IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sala Adrian-Lucian
13-08-2023

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 from Machine Learning Lab

Introduction

SpaceX has fundamentally transformed the space industry with its groundbreaking innovations. They've notably reduced the cost of rocket launches, offering Falcon 9 launches for as little as $62 million, in stark contrast to other providers who charge upwards of $165 million per launch. A significant portion of these savings is attributed to SpaceX's pioneering approach of reusing the first stage of the rocket. By successfully landing and refurbishing it for subsequent missions, they can further drive down costs. As a data scientist for a startup competing with SpaceX, the objective of this project is to design a machine learning pipeline that predicts the landing outcomes of first-stage rockets in future missions. Determining this is essential for our company to strategically price our bids against SpaceX for rocket launches.

Key challenges encompass:

- Pinpointing all variables that have a bearing on the landing result.
- Understanding the interplay between these variables and their impact on the landing outcome.
- Identifying optimal conditions to maximize the likelihood of a successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
  - Data was processed using one-hot encoding for 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

- The data required for this project was collects as follows:
  - Data was collected using the get request to the SpaceX API;
  - The retrieved data was shaped as a Json structure, being transformed into dataframe via normalization;
  - The next steps performed on the data focused on cleaning the data, checking for missing values and filling missing values where required;
  - The data was enhanced with data from Wikipedia for Falcon 9 lunch records, retrieved via web scraping using BeautifulSoup;

Data Collection – SpaceX API

- The get request was used to to collect data from the SpaceX API. The collected data was cleaned, and some did some basic data wrangling and formatting was applied.
- Link to GitHub notebook: https://github.com/LucianSala/SpaceX-Data-Science-Project/blob/main/Lab%201%20Collecting%20de%20Data%20-%20API%20Lab.ipynb

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

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

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

```
# Lets take a subset of our dataframe keeping only the features we want a
nd the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number',
'date_utc']]

# We will remove rows with multiple cores because those are falcon rocket
s with 2 extra rocket boosters and rows that have multiple payloads in a
single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the s
ingle value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then ex
tracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Collection - Scraping

- The Falcon 9 launch records data was web scrapped using BeautifulSoup, normalized and coverted to a df
- Link to GitHub notebook: https://github.com/LucianSala/SpaceX-Data-Science-Project/blob/main/Lab%203%20Data%20Collection%20%26%20Web%20Scraping.ipynb

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response te
xt content
soup = BeautifulSoup(data,'html.parser')
```

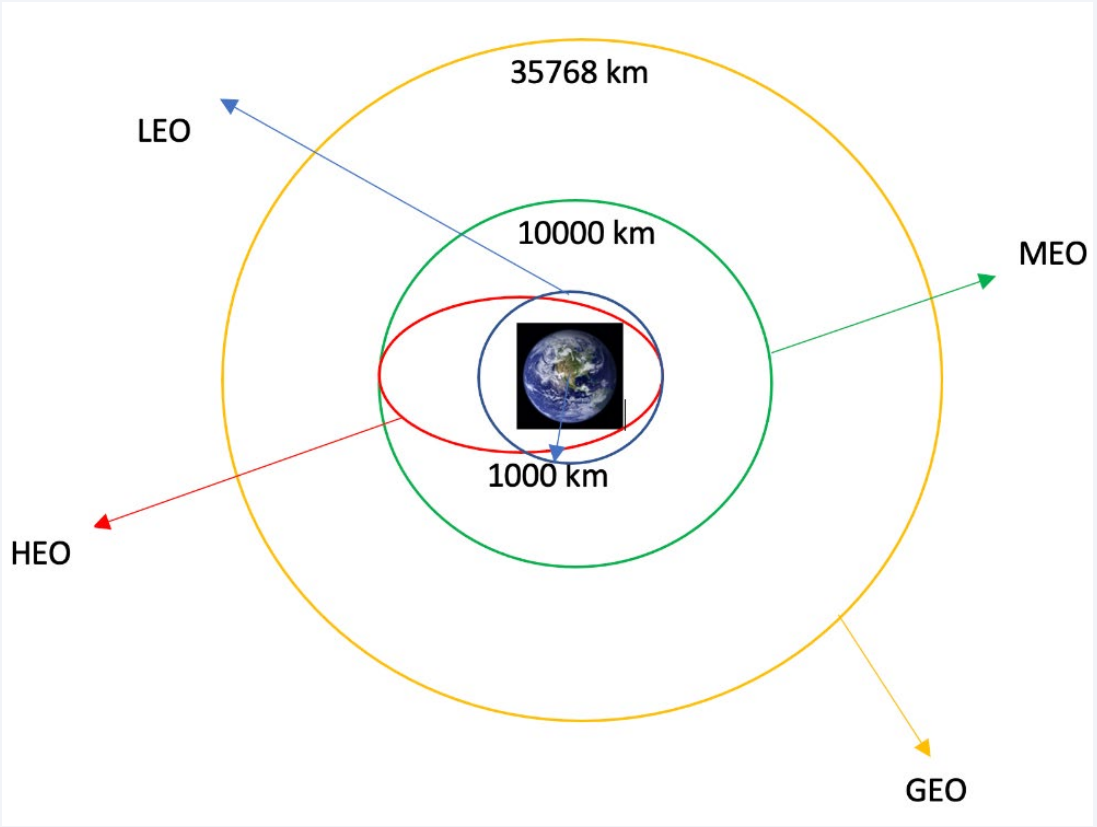
In [10]:

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
temp = soup.find_all('th')
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
for x in range(len(temp)):
    try:
        name = extract_column_from_header(temp[x])
        if (name is not None and len(name)>0):
            column_names.append(name)
    except:
        pass
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
```

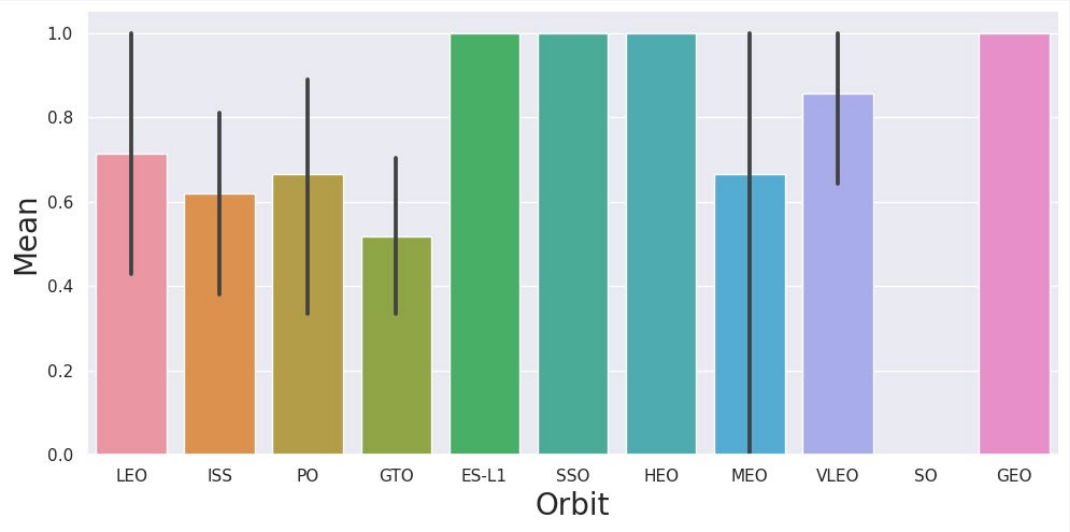
Data Wrangling

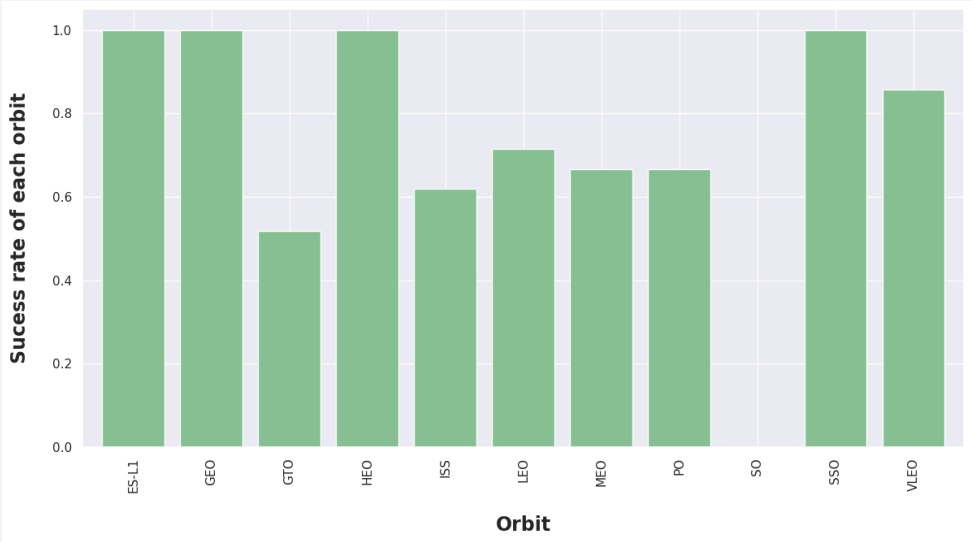
- Data Wrangling was performed to prepare the data for analysis
- EDA was performed, following by labeling of the data
- The number of launches was calculated for each site, followed by the number of occurrences of mission outcomes per orbit type. The landing outcome was labeled for future processing.
- GitHub Link: https://github.com/LucianSala/SpaceX-Data-Science-Project/blob/main/Lab%202%20Data%20Wrangling%20(EDA).ipynb



EDA with Data Visualization

- The data was explored by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type.





- GitHub link: https://github.com/LucianSala/SpaceX-Data-Science-Project/blob/main/Lab%205%20EDA%20using%20Graphs.ipynb

EDA with SQL

- Using SQL, a series of queries the following points can be made about the data:
  - 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

- GitHub link: https://github.com/LucianSala/SpaceX-Data-Science-Project/blob/main/Lab%204%20EDA%20using%20SQL.ipynb

Build an Interactive Map with Folium

- All the launch sites have been marked with additional map objects being added (circles, lines), to mark the success or failure of launches for each site on the folium map.
- All launch outcomes have been assigned to a class (0 for failure, or 1 for success), with red and green markers on the map in MarketCluster().
- The distance of the launch sites have been calculated to different points of interest to answer the following questions:
  - How close the launch sites with railways, highways and coastlines?
  - How close the launch sites with nearby cities?
- GitHub link: https://github.com/LucianSala/SpaceX-Data-Science-Project/blob/main/Lab%206%20Interactive%20Visualizations%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- An interactive dashboard was built with Plotly dash which allowing the user to explore the data.
- Pie charts have been used to show the total launches by site.
- A scatter graph was used to show the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- GitHub link: https://github.com/LucianSala/SpaceX-Data-Science-Project/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- The data was loaded using numpy and pandas, transformed and split into training-test sets.
- A series of ML models have been built and finetuned hyperparameters using GridSearchCV.
- The metrics of the model have been assessed and improved using feature engineering and algorithm tuning.
- Following the above steps, the most well performing classification model was selected.
- GitHub link: https://github.com/LucianSala/SpaceX-Data-Science-Project/blob/main/Lab%207%20ML%20Predictiom%20Lab.ipynb

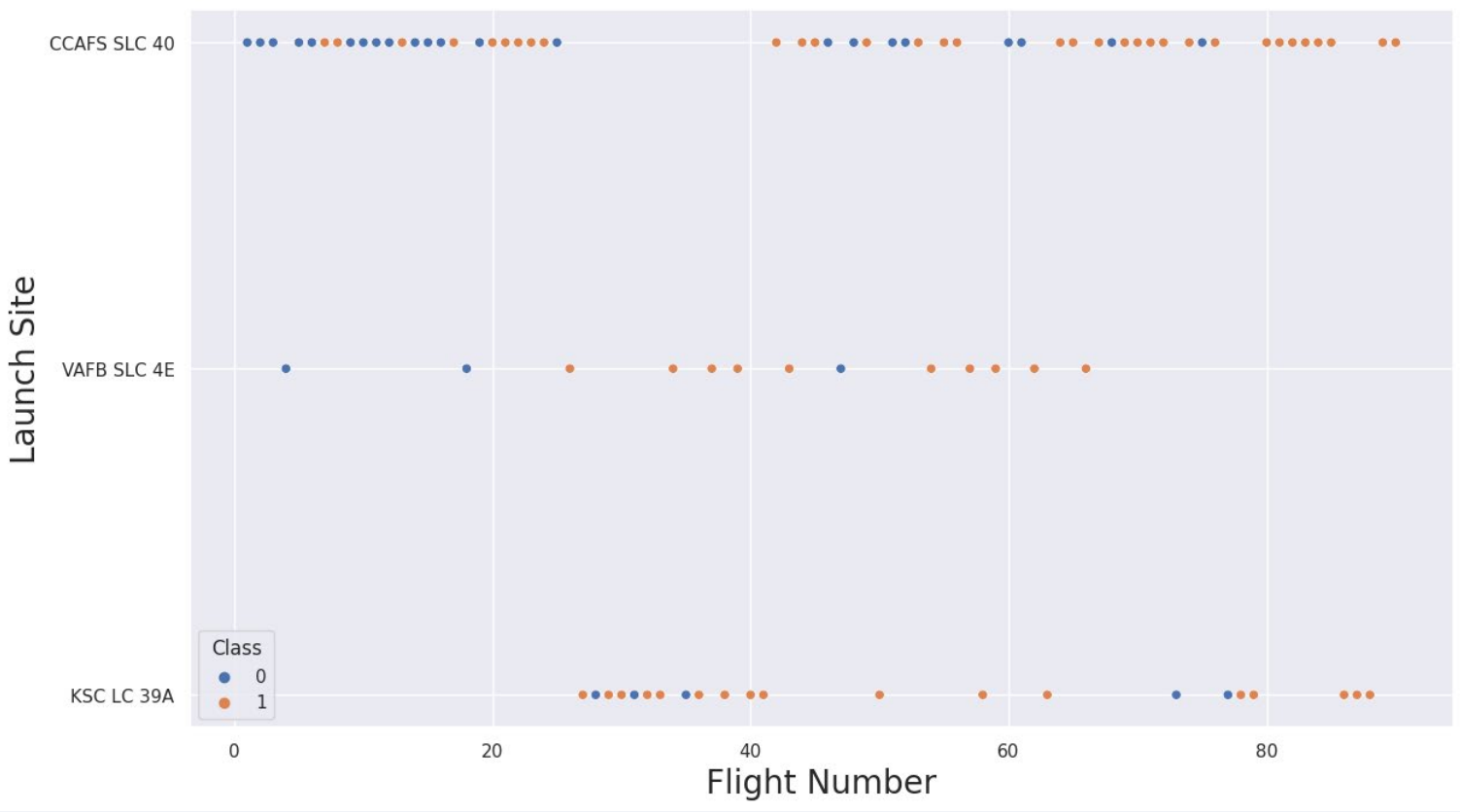
Results

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

Section 2

Insights drawn from EDA

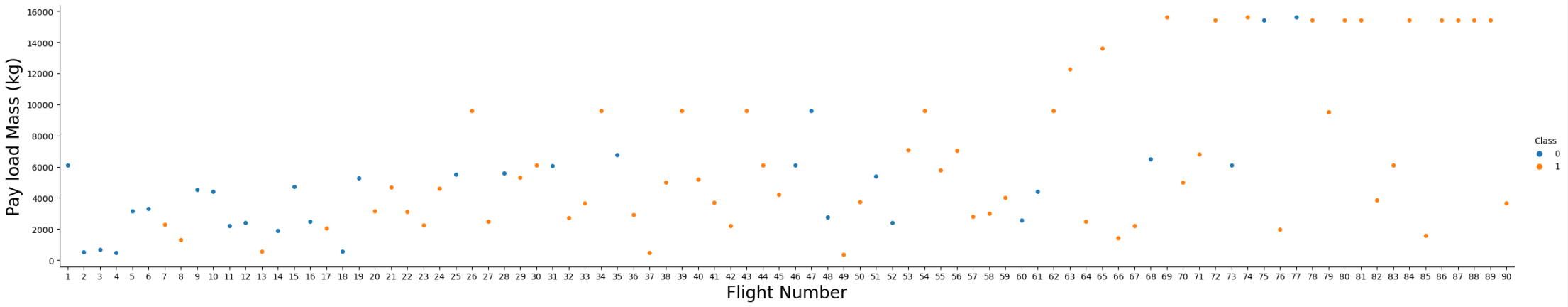
Flight Number vs. Launch Site



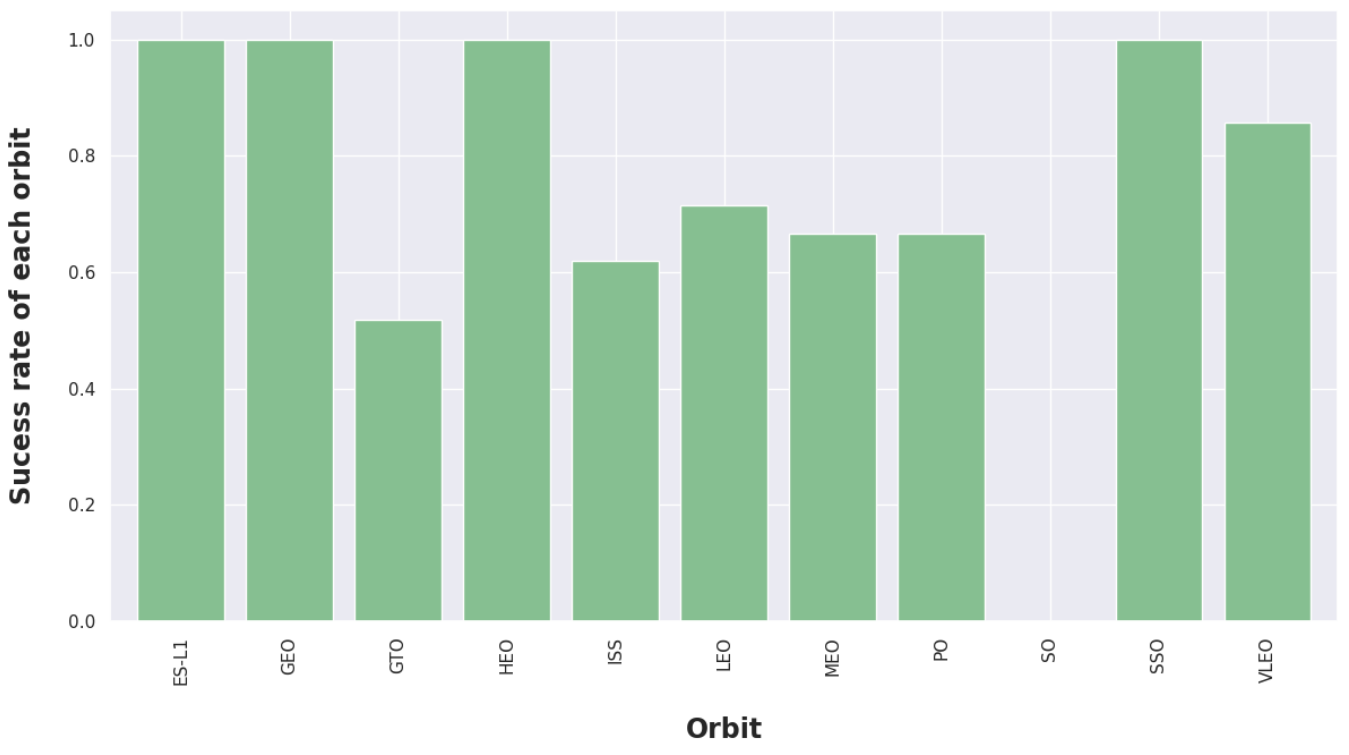
- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.
- However, site CCAFS SLC40 shows the least pattern of this.

Payload vs. Launch Site

- The scatter plot below shows once the pay load mass is greater than 7000kg, the probability of the success rate will be highly increased.
- However, there is no clear observable pattern to say the launch site is dependent to the pay load mass for the success rate.



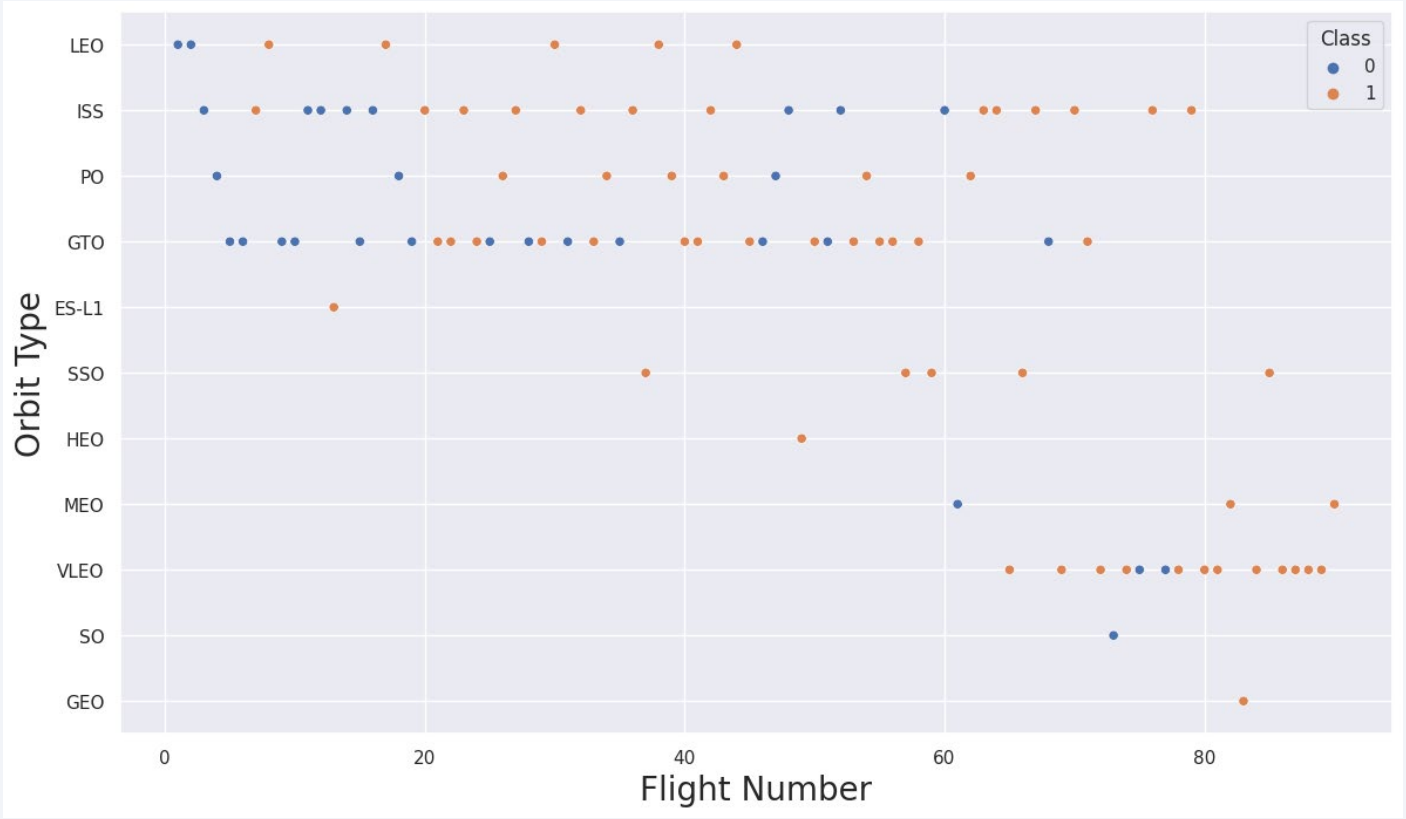
Success Rate vs. Orbit Type



- From the plot on the left, we can see that ES L1, GEO, HEO, SSO, VLEO had the most success rate.

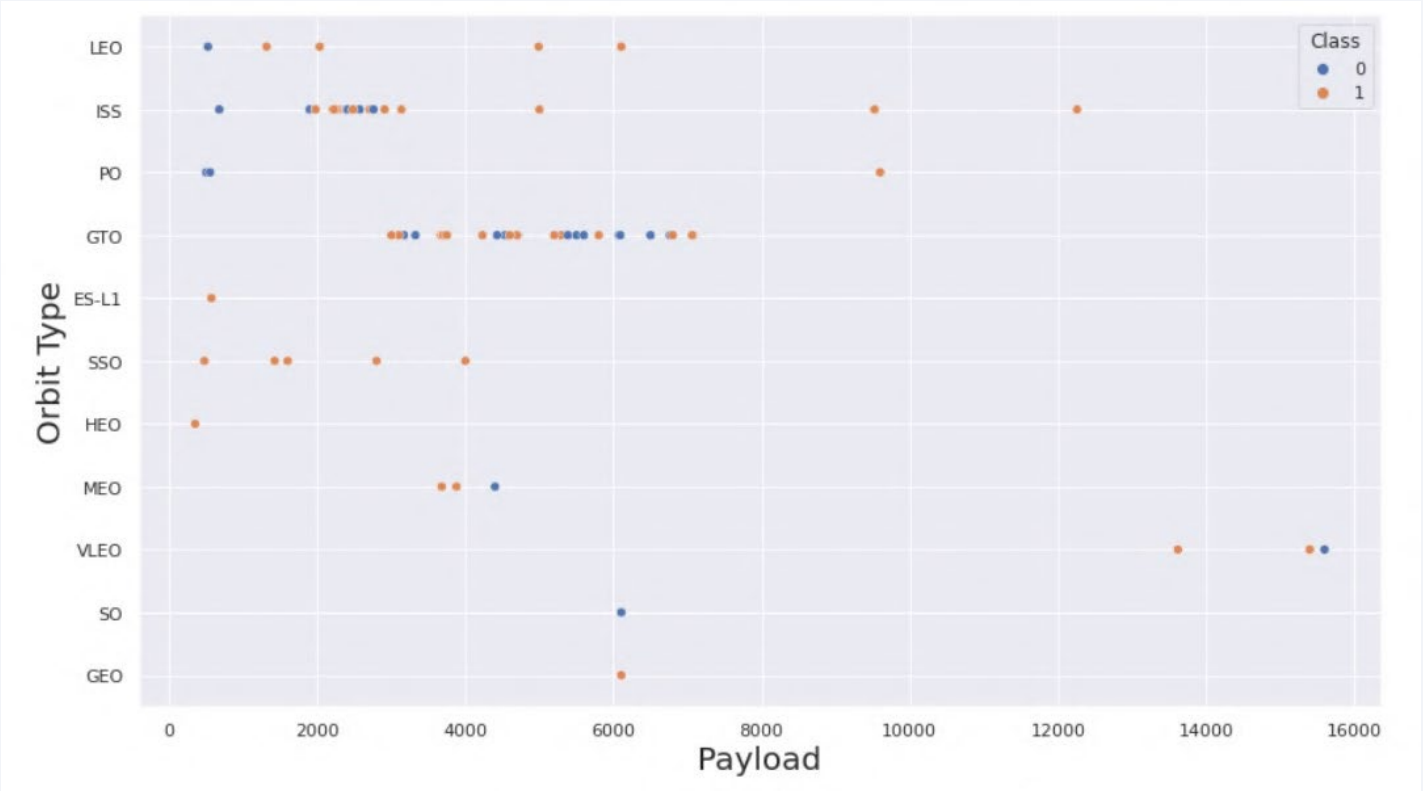
Flight Number vs. Orbit Type

- This scatter plot, on the right, shows that generally, the larger the flight number on each orbits, the greater the success rate except for GTO orbit which depicts no relationship between both attributes.



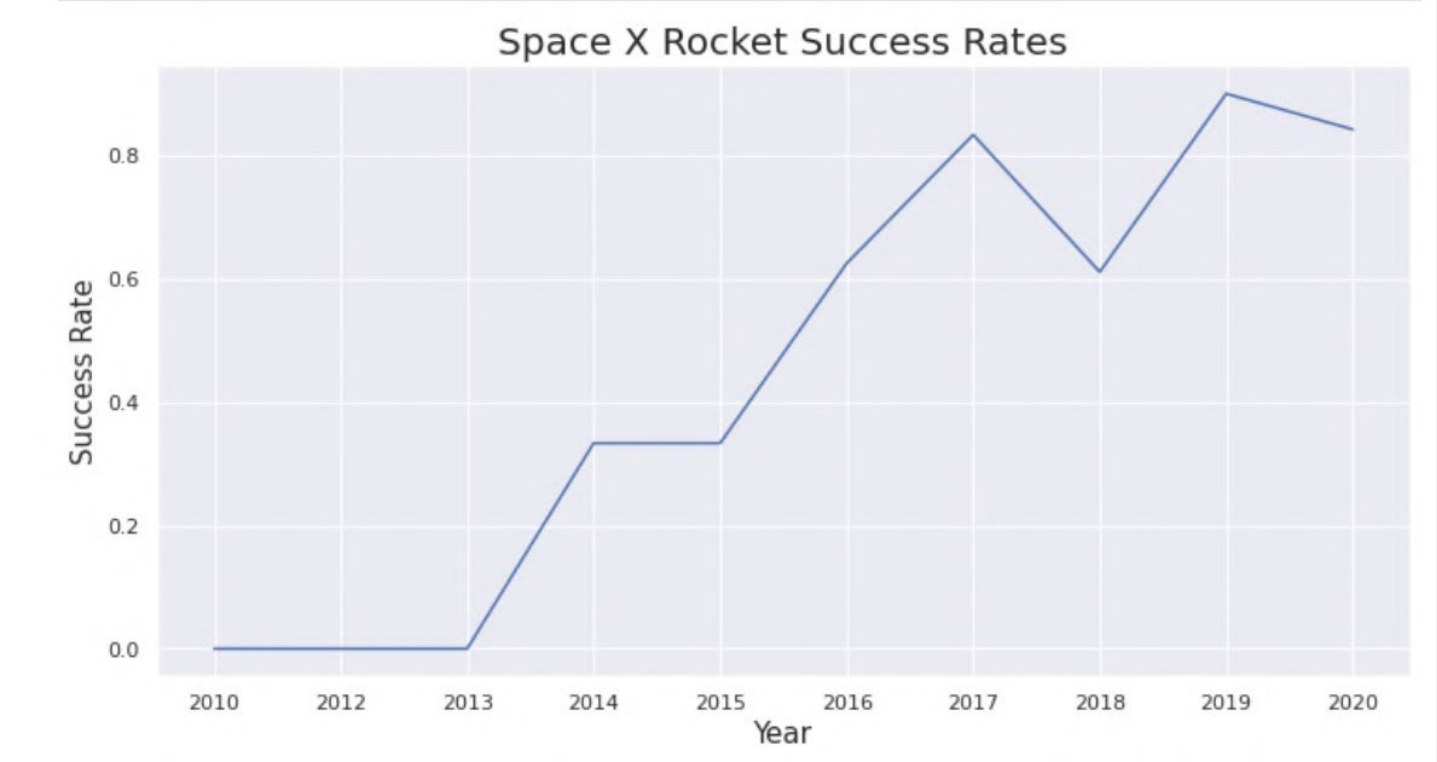
Payload vs. Orbit Type

- From the chart on the right, we can determine that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



Launch Success Yearly Trend

- The figure presents an increasing trend between 2013 and 2020.



All Launch Site Names

- The query used includes the keyword **DISTINCT** and the output of the query is 4 launch sites.

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

In [7]:
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

Out[7]:

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

Launch Site Names Begin with 'CCA'

- The 5 records where launch sites begin with `CCA`, retrieved using the below query:

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

In [8]:
```
%sql SELECT LAUNCH_SITE FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

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

Out[8]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- The total payload carried by boosters from NASA is 45596, extracted using the query below:

```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

- The average payload mass carried by the booster version F9 v1.1 is 2928

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

First Successful Ground Landing Date

- Based on the query below the successful landing outcome on the ground pad was 22nd Dec 2015

In [14]:
```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- The **WHERE** clause was used to filter for boosters which have successfully landed on the drone ship and applied the **AND** condition to determine successful landing with payload mass that was >4000 and 6000<

```
In [12]: %sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND F
 * sqlite:///my_data1.db
Done.
```

Out[12]:

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

Total Number of Successful and Failure Mission Outcomes

- The wildcard '**%**' symbol was used to filter for **WHERE** MissionOutcome

List the total number of successful and failure mission outcomes

```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

```
The total number of successful mission outcome is:
```

	successoutcome
0	100

```
The total number of failed mission outcome is:
```

	failureoutcome
0	1

Boosters Carried Maximum Payload

In [16]:
```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

Out[16]:

Booster Versions which carried the Maximum Payload Mass
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

- The boosters carrying the maximum payload where determined using a subquery

2015 Launch Records

- A combination of **WHERE**, **LIKE**, **AND** and **BETWEEN** clauses have been used to reach the final result of failed landing outcomes on drone ship, their booster versions, and launch site names for year 2015

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

In [18]:
```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- Landing outcome have been selected and a count was performed using the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2017-03-20. GROUP BY and ORDER BY have been used to organize the results.

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

In [19]:
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Section 3

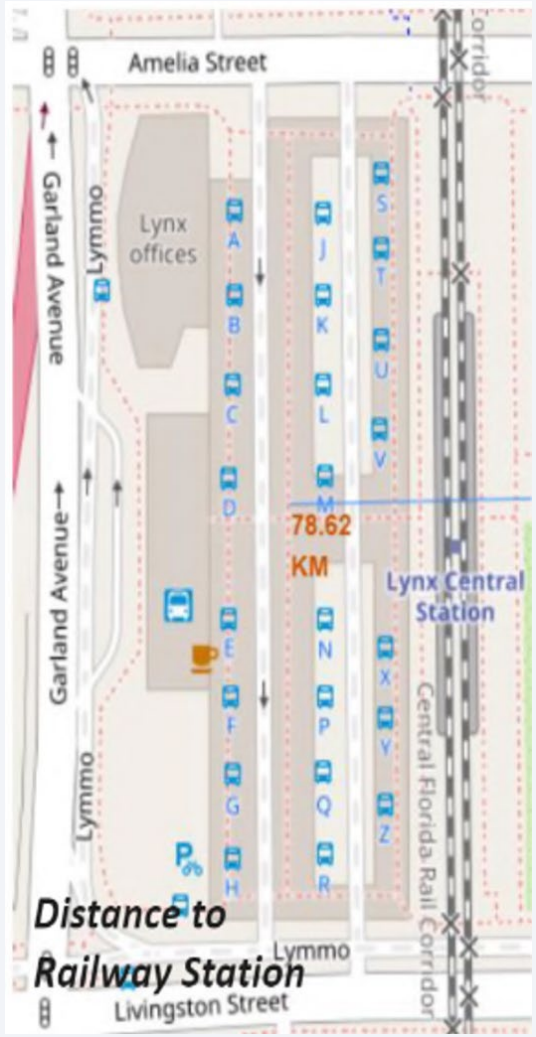
Launch Sites Proximities Analysis

Launch Sites Location

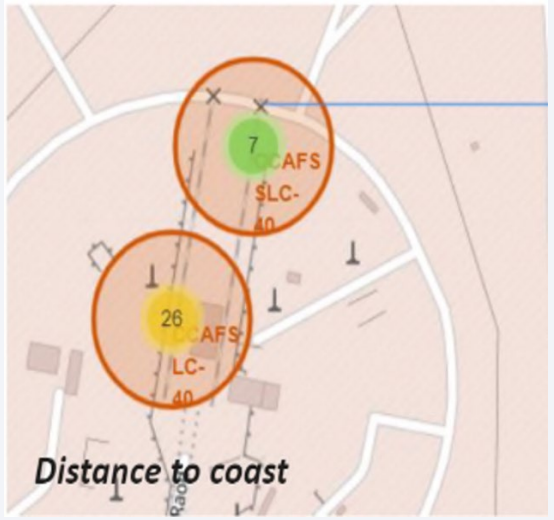
- All launch sites are included within the USA

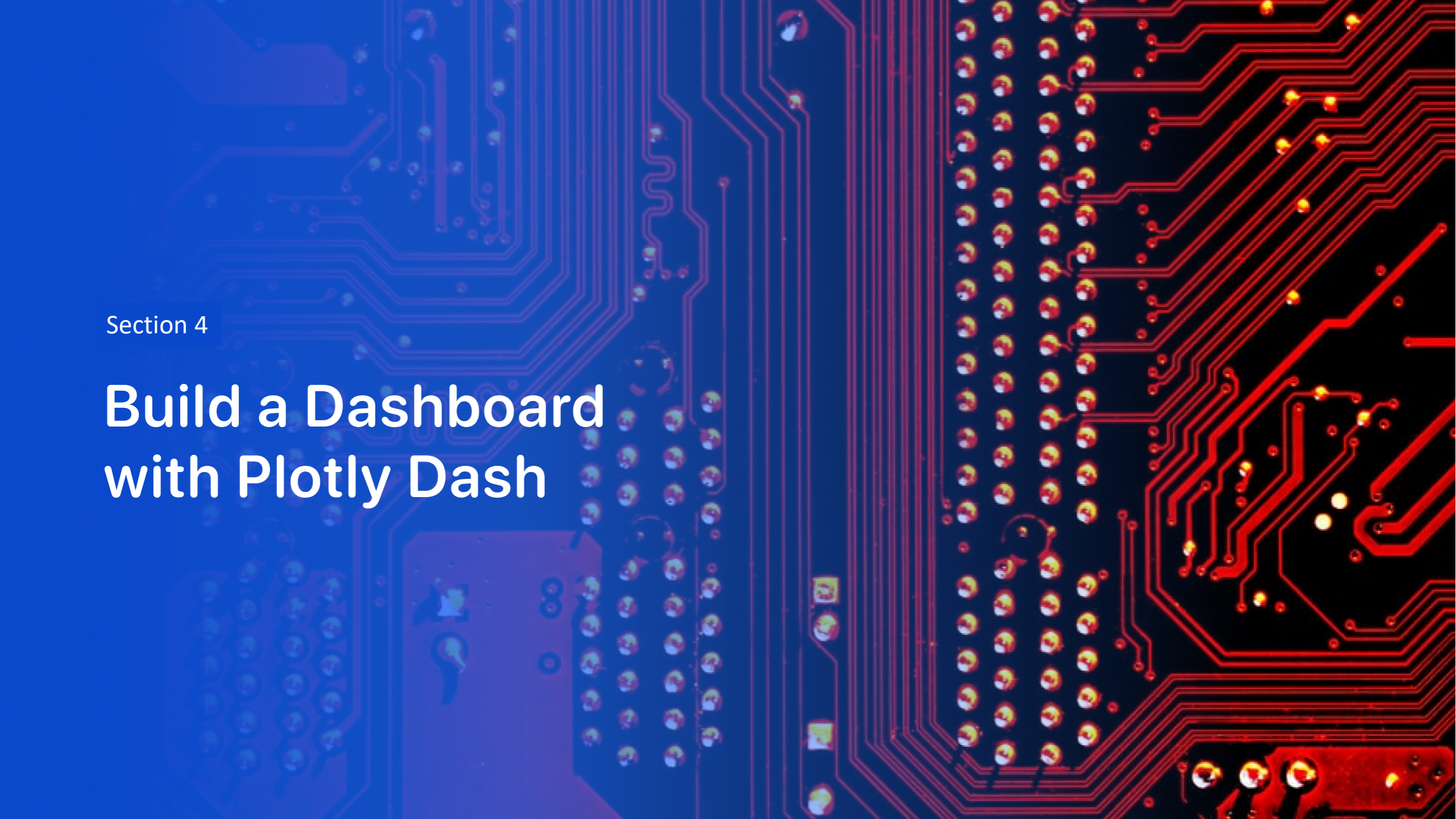
Launch sites are marked with colored labels

Distance from Launch Sites to Landmarks



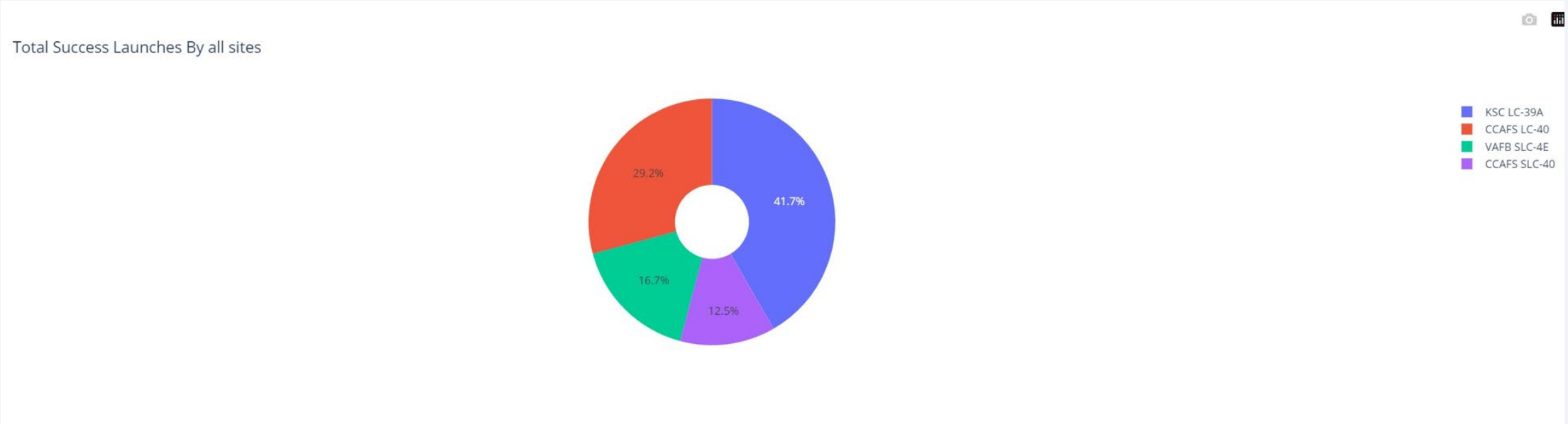
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes




Section 4
Build a Dashboard with Plotly Dash

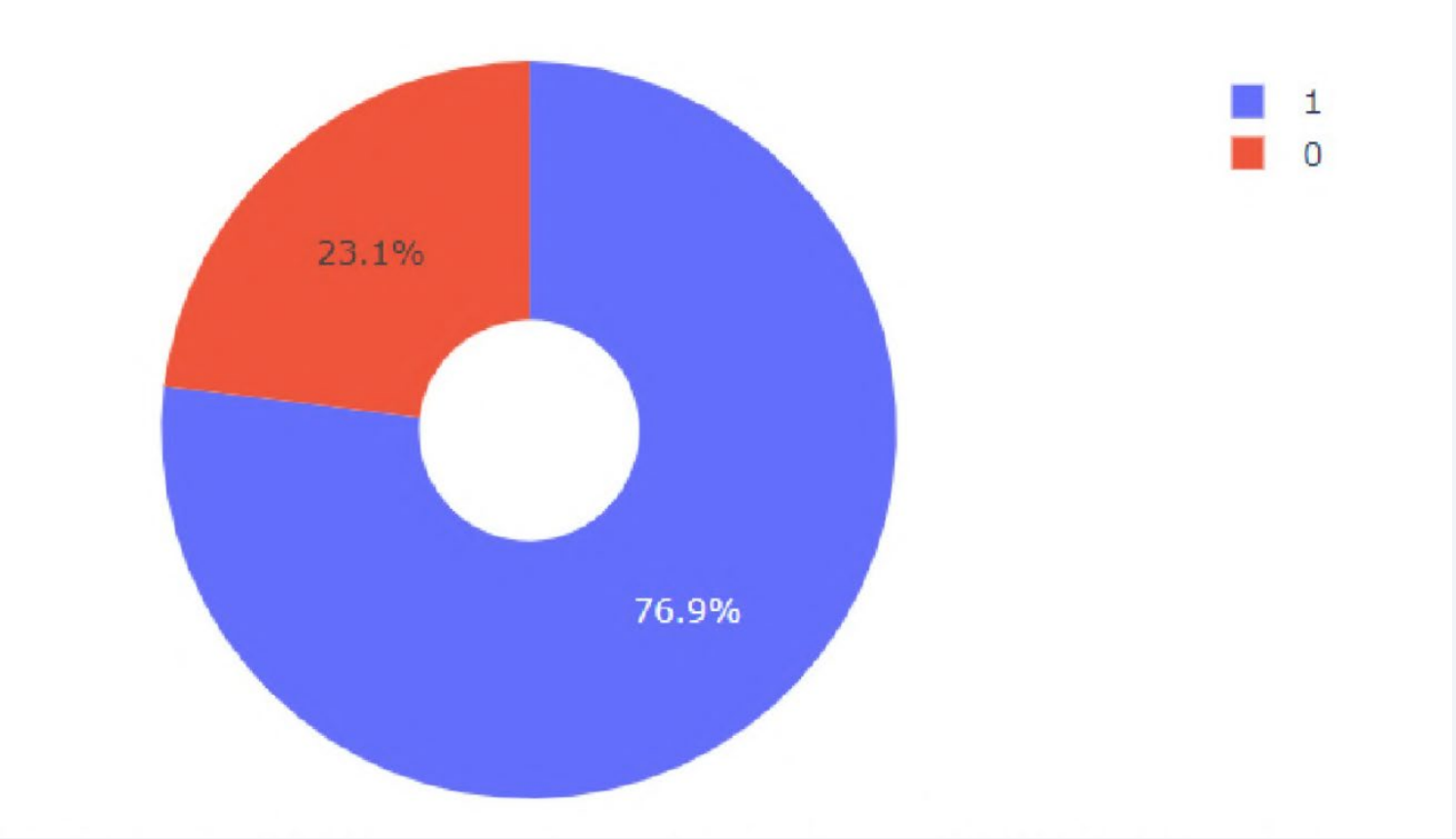
Success by site

- KSC LC-39A had the most successful launches

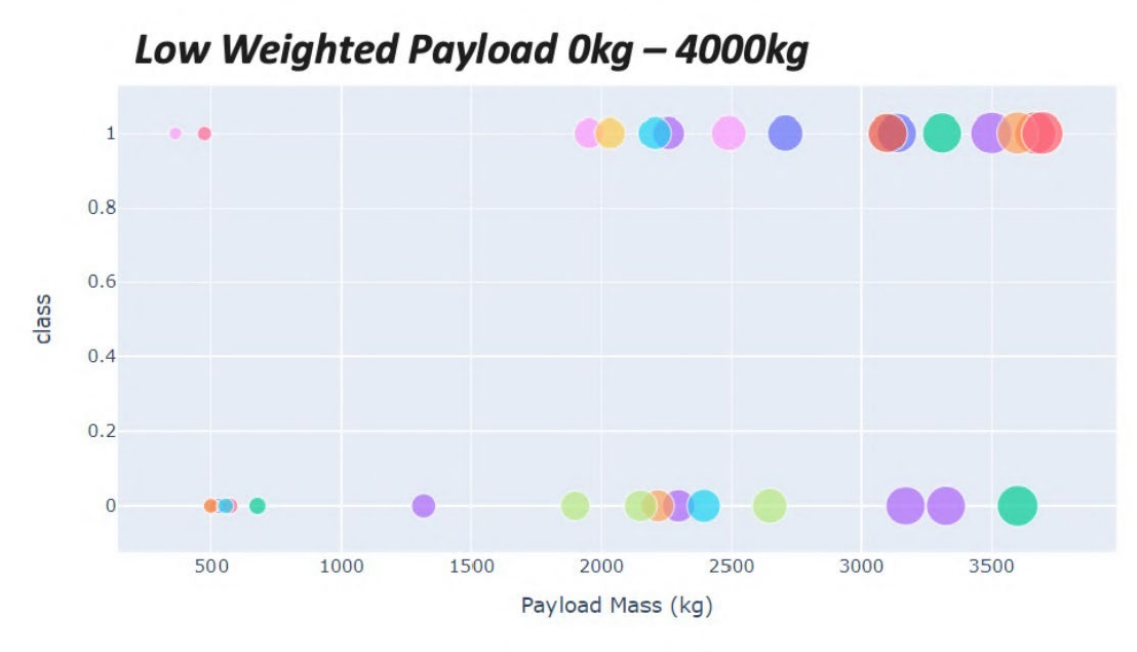


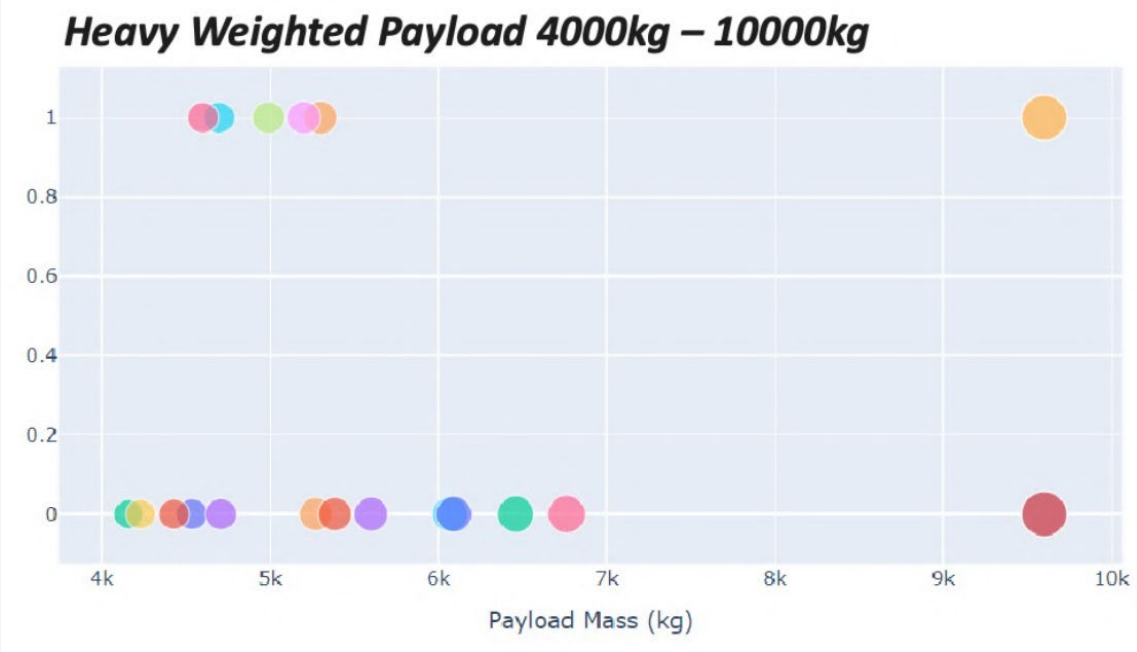
Success/Failure for KSC LC-39A

- The success rate in blue vs the failure rate in red for KSC LC-39A



Payload vs Launch Outcome Scatter Plot





Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Base on the code below and the results obtained, we can summarize that the best algorithm is the Tree Algorithm

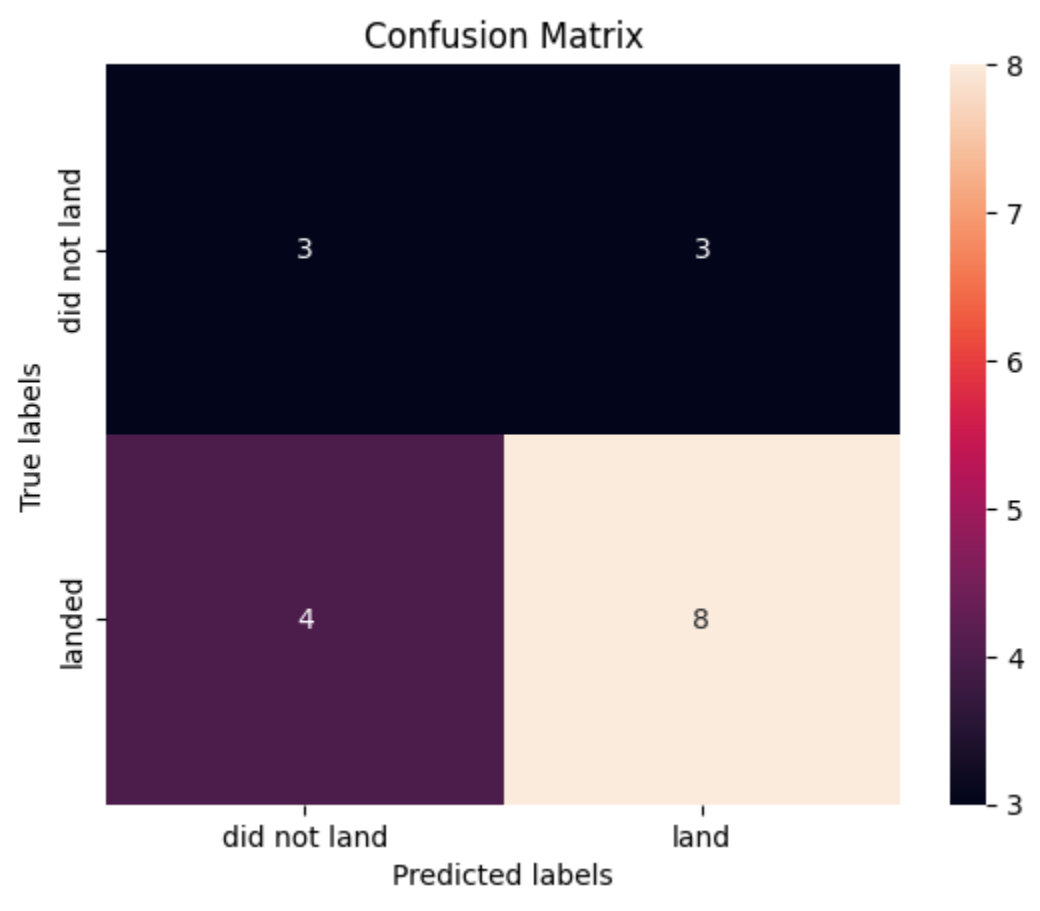
In [38]:

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.875
Best Params is : {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2,
'splitter': 'random'}
```

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., successful landing marked as successful landing by the classifier.



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!