IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Yang Jiawei
03/01/2023

Outline

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

Executive Summary

Summary of Methodologies:

Methods

- Data Collection Based on API and Web Scraping
- Data Wrangling
- Exploratory Data Analytics (EDA)
- Interactive Visual Analytics
- Predictive Analysis
- EDA with SQL
- Interactive Map with Folium
- Dashboard with Plotly Dash

Summary of Results:

Outcomes

- EDA Results
- Interactive Maps and Dashboard
- Predictive Analysis of Classification Models

Introduction

- **Project background and context:**
- The commercial space age is growing, SpaceX Falcon 9 rockets are leading in the industry currently as they are considerably cheaper than any alternatives provided by competitors, this is because they are reusable. The aim of this project is to predict if the Falcon 9 first stage will successfully landed. Based on the results, the cost of a launch can be determined.
- **Problems to find:**
- What is the main factor of a successful or failure landing?
- What is the impact of its correlation if the rocket variables on the successful rate of a landing?
- What is the condition that allows SpaceX to achieve the highest success rate?

Section 1

Methodology

Methodology

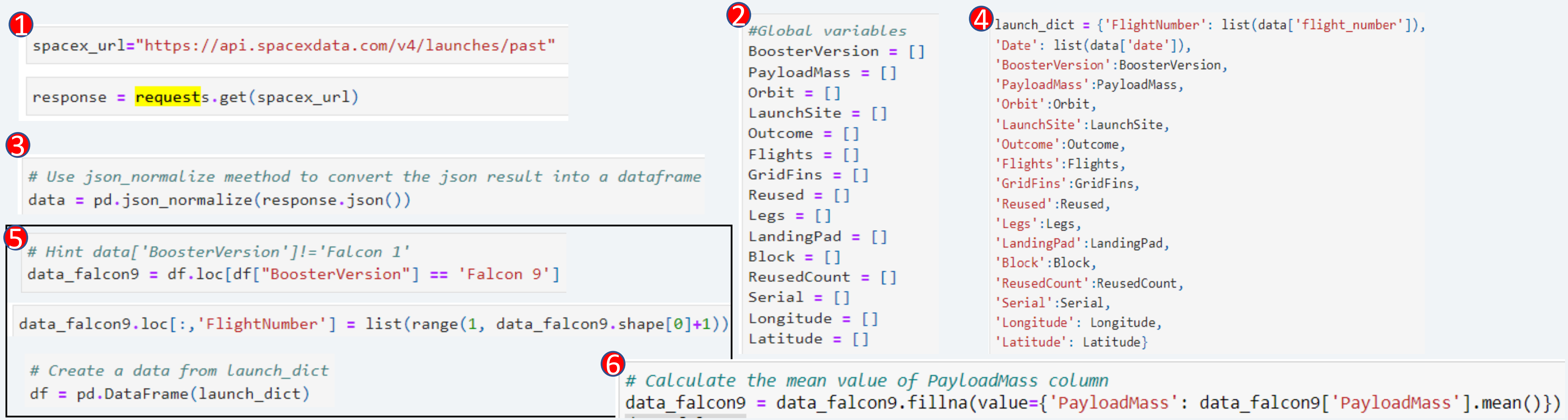
Executive Summary

1. Data Collection
   - Collecting Data from SpaceX API
   - Web Scrapping from Wikipedia
2. Data Wrangling
   - Classifying the status of the landings
3. EDA
   - SQL and Visualization
4. Interactive visual analytics using Folium and Plotly Dash
5. Predictive analysis using classification models

Data Collection – SpaceX API

GitHub URL

https://github.com/Mohicansy/IBM-Applied-Data-Science-Capstone-YYJW/blob/main/1)%20jupyter-labs-spacex-data-collection-api.ipynb



1

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

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

2

```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

3

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

4

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

5

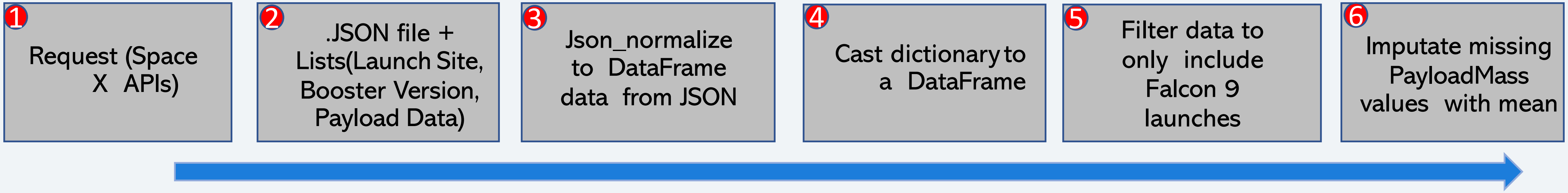
```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = df.loc[df["BoosterVersion"] == 'Falcon 9']
```

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
```

```
# Create a data from launch_dict
df = pd.DataFrame(launch_dict)
```

6

```
# Calculate the mean value of PayloadMass column
data_falcon9 = data_falcon9.fillna(value={'PayloadMass': data_falcon9['PayloadMass'].mean()})
```



1 Request (Space X APIs)

2 .JSON file + Lists(Launch Site, Booster Version, Payload Data)

3 Json_normalize to DataFrame data from JSON

4 Cast dictionary to a DataFrame

5 Filter data to only include Falcon 9 launches

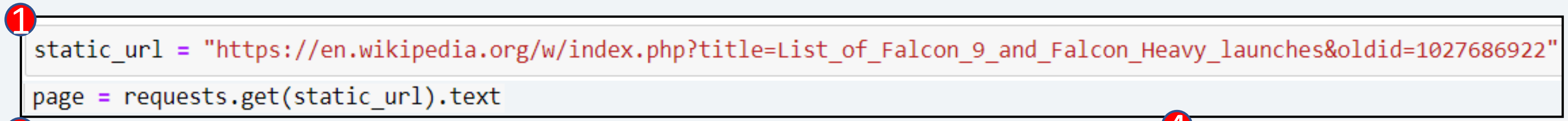
6 Imputate missing PayloadMass values with mean

Data Collection - Scraping

GitHub URL

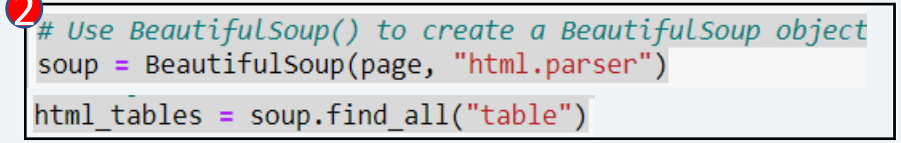
https://github.com/Mohicansy/IBM-Applied-Data-Science-Capstone-YYJW/blob/main/2)%20jupyter-labs-webscraping.ipynb

1



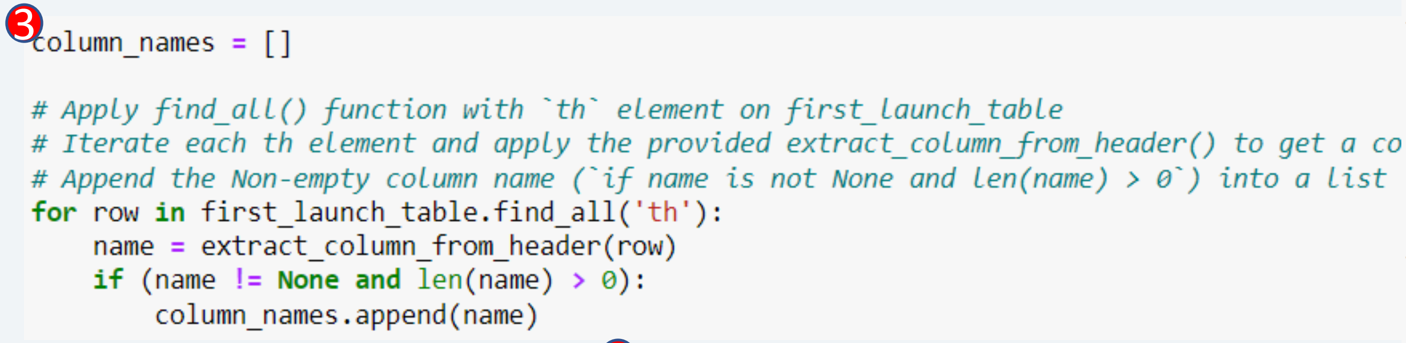
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
page = requests.get(static_url).text
```

2



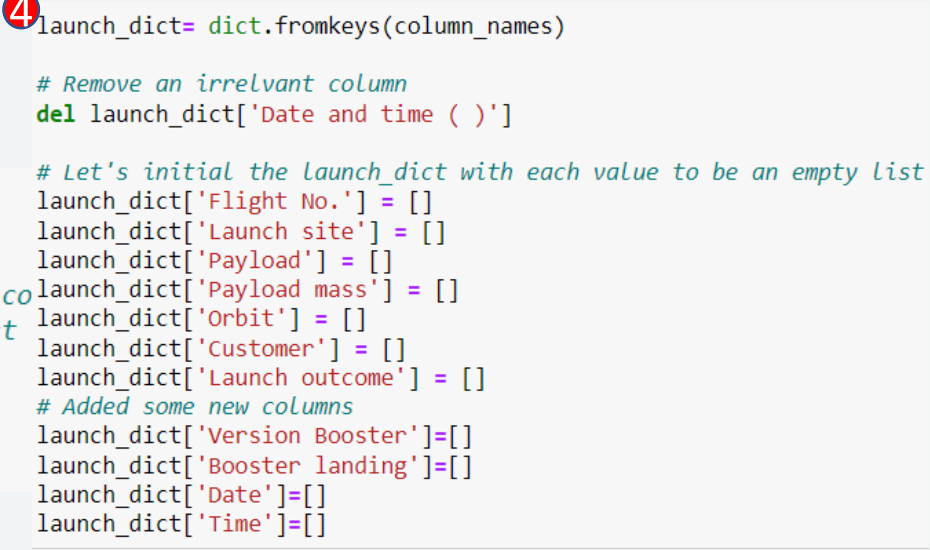
```
# Use BeautifulSoup() to create a BeautifulSoup object
soup = BeautifulSoup(page, "html.parser")
html_tables = soup.find_all("table")
```

3



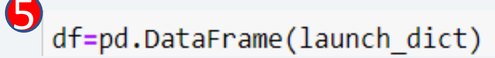
```
column_names = []

# 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 co
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if (name != None and len(name) > 0):
        column_names.append(name)
```

4

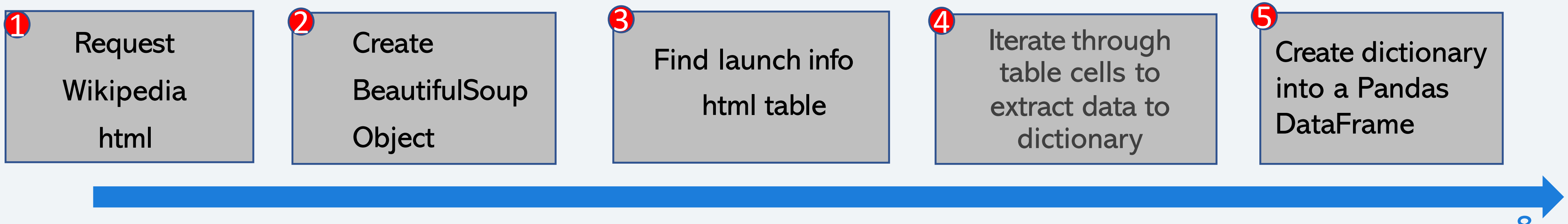


```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

5



```
df=pd.DataFrame(launch_dict)
```



1 Request Wikipedia html

2 Create BeautifulSoup Object

3 Find launch info html table

4 Iterate through table cells to extract data to dictionary

5 Create dictionary into a Pandas DataFrame

Data Wrangling

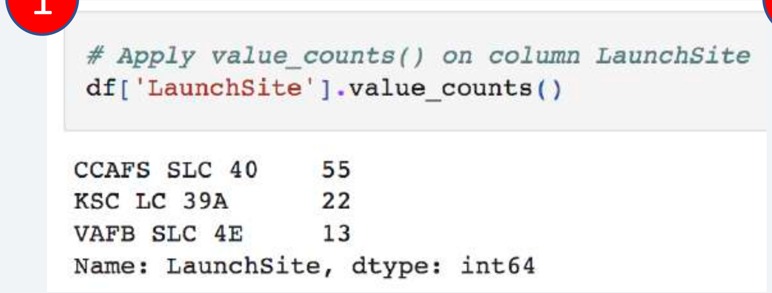
GitHub URL

https://github.com/Mohicansy/IBM-Applied-Data-Science-Capstone-YYJW/blob/main/3)IBM-DS0321EN-SkillsNetwork_labs_module_1_L3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

According to the dataset, there are cases where the booster failed to land (True: success; False: failure).

The string variable is needed to be transformed into categorical variables where '1=>success' and '0=>failure'.

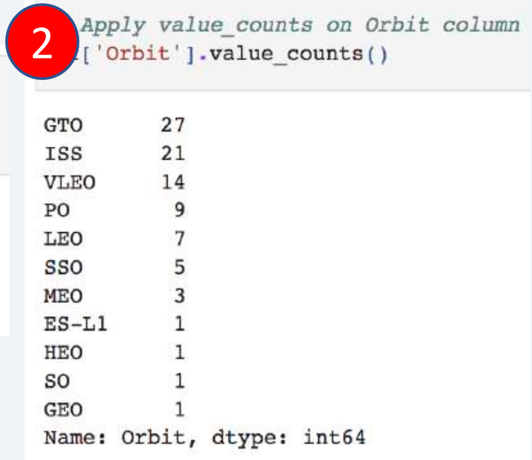
1



```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

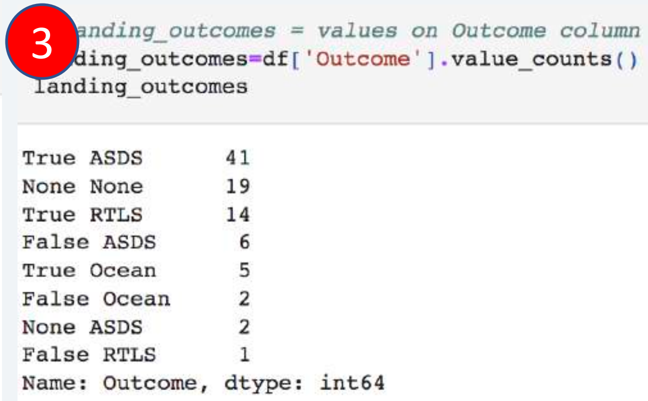
2



```
 Apply value_counts on Orbit column
[ 'Orbit' ].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

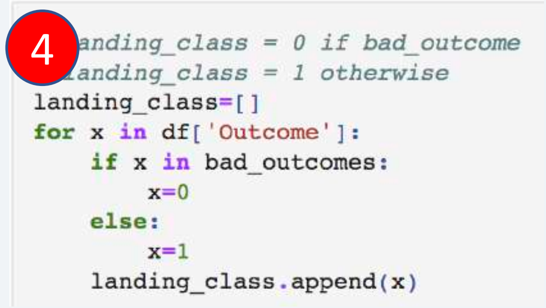
3



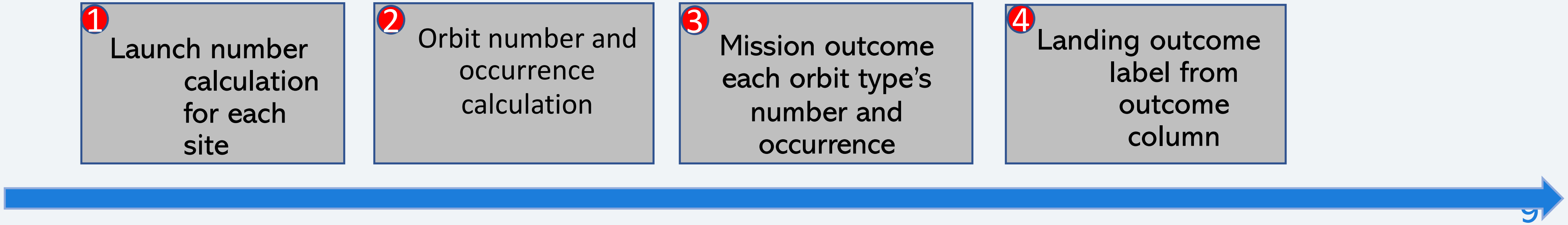
```
anding_outcomes = values on Outcome column
ding_outcomes=df['Outcome'].value_counts()
landing_outcomes
```

```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

4



```
anding_class = 0 if bad_outcome
anding_class = 1 otherwise
landing_class=[]
for x in df['Outcome']:
    if x in bad_outcomes:
        x=0
    else:
        x=1
    landing_class.append(x)
```



1 Launch number calculation for each site

2 Orbit number and occurrence calculation

3 Mission outcome each orbit type's number and occurrence

4 Landing outcome label from outcome column

EDA with Data Visualization

GitHub URL
https://github.com/Mohicansy/IBM-Applied-Data-Science-Capstone-YYJW/blob/main/5)IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

- Exploratory Data Analysis performed on variables Flight Number, Payload Mass, Launch Site, Orbit, Class and Year.
- Plots Method Used:
- **Scatter plots**: Flight Number vs. Payload Mass, Flight Number vs. Launch Site, Payload Mass vs. Launch Site, Flight Number vs. Orbit and Payload vs Orbit.
- **Line graph**: Years vs. Success Rate
- **Bar plots**: Orbit vs. Success Rate
- These method were used to compare relationships between variables to decide if a relationship exists so that they could be used in training the machine learning model.

EDA with SQL

GitHub URL
https://github.com/Mohicansy/IBM-Applied-Data-Science-Capstone-YYJW/blob/main/4)jupyter-labs-eda-sql-coursera_sqllite.ipynb

The sql queries are performed to gather data from 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 v1.1
- - 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. Use 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

Build an Interactive Map with Folium

GitHub URL
https://github.com/Mohicansy/IBM-Applied-Data-Science-Capstone-YYJW/blob/main/6)IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

The Folium map is used as centered on NASA JSC at Huston

1. Mark all launch sites on a map
   - Initialise the map using a Folium Map object
   - Add a folium.Circle and folium.Marker for each launch site on the launch map
2. Mark the success/failed launches for each site on a map
   - As many launches have the same coordinates, it makes sense to cluster them together.
   - Before clustering them, assign a marker colour of successful (class = 1) as green, and failed (class = 0) as red.
   - To put the launches into clusters, for each launch, add a folium.Marker to the MarkerCluster() object.
   - Create an icon as a text label, assigning the icon_color as the marker_colour determined previously.
3. Calculate the distances between a launch site to its proximities
   - To explore the proximities of launch sites, calculations of distances between points can be made using the Lat and Long values.
   - After marking a point using the Lat and Long values, create a folium.Marker object to show the distance.
   - To display the distance line between two points, draw a folium.PolyLine and add this to the map.

The launch sites, surroundings and the number of success and failure landing are shown in the above link.

Build a Dashboard with Plotly Dash

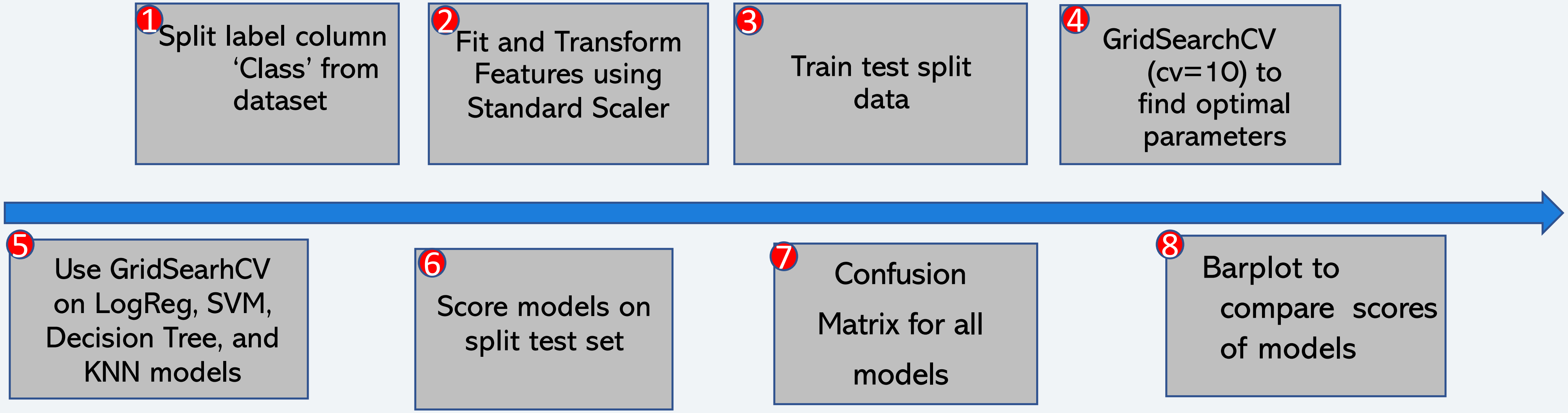
GitHub URL
https://github.com/Mohicansy/IBM-Applied-Data-Science-Capstone-YYJW/blob/main/7)spacex_dash_app.py

Dashboard has dropdown, pie chart, rangeslider and scatter plot components

- Dropdown: choose launch sites
- Pie chart: the total success or failure of the chosen launch sites
- Rangslider: select playload mass in a fixed range
- Scatter chart: the correlation between Success vs Payload Mass

Predictive Analysis (Classification)

GitHub URL
https://github.com/Mohicansy/IBM-Applied-Data-Science-Capstone-YYJW/blob/main/8)IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb



1 Split label column 'Class' from dataset

2 Fit and Transform Features using Standard Scaler

3 Train test split data

4 GridSearchCV (cv=10) to find optimal parameters

5 Use GridSearhCV on LogReg, SVM, Decision Tree, and KNN models

6 Score models on split test set

7 Confusion Matrix for all models

8 Barplot to compare scores of models

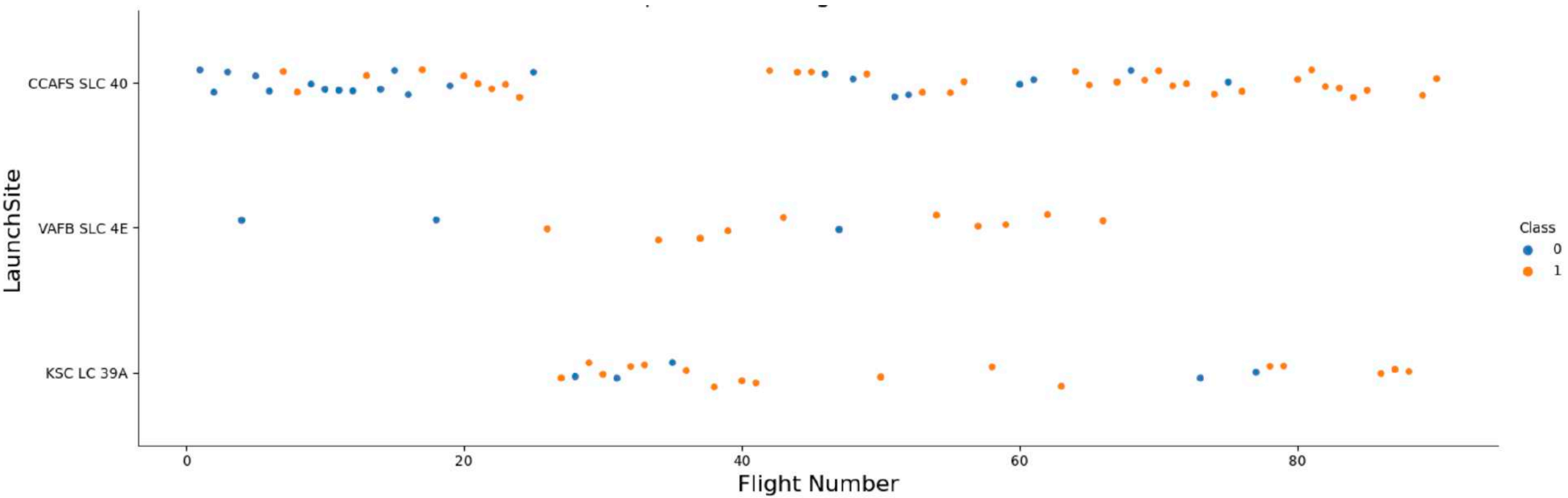
Results

- Decision Tree model is the best in terms of accuracy.
- Less weighted payloads have more success rate than heaver payloads.
- KSC LC 39A Launch Site is the best of all sites
- Orbits GEO, HEO, SSO, ES L1 have the best success rate.

Section 2

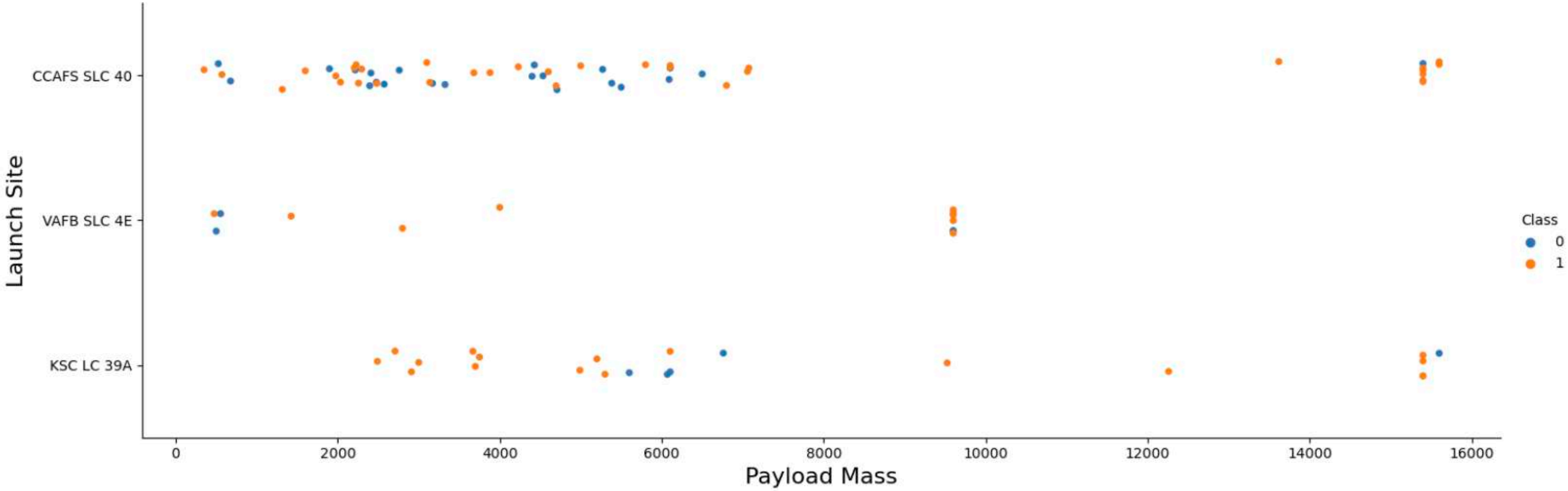
Insights drawn from EDA

Flight Number vs. Launch Site



According to the figure, an increase in success rate as more Flight were launched. It should be noted that when it is around flight 20 which significantly increased success rate. Above a flight number of 30, significantly more successful landings happened.

Payload vs. Launch Site

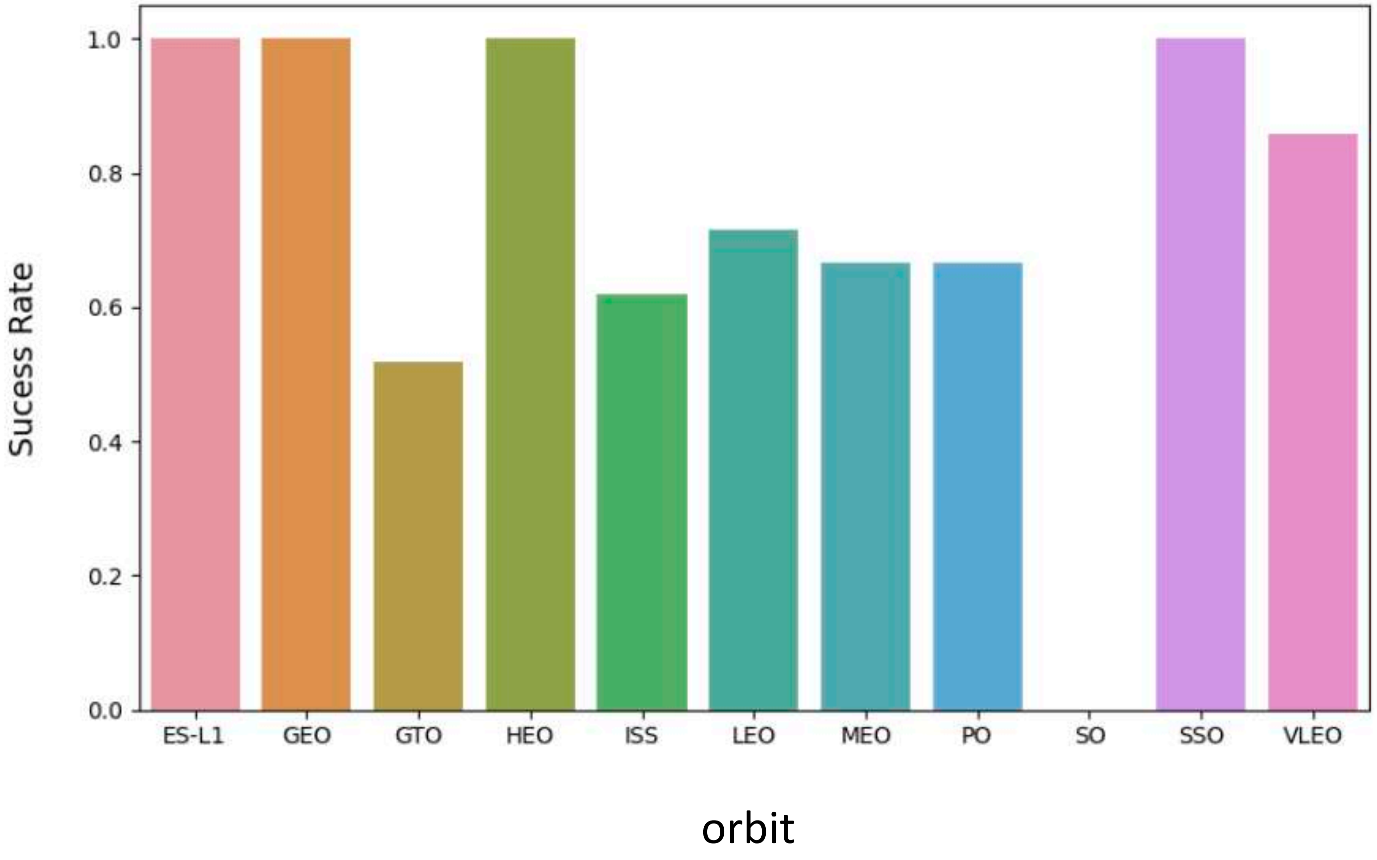


According to the figure, a heavier payload may be a consideration for a successful landing. Adversely, the payload could fail in landing when it is too heavy. There is no clear correlation between payload mass and success rate for a given launch site.

Success Rate vs. Orbit Type

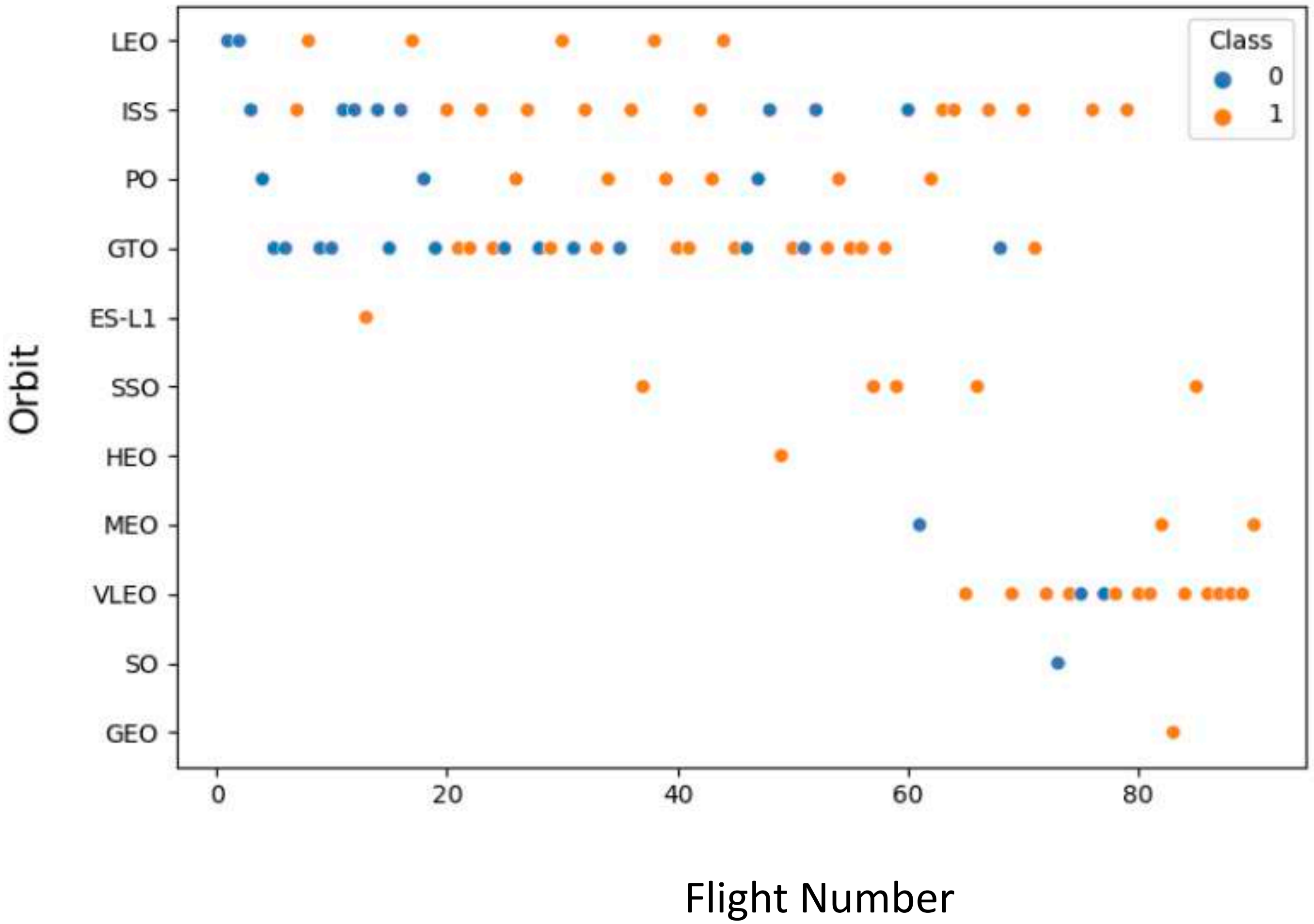
- ES-L1, GEO, HEO and SSO orbits have the highest successful rates.

- SO has the lowest success rate.



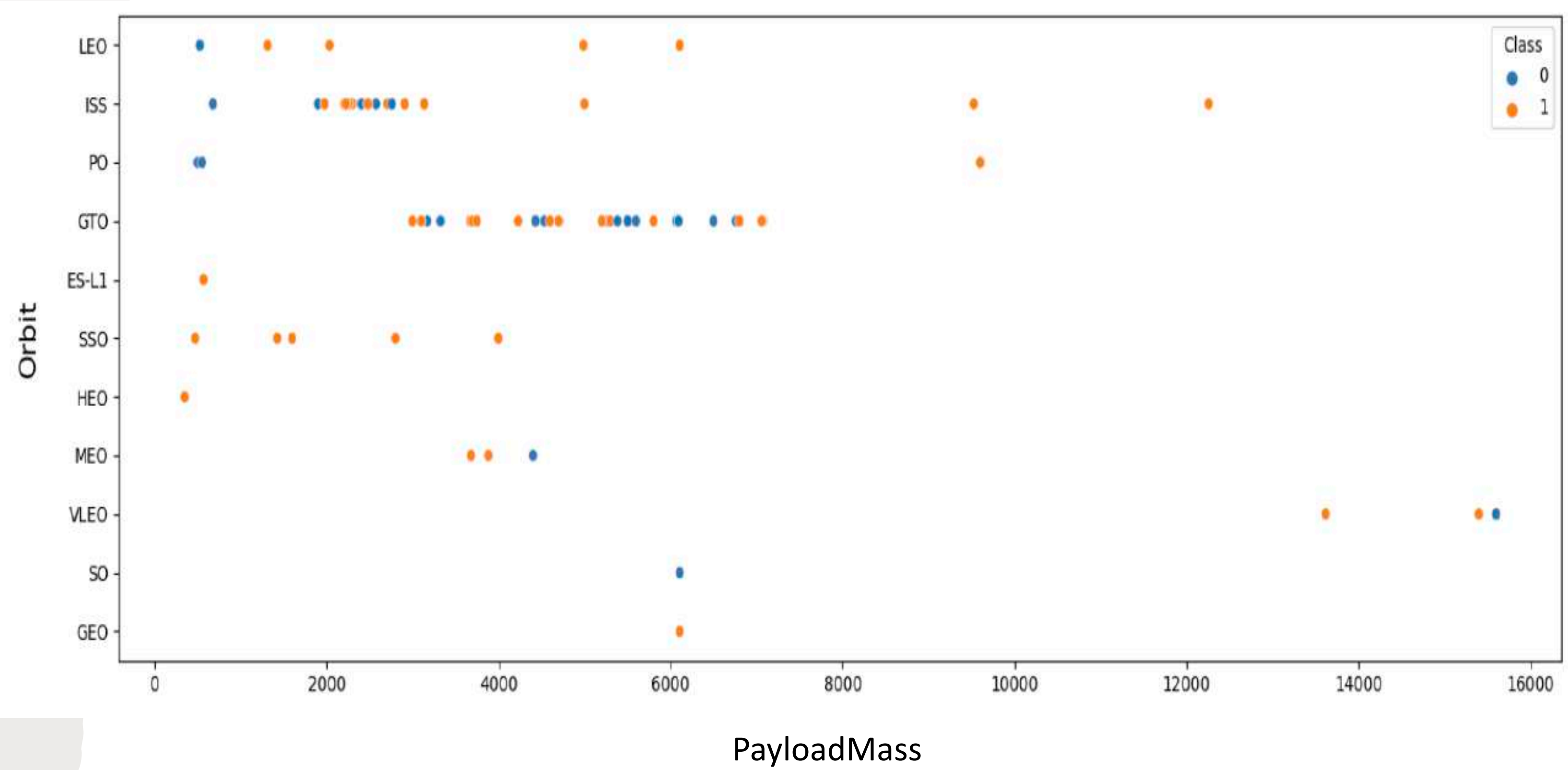
Flight Number vs. Orbit Type

Success rate increases with the number of flights for the LEO orbit. No obvious correlation between the success rate and the number of flights for the other orbits like GTO.



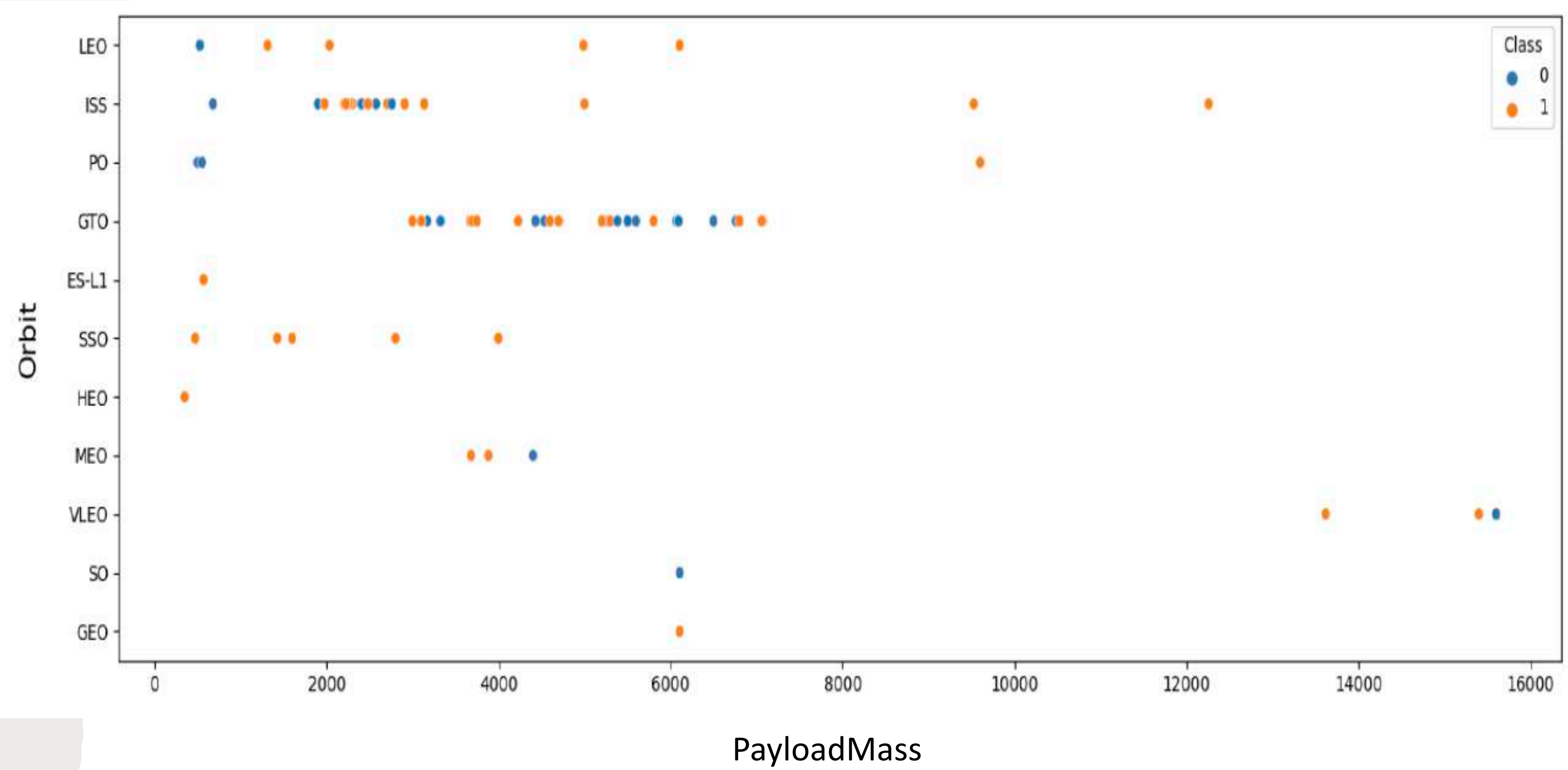
Payload vs Orbit Type

Correlation between GTO Orbit and Pay Load Mass is unclear. The weight of payloads can have a great influence on the success rate of the launches in orbits such as LEO orbit.



Payload vs Orbit Type

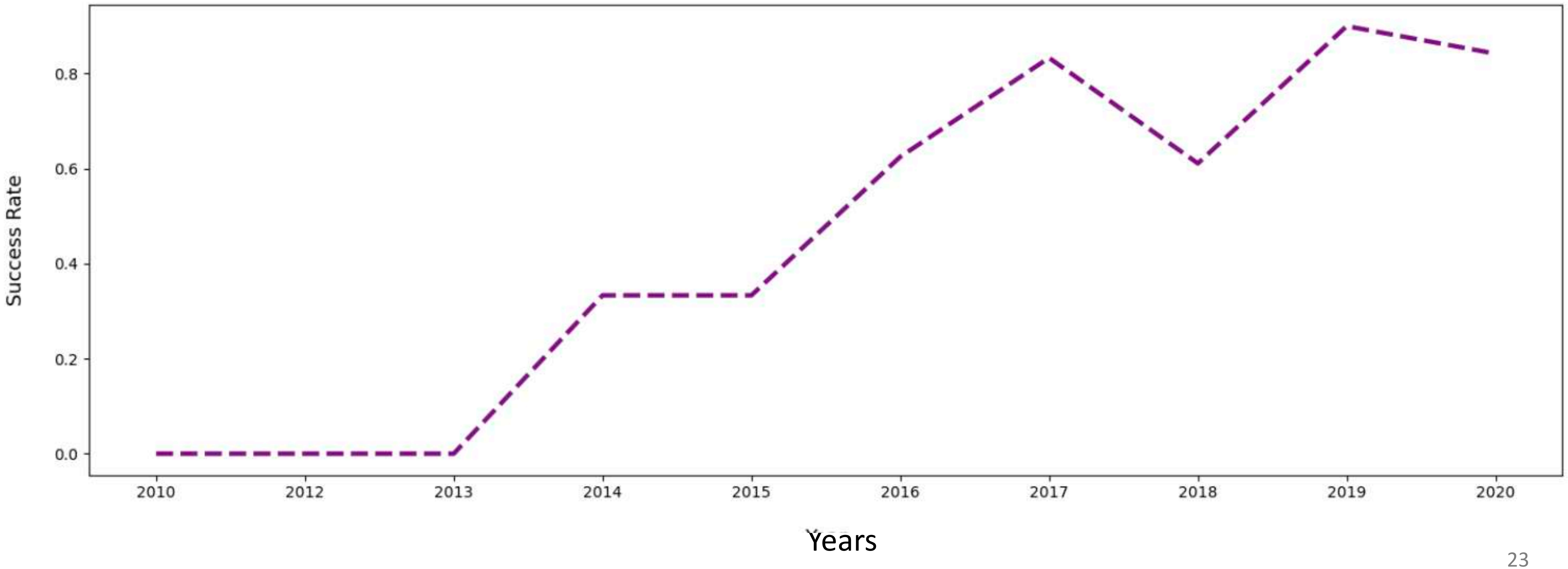
Correlation between GTO Orbit and Pay Load Mass is unclear. The weight of payloads can have a great influence on the success rate of the launches in orbits such as LEO orbit.



Launch Success Yearly Trend

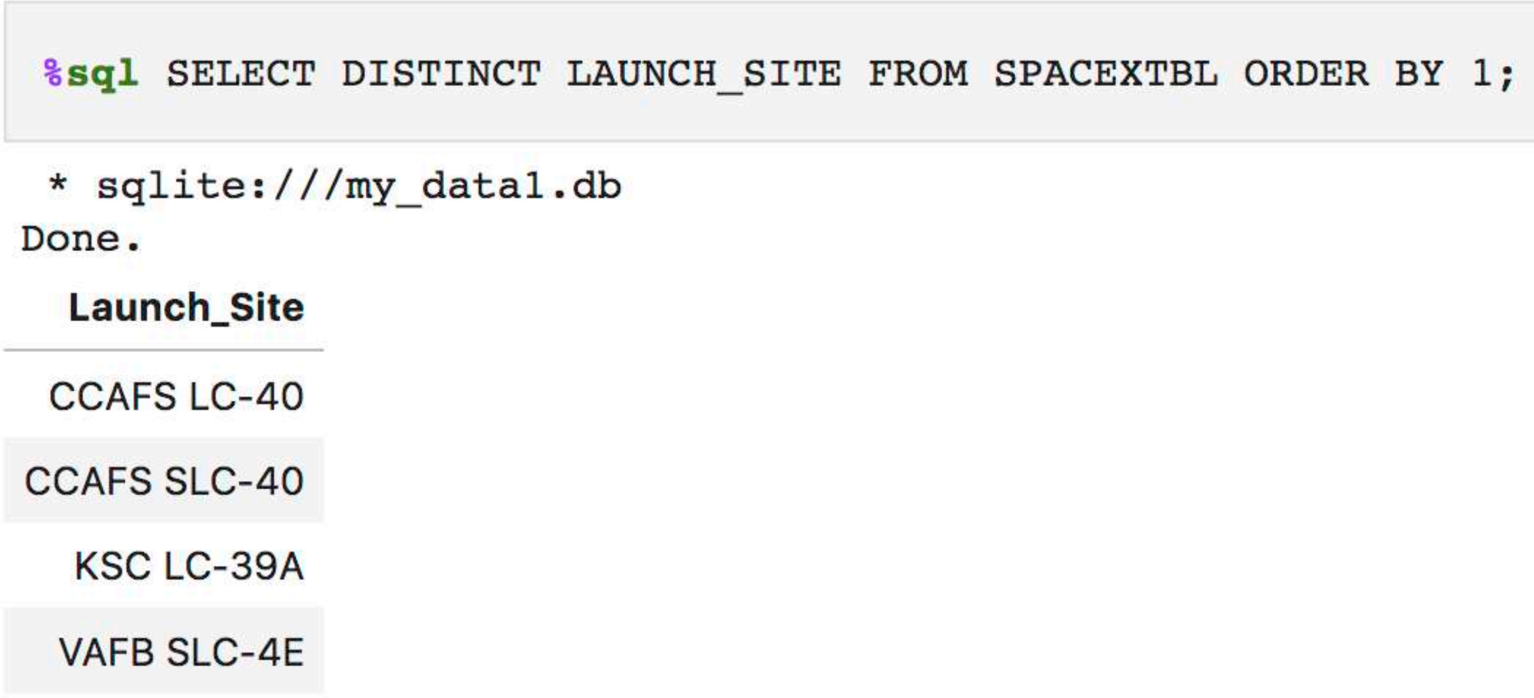
- Success generally increases over time since 2013 with a slight decrease in 2018

- Between 2010 and 2013, all landings were failed.



All Launch Site Names

- The task is for all Launch site names.



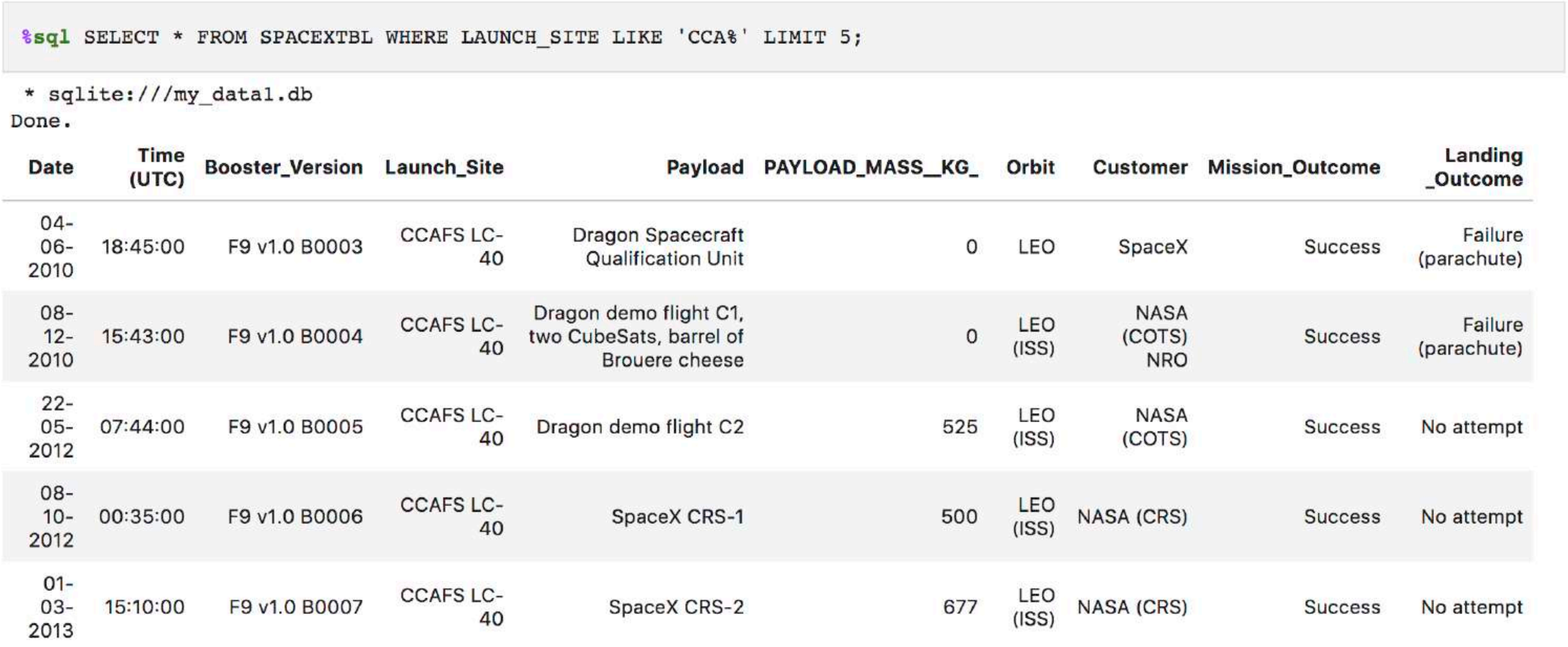
```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL ORDER BY 1;
```

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

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

Launch Site Names Begin with 'CCA'

- First five records in database with Launch Site name beginning with CCA.



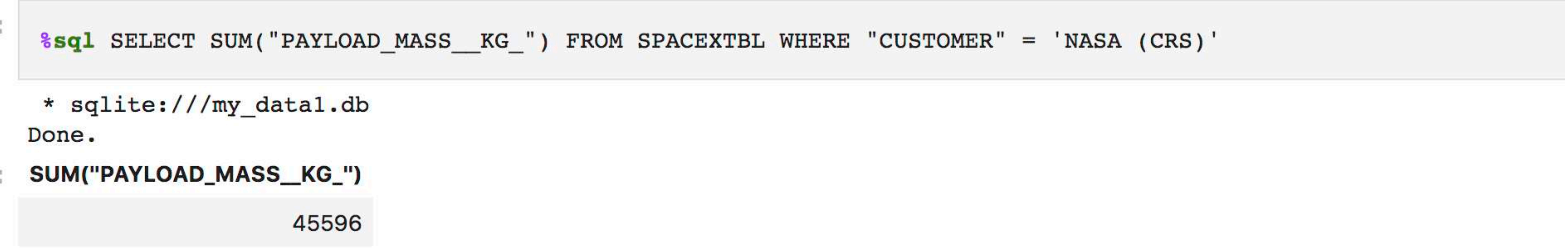
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

```
 * 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

- This query returns the sum of all payload masses.

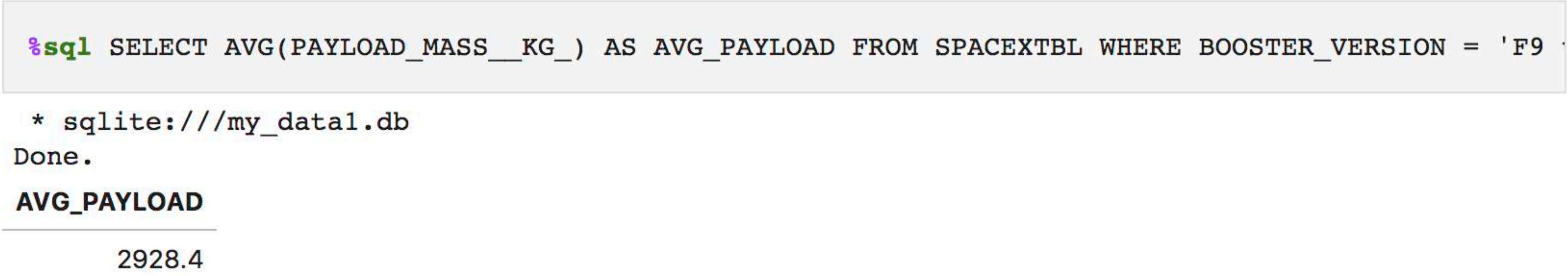


```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

```
 * sqlite:///my_data1.db
Done.
SUM("PAYLOAD_MASS__KG_")
                   45596
```

Average Payload Mass by F9 v1.1

- This query returns the average payload mass which used booster version F9 v1.1



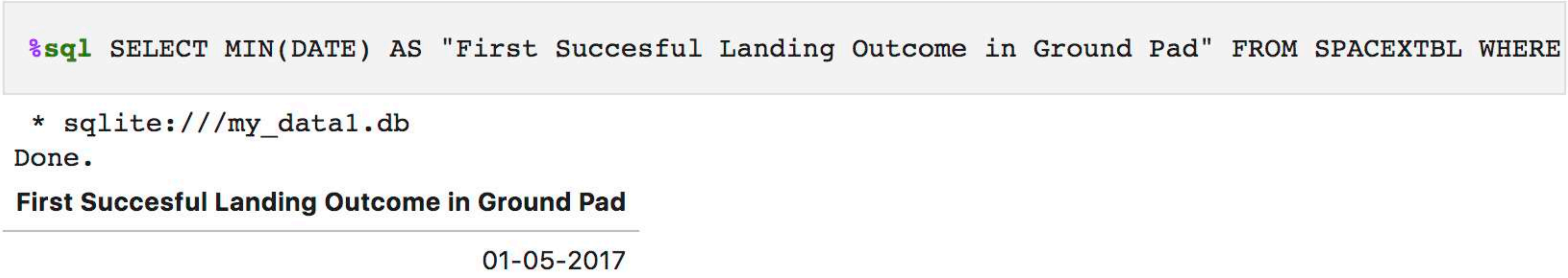
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9
```

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

AVG_PAYLOAD
2928.4

First Successful Ground Landing Date

- This query returns the first successful ground pad landing date.



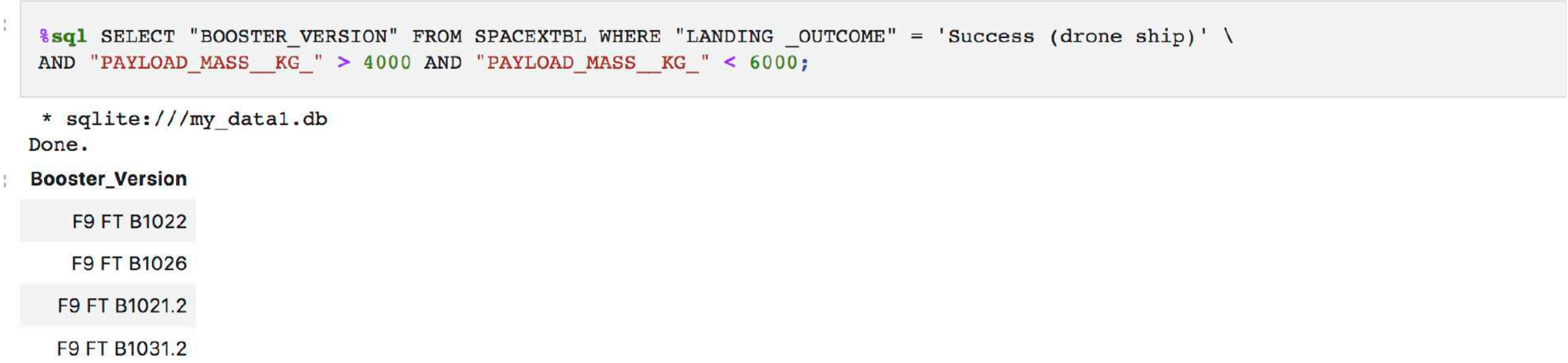
```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad" FROM SPACEXTBL WHERE
```

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

First Succesful Landing Outcome in Ground Pad
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

- This query returns four booster versions that had successful drone ship landings and a payload mass between 4000 and 6000 nonexclusively.



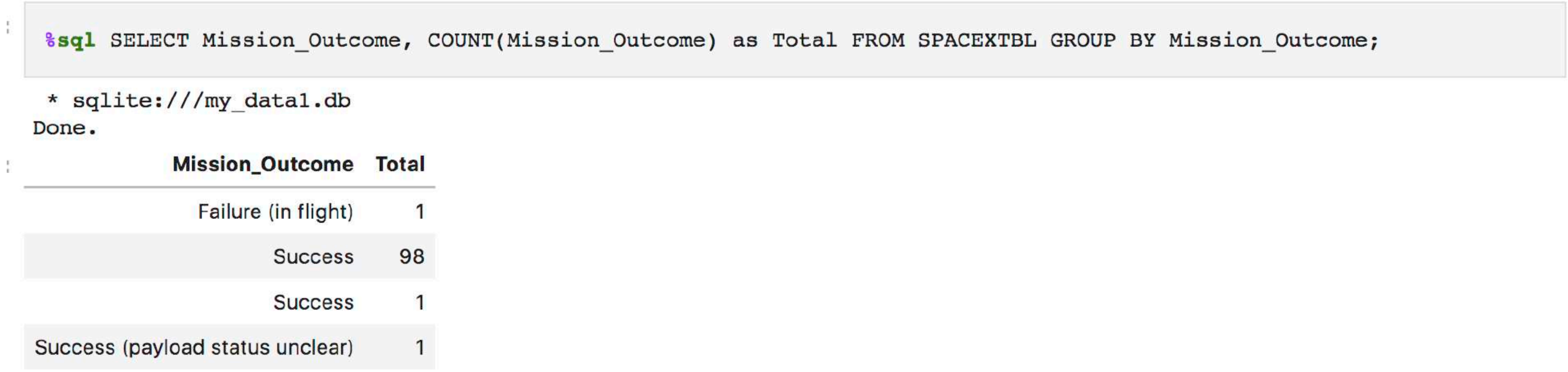
```
%sql SELECT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "LANDING _OUTCOME" = 'Success (drone ship)' \
AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000;
```

```
 * 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

- This query returns the count of each mission outcome.



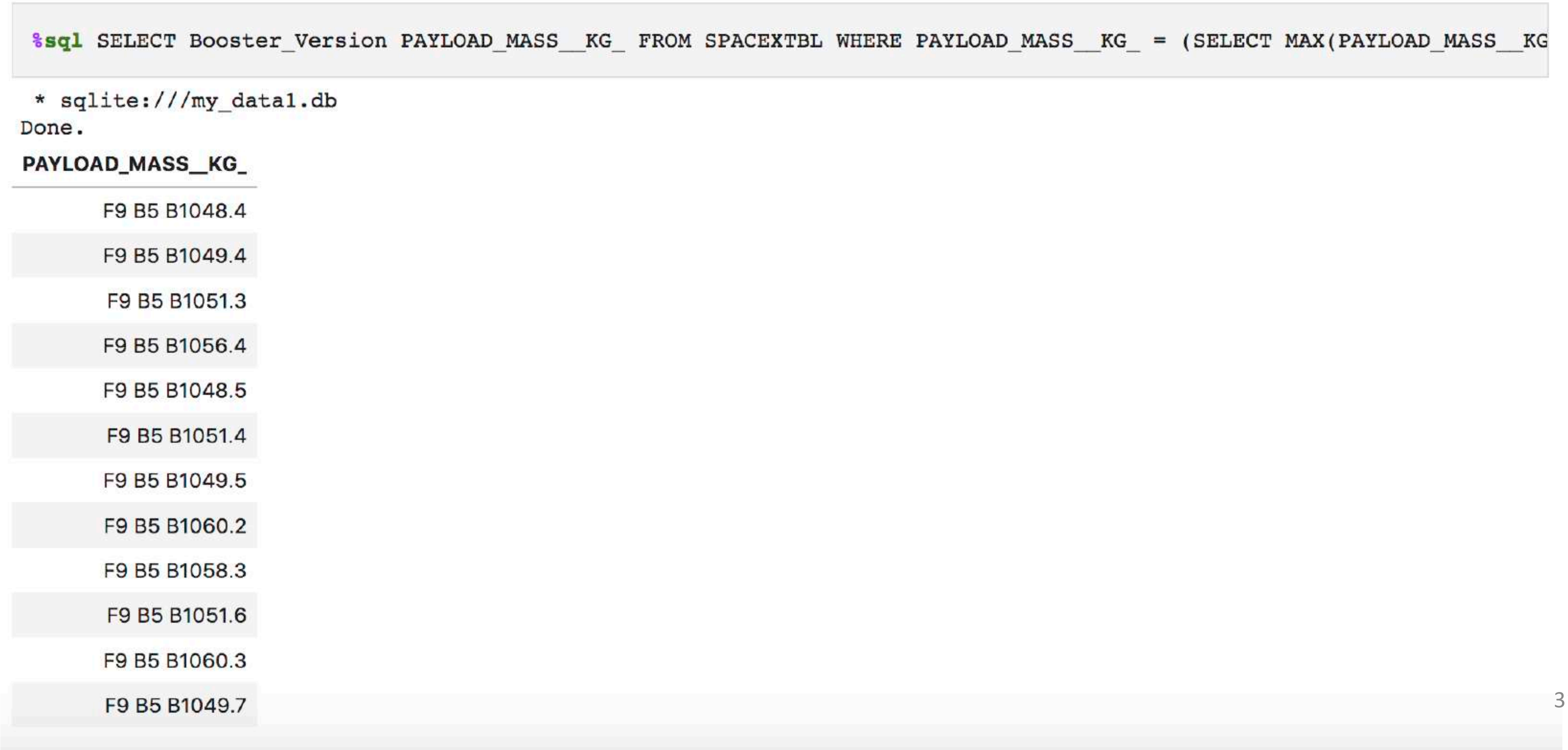
```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) as Total FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

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

Boosters Carried Maximum Payload

- This query returns the booster versions that carried the highest payload mass.



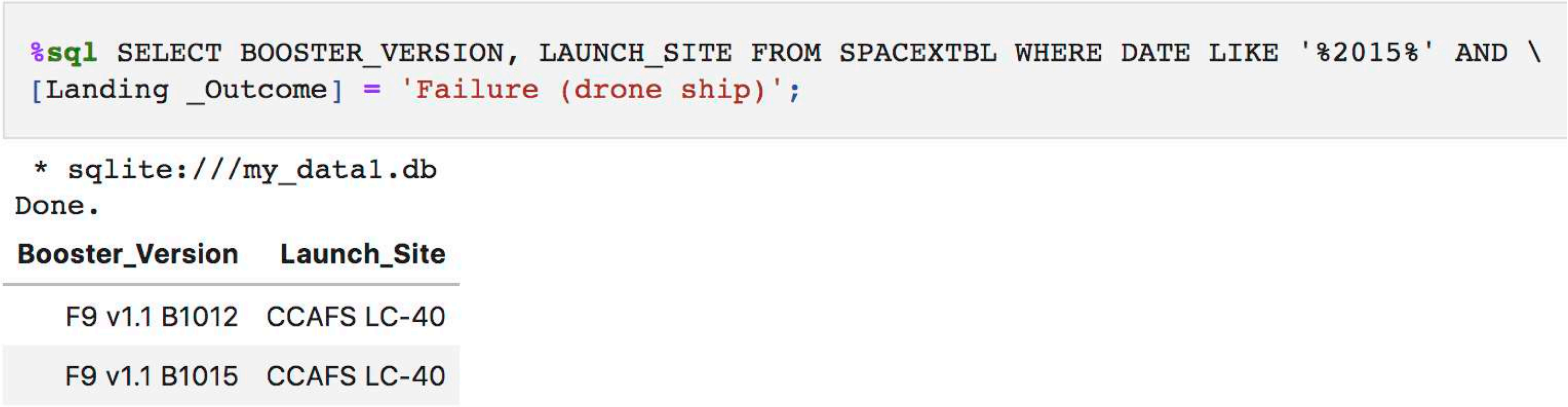
```
%sql SELECT Booster_Version PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG
```

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

PAYLOAD_MASS__KG_
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

- This query returns the booster version and launch site where landing was unsuccessful in 2015.



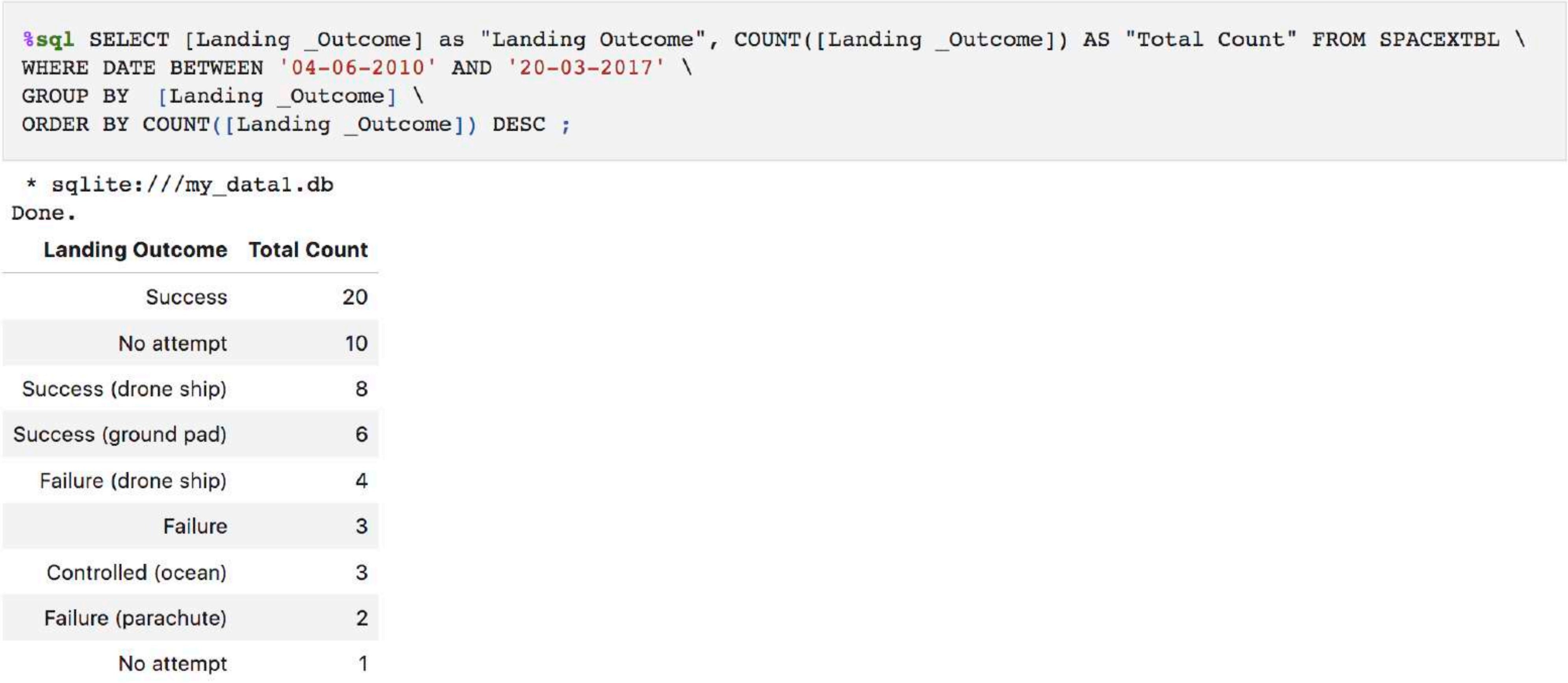
```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE DATE LIKE '%2015%' AND \
[Landing _Outcome] = 'Failure (drone ship)';
```

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

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

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

- This query returns every possible landing outcome with its total counts between 2010-06-04 and 2017-03-20 which is shown in decreasing order.



```
%sql SELECT [Landing _Outcome] as "Landing Outcome", COUNT([Landing _Outcome]) AS "Total Count" FROM SPACEXTBL \
WHERE DATE BETWEEN '04-06-2010' AND '20-03-2017' \
GROUP BY  [Landing _Outcome] \
ORDER BY COUNT([Landing _Outcome]) DESC ;
```

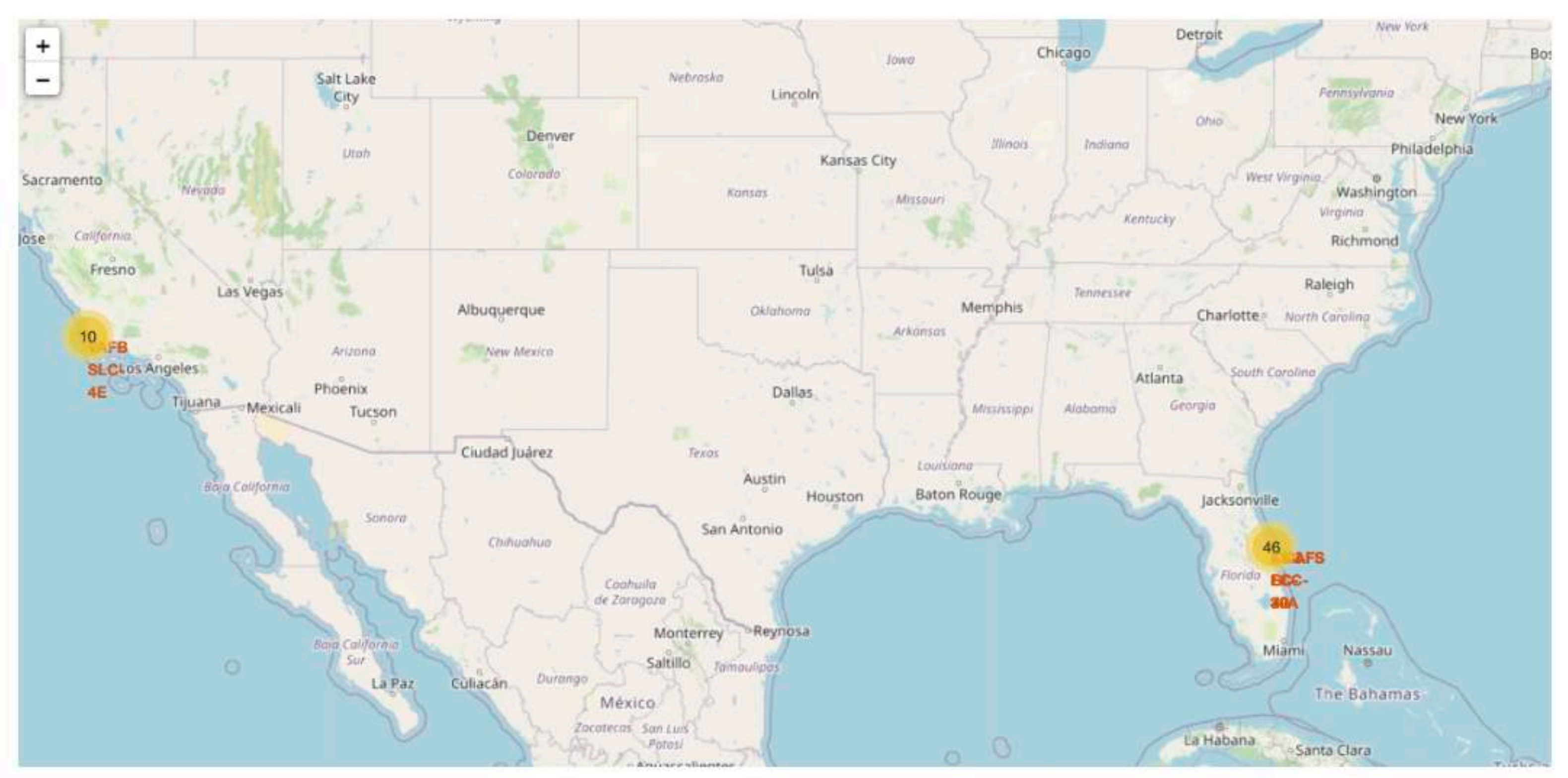
```
 * sqlite:///my_data1.db
Done.
```

Landing Outcome	Total 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

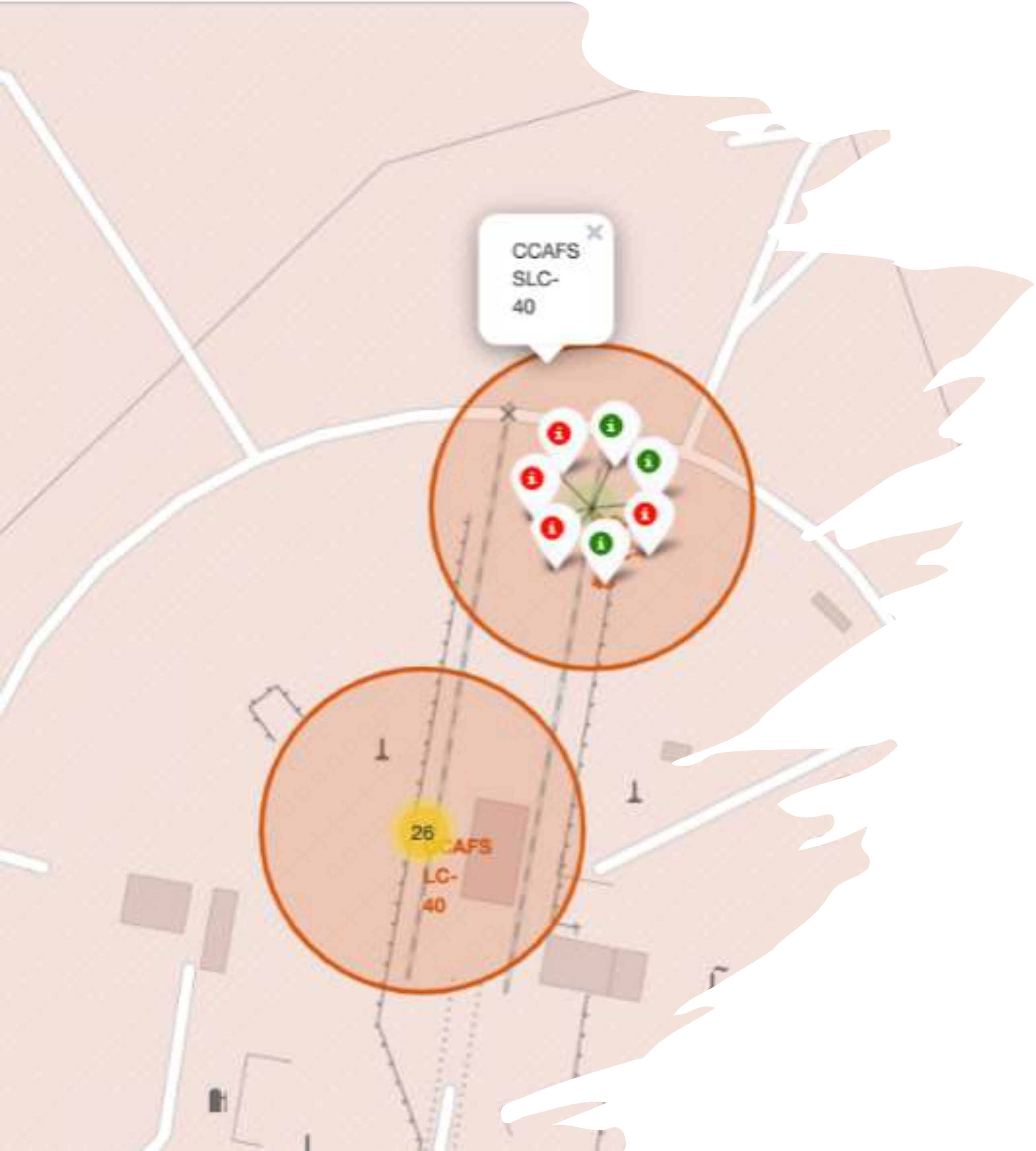
Section 3

Launch Sites Proximities Analysis

Launch Site locations

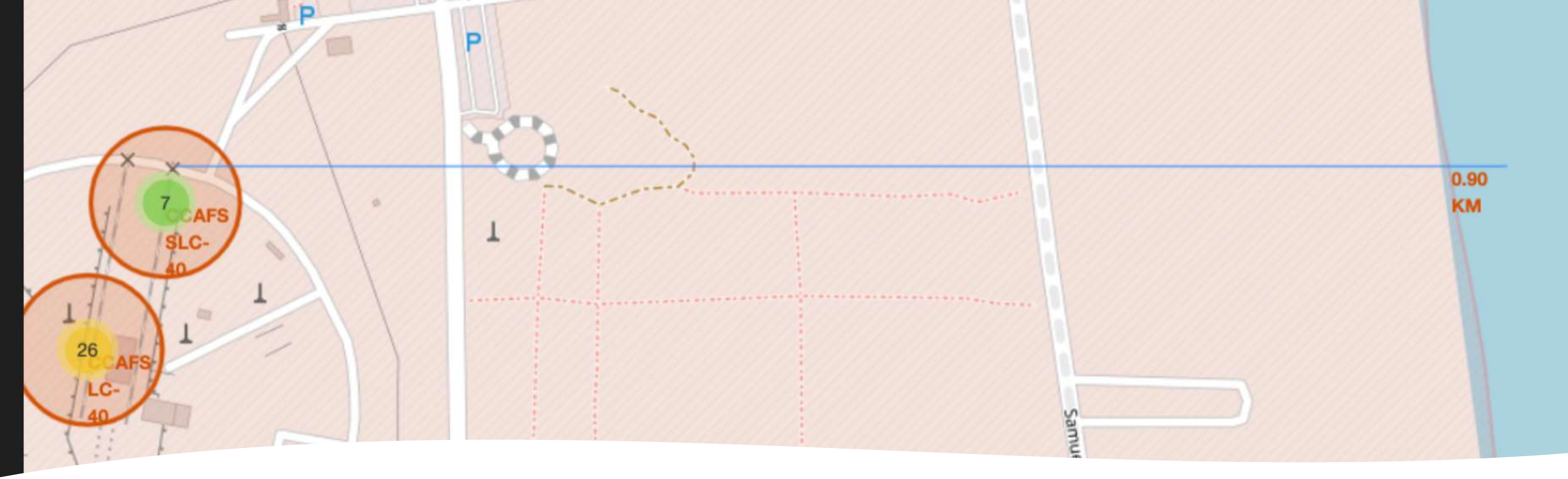


According to the figure, the spaceX launch sites are located on the coast of USA.



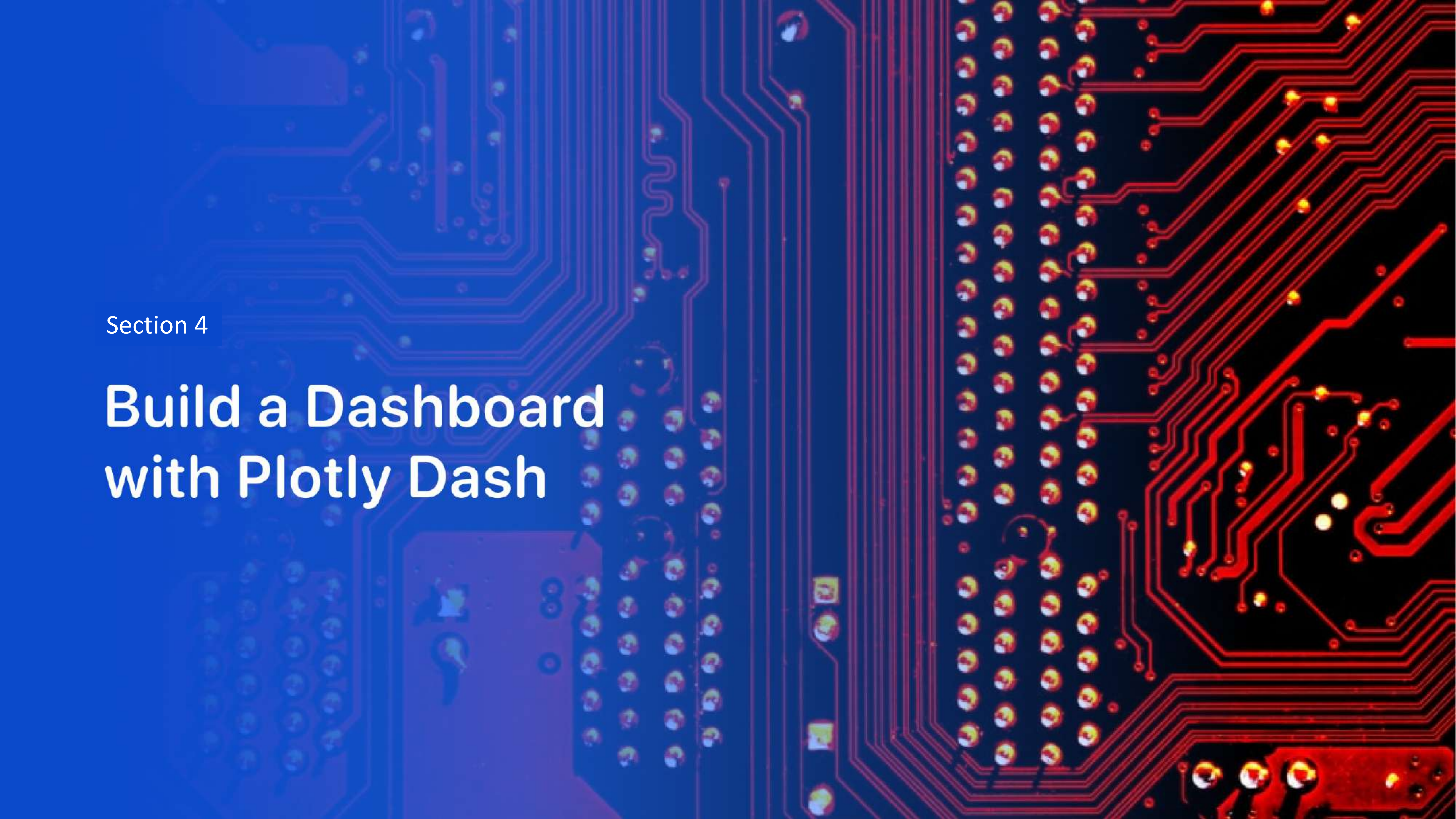
Launch markers

- Launches have been grouped into clusters, the green icons stand for successful launches and red for unsuccessful launches.



Key Location Distance

- According to figure, the launch site is very close the water, incase something goes wrong with the launch. It could be noted that the launch site is considerable distance from any densely populated area.
- The blue line represents the distance from launch area to any location of choice.

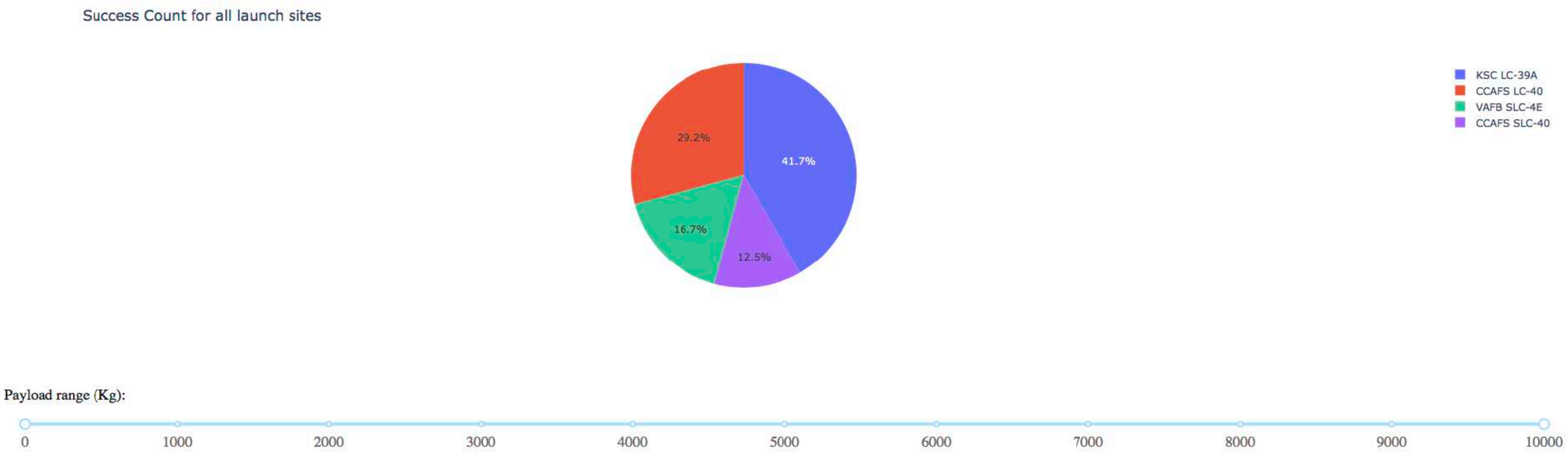


Section 4

Build a Dashboard with Plotly Dash

<Dashboard for Total Success Sites>

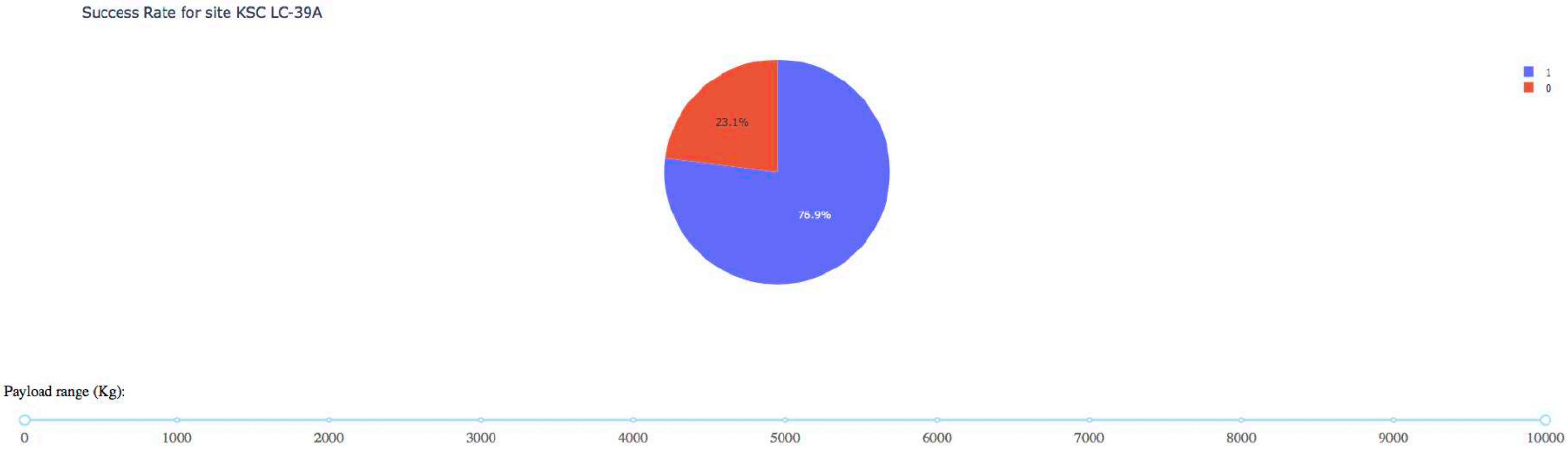
According to the figure, KSC LC-39A has the larges success rate of all sites, while CCAFS SLC-40 has the least.



Success Count for all launch sites

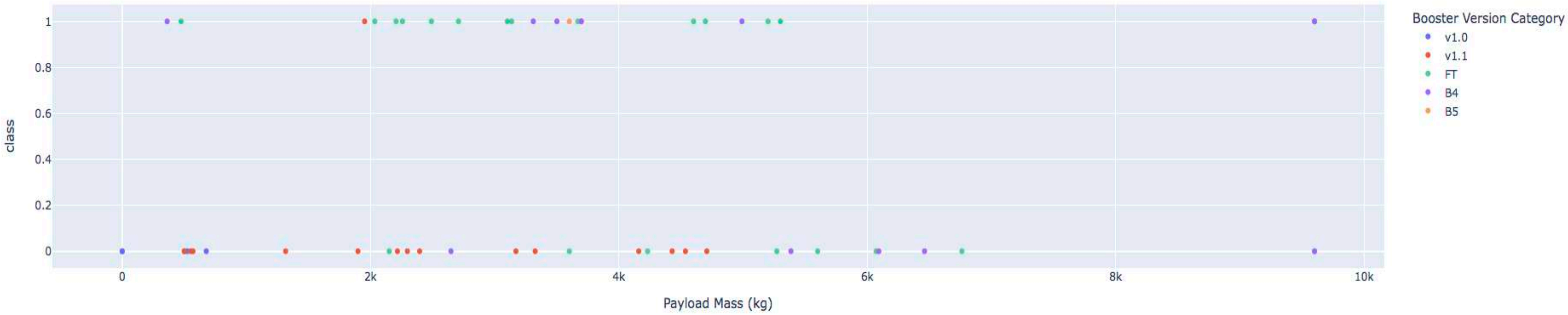
<Dashboard-Launch Site of KSC LC-39A>

- **KSC LC-39A reached 76.9% success rate but still has 23.1% failure rate.**



<Dashboard-Payload vs Launch Outcome for All Sites>

- **According to the figure, low weighted payloads have a better success rate than the heavier ones.**

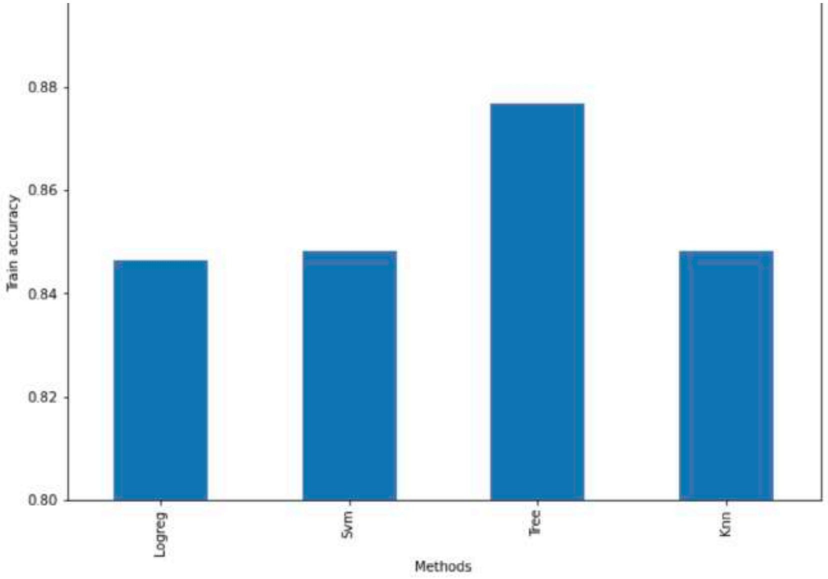


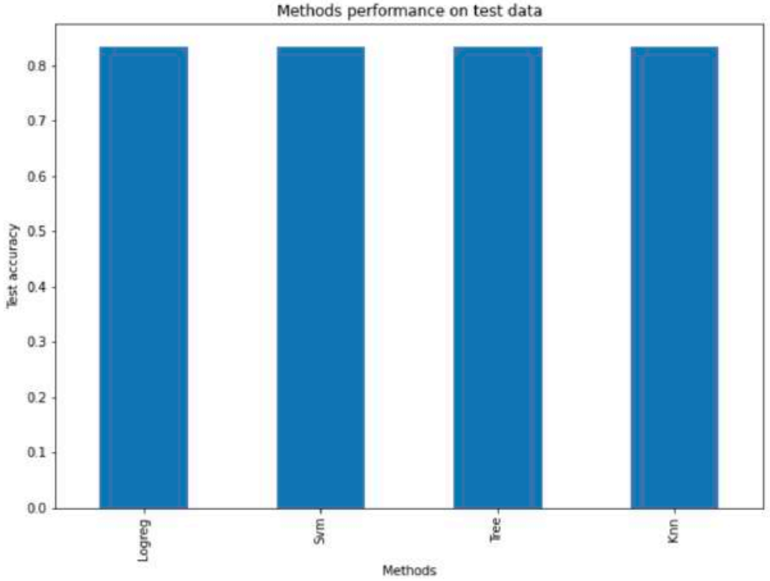
Section 5

Predictive Analysis (Classification)

Classification Accuracy

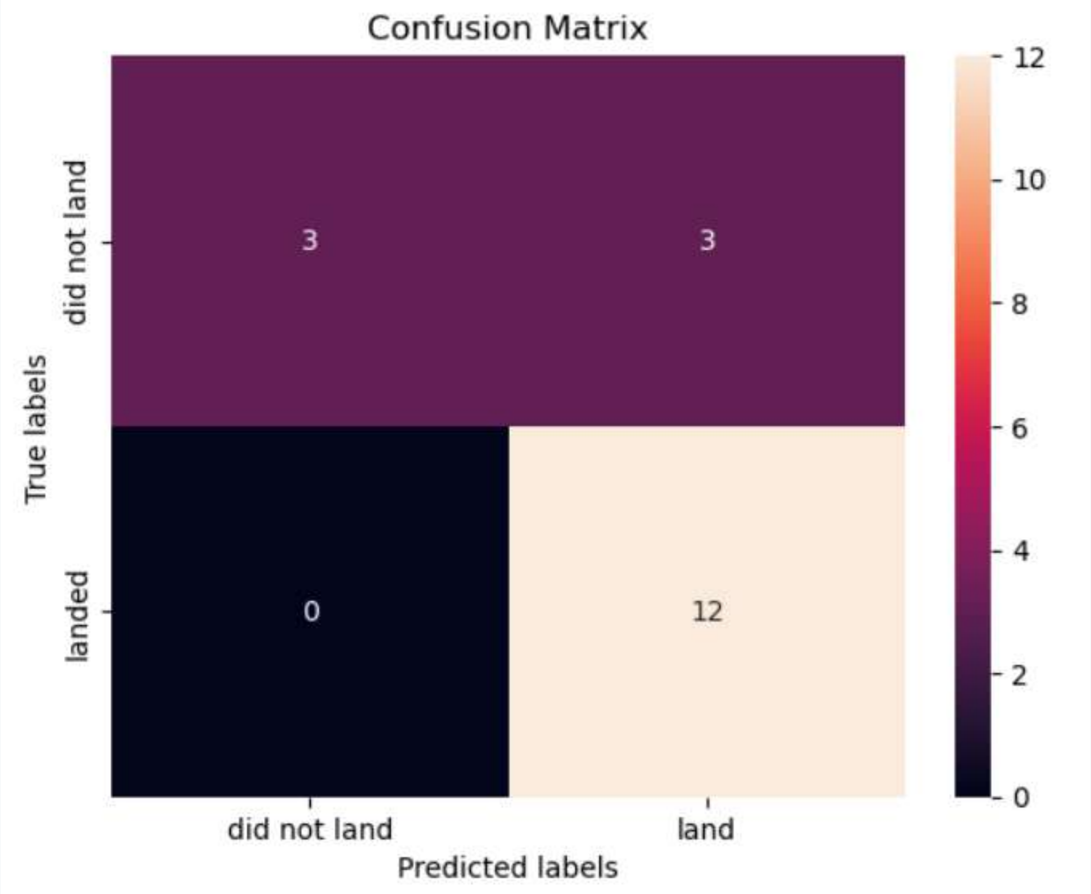
- From the performance on train data, the decision tree reached the highest accuracy among all the methods.
- From the performance of test data, there no big difference among these methods





Confusion Matrix

- Since the accuracy of these four methods are the same based on the performance from the test data, the confusion matrix are also identical.



Conclusions

- The success rate of missions is affected by many factors, including the launch sites, the orbit and its previous launches.
- The types of orbits reaching the highest success rates are GEO, HEO, SSO and ES-L1.
- The payload mass can be considered to decide the success rate of a mission. In general, low weighted payloads lead to better success rate than the heavy ones.
- Further research needs to be conducted to explain why KSC LC-39A performed as the best sites.
- In this project, Decision tree is selected as the best model since it has the best accuracy in train data although all of the methods share the same test accuracy.

Thank you!