Winning Space Race with Data Science

Wade Bouley
09/23/2022

Outline

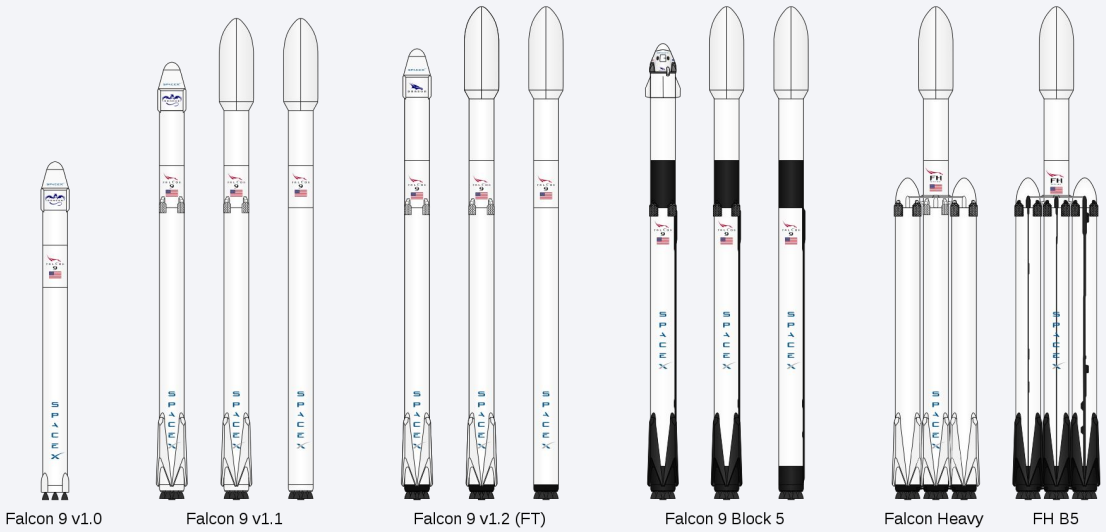
- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

- Data Collection Methods such as:
  - API data collection
  - Web Scraping
  - Data Wrangling
  - Data analysis with SQL
  - Interactive data visualization with Folium and Plotly
  - Making predictions using machine learning
- Summary of all results
  - Interactive analytics in screenshots
  - Machine learning results
  - Data analysis results

Introduction

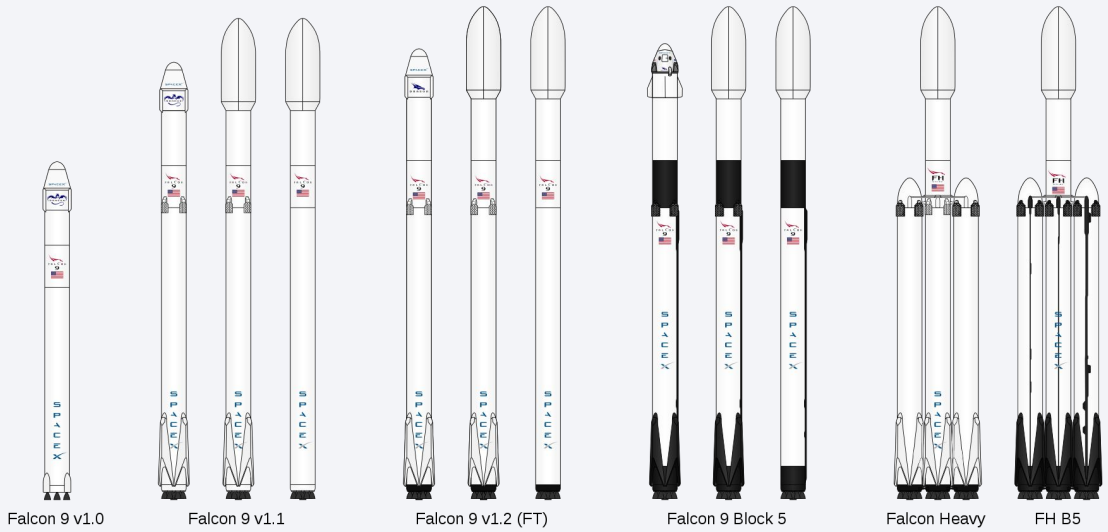
SpaceX is a private spaceflight company founded in March of 2002 by Elon Musk. SpaceX sends satellites and people to space, including NASA crews to the International Space Station (ISS). In this capstone, we will be predicting whether or not the first stage of Falcon 9 will land successfully.



Introduction cont.

Problems:

- Identifying the factors that affect the outcome of the landings.
- Learn about the relationship between each variable and figure out how they affect the landing outcome.
- What are the optimal conditions needed to maximize the probability of a successful landing.



Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Collected data with REST API's and web scraping from Wikipedia
- Perform data wrangling
  - Processed data using categorical features and one-hot encoding.
- 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 is the process of gathering and analyzing accurate data from various sources to find answers to research problems, trends, as well as probabilities to evaluate possible outcomes.

- Using get request for the REST API gave a response in JSON format which was then transformed into a pandas dataframe using the json_normalize() method. Next, the data was cleaned and checked for anomalies, missing values, and outliers.

- To web scrape, python package BeautifulSoup was used to gather the details of the launch records of the Falcon 9 as an HTML table. Then the table was parsed and converted into a pandas dataframe to analyze further.

Data Collection – SpaceX API

- This is the process that I used to collect the necessary data using API's and requests.

From:

- https://github.com/gituser111111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/jupyter-labs-spacex-data-collection-api.ipynb

```
In [7]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [8]: response = requests.get(spacex_url)
```

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

```
# 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 and 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 rockets 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 single 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 extracting 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

- This is how I scraped the web for the launch tables

From:

- https://github.com/gituser11111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/Data_Collection_with_Web_Scraping.ipynb

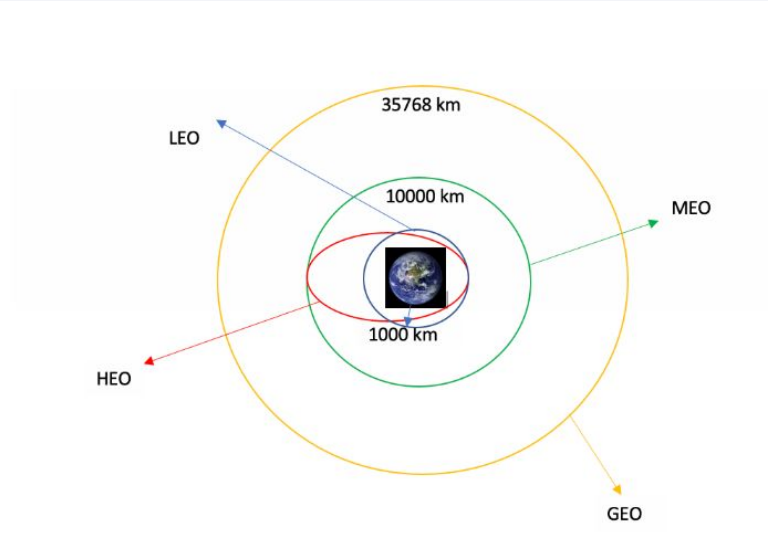
```
# use requests.get() method with the provided static_url

response = requests.get(static_url).text
```

```
# Use BeautifulSoup() to create a BeautifulSoup obj
soup = BeautifulSoup(response, 'html.parser')
```

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
```

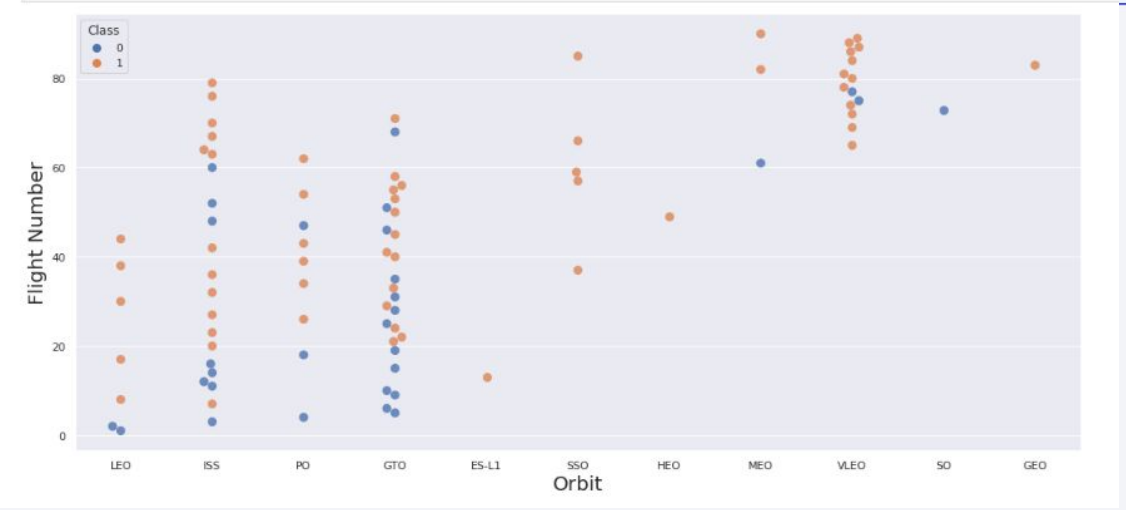
Data Wrangling

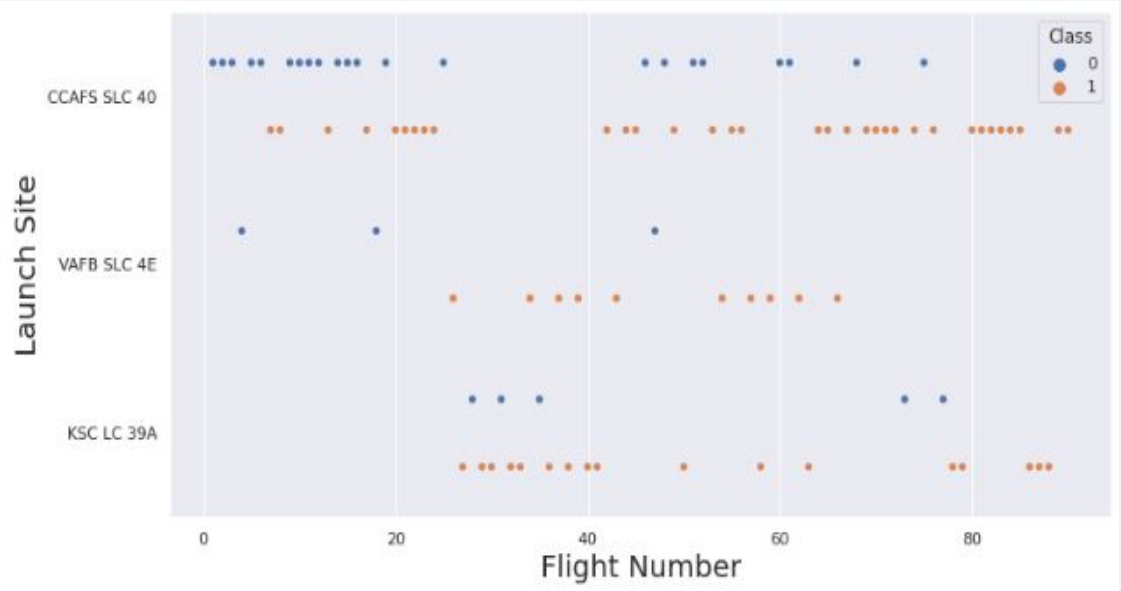


- Data wrangling refers to a variety of processes designed to transform raw data into more readily used formats. Data wrangling methods vary from project to project and is also known as data cleaning.
- First, I calculated the number of launches from each launch site and then calculated the the occurrence of each launch outcome per orbit type. (Success or Failure)
- Next, I created a landing outcome label from the column that contained the outcome of each mission. We believe that this will make the data easier to analyze and visualize.
- Finally, I exported this data to a .csv file for further analysis.

https://github.com/gituser111111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/Data_Wrangling_spaceX_08.15.2022.ipynb

EDA with Data Visualization

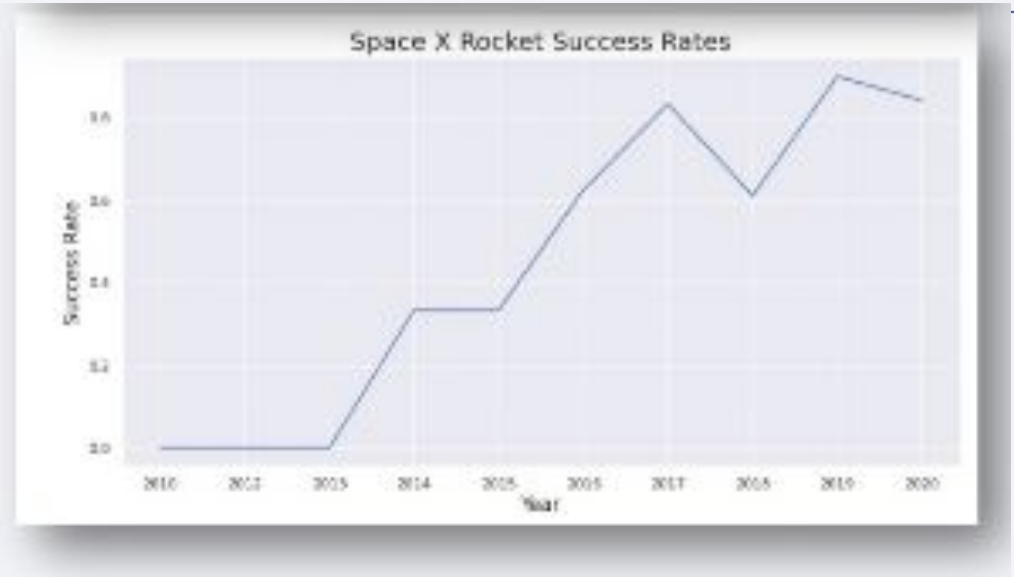




- I created a scatter plot to find if there are any relationships between variables such as:
  - Flight Number and Launch Site
  - Orbit Type and Payload
  - Orbit Type and Flight Number
  - Flight Number and Payload
  - Launch Site and Payload
- Scatter plots help me find relationships between dependant and independent variables like the variables listed above.

https://github.com/gituser11111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/Complete%20the%20EDA%20with%20Visualization%20lab.ipynb

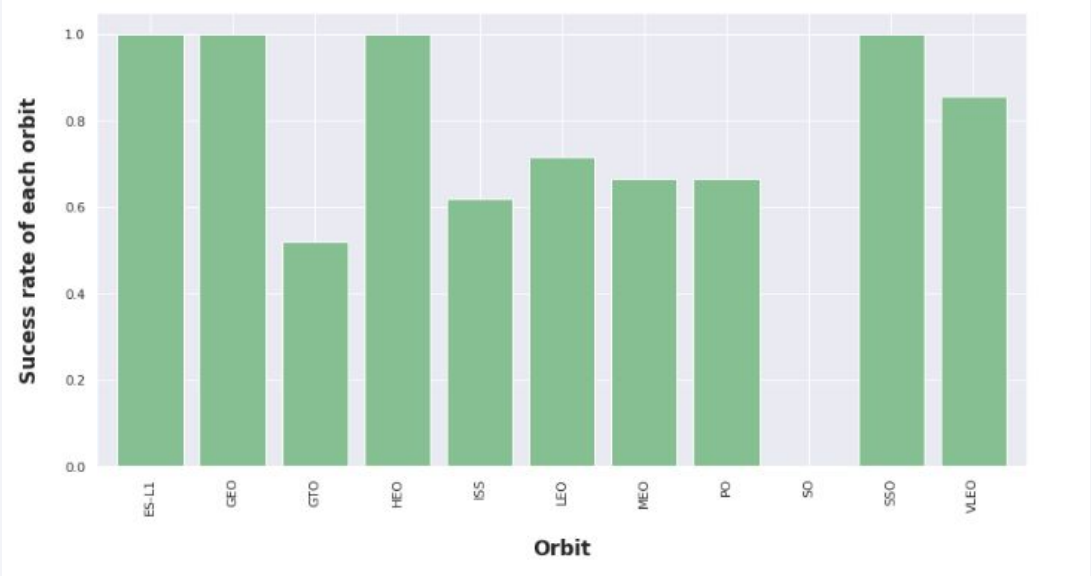
EDA with Data Visualization



We can use line and bar graphs to further analyze the data alongside of the scatter plots that we created above.

Bar graphs and line charts are some of the easiest ways we can interpret relationships between variables.

As we can see, the success rate for SpaceX launches has increased dramatically since the very start.



https://github.com/gituser11111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/Complete%20the%20EDA%20with%20Visualization%20lab.ipynb

EDA with SQL

- I used SQL to perform multiple queries to help us get a better understanding of the SpaceX dataset.
  - Identifying the names of the launch sites
  - Identifying the total payload mass carried by booster launched by NASA (CRS).
  - Showing 5 records where the launch site begins with ‘CCA’.
  - Displaying the average payload mass carried by booster version F9 v1.1.
  - Listing the total number of each mission outcome.
  - Listing the names of the boosters that have carried the maximum payload mass.
  - Listing the date when the first successful landing outcome in ground pad was achieved.
  - Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - Rank the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20 in descending order.

https://github.com/gituser11111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/jupyter-labs-eda-sql-coursera.ipynb

Build an Interactive Map with Folium

- I visualized the SpaceX launch data into an interactive map using Folium. The latitude and longitude coordinates of each launch site were used to create labels which represent successful launches and failed launches.
- This helped me easily understand and visualize which outcome each launch had. We used green markers to represent successful launches and red markers to represent failed launches using MarkerCluster() front he launch_outcomes dataframe.
- Next, I used Haversine's formula to determine the distance of the launch sites to various landmarks such as railways, highways, coastlines, and cities.

https://github.com/gituser11111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

I created a dashboard which is interactive so users can see different scenarios within the data.

A scatter chart was created to show the relationship between Payload Mass in (Kg) and the result of the launch for the different versions of the boosters.

A pie chart was created as well to highlight the total number of launches per site.

https://github.com/gituser11111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/Build%20an%20interactive%20dashboard%20with%20plotly_spacex_09.02.2022.ipynb

Predictive Analysis (Classification)

Build the ML Model

1. Load the dataset into Pandas and NumPy
2. Transform the data and then split the data into training and test datasets
3. Which type of Machine Learning should I use?
4. Next, set the parameters and algorithms to GridSearchCV and fit it to the dataset

Evaluate the ML Model

1. Check the accuracy for each of the ML models.
2. Tune the hyperparameters for each type of algorithms.
3. Next, plot the confusion matrix.

Improve the ML Model

1. Algorithm Tuning and Feature Engineering are used.

Find the Best ML Model

1. The ML model with the best accuracy score is going to be the best performing model.

From: https://github.com/giuser11111111/IBM-Data-Science-Professional-Certificate-Coursera/blob/main/Capstone_Projects/SpaceX_dash_payload_graphs.ipynb

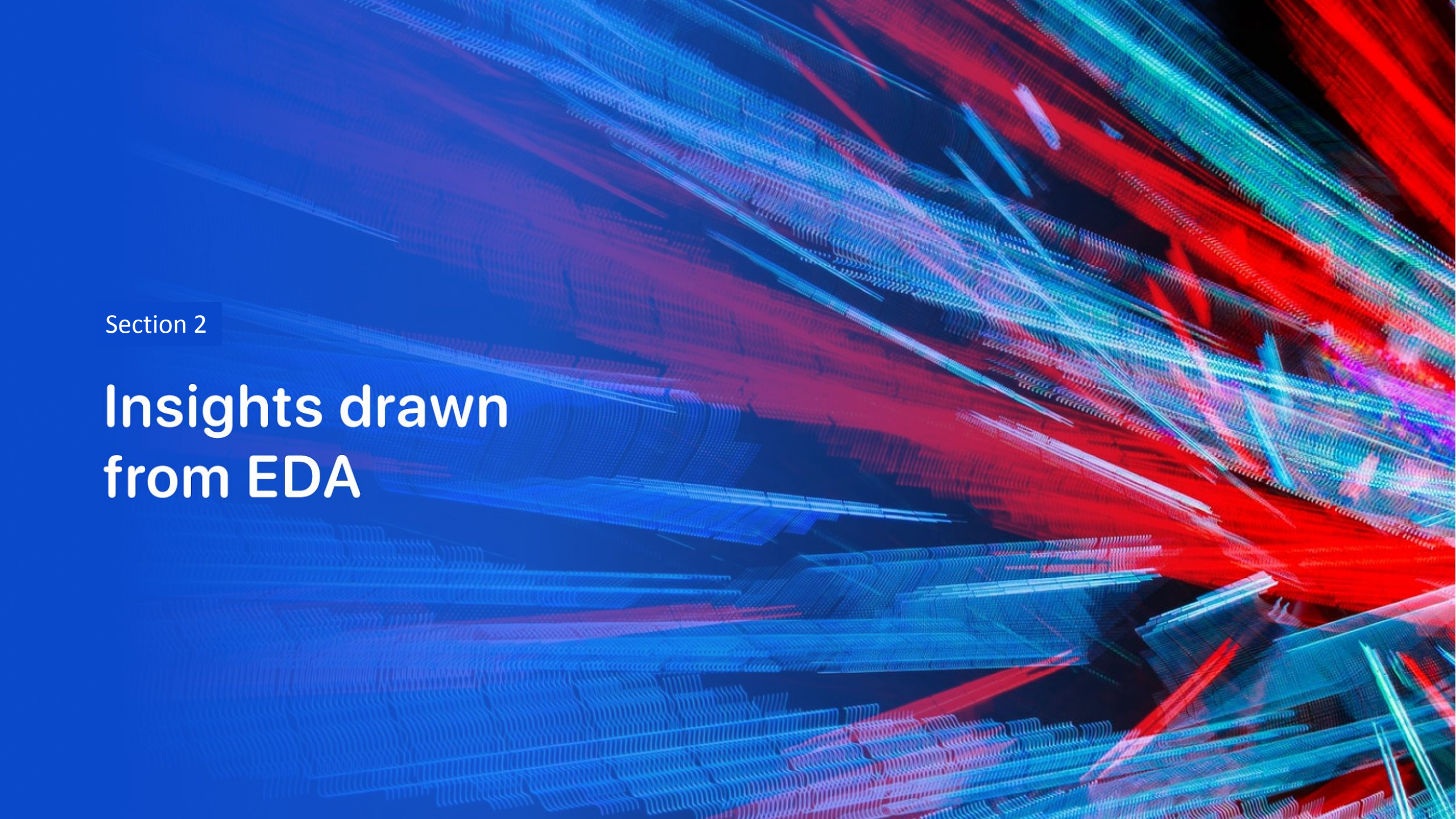
Results

The results will be placed into 3 categories:

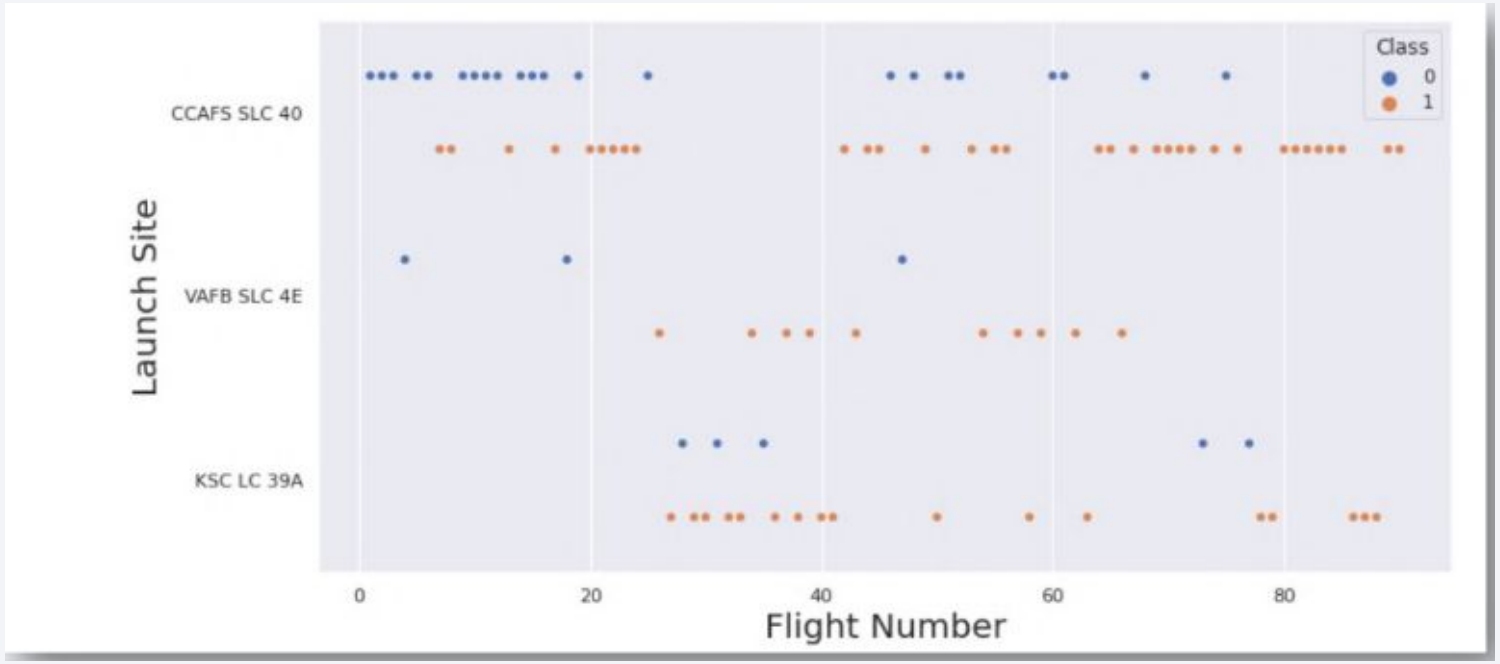
1. Exploratory data analysis results
2. Interactive analytics results
3. Predictive analysis results.

Section 2

Insights drawn from EDA



Flight Number vs. Launch Site



This scatter plot shows that the more flight attempts at each launch site, the better the odds of a successful launch

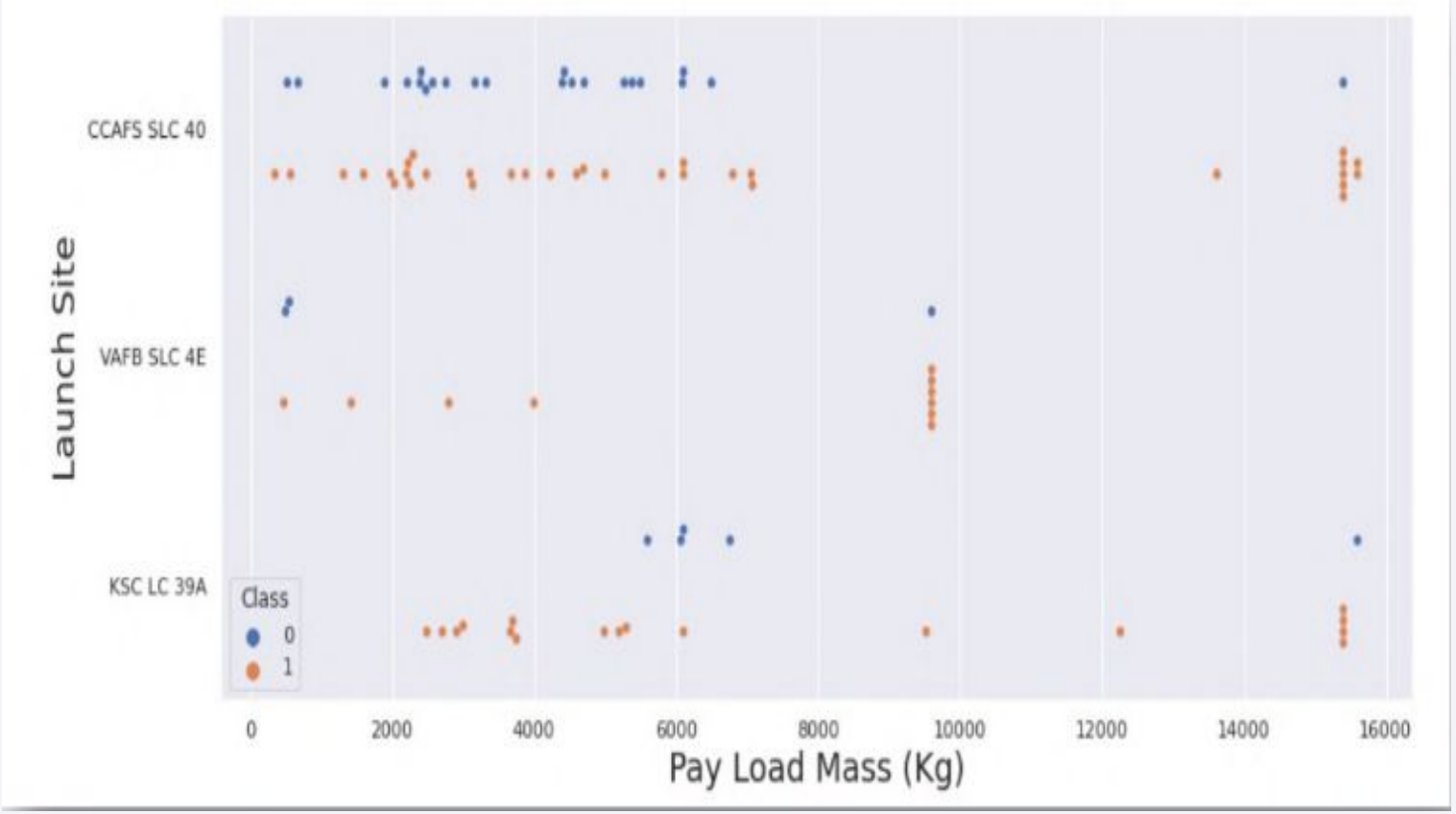
However, the launch site CCAFS SLC 40 isn't the best example of this patter.

It is worth noting that class 1 represents a successful launch and class 0 represents a failed launch.

Payload vs. Launch Site

Keeping in mind that class 1 launches represent successful launches, we can see that when the payload mass becomes greater than 7000kg, the higher the probability of a successful launch.

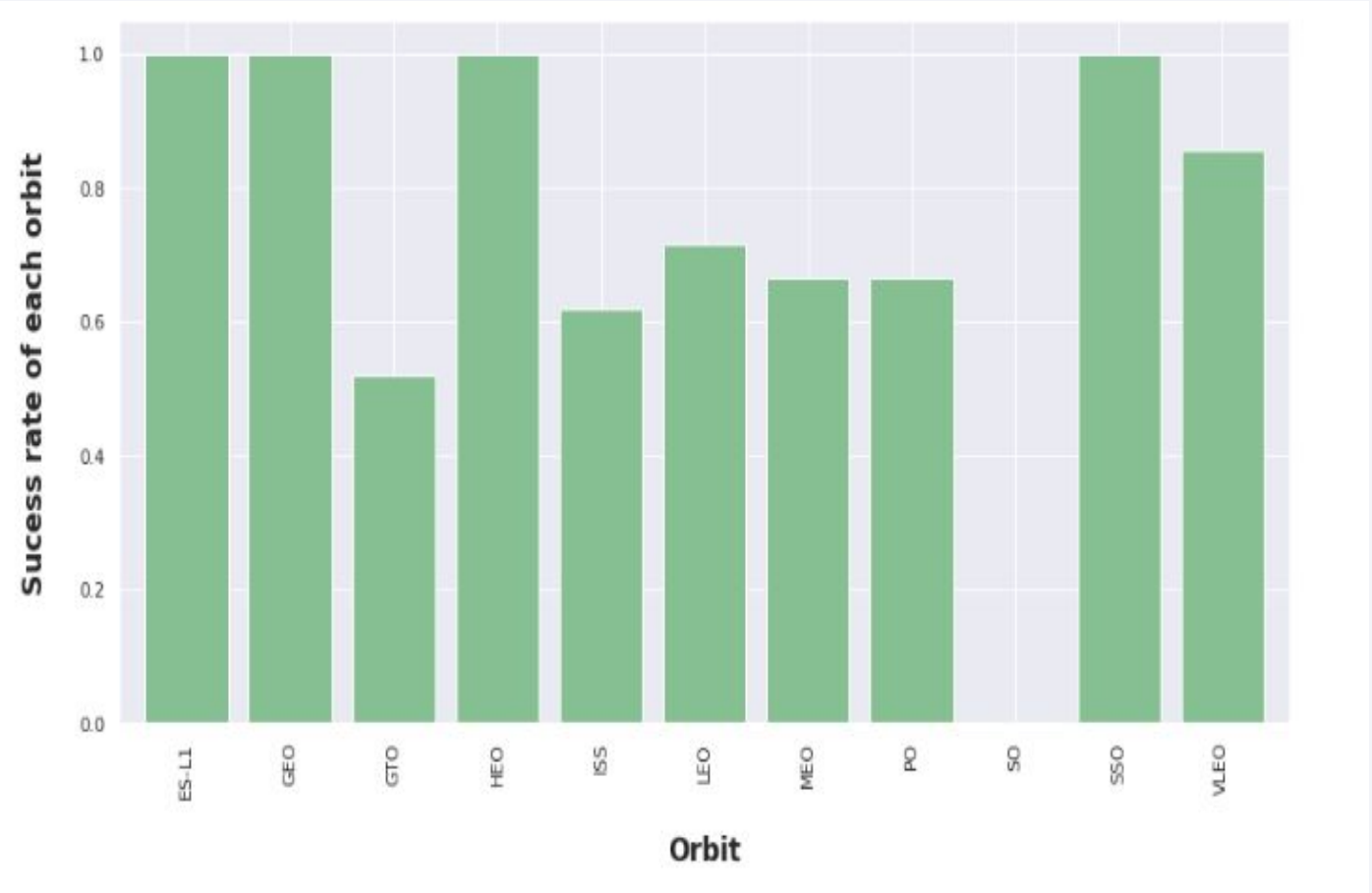
We can make an argument that launch site CCAFS SLC 40 has an issue with unsuccessful launches with a payload mass that is less than 6000kg, although we noticed in our last slide that launch site CCAFS SLC 40 has already been problematic.



Success Rate vs. Orbit Type

- This bar chart represents the success rate of each launch relative to the type of orbit.
- 4 launches had a 100% success rate. (SSO, HEO, GEO, and ES-L1)
- Orbit type SO had a 0% success rate.

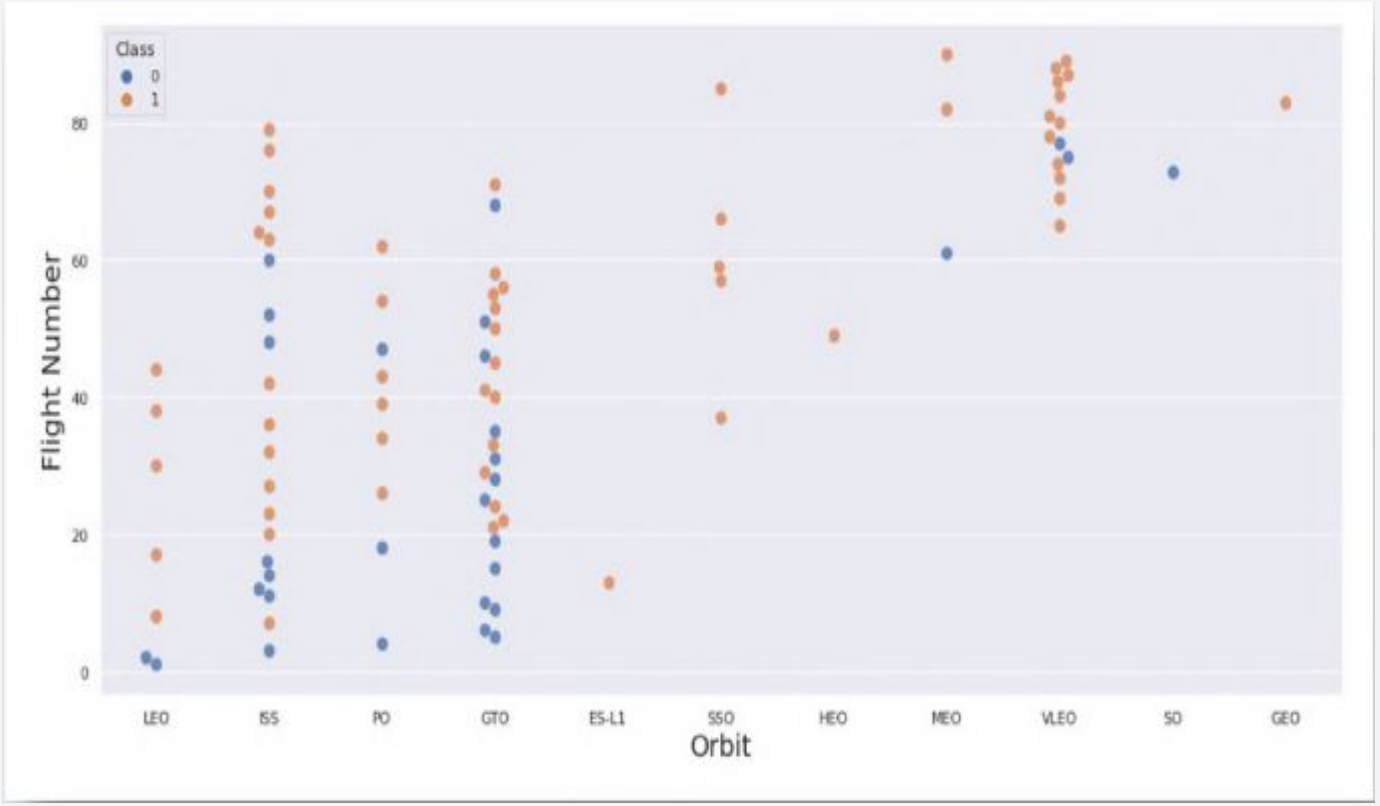
However, it is worth noting that some of these launches had only 1 occurrence such as the launches with orbits SO, GEO, HEO, and ES-L1. That tells us that more data is needed to draw any conclusions with this particular dataset.



Flight Number vs. Orbit Type

This scatter plot shows us a pattern in which the larger the flight number of each orbit type, the greater the success rate. Although, this is not the case with the GTO orbit.

Orbits with only one or two flight numbers should possibly be excluded from the overall conclusion since we don't have that much data.

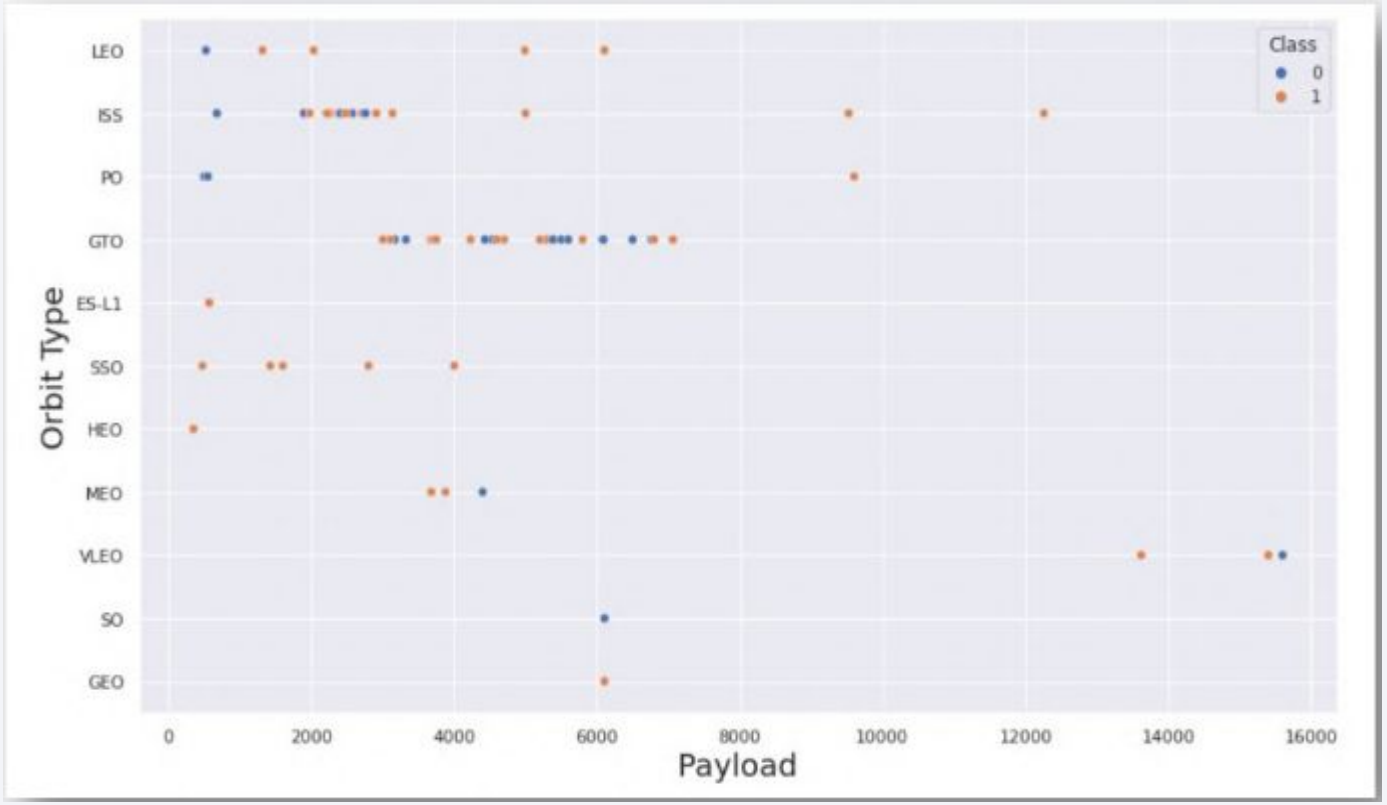


Payload vs. Orbit Type

Right away, we can see that orbit types SO, GEO, HEO, and ES-L1 need more data for us to make a conclusion.

Heavier payload seems to increase the success rate for orbit types LEO, ISS, and PO.

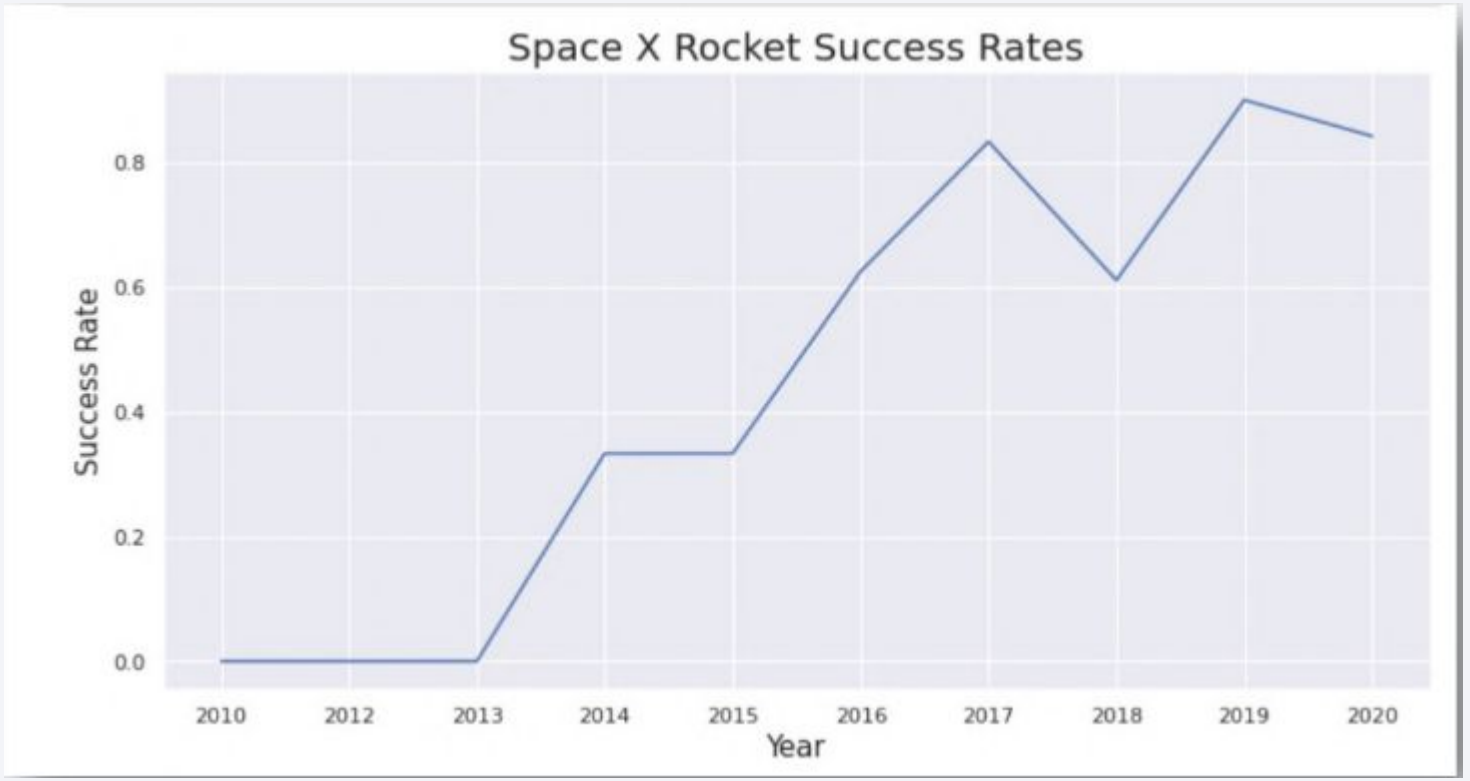
We don’t quite see a solid pattern for orbit type GTO.



Launch Success Yearly Trend

This line chart shows the yearly success rates trend for SpaceX rocket launches.

We can see that the overall trend rate has a positive slope since 2013.



All Launch Site Names

- This SQL query gives us a list of unique launch sites. The distinct keyword helps us achieve the goal of finding unique values

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

 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu01qde00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[5]:

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

Launch Site Names Begin with 'CCA'

- This SQL query helped us find launch sites that begin with ‘CCA’. We accomplished this by specifying what we wanted the launch site to begin with with the WHERE clause in the query.
- We only wanted to return 5 results and we accomplished this by using the LIMIT clause in the query.

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 SQL query helped us find the total payload mass in kg. I achieved this by using the SUM function within the query. I renamed that result as “Total Payload Mass by NASA (CRS)” The total payload mass is 45,596 kg

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)
```

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

Total Payload Mass by NASA (CRS)
45596

Average Payload Mass by F9 v1.1

- This SQL query helped me calculate the average payload mass for booster version F9 v1.1.
- I used the AVG function to determine and the average payload and I used the WHERE clause to drill down on the F9 v1.1 booster version and only show the average payload for that particular booster version.

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

This SQL query helped me find the first successful ground landing date.

To accomplish this, I used the MIN function on the date column to find the earliest date and then used the WHERE clause to drill down on a condition within the Landing_Outcome column.

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu01qde00.databases.appdomain.cloud:32731/bludb
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

This SQL query helped me find successful drone ship landings with a payload between 4000 and 6000 kg.

The SELECT statement gave me the booster version name and the WHERE clause helped me find landing outcomes that were only successful drone ship landings.

The AND conditions helped return values that ONLY met all of the conditions in the WHERE clause.

```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
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

Using the wildcard sign % in the WHERE clause, I was able to determine whether or not the mission was a success or failure.

List the total number of successful and failure mission outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Successful Mission
100

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Failure Mission
1

Boosters Carried Maximum Payload

By using a subquery with a MAX() function in the WHERE clause, I was able to determine the booster versions that carried the maximum payload

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Booster Versions which carried the 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

By using a combination of functions within the WHERE clause such as LIKE and AND, I was able to filter for missions that failed, the launch site names, and the booster version in the year 2015.

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu01qde00.
databases.appdomain.cloud:32731/bludb
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

I was able to select the total count of landing outcomes between 2010-06=04 and 2017-03-20 by using an aggregate COUNT function. I was able to filter the date column by using a BETWEEN clause within the WHERE clause. Then, I grouped and ordered the total count of each landing outcome in descending order by using the GROUP BY and ORDER BY clause.

```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" 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://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

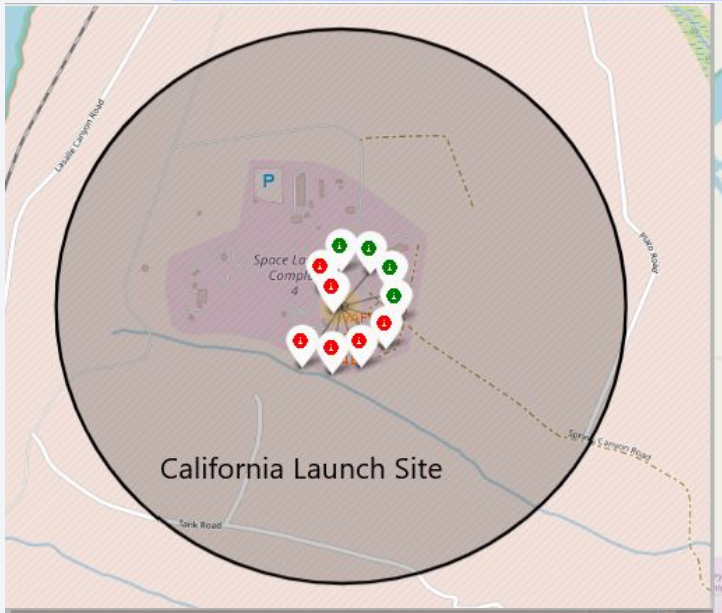
Landing Outcome	Total Count
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

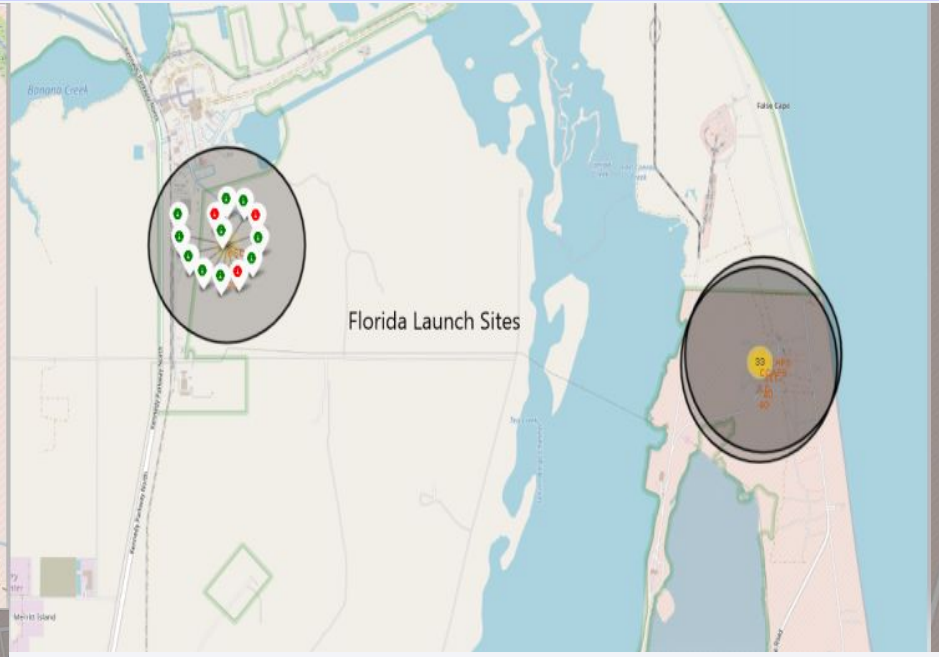

Section 3
Launch Sites
Proximities Analysis

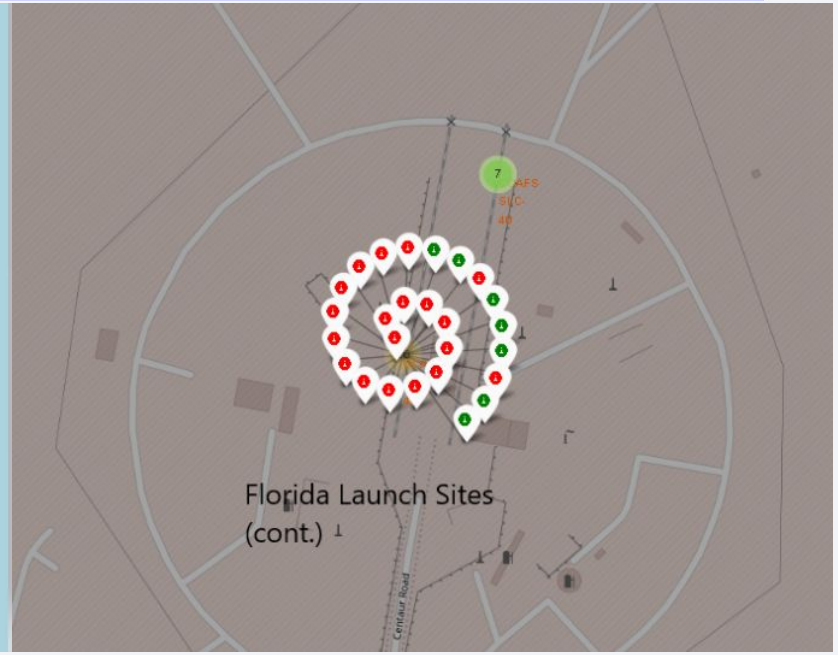
SpaceX Global Launch sites

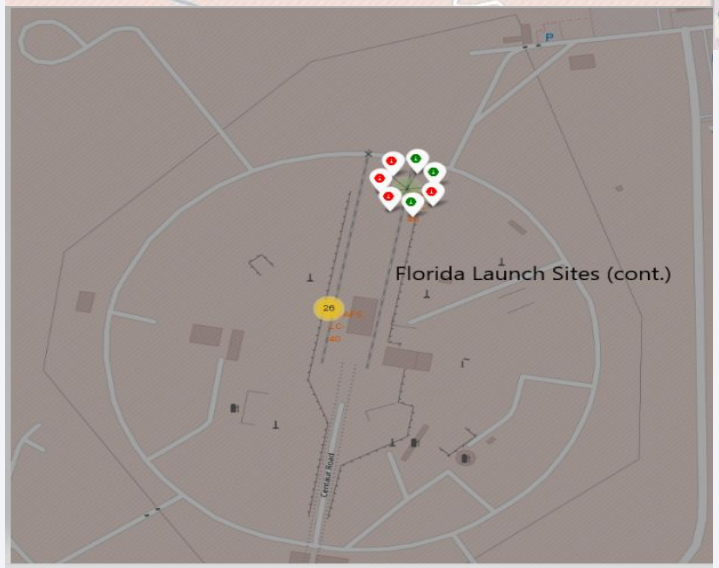
In the screenshot below, we can see that the launch sites that SpaceX uses are all in the United States, specifically in Florida and California.

Markers with launch sites and color labels.





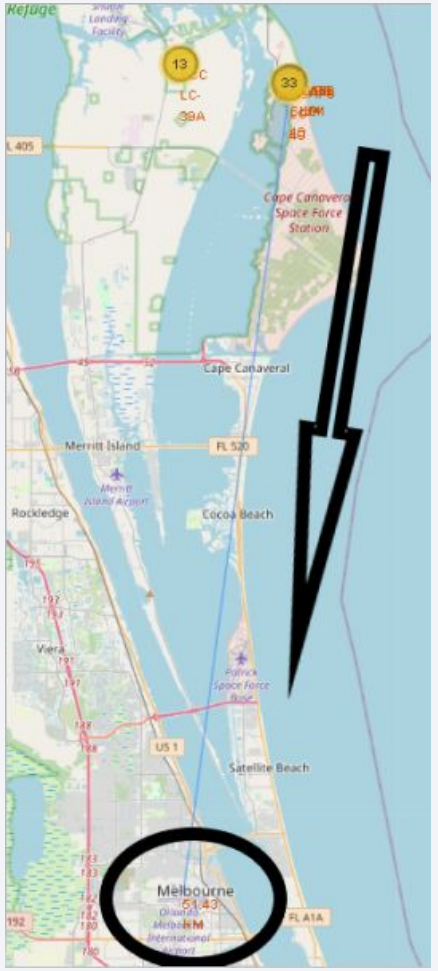


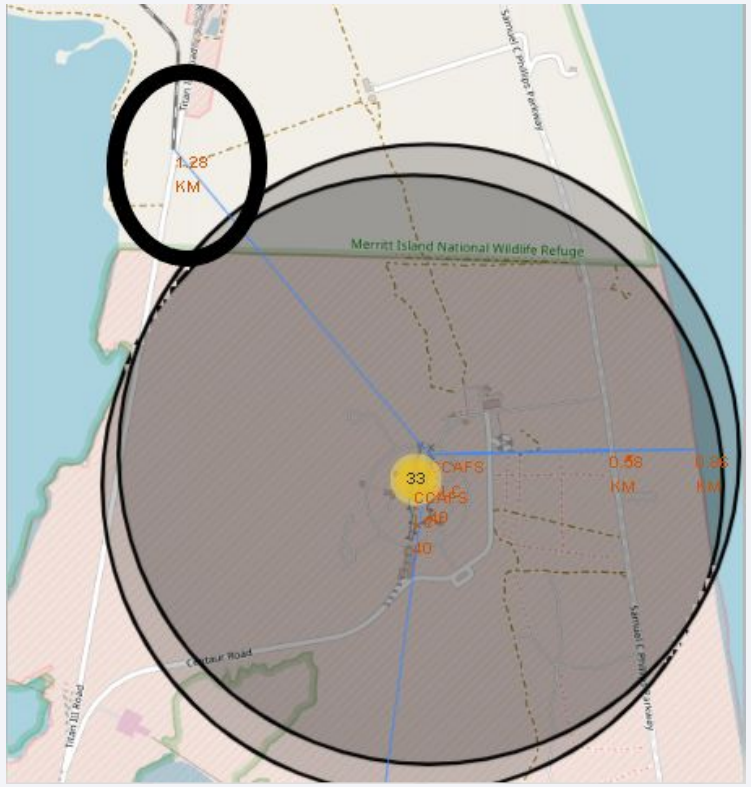


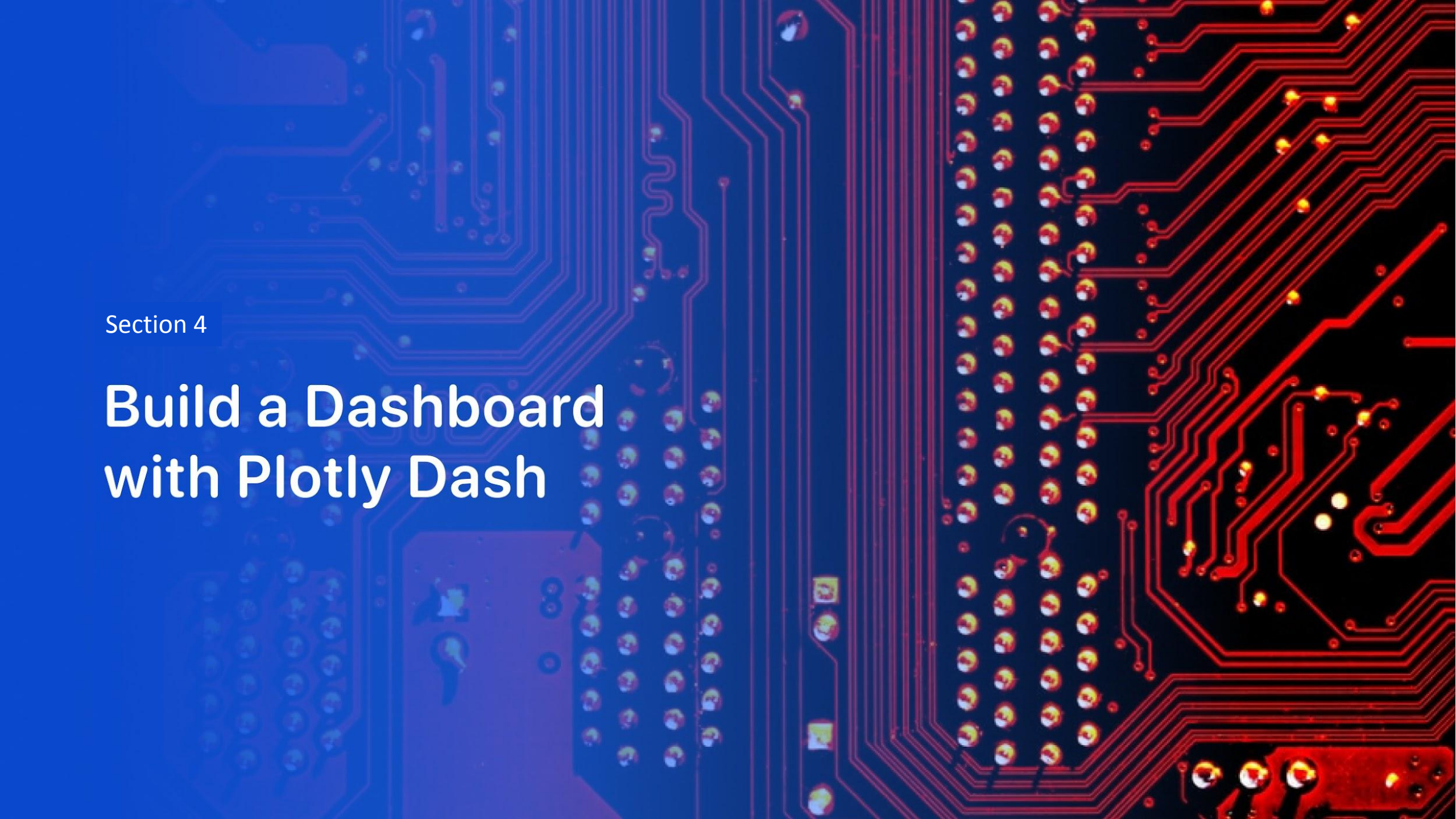
Green markers represent successful launches.
Red markers represent failed launches.

SpaceX location Proximities

As we can see in the screenshot below, the nearest coastline is approximately 0.90 KM from the launch sites. The closest city to the Florida launch site is approximately 51.5 KM away. The closest railroad is about 1.28 KM from the launch site.





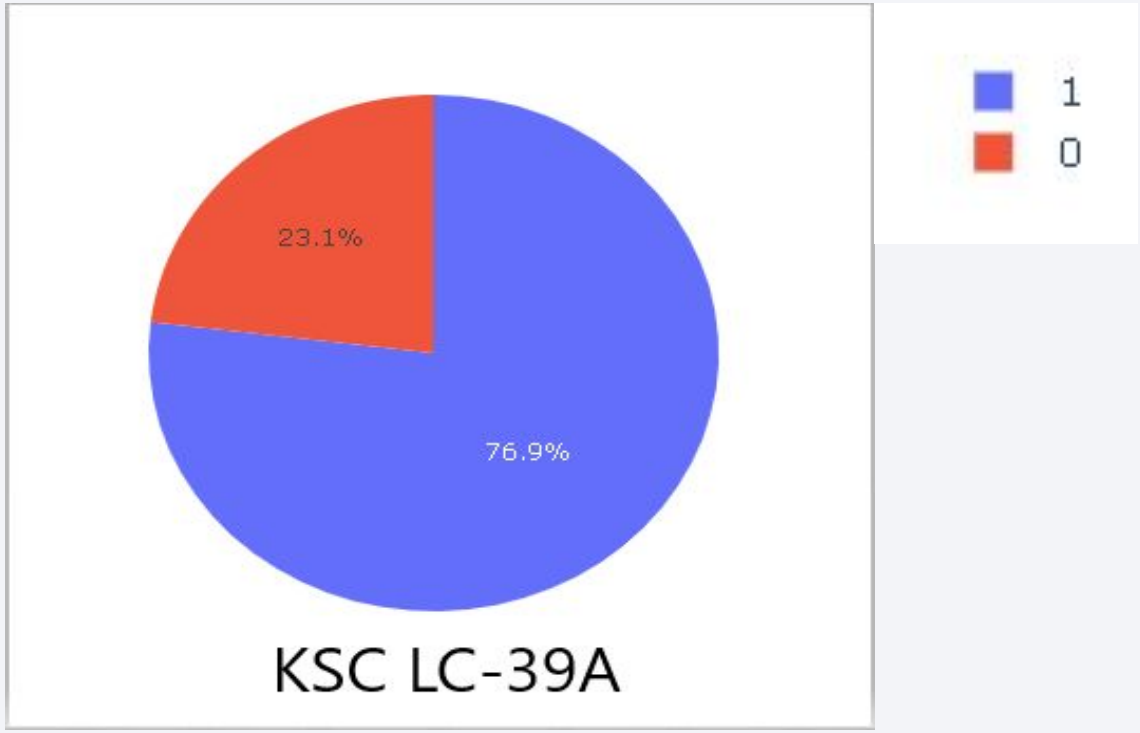

Section 4
Build a Dashboard with Plotly Dash

Total Successful Launches / Site

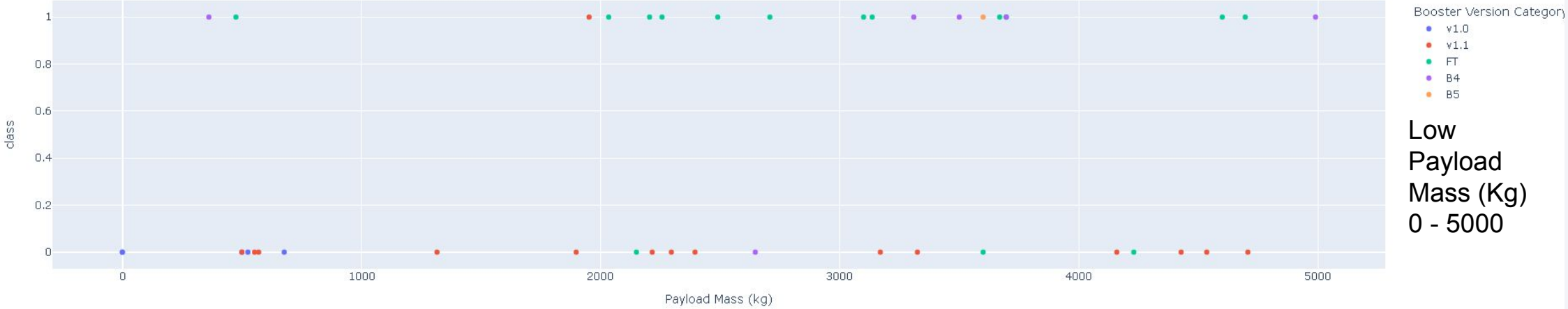
The pie charts below help us visualize the success rate for the number of launches per site. As we can see in the pie-chart below, launch site KSC LC-39A had the most successful launches out of all of the sites.

Most Successful launch site

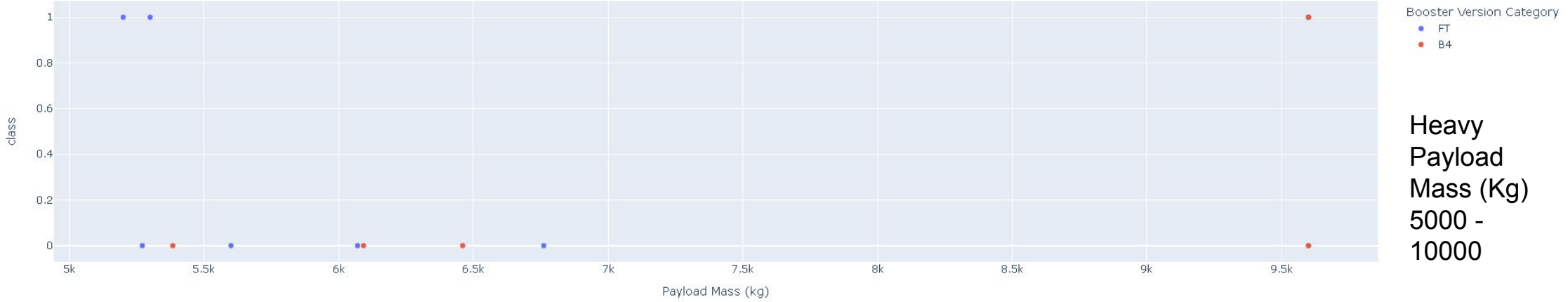
Keep in mind that class 0 launches represent a failed launch and class 1 launches represent a successful launch. Launch site KSC LC-39A had the highest success rate of 76.9%



Payload Mass Launch Outcomes All Sites



Low Payload Mass (Kg) 0 - 5000



Heavy Payload Mass (Kg) 5000 - 10000

Section 5

Predictive Analysis (Classification)

Classification Accuracy

The decision tree classifier is the model with the highest classification accuracy with an accuracy level of 0.8875

In [41]:
```
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier()
```

In [46]:
```
grid_search = GridSearchCV(tree, parameters, cv=10)
tree_cv = grid_search.fit(X_train, y_train)
```

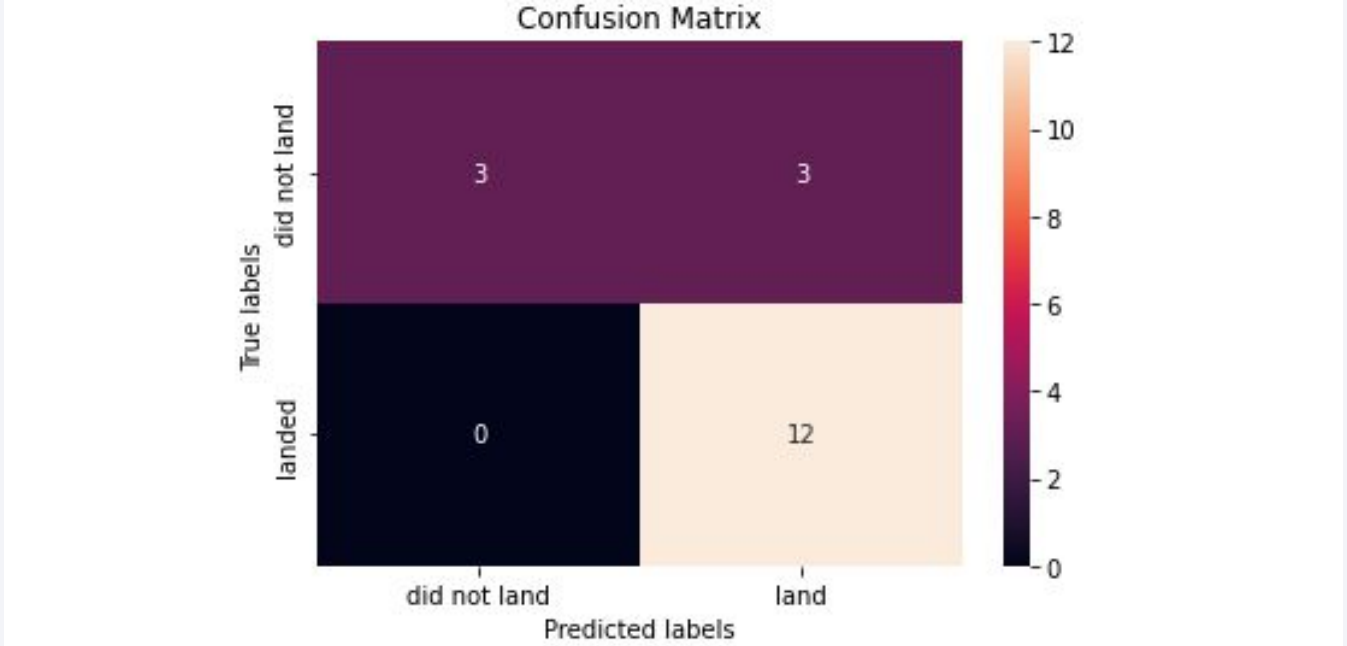
In [48]:
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 4, 'max_features': 'auto', 'min_samples_lea
f': 1, 'min_samples_split': 5, 'splitter': 'best'}
accuracy : 0.8875000000000002
```

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 or, unsuccessful landing marked as successful landing by the classifier.

```
In [50]:
1 yhat = svm_cv.predict(X_test)
2 plot_confusion_matrix(y_test,yhat)
```



Conclusions

- The more flight attempts at a launch site, the greater the success rate.
- Overall launch success rates have increased from 2013 through 2020.
- The Decision Tree Classifier is the best machine learning algorithm for this task.
- Launch site KSC LC-39A has been the most successful site for launches.
- Orbits GEO, HEO, VLEO, SSO, and ES-L1 had the highest success rates, although more data is needed.

Thank you!