Winning Space Race with Data Science

Alexander Hoffmann
19.01.2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection with API and Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Visualization
  - Interactive Visual Analytics with Folium and Dash
  - Machine Learning Prediction
- Summary of all results
  - EDA Results
  - Interactive Analytics Screenshots
  - Predictive Analysis Results

Introduction

- Project background and context

SpaceX is a highly successful company in the commercial space age and wants to make space travel affordable. SpaceX 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. Based on public information and machine learning we will predict if SpaceX will reuse the first stage.

- Problems you want to find answers

Identifying factors that influence the landing outcome.

Relationship between each variable and how it is affecting the outcome.

What conditions are best to ensure a successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Web Scrapping form Wikipedia and SpaceX API
- Perform data wrangling
  - Using one-hot encoding for categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

The Data was collected through web scrapping from Wikipedia and the SpaceX REST API.

With the API you can use a get request. After that the response content is decoded as Json and turned into a pandas dataframe using json_normalize(). Then the data is cleaned, checked for missing values and filled in with an appropriate value.

For web scrapping we use the BeatifulSoup library to extract the launch records as HTML table, then parse the table and convert it also to a pandas dataframe for further analysis.

Data Collection – SpaceX API

- Get request for rocket launch with API
- Json_normalize to convert from json to dataframe
- Data cleaning and added missing values
- https://github.com/AlexH1998/IBM/blob/main/Capstone%20Data%20Collection.ipynb

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

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

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

```
# Lets take a subset of our dataframe keeping only the features we want 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
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

- Request Wikipedia Page for Falcon9 Launch
- Create BeautifulSoup from HTML
- Extract all column names from HTML header

https://github.com/AlexH1998/IBM/blob/main/Capstone%20Web%20Scraping.ipynb

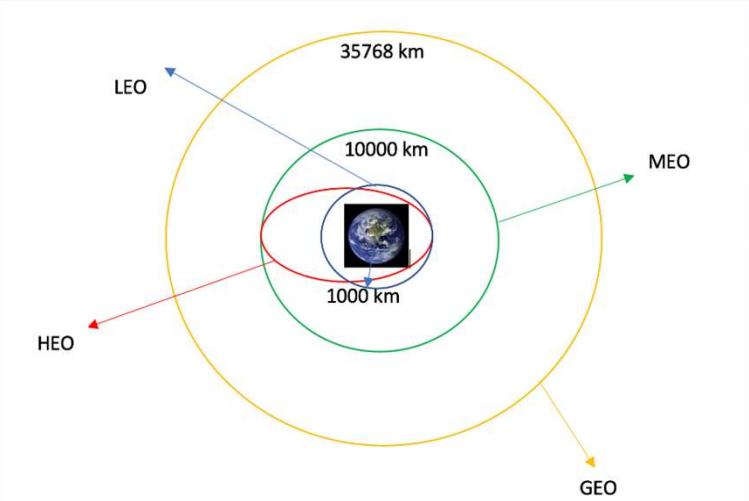
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
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
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`

            launch_dict['Flight No.'].append(flight_number)
            datatimelist=date_time(row[0])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`

            date = datatimelist[0].strip(',')
            launch_dict['Date'].append(date)


            # Time value
            # TODO: Append the time into launch_dict with key `Time`
            time = datatimelist[1]
            launch_dict['Time'].append(time)
```

Data Wrangling

- Calculate the number of launches on each site, then the number and occurrence of mission outcome per orbit type.
- Then we create a landing outcome label from the outcome column. This will make it easier for further analysis, visualization and machine learning. The results will be exported to a csv file.



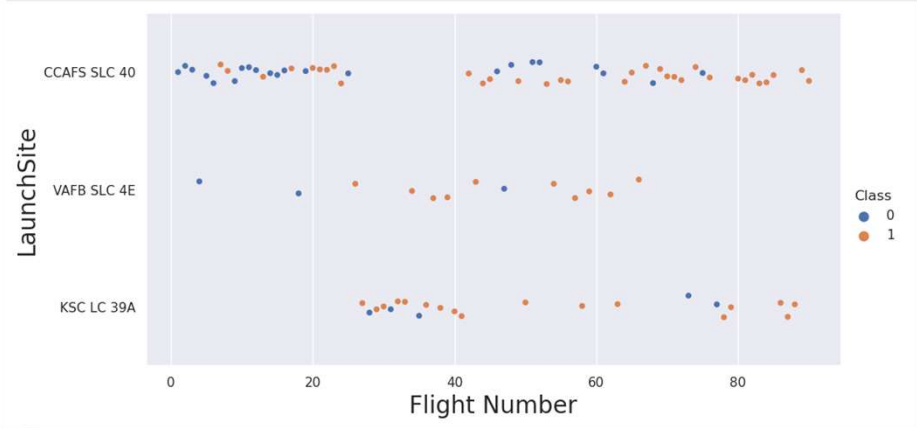
https://github.com/AlexH1998/IBM/blob/main/Capstone%20Data%20Wrangling.ipynb

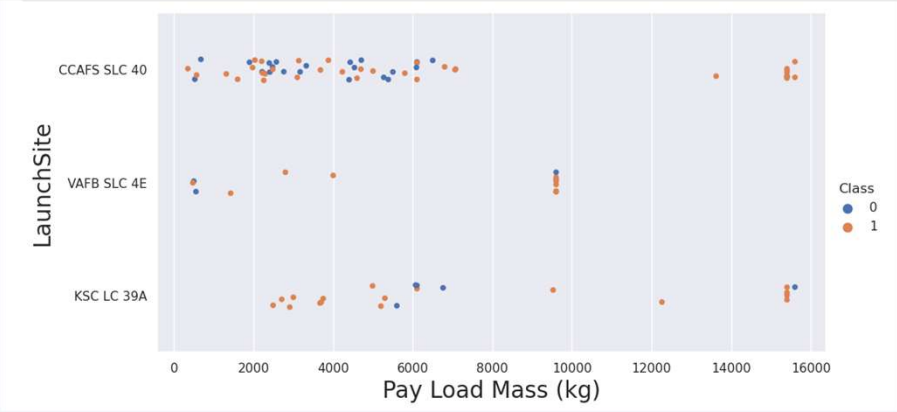
EDA with Data Visualization

- Create scatter graph to find the relationship between different attributes:
  - Launch Site and Flight Number
  - Launch Site and Payload
  - Etc.

  After that we can use further visualization like bar graphs and line graphs to show patterns of attributes.

https://github.com/AlexH1998/IBM/blob/main/Capstone%20EDA%20Visualization.ipynb





EDA with SQL

- We used SQL to perform queries on the dataset:
  - Display name of launch sites
  - Display 5 records where launch sites begin with ‘CCA’
  - Display total payload mass carried by booster launched with NASA
  - Display average payload mass carried by booster version F9 v1.1.
  - And more..
- https://github.com/AlexH1998/IBM/blob/main/Capstone%20EDA%20(1).ipynb

Build an Interactive Map with Folium

- To visualize the launch data we build and interactive map with folium. We added circle markers around each launch site with the label of the name of the launch site.
- They were also assigned to classes 0 and 1 with red and green to indicate whether it failed or succeeded.
- Also we calculated the distance to various landmarks on the map.
- https://github.com/AlexH1998/IBM/blob/main/Capstone%20Folium.ipynb

Build a Dashboard with Plotly Dash

- We build an interactive dashboard with plotly dash which allowed us to play around with the data
- We also plotted piecharts to show the total launches by a certain site or all sites
- Then we plotted a scatter graph showing the relationship with outcome and payload mass kg for the different booster versions.
- https://github.com/AlexH1998/IBM/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

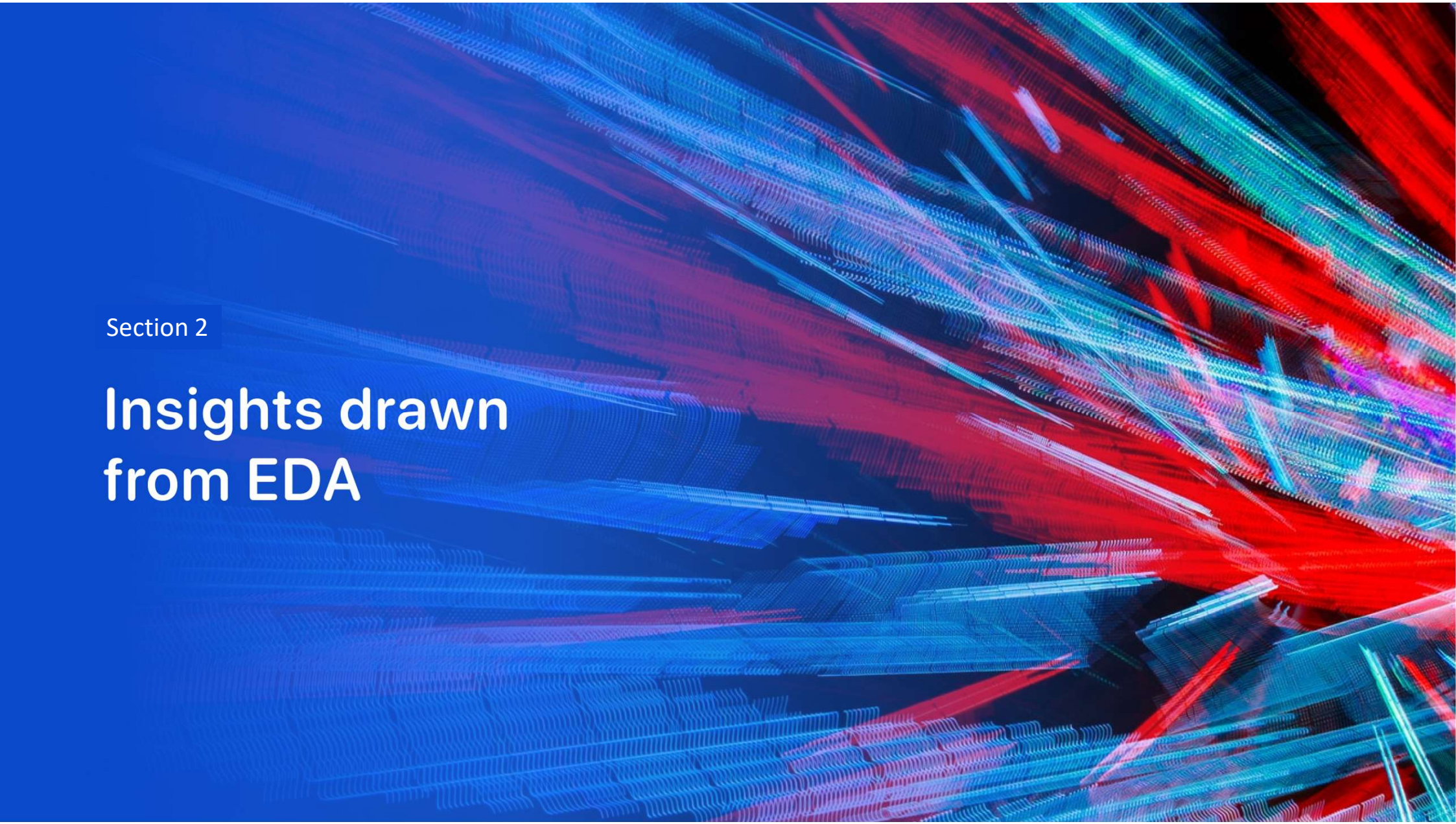
- First we loaded the dataset with numpy and pandas.
- We then transformed the data and split them in train and test set
- Apply a machine learning model on that data
- Set the parameters and algorithms to GridSearchCV and fit it to the dataset for best results
- We checked the accuracy of each model and then decided to use the one with the highest accuracy
- https://github.com/AlexH1998/IBM/blob/main/Capstone%20Machine%20Learning.ipynb

Results

Exploratory data analysis results

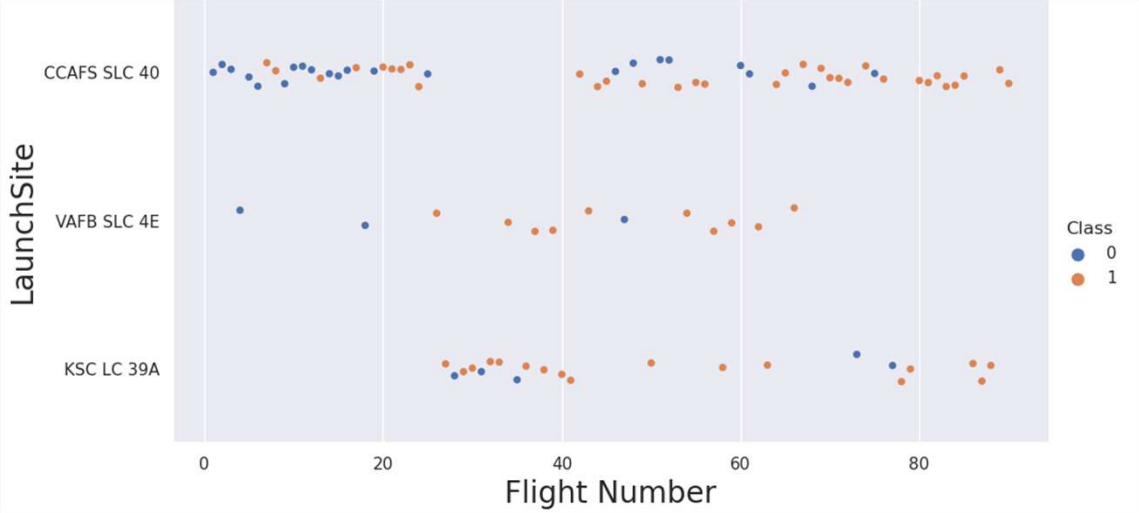
Interactive analytics demo in screenshots

Predictive analysis results


Section 2
Insights drawn
from EDA

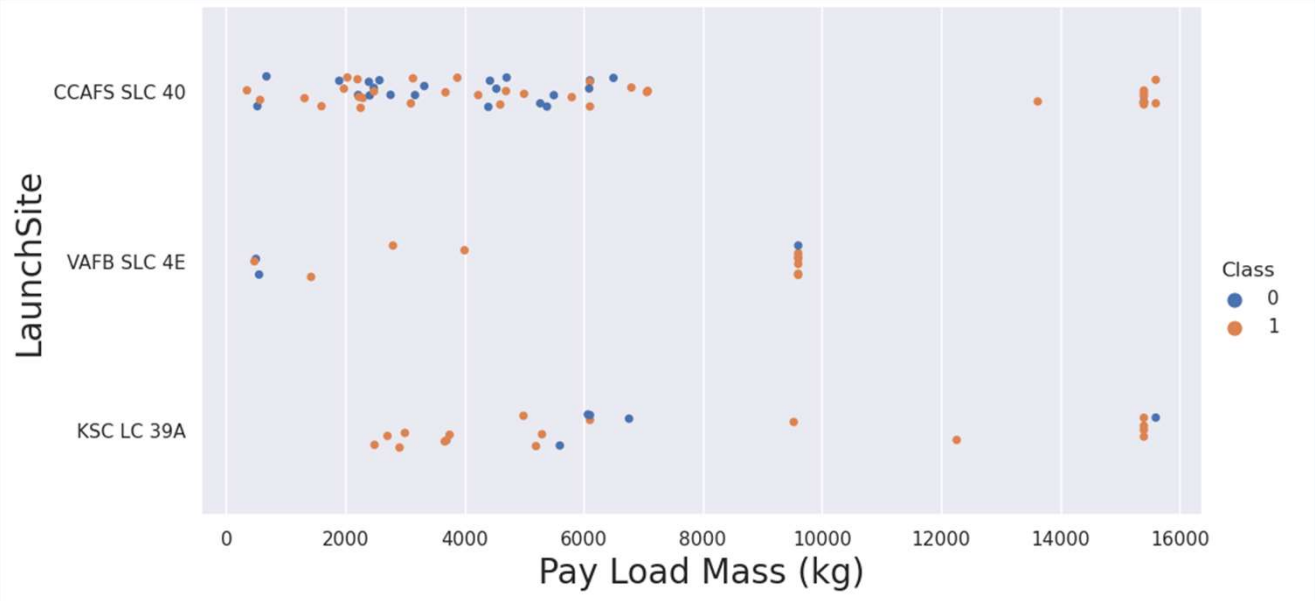
Flight Number vs. Launch Site

- The larger the flight amounts of the launch site, the greater the success rate will be



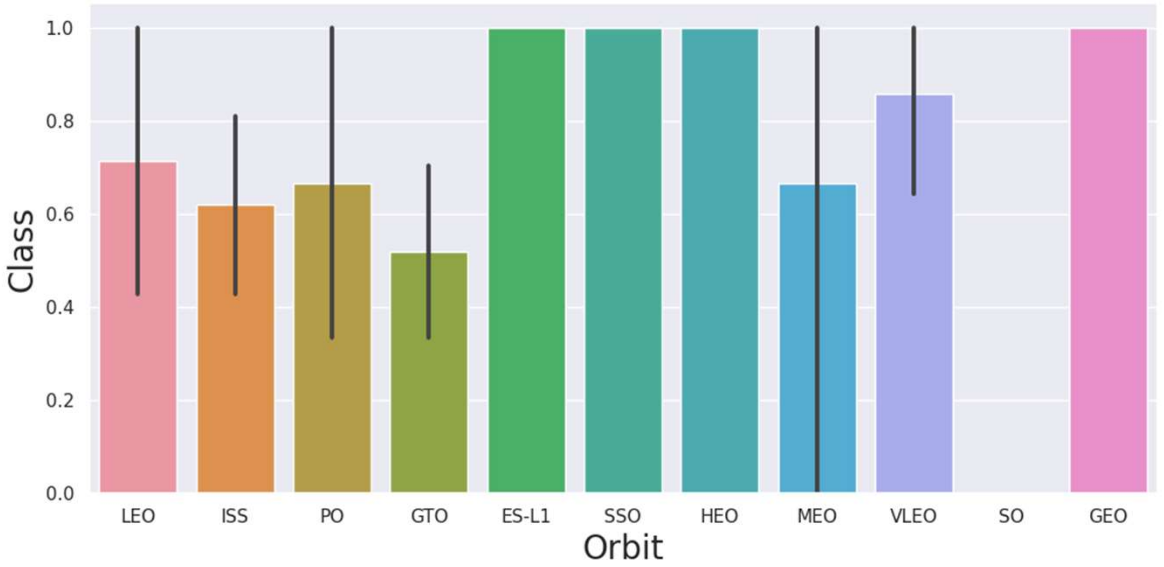
Payload vs. Launch Site

- Once the payload is higher than 7000kg, the success rate will increase.



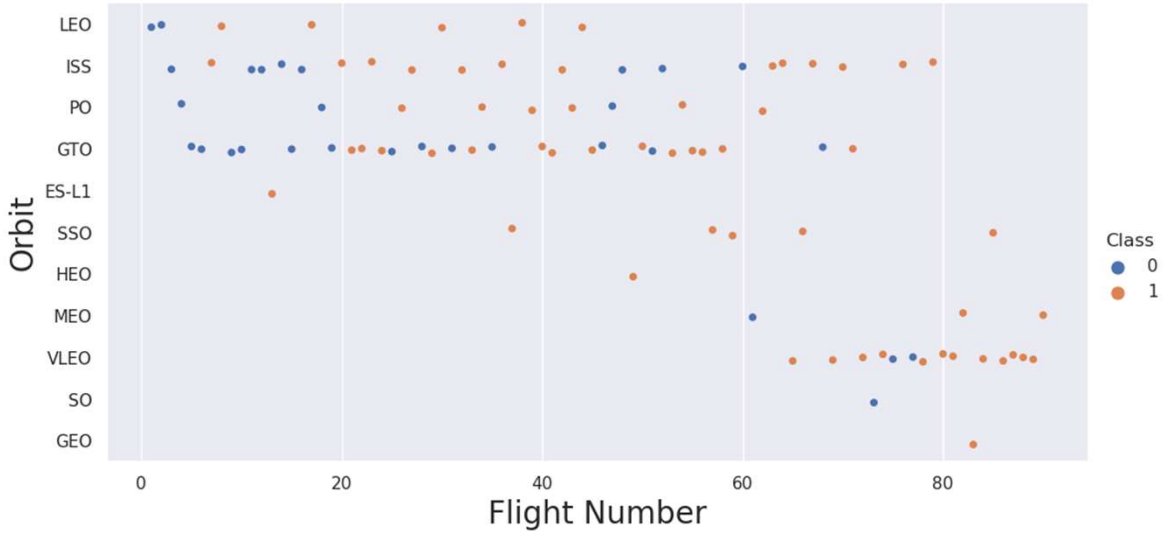
Success Rate vs. Orbit Type

- Some orbits have a higher success rate than others. However some have only very small data available.



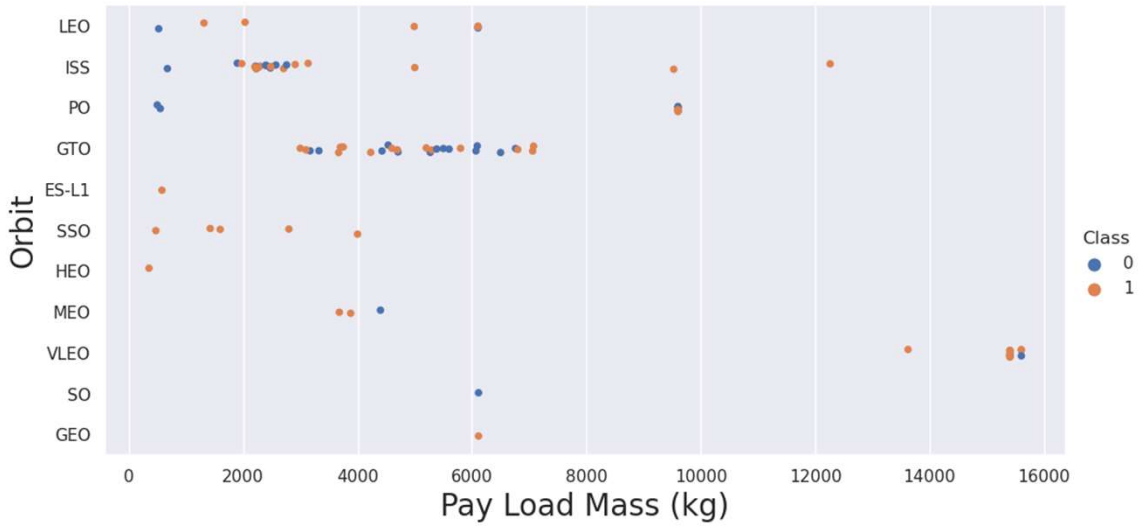
Flight Number vs. Orbit Type

- The larger the flight number on each orbit, the greater the success rate



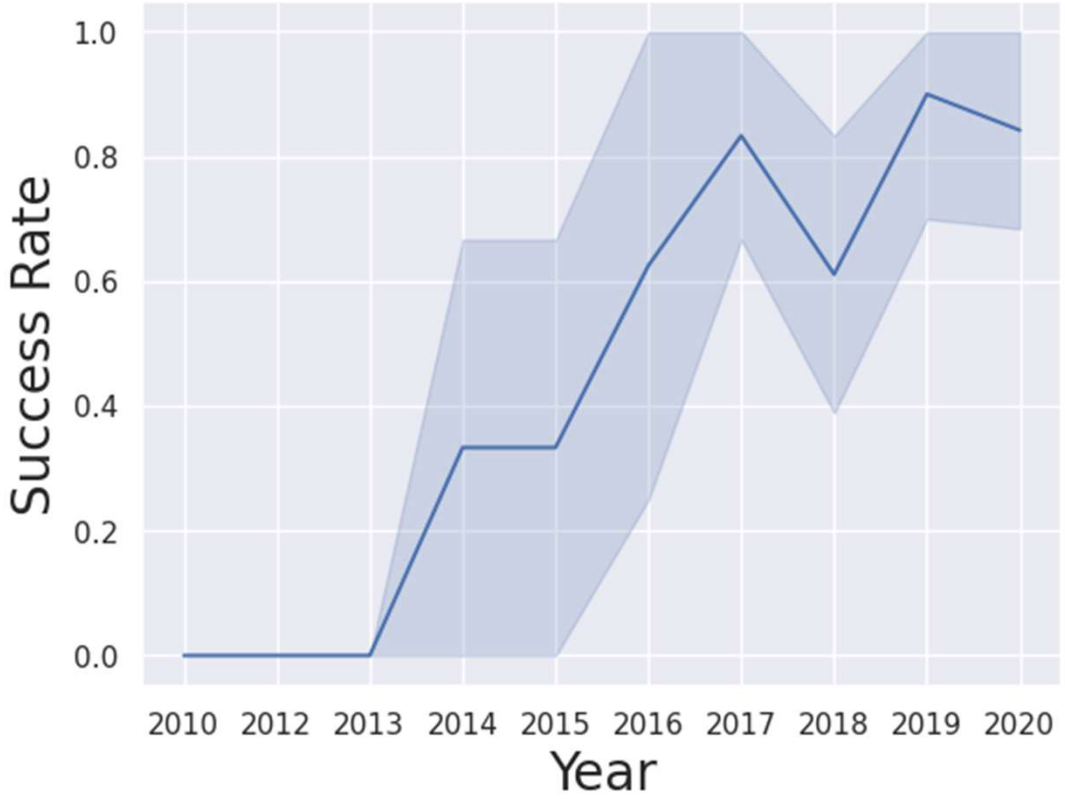
Payload vs. Orbit Type

- Heavier payload has a positive impact on some orbits, a negative impact on some others. Again small dataset for some orbits to make a conclusion.



Launch Success Yearly Trend

- Strong increase from 2013 to 2017. Then one bad year in 2017-2018 with increase again afterwards.



All Launch Site Names

- Distinct to only show unique launch sites from the spacex dataset

```
In [8]: %sql select distinct launch_site as "Launch_Sites" from spacex

 * ibm_db_sa://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2
Done.
```

Out[8]:

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

Launch Site Names Begin with 'CCA'

- Filter for launch sites beginning with CCA and limit the output to 5

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

In [9]:
```
%sql select * from spacex where launch_site like 'CCA%' limit 5
```

```
 * ibm_db_sa://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

Out[9]:

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

- Filter for customer = NASA and take the sum of payload mass

Task 3

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

In [13]:
```
%sql select sum(payload_mass__kg_) as "Total payload mass carried by booster launched by NASA(CRS)" from spacex where customer = 'NASA (CRS)';
```

```
 * ibm_db_sa://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

Out[13]:

Total payload mass carried by booster launched by NASA(CRS)
45596

Average Payload Mass by F9 v1.1

- Filter for the right booster version and take the avg of the payload mass

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

In [14]:
```
%sql select avg(payload_mass__kg_) as "Average payload mass carried by booster version F9 v1.1" from spacex where booster_version = 'F9 v1.1'
```

```
 * ibm_db_sa://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

Out[14]:

Average payload mass carried by booster version F9 v1.1
2928

First Successful Ground Landing Date

- Take the lowest data where the landing outcome was successful

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

Hint:Use min function

```
%sql select min(date) as "First successful landing outcome in ground pad" from spacex where landing__outcome = 'Success (ground pad)'
```

```
 * ibm_db_sa://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

First successful landing outcome in ground pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Taking the booster version, filter for successful landings and only select payload between 4000 and 6000

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

In [18]:
```
%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://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

Out[18]:

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

Total Number of Successful and Failure Mission Outcomes

- Count both the successful and failed mission outcomes

List the total number of successful and failure mission outcomes

In [20]:
```
%sql select count(mission_outcome) as "Total Missions" from spacex where mission_outcome like 'Success%' or mission_outcome like 'Failure%'
```

```
 * ibm_db_sa://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

Out[20]:

Total Missions
101

Boosters Carried Maximum Payload

- Using distinct values for booster version with subquery where the payload masses are the highest

Task 8

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

In [21]:
```
%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://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

Out[21]:

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

- Use booster version and launch site where the data starts with 2015 and the outcome is failure

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

In [23]:
```
%sql select booster_version, launch_site from spacex where date like '2015-%' and landing__outcome = 'Failure (drone ship)'
```

```
 * ibm_db_sa://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

Out[23]:

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

- Grouping the landing outcomes and counting how many in total there are for every outcome

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

In [26]:
```
%sql select landing__outcome as "Landing outcome", count(landing__outcome) as "Total" 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://jqg79642:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

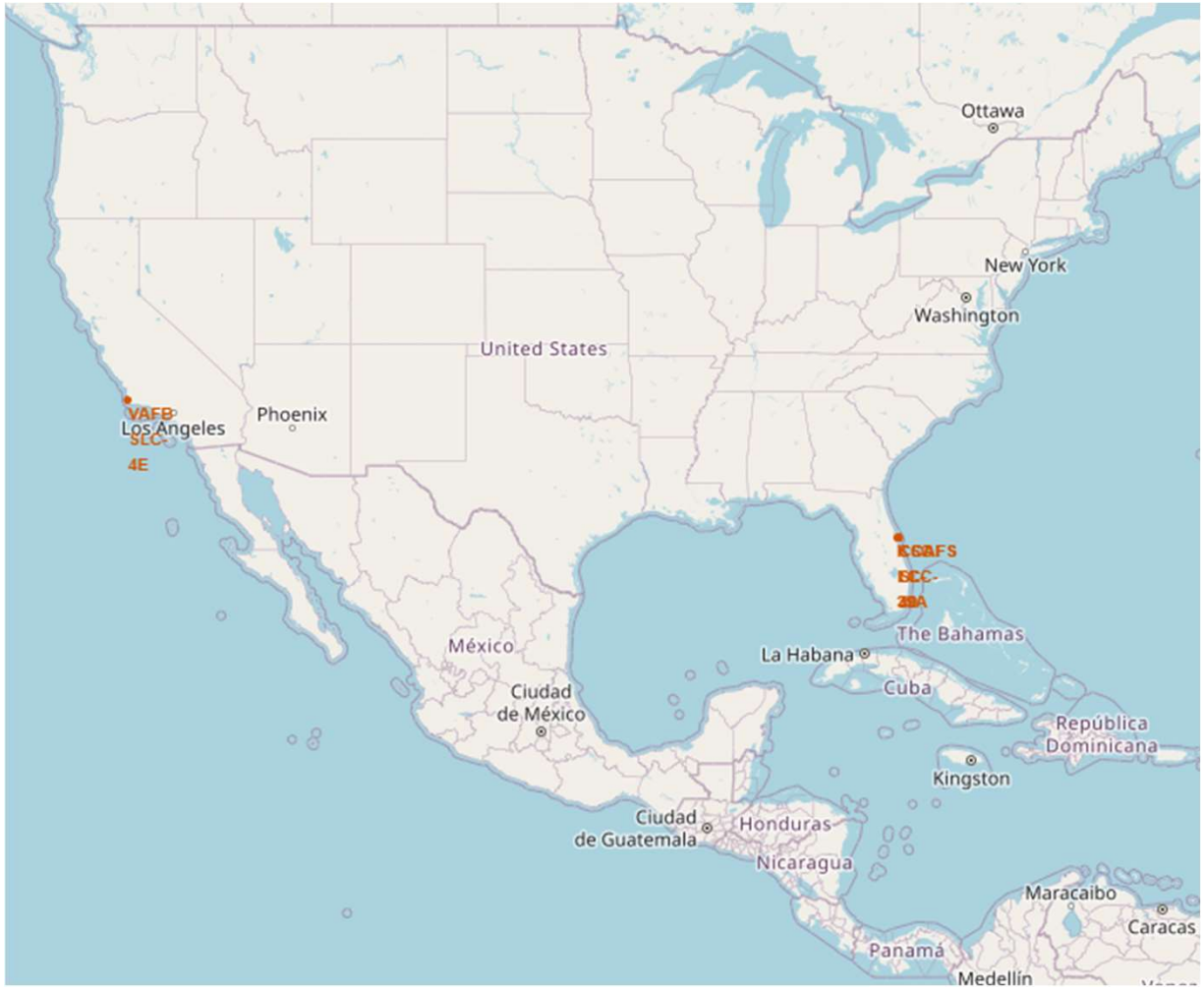
Out[26]:

Landing outcome	Total
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

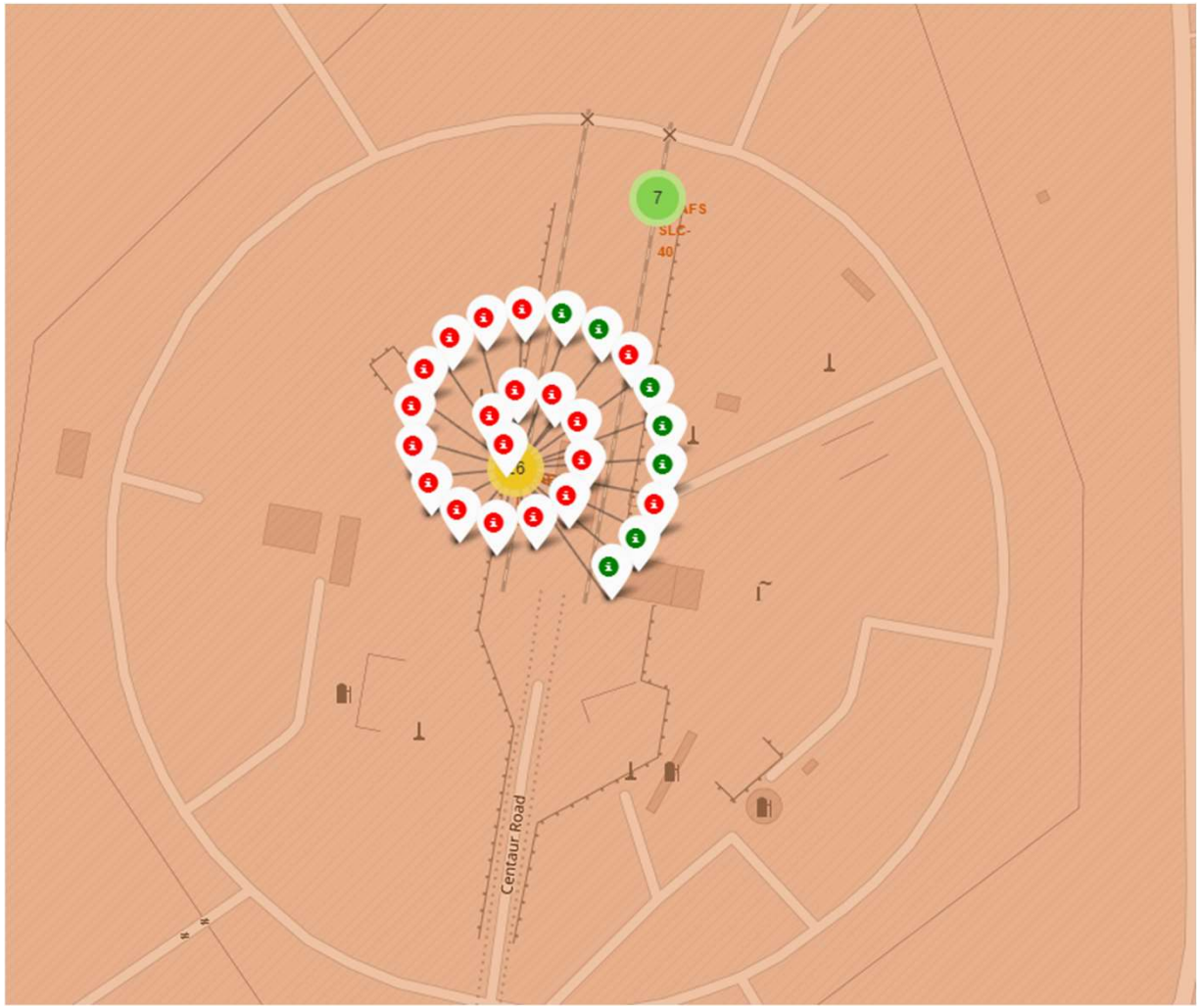
Location of all Launch Sites

- All launch Sites are located in the USA



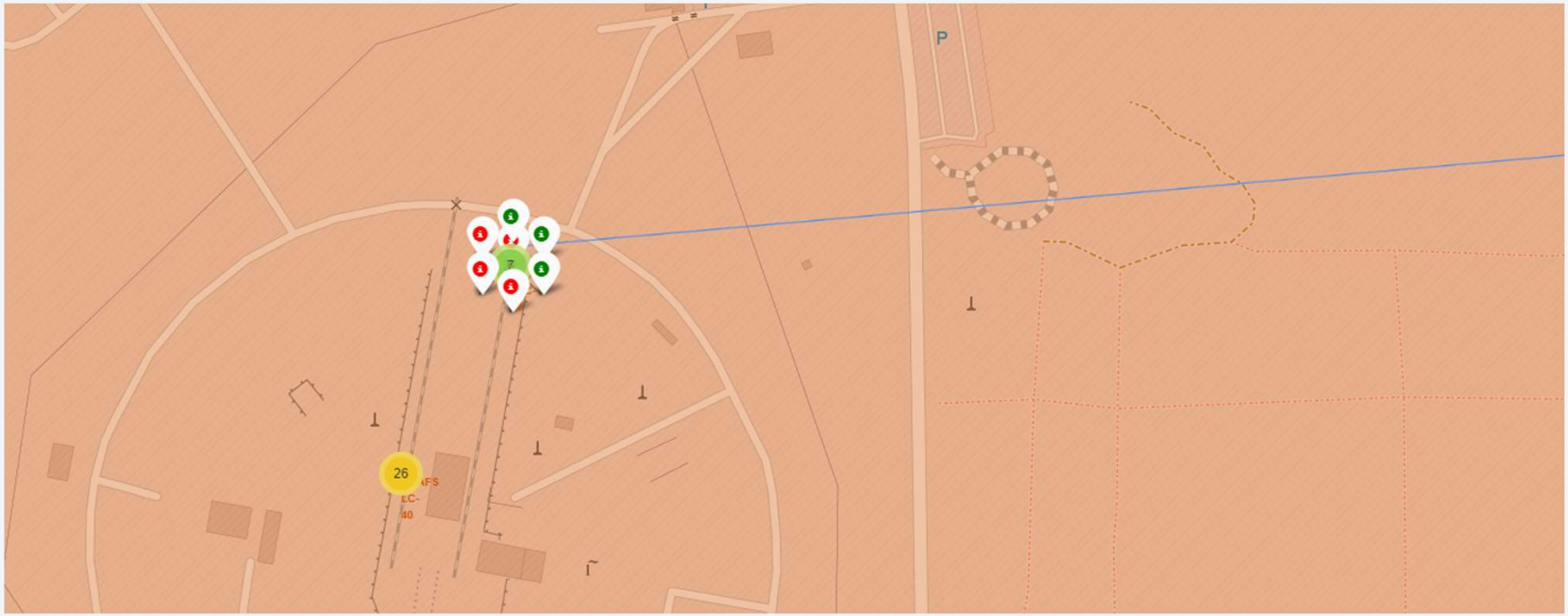
Launch Site Markers with color labels

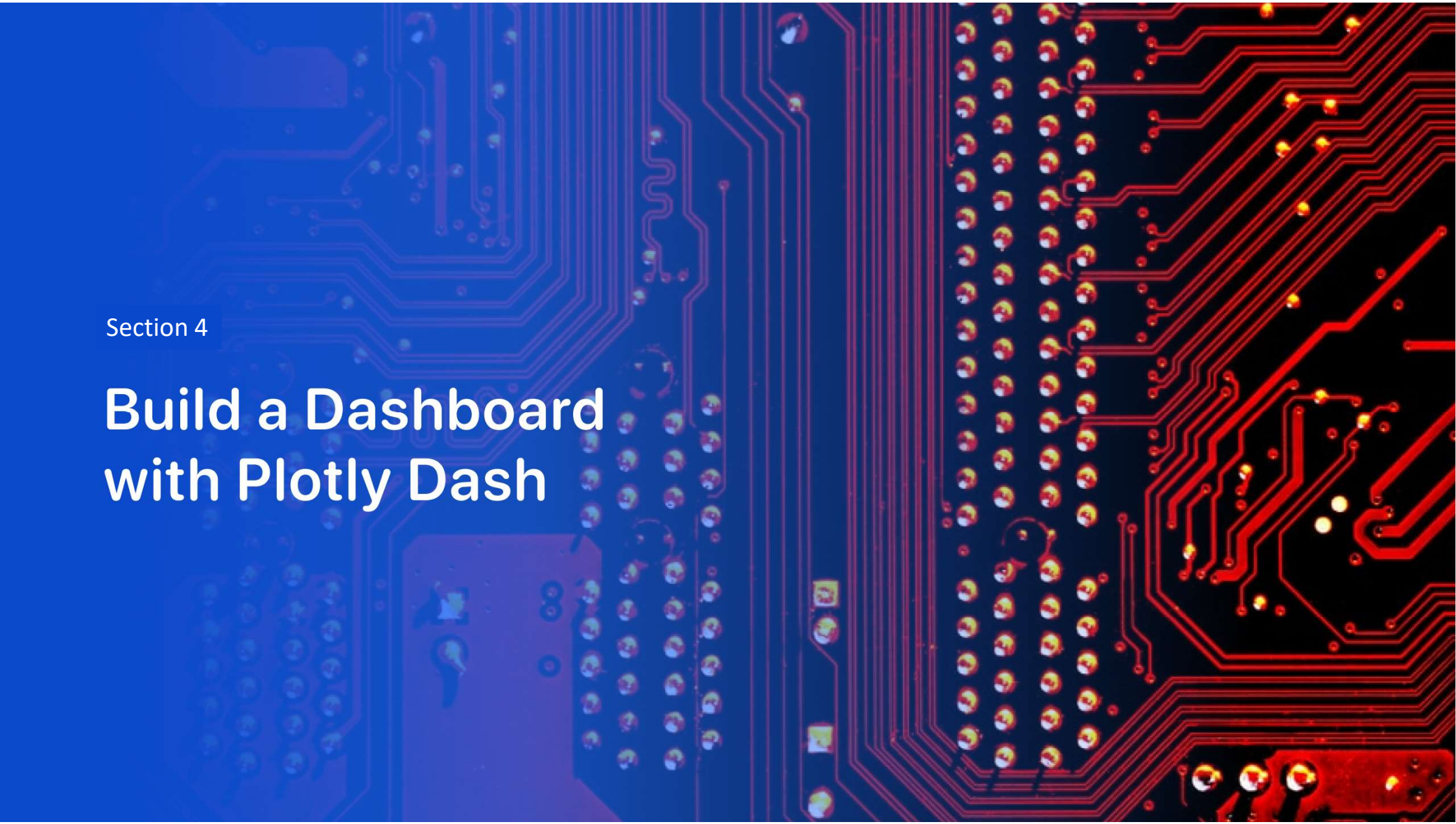
- Red indicates failure and green successful launches



<Folium Map Screenshot 3>

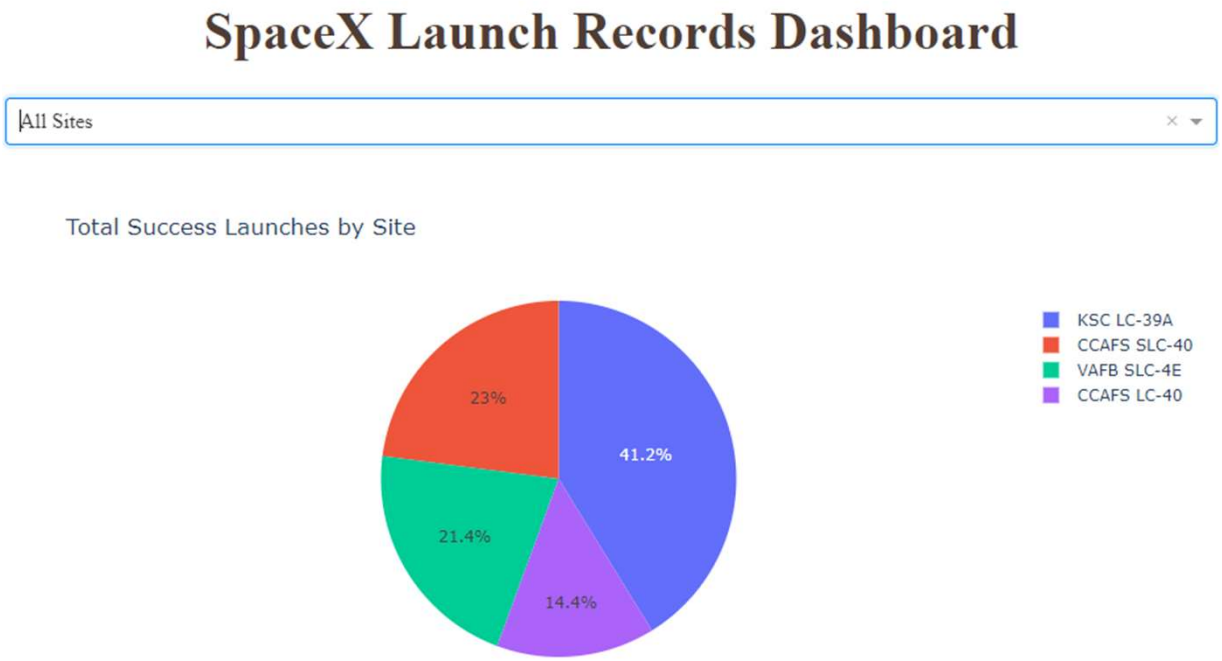
- Distance from this launch Site to the ocean




Section 4
Build a Dashboard with Plotly Dash

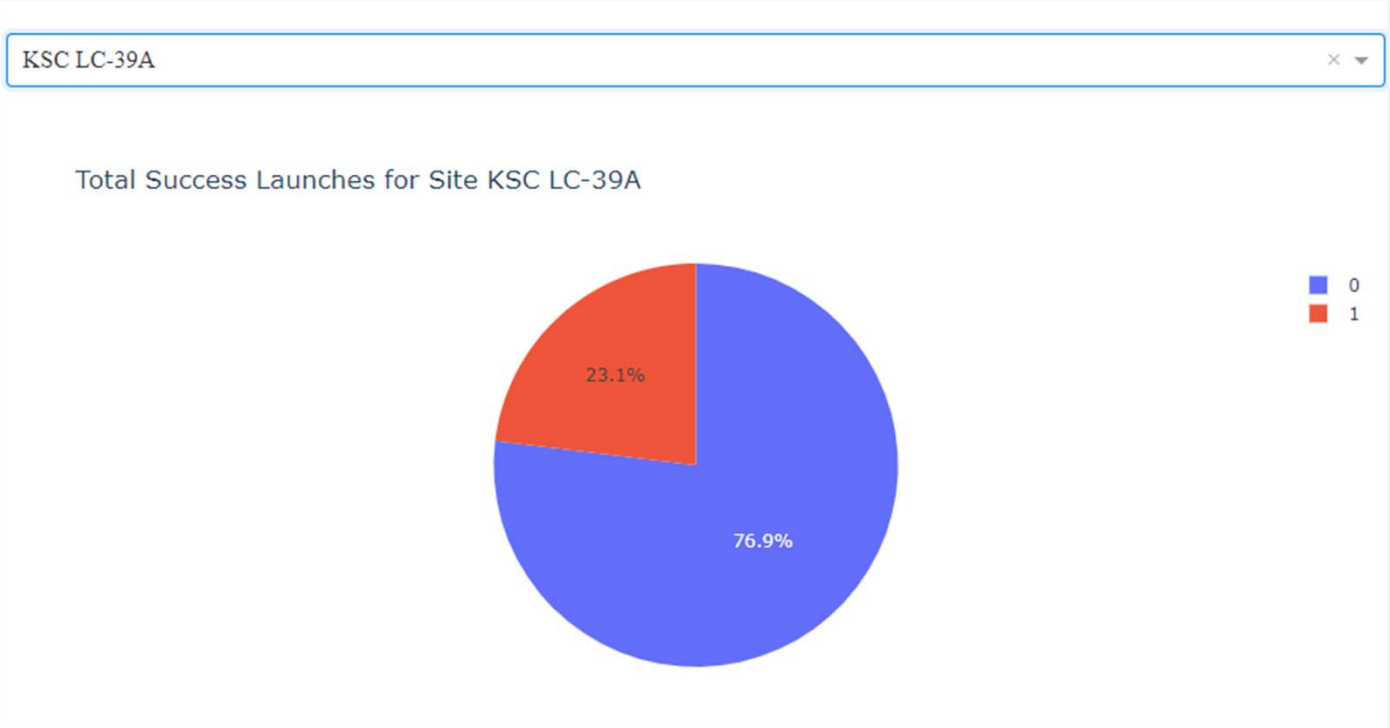
Success percentage for all sites

- KSC LC-39A has the most successful launches from all the sites



Highest launch success ratio

KSC LC-39A had 76.9% of launched succeed



Payload vs Launch Outcome Scatter Plot

- The success rate for low weighted payload is higher than heave weighted payload

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The model with the highest accuracy was in my case the SVM with more than 84%

In [39]:
```
parameters = {'kernel':('linear', 'rbf','poly','rbf', 'sigmoid'),
              'C': np.logspace(-3, 3, 5),
              'gamma':np.logspace(-3, 3, 5)}
svm = SVC()
```

In [40]:
```
svm_cv = GridSearchCV(svm,parameters,cv=10)
svm_cv.fit(X_train,Y_train)
```

Out[40]:
```
GridSearchCV(cv=10, estimator=SVC(),
             param_grid={'C': array([1.00000000e-03, 3.16227766e-02, 1.00000000e+00, 3.16227766e+01,
       1.00000000e+03]),
                         'gamma': array([1.00000000e-03, 3.16227766e-02, 1.00000000e+00, 3.16227766e
       1.00000000e+03]),
                         'kernel': ('linear', 'rbf', 'poly', 'rbf', 'sigmoid')})
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

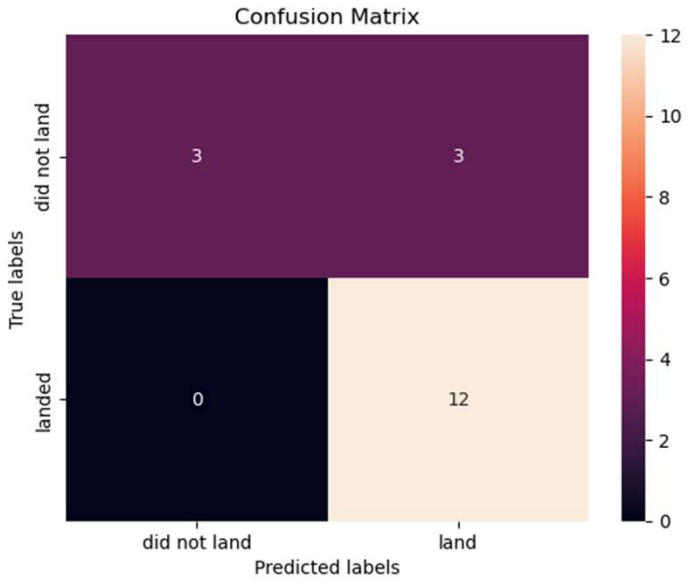
In [42]:
```
print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
print("accuracy :",svm_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

Confusion Matrix

- 3 landings were predicted as landed, however the didn't land (false positive). The rest is correct

```
In [44]: yhat=svm_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- The larger the flight amounts of the launch site, the greater the success rate will be
- Some orbits have a higher success rate than others. Point 3
- The larger the flight number on each orbit, the greater the success rate
- KSC LC-39A had 76.9% of launched succeed which is the most
- The success rate for low weighted payload is higher than heave weighted payload
- With machine learning we predict with a 84% chance the outcome of the landing

Thank you!