Winning Space Race with Data Science

Emilia Millan Zaragoza
January 13th, 2024

Project Link

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- **Summary of all results**
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result from Machine Learning Lab

Introduction

SpaceX, an innovative and space-oriented organization is currently offering rocket launches specifically Falcon 9 as low as 62 million dollars. In the space market we often find costs around 165 million dollar per launch. Low prices of SpaceX are due to the idea of reusing the first stage of the launch by re-land the rocket to be used on the next mission. Repeating this process can make the price go down even more.

As a data scientist of SpaceX, the goal of this project is to create the optimal machine learning pipeline to predict the landing outcome of the first stage in the future. This project is crucial in identifying the competitive low price to bid against SpaceX for a rocket launch.

The problems included:

- Identifying all critical factors that influence the landing outcome.
- The relationship between each variables and how it is affecting the outcome.
- The optimal condition required to increase probability of successful landing.

Section 1

Methodology

Methodology

- Executive Summary
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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 measuring information on targeted variables in an established system. This process enables to answer relevant questions and evaluate outcomes. As mentioned, the dataset was collected by REST API and Web Scrapping from Wikipedia
- For **REST API**, started by using the **get request**. Then, decoded the response content as **JSON** and convert into a pandas dataframe using **json_normalize()**.
- Next step: cleaned the data, checked for missing values and fill with whatever needed.
- For web scrapping, used **BeautifulSoup** to extract the launch records as **HTML** table, parse the table and convert it to a pandas dataframe for further data analysis.

Data Collection – SpaceX API

Get request for rocket launch data using API

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

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

Use **json_normalize** method to convert **json** result to **dataframe**

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

Performed data cleaning and filling the missing value

```
[13]: # 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
      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 replac
      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)]
```

[Notebook Data Collection API]

Data Collection - Scraping

Reuest the Falcon9 Launch Wiki page from url

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

Create a BeautifulSoup from the HTML response

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

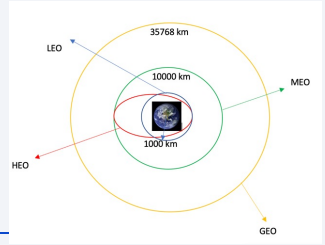
Extract all column/variable names from the HTML header

```
[15]: 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)
            print(flight_number)
            datatimelist=date_time(row[0])

            # Date value
```

[Notebook Data Collection - Scraping]

Data Wrangling



Data Wrangling is the process of cleaning and unifying messy and complex data sets for easy access and Exploratory Data Analysis (EDA).

First step is to calculate the number of launches on each site, then calculate the number and occurrence of mission outcome per orbit type.

Then we create a landing outcome label from the outcome column. This will facilitate further analysis, visualization, and ML. Lastly, we will export the result to a CSV.

```
In [13]: # Apply value_counts on Orbit column
         df['Orbit'].value_counts()

Out[13]: GTO      27
         ISS      21
         VLEO     14
         PO        9
         LEO       7
         SSO       5
         MEO       3
         ES-L1     1
         HEO       1
         SO        1
         GEO       1
         Name: Orbit, dtype: int64
```

```
In [16]: # landing_outcomes = values on Outcome column
         landing_outcomes = df['Outcome'].value_counts()

         landing_outcomes

Out[16]: True ASDS      41
         None None      19
         True RTLS      14
         False ASDS      6
         True Ocean      5
         False Ocean     2
         None ASDS       2
         False RTLS      1
         Name: Outcome, dtype: int64
```

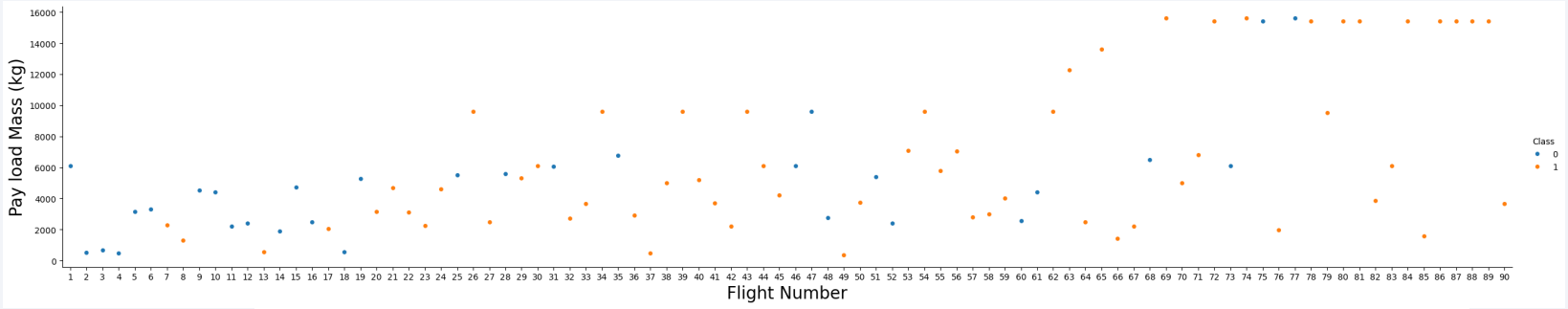
```
df.to_csv("dataset_part_2.csv", index=False)
```

Notebook Data Wrangling

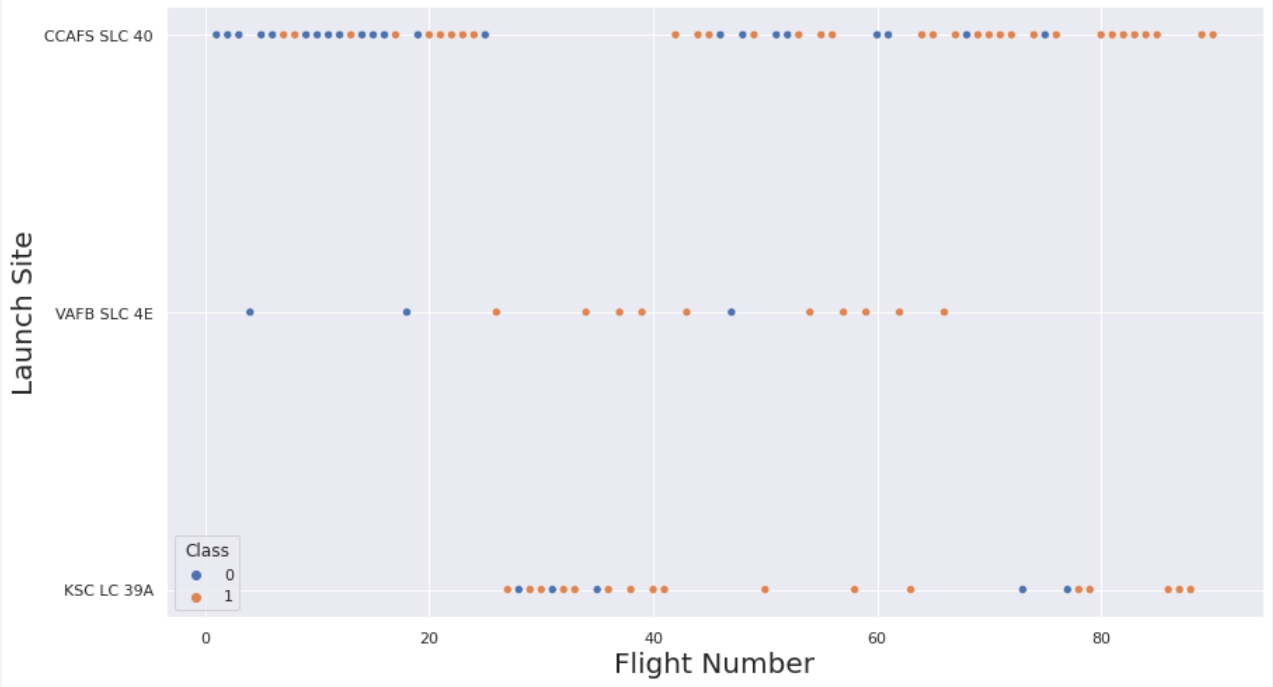
EDA with Data Visualization

We first started by using scatter graph to find the relationship between the attributes such as between:

Pay load and Flight Number.



Flight Number and Launch Site

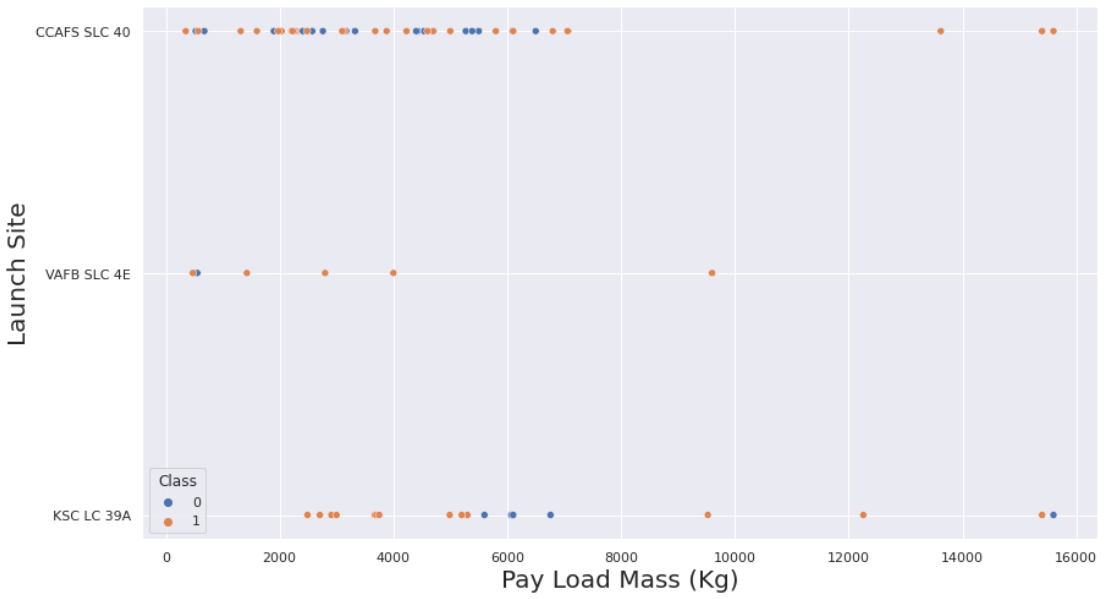


[Notebook Data Visualization]

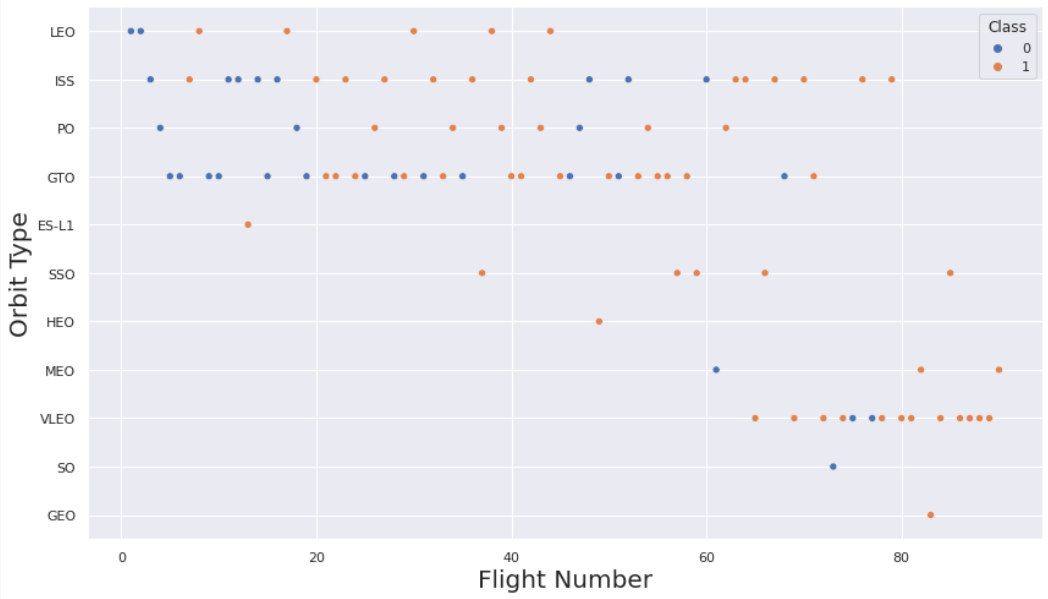
EDA with Data Visualization

We first started by using scatter graph to find the relationship between the attributes such as between:

Pay load and Launch Site



Flight Number and Orbit Type

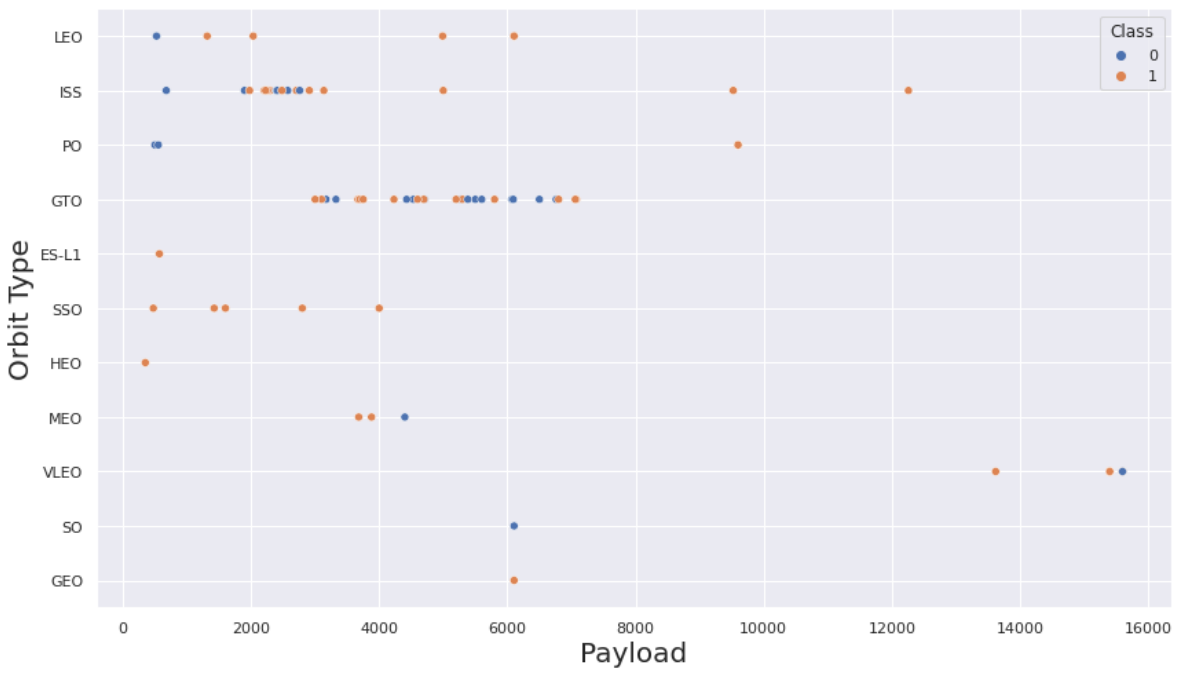


[Notebook Data Visualization]

EDA with Data Visualization

We first started by using scatter graph to find the relationship between the attributes such as between:

Payload and Orbit Type.



Scatter plots show dependency of attributes on each other.

Once a pattern is determined from the graphs. It's very easy to see which factors affecting the most to the success of the landing outcomes.

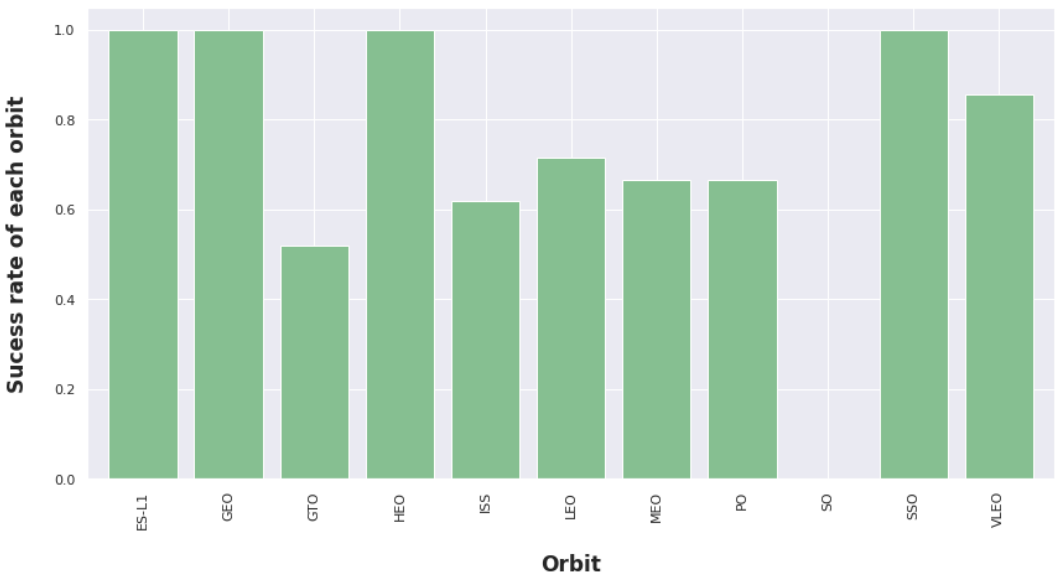
Notebook Data Visualization

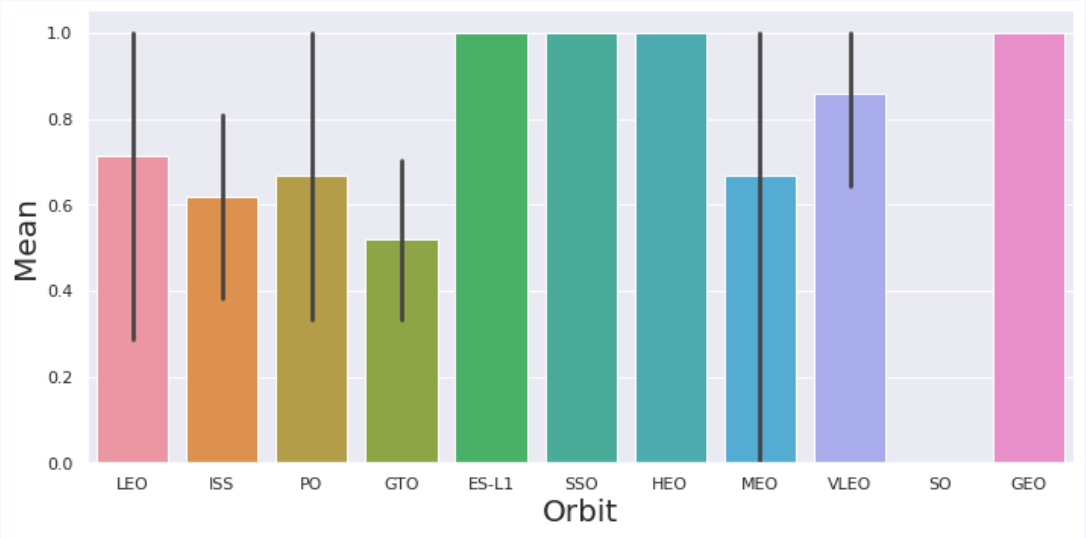
EDA with Data Visualization

For further visualization analysis we used tools such as bar graph and line plots graph

Bar graphs is one of the easiest way to interpret the relationship between the attributes. In this case, we will use the bar graph to determine which orbits have the highest probability of success.

We then use the line graph to show a trends or pattern of the attribute over time which in this case, is used for see the launch success yearly trend.

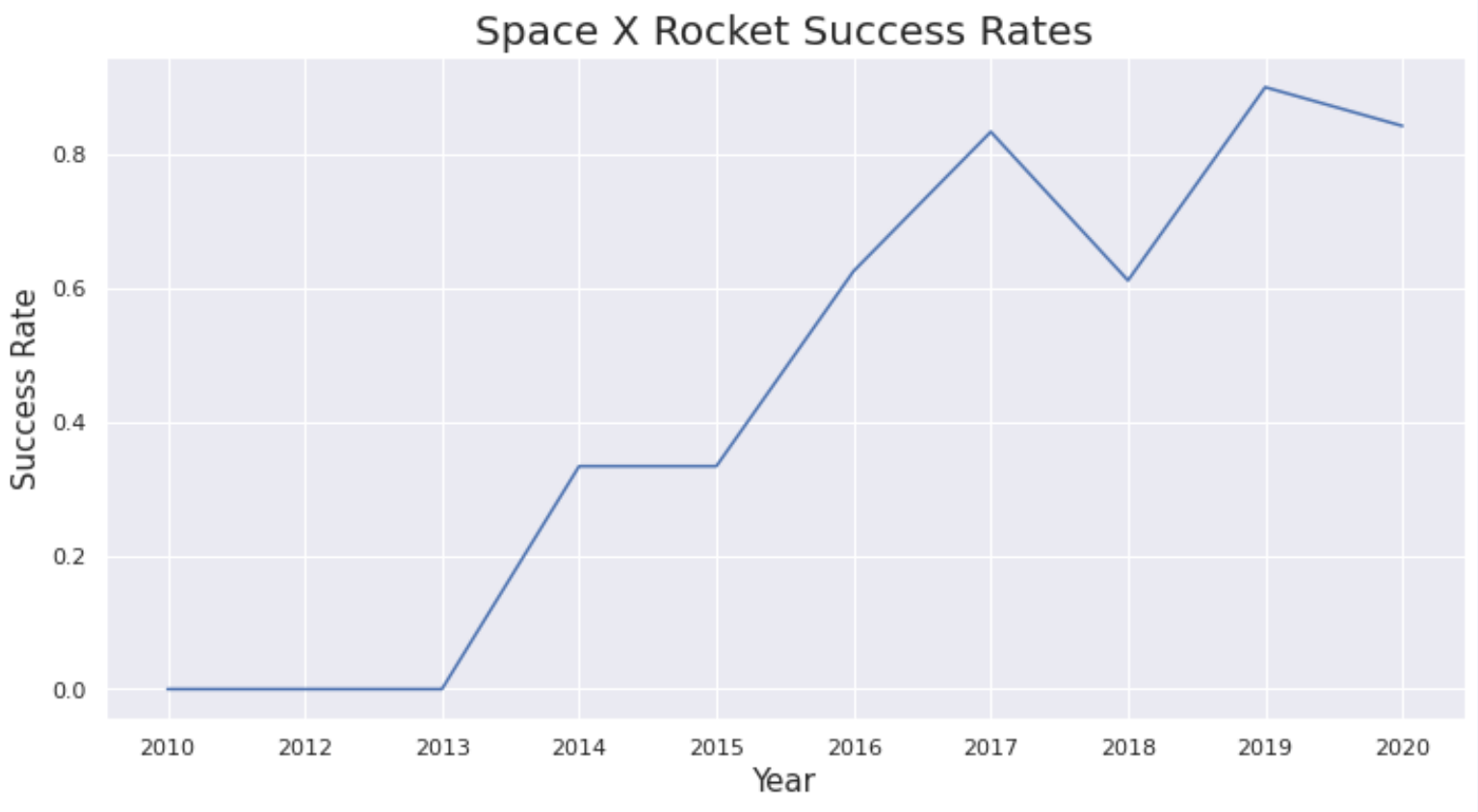




Notebook Data Visualization

EDA with Data Visualization

We then use Feature Engineering to be used in success prediction in the future module by created the dummy variables to categorical columns.



Notebook Data Visualization

EDA with SQL

Using SQL, we had performed many queries to get better understanding of the dataset, Ex:

- Displaying the names of the launch sites.
- Displaying 5 records where launch sites begin with the string 'CCA'.
- Displaying the total payload mass carried by booster launched by NASA (CRS).
- Displaying the average payload mass carried by booster version F9 v1.1.
- 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.
- Listing the total number of successful and failure mission outcomes.
- Listing the names of the booster_versions which have carried the maximum payload mass
- Listing the failed landing_outcomes in drone ship, their booster versions, and launch sites names for in year 2015.
- Rank the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20, in descending order.

Notebook EDA with SQL

Build an Interactive Map with Folium

To visualize the launch data into an interactive map. We took the latitude and longitude coordinates at each launch site and added a circle marker around each launch site with a label of the name of the launch site.

We then assigned the dataframe launch_outcomes(failure,success) to classes 0 and 1 with Red and Green markers on the map in MarkerCluster().

We then used the Haversine's formula to calculated the distance of the launch sites to various landmark to find answer to the questions of:

How close the launch sites with railways, highways and coastlines?

How close the launch sites with nearby cities?

[Notebook Map with Folium]

Build a Dashboard with Plotly Dash

We built an interactive dashboard with Plotly dash which allowing the user to play around with the data as they need.

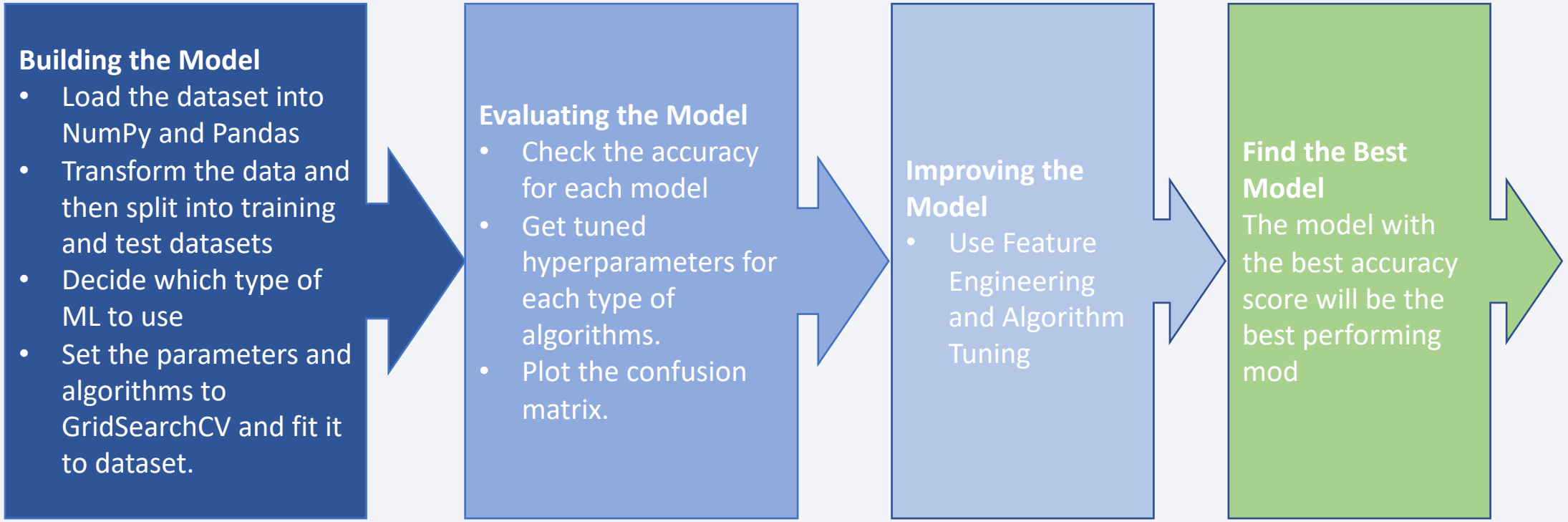
We plotted pie charts showing the total launches by a certain sites.

We then plotted scatter graph showing the relationship with Outcome and Payload

Mass (Kg) for the different booster version.

Notebook Plotly Dash

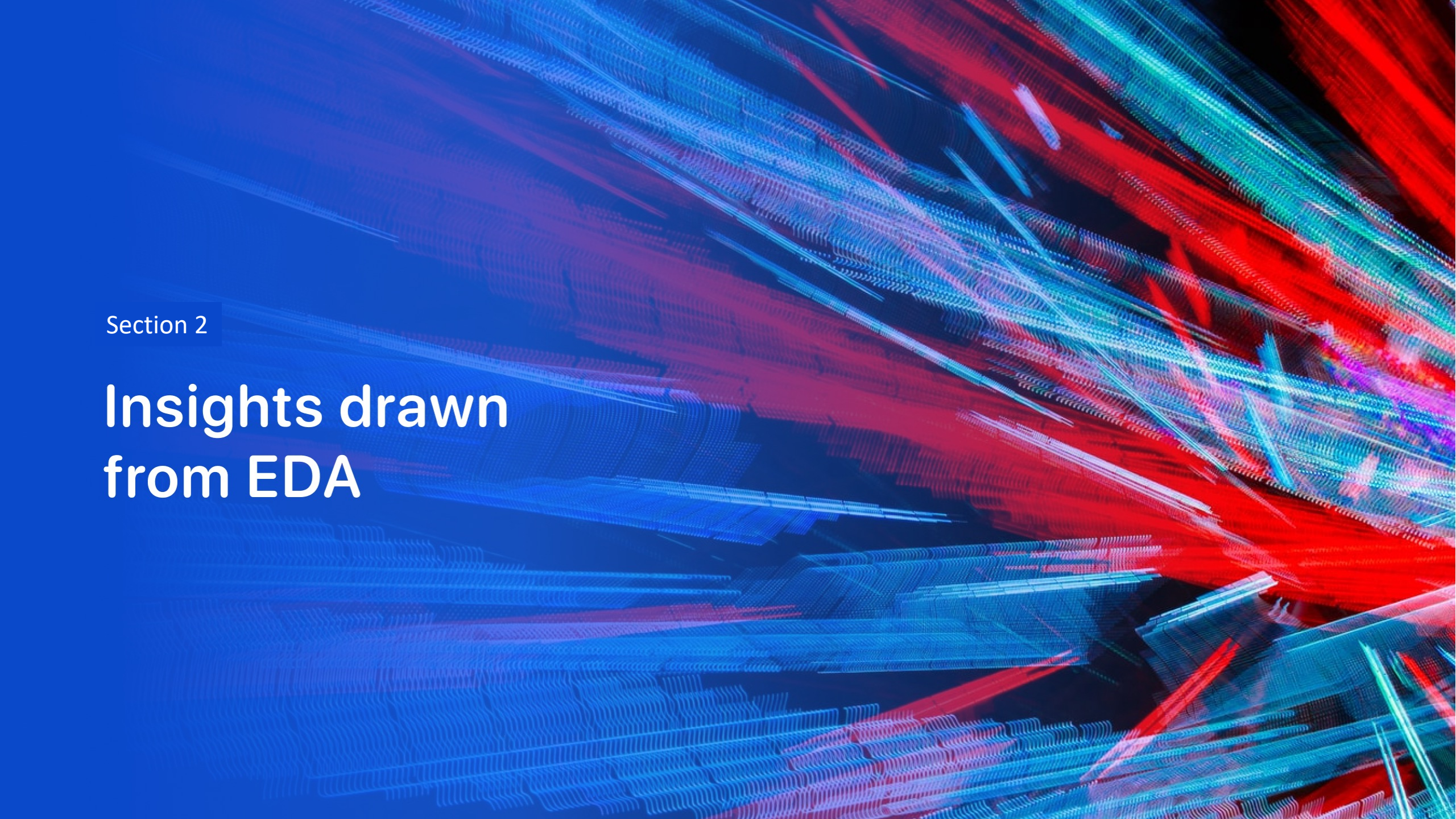
Predictive Analysis (Classification)



Notebook Plotly Dash

Results

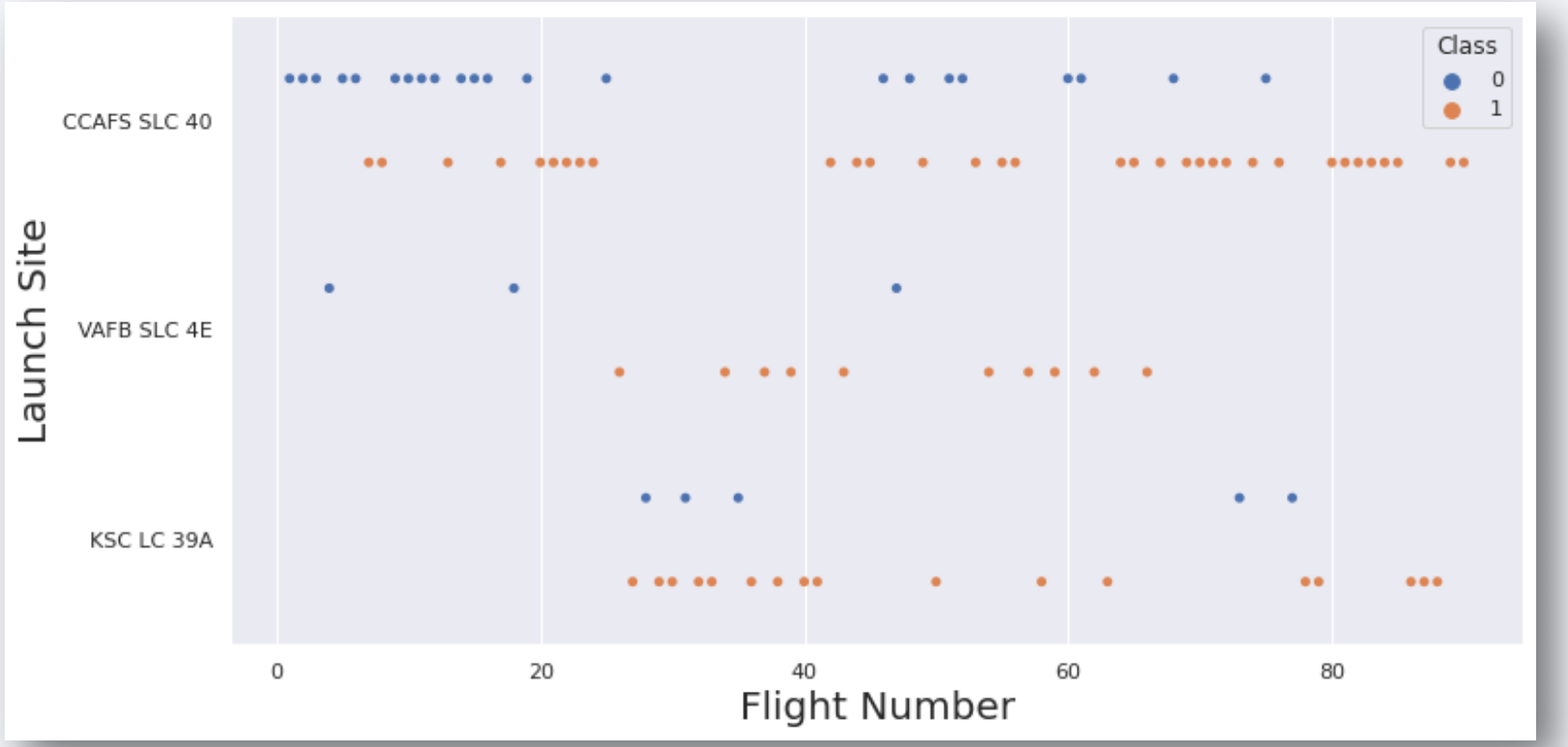
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

This scatter plot interpretation is that the larger the flights amount of the launch site, the greater the the success rate will be.

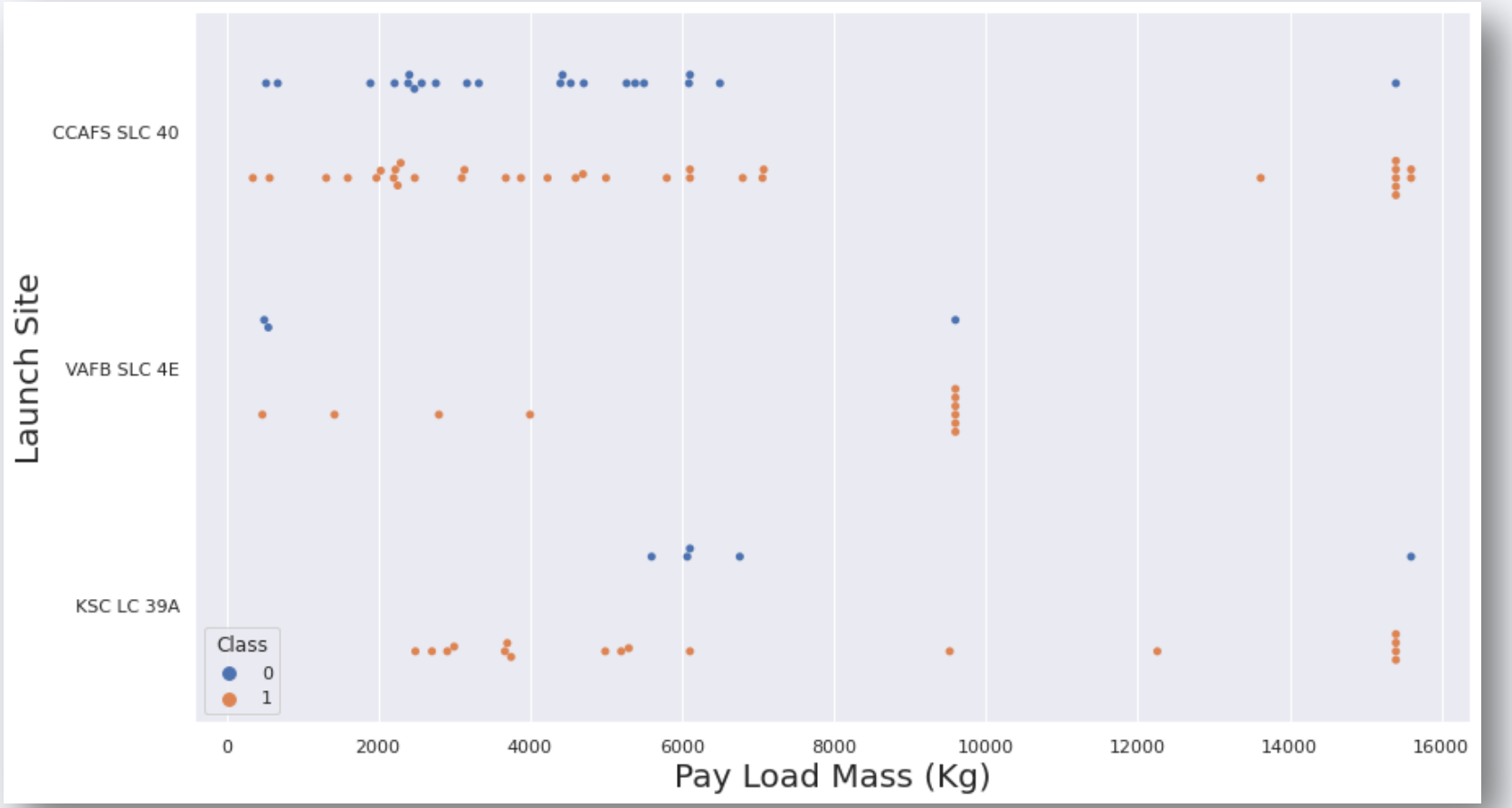
However, site **CCAFS SLC40** shows the least pattern of this rate



Payload vs. Launch Site

This scatter plot interpretation is that once the pay load mass goes greater than 7000kg, the probability of the success rate will be highly increased.

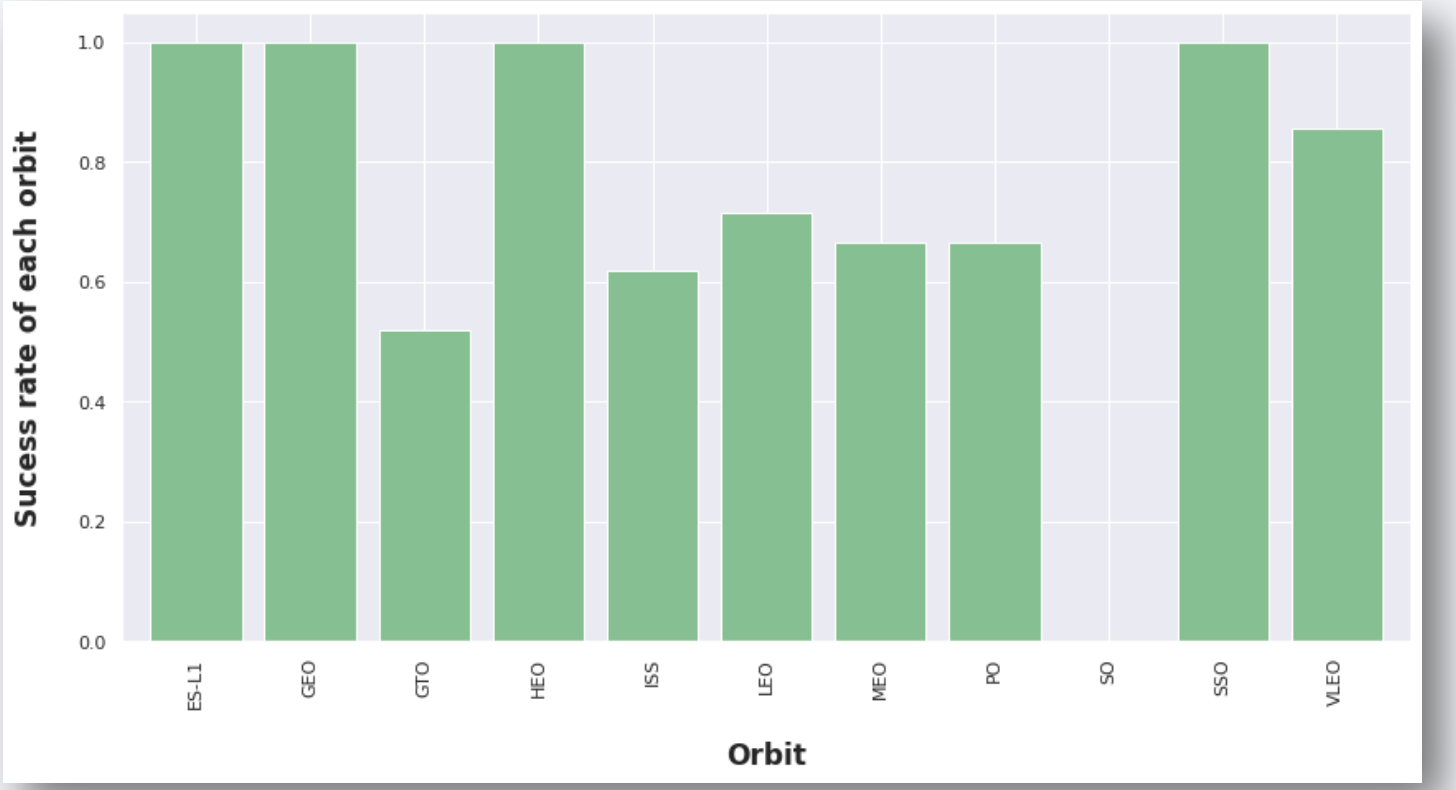
However, there is no clear pattern to say the launch site is dependent to the pay load mass for the success rate



Success Rate vs. Orbit Type

This bar graph compares the possibility of the orbits to influence the landing outcomes as some orbits has 100% success rate. Such as SSO, HEO, GEO AND ES-L1 while SO orbit produced 0% rate of success.

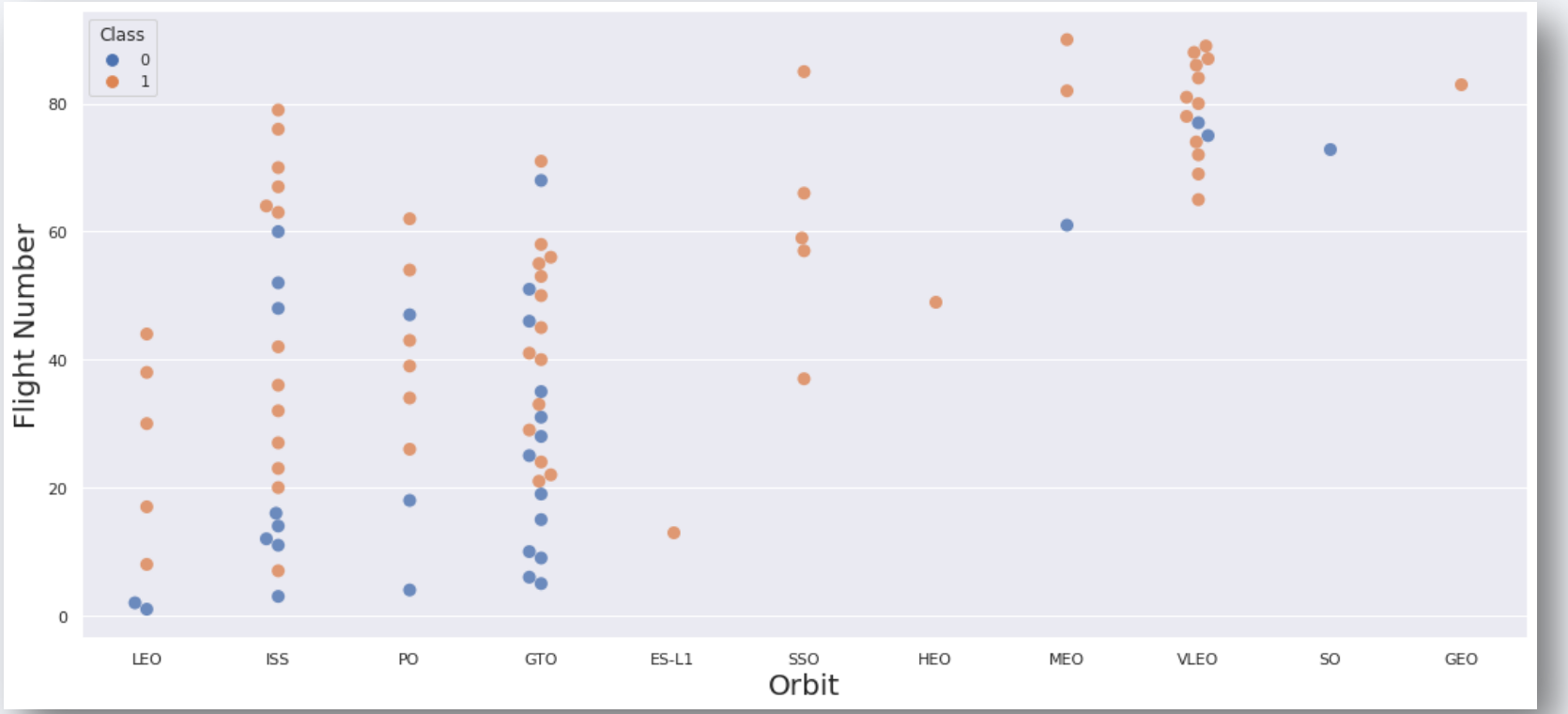
However, further analysis show that some of this orbits have only 1 occurrence (GEO, SO, HEO and ES-L1) which means this data needs more dataset to see patterns or trends before we draw any conclusion.



Flight Number vs. Orbit Type

This scatter plot interpretation is the larger the flight number on each orbits, the greater the success rate (especially LEO orbit) except for GTO orbit which depicts no relationship between both attributes.

Orbit with only 1 occurrence should also be excluded from above statement as it's needed more dataset.

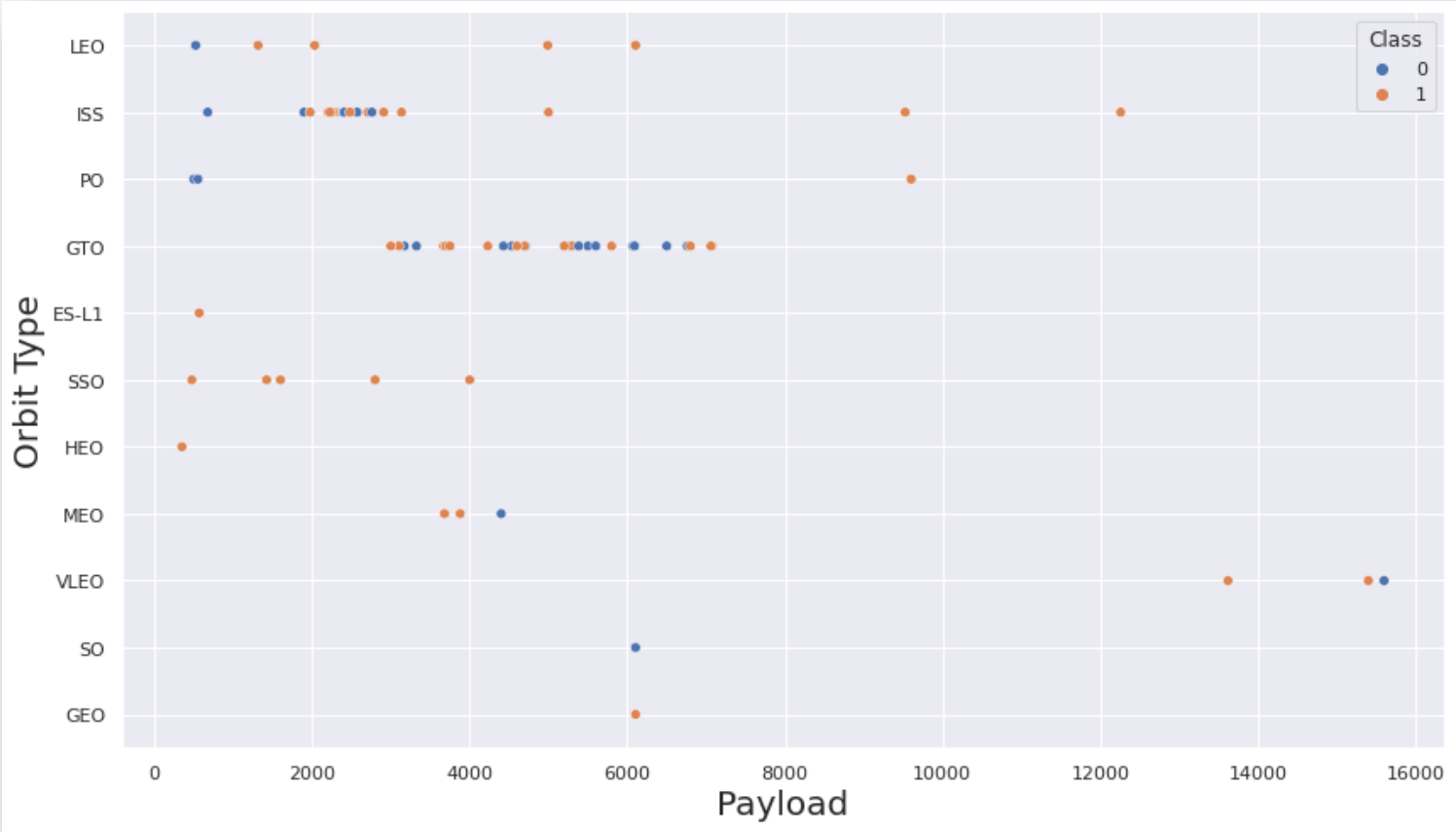


Payload vs. Orbit Type

Heavier payload has positive impact on LEO, ISS and PO orbit. However, it has negative impact on MEO and VLEO orbit.

GTO orbit seem to depict no relation between the attributes.

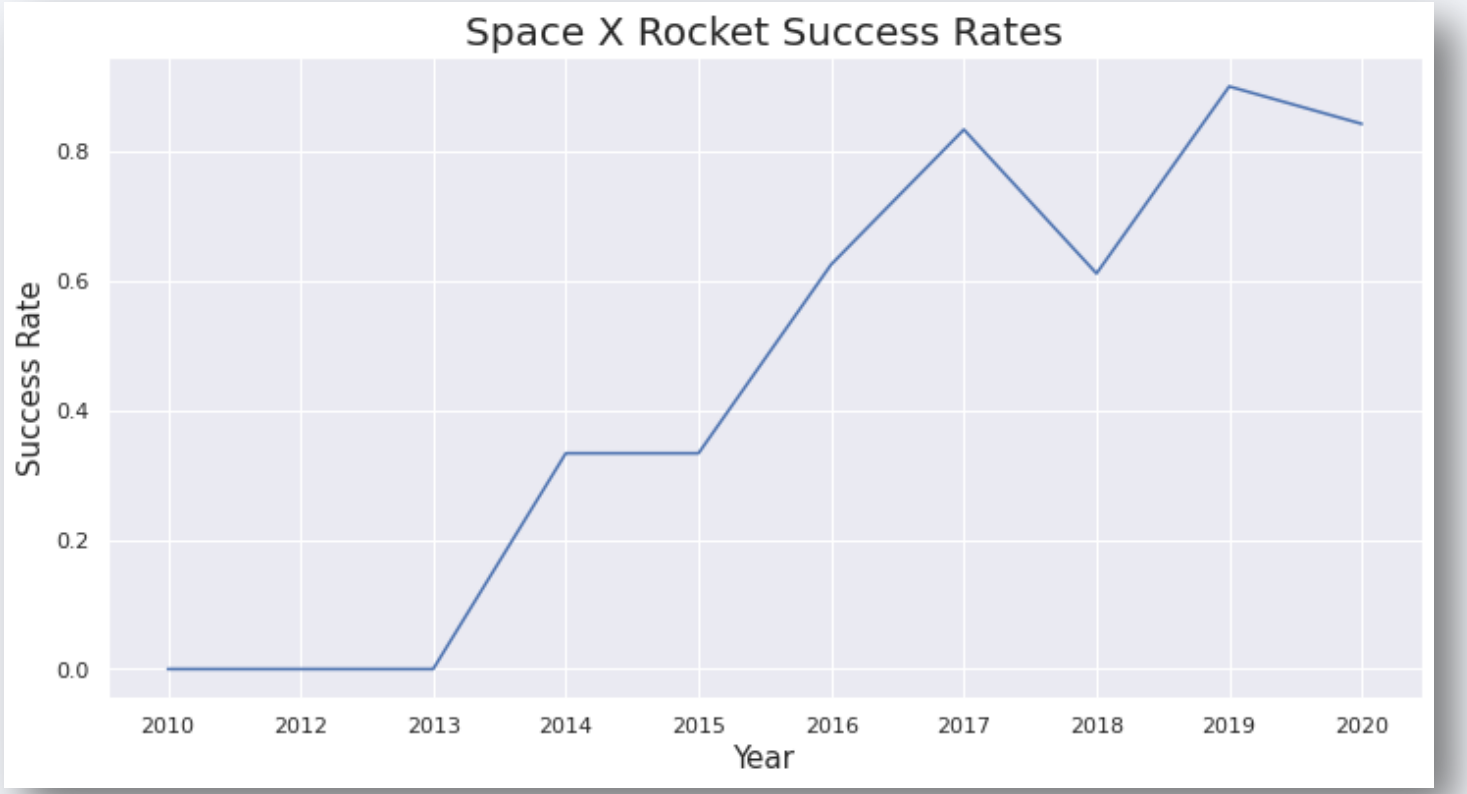
Meanwhile, again, SO, GEO and HEO orbit need more dataset to see any pattern or trend.



Launch Success Yearly Trend

This line chart clearly describes the increasing trend from the year 2013 until 2020.

If this trend continues for the next year onward. The success rate will steadily increase until reaching 100% success rate.



All Launch Site Names

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

```
[11]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
 * sqlite:///my_data1.db
Done.
```

[11]:

Launch_Sites
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`

```
[10]: %sql SELECT LAUNCH_SITE FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
 * sqlite:///my_data1.db
Done.
```

[10]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

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 by NASA (CRS)
```

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

Total Payload Mass by NASA (CRS)
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_) AS "Average Payload Mass by Booster
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

First Successful Ground Landing Date

We use the min() function to find the result
We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

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

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

Done.
```

First Succesful Landing Outcome in Ground Pad
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 SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
* 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 Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

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

Successful Mission
100

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

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

Failure Mission
1

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 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);
```

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

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

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

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

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

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 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 ;
```

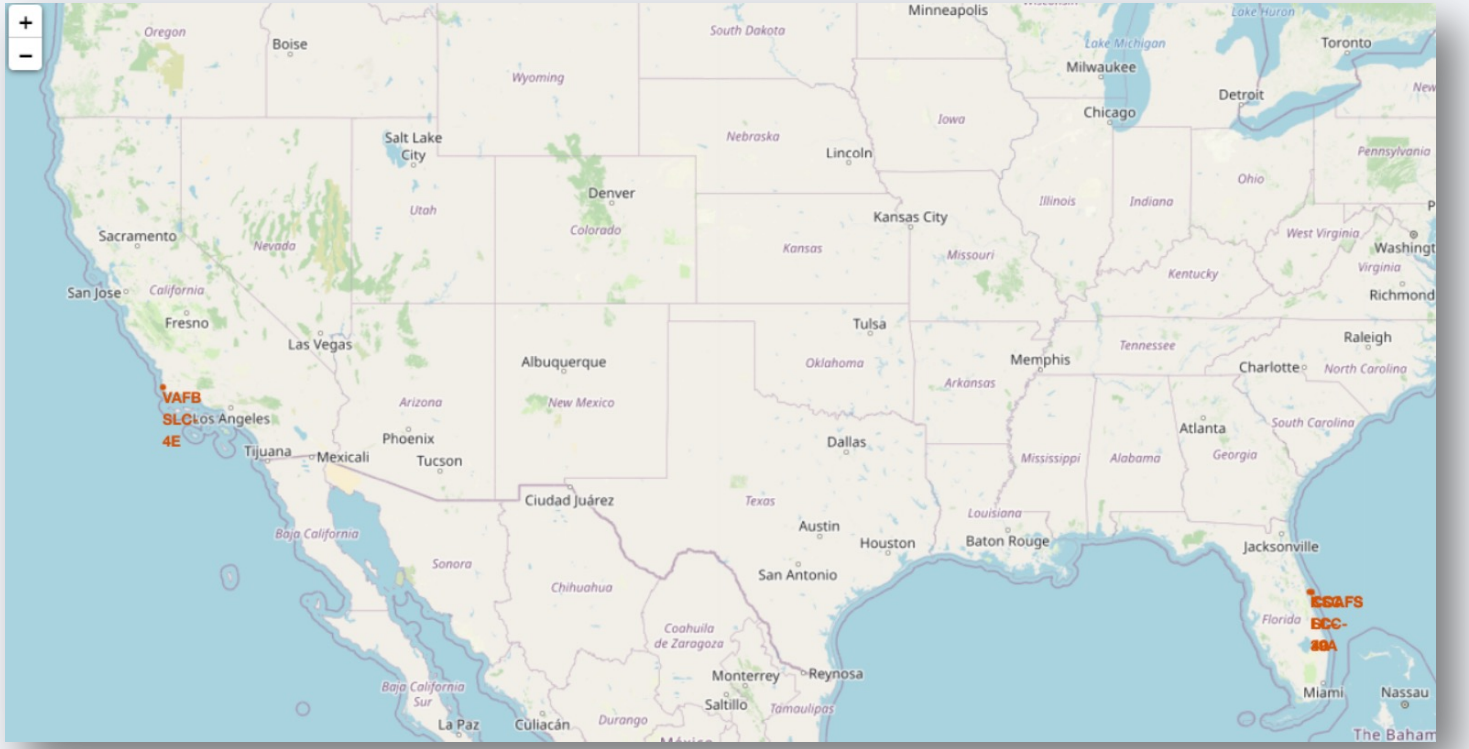
```
 * sqlite:///my_data1.db
Done.
```

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

Location of all the Launch Sites

We used wildcard like '%' to filter for WHERE MissionOutcome was a success or a failure.

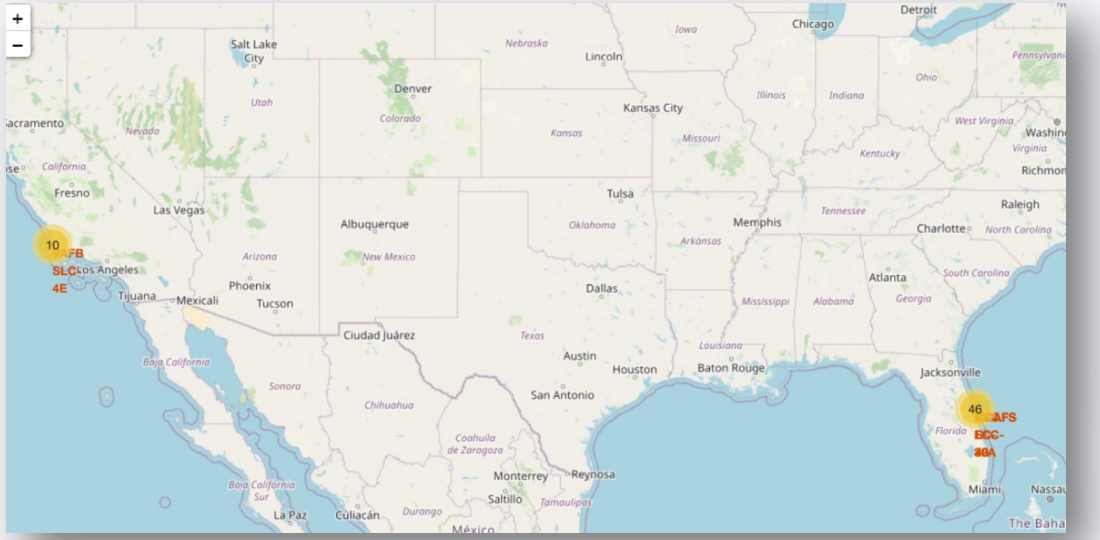


Red point indicate missions with success or failure21qw

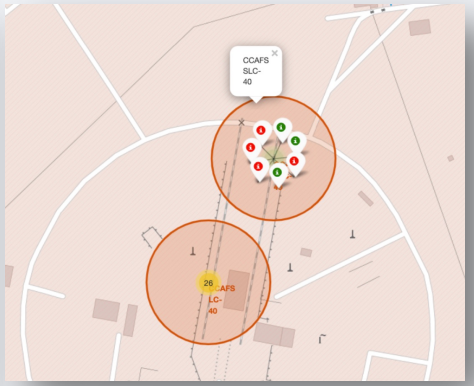
Markers showing launch sites with color labels

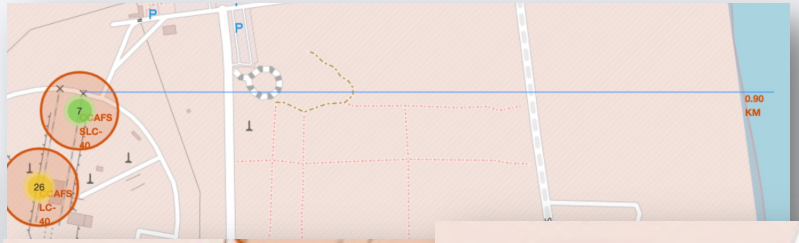
Green Markers show successful Launches and Red Markers shows failures

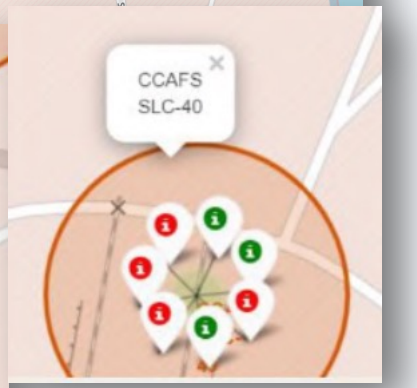
US MAP



Florida Launch Site



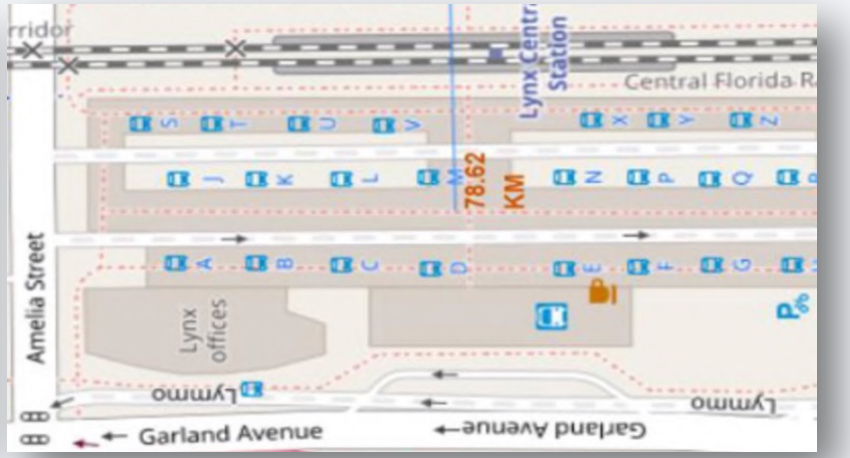




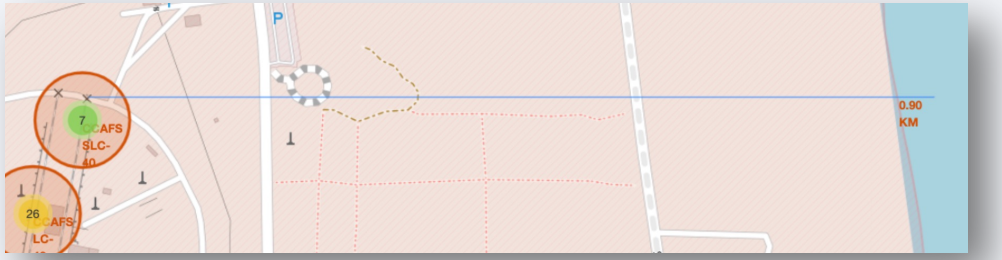
California Launch Site

Launch Sites Distance to Landmarks

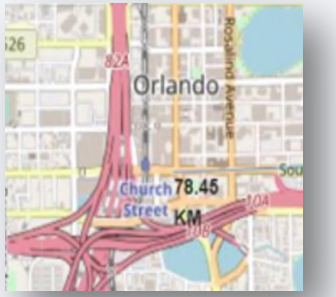
Launch sites are not near railways and highways

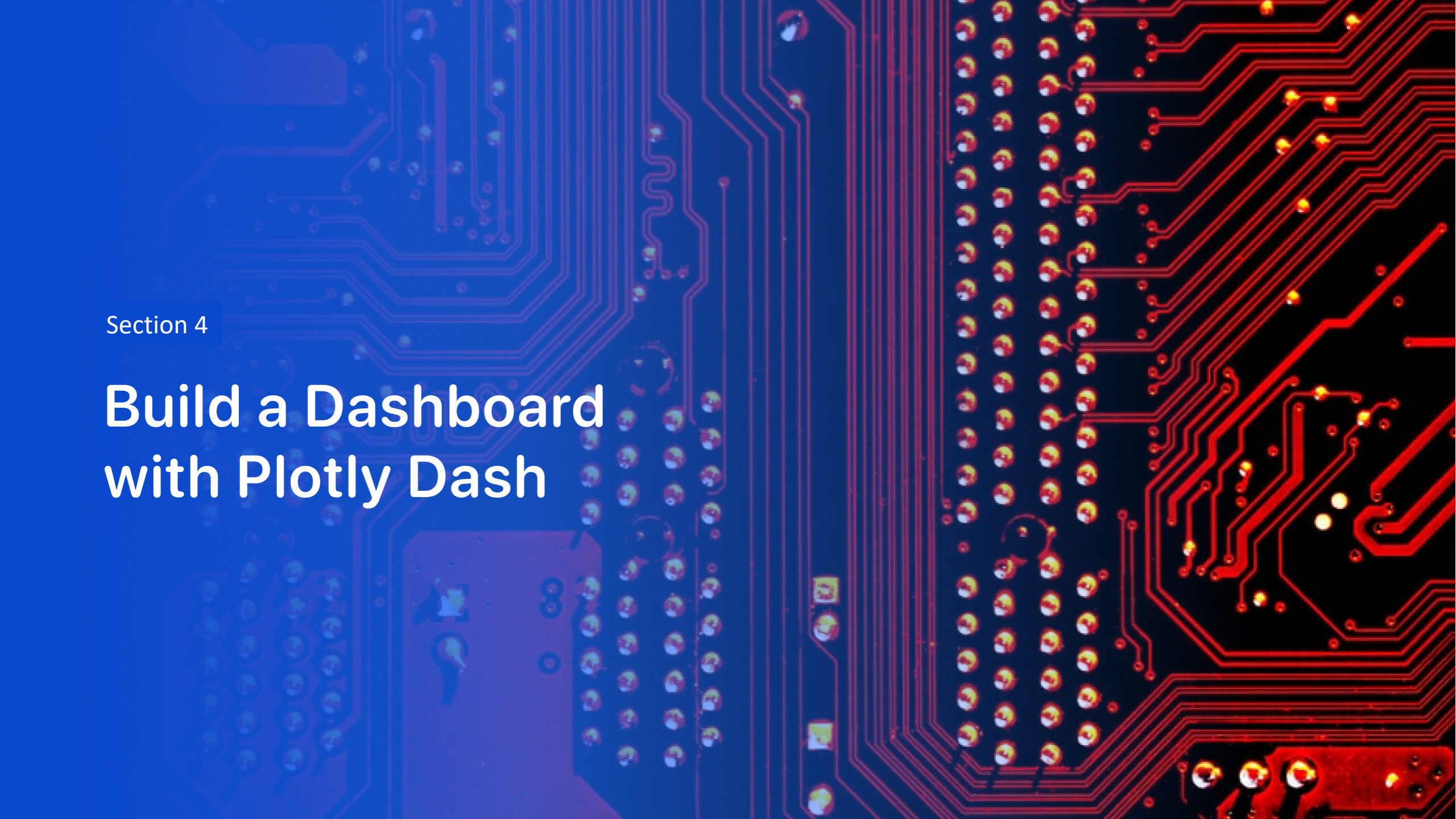


Launch sites are near coastlines

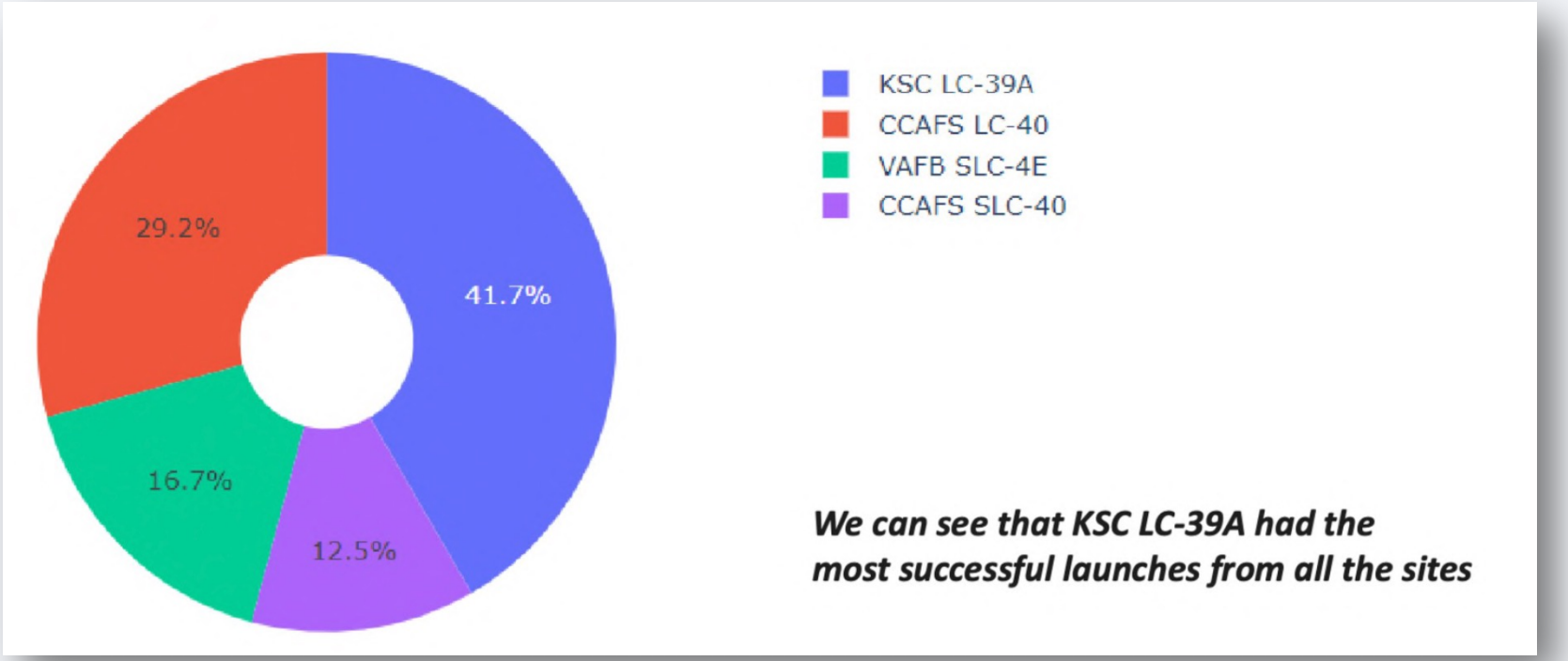


Launch sites keep certain distance from cities

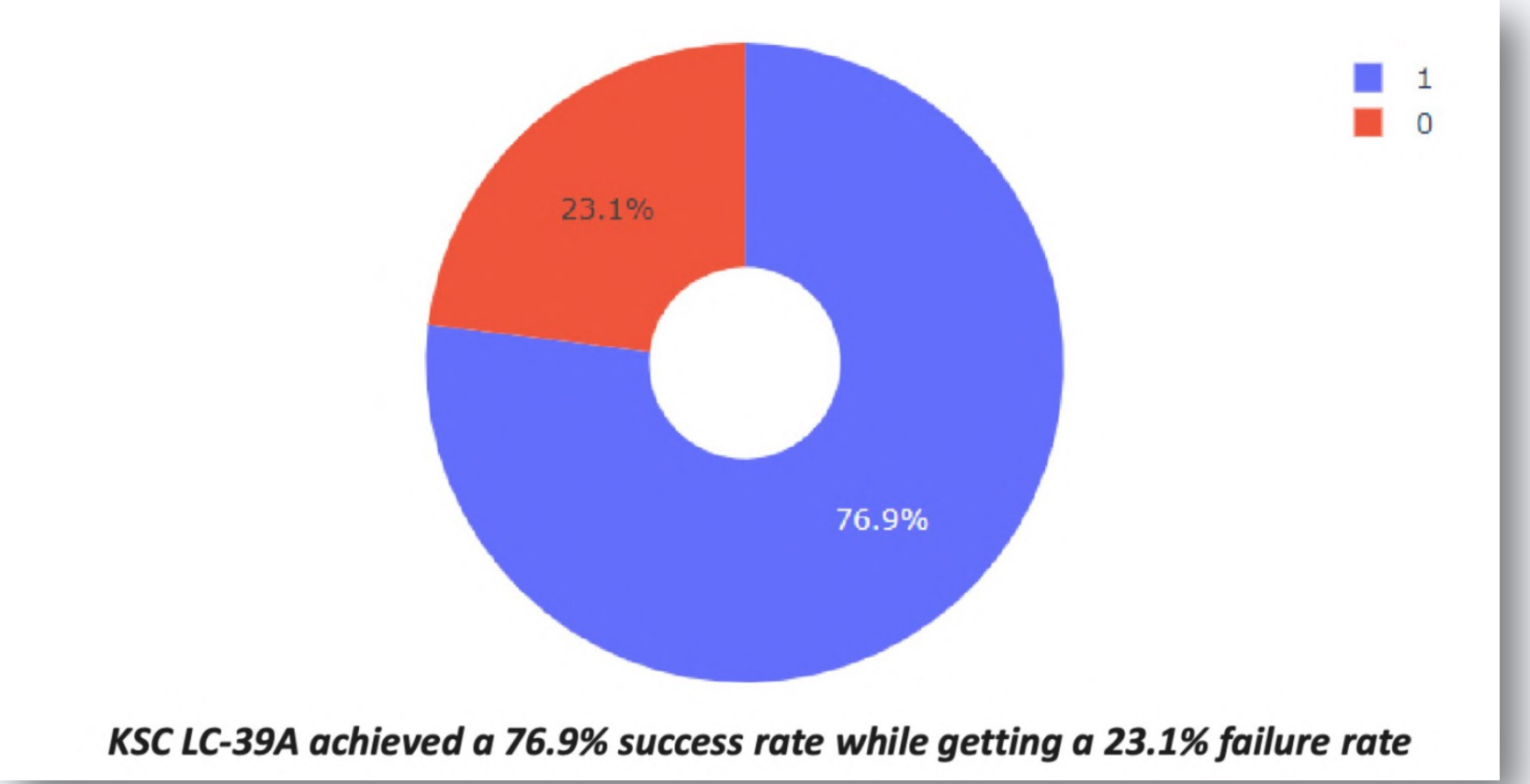



Section 4
Build a Dashboard with Plotly Dash

The success percentage by each sites.



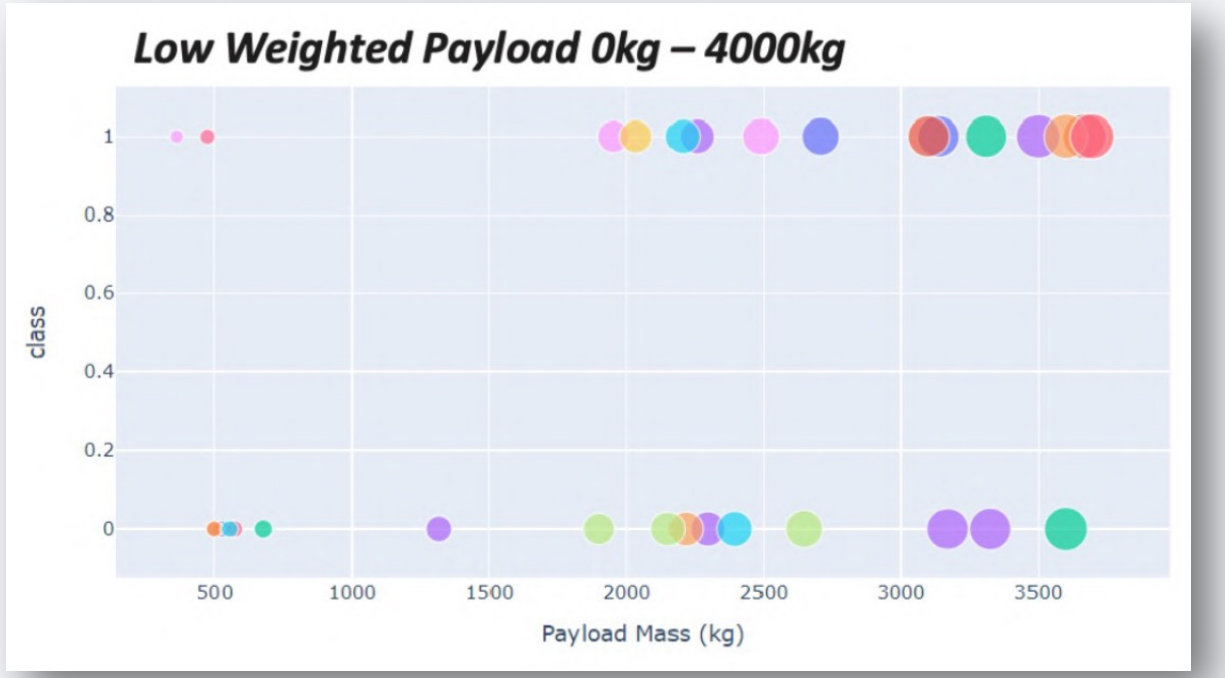
The highest launch-success ratio: KSC LC-39A

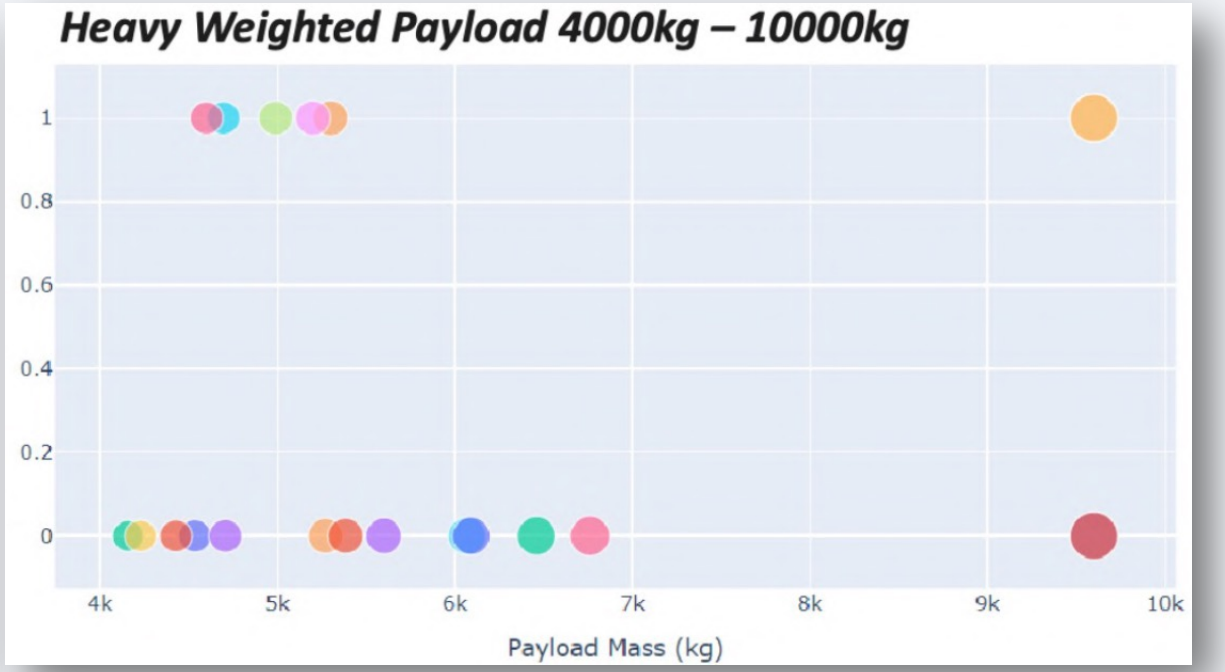


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch Outcome Scatter Plot

We can see that all the success rate for low weighted payload is higher than heavy weighted payload





Section 5

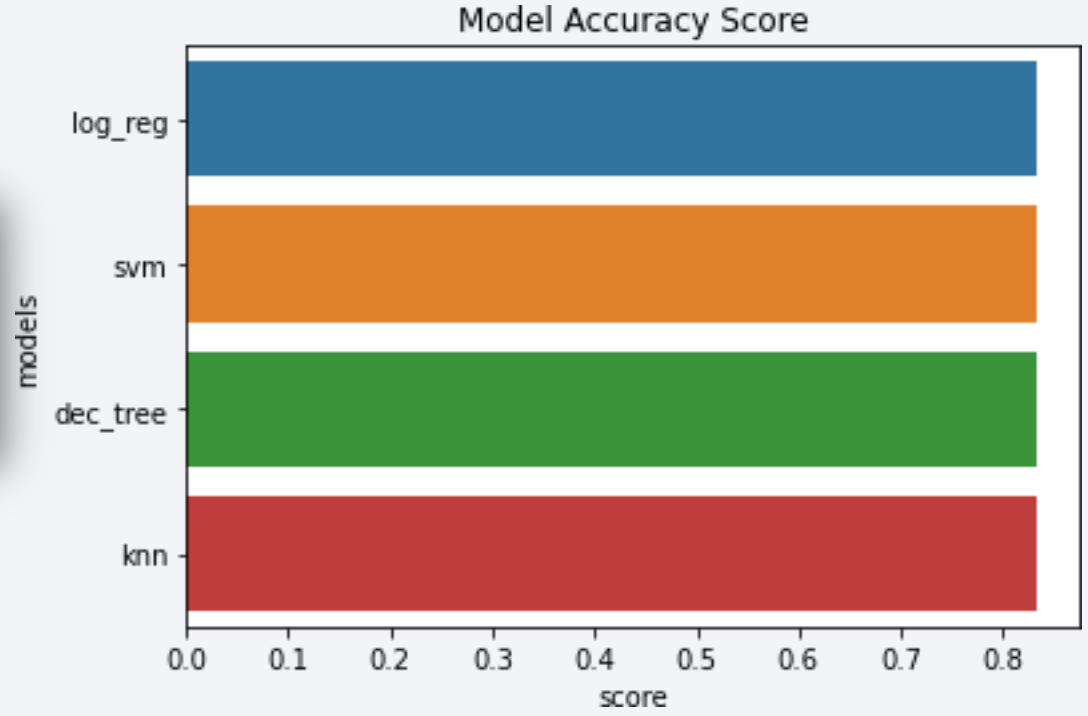
Predictive Analysis (Classification)

Classification Accuracy

As we can see, by using the code as below: we could identify that the best algorithm to be the Tree Algorithm which have the highest classification accuracy.

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

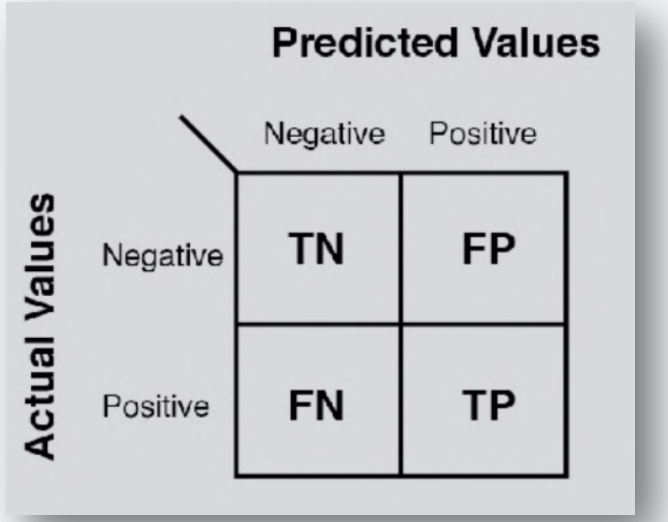
```
Best Algorithm is Tree with a score of 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'random'}
```

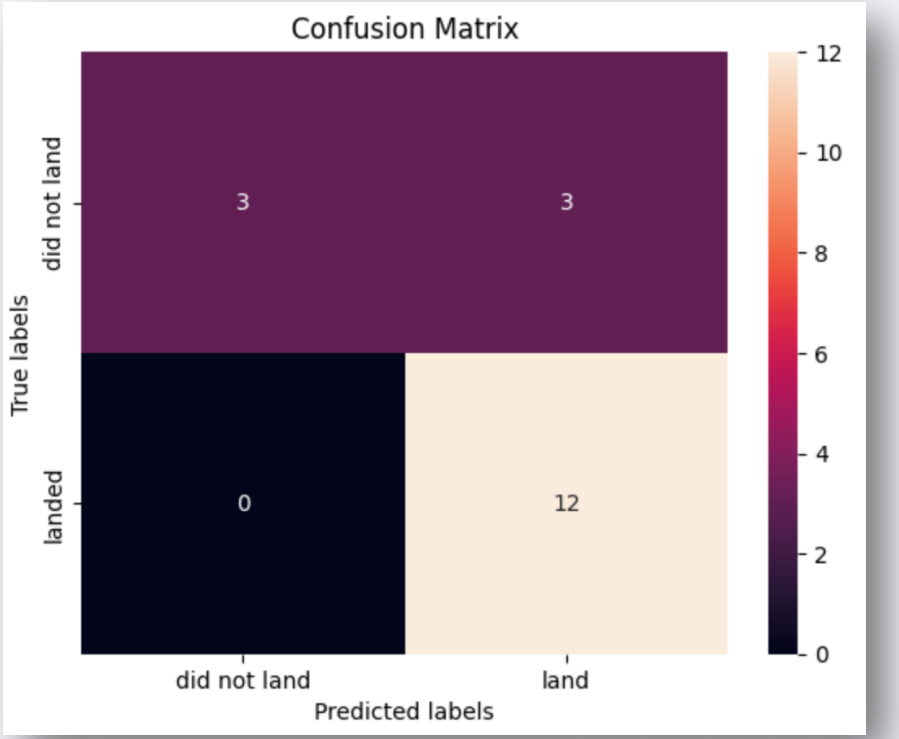


Confusion Matrix

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.

Reference





Conclusions

- The goal of model is to predict when Stage 1 will successfully land to save ~$100 million USD
- The Tree Classifier Algorithm is the best Machine Learning approach for this dataset.
- he low weighted payloads (which define as 4000kg and below) performed better than the heavy weighted payloads.
- Starting from the year 2013, the success rate for SpaceX launches is increased, directly proportional time in years to 2020, which it will eventually perfect the launches in the future.
- Created a dashboard for visualization years
- SSO orbit have the most success rate; 100% and more than 1 occurrence.
- Created a machine learning model with an accuracy of 83%

Appendix

- Skills Labs
- IBM Data Science Professional Certificate Course

Thank you!