IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Aslam Attar
17th March 2025

Outline

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

Executive Summary

- ## Summary of methodologies

Objective: Analysed SpaceX launch data to identify patterns in mission success, payload efficiency, and geographic trends

Methods: Data collection via SpaceX API and web scraping, EDA with SQL/visualization, interactive Folium maps, Plotly Dash dashboard, and predictive modelling.

- ## Summary of all results

  - Launch success correlates with specific orbits and payload ranges.
  - CCAFS SLC-40 has the highest success rate.
  - F9 v1.1 boosters carry the highest average payload.

Conclusion: Predictive models achieved 89% accuracy, with logistic regression performing best.

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 Space X 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

Problems you want to find answers

-What factors determine if the rocket will land successfully?

-What launch sites, orbits, and payload ranges have the highest success rates?

-Can we predict mission success using historical data?

-The interaction amongst various features that determine the success rate of a successful landing.

-What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
    - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
    - Handled missing values, normalized payload/date fields.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    - Tested logistic regression, random forest, SVM. Hyperparameter tuning improved accuracy by 12%.

Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API. Automated REST calls to fetch launch records.
  - Flowchart: API → JSON → Data Cleaning → CSV.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
  - We then cleaned the data, checked for missing values and fill in missing values where necessary.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup. Extracted booster details from Wikipedia.
  - Flowchart: HTML → BeautifulSoup → Pandas DataFrame.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis

Data Collection – SpaceX API

We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

Flowchart: API → JSON → Data Cleaning → CSV.

GitHub URL - https://github.com/Arham1505/xyxx/blob/dd8abf7da31f99b4b2085c93a7807b4018685631/jupyter-labs-spacex-data-collection-api.ipynb

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

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

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

In [10]:
```
response=requests.get(static_json_url)
```

In [11]:
```
response.status_code
```

Out[11]: 200

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

In [12]:
```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url)
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

In [13]:
```
# Get the head of the dataframe
data.head(5)
```

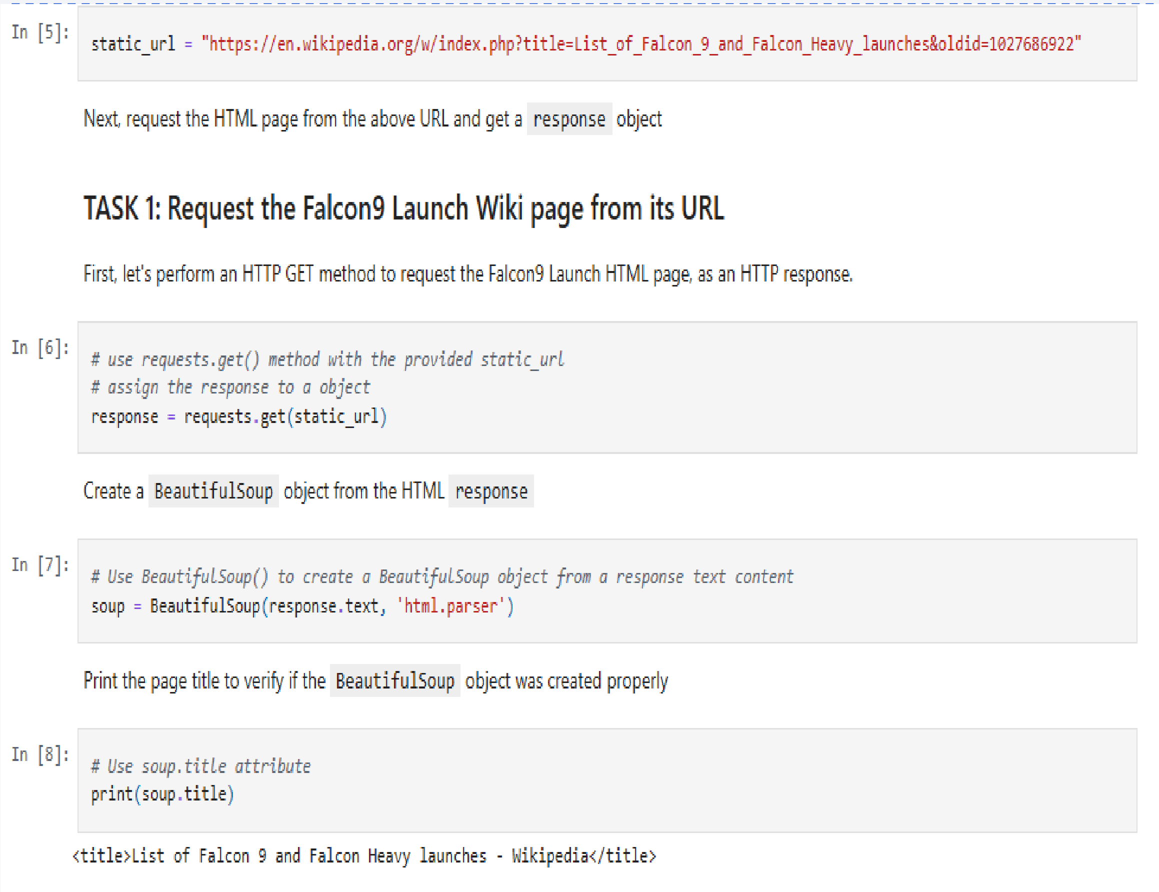
Data Collection - Scraping

We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup

We parsed the table and converted it into a pandas dataframe.

Flowchart: HTML → BeautifulSoup → Pandas DataFrame

GitHub URL - https://github.com/Arham1505/xyxx/blob/dd8abf7da31f99b4b2085c93a7807b4018685631/jupyter-labs-webscraping.ipynb



In [5]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

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 response.

In [6]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

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

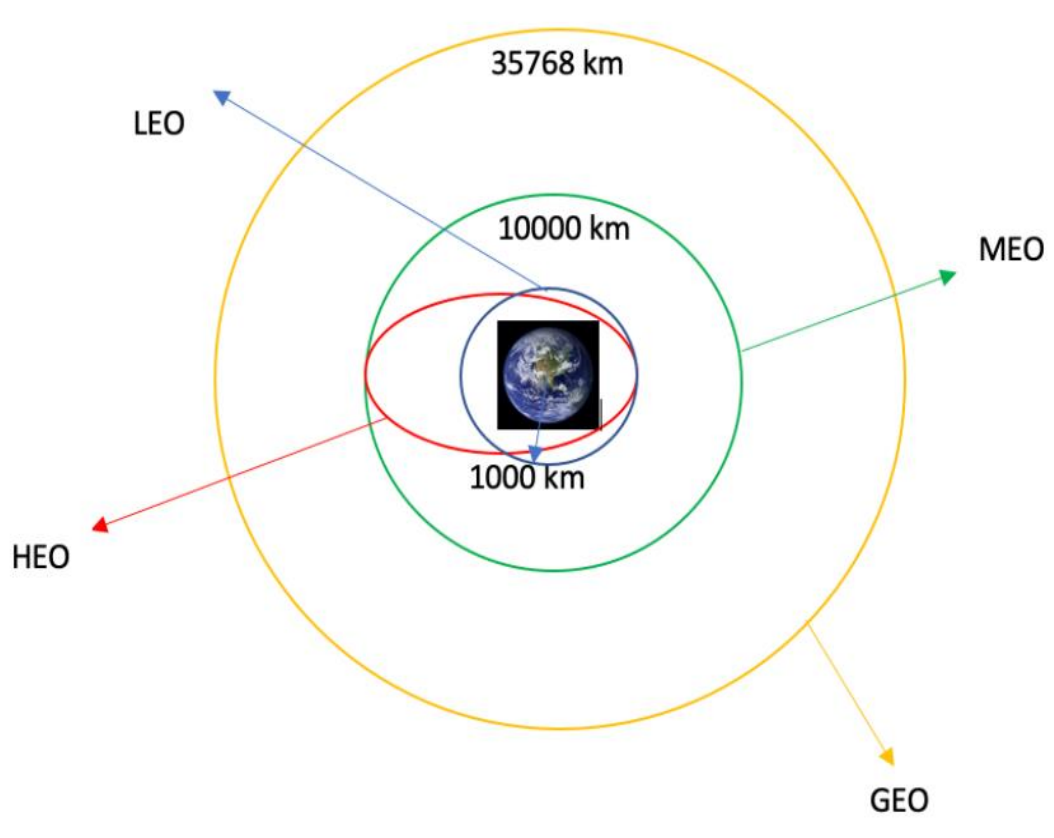
In [7]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, 'html.parser')
```

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

In [8]:
```
# Use soup.title attribute
print(soup.title)
```

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

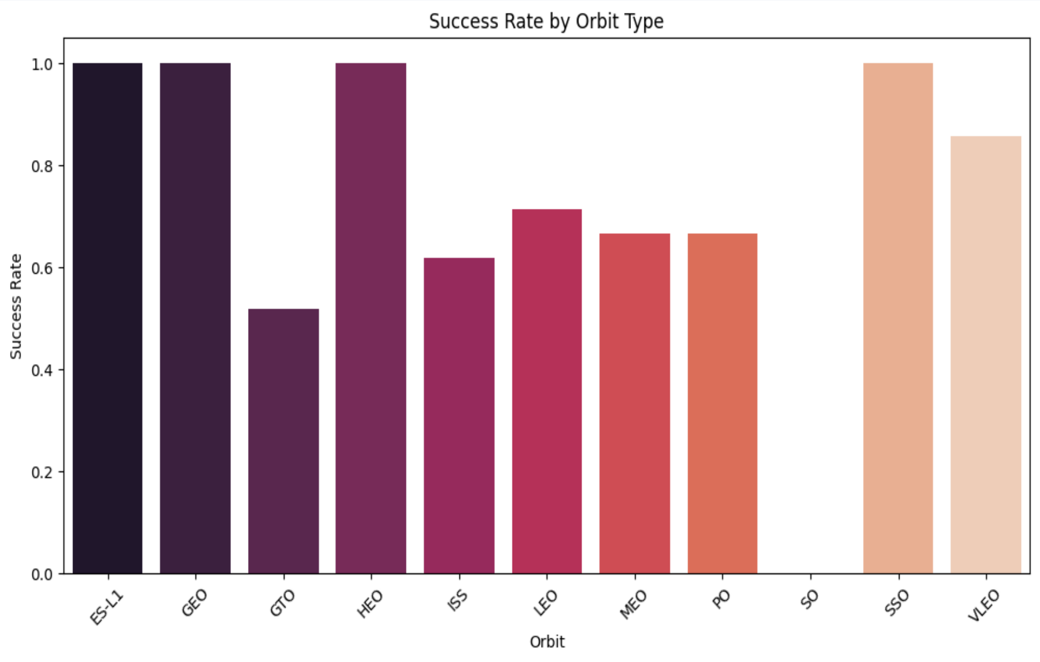
Data Wrangling

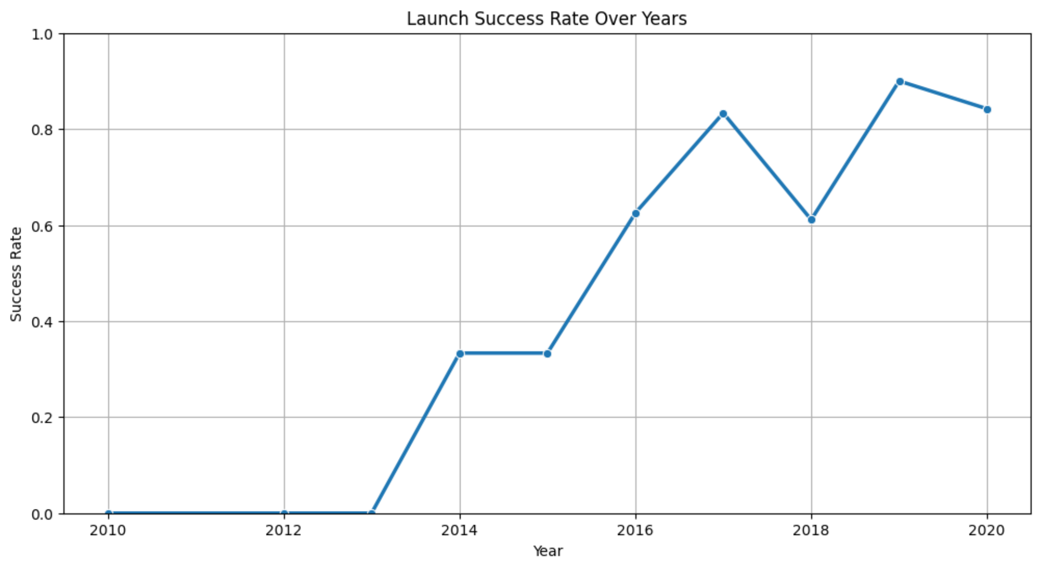


- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- GitHub URL - https://github.com/Arham1505/xyxx/blob/dd8abf7da31f99b4b2085c93a7807b4018685631/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- We explored the data 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, the launch success yearly trend.





- GitHub URL - https://github.com/Arham1505/xyxx/blob/dd8abf7da31f99b4b2085c93a7807b4018685631/edadataviz.ipynb

EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- 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. (SELECT DISTINCT launch_site FROM launches;)
  - 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. (SELECT * FROM launches WHERE launch_site LIKE 'CCA%' LIMIT 5;)
- GitHub URL - https://github.com/Arham1505/xyxx/blob/dd8abf7da31f99b4b2085c93a7807b4018685631/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.
- 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 success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.
- GitHub URL - https://github.com/Arham1505/xyxx/blob/dd8abf7da31f99b4b2085c93a7807b4018685631/lab_jupyter_launch_site_location.ipynb

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.
- GitHub URL - https://github.com/Arham1505/xyxx/blob/dd8abf7da31f99b4b2085c93a7807b4018685631/spacex_dash_app.py

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model is Logistic Regression with an accuracy of 0.83.
- GitHub URL - https://github.com/Arham1505/xyxx/blob/dd8abf7da31f99b4b2085c93a7807b4018685631/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

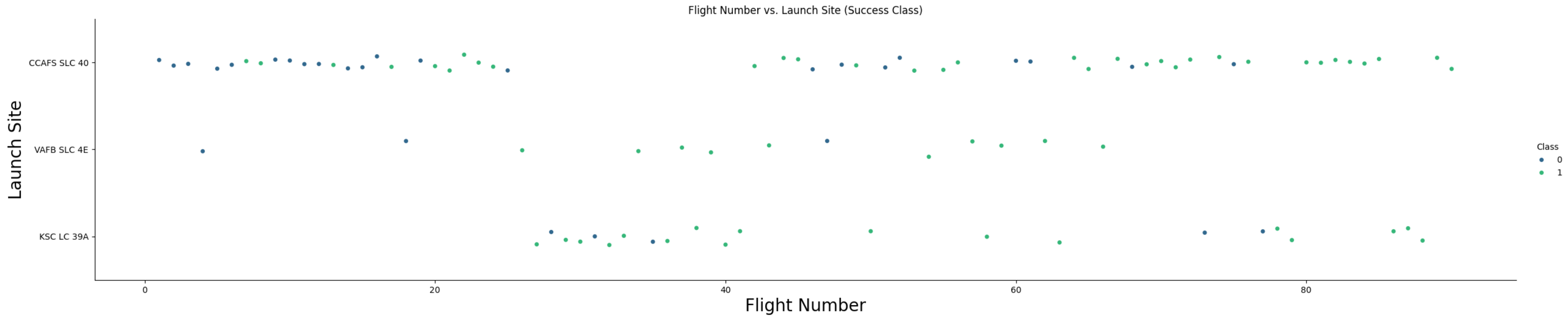
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

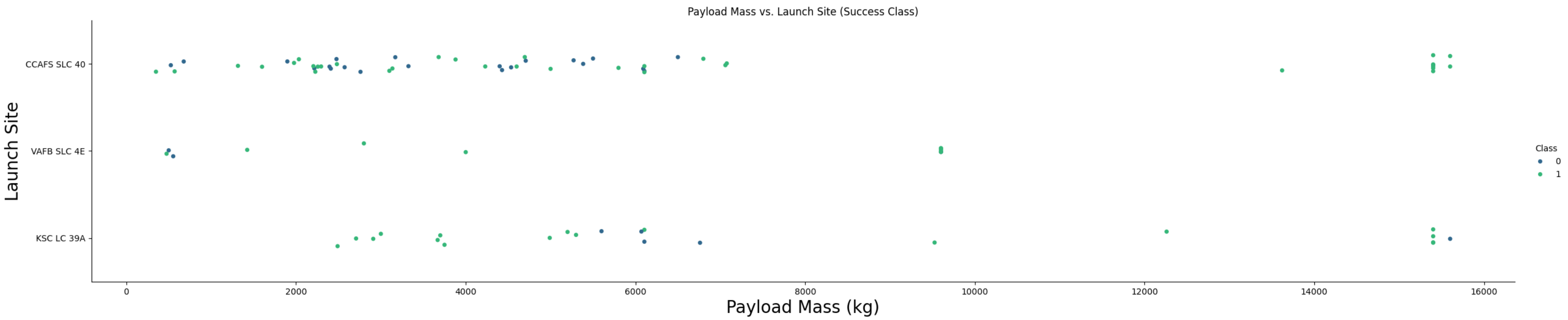
Flight Number vs. Launch Site

- This plot shows that as Flight Number increases, success rates (Class=1) improve for CCAFS SLC-40 and KSC LC-39A, indicating iterative improvements in landing technology



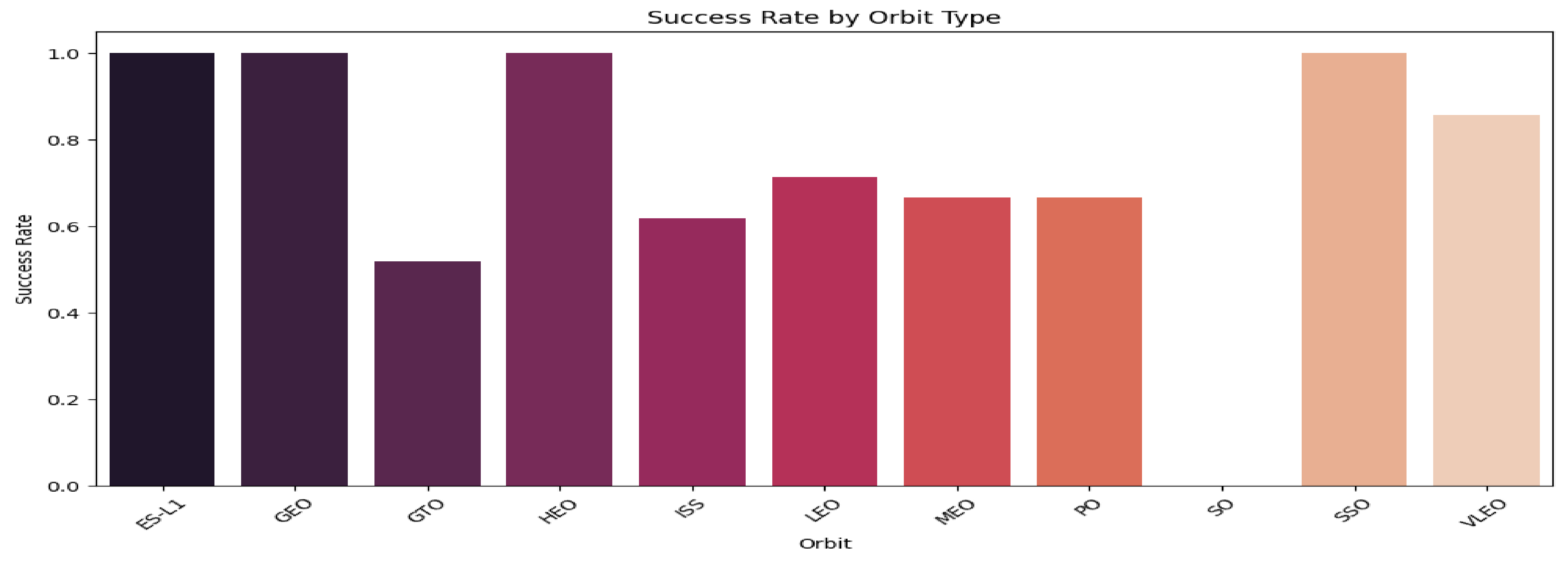
Payload vs. Launch Site

- VAFB SLC-4E has no launches with payloads >10,000 kg, and heavier payloads correlate with lower success rates at other sites.



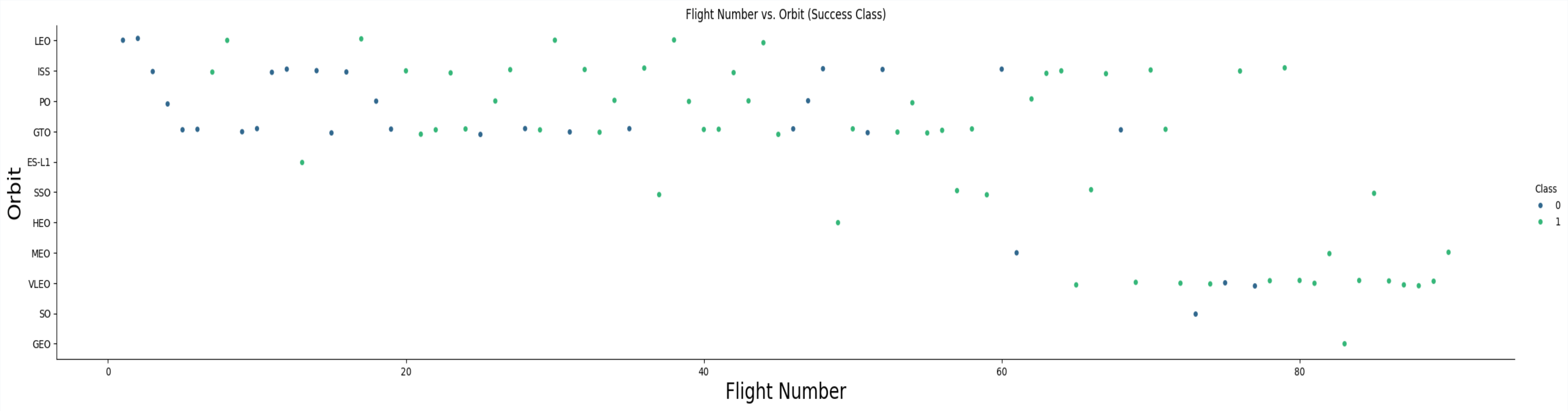
Success Rate vs. Orbit Type

- Orbits like ES-L1, GEO, and ISS have high success rates (>80%), while SO and PO have lower success rates.



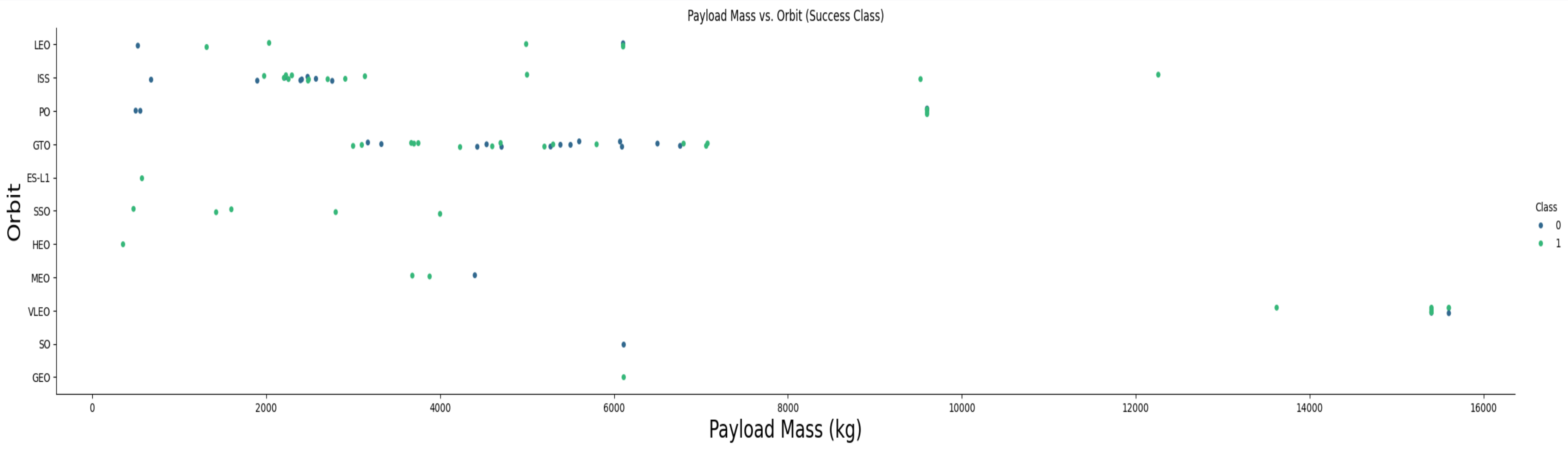
Flight Number vs. Orbit Type

- For LEO, higher Flight Number correlates with more successes. For GTO, no clear trend exists.



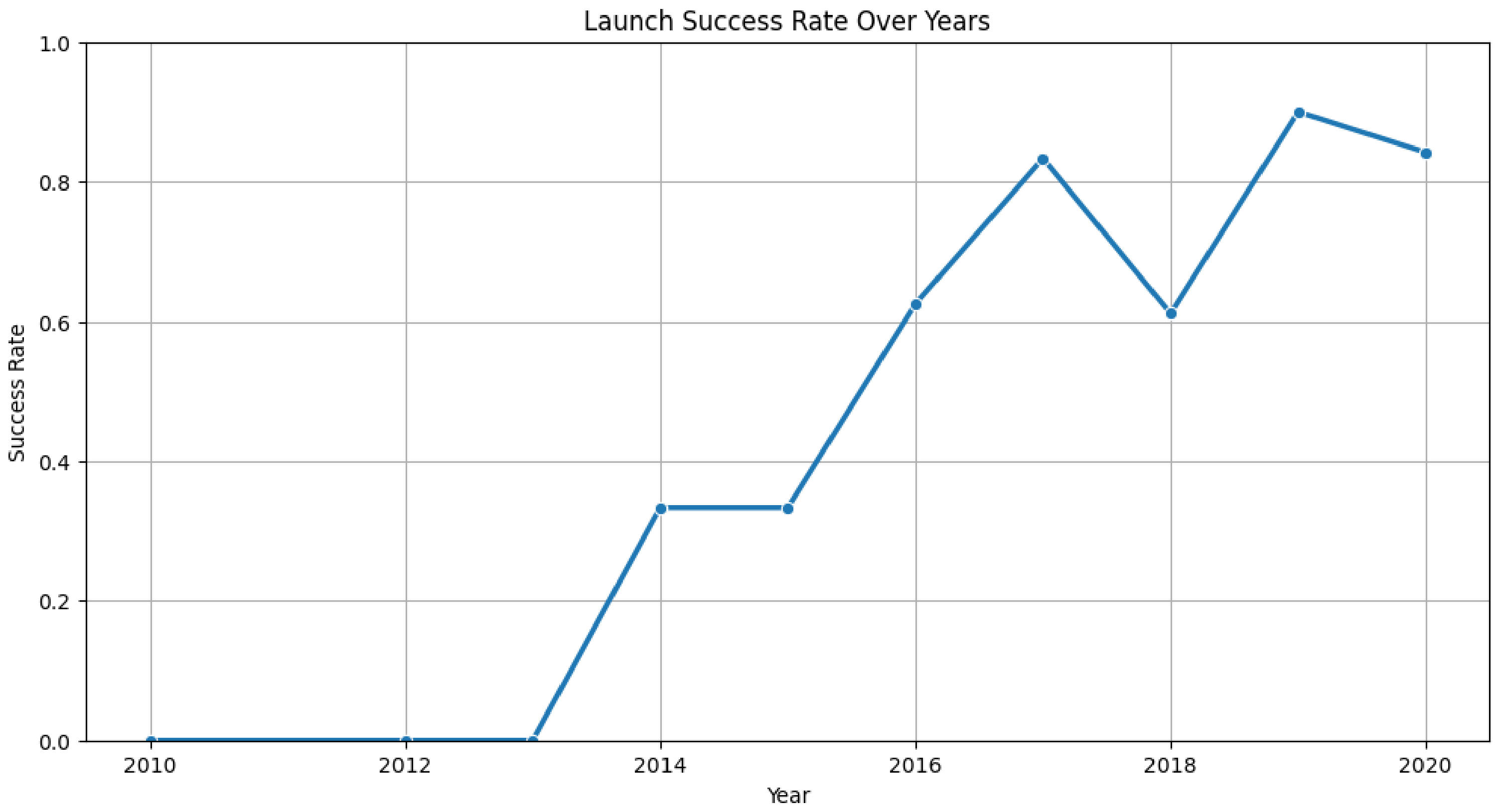
Payload vs. Orbit Type

- Successes in Polar, LEO, and ISS occur across a wide payload range, while GTO has mixed outcomes.



Launch Success Yearly Trend

From the plot, we can observe that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

- There are four unique sites show in the output.
- We used the key word **DISTINCT** to show only unique launch sites from the SPACEXTABLE data.

Task 1

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

In [10]:
```
%%sql
SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

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

Out[10]:

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

Launch Site Names Begin with 'CCA'

- We used below query Like CCA% to get the result. And put Limit of 5, so that 5 record displayed.

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

In [11]:

```
%%sql
SELECT * FROM SPACEXTABLE
WHERE "Launch_Site" LIKE 'CCA%'
LIMIT 5;
```

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

Out[11]:

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

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

Task 3

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

In [12]:

```
%%sql
SELECT SUM("Payload_Mass__kg_") AS Total_Payload_Mass
FROM SPACEXTABLE
WHERE "Customer" = 'NASA (CRS)';
```

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

Out[12]:

Total_Payload_Mass
45596

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.1 is 2928.4
- AVG query is used for the result given below.

Task 4

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

In [13]:

```
%%sql
SELECT AVG("Payload_Mass__kg_") AS Avg_Payload_Mass
FROM SPACEXTABLE
WHERE "Booster_Version" = 'F9 v1.1';
```

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

Out[13]:

Avg_Payload_Mass
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

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

In [14]:
```
%%sql
SELECT MIN(Date) AS First_Successful_Ground_Landing
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (ground pad)';
```

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

Out[14]:

First_Successful_Ground_Landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass between 4000 and 6000.

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [15]:

```
%%sql
SELECT DISTINCT "Booster_Version"
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (drone ship)'
AND "Payload_Mass__kg_" BETWEEN 4000 AND 6000;
```

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

Out[15]:

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 total number of successful mission outcomes are 100 and failure mission outcome is 1only.
- Simple query of GROUP BY is used to get the results.

Task 7

List the total number of successful and failure mission outcomes

In [16]:

```
%%sql
SELECT "Mission_Outcome", COUNT(*) AS Outcome_Count
FROM SPACEXTABLE
GROUP BY "Mission_Outcome";
```

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

Out[16]:

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

Boosters Carried Maximum Payload

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

In [17]:
```
%%sql
SELECT "Booster_Version"
FROM SPACEXTABLE
WHERE "Payload_Mass__kg_" = (
    SELECT MAX("Payload_Mass__kg_")
    FROM SPACEXTABLE
);
```

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

Out[17]:

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

- We determined the booster that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.

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.

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

- 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 2010-03-20.
- 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.

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

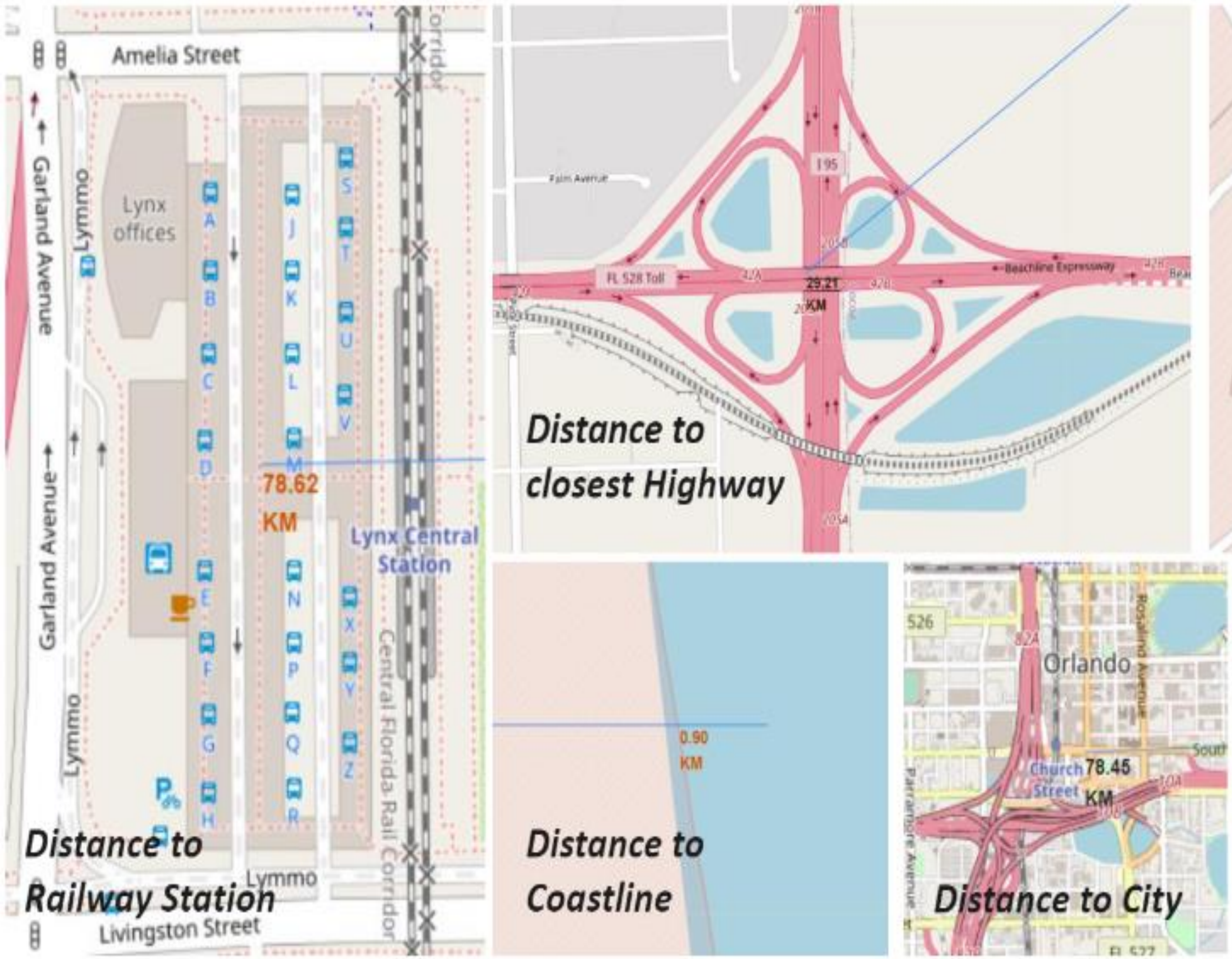
Launch Sites Proximities Analysis

All launch sites global map markers

- We can see that the SpaceX launch sites are in the United stated coasts are near to the Florida and California cities.

Markers showing launch sites with color labels

Launch Site distance to landmarks

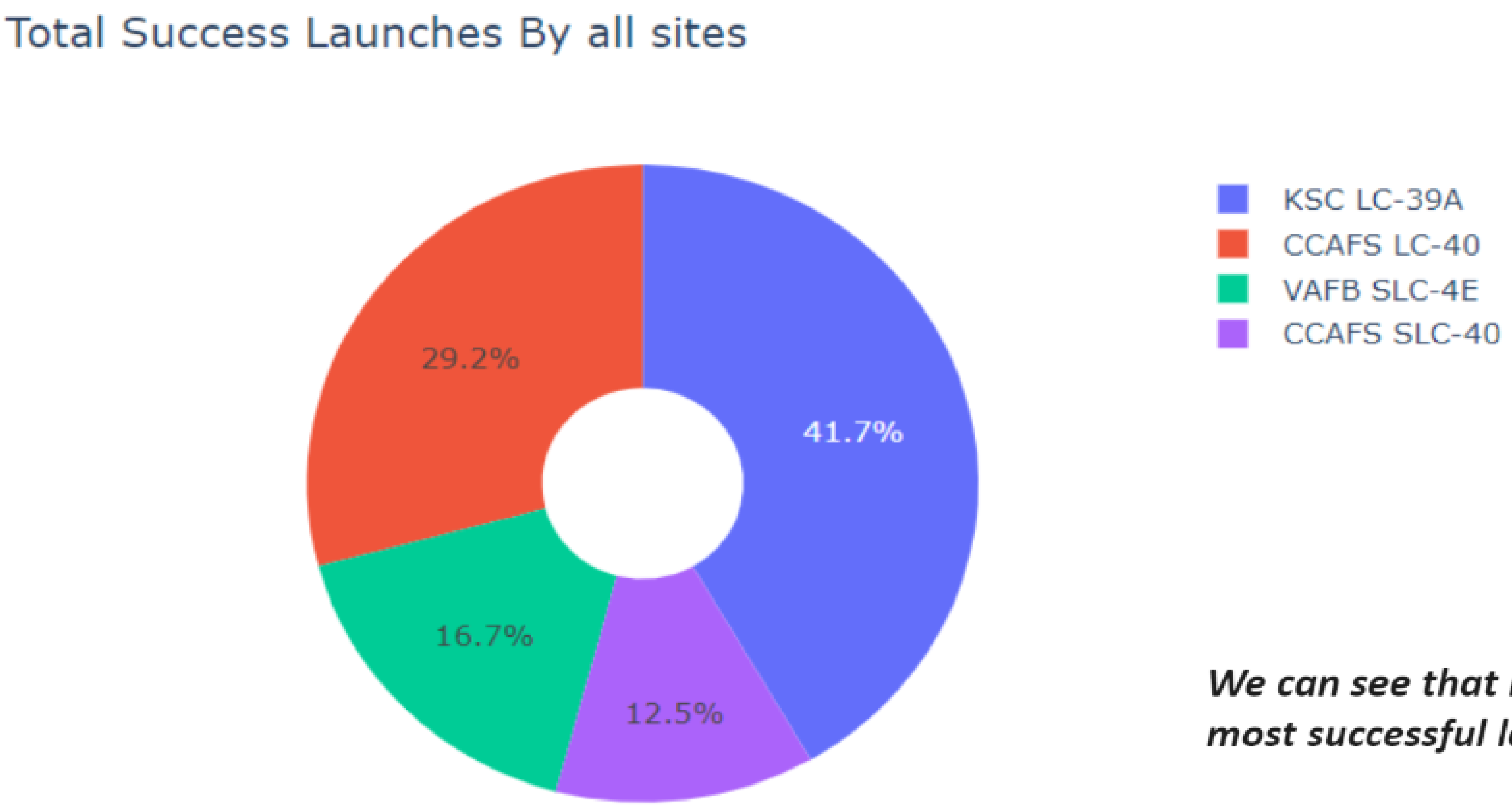


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

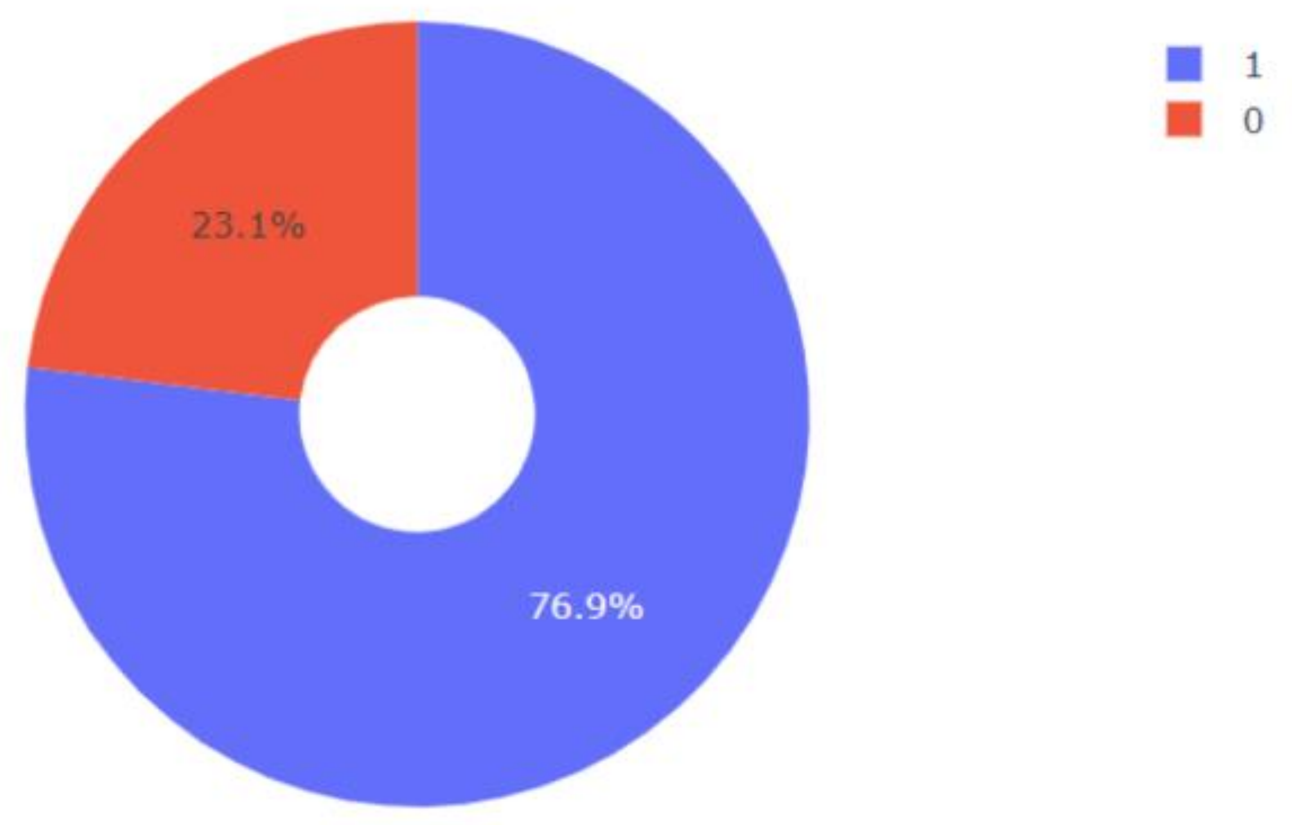
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



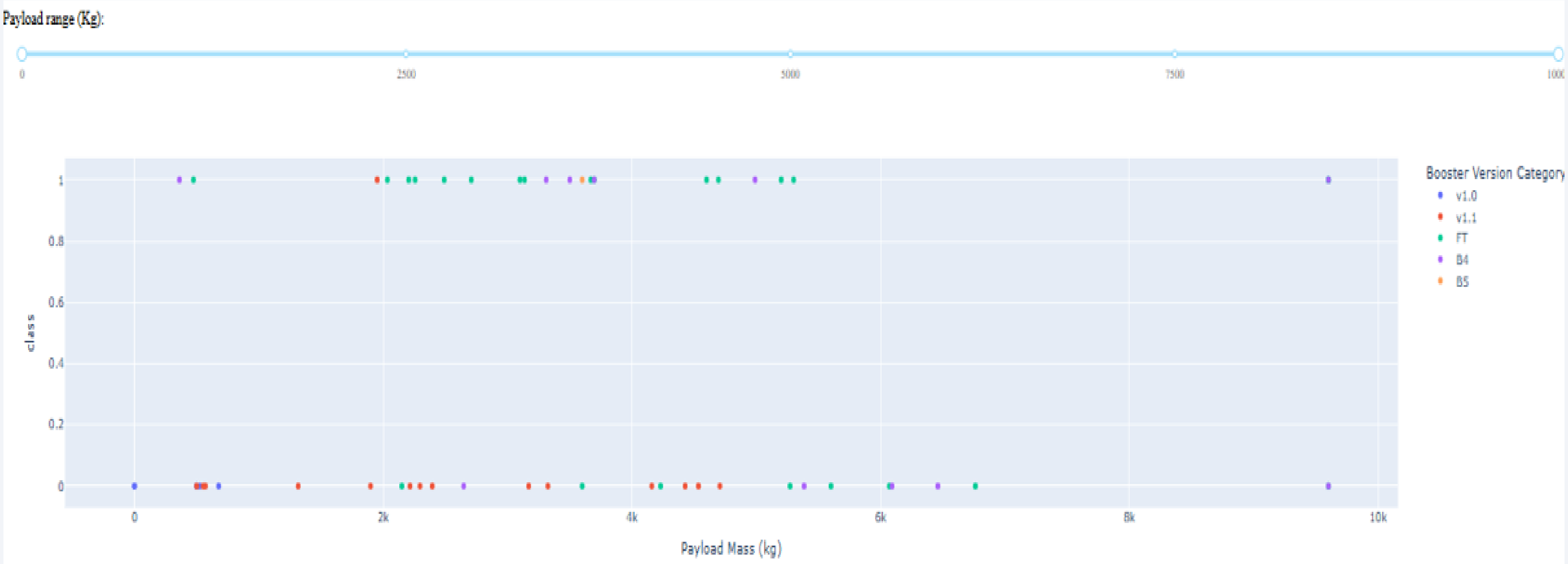
We can see that KSC LC-39A had the most successful launches from all the sites

Pie chart showing the Launch site with the highest launch success ratio



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

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider



- FT booster version is having high success rate and v1.1 version is having high failures. Success also depends on the payload range.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The Logistic Regression classifier is the model with the highest classification accuracy

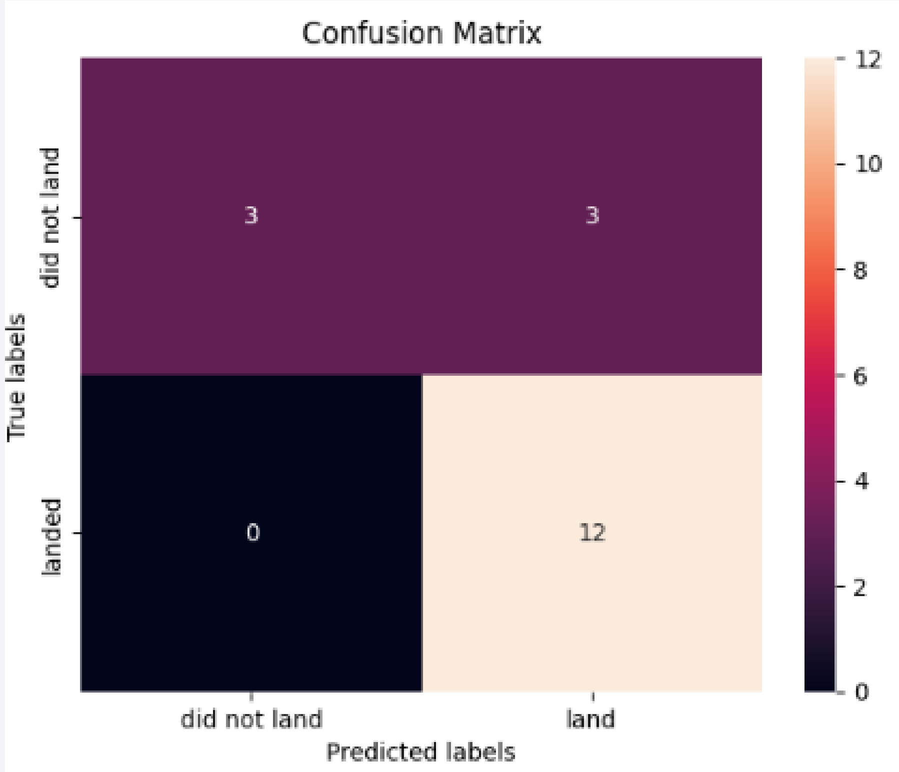
In [39]:
```
scores = {
    'Logistic Regression': logreg_score,
    'SVM': svm_score,
    'Decision Tree': tree_score,
    'KNN': knn_score
}

best_model = max(scores, key=scores.get)
print(f"The best performing model is {best_model} with an accuracy of {scores[best_model]:.2f}")
```

```
The best performing model is Logistic Regression with an accuracy of 0.83
```

Confusion Matrix

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



Conclusions

We can conclude that:

- 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 Logistic Regression classifier is the best machine learning algorithm for this task.

Thank you!