Winning Space Race with Data Science

Kian Mohseni
May 21, 2023

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

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
  - Visual Dashboard with Plotly Dash
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis
  - Interactive Visual Analytics
  - Predictive Modeling

Introduction

- Project background and context:

  Space X 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 we want to find answers:

  1. What factors determine if the rocket will land successfully?
  2. What is the relationship between various features that determine the success rate of a landing?
  3. What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API and web scraping from Wikipedia
- Perform data wrangling
  - One-hot encoding 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
  - Build, tune, and evaluate classification models

Data Collection

API

1. Get data from SpaceX API
2. Convert data into pandas dataframe using json function
3. Filter dataframe by filling in missing values
4. Export to csv file

Web Scraping

1. Web scraping from Wikipedia using BeautifulSoup function
2. Extract HTML table, parse table, and convert into pandas dataframe
3. Export to csv file

Data Collection – SpaceX API

- We called the get request to the SpaceX API to retrieve the data, then converted it to a pandas dataframe, before finally cleaning and filling in any missing values.

- The link to my notebook is https://github.com/kianmessi/SpaceX/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

1. Get request for rocket launch data using API

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- We first used BeautifulSoup to scrape Falcon 9 launch records from Wikipedia, then parsed the data into a pandas dataframe.

- The link to my notebook is https://github.com/kianmessi/SpaceX/blob/main/jupyter-labs-webscraping.ipynb

1. Apply HTTP Get method to request the Falcon 9 rocket launch page

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

In [5]:
```
# 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
```

Out[5]:
```
200
```

2. Create a BeautifulSoup object from the HTML response

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text, 'html.parser')
```

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

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

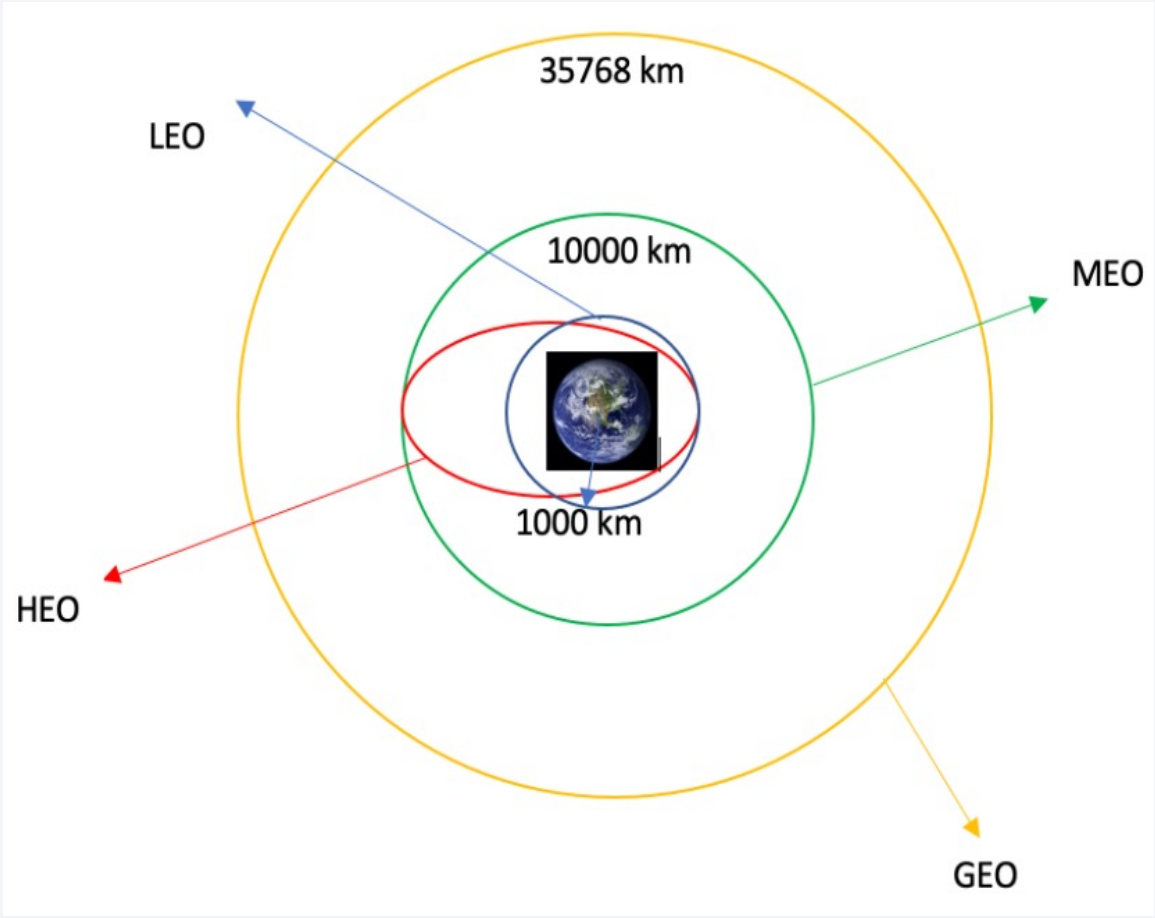
3. Extract all column names from the HTML table header

In [10]:
```
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 column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

4. Create a dataframe by parsing the launch HTML tables

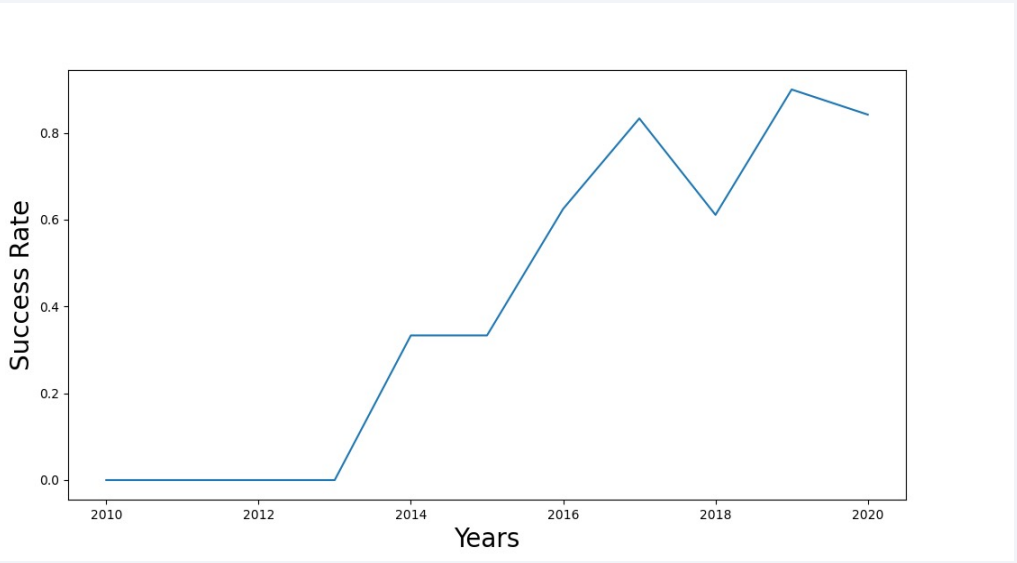
Data Wrangling

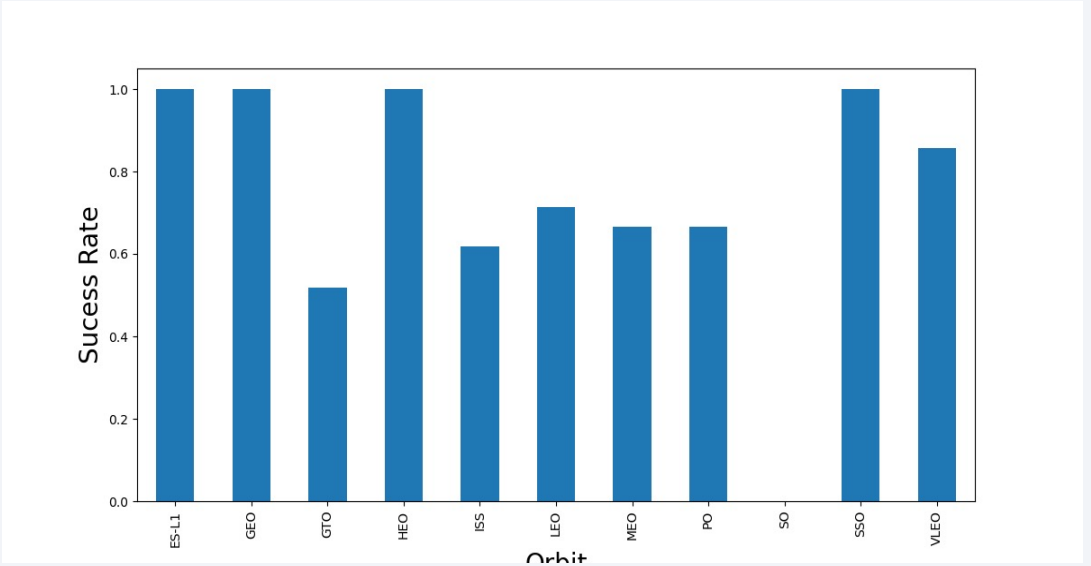
- Exploratory data analysis was used to determine the training labels.
- The number of launches at each site and number of each orbit were calculated
- From the outcome column in the data, we created a landing outcome label and exported the results to a csv
- The link to my notebook is https://github.com/kianmessi/SpaceX/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_1_L3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb



EDA with Data Visualization

- We explored the data by visualizing the landing outcome success rate over the lats 10 years and the compared the success rates of each orbit type.
- The link to my notebook is https://github.com/kianmessi/SpaceX/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb





EDA with SQL

- The following SQL queries were executed to apply EDA on the data:
  - The failed landing outcomes in drone ship, their booster version and launch site names
  - The names of unique launch sites in the space mission
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The total number of successful and failure mission outcomes
  - The average payload mass carried by booster version F9 v1.1
- The link to my notebook is https://github.com/kianmessi/SpaceX/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

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.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate (green being success and red being failure).
- We calculated the distances between a launch site and its proximities, such as urban cities, highways, coastlines, etc.
- The link to my notebook is https://github.com/kianmessi/SpaceX/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

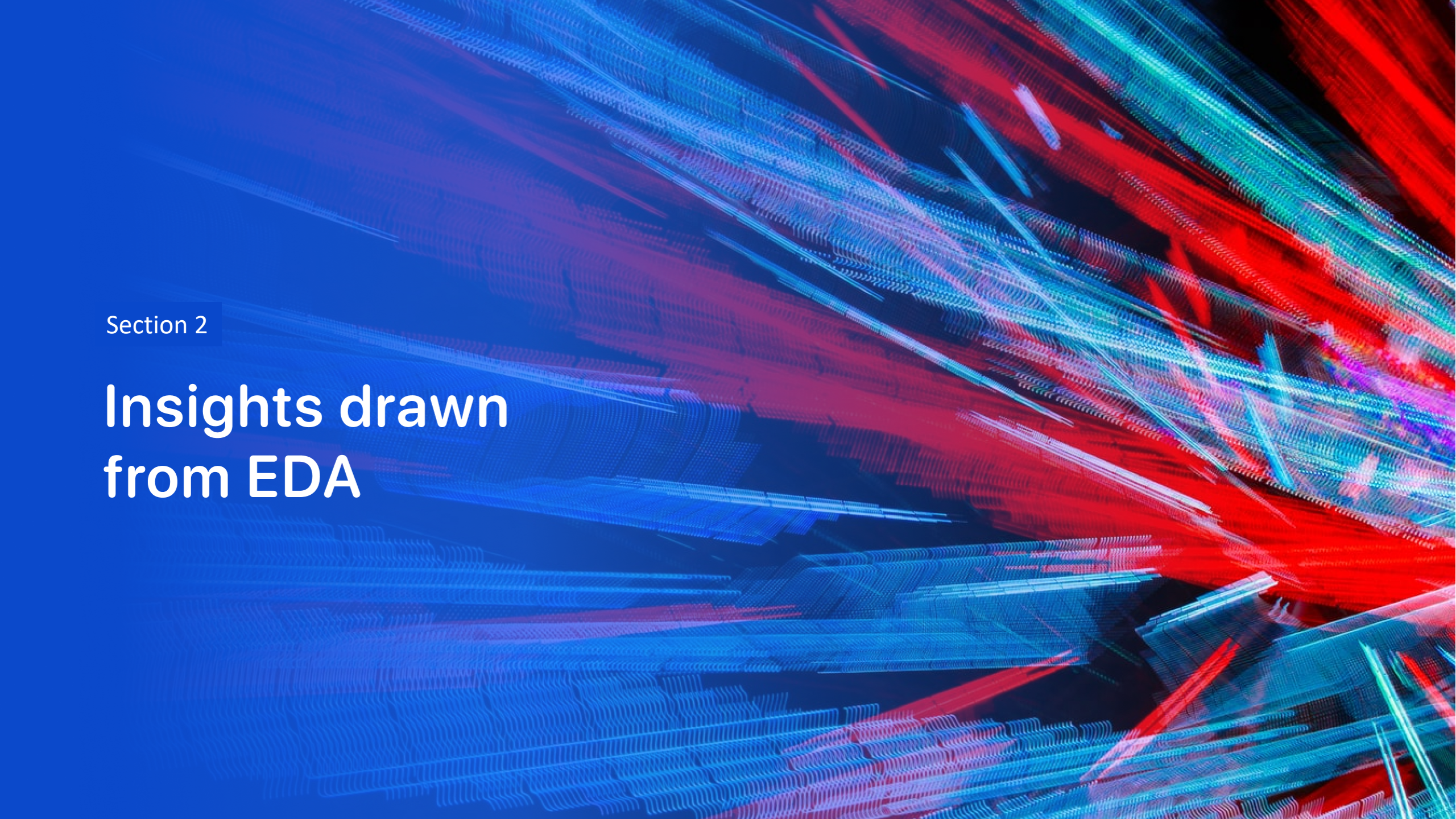
- Pie charts were used to display the total number of launches from each site.
- Scatter plots show the relationship between landing outcome and payload mass for each launch instance.

Predictive Analysis (Classification)

- We built our model by loading our data into a dataframe, transforming the data, splitting the data into training and test sets, set our machine learning algorithms and parameters to GridSearchCV, and fit our data sets to the objects to train our model
- We evaluated the accuracy of each model and determined the best hyperparamters for each type of algorithm before plotting the Confusion Matrix
- We continued to tune our models and found the best performing classification model based on the best accuracy score.
- The link to my notebook is https://github.com/kianmessi/SpaceX/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

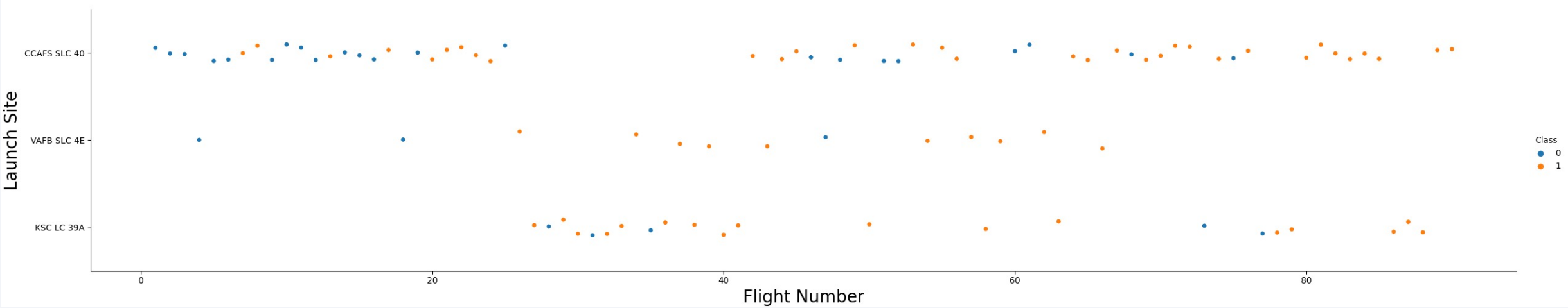
Results

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


Section 2
Insights drawn from EDA

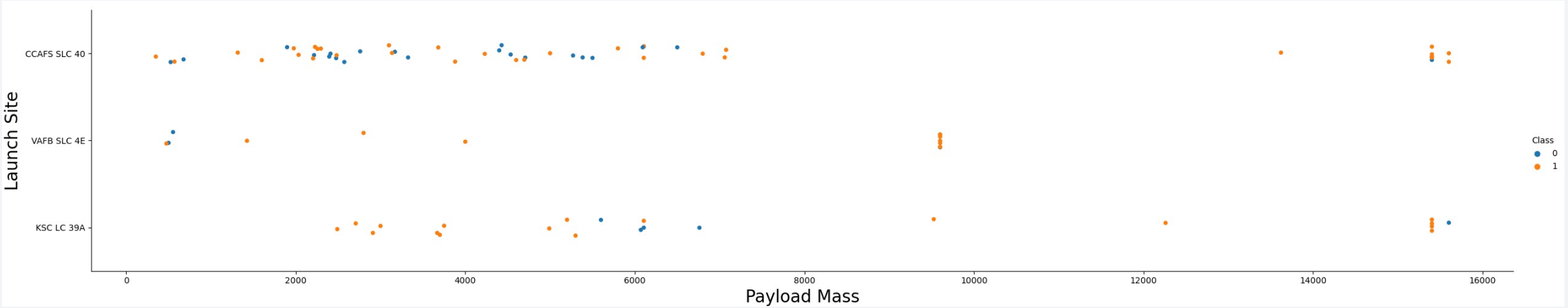
Flight Number vs. Launch Site

- The plot shows us the greater the flight number at a launch site, the greater its success rate



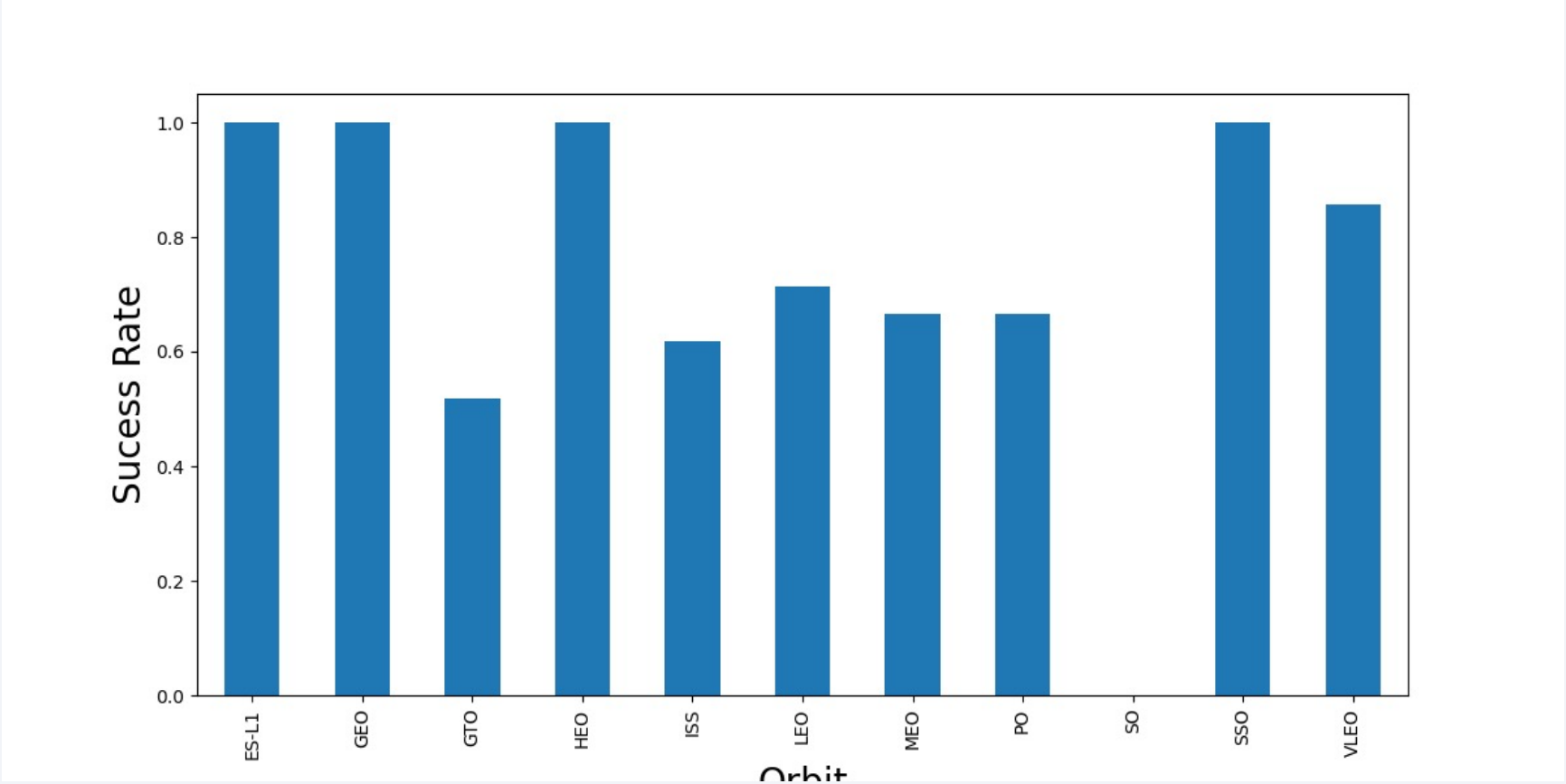
Payload vs. Launch Site

- For each launch site, we can see that as Payload Mass increases, the higher the success rate of the landing generally.



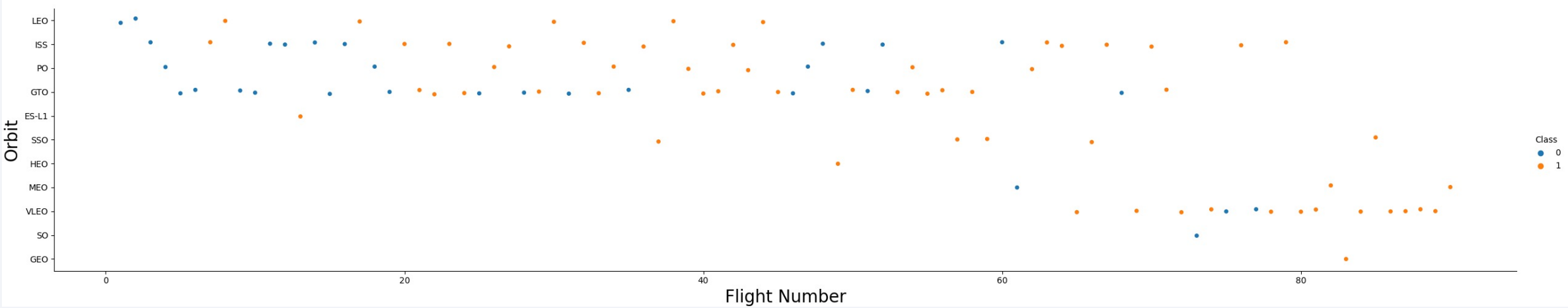
Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, and VLEO are the orbit types with the highest success rates.



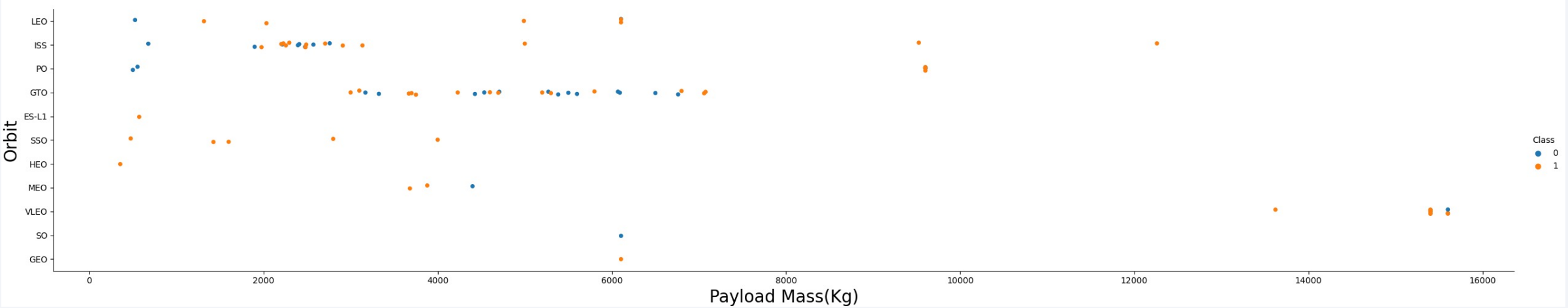
Flight Number vs. Orbit Type

- The plot shows that in the LEO orbit, success is positively correlated to the number of flights however in the GTO orbit, there appears to be no relationship between success rate and number of flights.



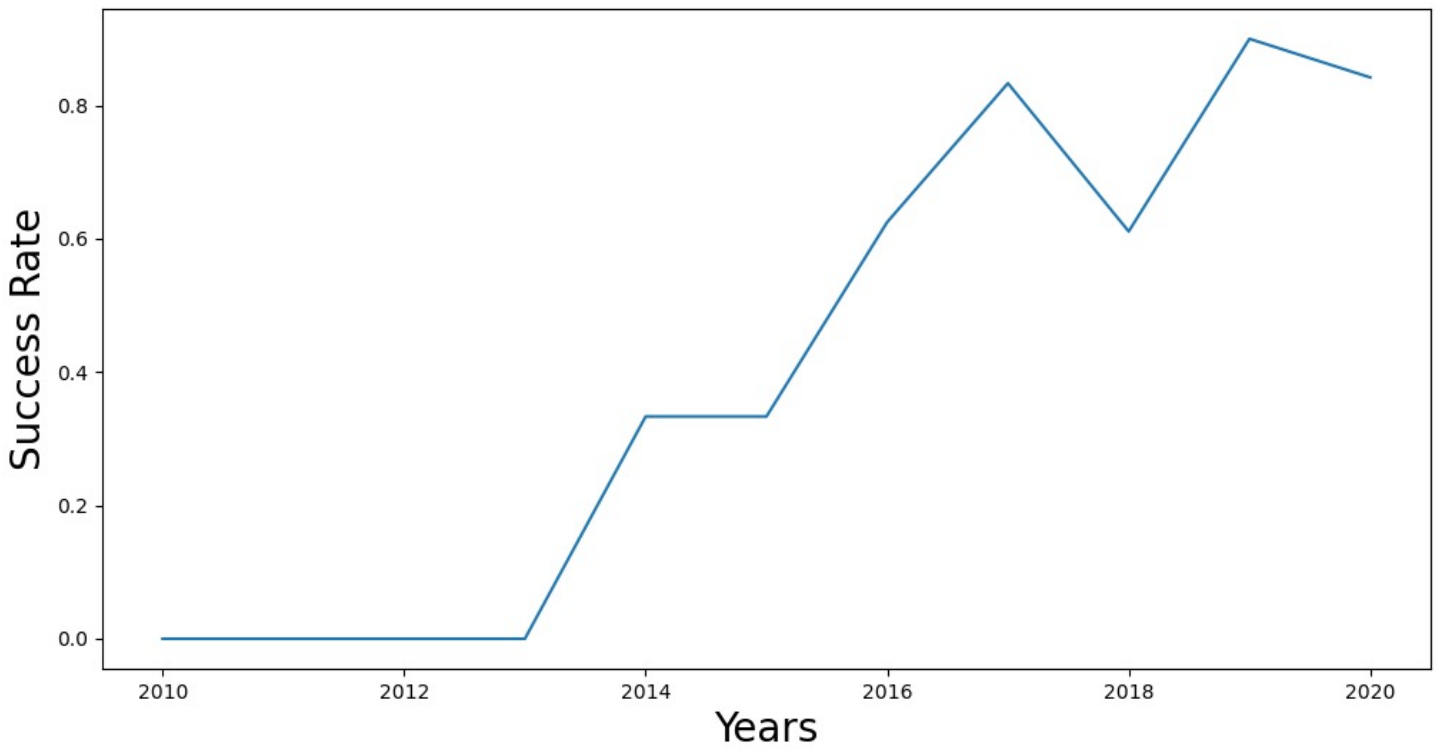
Payload vs. Orbit Type

- It is shown that as payloads get heavier, the rate of successful landings increases for the LEO, ISS, and PO orbit types.



Launch Success Yearly Trend

- The success rate of launches has steadily increased on average from the years of 2013-2020.



All Launch Site Names

- The DISTINCT function was used in our query to show all the unique launch site names in the space mission.

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

In [10]:
```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- This query was used to filter and display the first 5 records of launch sites that began with the string ‘CCA’.

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

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	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 calculated the total payload mass carried by boosters launched by NASA (CRS) as 45,596 kg.

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

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

- This query calculated the average payload mass carried by booster version F9 v1.1.

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

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- Query was used to list the date when the first successful landing outcome in ground pad was achieved.

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

- In the query we used the WHERE and AND functions to filter and list the names of the drone ship landings that were successful and had payload mass greater than 4000 but less than 6000.

In [15]:
```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- In our query, we used the WHERE and % functions to list the total number of successful and failure mission outcomes of 100 and 1, respectively.

List the total number of successful and failure mission outcomes

In [16]:
```
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)
```

Out[16]:

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

- We used a subquery with a MAX function in the WHERE function to list the names of the booster versions which have carried the maximum payload mass.

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

- In our query, we combined the WHERE, LIKE, AND, and BETWEEN functions to list the booster version, launch site, and landing outcome for all failed drone ship landing outcomes in the year of 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

- In our query, we selected both landing outcomes and their COUNT in the data by using the WHERE function to filter for landing outcomes BETWEEN 2010-06-04 to 2017-03-20.
- We applied the GROUP By and ORDER BY functions to order the grouped landing outcomes in descending order of their count.

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

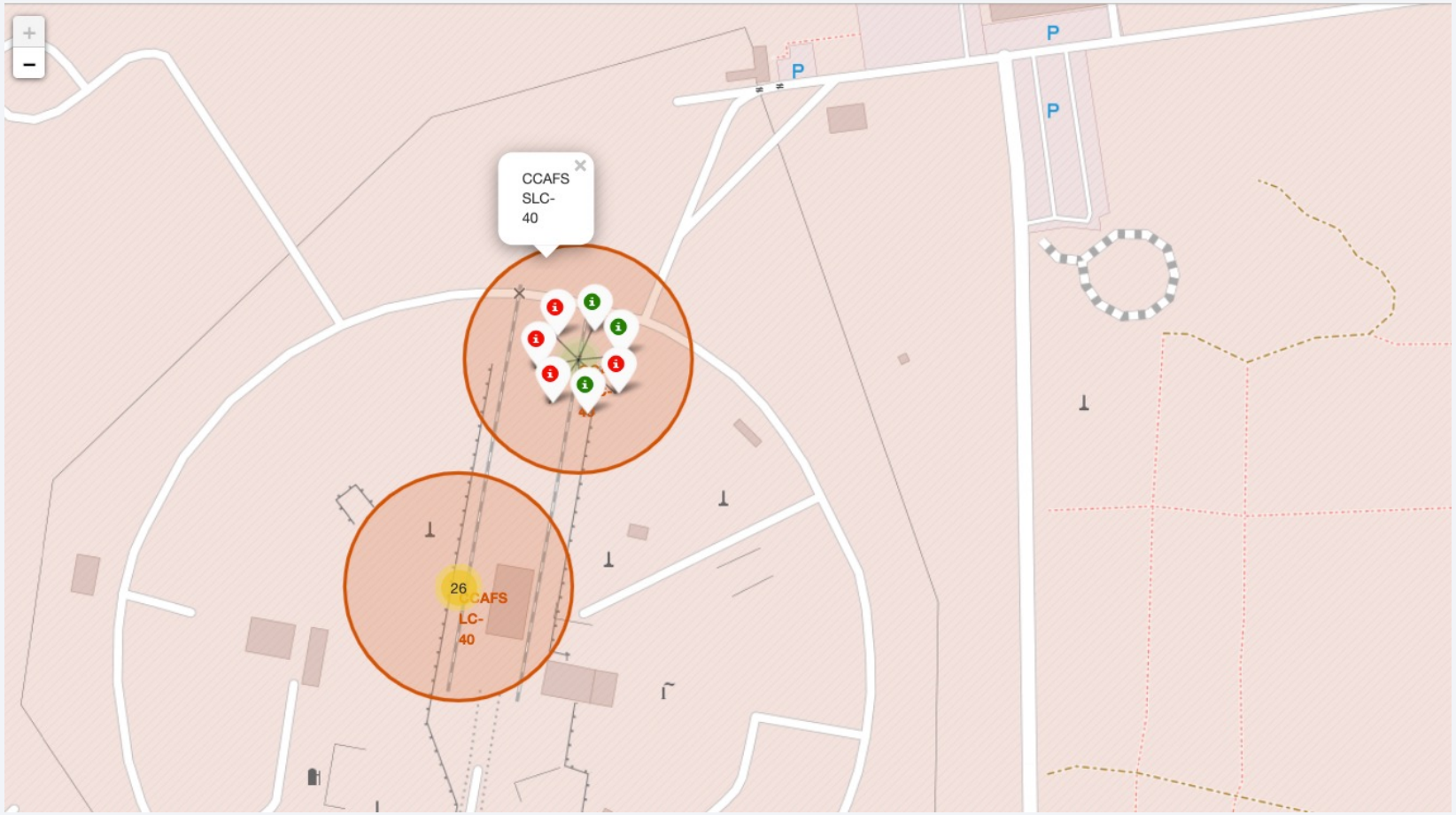

Section 3
Launch Sites
Proximities Analysis

All Global Launch Sites

- The map clearly shows that all SpaceX launch sites lie on either coast of the United States of America, particularly in Southern California and Eastern Florida.

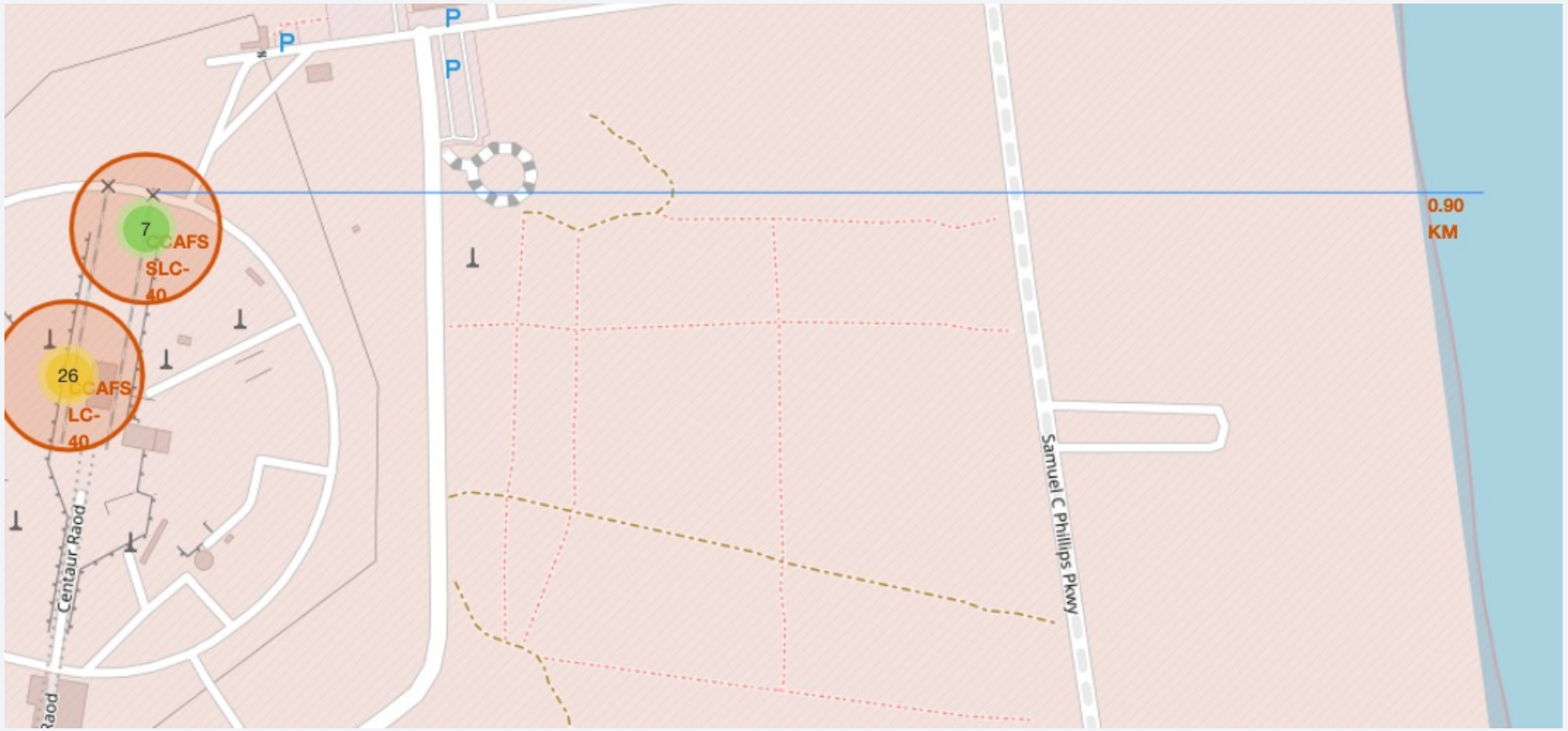
Launch Site Successes and Failures

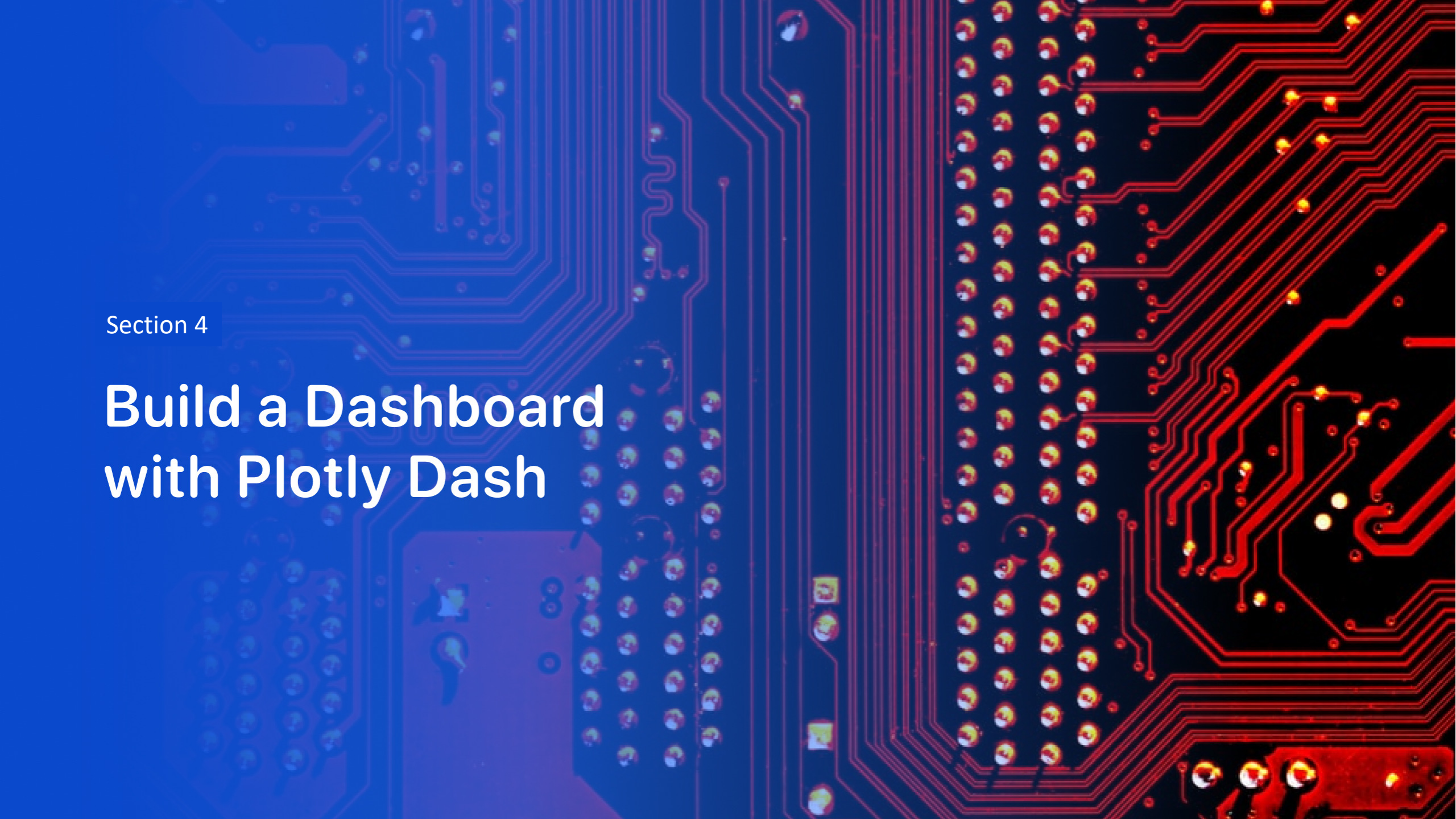
- For the Florida Launch Site shown to the right (CCAFS SLC-40), the green markers indicate successful launches, while the red markers represent failed attempts.



Launch Site Proximities

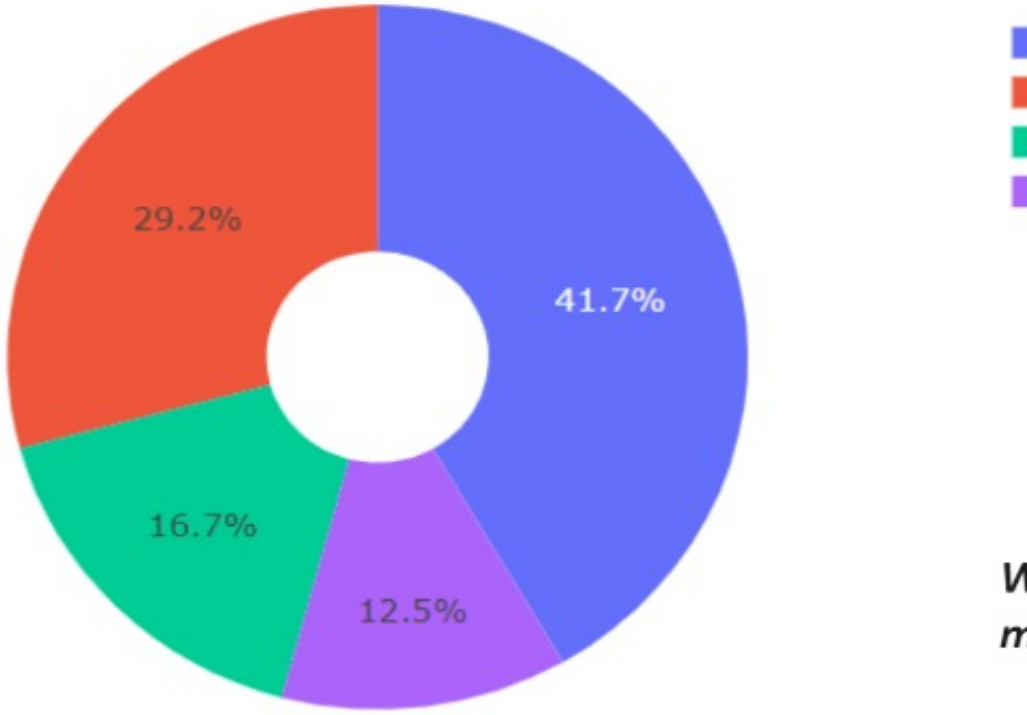
- The map shows the distance from the CCAFS SLC-40 launch site in Florida to the coast as 0.90 km. In general, we can see that all launch sites are located close to cities, but keep their distance from cities, railways, and highways.




Section 4
Build a Dashboard with Plotly Dash

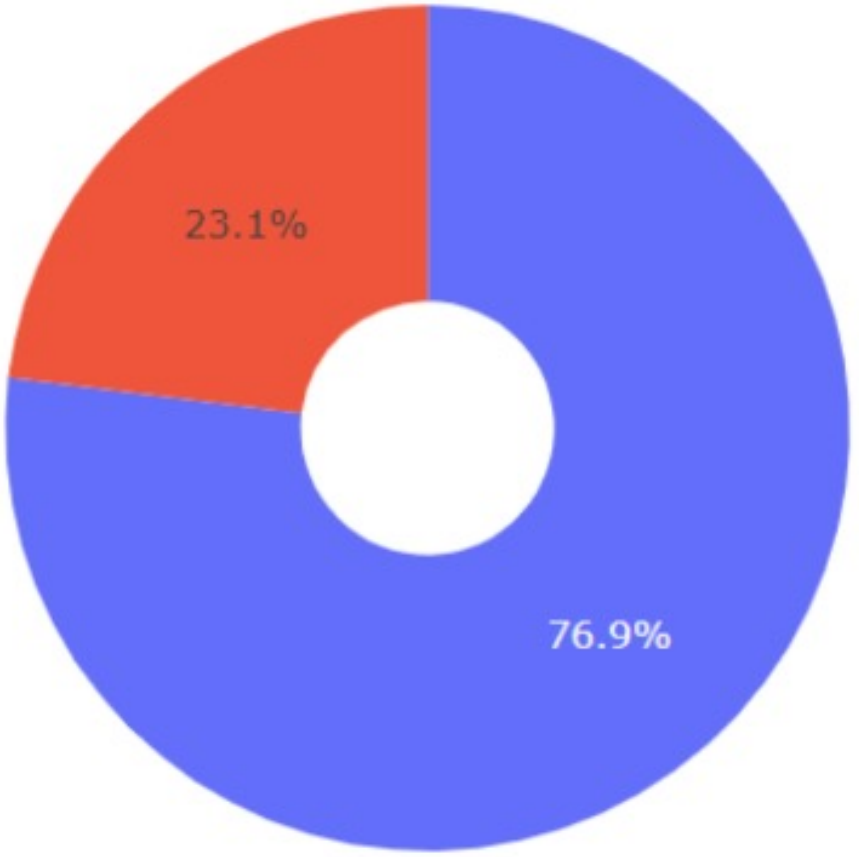
Success Percentage By Launch Site

Total Success Launches By all sites



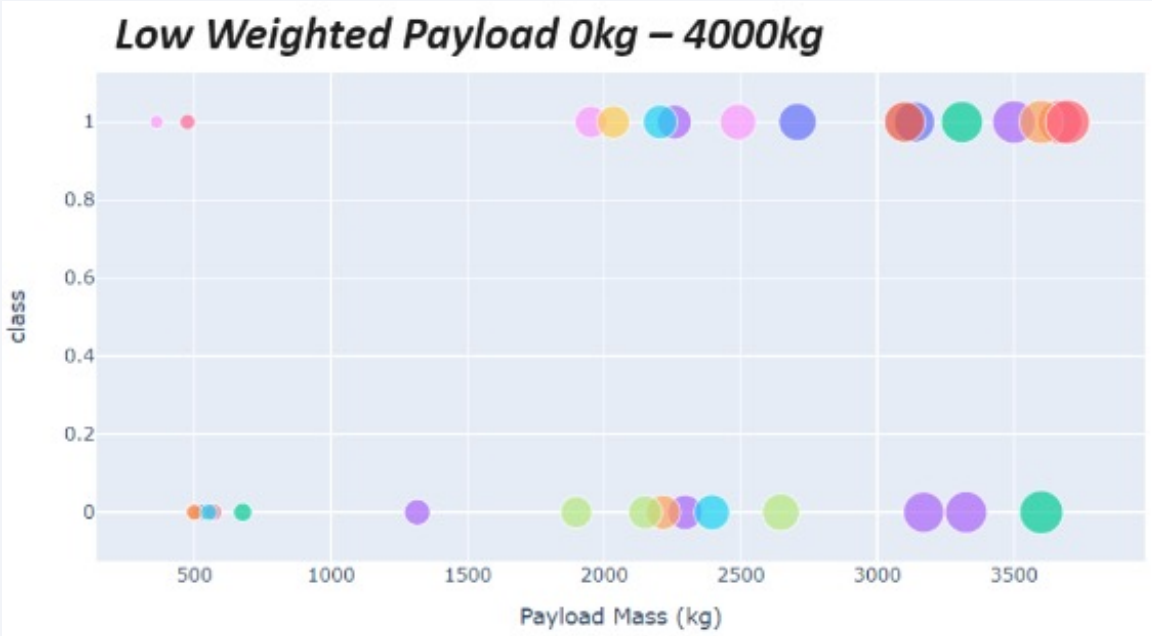
We can see that KSC LC-39A had the most successful launches from all the sites

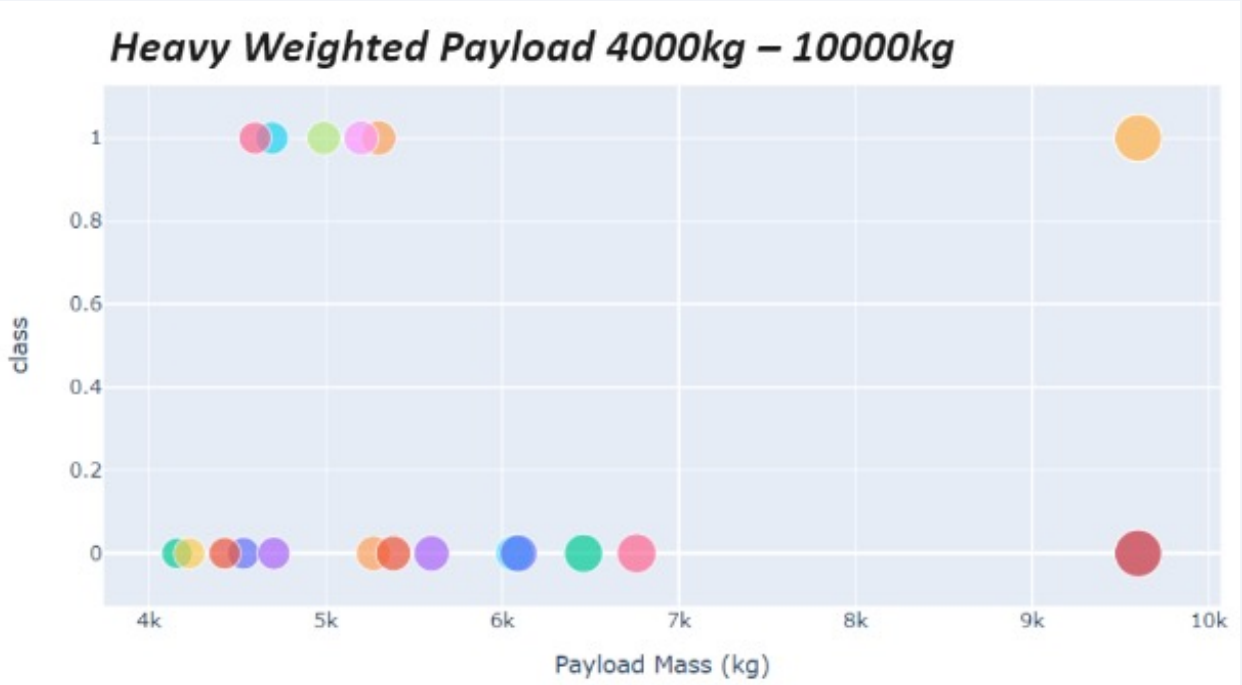
Highest Success Rate Launch Site



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Launch Outcome vs Payload Across All Launch Sites





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

Section 5

Predictive Analysis (Classification)

Classification Accuracy

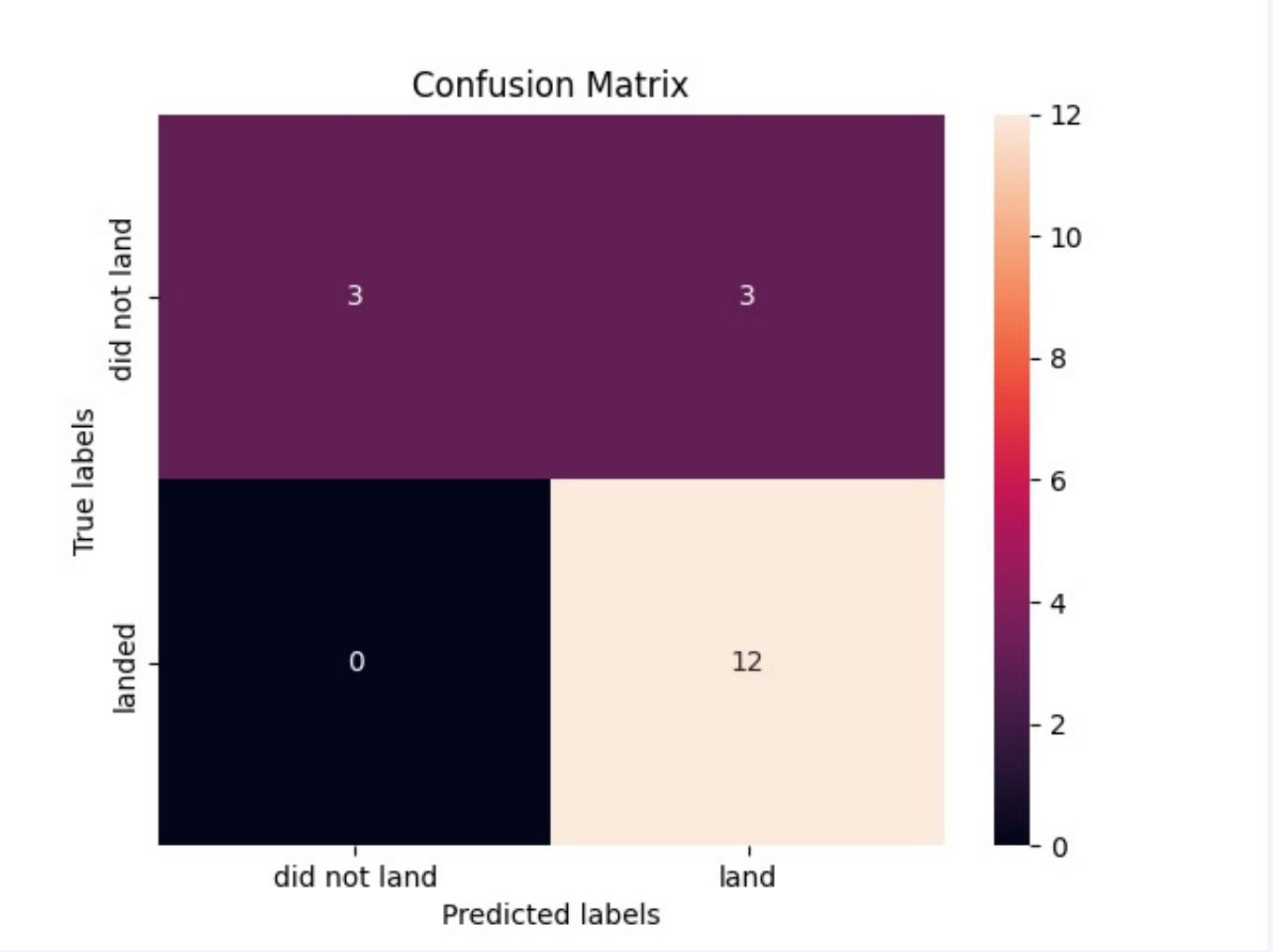
- The Decision Tree classifier is the model with the greatest classification accuracy of 0.873

```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix

- The confusion matrix is used to show how the decision tree classifier can distinguish between the different classes, however the 3 false positives reported by the matrix are a cause for concern of its accuracy.



Conclusions

- The larger the flight number at a launch site, the greater the success rate.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- Launch sites are located close to coastlines, but away from cities and highways.
- KSC LC-39A had the most successful launches of any site.
- Low weight payloads have higher success rates.
- The Decision tree classifier is the best machine learning algorithm for our application, with only a few Type 1 errors.

Thank you!