IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Shitanshu Ghildiyal
28th September, 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 Analysis with Folium
  - Machine Learning Prediction

- Summary of all results

  - Exploratory Data Analysis Result
  - Interactive Analytics in Screenshots
  - Predictive Analytics Result

# Introduction

- ## Project background and context

SpaceX offers Falcon 9 rocket launches at a competitive price of $62 million on its website, significantly undercutting other providers whose costs can go as high as $165 million per launch. A substantial portion of this cost-saving advantage stems from SpaceX's pioneering ability to reuse the first stage of the rocket. Therefore, by accurately predicting whether the first stage will successfully land after launch, we can effectively estimate the overall cost of a rocket launch.

This information holds valuable implications for potential competitors seeking to bid against SpaceX for rocket launch contracts. The primary objective of this project is to develop a machine learning pipeline that can predict with precision whether the first stage of the Falcon 9 rocket will achieve a successful landing.

- ## Problems you want to find answers

- What factors determine if the rocket will land successfully?
- 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
  - One-hot encoding was used on 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 was collected using various methods:

- Data collection was done using get request to the SpaceX API.

- Then, we decoded the response content as Json using .json() function call and turn it into a pandas dataframe using .json_normalize().

- Then the data was cleaned and checked for missing values. The missing values were replaced by mean wherever it was necessary.

- We performed web scraping from Wikiopedia for Flacon 9 launch records using BeautifulSoup.

- Here, extracted data from HTML table, parse the table and convert it to a pandas dataframe for further 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.
- The link to the notebook is https://github.com/ghildiyalshitanshu/CapstoneProjectIBM/blob/main/Data%20Collection%20API.ipynb

1. Get request for rocket launch data using API

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

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

2. Use json_normalize method to convert json result to dataframe

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

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

```
# Replace the np.nan values with its mean value
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 applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The link to the notebook is [https://github.com/ghildiyalshitanshu/CapstoneProjectIBM/blob/main/Data%20Collection%20-%20Web%20Scraping.ipynb](https://github.com/ghildiyalshitanshu/CapstoneProjectIBM/blob/main/Data%20Collection%20-%20Web%20Scraping.ipynb)

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

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

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

```
200
```

2. Create a BeautifulSoup object from the HTML response

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

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

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

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

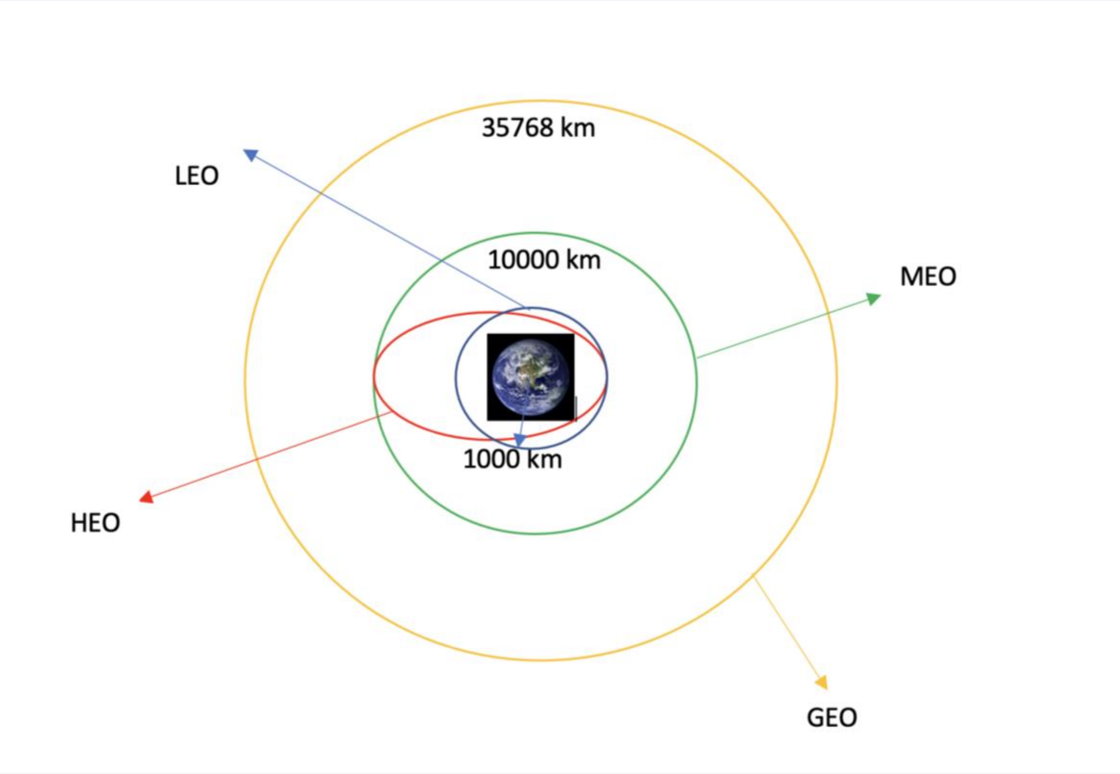
Extract all column names from the HTML table header

```
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
```

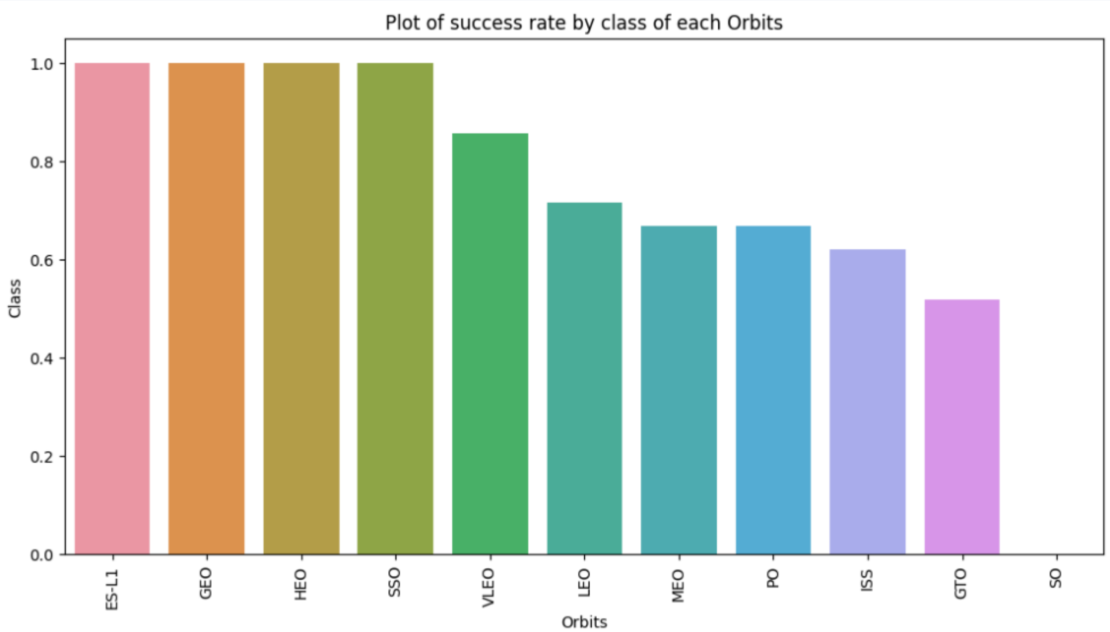
# Data Wrangling

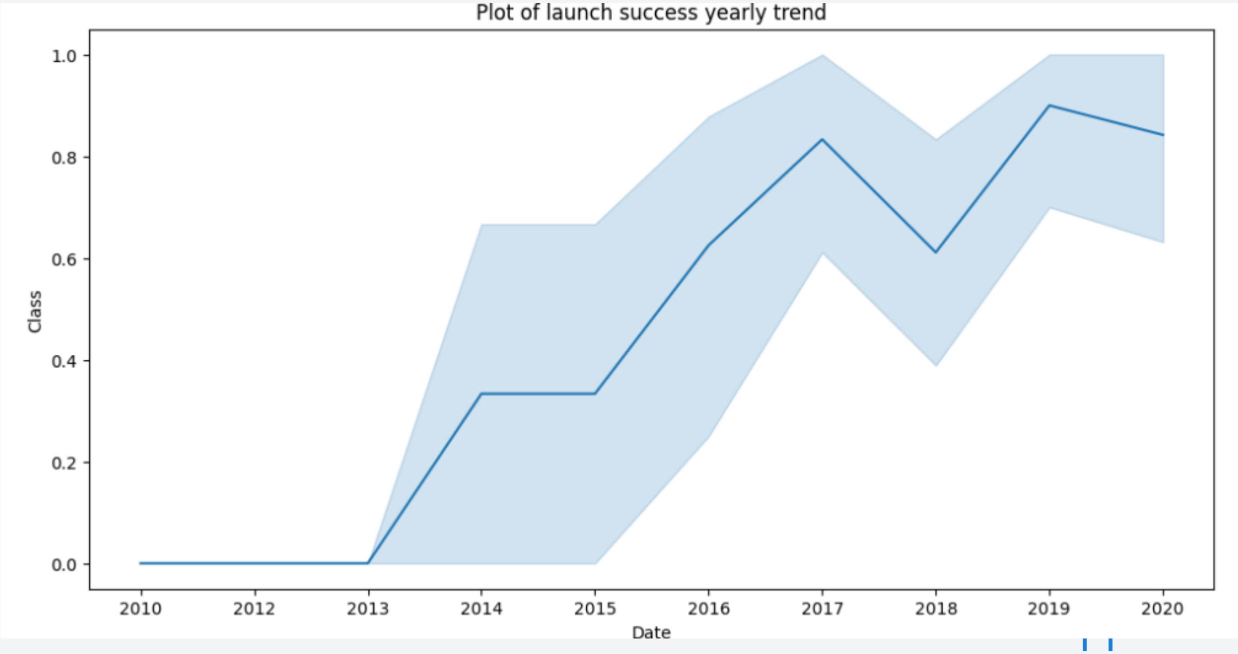
- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits.
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is https://github.com/ghildiyalshitanshu/CapstoneProjectIBM/blob/main/Data%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.
- The link to the notebook is



# EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
- We loaded the SpaceX dataset into a SQL database.
- We applied EDA with SQL to get insights from data. We wrote queries to find out:

  - 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 link to the notebook is https://github.com/ghildiyalshitanshu/CapstoneProjectIBM/blob/main/EDA%20with%20SQL.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.
- The link to the notebook is https://github.com/ghildiyalshitanshu/CapstoneProjectIBM/blob/main/Interactive%20Visual%20Analytics%20with%20Folium%20lab.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.
- The link to the notebook is https://github.com/ghildiyalshitanshu/CapstoneProjectIBM/blob/main/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.
- The link to the notebook is https://github.com/ghildiyalshitanshu/CapstoneProjectIBM/blob/main/Machine%20Learning%20Prediction%20lab.ipynb

# Results

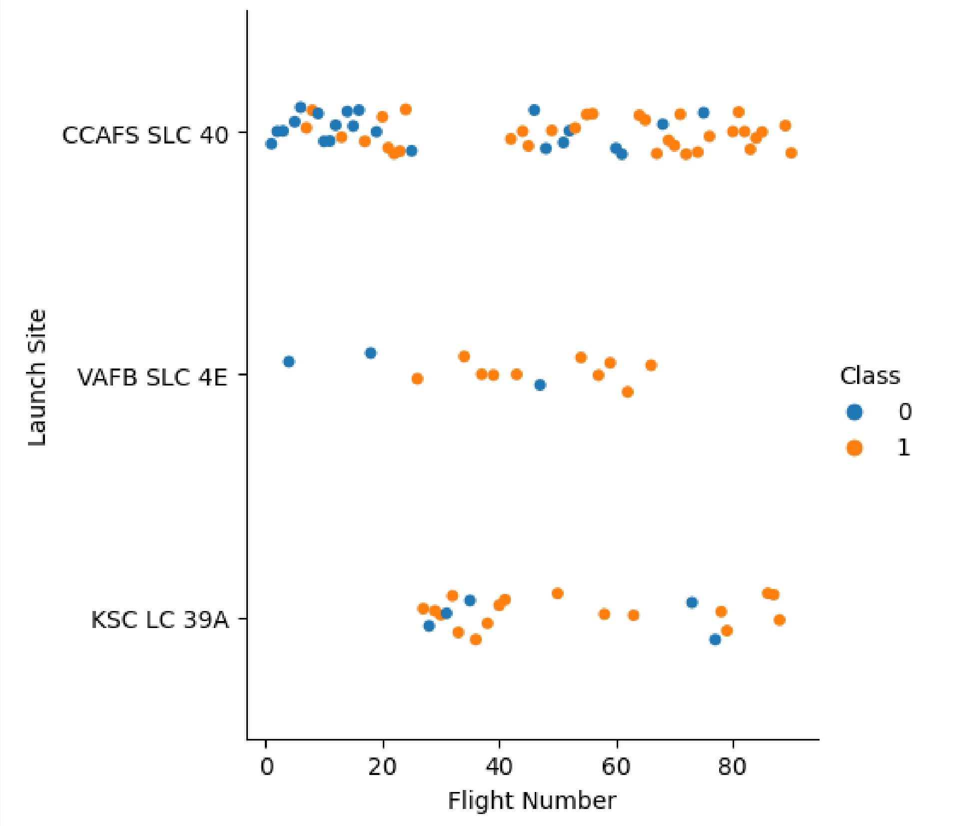
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

# Insights drawn from EDA

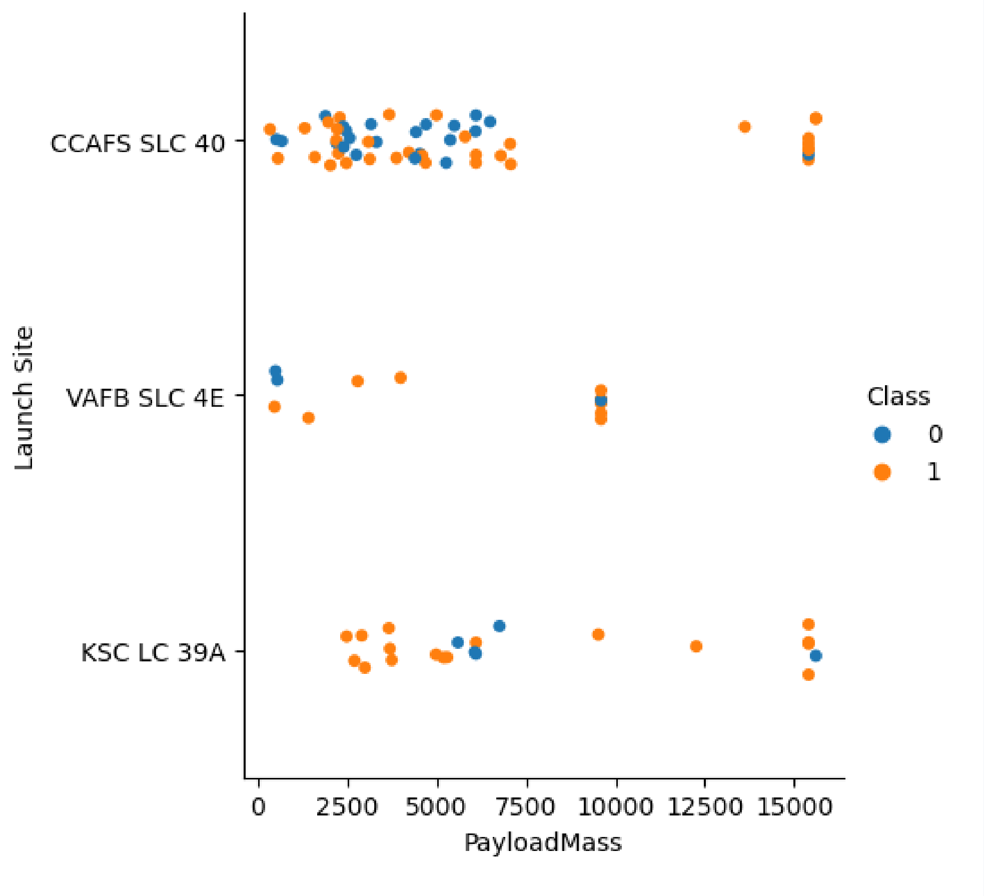
# Flight Number vs. Launch Site

- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.



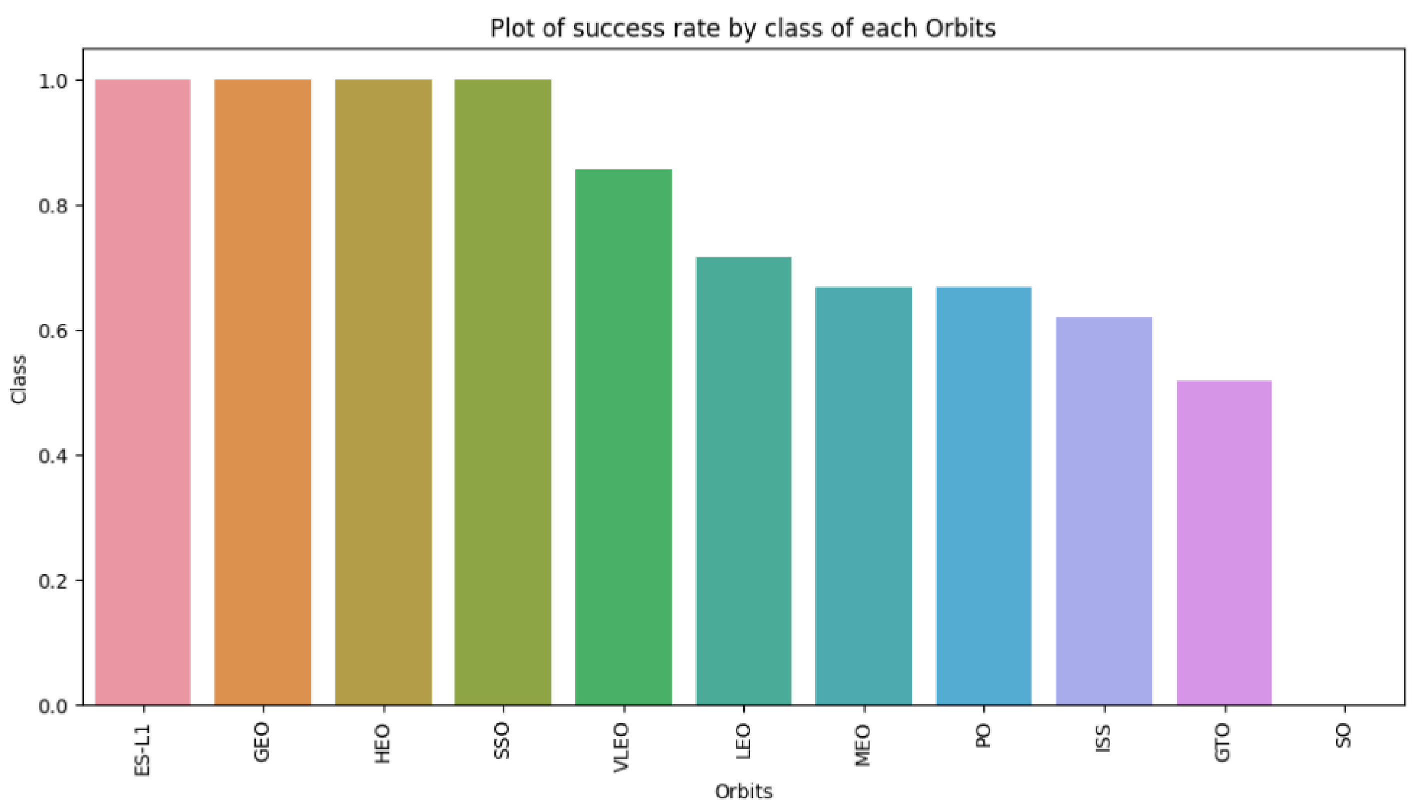
# Payload vs. Launch Site

- Payload vs. Launch Site



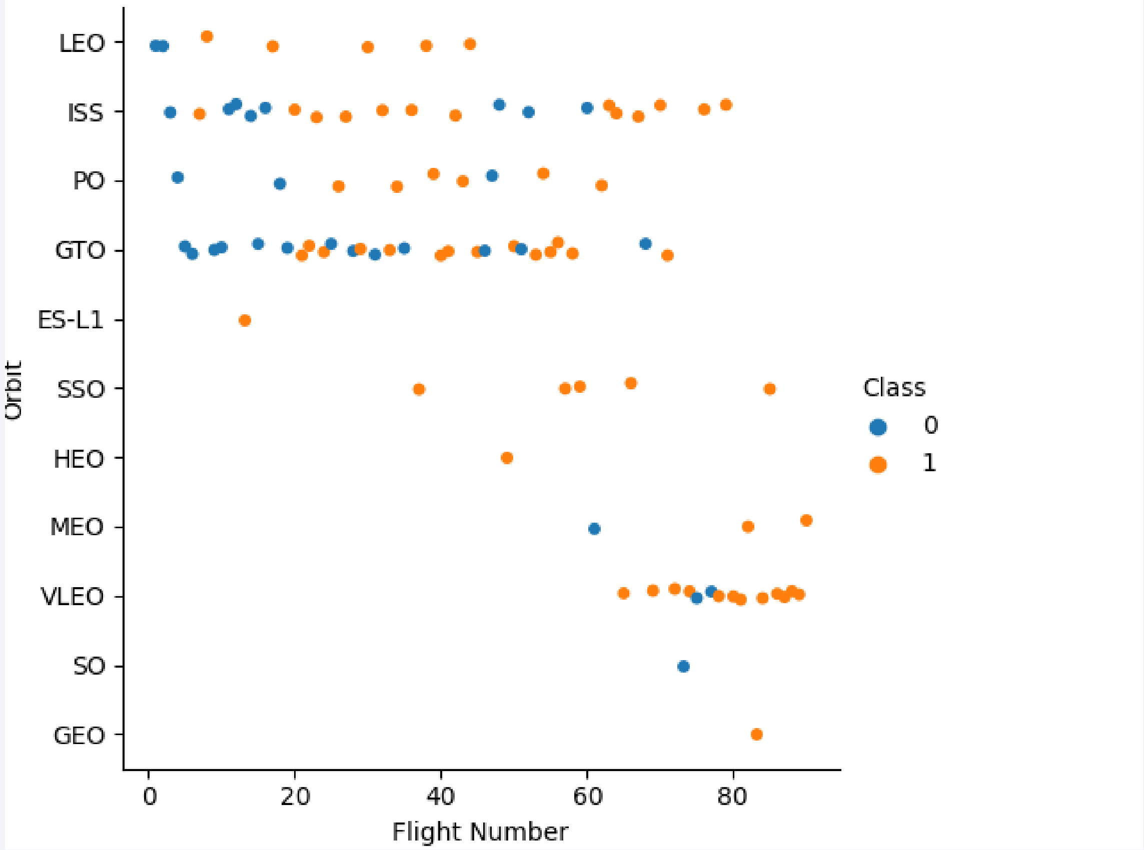
# Success Rate vs. Orbit Type

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



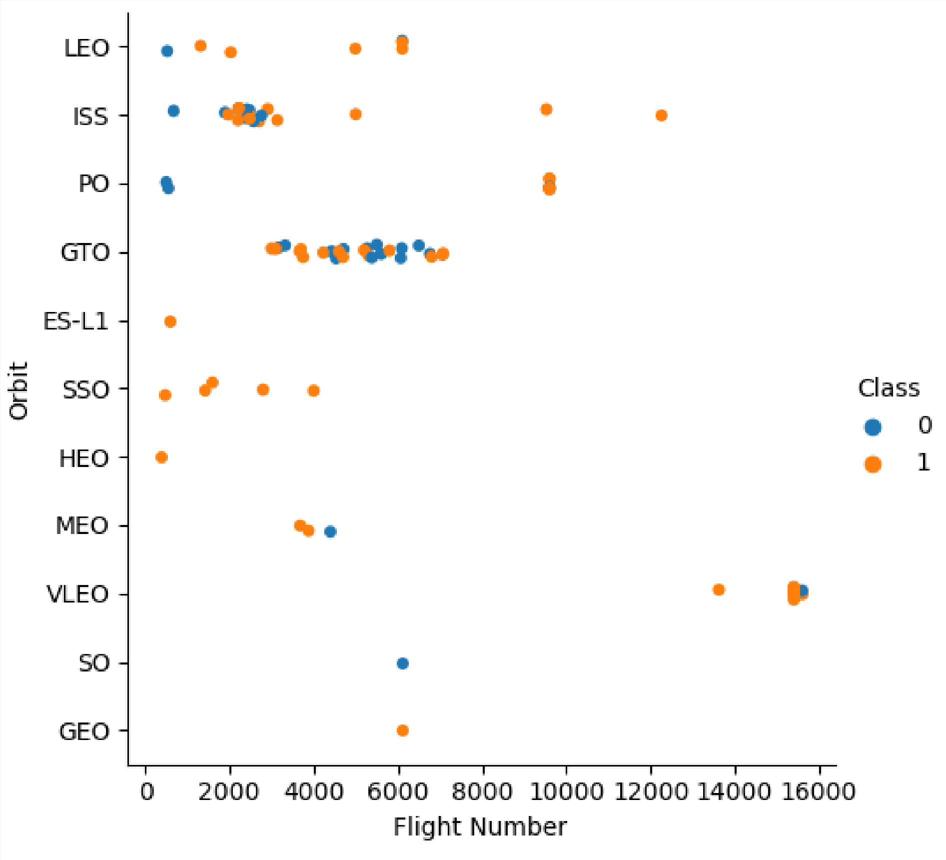
# Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



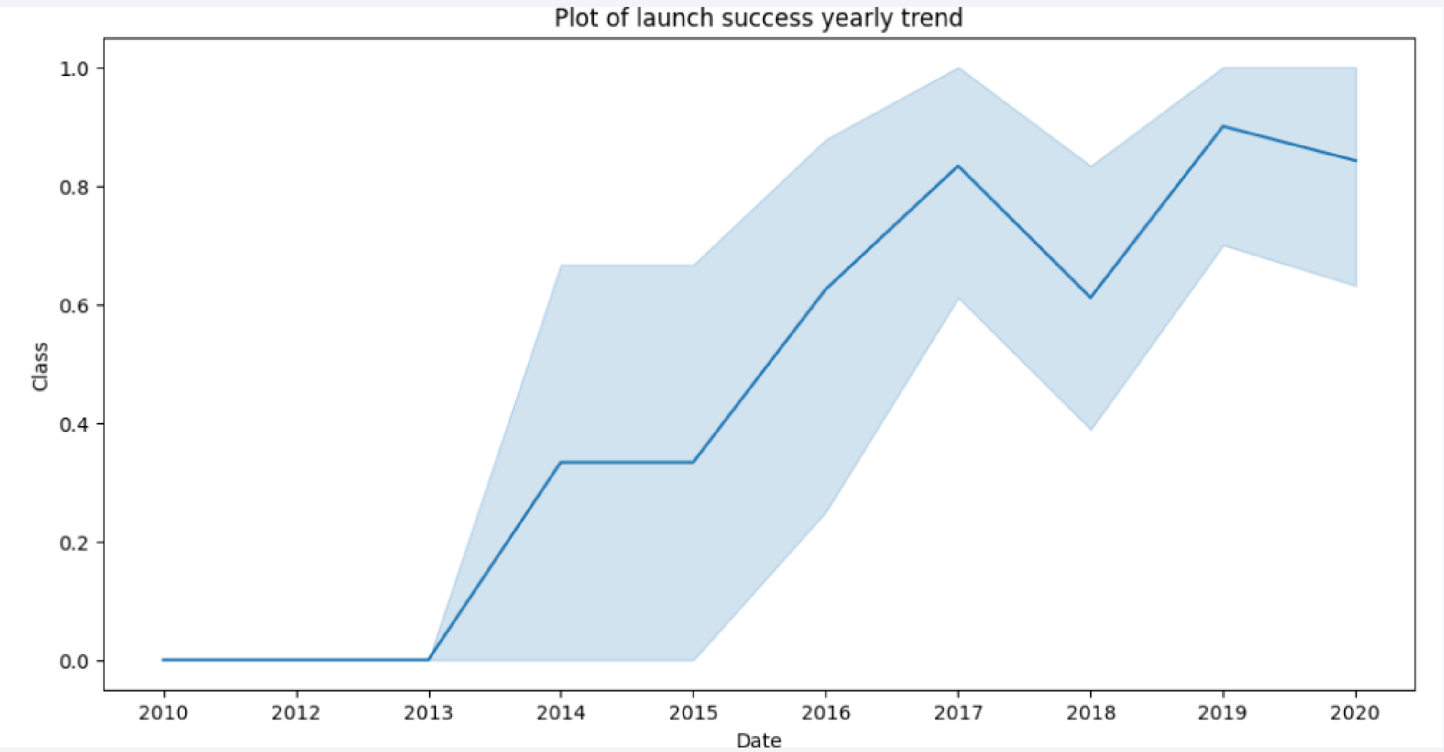
# Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



# Launch Success Yearly Trend

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



# All Launch Site Names

- We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

```
%sql select DISTINCT(Launch_Site) from SPACEXTABLE
```

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

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

# Launch Site Names Begin with 'CCA'

- We used the query above to display 5 records where launch sites begin with \`CCA\`

```
%sql SELECT * from SPACEXTABLE where Launch_Site like 'CCA%' limit(5)
```

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

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing_Outcome |
|---|---|---|---|---|---|---|---|---|---|
| 2010-04-06 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success | Failure (parachute) |
| 2010-08-12 | 15:43:00 | F9 v1.0 B0004 | CCAFS LC-40 | Dragon demo flight C1, two CubeSats, barrel of Brouere cheese | 0 | LEO (ISS) | NASA (COTS) NRO | Success | Failure (parachute) |
| 2012-05-22 | 07:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success | No attempt |
| 2012-08-10 | 00:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500 | LEO (ISS) | NASA (CRS) | Success | No attempt |
| 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

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS FROM SPACEXTABLE where Customer = 'NASA (CRS)'
```

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

| TOTAL_PAYLOAD_MASS |
|---|
| 45596 |

# Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'
```

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

| AVG(PAYLOAD_MASS__KG_) |
| ---: |
| 2928.4 |

# First Successful Ground Landing Date

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

```
%sql SELECT MIN(DATE) FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)'
```

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

| MIN(DATE) |
|---|
| 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 greater than 4000 but less than 6000

```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_>= 4000 AND PAYL
```

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

- We used wildcard like ‘%’ to filter for **WHERE** MissionOutcome was a success or a failure.

```
%sql SELECT COUNT(Mission_Outcome) AS Successful_Outcome FROM SPACEXTABLE WHERE Mission_Outcome LIKE 'Success%'
```

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

| Successful_Outcome |
|---|
| 100 |

# Boosters Carried Maximum Payload

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

```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ =  (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE ) ORDER E
```

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

| Booster_Version |
|---|
| 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

- 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

```
%sql SELECT SUBSTR(Date,4,2) AS Month, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTABLE WHERE Landing_Outcome LIKE
```

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

| Month | Landing_Outcome | Booster_Version | Launch_Site |
|---|---|---|---|
| 5- | Failure (drone ship) | F9 v1.1 B1012 | CCAFS LC-40 |
| 5- | Failure (drone ship) | F9 v1.1 B1015 | CCAFS LC-40 |

# 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.

```
%sql SELECT Landing_Outcome, COUNT(Landing_Outcome) FROM SPACEXTABLE WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY La
```

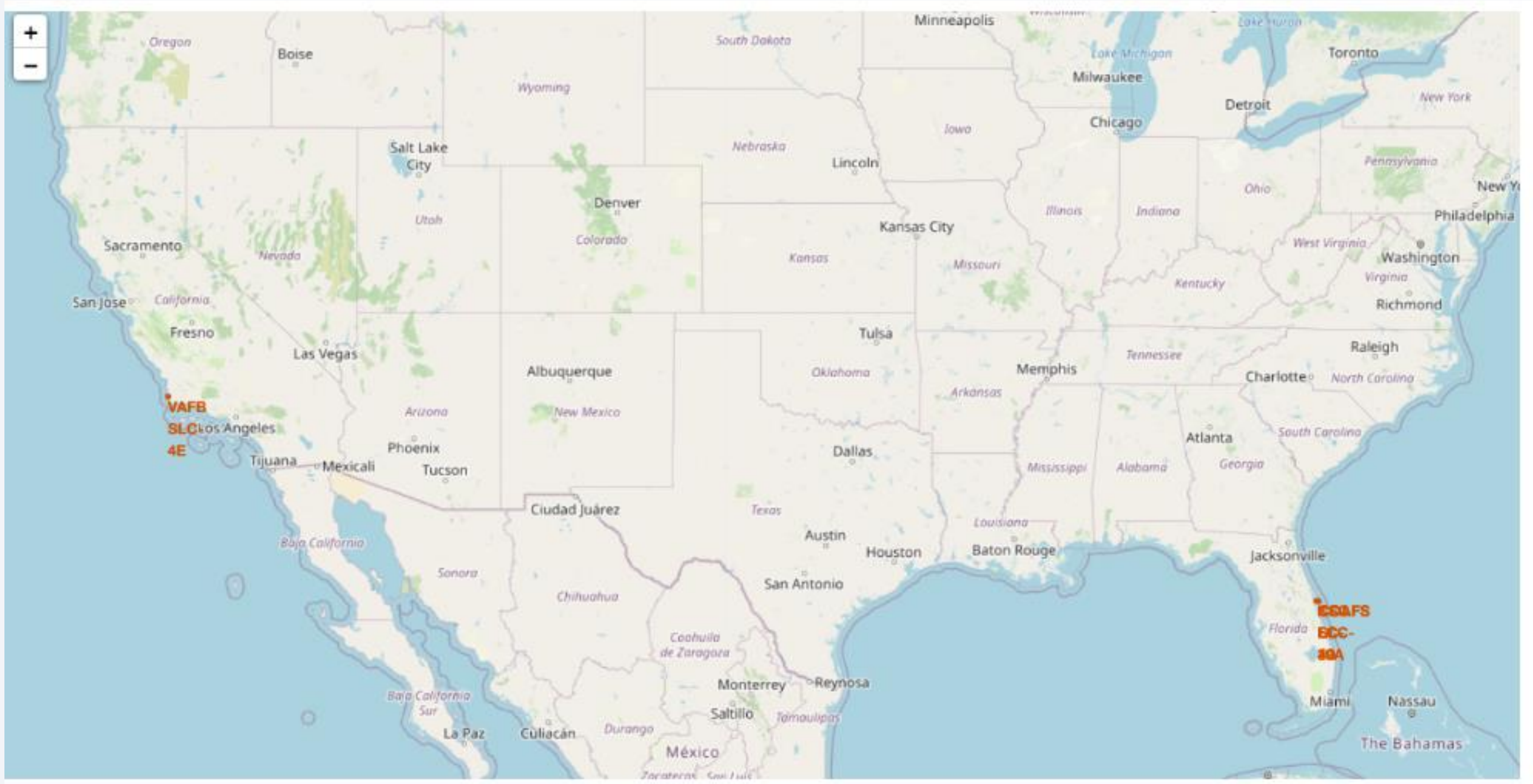
```
 * sqlite:///my_data1.db
Done.
```

| Landing_Outcome | COUNT(Landing_Outcome) |
|---|---|
| No attempt | 10 |
| Success (ground pad) | 5 |
| Success (drone ship) | 5 |
| Failure (drone ship) | 5 |
| Controlled (ocean) | 3 |
| Uncontrolled (ocean) | 2 |
| Precluded (drone ship) | 1 |
| Failure (parachute) | 1 |

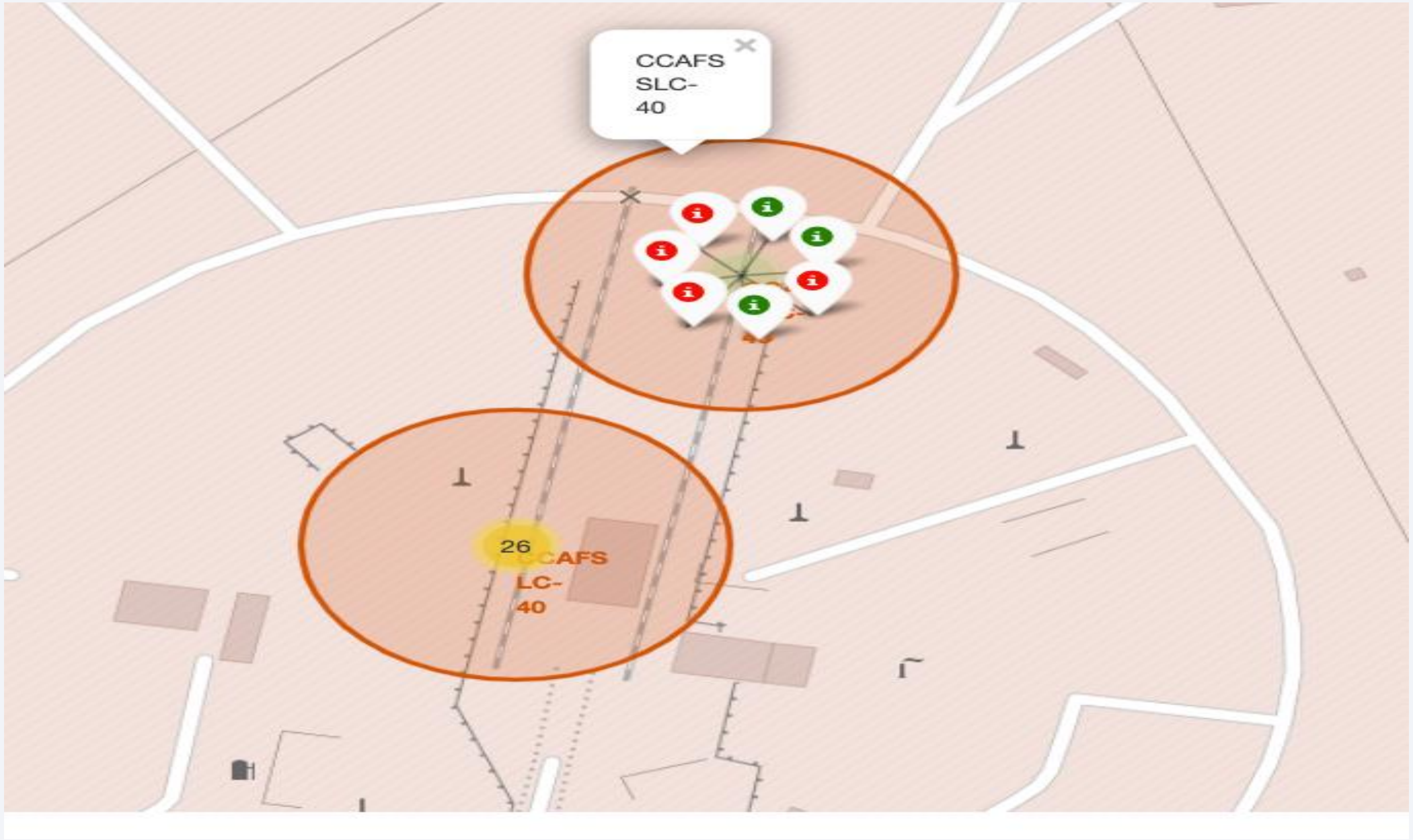
Section 3

# Launch Sites Proximities Analysis

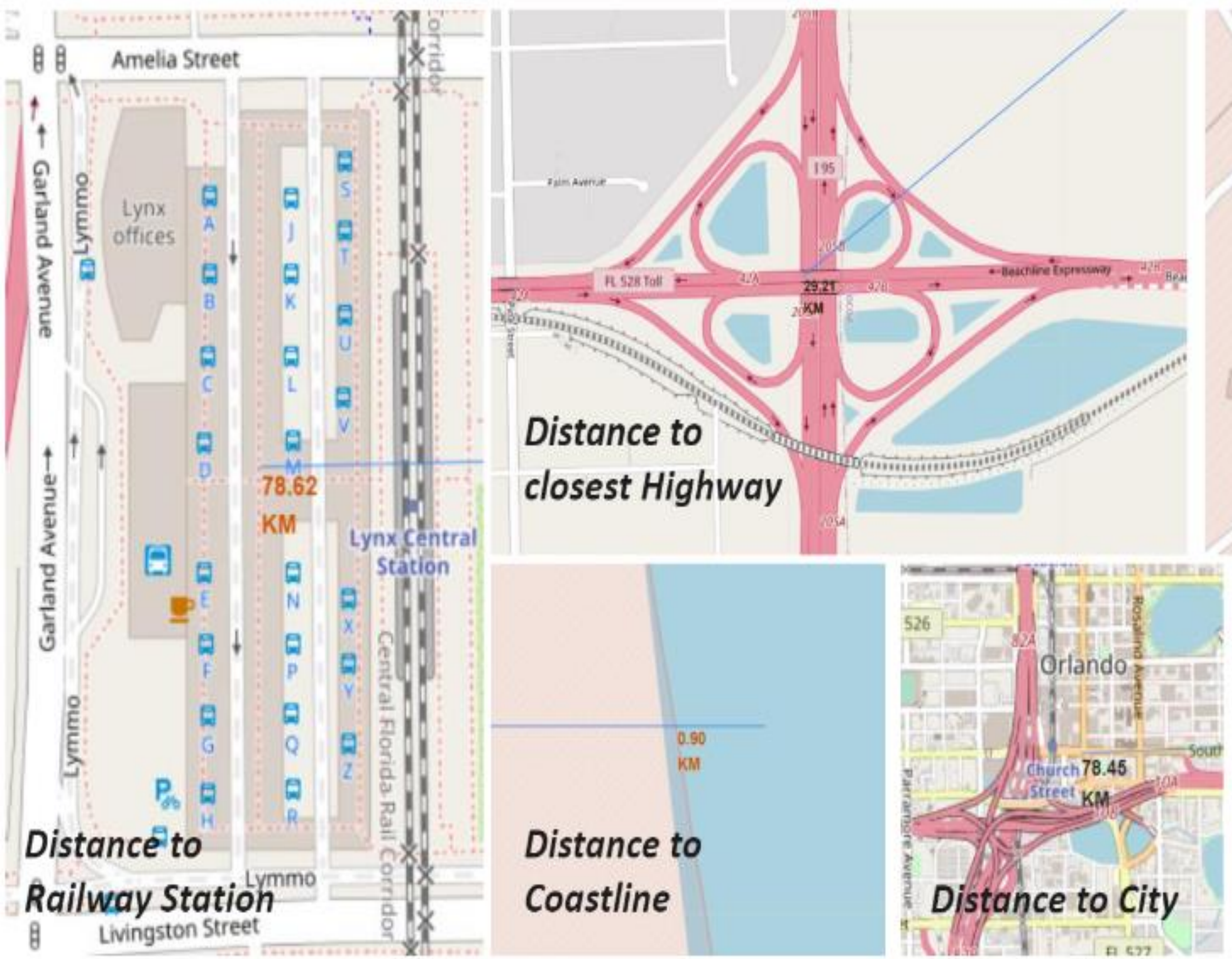
# All launch sites global map markers



# Markers showing launch sites with color labels



# Launch Site distance to landmarks



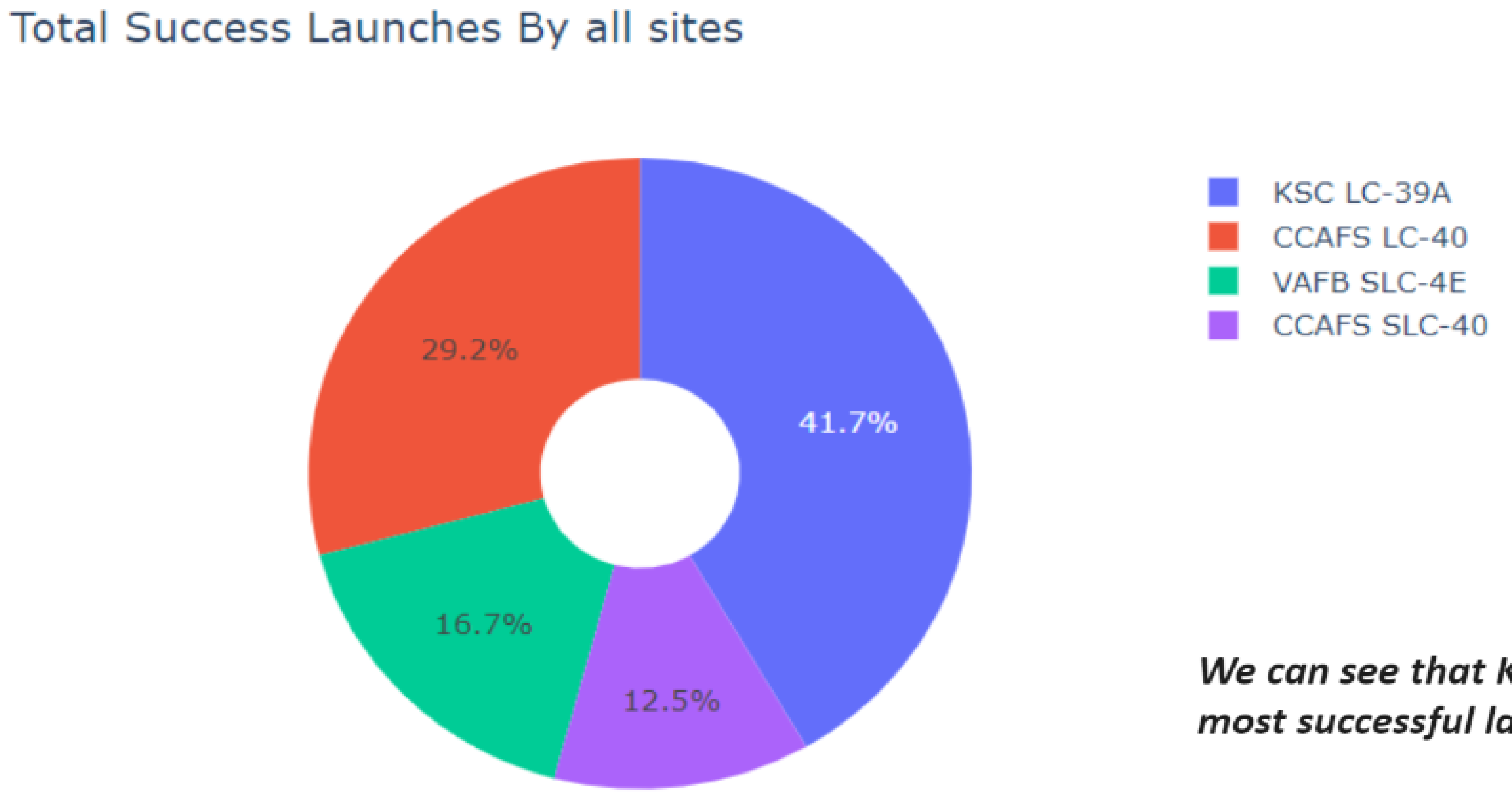


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

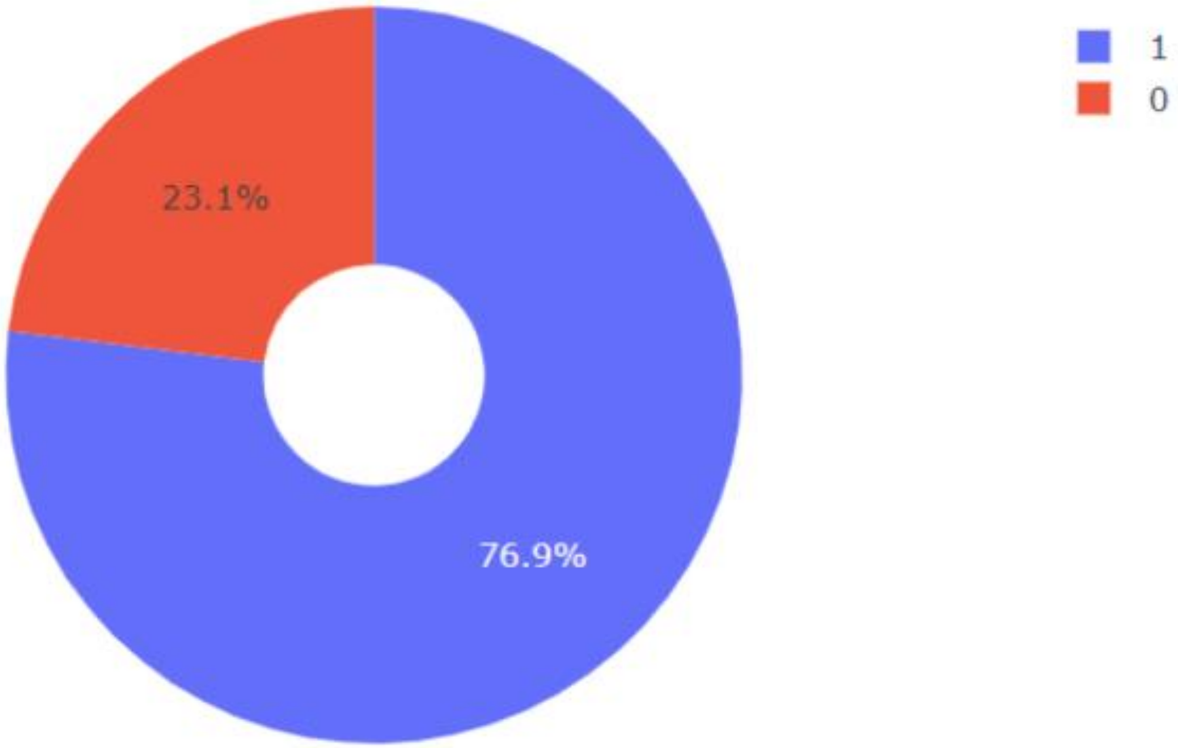
# Build a Dashboard with Plotly Dash

# Pie chart showing the success percentage achieved by each launch site



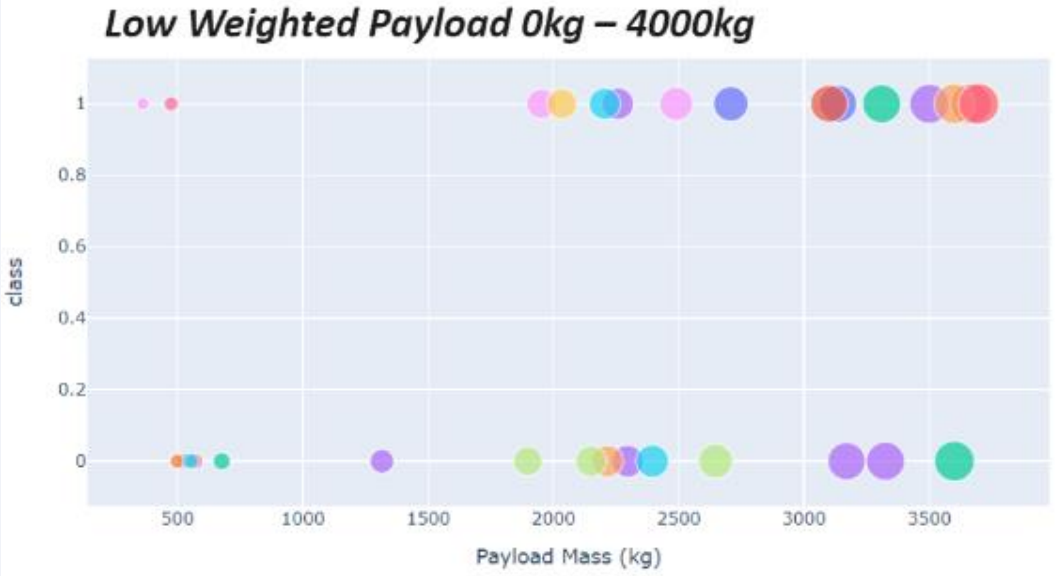

***We can see that KSC LC-39A had the most successful launches from all the sites***

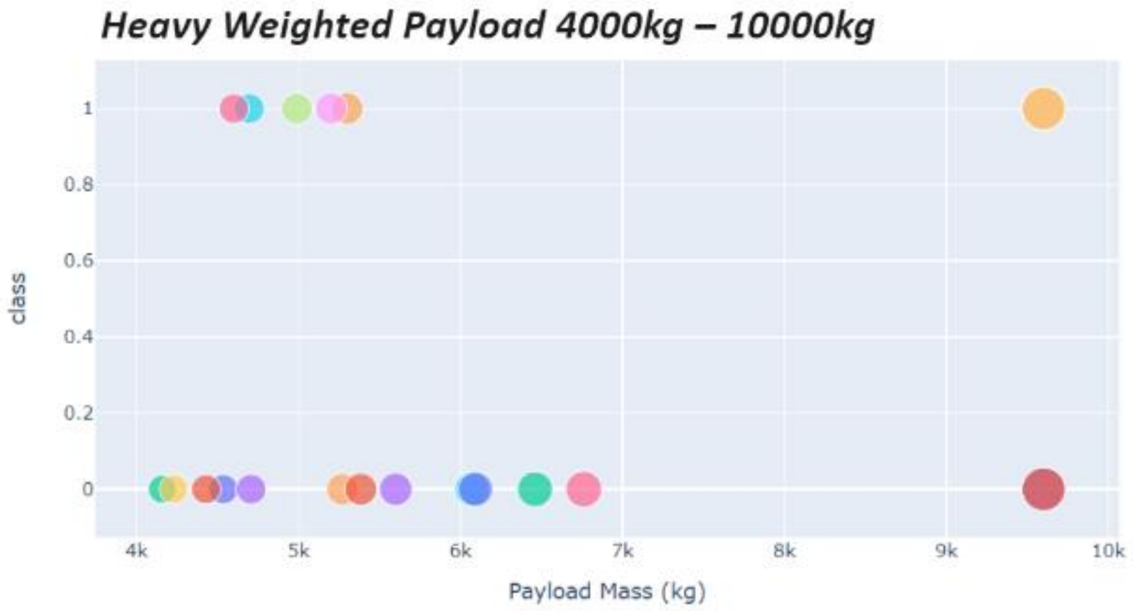
# Pie chart showing the Launch site with the highest launch success ratio




***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




*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 highest classification accuracy

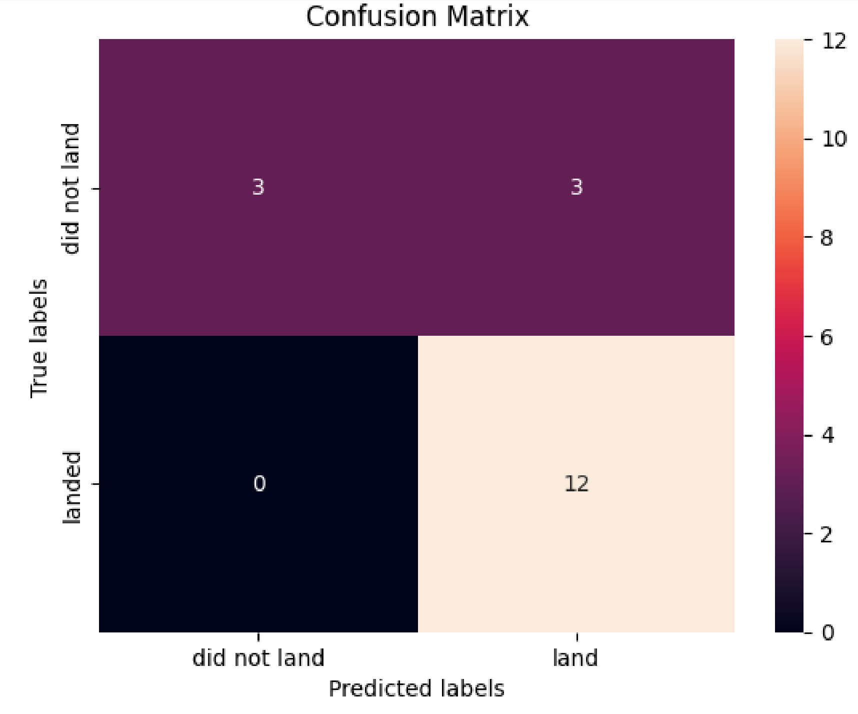
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

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 for the decision tree 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 Decision tree classifier is the best machine learning algorithm for this task.

Thank you!