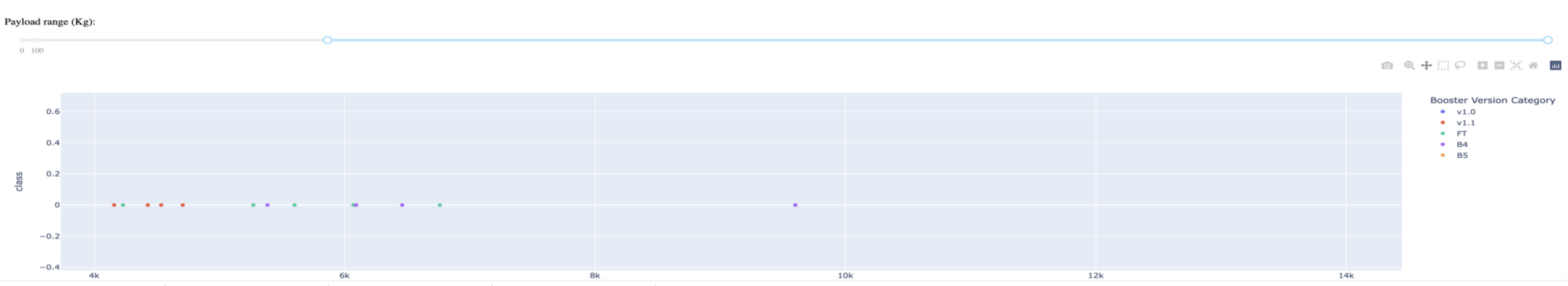


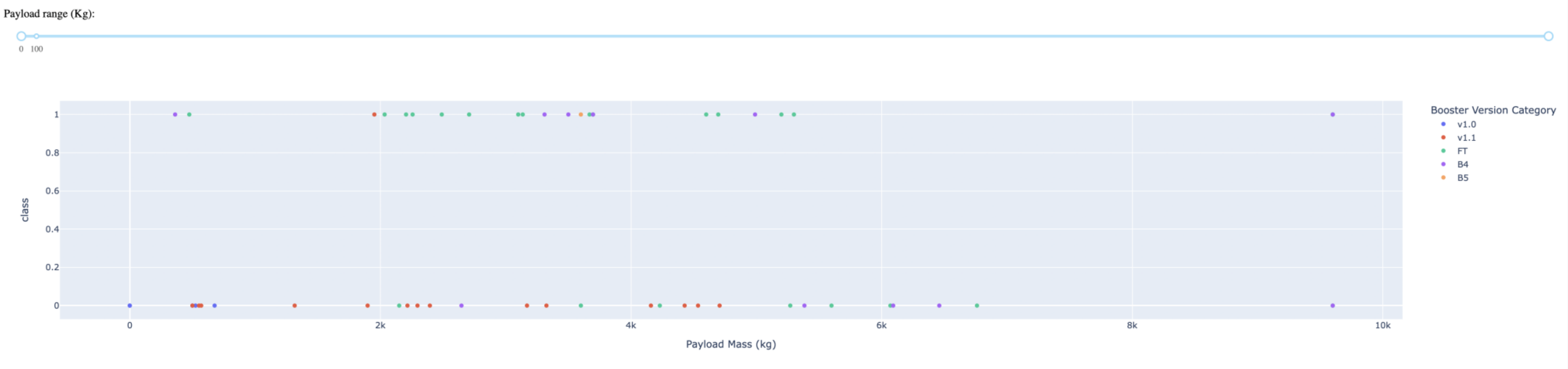
# KSC LC-39A launch success ratio

- We can see that almost 77% of the launches where success



# Payload vs Launch outcome scatter plot



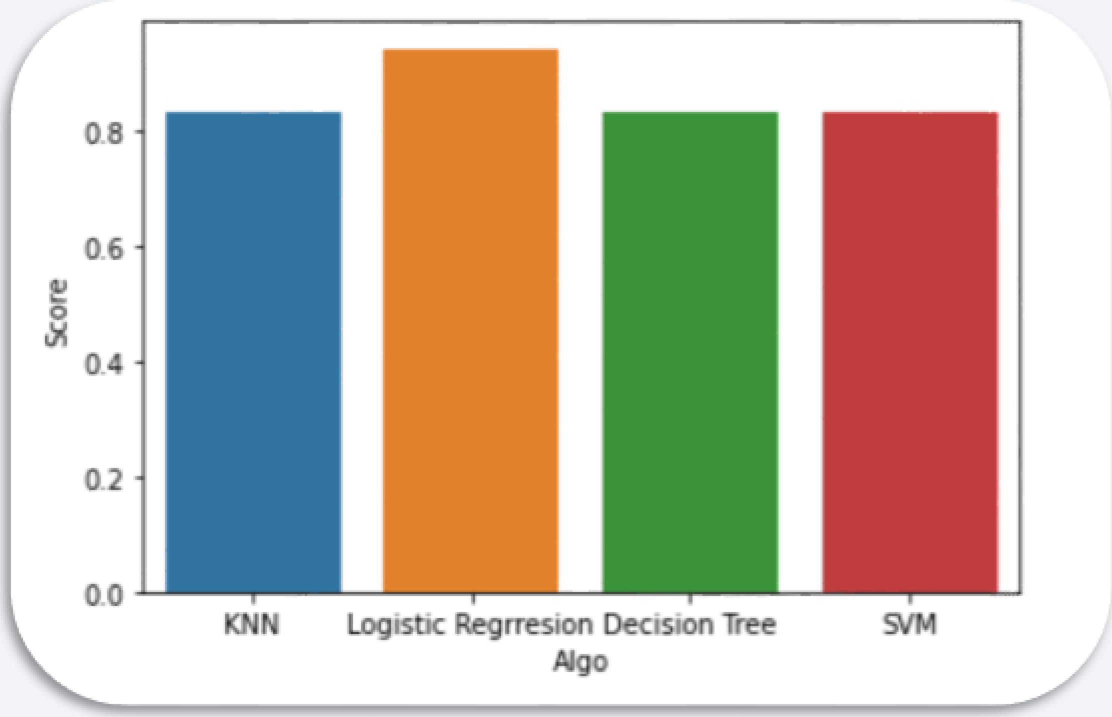


Section 5

# Predictive Analysis (Classification)

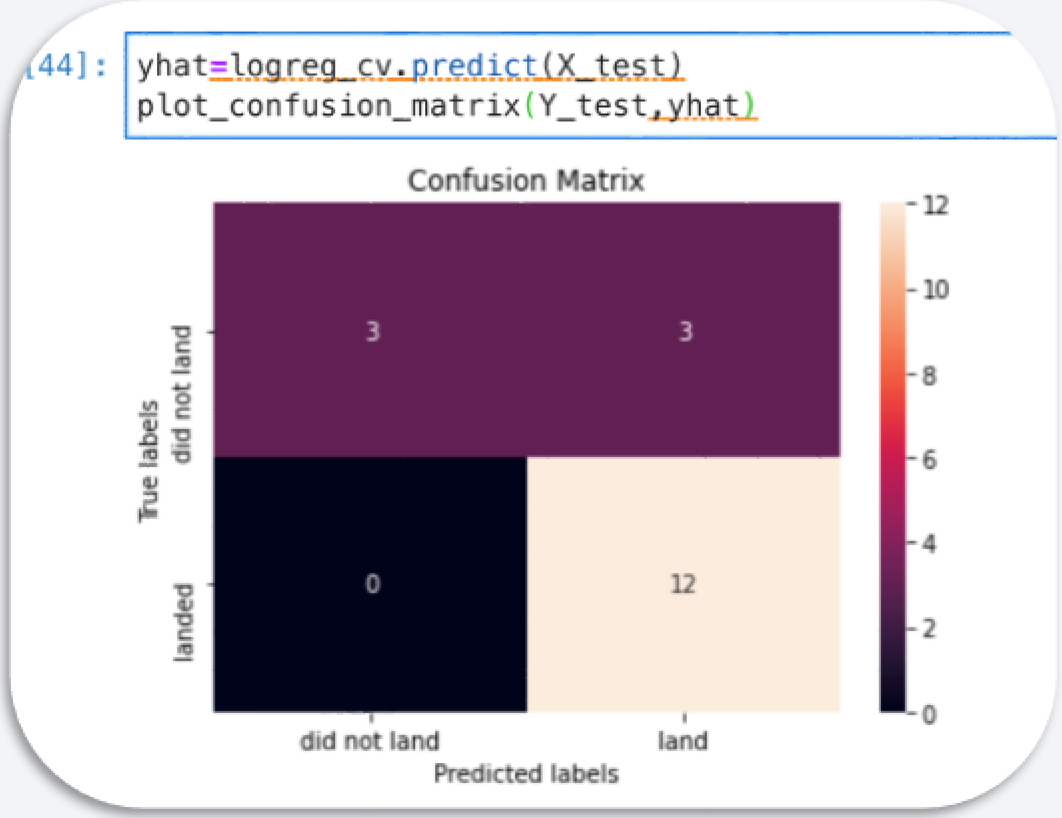
# Classification Accuracy

- We plot the score of the test set for each algo



# Confusion Matrix

- We see that the major problem is false positives.
- logistic regression can distinguish between the different classes.



```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

# Conclusions

- logistic regression can distinguish between the different classes with high confidence.
- Since we want to predict success we would need to try and improve the data or the model to decrease the false positive cases.

# 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!