Winning Space Race with Data Science

Sonali Arora
16th December 2023

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
  - Prediction Analysis using Machine Learning
- Summary of all results
  - Exploratory Data Analysis with SQL
  - Interactive Analytics through screenshots
  - Predictive Analysis results from Machine Learning Lab

Introduction

SpaceX stands out as an innovative organization that has transformed the space sector by providing rocket launches, specifically the Falcon 9, at a significantly reduced cost of 62 million dollars. This is in stark contrast to other providers whose prices can go as high as 165 million dollars per launch. The noteworthy cost savings achieved by SpaceX can be attributed to their groundbreaking approach of reusing the initial stage of the launch. This involves successfully re-landing the rocket for deployment in subsequent missions. Iterating through this cycle is anticipated to further decrease the overall launch costs.

As a data scientist employed by a startup in competition with SpaceX, the primary objective of our project is to develop a machine learning pipeline capable of predicting the landing outcome of the first stage in upcoming launches. This initiative is of paramount importance in determining the optimal bidding price against SpaceX for rocket launches.

The problems included in this presentation are:

1. Identifying all factors that influence the landing outcome.
2. The relationship between each variables and how it is affecting the outcome.
3. The best condition needed to increase the probability of successful landing

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using the SpaceX's REST API and Web Scrapping from Wikipedia/
- Perform data wrangling
  - Data was processed using one-hot encoding technique 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

Data collection involves the systematic acquisition and measurement of information related to specific variables within an established system. This process facilitates the ability to address pertinent questions and assess outcomes. In this context, the dataset was obtained through a combination of REST API and web scraping techniques from Wikipedia.

For the REST API approach, the process commenced with a get request. Subsequently, the response content was decoded as JSON, and the data was transformed into a pandas DataFrame using the json_normalize() function. Following this, data cleaning procedures were implemented, including the identification and handling of missing values.

In the case of web scraping, BeautifulSoup was employed to extract launch records presented as an HTML table. The table was then parsed, and the extracted information was converted into a pandas DataFrame for in-depth analysis.

Data Collection – SpaceX API

- Get Request for Rocket Launch using the API
- Using json_normalize() method to convert JSON result to DataFrame
- Performing data cleaning and filling of the missing values.

Github URL:

```
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 a
nd 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 rocket
s with 2 extra rocket boosters and rows that have multiple payloads in a
single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the s
ingle 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 ex
tracting 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 Falcon 9 Launch Wikipedia Page from given URL
- Create a BeautifulSoup object from the response text
- Extract all column or variable names from HTML header

GitHub URL:

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

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

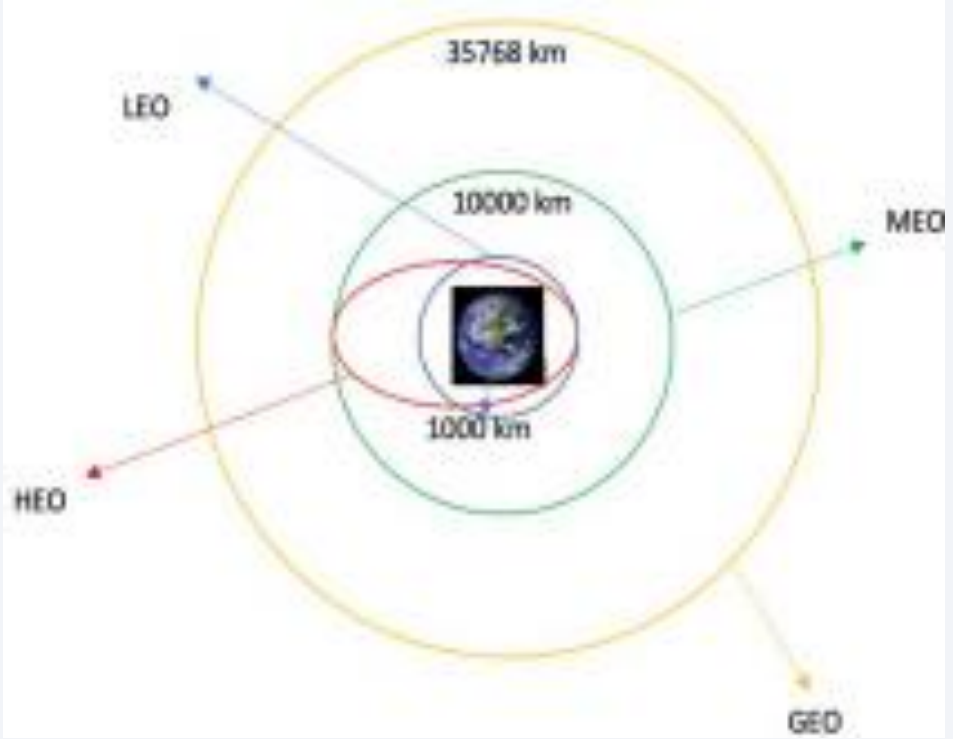
```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plai
nrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding t
o launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
```

Data Wrangling

Data wrangling is the practice of refining and consolidating untidy and intricate datasets to facilitate seamless access and support Exploratory Data Analysis (EDA). Our approach involves several key steps to enhance the dataset's usability.

Initially, we will compute the count of launches at each site. Following that, we will determine the number and frequency of mission outcomes based on orbit types. To streamline further analysis, visualization, and machine learning (ML) applications, we will generate a landing outcome label derived from the existing outcome column. This label will simplify subsequent processes.

Ultimately, we plan to export the refined dataset, along with the incorporated changes, to a CSV file for broader usability and dissemination of results.



GitHub URL:

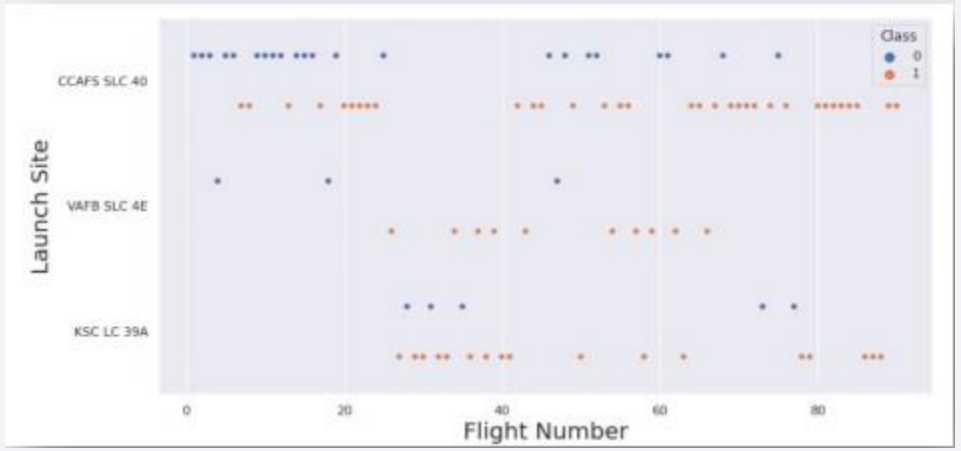
EDA with Data Visualization

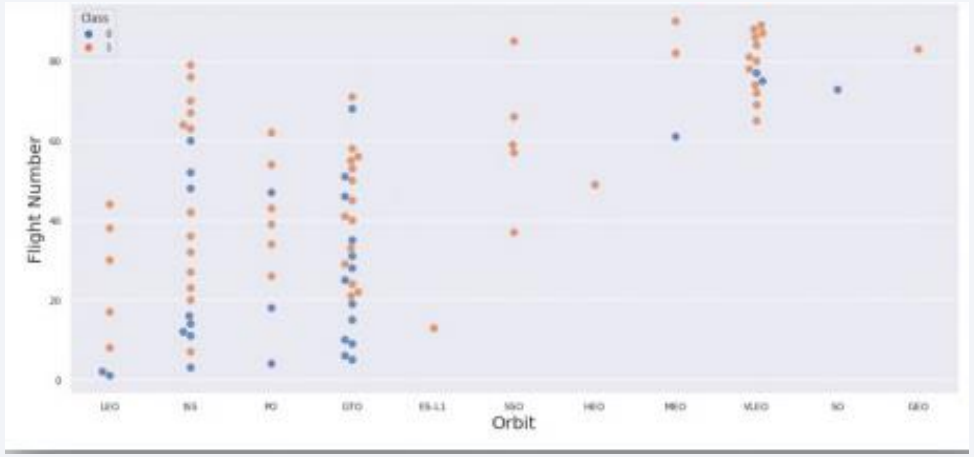
We first started by using scatter graph to find the relationship between the attributes such as between:

- Payload and Flight Number.
- Flight Number and Launch Site.
- Payload and Launch Site.
- Flight Number and Orbit Type.
- 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.

GitHub URL:





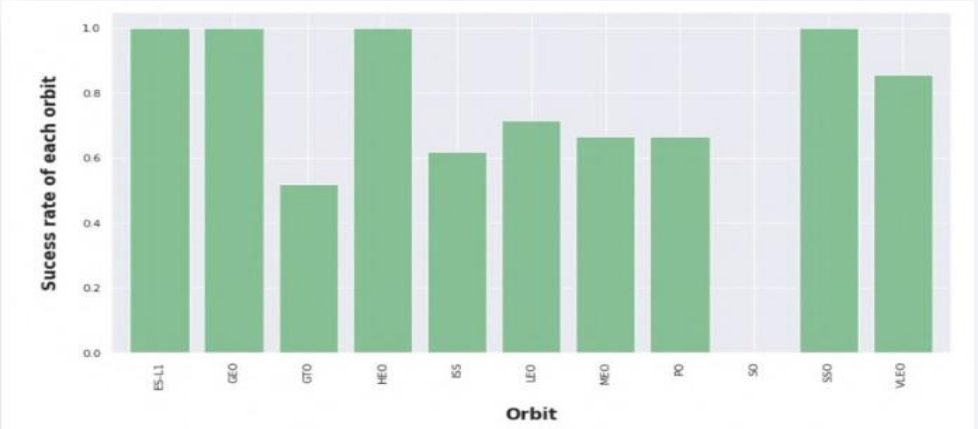
EDA with Data Visualization

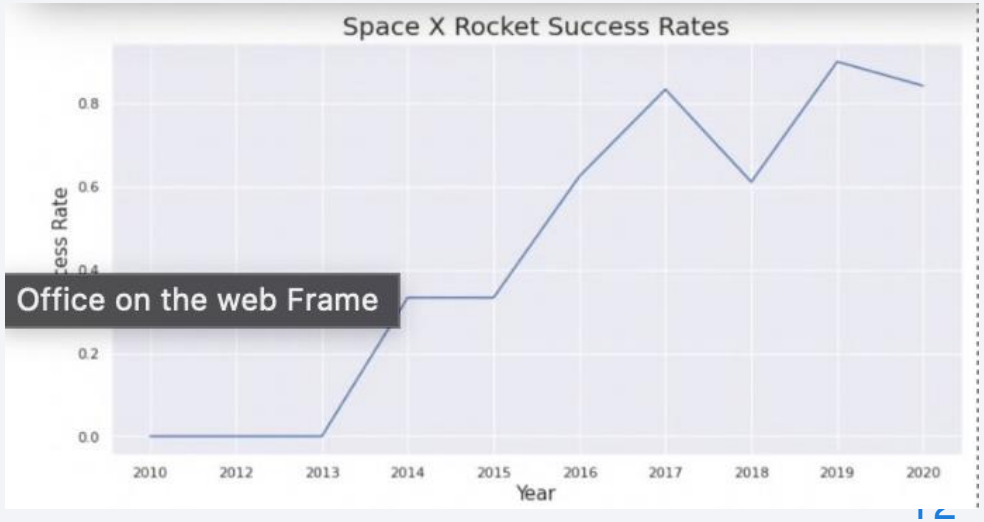
Once we detect potential relationships through a scatter plot, we will employ additional visualization tools such as bar graphs and line plots for a more in-depth analysis.

Bar graphs provide a straightforward method for interpreting relationships between attributes. In this instance, we will utilize a bar graph to identify the orbits with the highest probability of success.

Subsequently, a line graph will be employed to illustrate trends or patterns of attributes over time. In this specific case, it will be used to observe the yearly trend in launch success.

To enhance success prediction in future modules, Feature Engineering will be implemented. This involves creating dummy variables for categorical columns





Office on the web Frame

GitHub URL:

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.

Build an Interactive Map with Folium

- To represent the launch data visually on an interactive map, we utilized the latitude and longitude coordinates for each launch site. We incorporated a circle marker around each launch site, accompanied by a label indicating the name of the respective 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 have also used the Harvesine's formula to calculate the distance of the launch sites to various landmarks to find the answer to the below questions:
  - How close the launch sites with railways, highways and coastlines?
  - How close the launch sites with nearby cities?

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash which allowing the user to play ar ound 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.
- GITHUB URL:

Predictive Analysis (Classification)

Building the Model

- Load the dataset into Numpy and Pandas
- Transform the data and then split into train and test datasets
- Decide which ML model to use on the same
- Set the parameters and algorithms to GridSearchCV and fit it to the dataset

Evaluating the model

- Check the accuracy of each model
- Get tuned hyperparameters for each type of algorithms
- Plot the confusion matrix for the same.

Enhance the model

- Make use of Feature Engineering and Algorithm Tuning

Find the Best fit

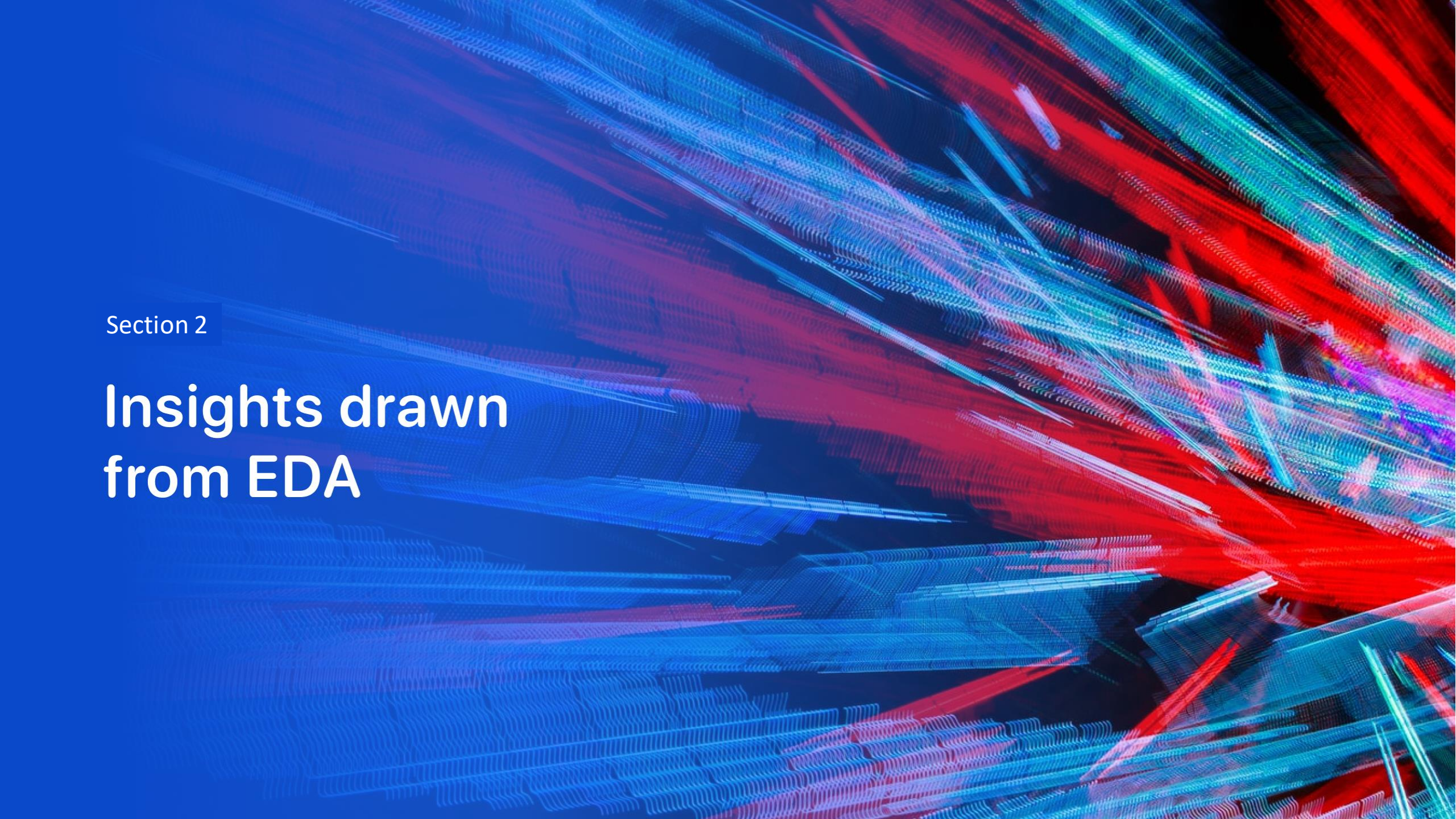
- The ML model with best accuracy will be considered as the best performing model.
- GitHub Link:

Results

- The results will be categorized to 3 main results which is:
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

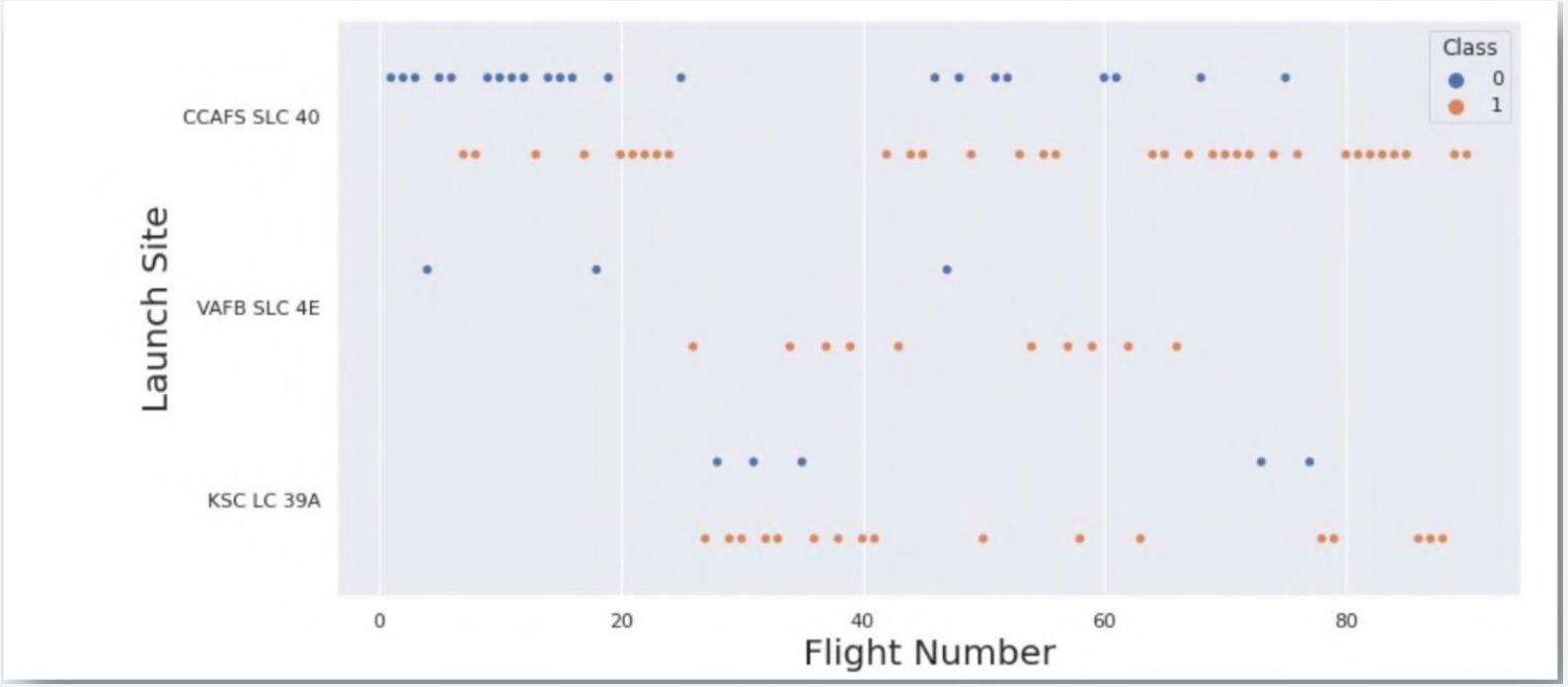
Section 2

Insights drawn from EDA



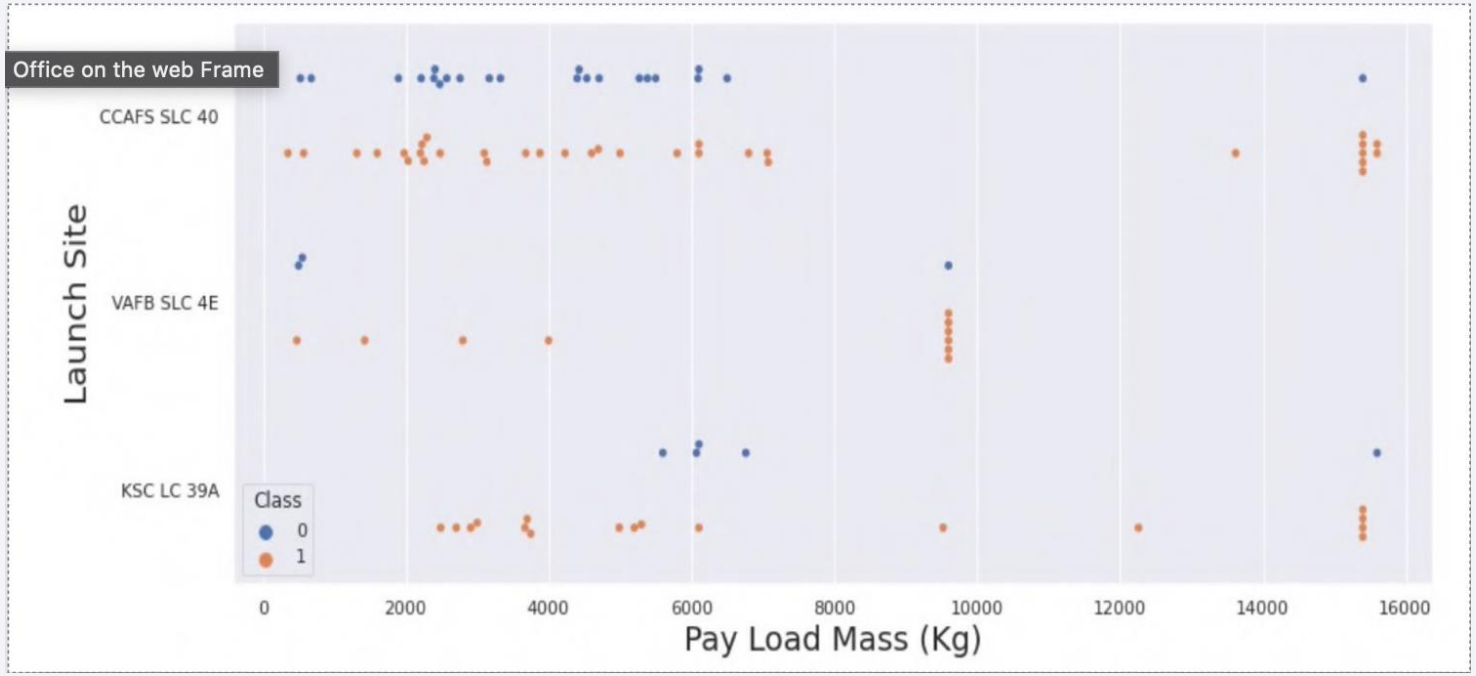
Flight Number vs. Launch Site

- This scatter plot shows that the larger the flights amount of the launch site, the greater the success rate will be.
- However, the site CCAFS SLC40 shows the least patter for this.



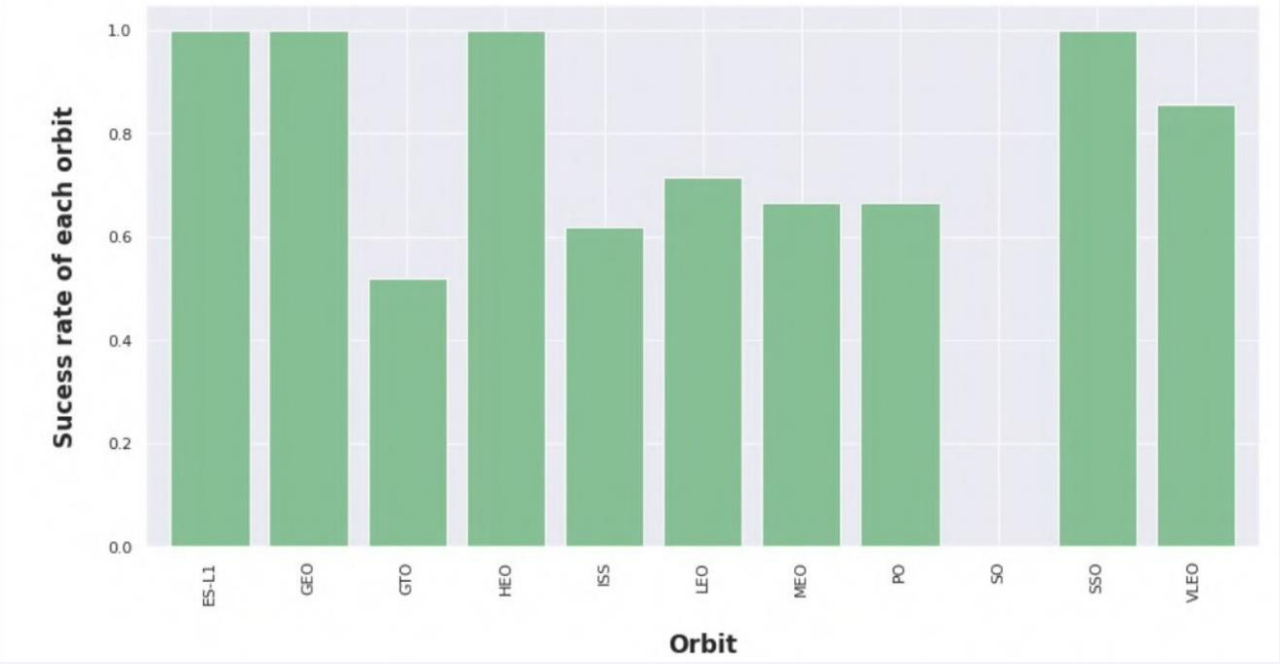
Payload vs. Launch Site

- This scatter plot shows once the payload mass is greater than 7000kg, the probability of th e success rate will be highly increased.
- However, there is no clear pattern to say the launch site is dependent to the payload mass f or the success rate.



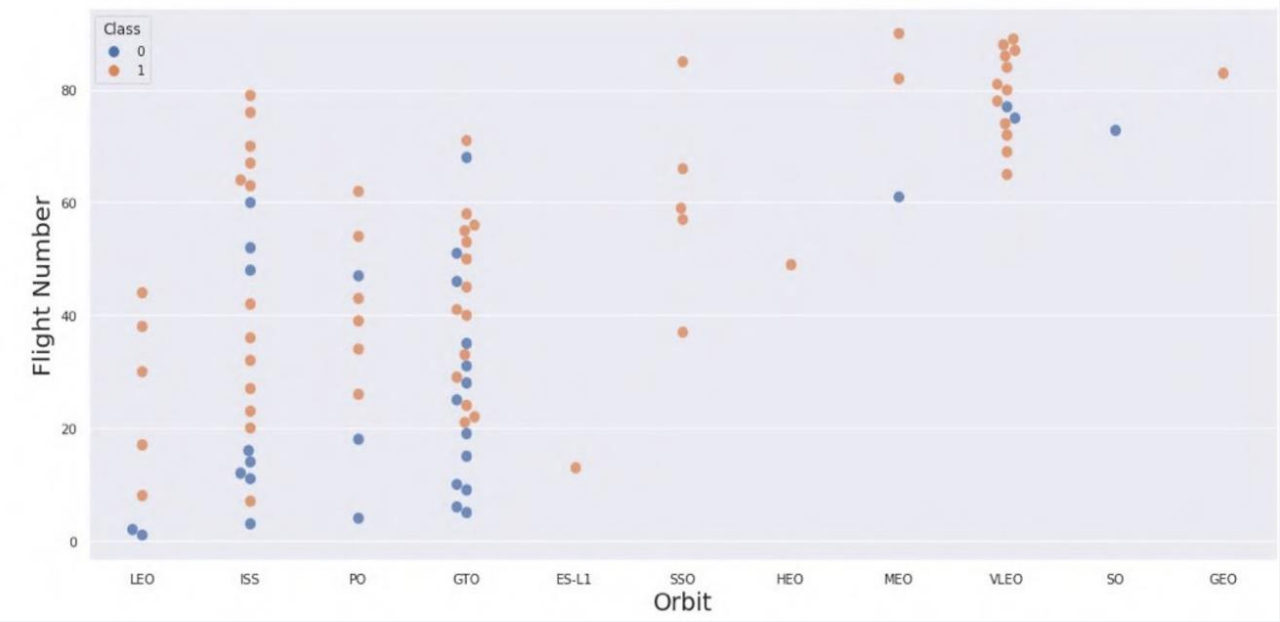
Success Rate vs. Orbit Type

- The figure illustrates the potential impact of different orbits on landing outcomes, revealing that certain orbits, such as SSO, HEO, GEO, and ES L1, boast a 100% success rate, while the SO orbit exhibits a 0% success rate.
- However, a more in-depth analysis reveals that some of these orbits have only one occurrence, including GEO, SO, HEO, and ESL1. This suggests that additional data is required to discern patterns or trends before drawing any conclusive interpretation



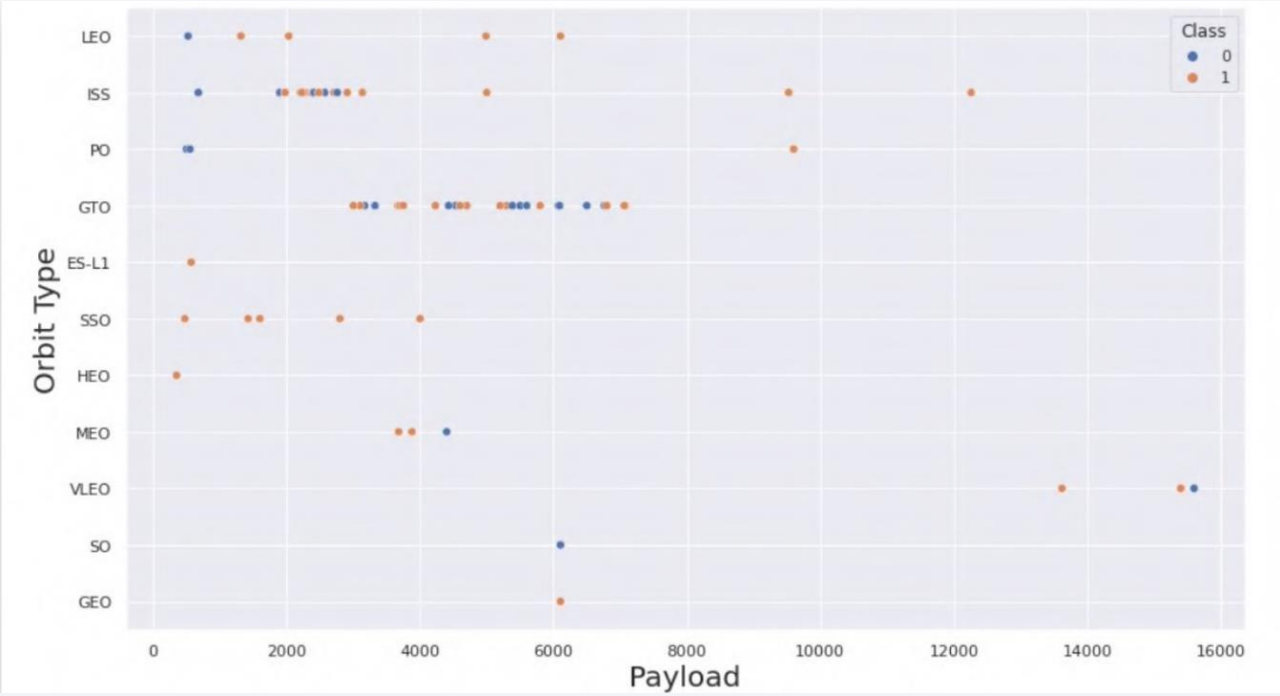
Flight Number vs. Orbit Type

- The figure portrays the potential influence of various orbits on landing outcomes, highlighting specific orbits like SSO, HEO, GEO, and ES L1, which demonstrate a 100% success rate, while the SO orbit shows a 0% success rate.
- Nevertheless, upon closer analysis, it becomes evident that some of these orbits, such as GEO, SO, HEO, and ESL1, have only one occurrence each. This underscores the need for additional data to uncover patterns or trends before making any definitive interpretations.



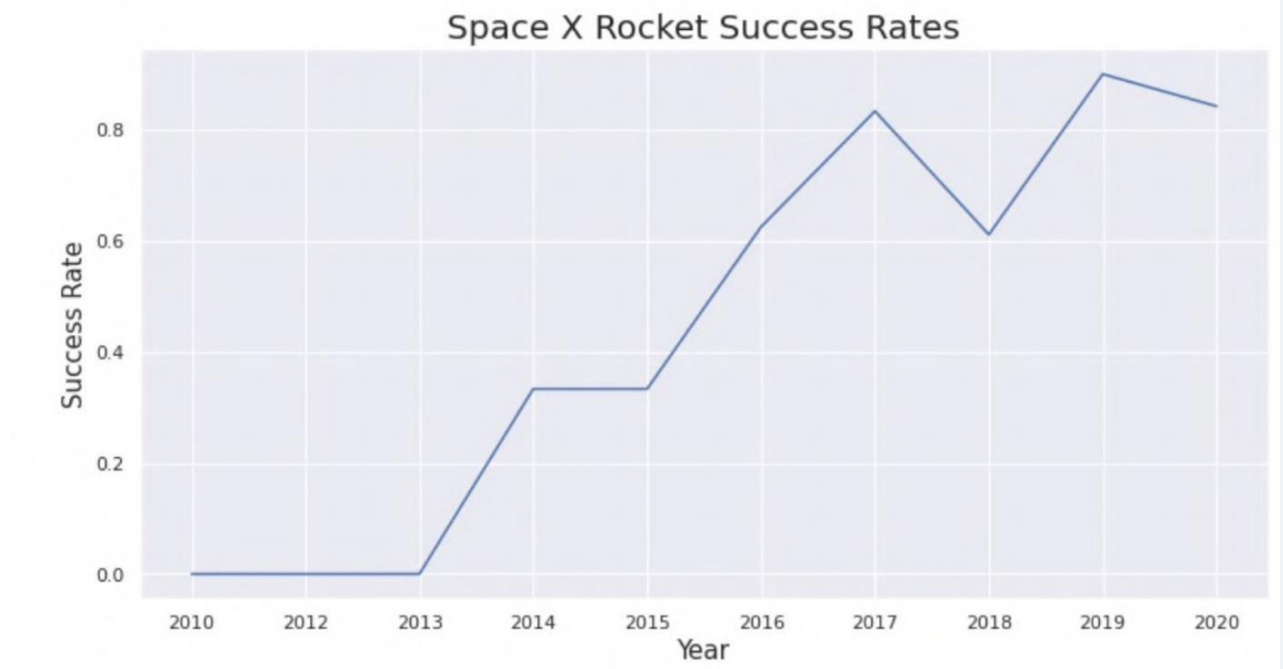
Payload vs. Orbit Type

- Heavier payload has positive impact o n LEO, ISS and PO orbit. However, it h as 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 p attern or trend.



Launch Success Yearly Trend

- Figures clearly depicted and increasing tr end from the year 2013 until 2020.
- If this trend continue for the next year onward. The success rate will steadily incre ase until reaching 1/100% success rate.



All Launch Site Names

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

```
In [5]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
Office on the web Frame 0.databases.appdomain.cloud:32731/bludb
Done.
```

Out[5]:

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

Launch Site Names Begin with 'CCA'

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

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

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

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

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

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

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

Total Payload Mass by NASA (CRS)
45596

Average Payload Mass by F9 v1.1

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

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

Average Payload Mass by Booster Version F9 v1.1
2928

First Successful Ground Landing Date

- 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)';
```

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

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

Successful Drone Ship Landing with Payload between 4000 and 6000

- 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;
```

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

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

Boosters Carried Maximum Payload

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

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

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

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

Total Number of Successful and Failure Mission Outcomes

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

List the total number of successful and failure mission outcomes

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

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

Successful Mission
100

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

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

Failure Mission
1

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)';
```

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

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

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.

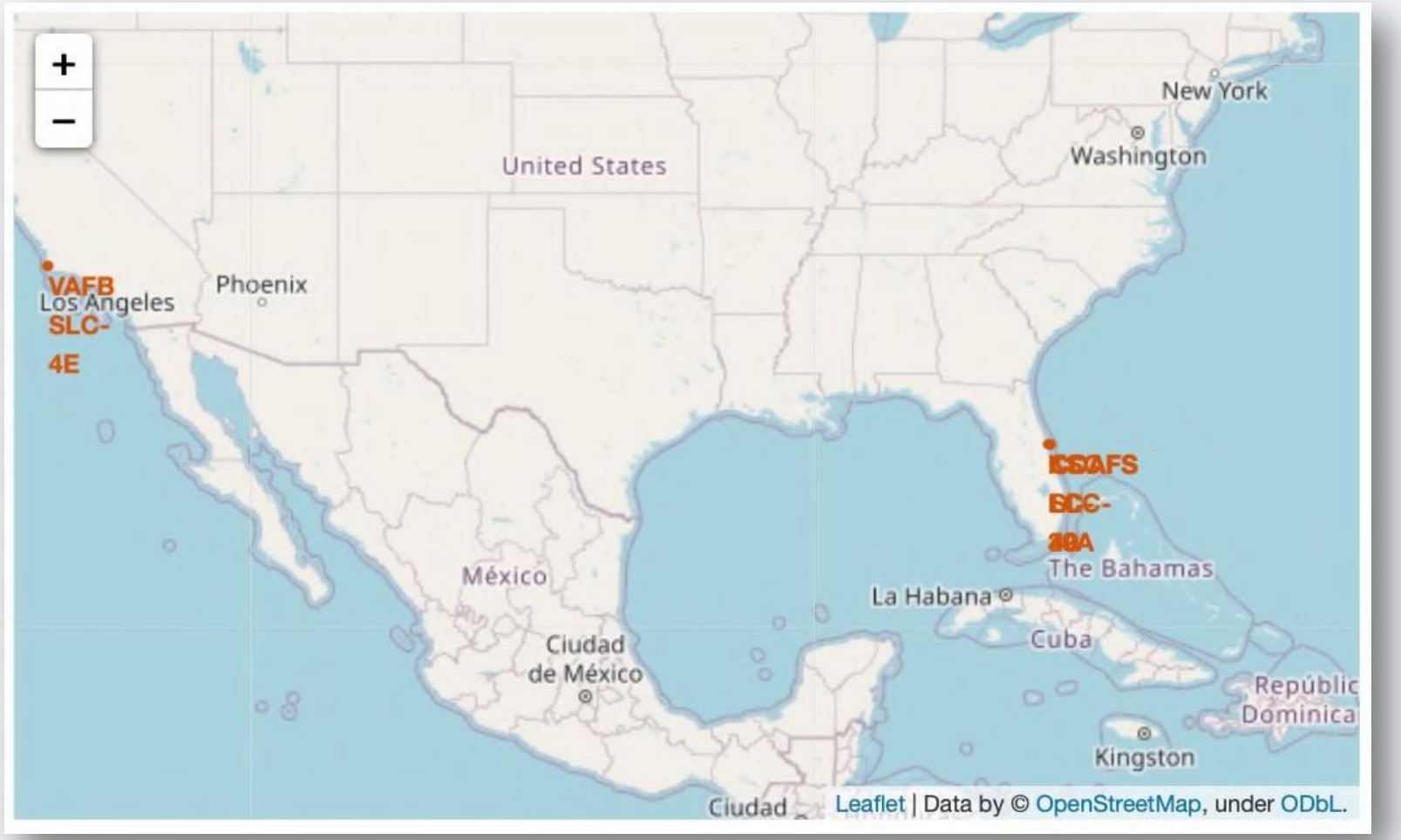
```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEX \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING__OUTCOME \
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

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

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


Section 3
Launch Sites
Proximities Analysis

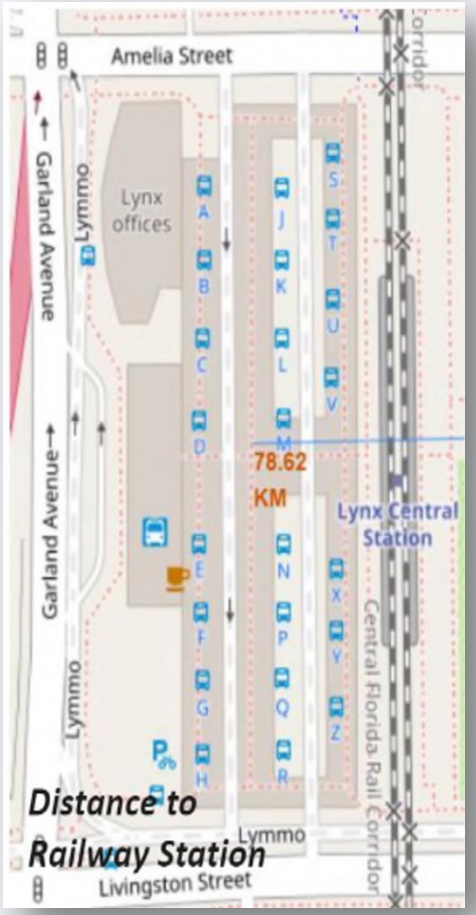
Location of all the Launch Sites

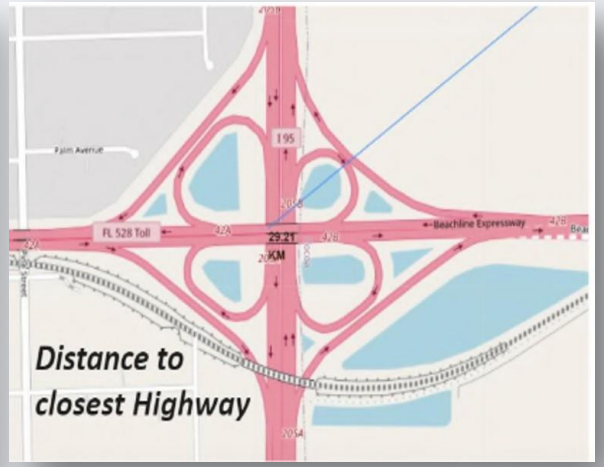


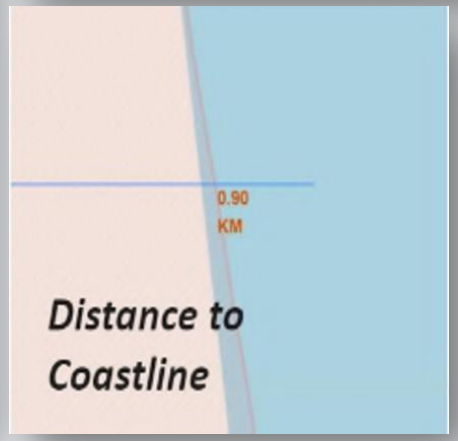
We can see from the image that all the SpaceX launch sites are located inside the United States

Markers showing launch sites with color labels

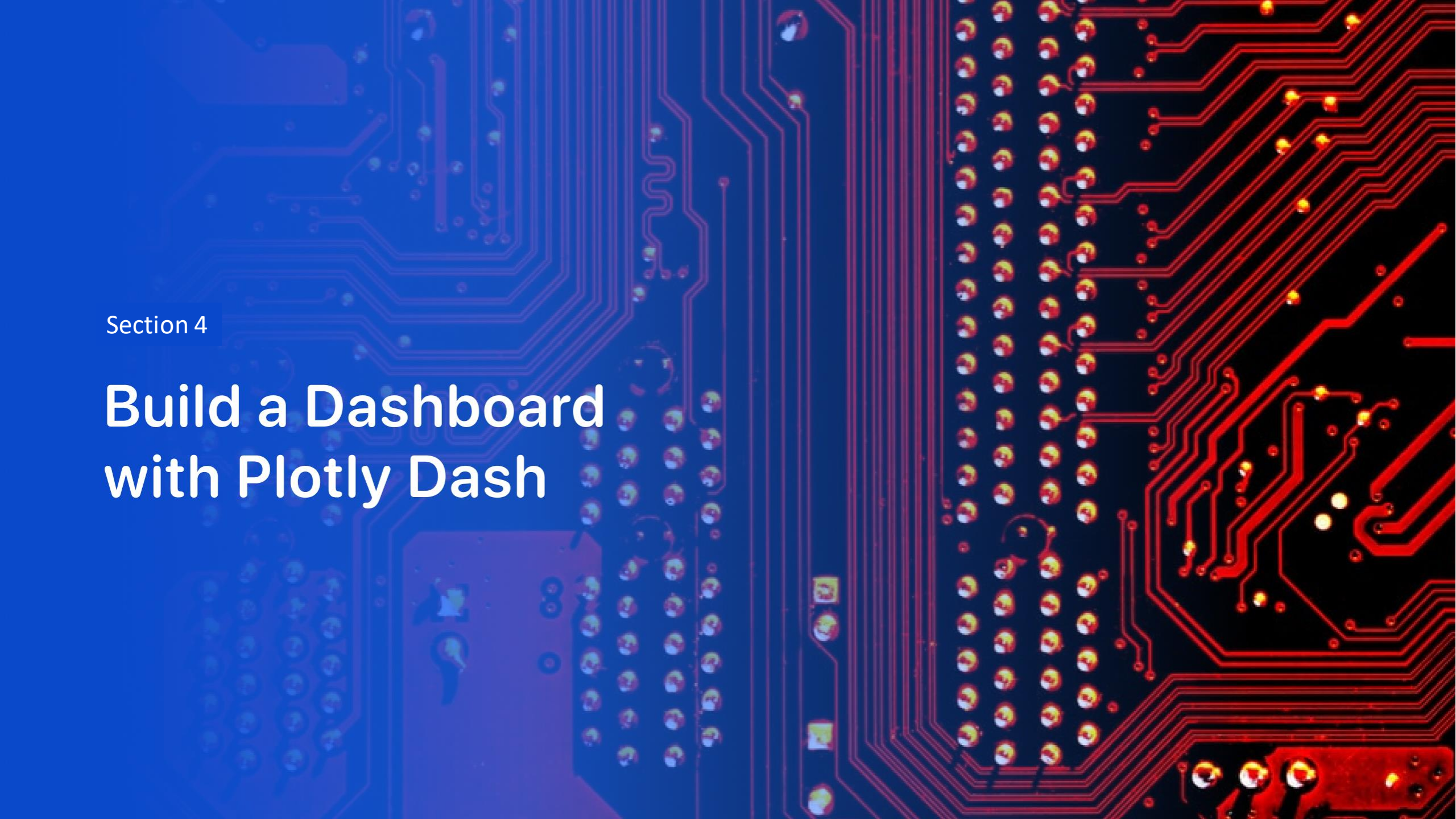
Launch Sites Distance to Landmarks



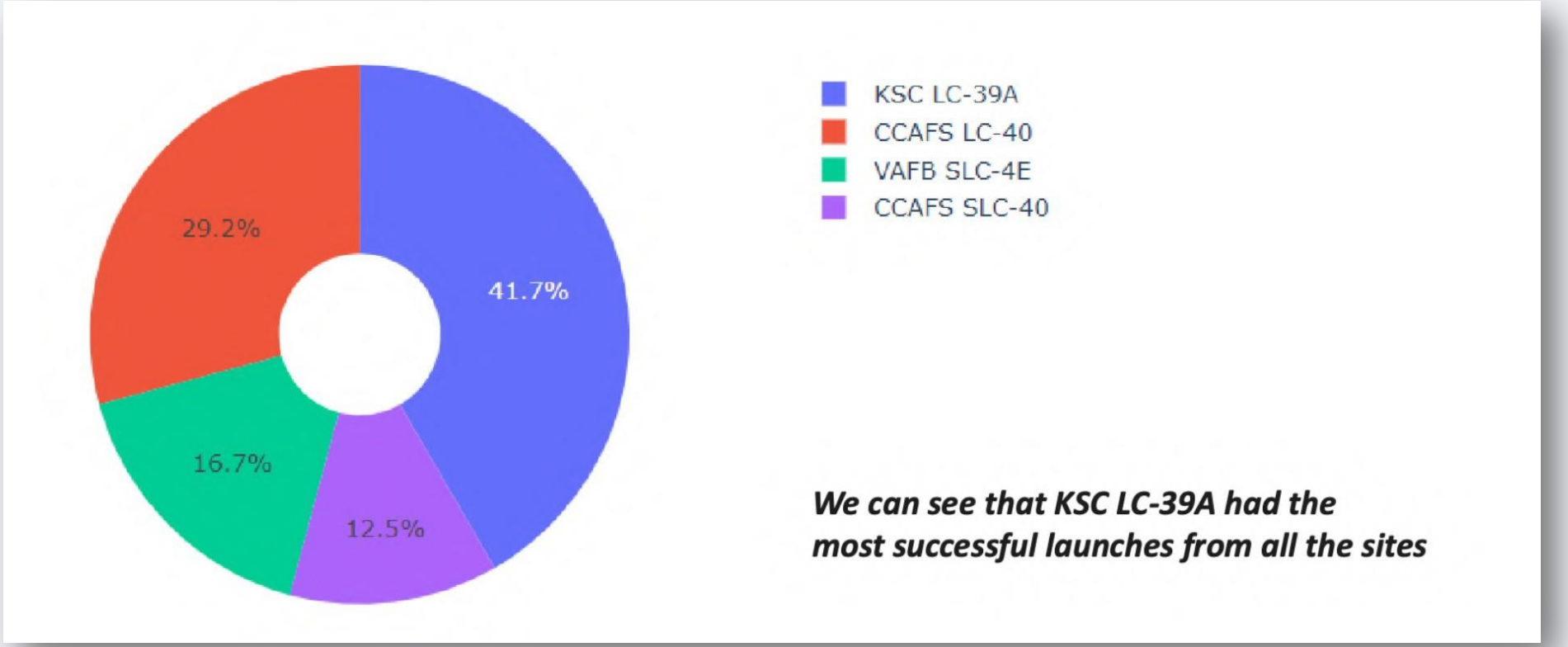




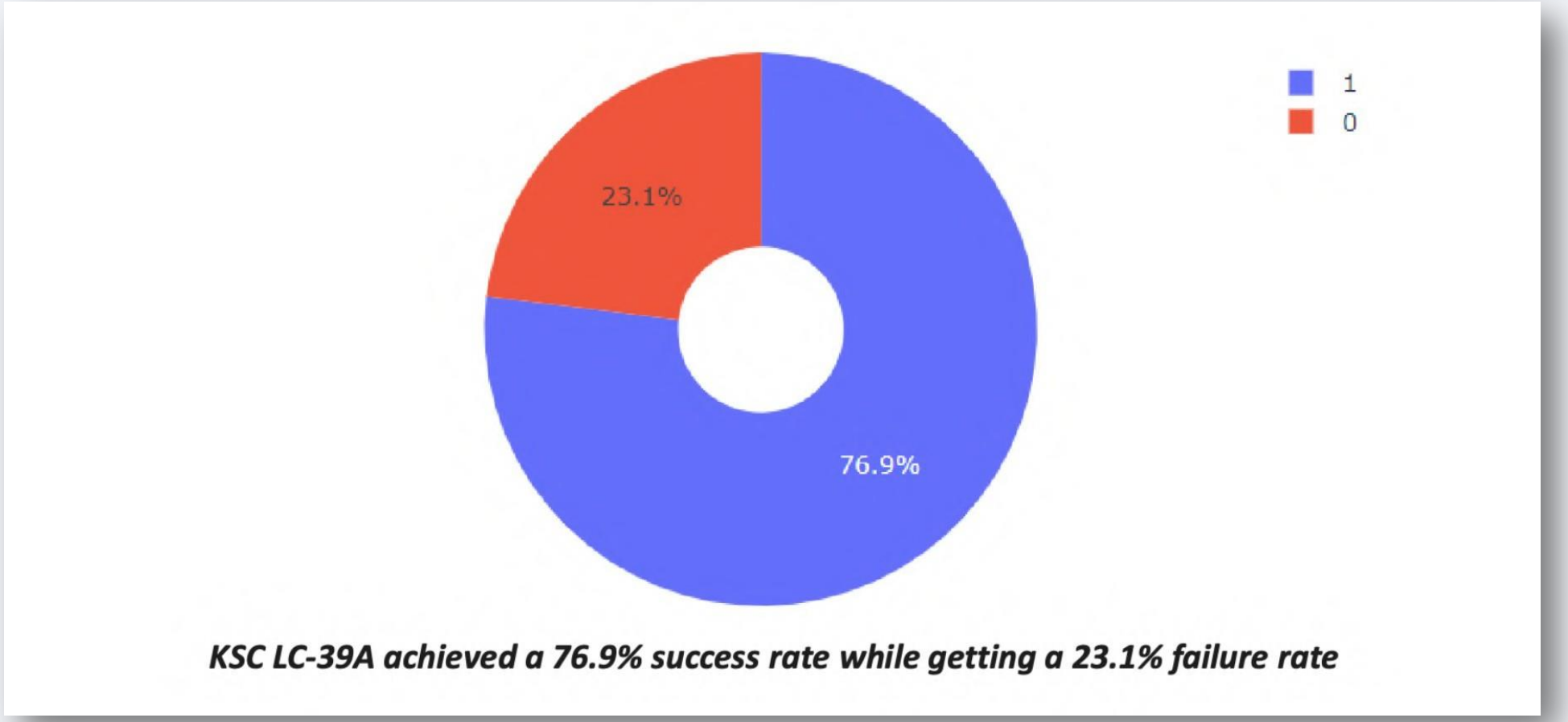
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes


Section 4
Build a Dashboard with Plotly Dash

The success percentage by each sites.



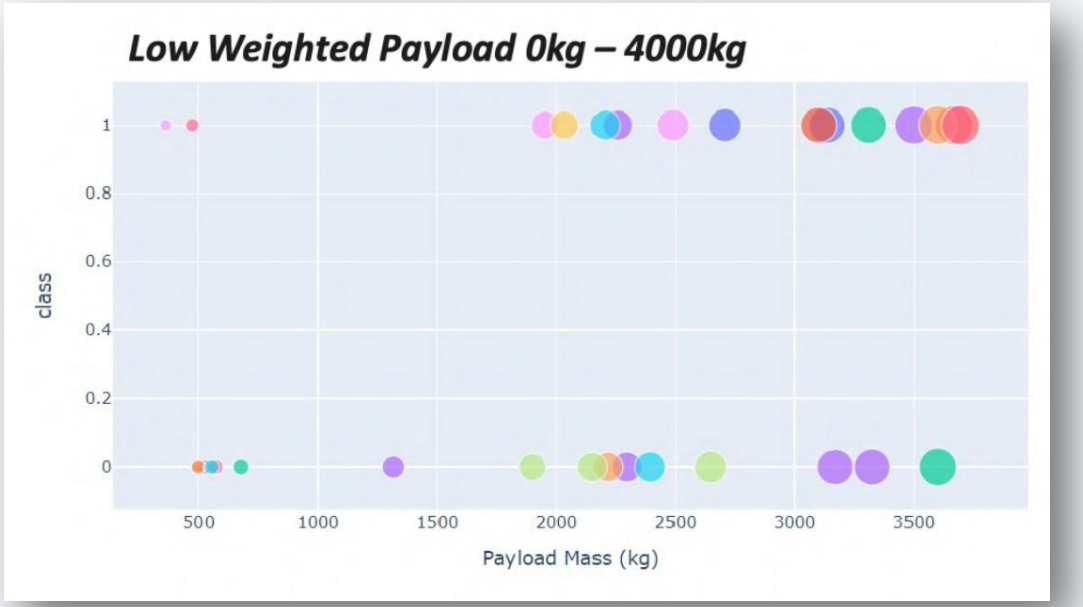
The highest launch-success ratio: KSC LC-39A

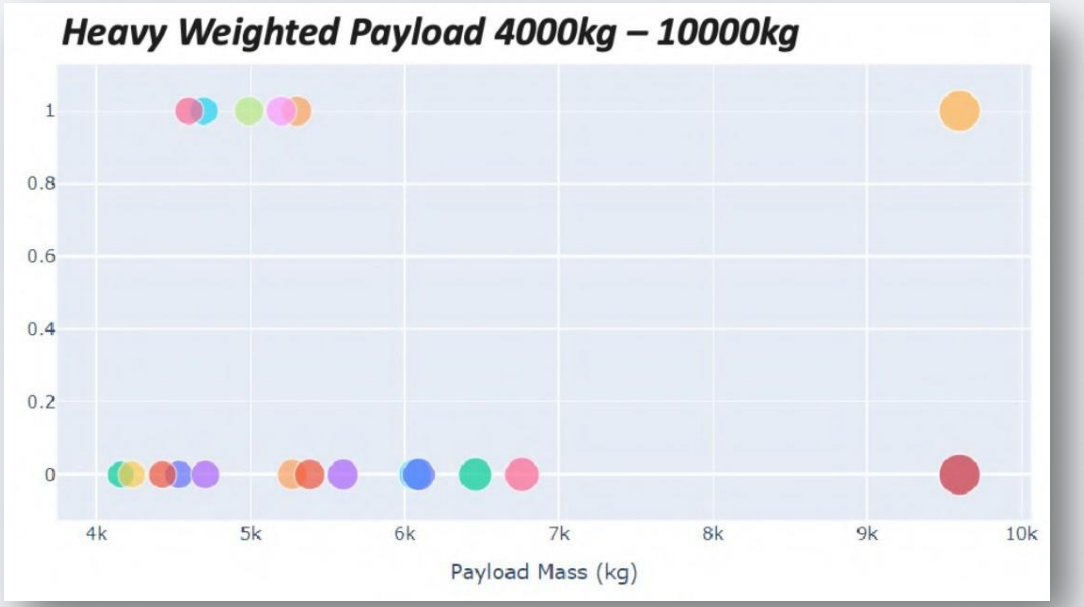


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

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.

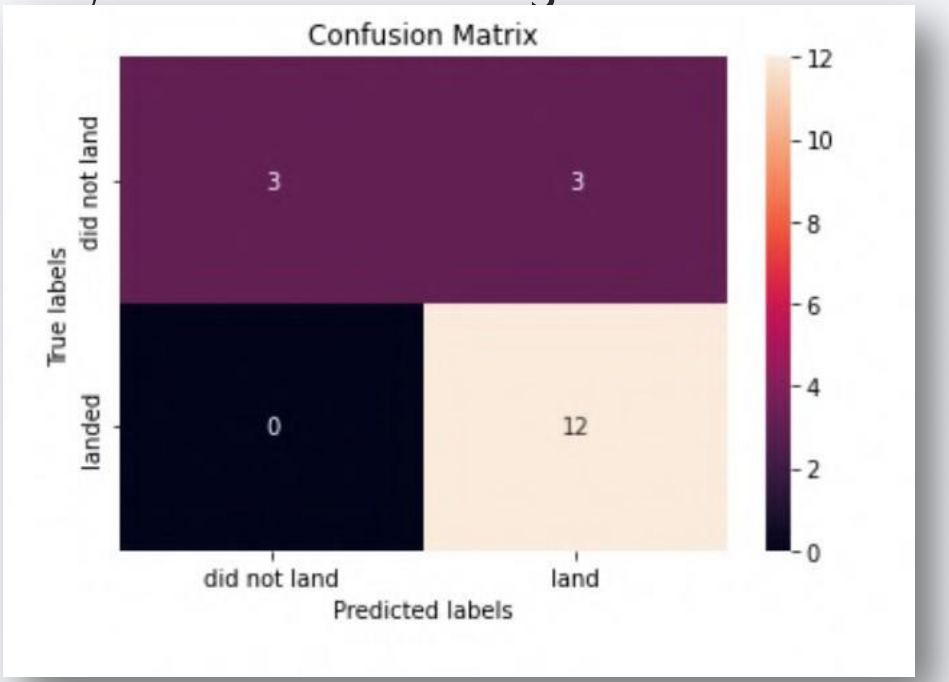
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params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

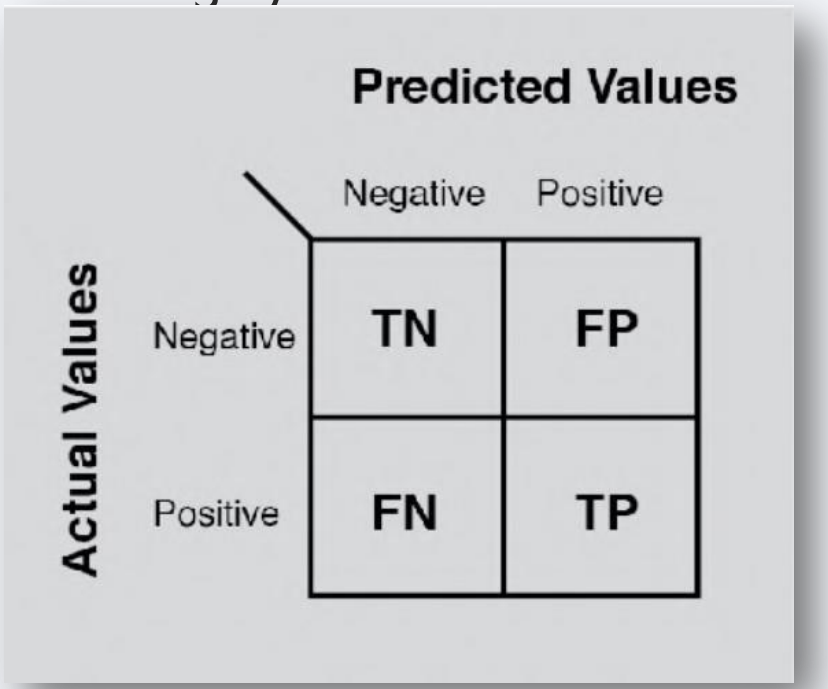
```
Best Algorithm is Tree with a score of 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'random'}
```

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.





Conclusions

From the presentation, we have the below findings:

- The Tree Classifier Algorithm is the best Machine Learning approach for this dataset.
- The low weighted payloads (which comes 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.
- KSC LC-39A have the most successful launches of any sites; 76.9%
- SSO orbit have the most success rate; 100% and more than 1 occurrence.

Thank you!