IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Rabbi Islam Yeasin

March22,2024

Outline

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

Executive Summary

- Summary of methodologies

1. Data Collection
2. Data Wrangling
3. Exploratory Data Analysis with SQL
4. Exploratory Data Analysis with Visualization
5. Exploratory Data Analysis with Folium
6. Machine Learning Prediction

- Summary of all results

1. Exploratory Data Analysis Result
2. Interactive Analysis with Folium Result
3. Prediction Result

Introduction

- Project background and context

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch

- Problems you want to find answers

1. Predicting if the Falcon 9 first stage will land successfully.
2. Factors that affect successful landing.
3. Interactions between various features that determine its success.
4. Operating conditions to be set in place for a successful outcome

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

  Data was collected by web scraping through the SpaceX API.

- Perform data wrangling

  One Hot encoding was applied to categorical features.

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.

We defined a series of helper functions that will help us use the API to extract information using identification numbers in the launch data.

- We used the get requests to collect the data from the SpaceX API.
- We used a static response object to make the JSON result more consistent.
- We converted the JSON result into a pandas dataframe using .json_normalize()
- We filtered the dataframe to keep only falcon9 launches.
- We checked for missing values and filled them up with its m

Data Collection – SpaceX API

- The get requests was used to collect data from the SpaceX API. The data was then cleaned and formatted as necessary, including some data wrangling.
- Notebook :[Data Collection]

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

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

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head()
```

Data Collection - Scraping

- We used an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.
- • A Beautiful Soup object was then created from the HTML response. • The table was then parsed and converted into a pandas data frame.
- [Web Scraping]()

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP re

```
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url)
html_data.status_code
```

```
200
```

Create a `BeautifulSoup` object from the HTML `response`

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text)
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```
# Use soup.title attribute
soup.title
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table hea

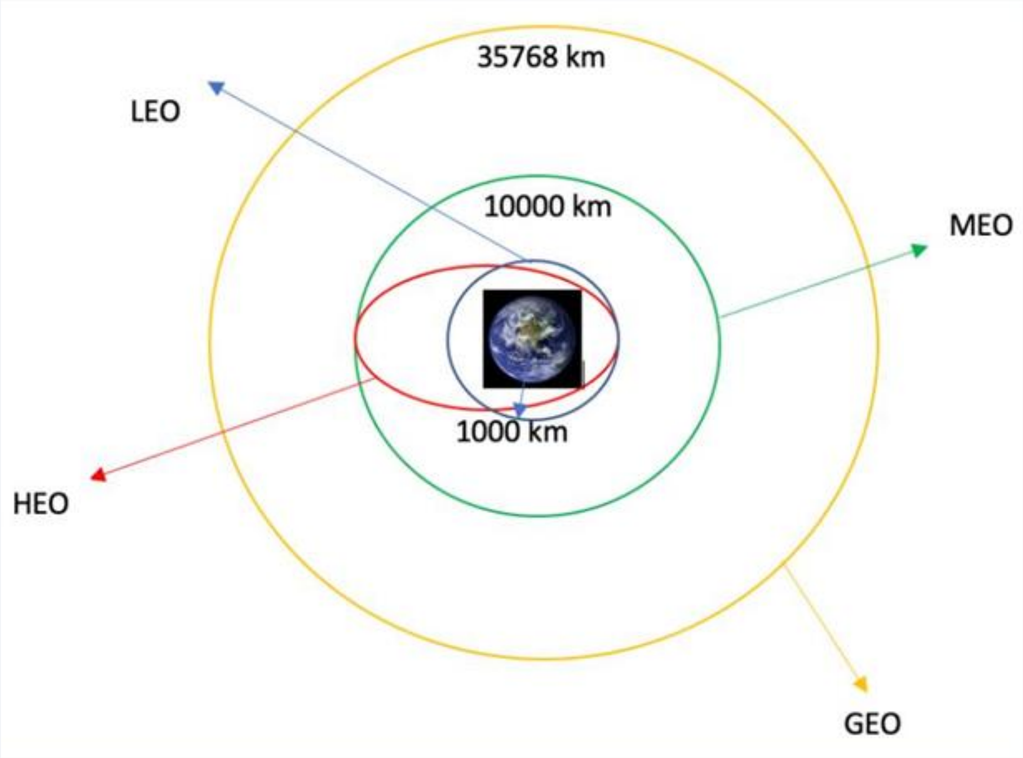
Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `Beaut`
reference link towards the end of this lab

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

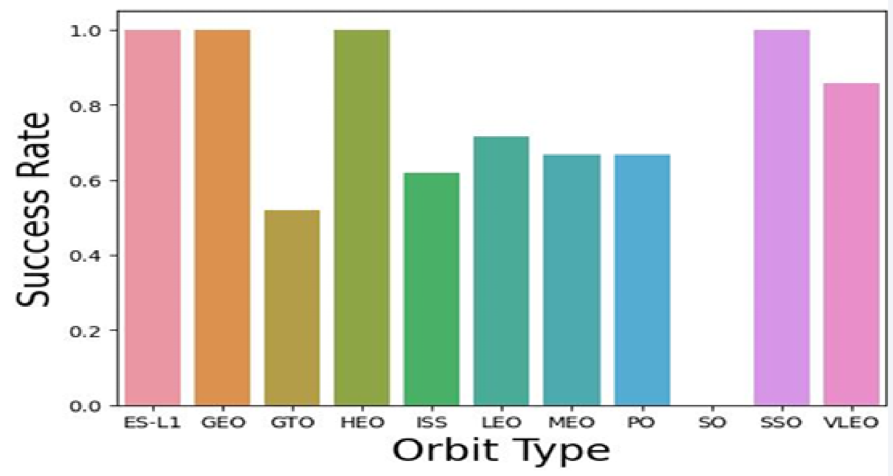
Data Wrangling

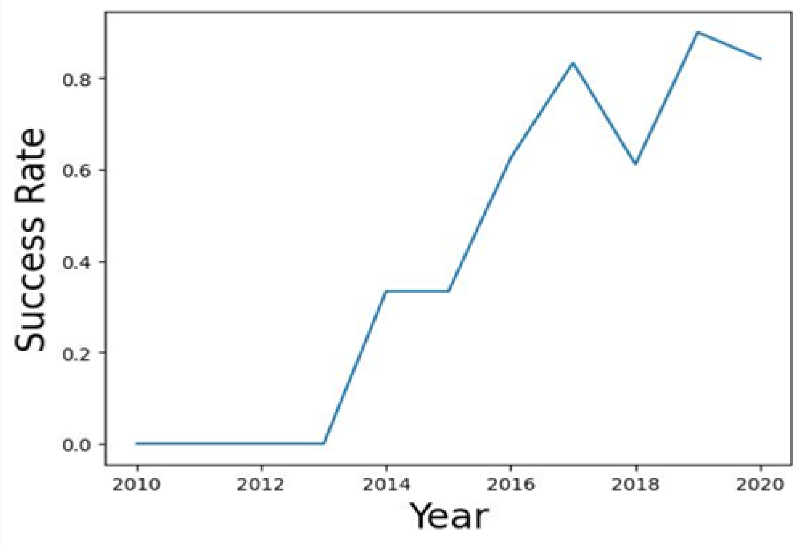
- We performed some Exploratory Data Analysis (EDA) to find some patterns in the data
- We also determined the label for training supervised models.
- We calculated the number of launches on each site. • We also calculated the number and occurence of mission outcome per orbit type.
- We created a landing outcome label from the Outcome column.



EDA with Data Visualization

- We observed the relationship between several factors.
- We observed if there is any relationship between the success rate and the type of orbit it reached.
- We also observed the launch success rate over the years to determine if there has been an improvement or decline.
- The github link to the notebook is:
- [Data Visulization]()





EDA with SQL

- We connected to the MySql database to access our data.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:

- The names of unique launch sites in the space mission.
- The total payload mass carried by boosters launched by NASA (CRS)- The average payload mass carried by booster version F9 v1.1
- The total number of successful and failure mission outcomes
- The failed landing outcomes in drone ship, their booster version and launch site names

[EDA with Notebook]

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1 i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high rate.
- We calculated the distances between a launch site to its proximities. We answered some questions for instance:

-Are launchsite in close proximity to railways, highways and coastline?

- Do launchsite keep certain distance from cities

[Data Visulization with Folium]()

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The github link to the notebook is: Interactive Design with Plotly

Predictive Analysis (Classification)

- We imported all the necessary libraries needed for the analysis and then loaded the data.
- We created a numpy array, standardized the data and then split the dataset into train and test data.
- We built different machine learning models and tuned into different parameters using GridSearchCV.
- We calculated the accuracy and confusion matrix of each model used.
- The github link to the notebook is: Predictive Analysis

Results

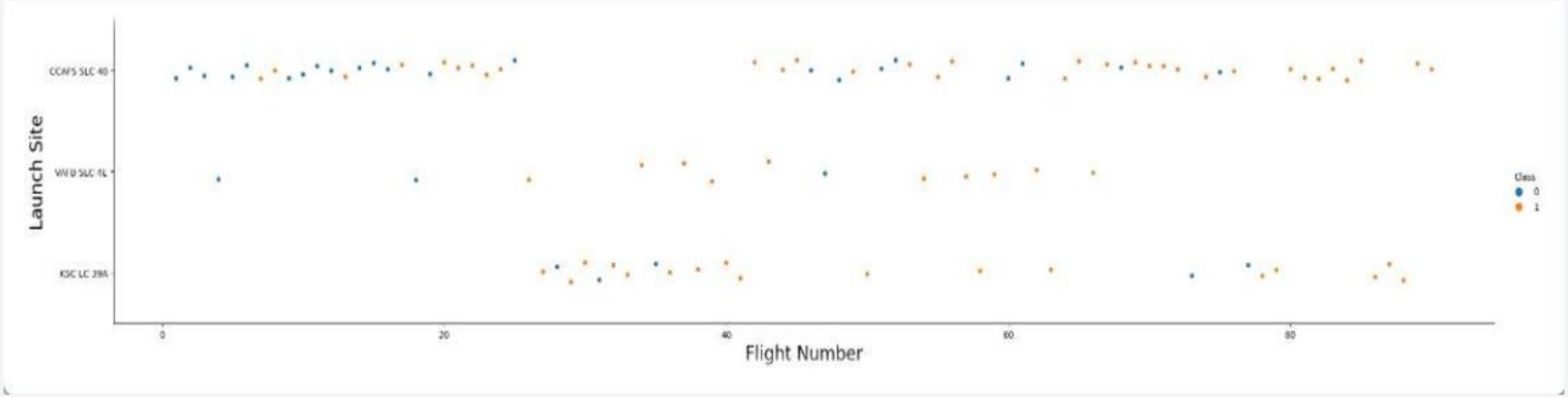
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

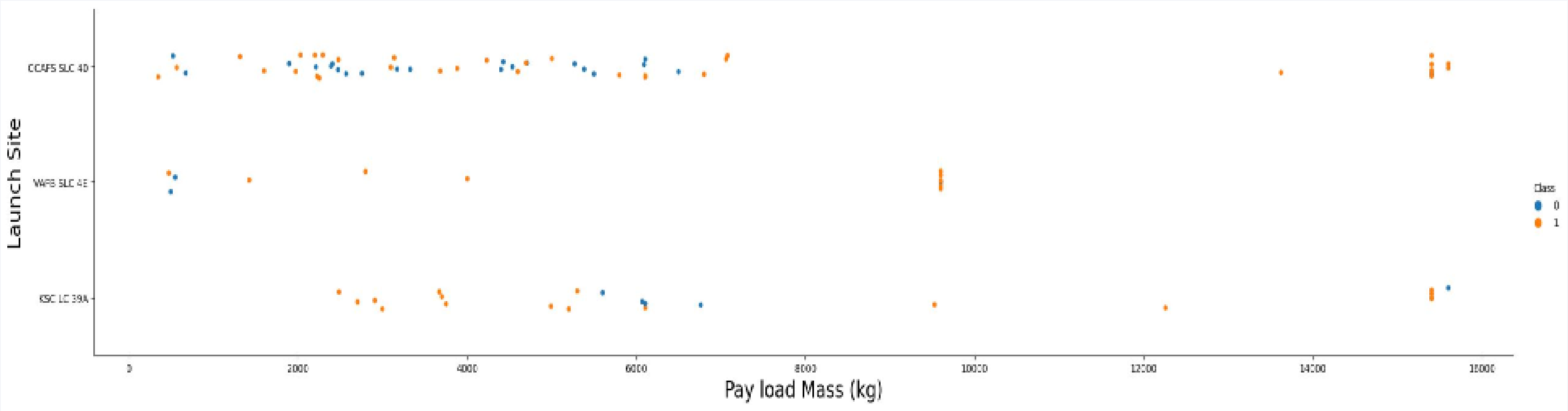
Flight Number vs. Launch Site

- From the scatter plot below, we noted that the higher/ larger the amount of flight at a launch site, the higher the number of successful launches or the greater the success rate



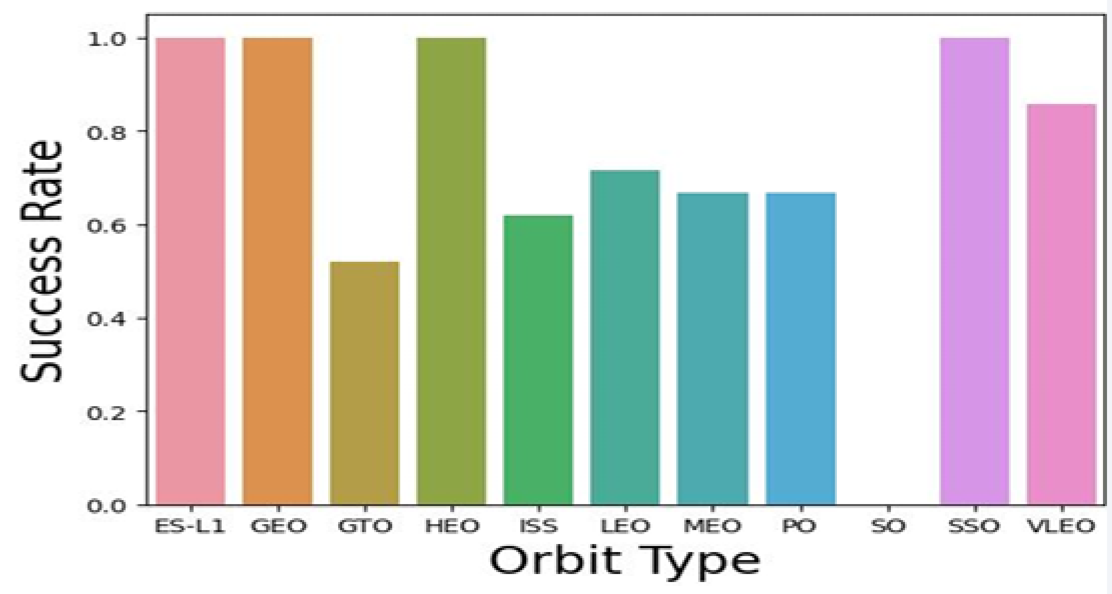
Payload vs. Launch Site

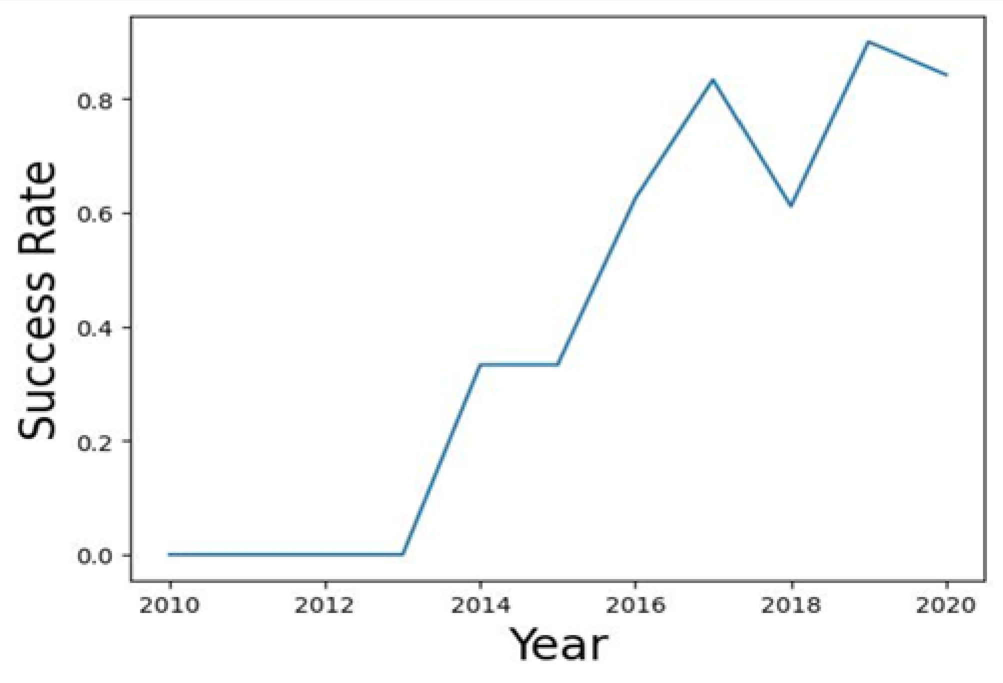
- From the plot below, we can note that the lower the payload mass at launchsite KSC LC-39A, the higher the number of successful launches.



Success Rate vs. Orbit Type

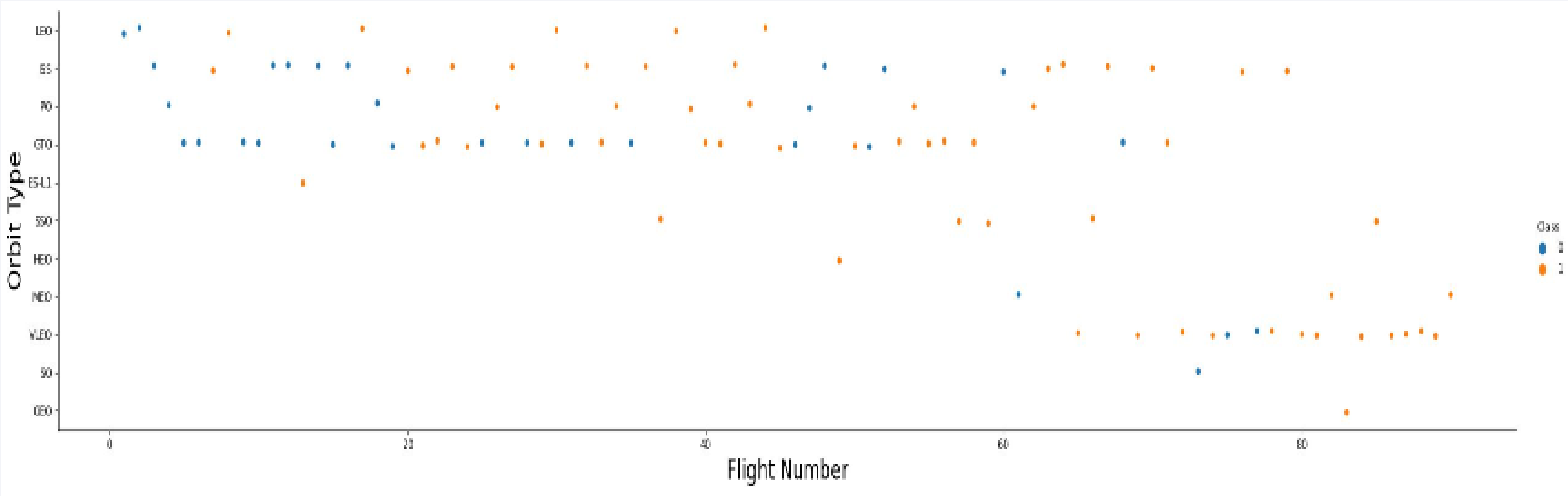
- From the chart, we can note that orbit type : ES-L1, GEO, HEO and SSO had the highest success rate.





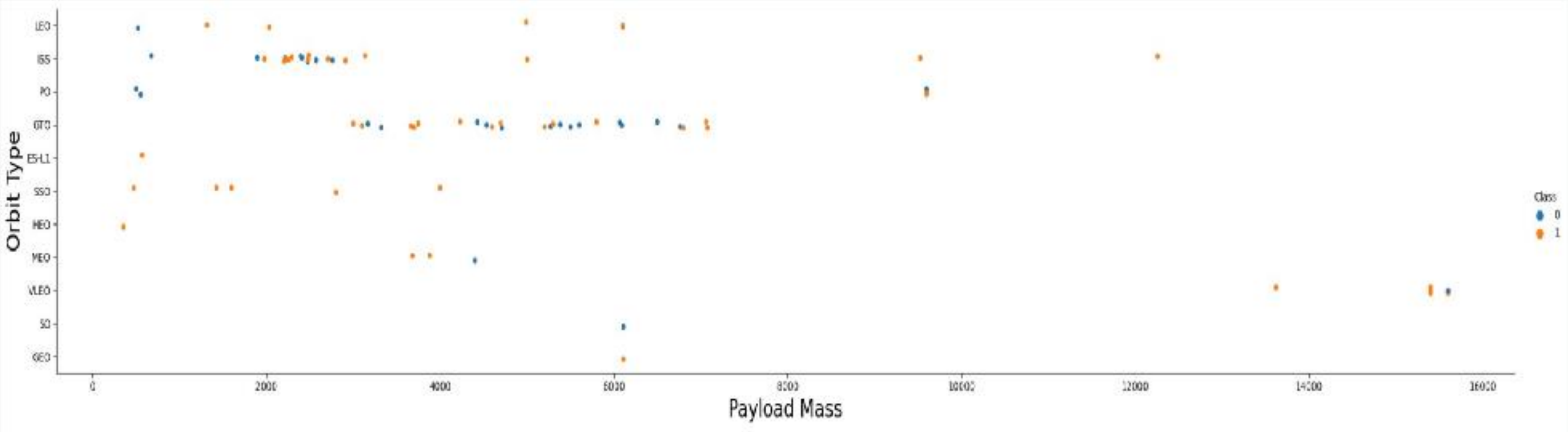
Flight Number vs. Orbit Type

- We can observe from the chart below, we observed that in the LEO and ISS orbit, success rate is relative to the number of flight taken. But for the GTO orbit, there is no observable relationship between the orbit type and number of flight



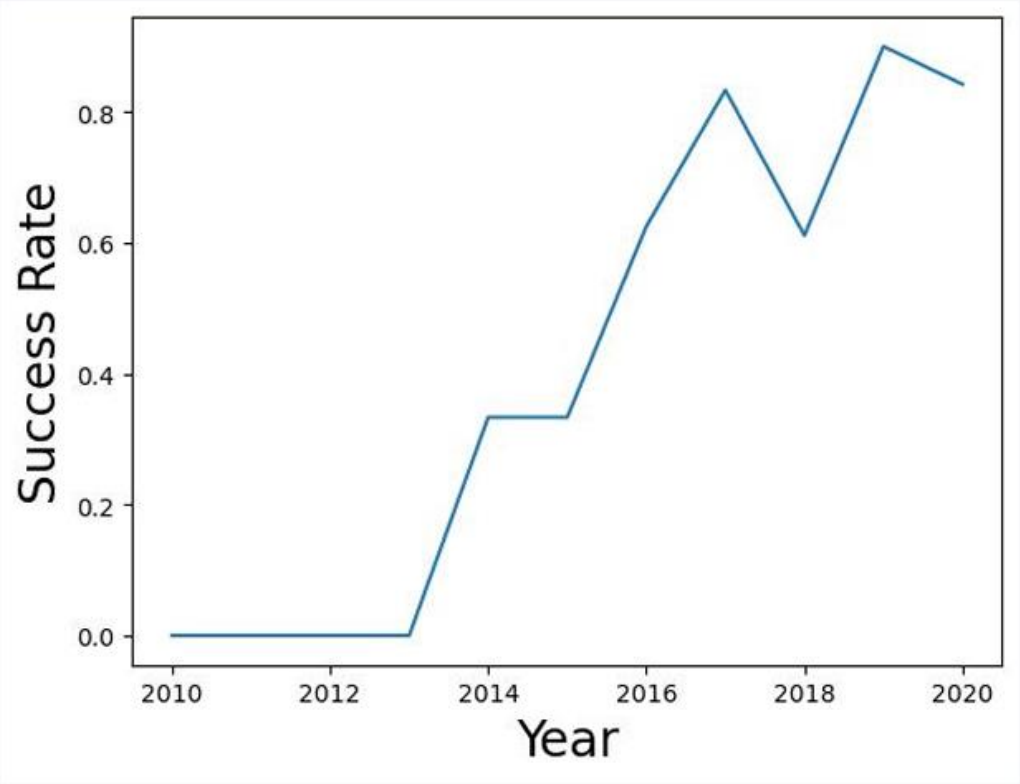
Payload vs. Orbit Type

- From the chart below, we can observe a noticeable relationship between payloadmass and orbit type in the orbits LEO, ISS. While there is no noticeable relationship in the orbit GTO.



Launch Success Yearly Trend

- We can observe that there was an impressive increase in success rate from 2013 to 2020.



All Launch Site Names

- We applied the SELECT DISTINCT query to access all launch site names

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

Launch Site Names Begin with 'CCA'

Out[13]:

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

- We used the LIKE query function to get launch site names that begins with CAA

Total Payload Mass

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- We used the sql query below to find the average payload mass of booster version F9.

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) \
    FROM SPACEXTBL \
    WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

```
In [39]: %%sql SELECT MIN(Date) AS FirstSuccessfull_landing_date
                FROM SpaceX
                WHERE Landing__outcome LIKE 'Success (ground pad)'

 * ibm_db_sa://ggy48494:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.

Out[39]: firstsuccessfull_landing_date
                      2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- These are the successful drone ship landing with a payload between the range of 4000 and 6000.

In [40]:

```
%%sql SELECT booster_version AS Succsess_booster
      FROM SPACEX
      WHERE Landing__outcome LIKE 'Success (drone ship)'
          AND Payload_mass__kg_ > 4000
          AND Payload_mass__kg_ < 6000 ;
```

```
 * ibm_db_sa://ggy48494:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

Out[40]:

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

Total Number of Successful and Failure Mission Outcomes

- We used wildcard like '%' to filter for WHERE MissionOutcome was a success or a failure
- The total number of successful mission outcome is 100.
- The total number of failed mission outcome is 1.

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

Boosters Carried Maximum Payload

- Using a subquery, we got the list of boosters that carried the maximum payload.

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

2015 Launch Records

- We used a combinations of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015.

In [52]:
```
%%sql SELECT booster_version, launch_site, landing__outcome
      FROM SPACEX
      WHERE landing__outcome LIKE 'Failure (drone ship)'
        AND DATE BETWEEN '2015-01-01' AND '2015-12-31';
```

```
 * ibm_db_sa://ggy48494:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

Out[52]:

booster_version	launch_site	landing_outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2017-03-20.
- We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order.

In [55]:

```
%%sql SELECT Landing__outcome, COUNT(Landing__outcome)
        FROM SPACEX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY Landing__outcome
        ORDER BY COUNT(Landing__outcome) DESC ;
```

```
 * ibm_db_sa://ggy48494:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

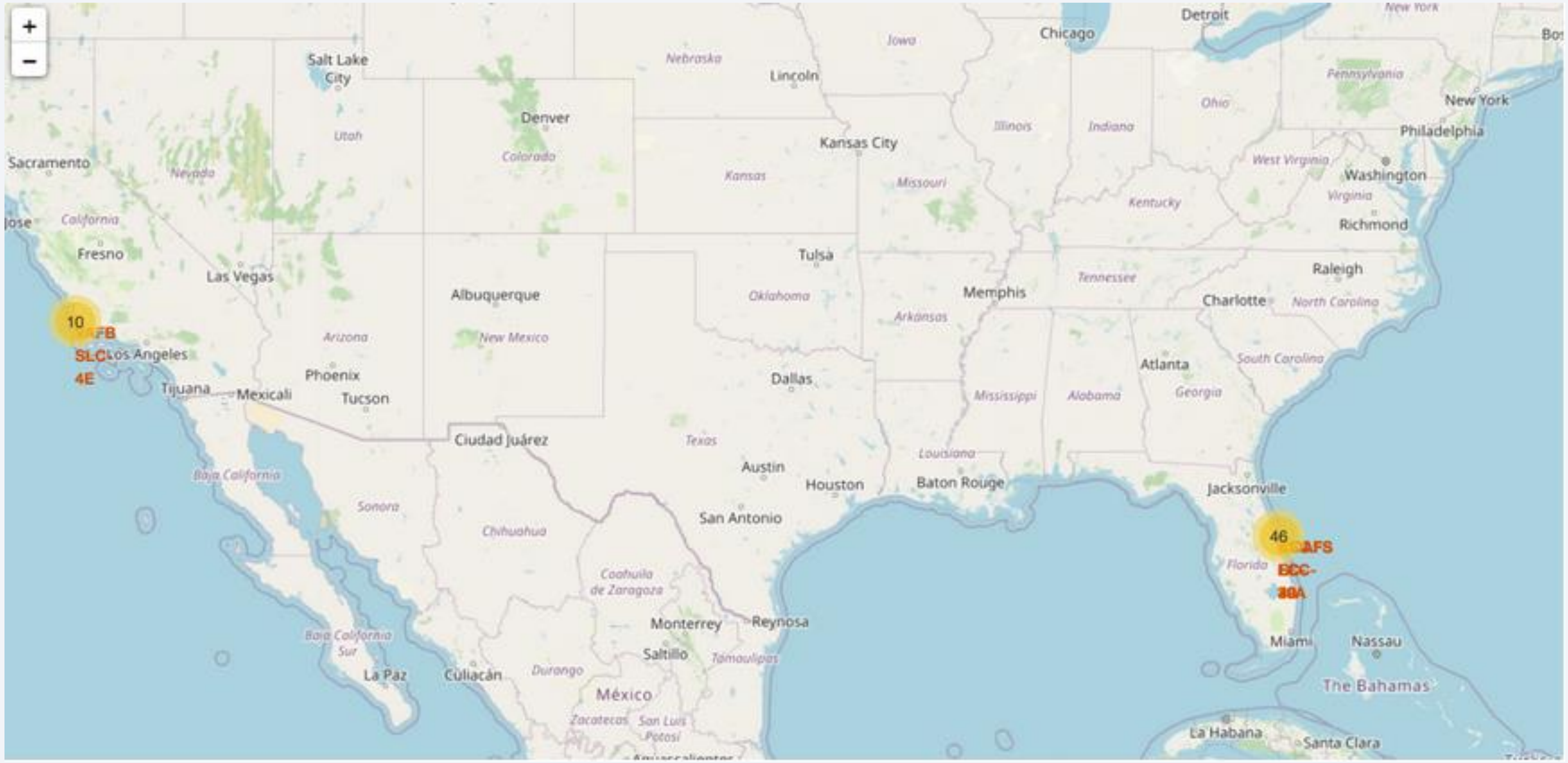
Out[55]:

landing__outcome	2
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

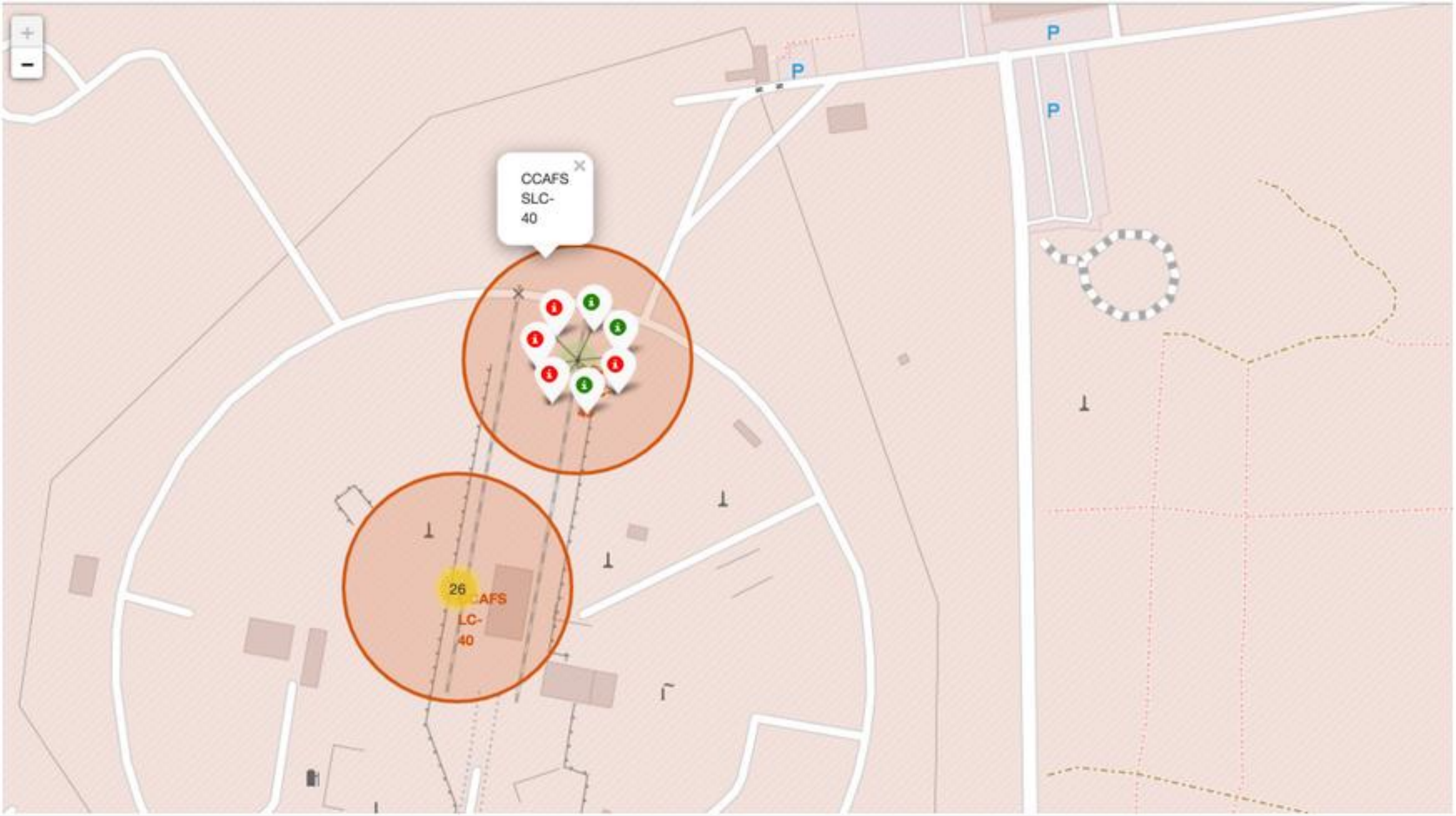
Section 3

Launch Sites Proximities Analysis

All launch sites global map markers



Markers showing launch sites with color labels

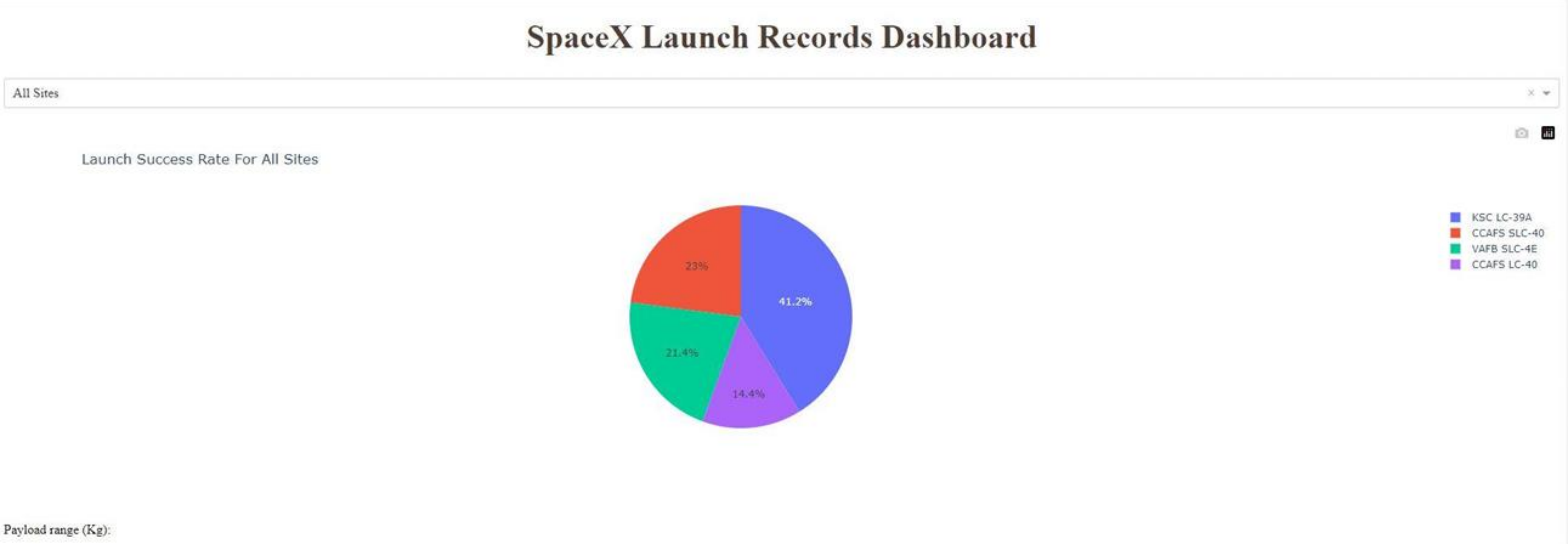


Launch site distance to landmarks

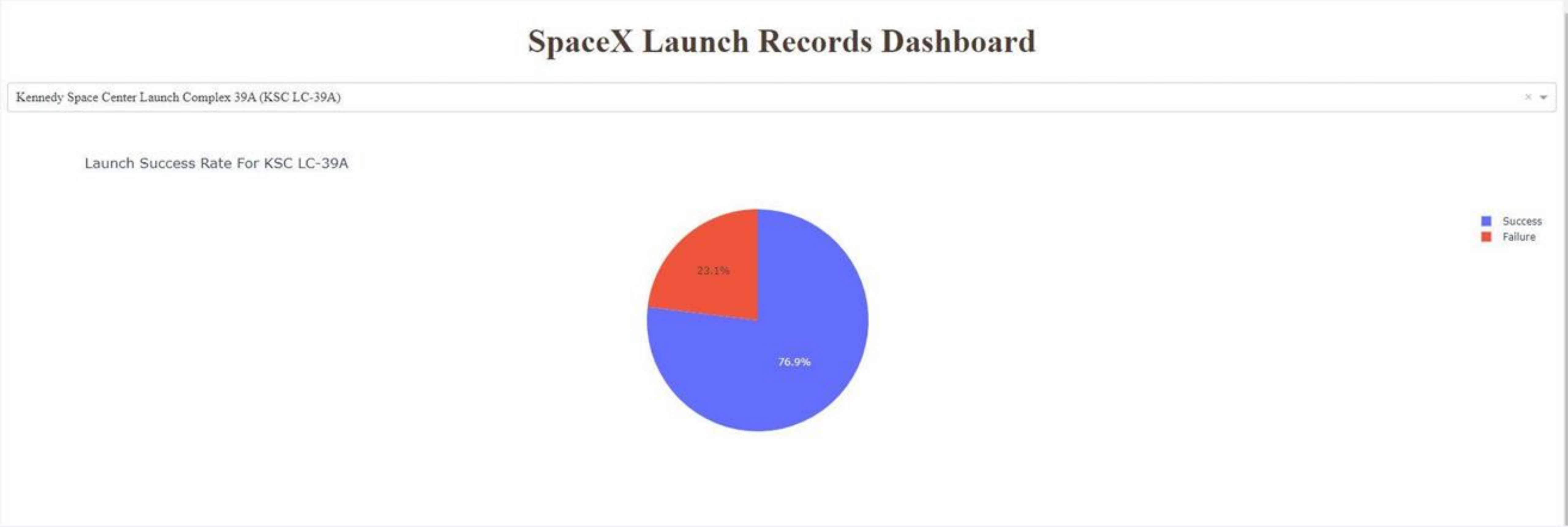
Section 4

Build a Dashboard with Plotly Dash

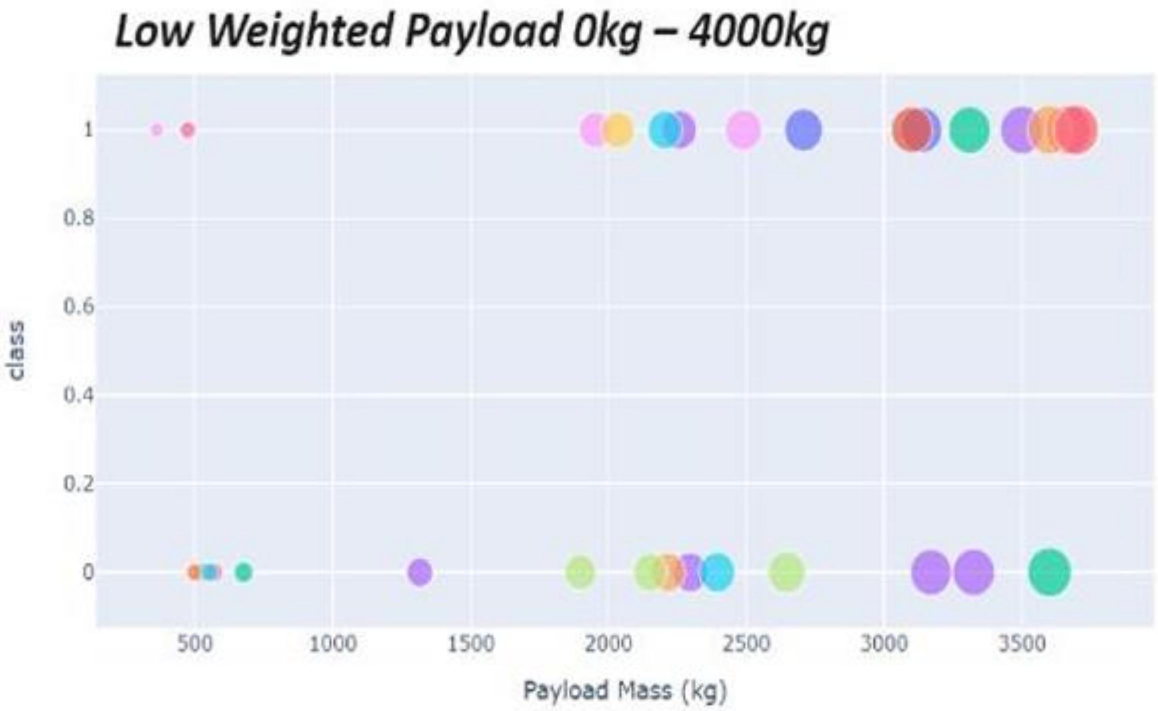
Launch Success For All Sites

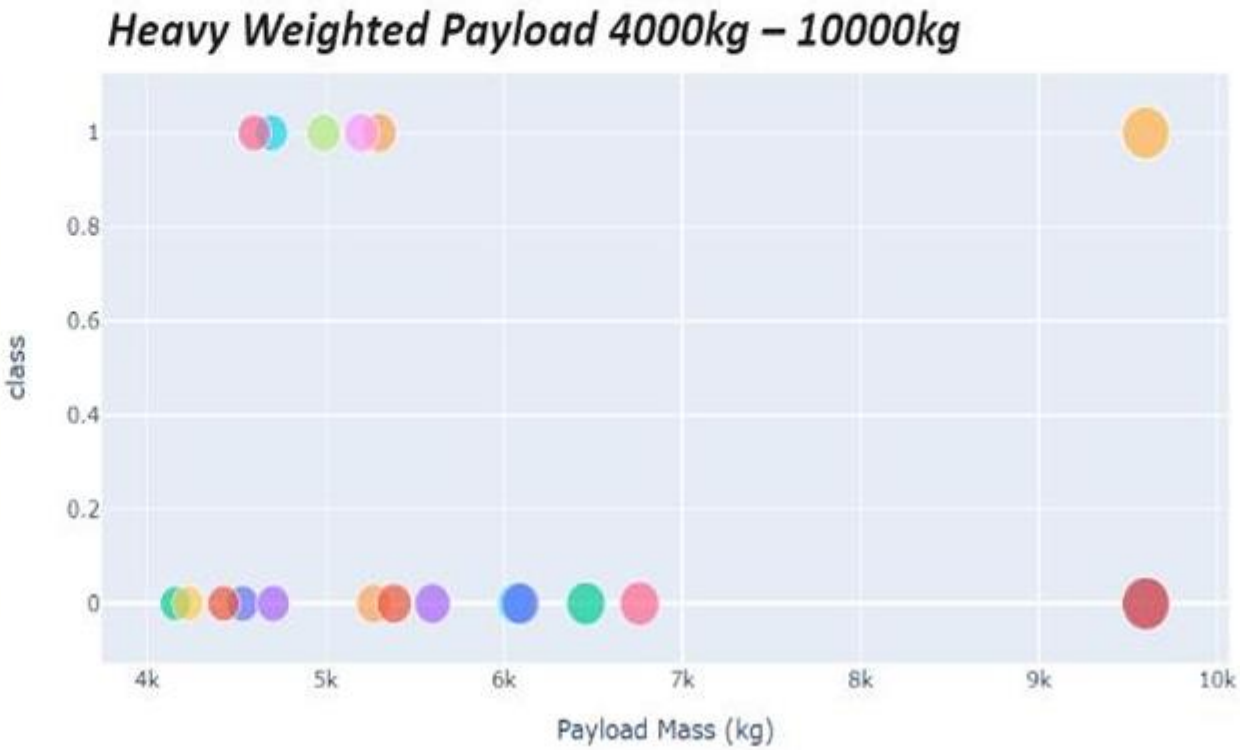


Launch Site with highest launch success ratio



Payload vs Launch outcome for all sites with different payload selection in range slider





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

Section 5

Predictive Analysis (Classification)

Classification Accuracy

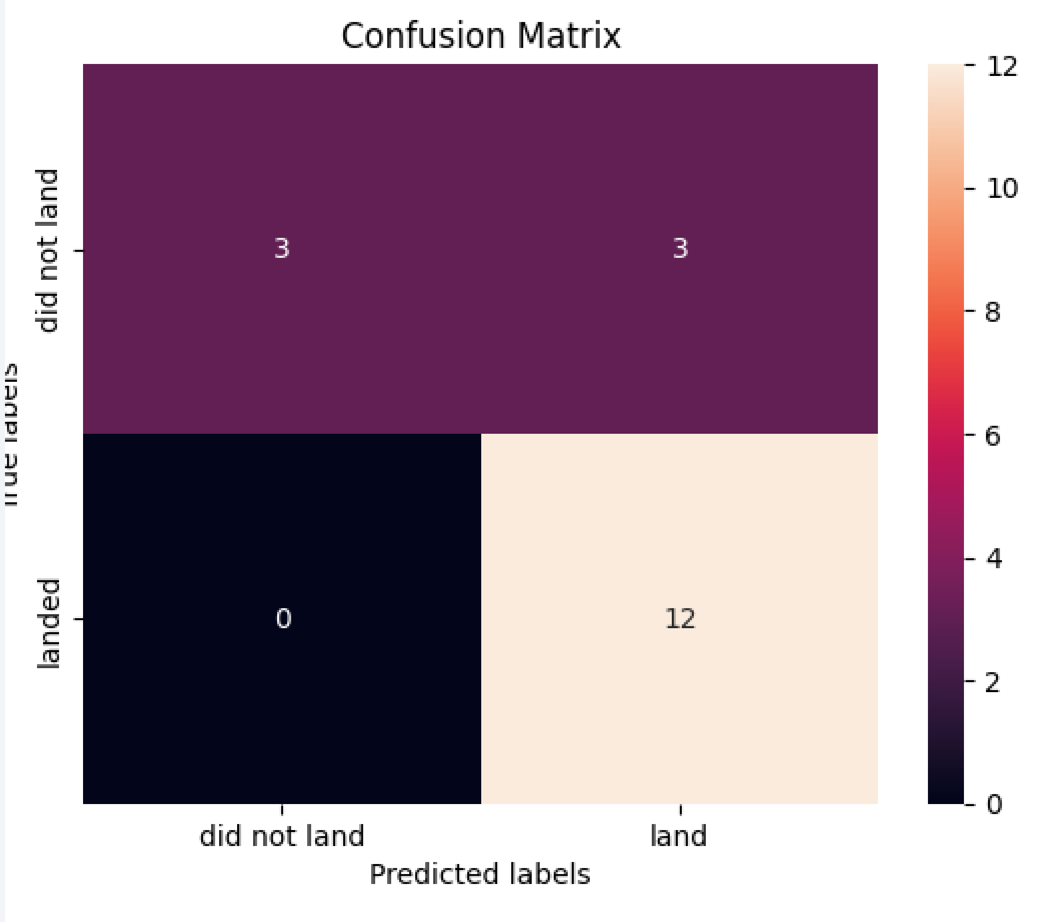
- Due to the small size of our test data, our different models would give similar accuracy score.
- Only the Decision Tree method gave a different accuracy score

```
In [32]: print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
         print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
         print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
         print('Accuracy for K nearsdt neighbors method:', knn_cv.score(X_test, Y_test))

Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.6111111111111112
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

Confusion Matrix

- Due to the small size of the test data, almost all the models had the same accuracy score and confusion matrix. • The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier



Conclusions

From all the analysis done so far, we can conclude that:

- The higher/ larger the amount of flight at a launch site, the higher the number of successful launches or the greater the success rate.
- Launch site KSC LC-39A has the highest the number of successful launches.
- Orbit types : ES-L1, GEO, HEO and SSO had the highest success rate.
- There was an increase in success rate from 2013 to 2020.
- The total number of successful outcome is 100.

Thank you!