IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

<Name>
<Date>

# Outline

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

# 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

SpaceX is a key company in the space industry, notably by offering rocket launches (specifically Falcon 9) for a cost as low as 62 million dollars, while other companies have a cost of around 165 million dollar for launch. Most of this saving is thanks o the innovative idea of reusing the first stage of the launch by re-landing the rocket and using it on other missions. If possible, the more the process is repeated, more the price goes down.

As a data scientist of a startup rivaling SpaceX, the goal of this project is to create the machine learning pipeline to predict the landing outcome of the first stage for future missions.

Mainly, the biggest challenges are:

- Discovering what variables have an influence on the landing outcome.
- Finding out the relationships between variables and how they affect the landing outcome.
- Given that, finding out the best combinations of variables to increase the chance of success for future missions.

Section 1

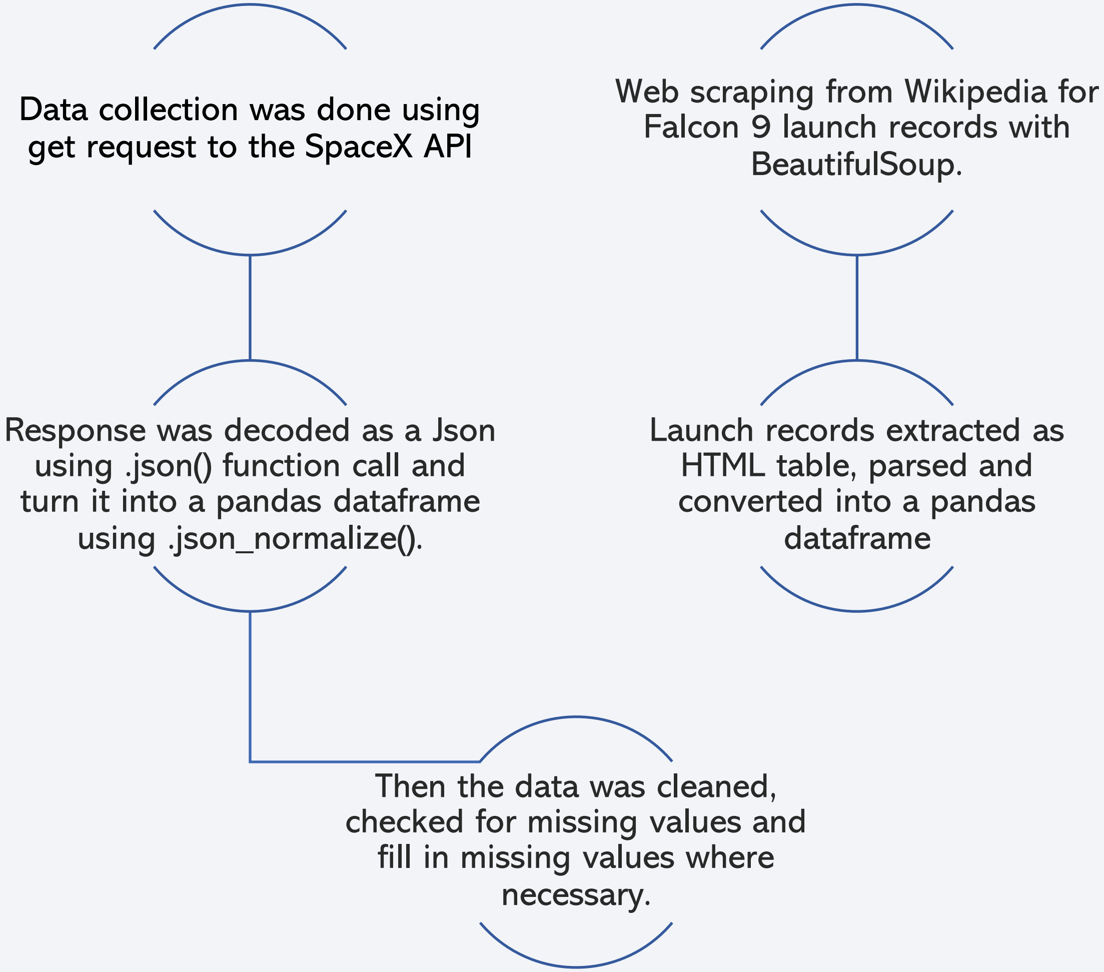
# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - Using SpaceX's own API and web scraping from a table on Wikipedia.
- Perform data wrangling
  - One-hot encoding for all categorical values
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

# Data Collection



Data collection was done using get request to the SpaceX API

Response was decoded as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().

Then the data was cleaned, checked for missing values and fill in missing values where necessary.

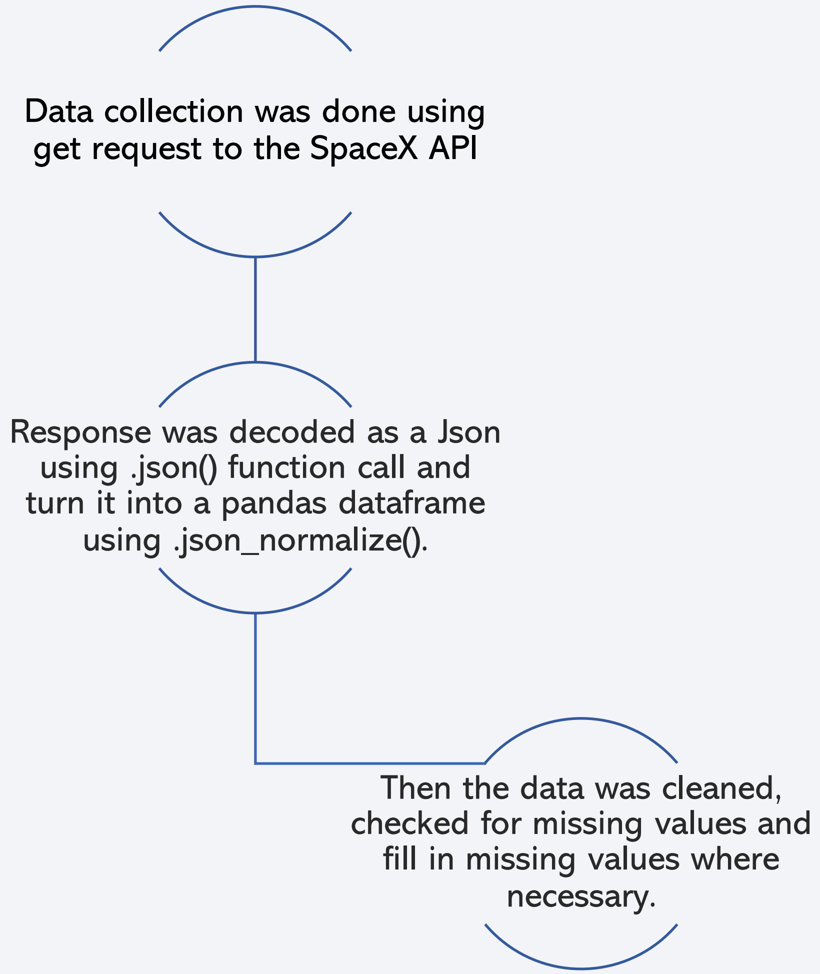
Web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.

Launch records extracted as HTML table, parsed and converted into a pandas dataframe

# Data Collection – SpaceX API

- Collect data
- Transform it to dataframe
- Cleaning and formatting

- https://github.com/rafandrzejewski/Falcon-9---Applied-Data-Science-Capstone/blob/main/notebook%20-%20Data%20Collection%20API.ipynb m



Data collection was done using get request to the SpaceX API

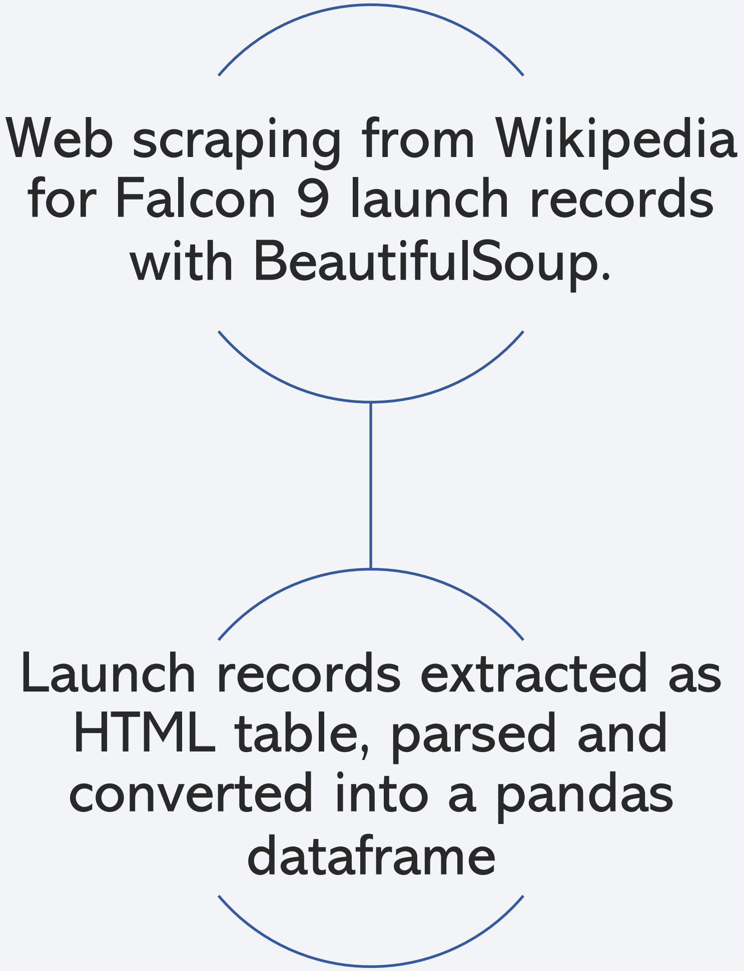
Response was decoded as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().

Then the data was cleaned, checked for missing values and fill in missing values where necessary.

# Data Collection - Scraping

- BeautifulSoup request to acess the table.
- Parsing and converting to dataframe

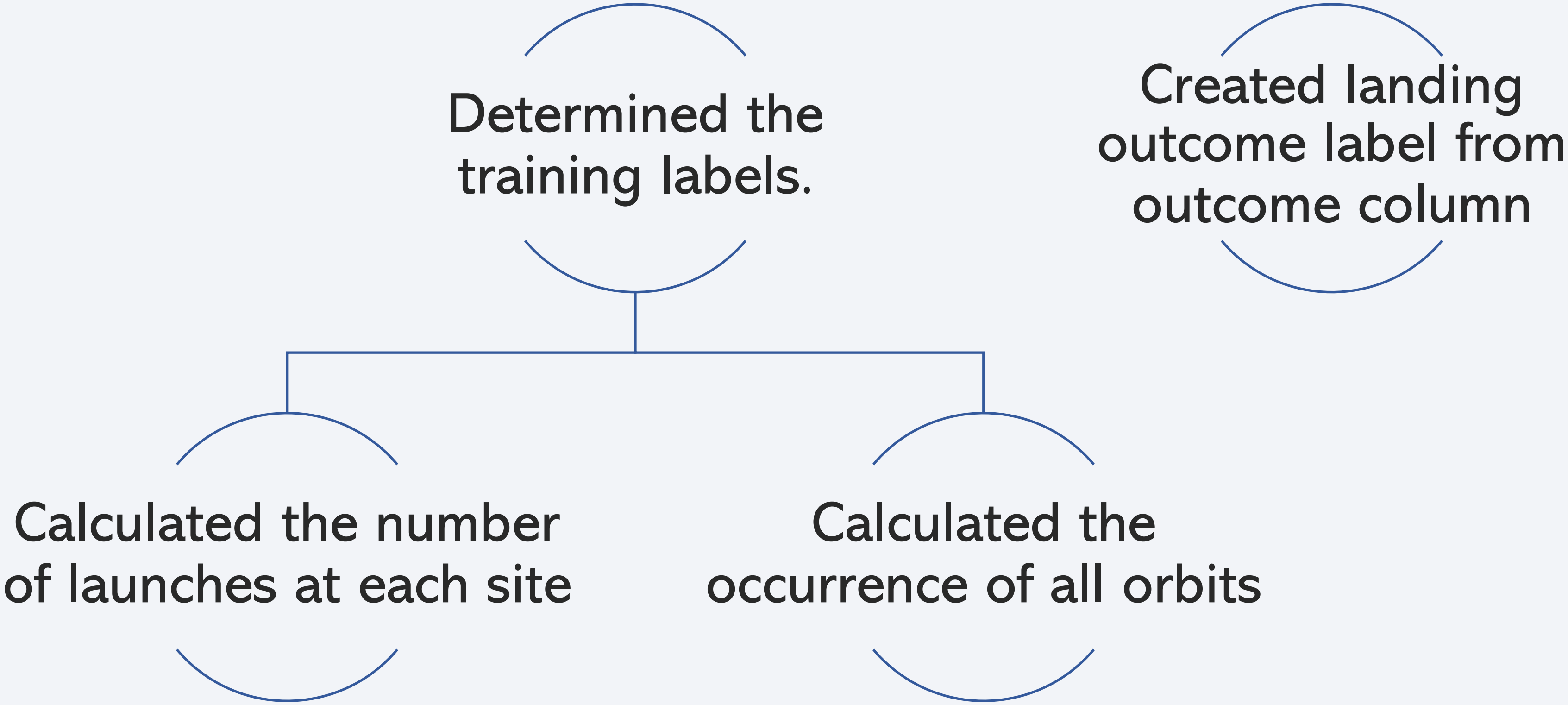
- https://github.com/rafandrzejewski/Falcon-9---Applied-Data-Science-Capstone/blob/main/notebook%20-%20Data%20Collection%20with%20Web%20Scraping.ipynb



Web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.

Launch records extracted as HTML table, parsed and converted into a pandas dataframe

# Data Wrangling



Determined the training labels.

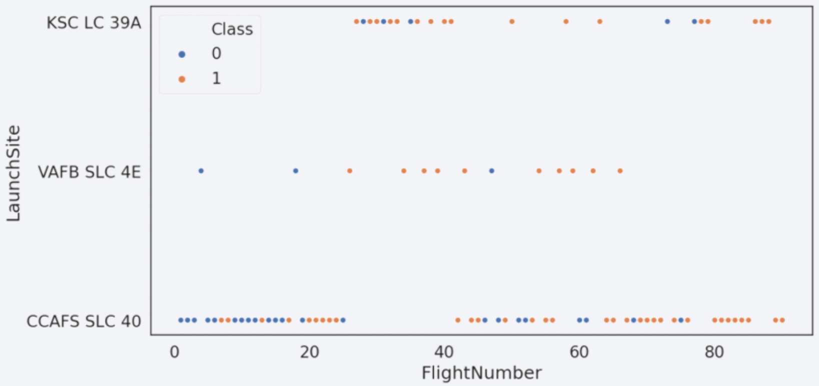
Created landing outcome label from outcome column

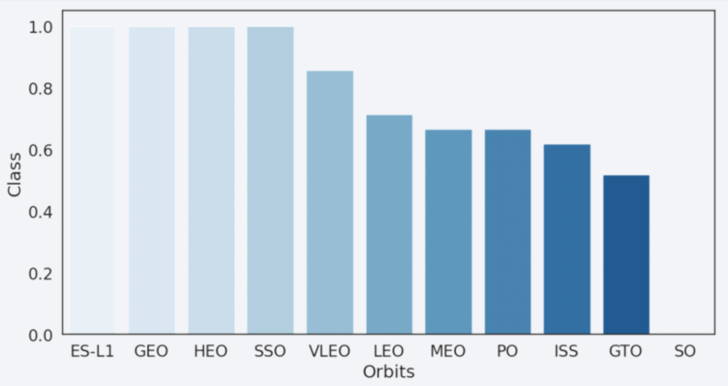
Calculated the number of launches at each site

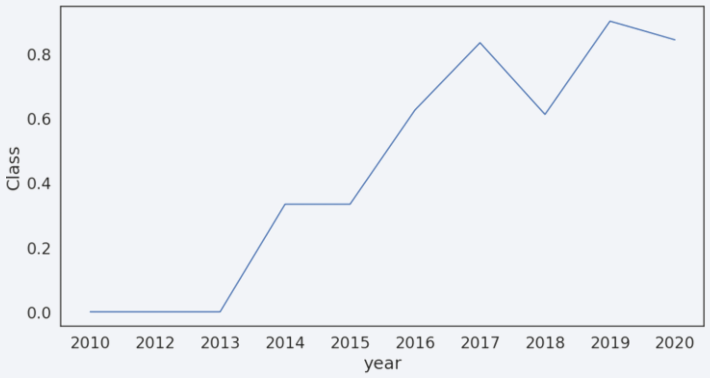
Calculated the occurrence of all orbits

- https://github.com/rafandrzejewski/Falcon-9---Applied-Data-Science-Capstone/blob/main/notebook%20-%20Data%20Wrangling.ipynb

# EDA with Data Visualization







## Scatter plots

To find the relationship between pairs of atributes

Payload vs. Launch Site

Payload vs. Orbit Type

## Bar plots

Determine which orbits have the highest probability of success

## Line plots

Evaluate trends, for example success ratio over time.

- https://github.com/rafandrzejewski/Falcon-9---Applied-Data-Science-Capstone/blob/main/notebook%20-%20EDA%20with%20Visualization.ipynb

# EDA with SQL

## Queries performed:

- Names of unique launch sites in the space mission.
- Total payload mass carried by boosters launched by NASA (CRS)
- Average payload mass carried by booster version F9 v1.1
- Total number of successful and failure mission outcomes
- Failed landing outcomes in drone ship, their booster version and launch site names
- Names of the boosters which have success in drone ship and have payload massgreater than 4000 but less than 6000.

- https://github.com/rafandrzejewski/Falcon-9---Applied-Data-Science-Capstone/blob/main/notebook%20-%20EDA%20with%20SQL.ipynb

# Build an Interactive Map with Folium

## Locate Launch sites on the map

With latitude and longitude coordinates at each launch site and added a circle marker around each launch site with a label of the name of the launch site;

## Visualize launch outcomes per site

Red and Green markers were added on the map with MarkerCluster(), representing failed and success launches.

## Calculate the distance of the launch sites to various landmark

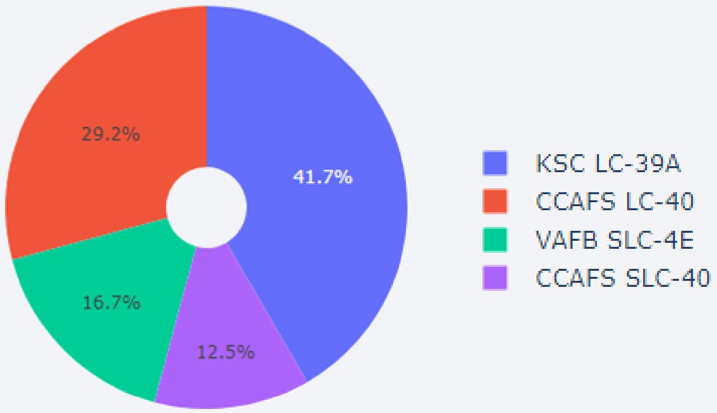
Lines were added to mark the distance between launch sites and important landmarks

- https://github.com/rafandrzejewski/Falcon-9---Applied-Data-Science Capstone/blob/main/notebook%20%20Interactive%20Visual%20Analytics%20with %20Folium.ipynb

# Build a Dashboard with Plotly Dash

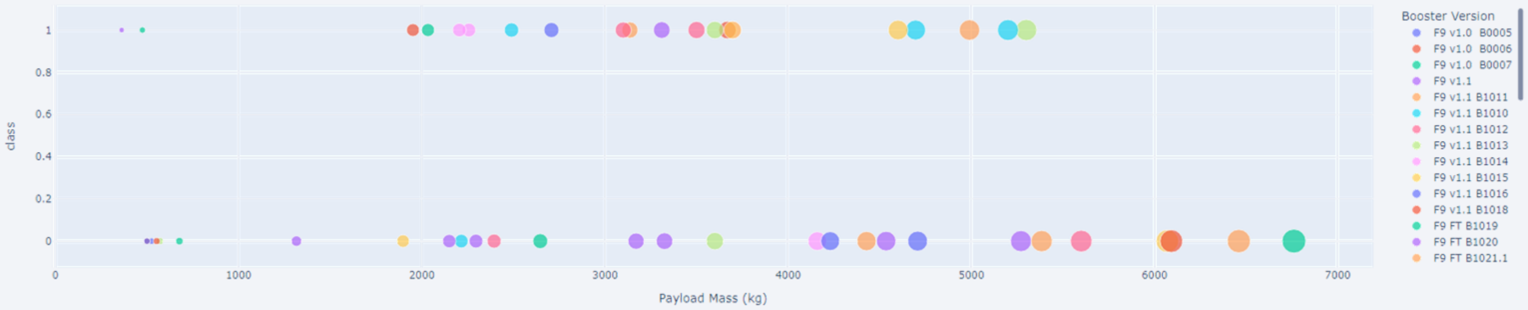
## Pie Charts

Visualize total launches by all or individual launch sites.



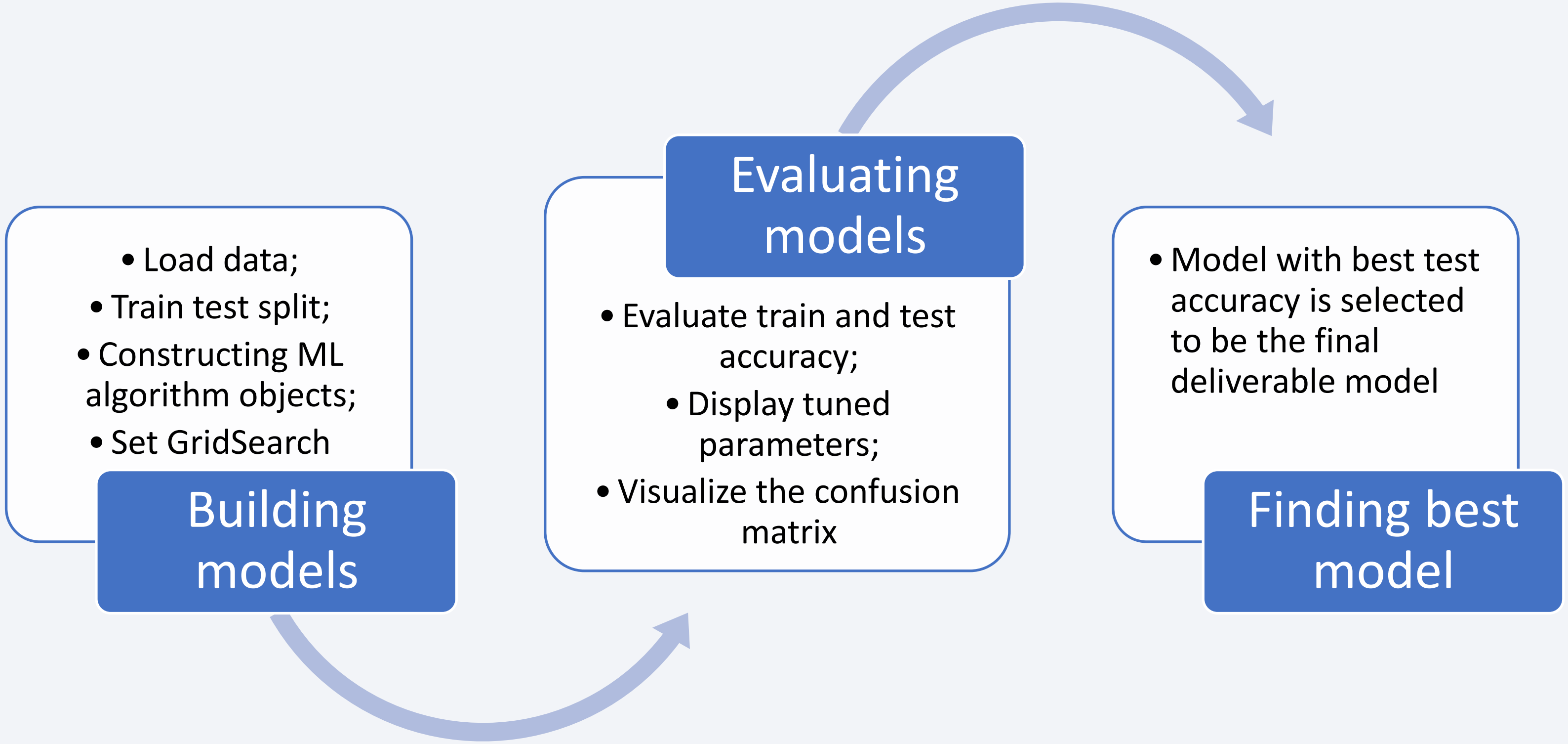
## Scatter plots

Visualize relationship between Outcome and Payload Mass (Kg) for the different booster version.



- https://github.com/rafandrzejewski/Falcon-9---Applied-Data-Science-Capstone/blob/main/spacex_dash_app_edited.py

# Predictive Analysis (Classification)



Building models

- Load data;
- Train test split;
- Constructing ML algorithm objects;
- Set GridSearch

Evaluating models

- Evaluate train and test accuracy;
- Display tuned parameters;
- Visualize the confusion matrix

Finding best model

- Model with best test accuracy is selected to be the final deliverable model

- https://github.com/rafandrzejewski/Falcon-9---Applied-Data-Science-Capstone/blob/main/notebook%20-%20Machine%20Learning%20Prediction.ipynb

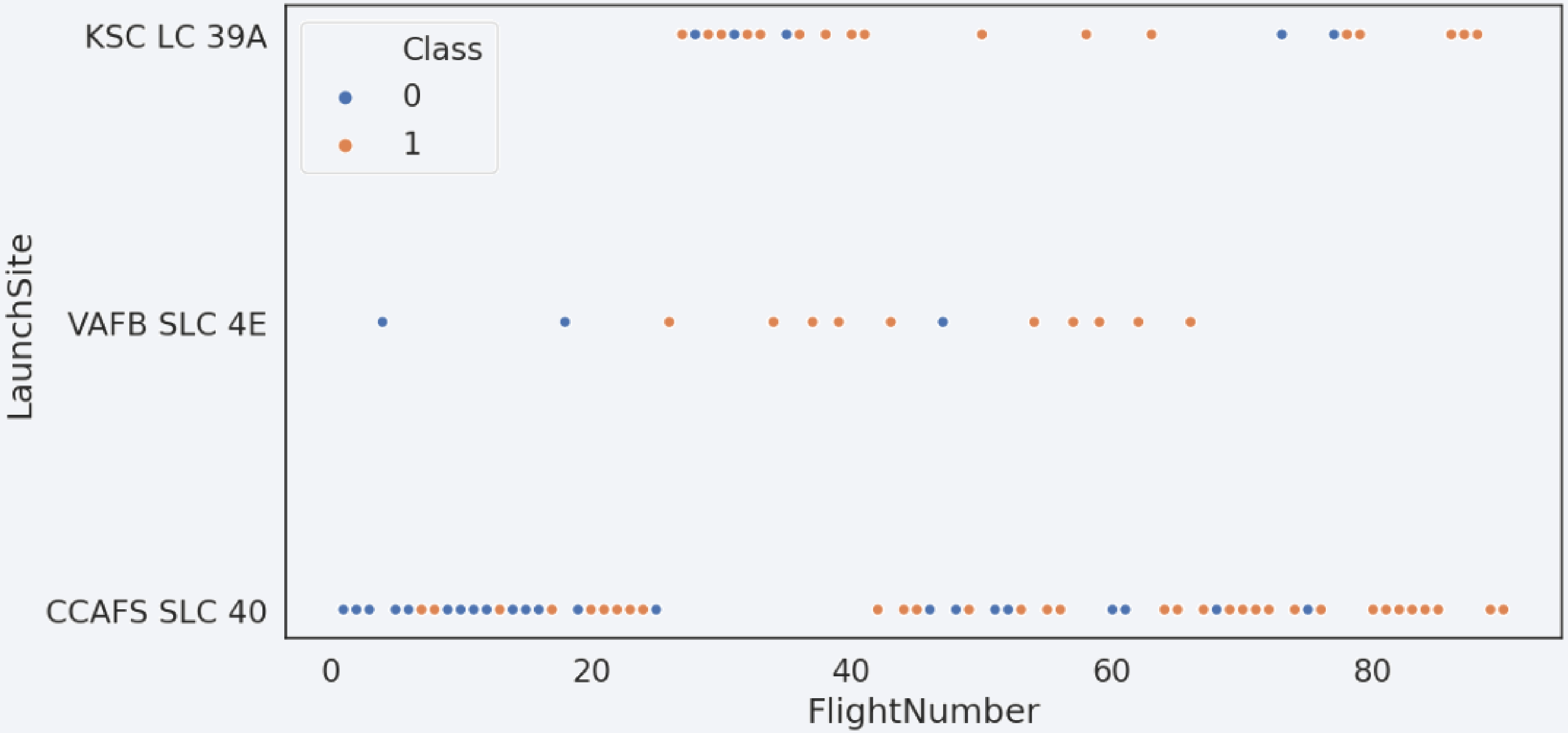
# Results

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

Section 2

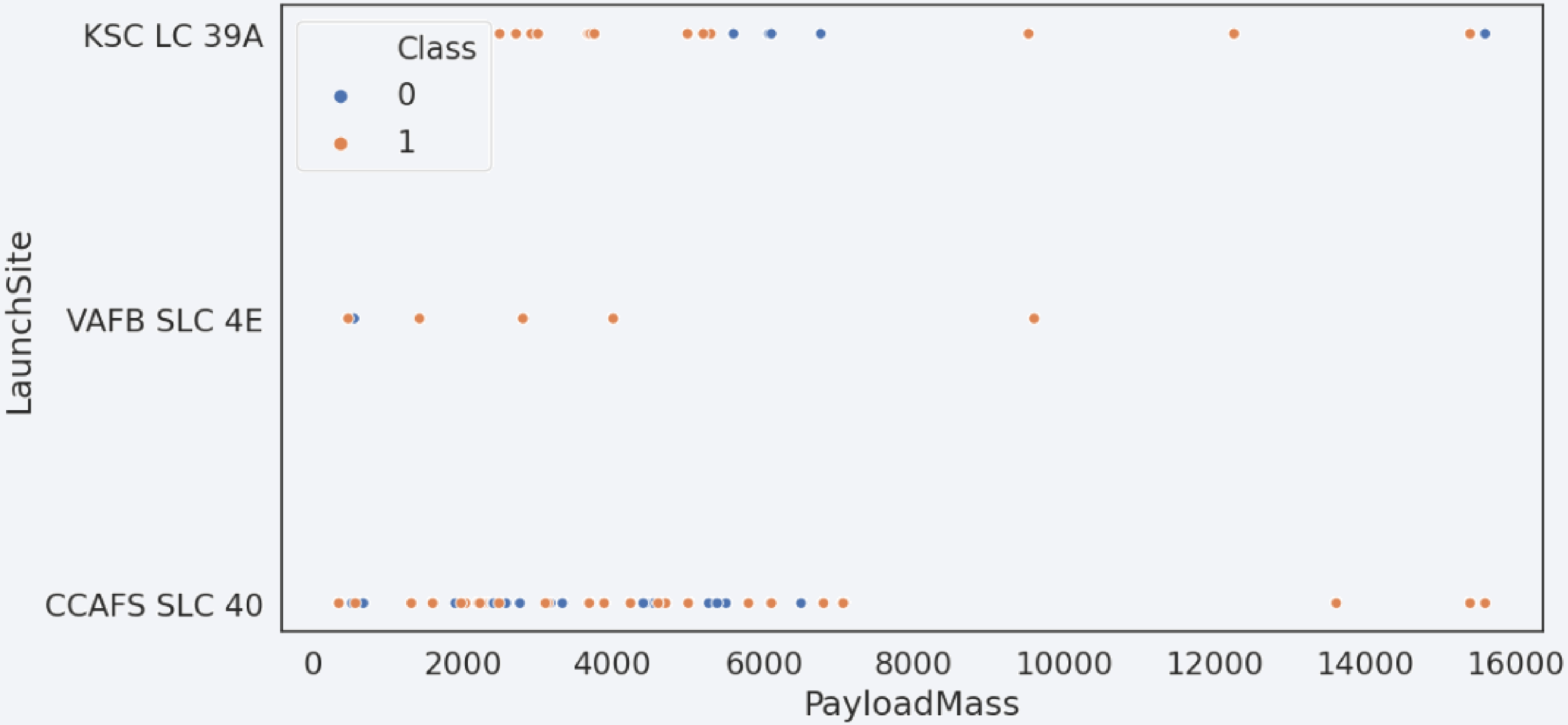
# Insights drawn from EDA

# Flight Number vs. Launch Site



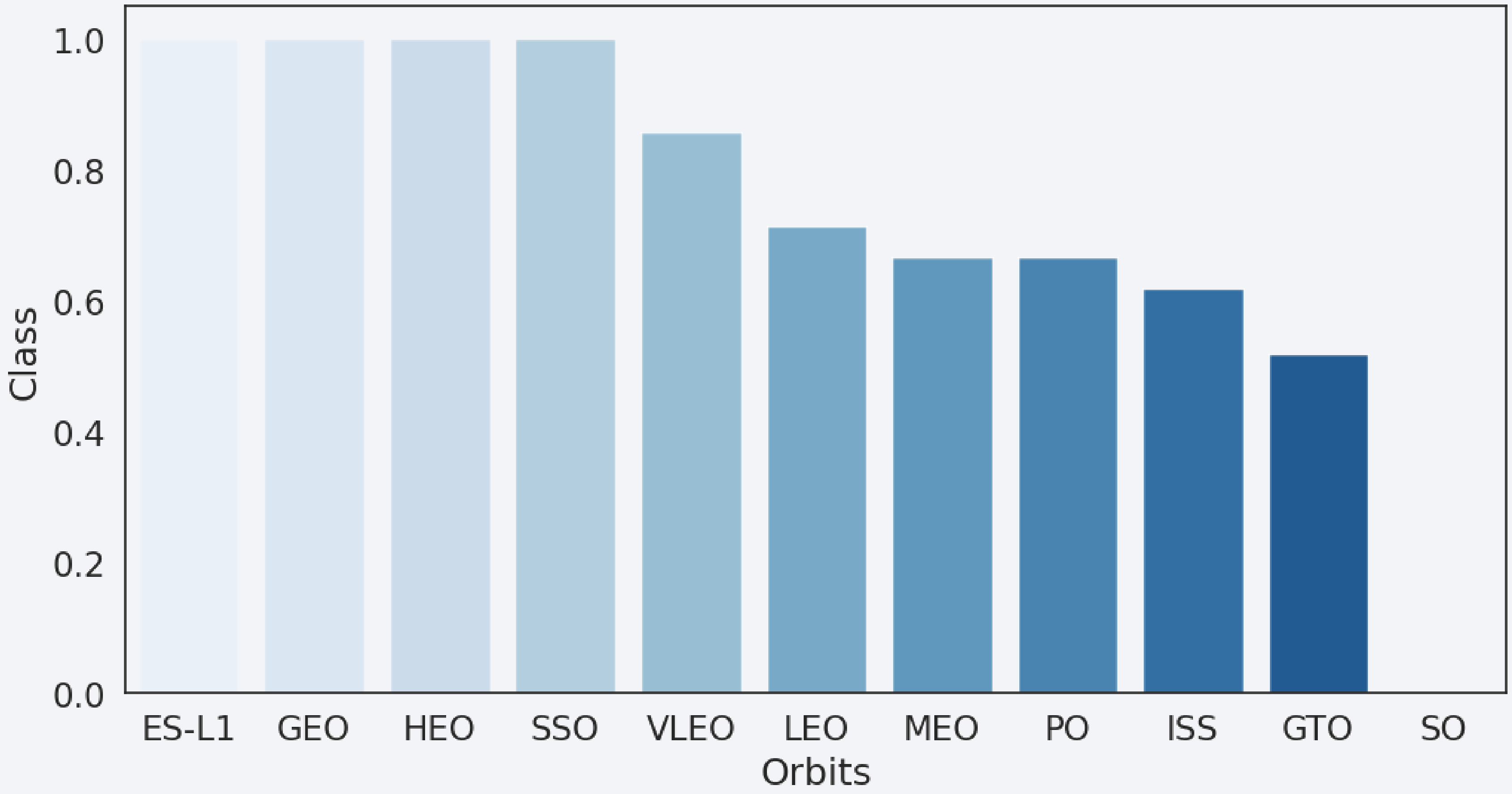
It looks like with more flights there's an increase in the success rate, but it's not a clean trend in the displayed data.

# Payload vs. Launch Site



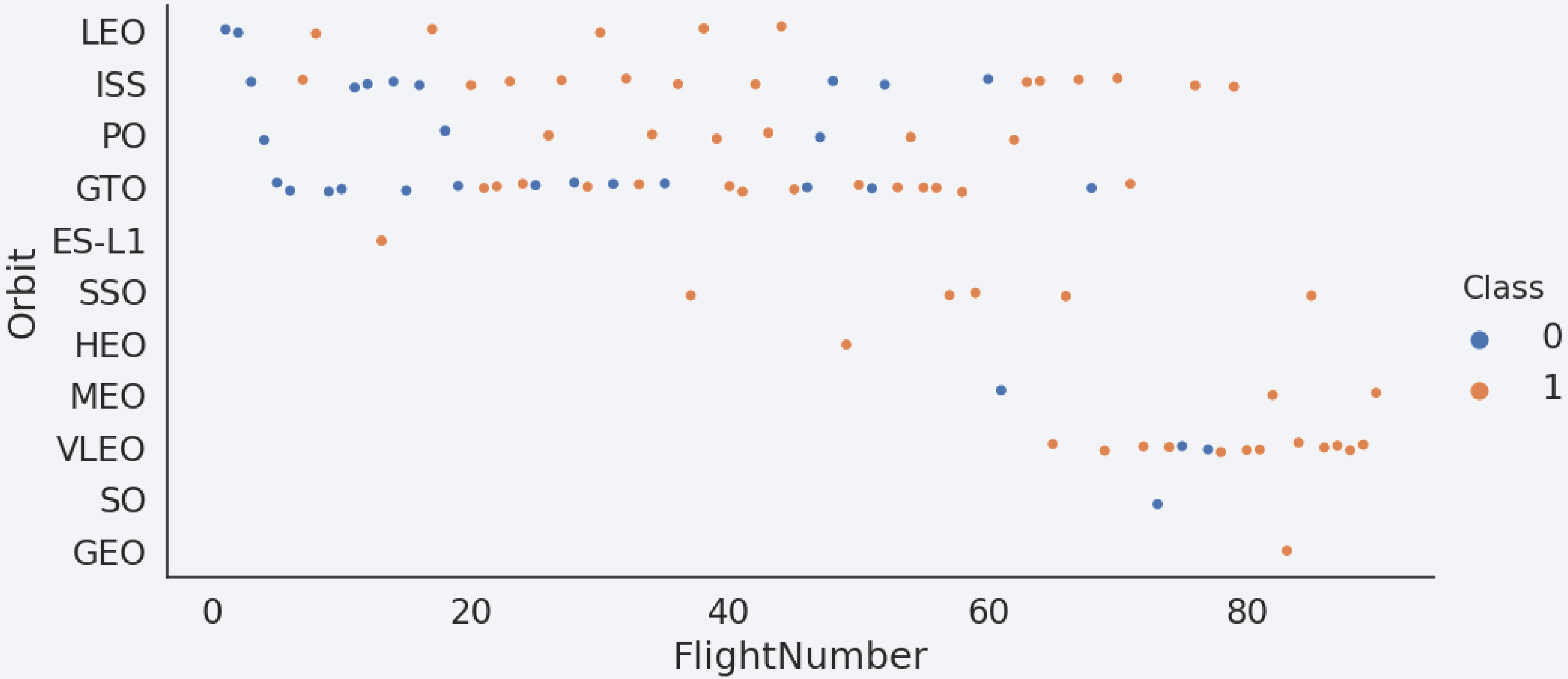
For payload masses above 10000 kg there seems to be an increase in the success rate

# Success Rate vs. Orbit Type



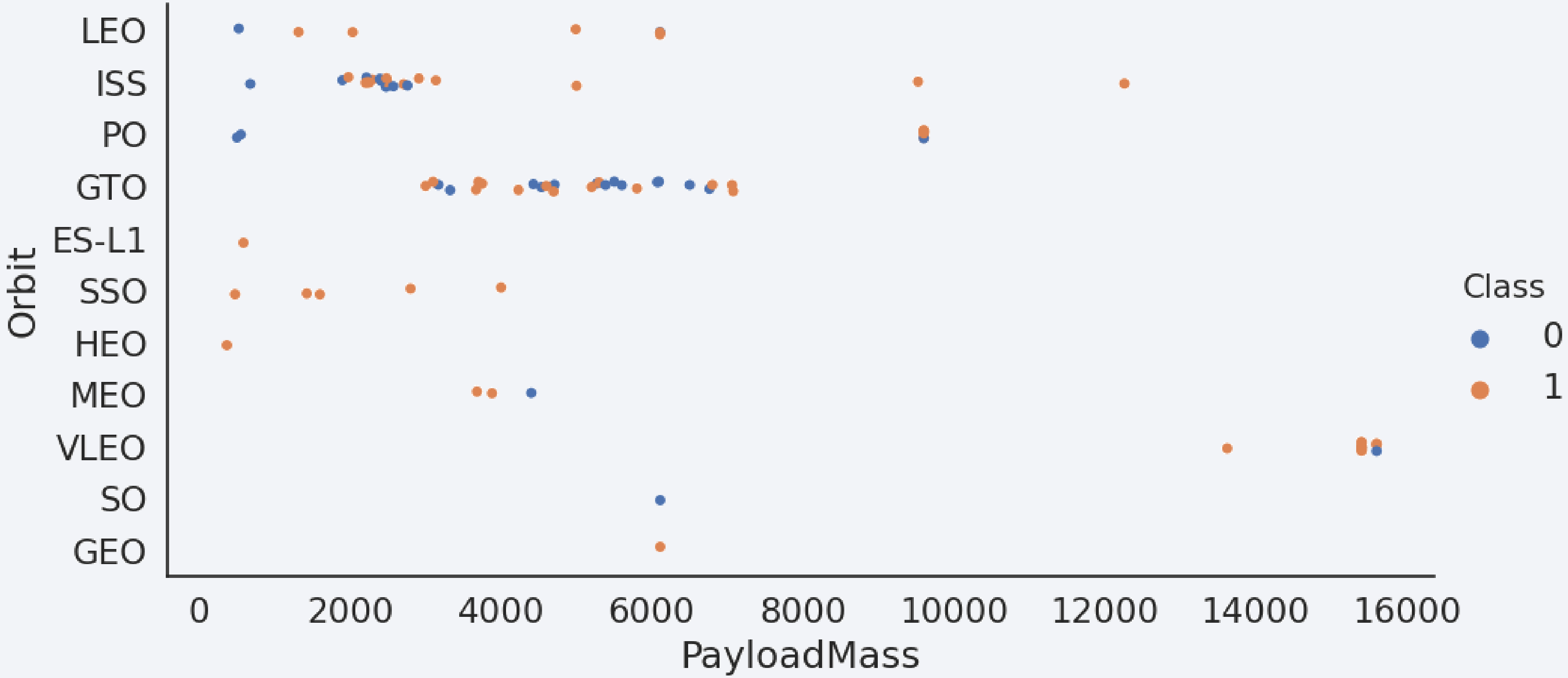
ES-L1, GEO, HEO and SSO have perfect success rate (no failures). VLEO has over 80%, which is also not bad.

# Flight Number vs. Orbit Type



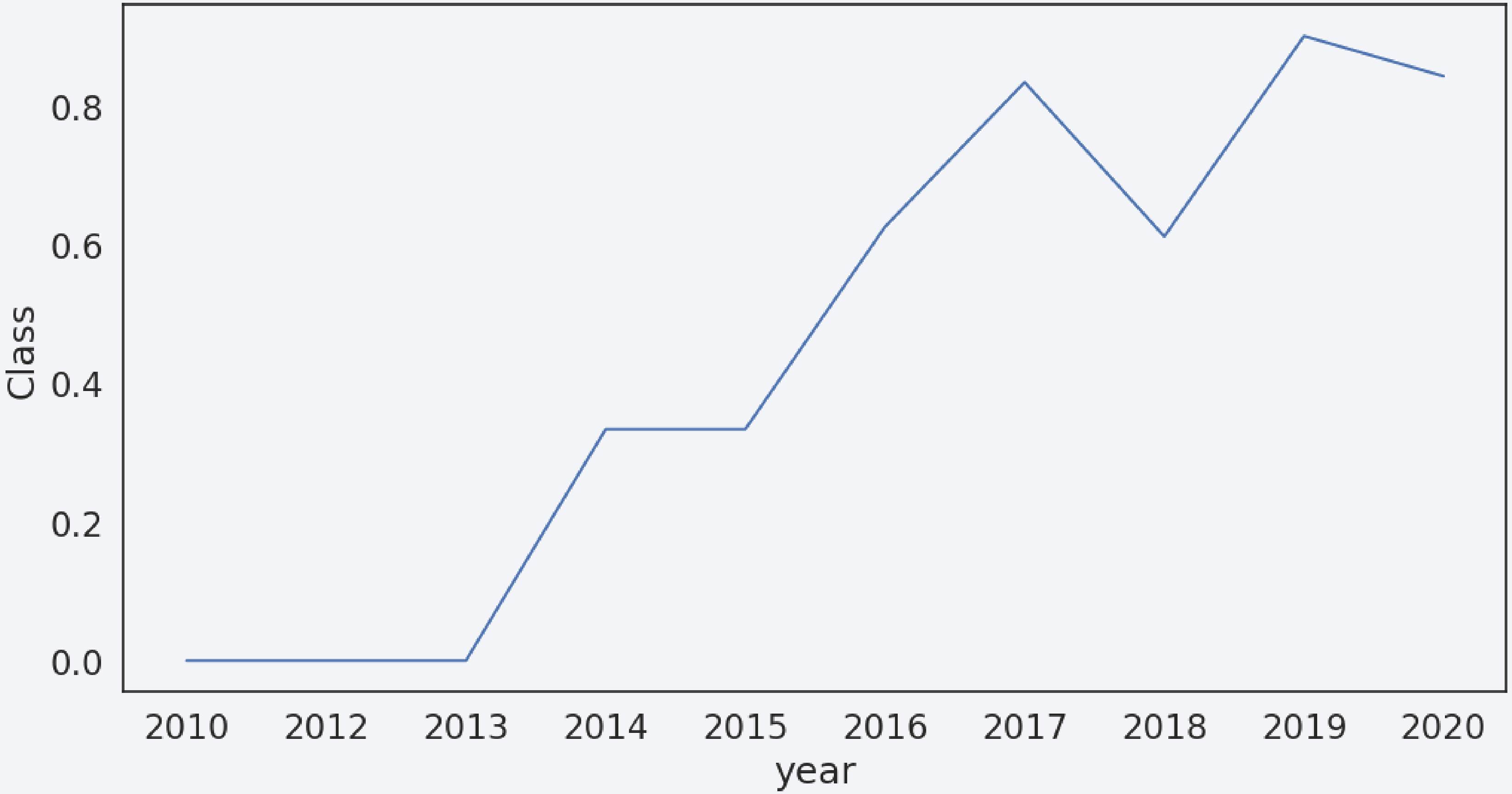
LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit, for example.

# Payload vs. Orbit Type



With heavy payloads the success rate is bigger for Polar,LEO and ISS orbits. On GTO orbit it's not possible distinguish between good and bad launches.

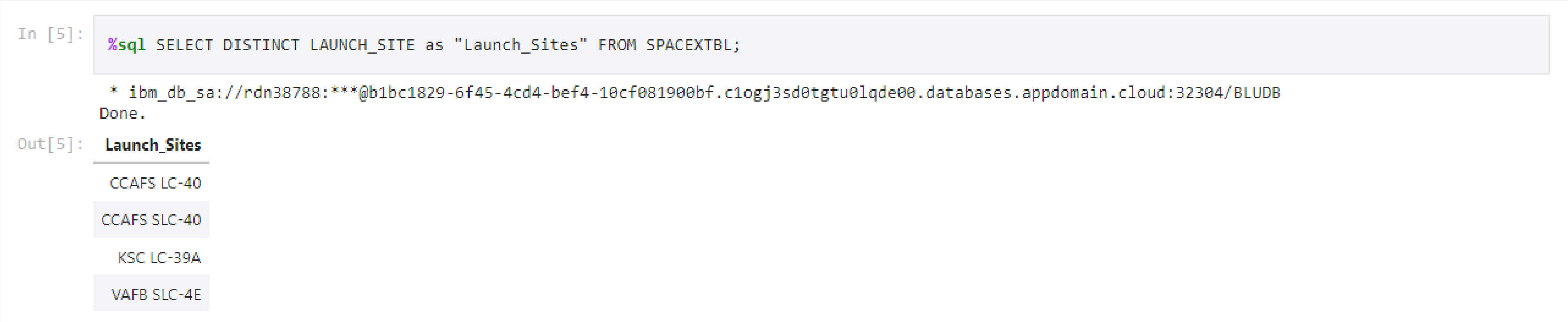
# Launch Success Yearly Trend



Success rate is increasing since 2013 till 2020

# All Launch Site Names

SQL keyword DISTINCT was used to show only unique launch sites from the SpaceX API data.

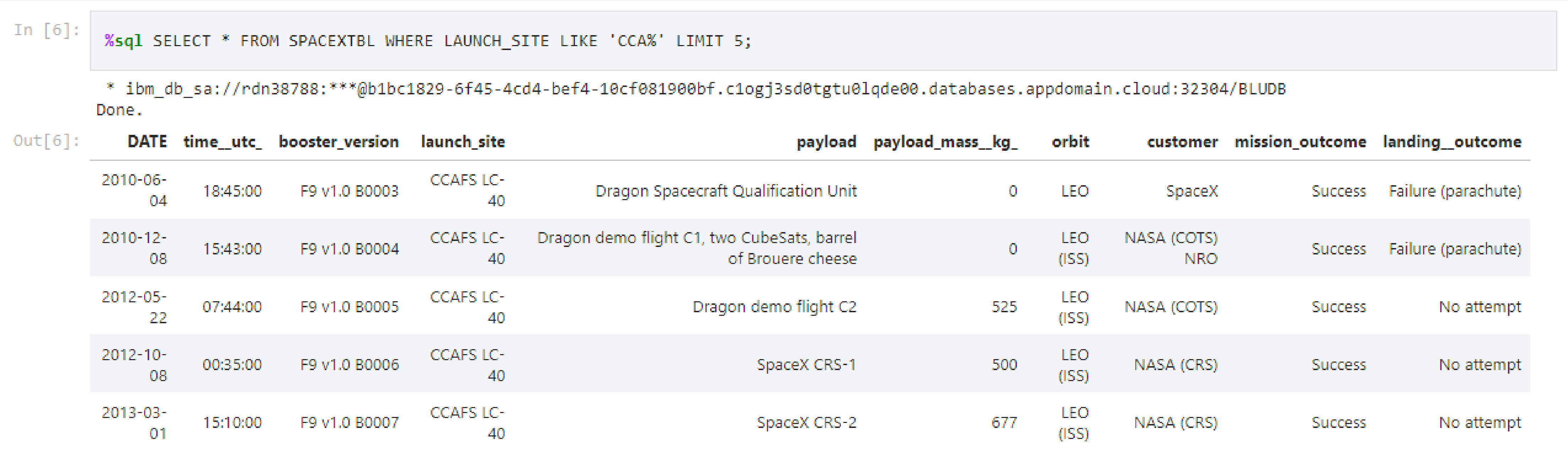


```
In [5]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;

         * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
        Done.
Out[5]:
```

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

# Launch Site Names Begin with 'CCA'

SQL keyword LIKE was used to find data that starts with CCA and keyword LIMIT was used to show only 5 records that meet the condition.



In [6]:

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

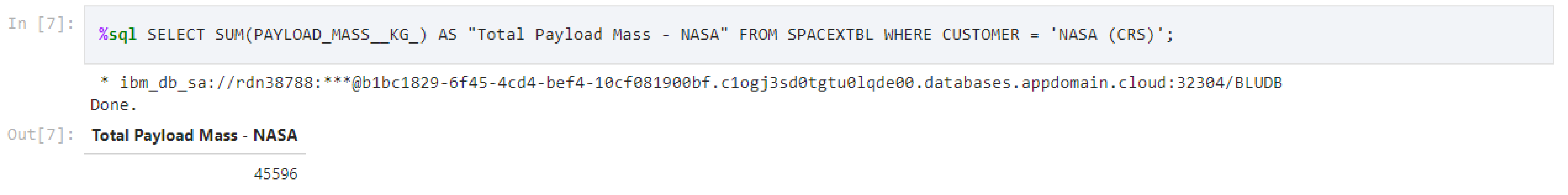
```
 * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
Done.
```

Out[6]:

| 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 | 07:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success | No attempt |
| 2012-10-08 | 00:35:00 | 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

SQL function SUM was used to calculate the total payload carried by all boosters and keyword WHERE was used to find data that starts with CCA and keyword LIMIT was used to show only 5 records that meet the condition.



```
In [7]: %sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass - NASA" FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';
```

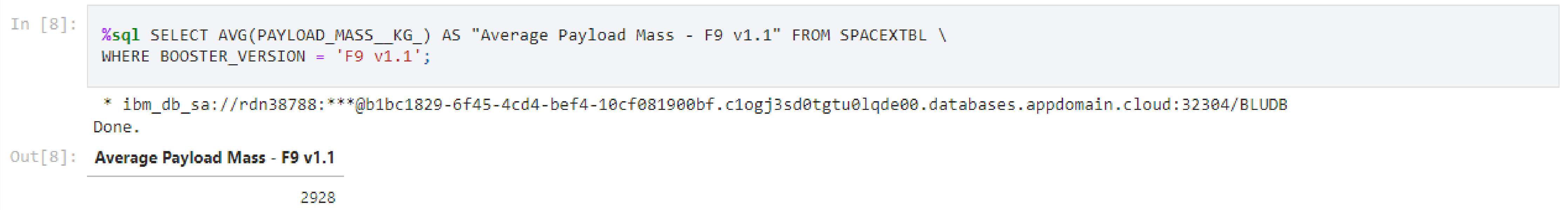
```
 * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
Done.
```

Out[7]:

| Total Payload Mass - NASA |
|---|
| 45596 |

# Average Payload Mass by F9 v1.1

SQL function SUM was used to calculate the payload mass carried and the keyword WHERE was used to limit the query to the booster version F9 v1.1.



In [8]:

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass - F9 v1.1" FROM SPACEXTBL \
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

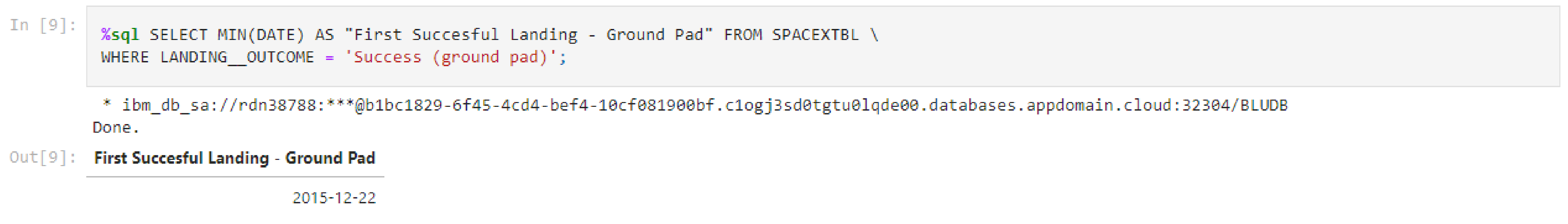
```
 * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
Done.
```

Out[8]:

| Average Payload Mass - F9 v1.1 |
|---|
| 2928 |

# First Successful Ground Landing Date

SQL function MIN was used to select the minimum of the DATE column, which means the first date stored and the keyword WHERE was used to specify the success condition of the landing.



In [9]:

```
%sql SELECT MIN(DATE) AS "First Succesful Landing - Ground Pad" FROM SPACEXTBL \
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

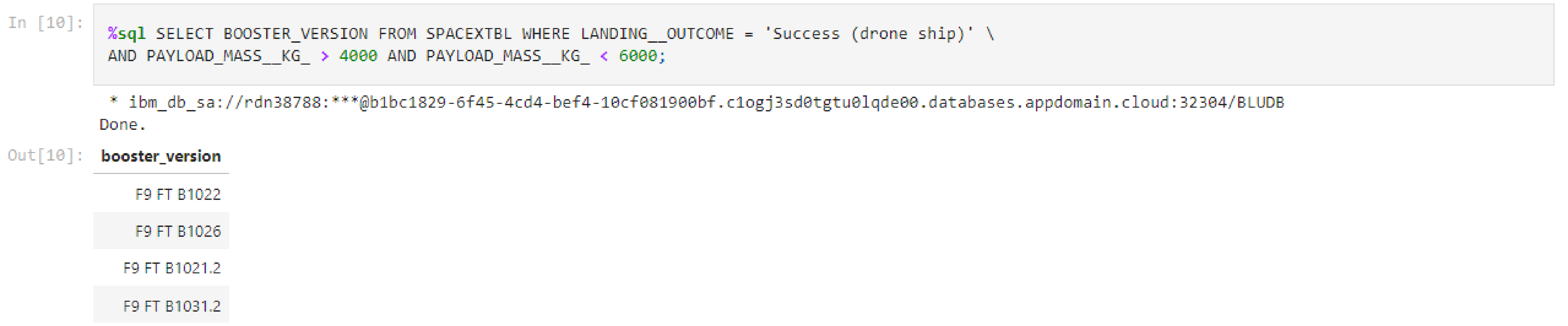
```
 * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
Done.
```

Out[9]:

| First Succesful Landing - Ground Pad |
|---:|
| 2015-12-22 |

# Successful Drone Ship Landing with Payload between 4000 and 6000

SQL keyword WHERE clause to filter for boosters which have successfully landed on drone ship and used the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000



```
In [10]: %sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Success (drone ship)' \
         AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

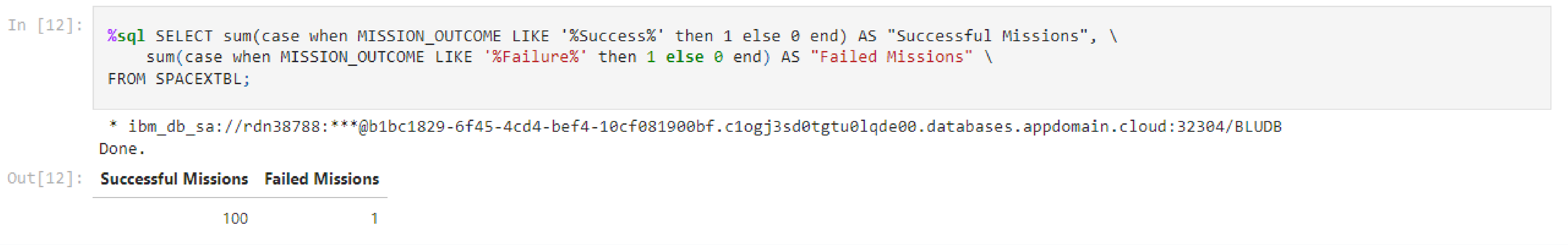
```
 * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
Done.
```

Out[10]:

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

# Total Number of Successful and Failure Mission Outcomes

SQL function SUM was used to count the occurrences and the keyword LIKE was used to find everything that has “Success” or “Failure” on any relative position in the text. The result is shown as a new table.



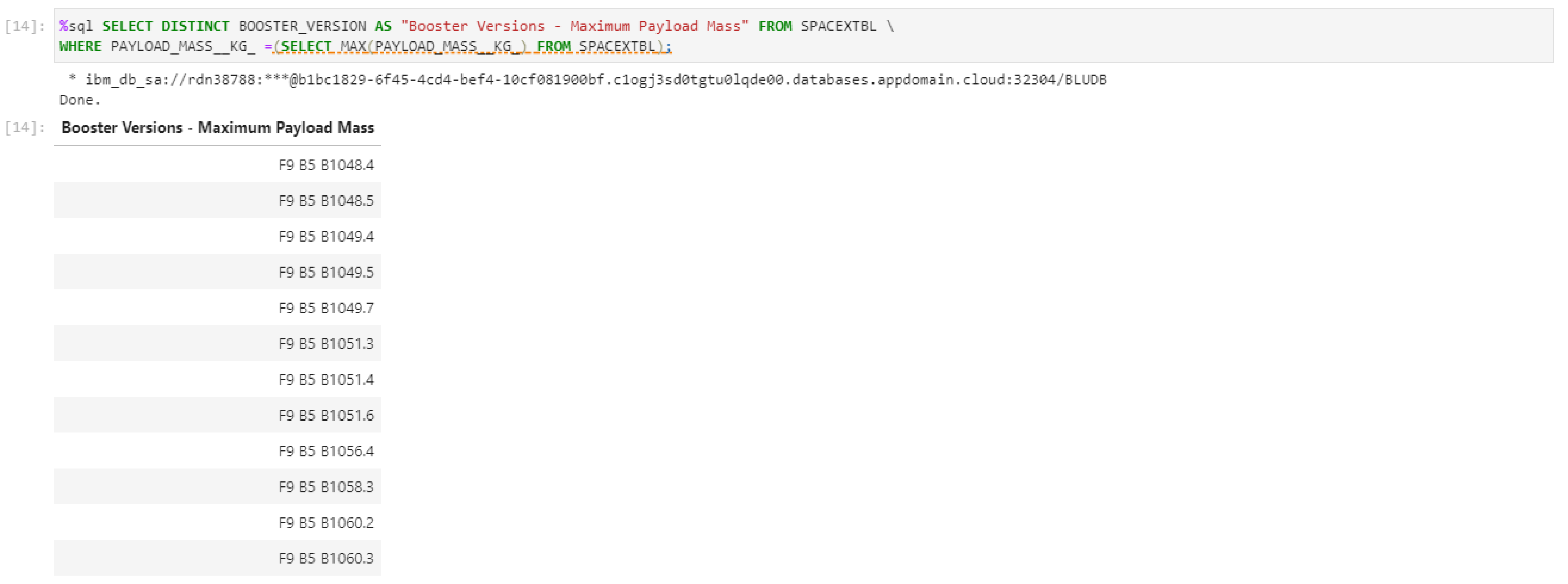
```
In [12]:  %sql SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "Successful Missions", \
              sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "Failed Missions" \
          FROM SPACEXTBL;

           * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
          Done.
```

Out[12]:

| Successful Missions | Failed Missions |
|---:|---:|
| 100 | 1 |

# Boosters Carried Maximum Payload

SQL function MAX was used to determine maximum payload values and the keyword WHERE was used to filter the booster versions according to the MAX value specified by the MAX function



[14]:
```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions - Maximum Payload Mass" FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

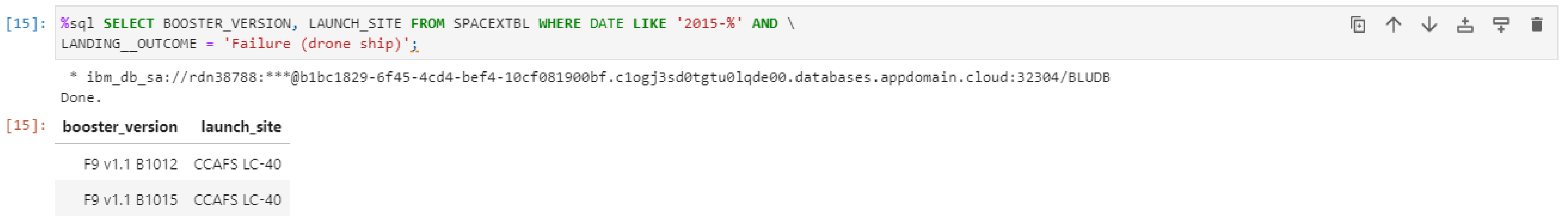
```
 * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
Done.
```

[14]:

| Booster Versions - Maximum Payload Mass |
|---:|
| F9 B5 B1048.4 |
| F9 B5 B1048.5 |
| F9 B5 B1049.4 |
| F9 B5 B1049.5 |
| F9 B5 B1049.7 |
| F9 B5 B1051.3 |
| F9 B5 B1051.4 |
| F9 B5 B1051.6 |
| F9 B5 B1056.4 |
| F9 B5 B1058.3 |
| F9 B5 B1060.2 |
| F9 B5 B1060.3 |

# 2015 Launch Records

SQL keyword WHERE as used to filter, the AND operator was used to combine filter conditions and the keyword LIKE was used to select only dates starting with 2015



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

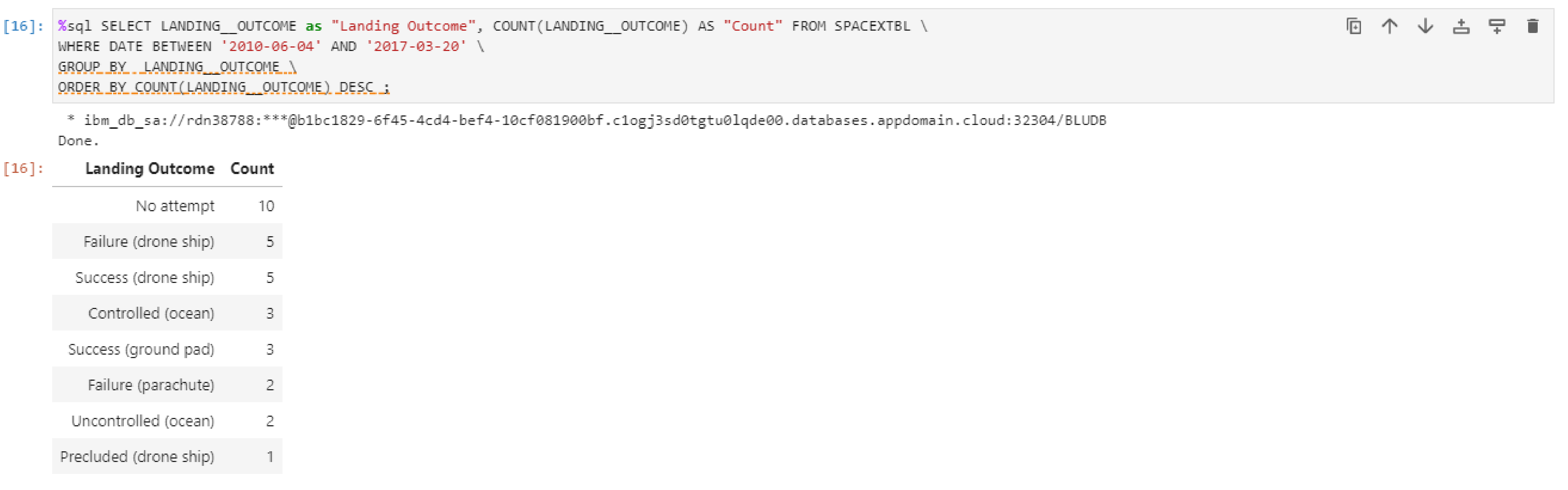
```
 * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
Done.
```

[15]:

| 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

SQL function COUNT was used to count landing outcomes, then WHERE keyword was used to filter BETWEEN 2010-06-04  and 2010-03-20. Finally, GROUP BY keyword was used to group landing outcomes and ORDER BY to make them descending.



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

       * ibm_db_sa://rdn38788:***@b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32304/BLUDB
      Done.
```

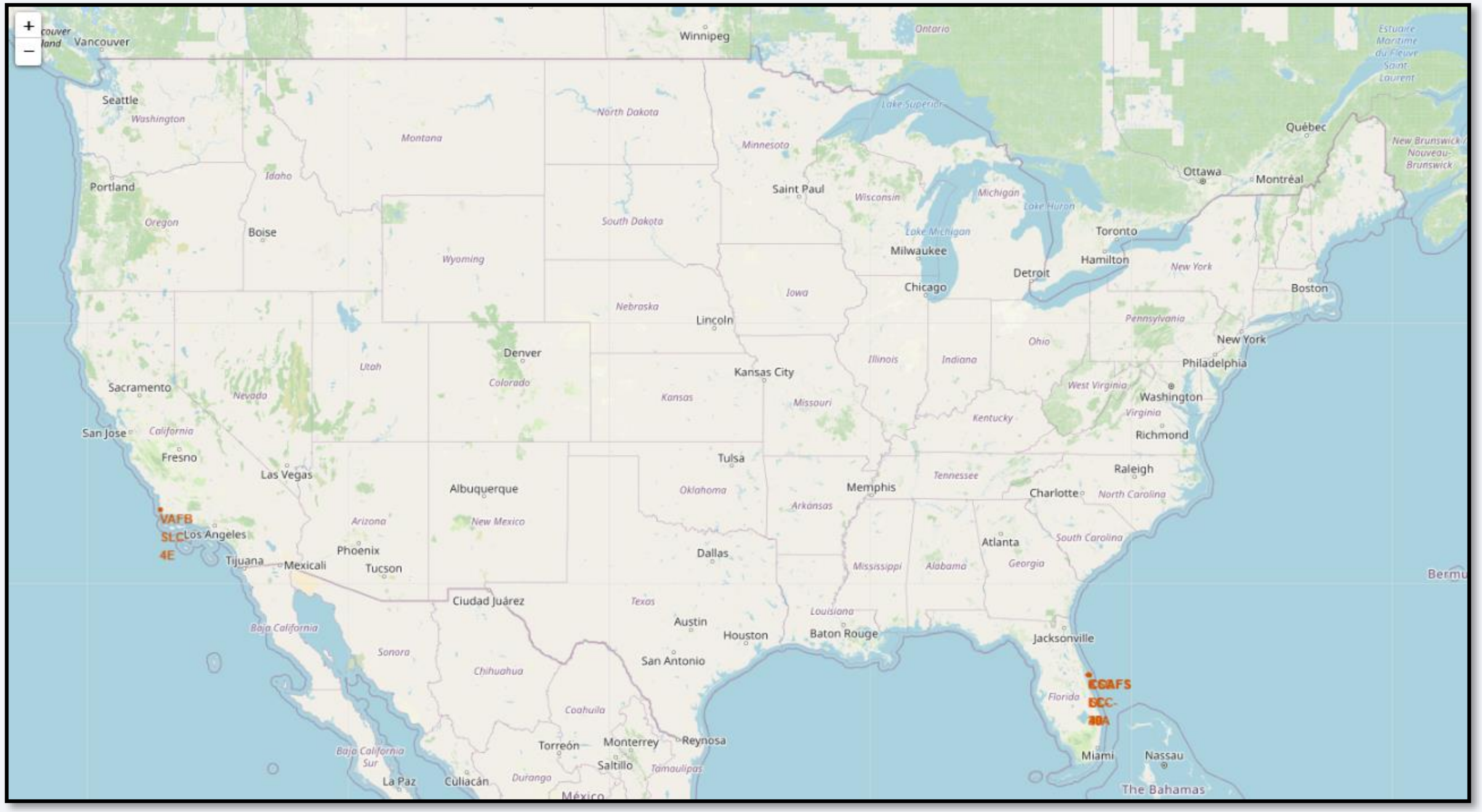
[16]:

| Landing Outcome | Count |
|---|---|
| No attempt | 10 |
| Failure (drone ship) | 5 |
| Success (drone ship) | 5 |
| Controlled (ocean) | 3 |
| Success (ground pad) | 3 |
| Failure (parachute) | 2 |
| Uncontrolled (ocean) | 2 |
| Precluded (drone ship) | 1 |

Section 3

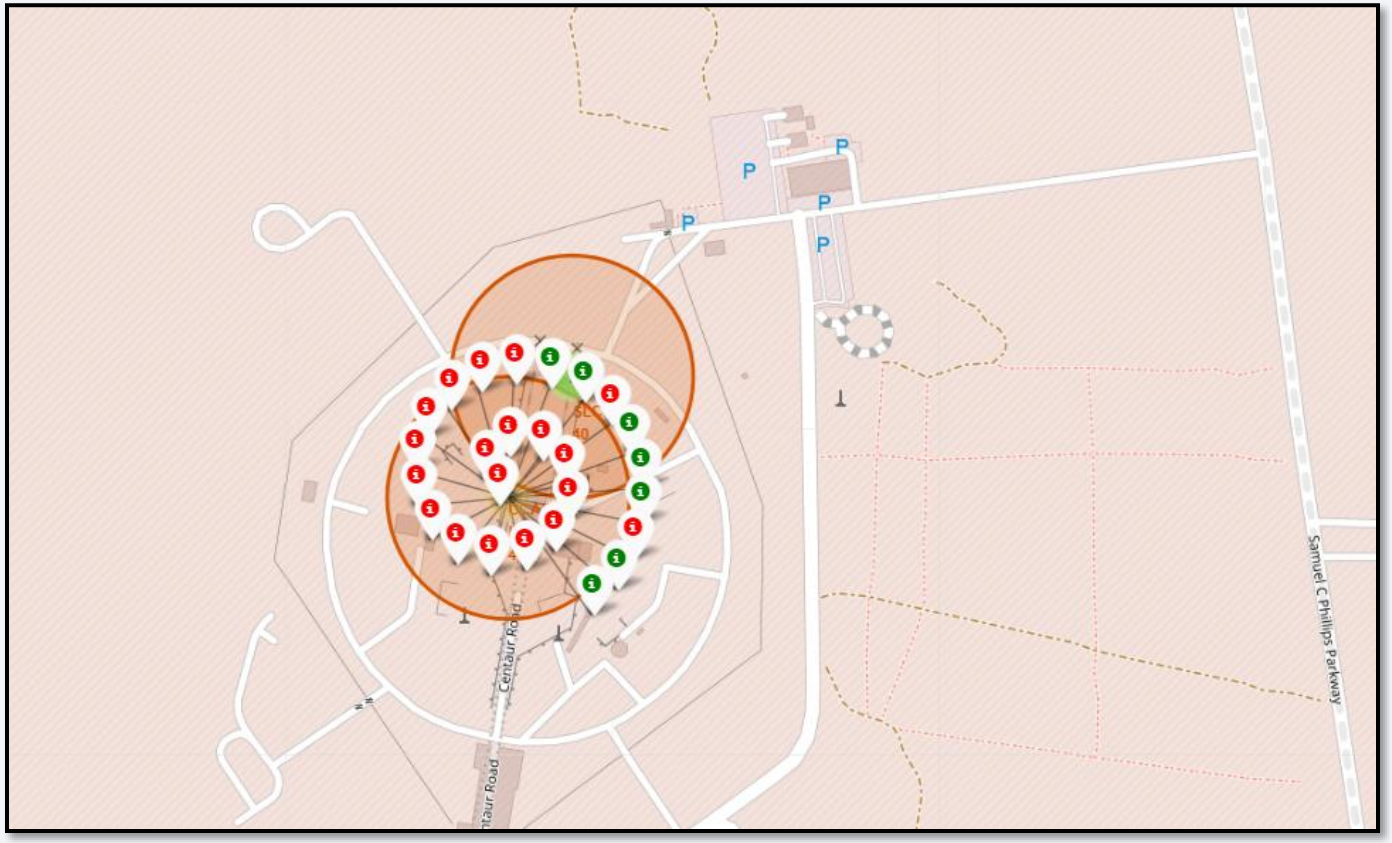
# Launch Sites Proximities Analysis

# Launch site location



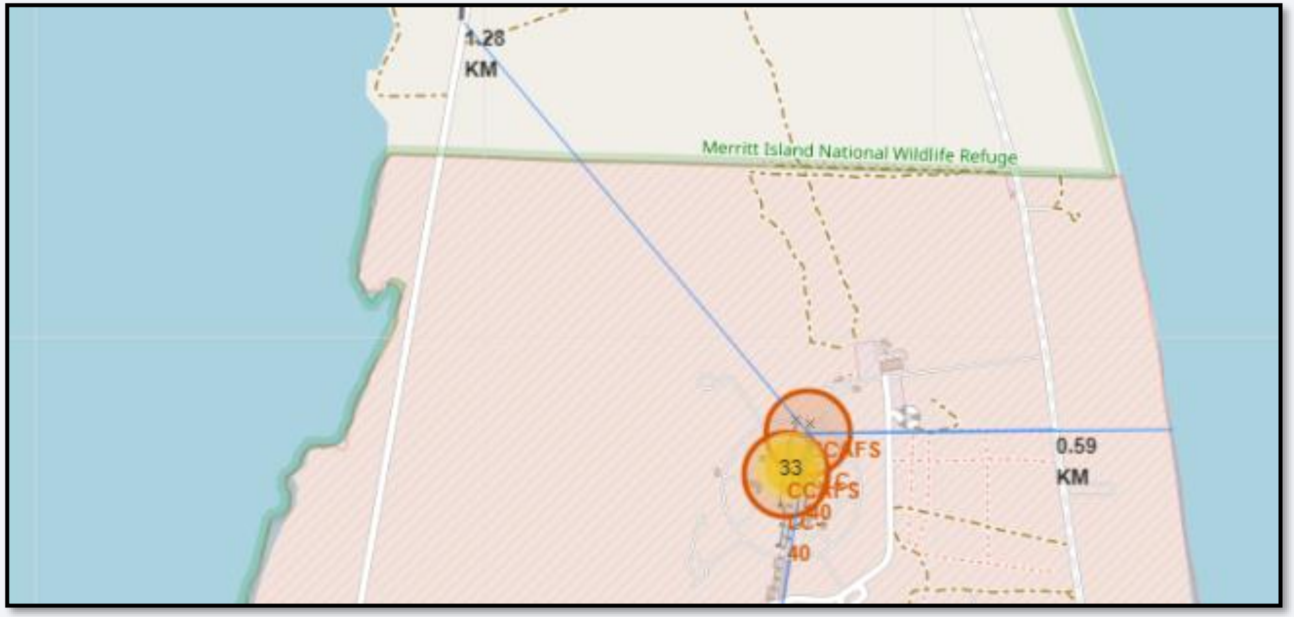
All launch sites are within US, mainly in Florida (East Coast) and California (West Coast)

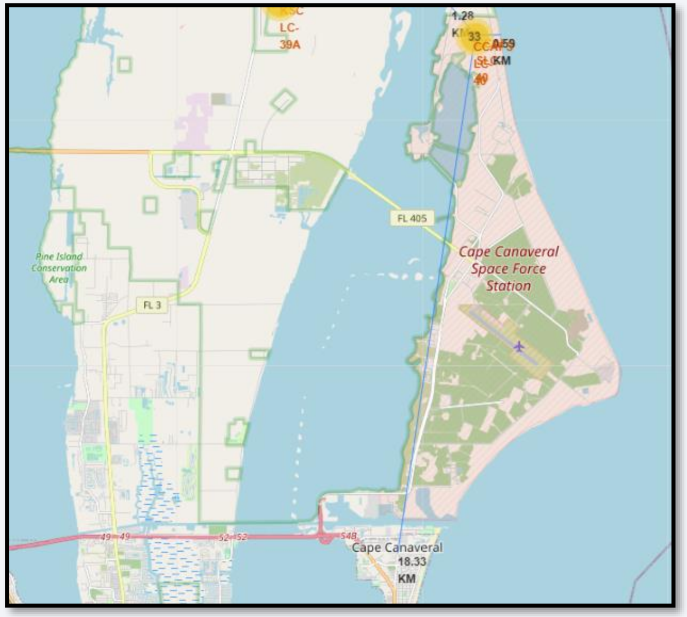
# Markers launch sites with color labels for outcomes



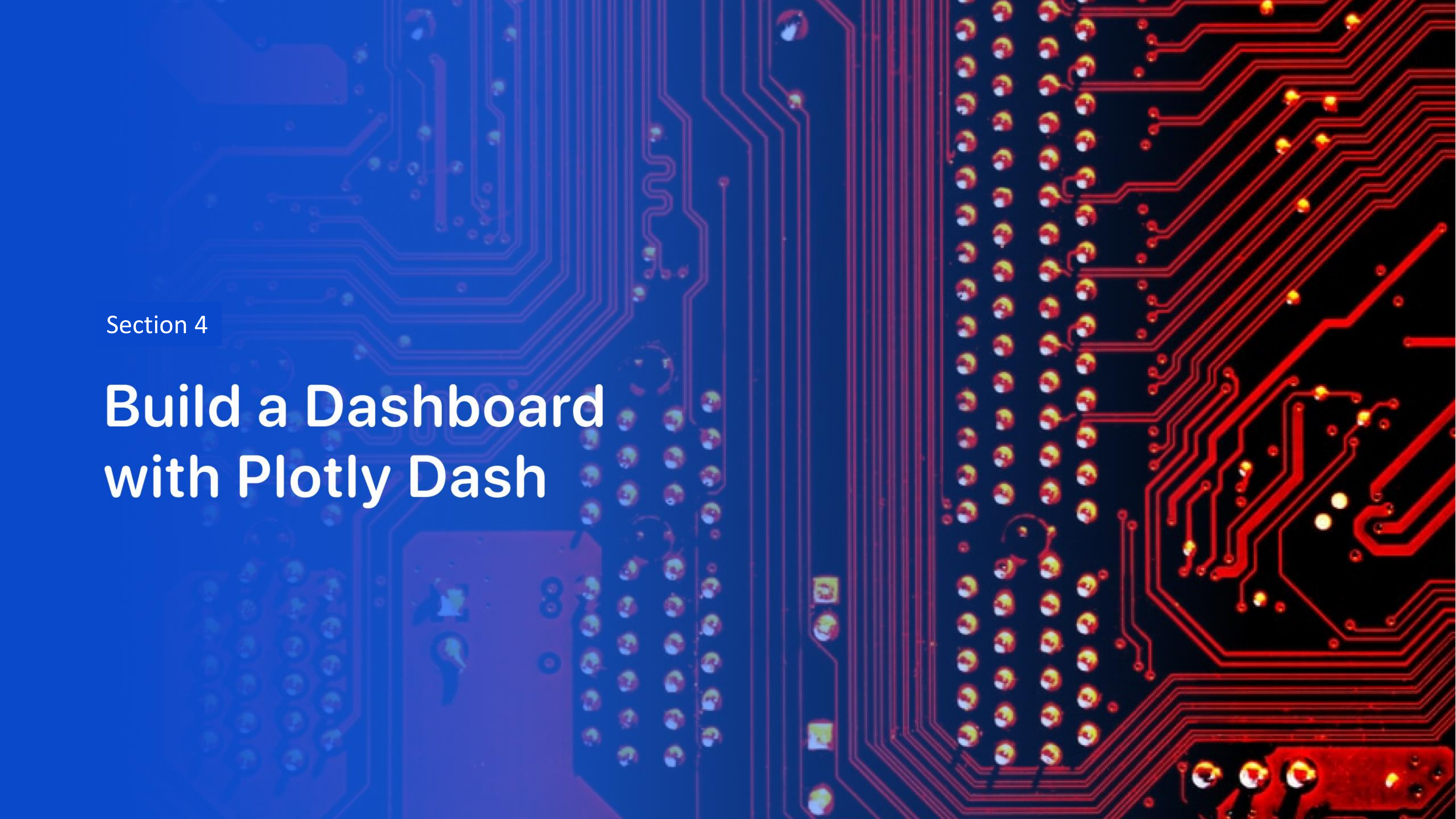
For each launch site, markers were added with RED color for failed landings and GREEN color for successful landings.

# Distance from launch sites do landmarks




LC-39A
1.28 KM
0.59 KM
FL 405
Cape Canaveral Space Force Station
Pine Island Conservation Area
FL 3
Cape Canaveral
18.33 KM

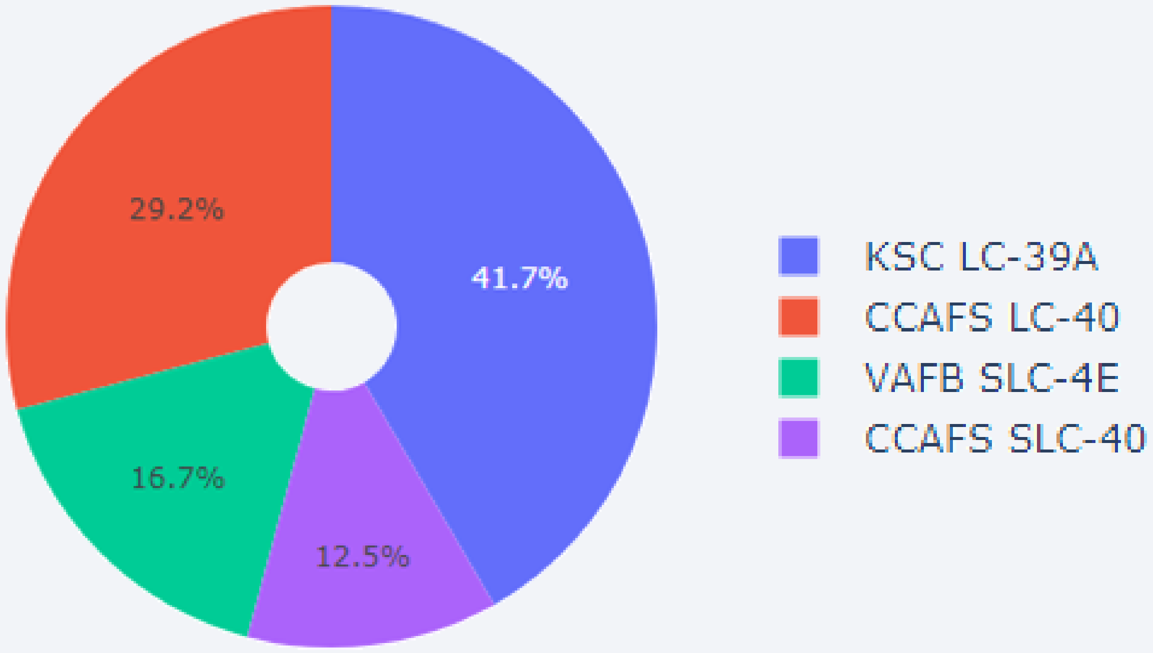
- For a Cape Canaveral (Florida) launch site, for example, it was found that the launch site is:
  - 900 m away from coastline
  - 600 m away from highway (which gives access to the launch site)
  - 1,28 km away from railway
  - 18.3 km away from a city



Section 4

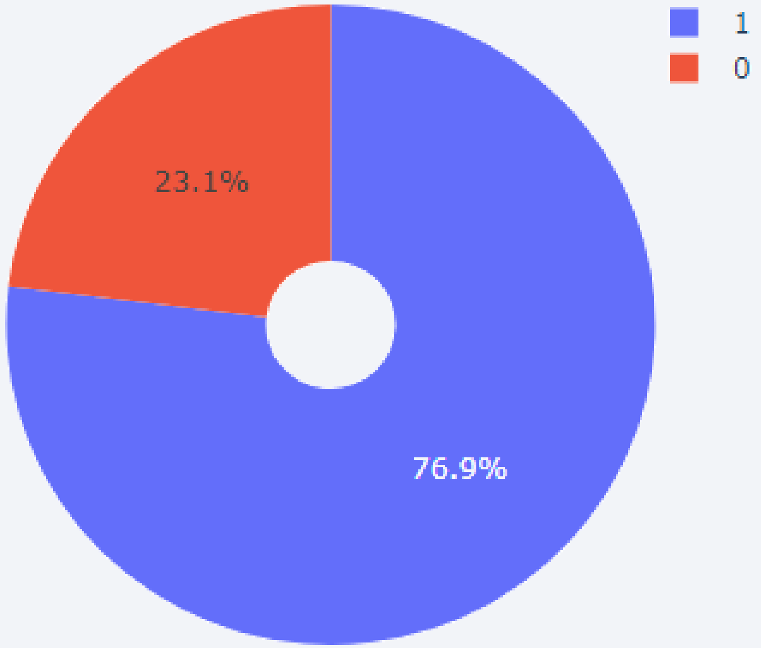
# Build a Dashboard with Plotly Dash

# Success rate by each launch site



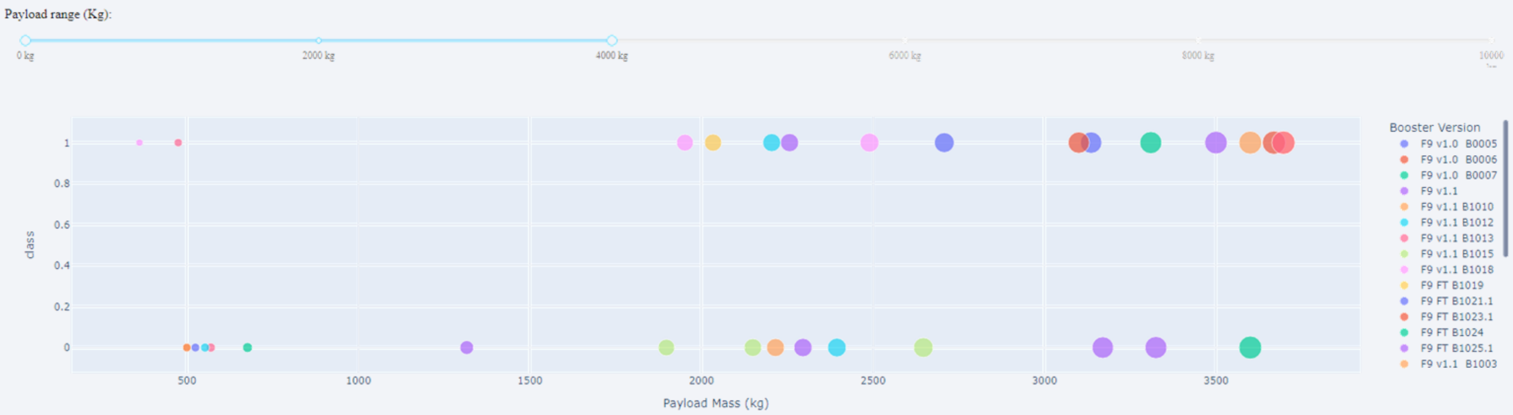
KSC LC-39A has the biggest success rate between all launching sites

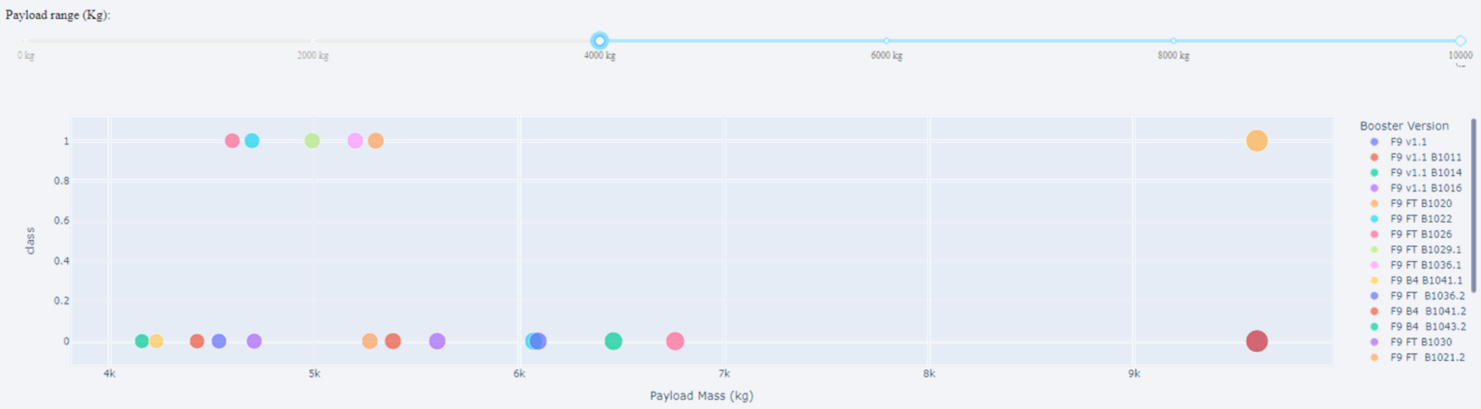
# Success rate for KSC LC-39A



KSC LC-39A has a 76.9% success rate

# Payload vs Launch Outcome for all sites



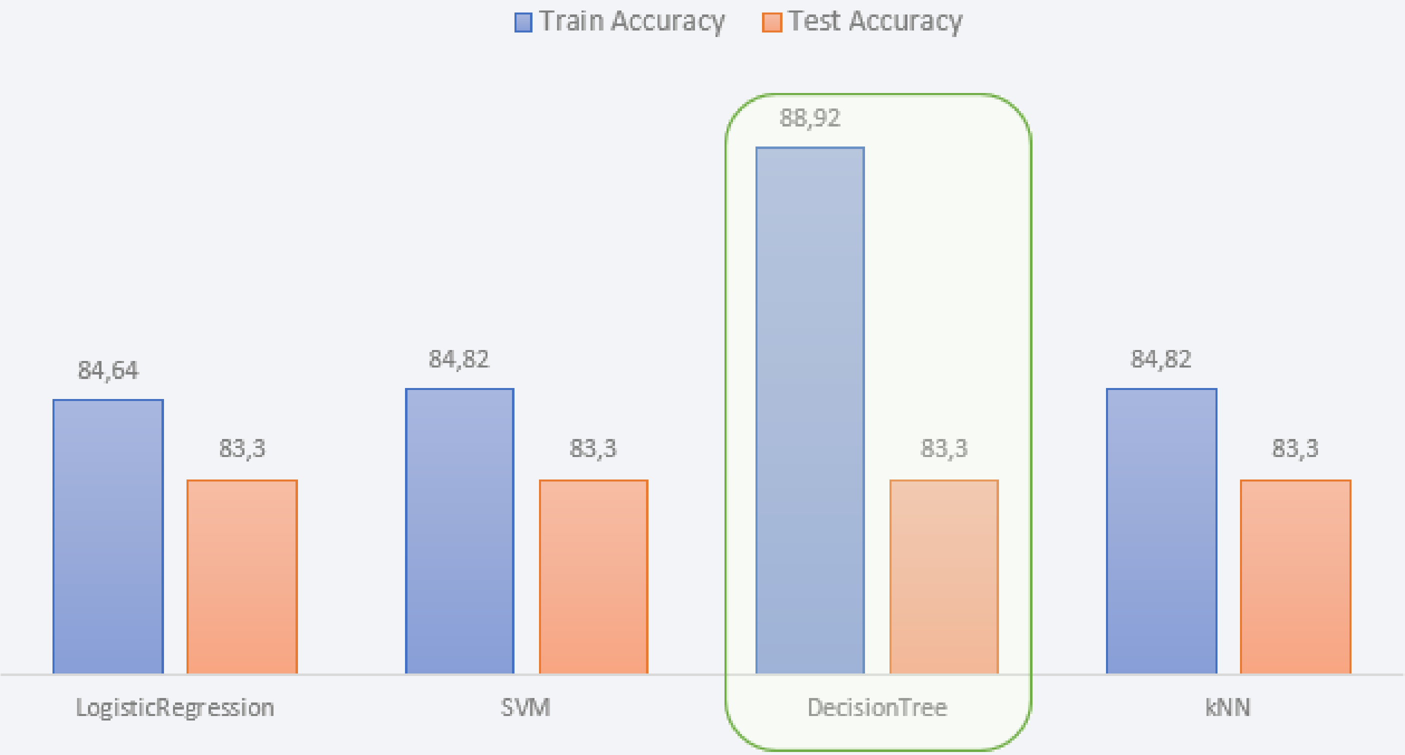


Success rates for low payloads are higher than for heavy weighted payloads

Section 5

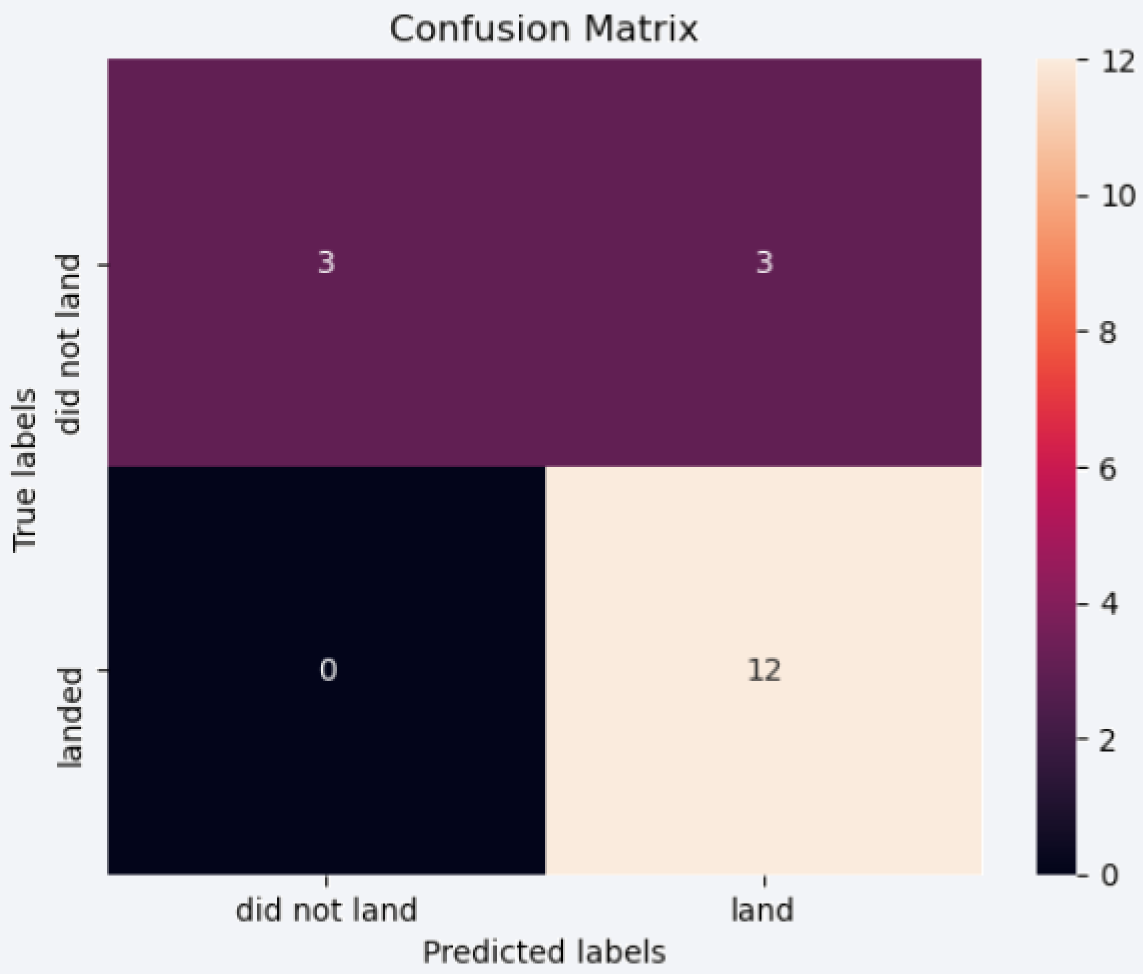
# Predictive Analysis (Classification)

# Classification Accuracy



The decision tree showed to be the best approach based on train accuracy

# Confusion Matrix



The confusion matrix shows that the classifier can distinguish between the different classes.

The problem is the 3 false positives: unsuccessful landing marked as successful landing by the classifier.

# Conclusions

- From 2013, the success rate for SpaceX launches is increased, directly proportional time in years to 2020;
- ES-L1, GEO, HEO and SSO have perfect success rate (no failures). VLEO has over 80%, which is also not bad;
- KSC LC-39A had the most successful launches;
- Larger number of flights in a launch site tend to increase the success rate for that specific site;
- Low weighted payloads (below 4000kg) performed better than the heavy weighted payloads;
- Decision Tree Classifier is the best Machine Learning approach for this task.

Thank you!