IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Fábio Soares de Lima
December 06, 2022

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 Results from Machine Learning Lab

Introduction

- SpaceX (Space Exploration Technologies Corporation) is a space transportation and aerospace manufacturer founded in 2002 by Elon Musk. Musk is also the CEO of electric car maker Tesla. And most recently, he entered a deal to purchase social networking site Twitter for $44 billion in April 2022. After 18 months of development, SpaceX unveiled a delivery vehicle in 2006 under the name Dragon. It was soon followed with Falcon, which was designed to lift humans and cargo into orbit.
- SpaceX has been a disruptive force in the worldwide launch industry as its launch services are less expensive than many of its competitors. It works closely with NASA to deliver cargo supplies and astronauts to the International Space Station (ISS), as well as to launch satellites into Earth orbit.
- One of the key differentiators from SpaceX and the NASA program is SpaceX's creation and use of reusable rocket launchers. Usually, NASA's rockets come in two or three stages. After the stage uses up its fuel, it is discarded and falls into the ocean.
- The main problems to solve:
    - Identifying all factor that influence the landing outcome
    - The relationship between each variables that affect the outcome
    - The best condition needed to increase the project successful.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API and web scrapping from Wikipedia were used to collect Data
- Perform data wrangling
  - One-hot Encoding for Categorical features was used to process data
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - The data was split into training and test data
  - SVM, Classification Trees and Logistic Regression were used to analyze

Data Collection

- Data collection is the process of collecting, measuring and analyzing different types of information using a set of standard validated techniques. The main objective of data collection is to gather information-rich and reliable data and analyze them to make critical business decisions. Once the data is collected, it goes through a rigorous process of data cleaning and data processing to make this data truly useful for businesses.

Data Collection – SpaceX API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```
✓ 0.1s

```
response = requests.get(spacex_url)
```
✓ 1.1s

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```
✓ 0.1s

```
# Lets take a subset of our dataframe keeping only the features we want and the flight r
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
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
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 da
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)]
```
✓ 0.1s

Get request for rocket launch data using API

Use json_normalize method to converto json result to dataframe

Performed data cleaning and filling the missing value

From: [fslima1980/courera: Applied Data Science Capstone (github.com)]

Data Collection - Scraping

```
# 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 text content

soup = BeautifulSoup(data,'html.parser')
```

✓ 3.1s

```
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
```

Request the Falcon9 Launch Wiki page from URL

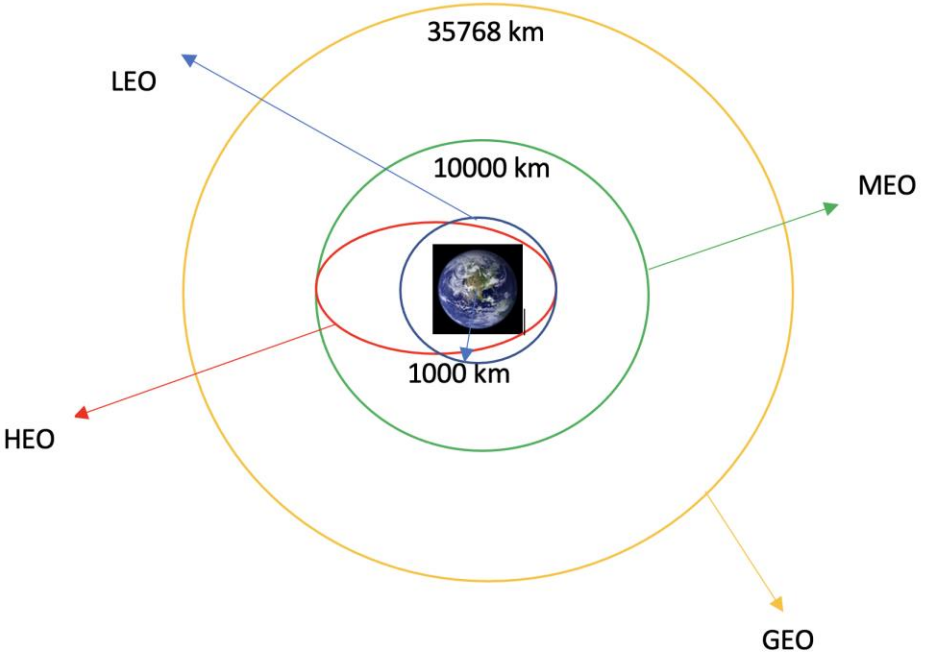
Create a BeatifulSoup from HTML response

Extractall column/variable names from the HTML header

From: [fslima1980/courera: Applied Data Science Capstone (github.com)]

Data Wrangling

Data wrangling is the process of transforming and mapping data from one "raw" data form into another format with the intent of making it more appropriate and valuable for a variety of downstream purposes such as analytics. The main objective is to assure quality and useful data. Data analysts typically spend most of their time in the process of data wrangling compared to the actual analysis of the data.

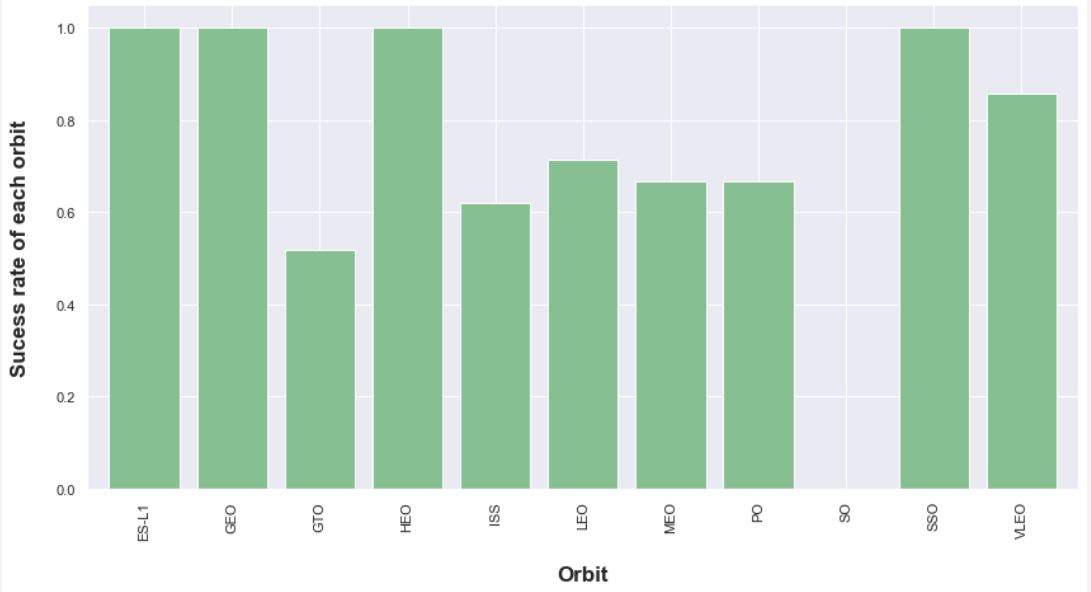


EDA with Data Visualization

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

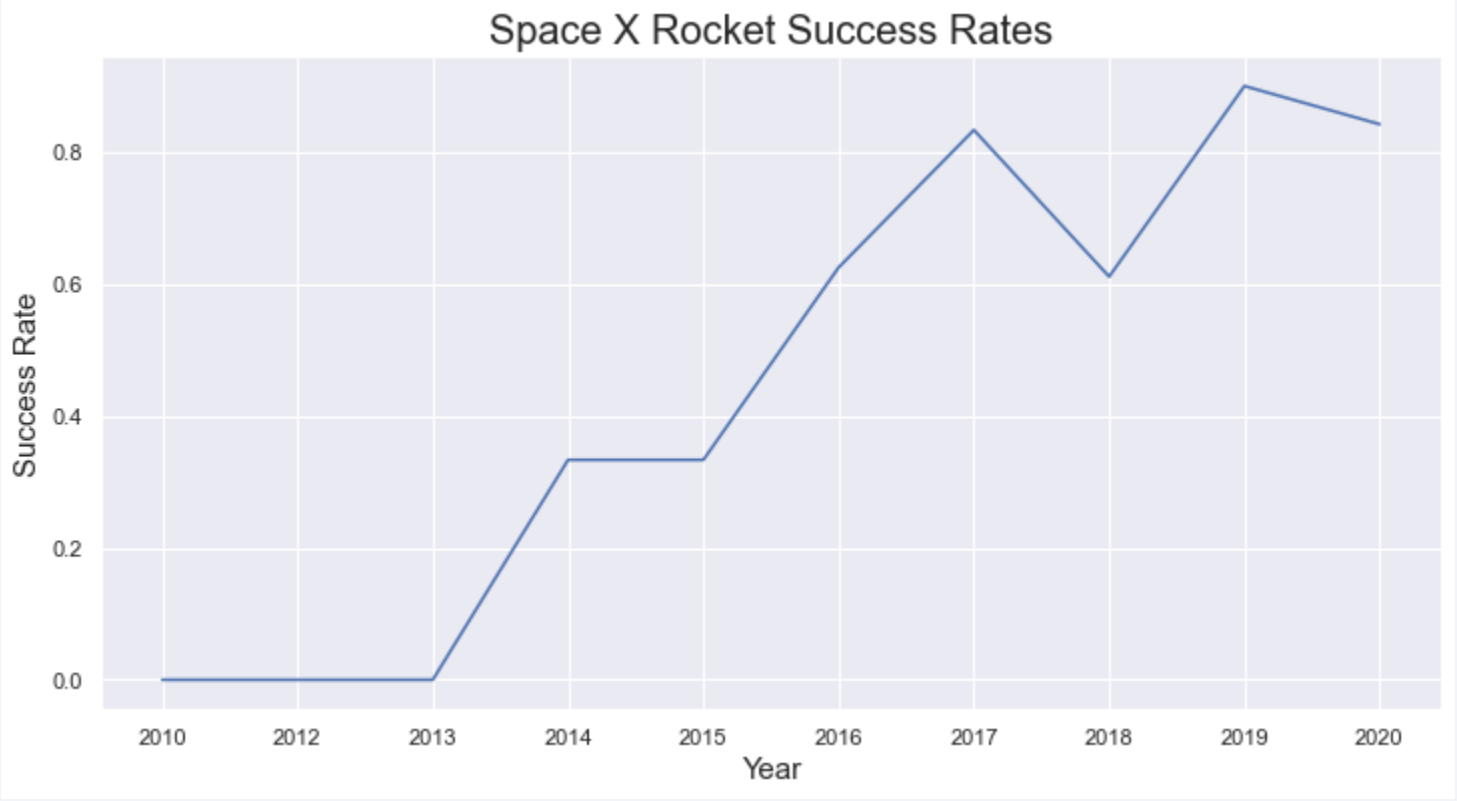
- Payload and Flight Number.
- Flight Number and Launch Site.
- Payload and Launch Site.
- Flight Number and Orbit Type.
- Payload and Orbit Type.

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



EDA with Data Visualization

- Once we get a hint of the relationships using scatter plot. We will then use further visualization tools such as bar graph and line plots graph for further analysis. 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. We then use Feature Engineering to be used in success prediction in the future module by created the dummy variables to categorical columns.



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 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 assigned the dataframe launch_outcomes(failures, successes) to classes 0 and 1 with Green and Red markers on the map in a MarkerCluster()
- Using Haversine's formula we calculated the distance from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns. Lines are drawn on the map to measure distance to landmarks
- Example of some trends in which the Launch Site is situated in.
    - 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

Build a Dashboard with Plotly Dash

- Used Python Anywhere to host the website live 24/7 so your can play around with the data and view the data
- The dashboard is built with Flask and Dash web framework.
- Graphs - Pie Chart showing the total launches by a certain site/all sites - display relative proportions of multiple classes of data. - size of the circle can be made proportional to the total quantity it represents
- Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions
    - It shows the relationship between two variables.
    - It is the best method to show you a non-linear pattern.
    - The range of data flow, i.e. maximum and minimum value, can be determined.
    - Observation and reading are straightforward.

Predictive Analysis (Classification)

Building the Model

- Load the dataset into NumPy and Pandas
- Transform the data and then split into training and test datasets
- Decide which type of ML to use
- Set the parameters and algorithms to GridSearchCV and fit it to dataset.

Evaluating the Model

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

Improving the Model

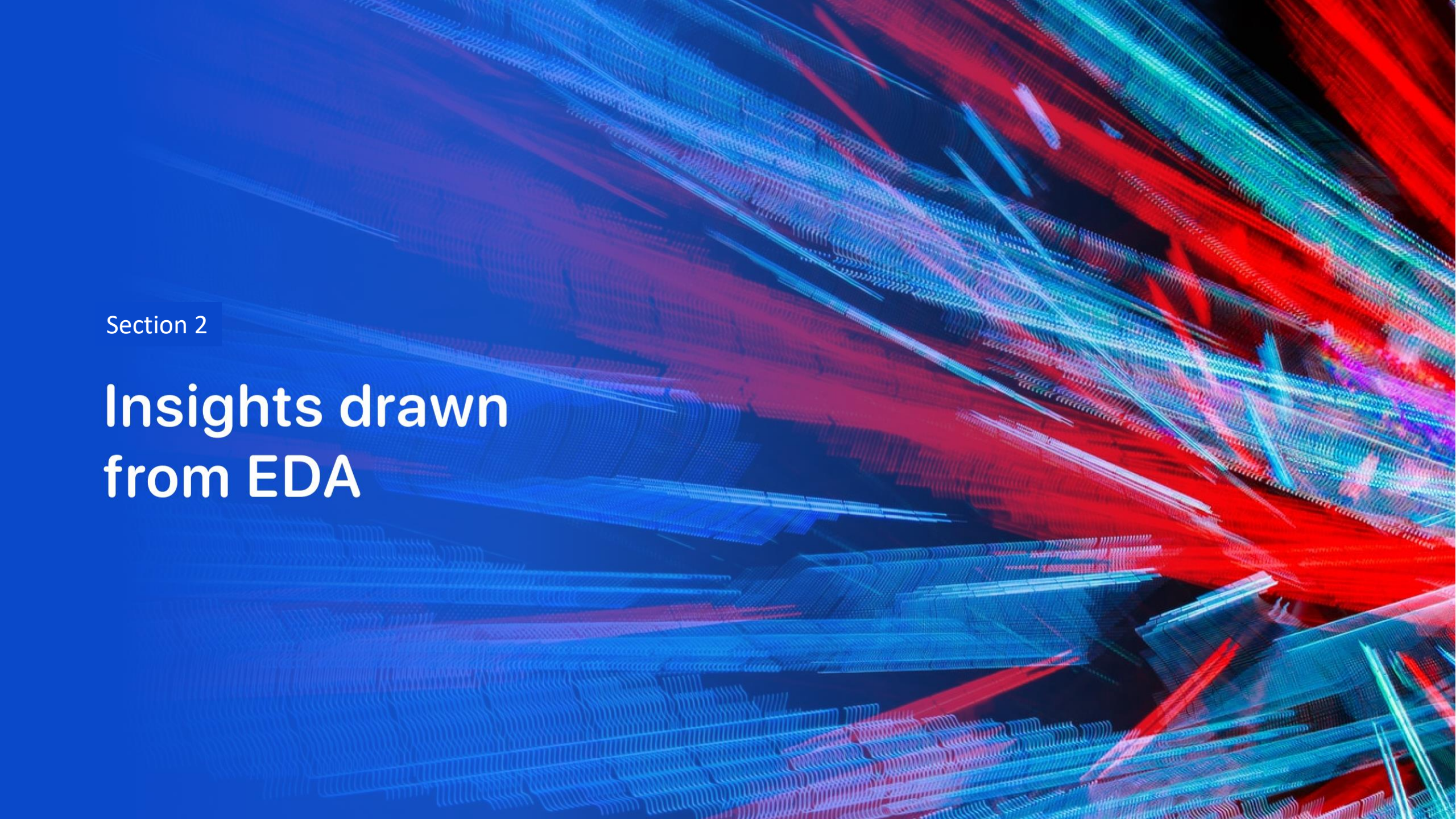
- Use Feature Engineering and Algorithm Tuning

Find the Best Model

- The model with the best accuracy score will be the best performing model.

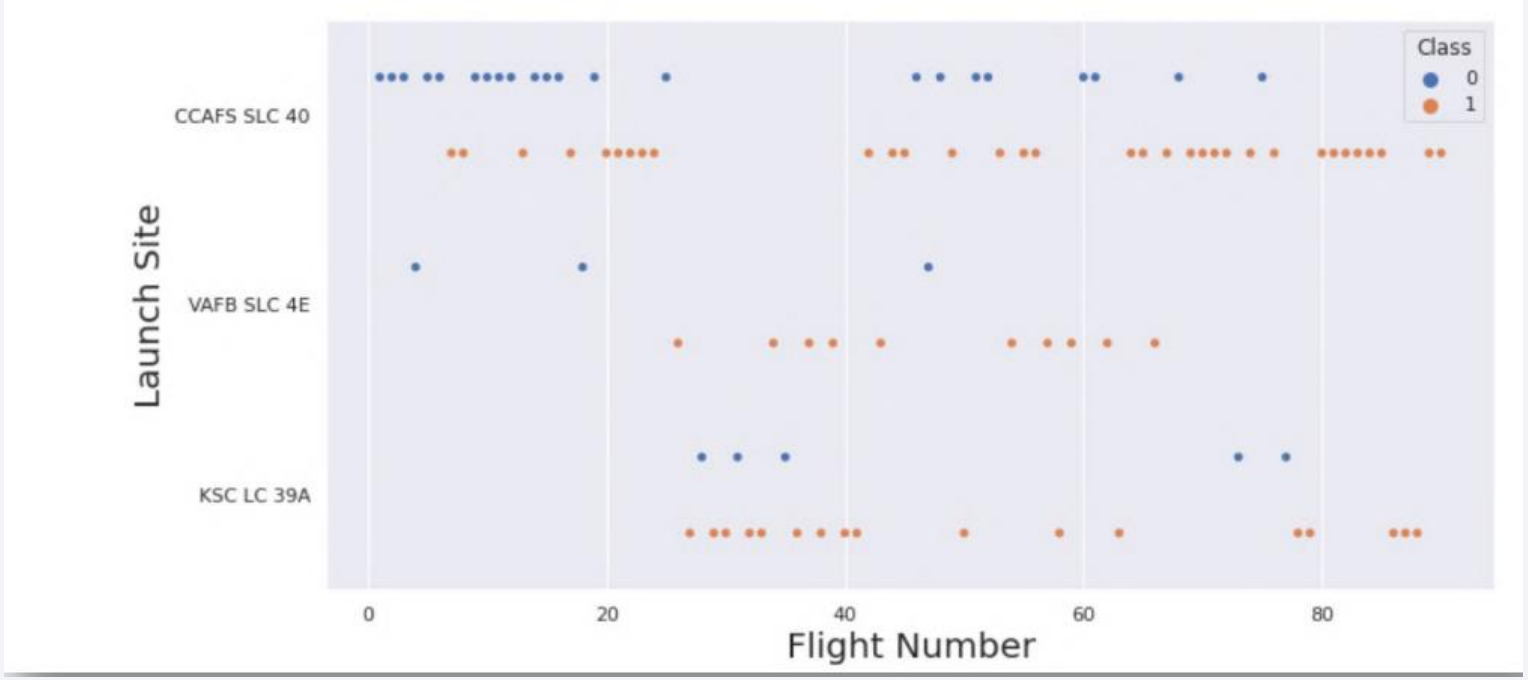
Results

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


Section 2
Insights drawn from EDA

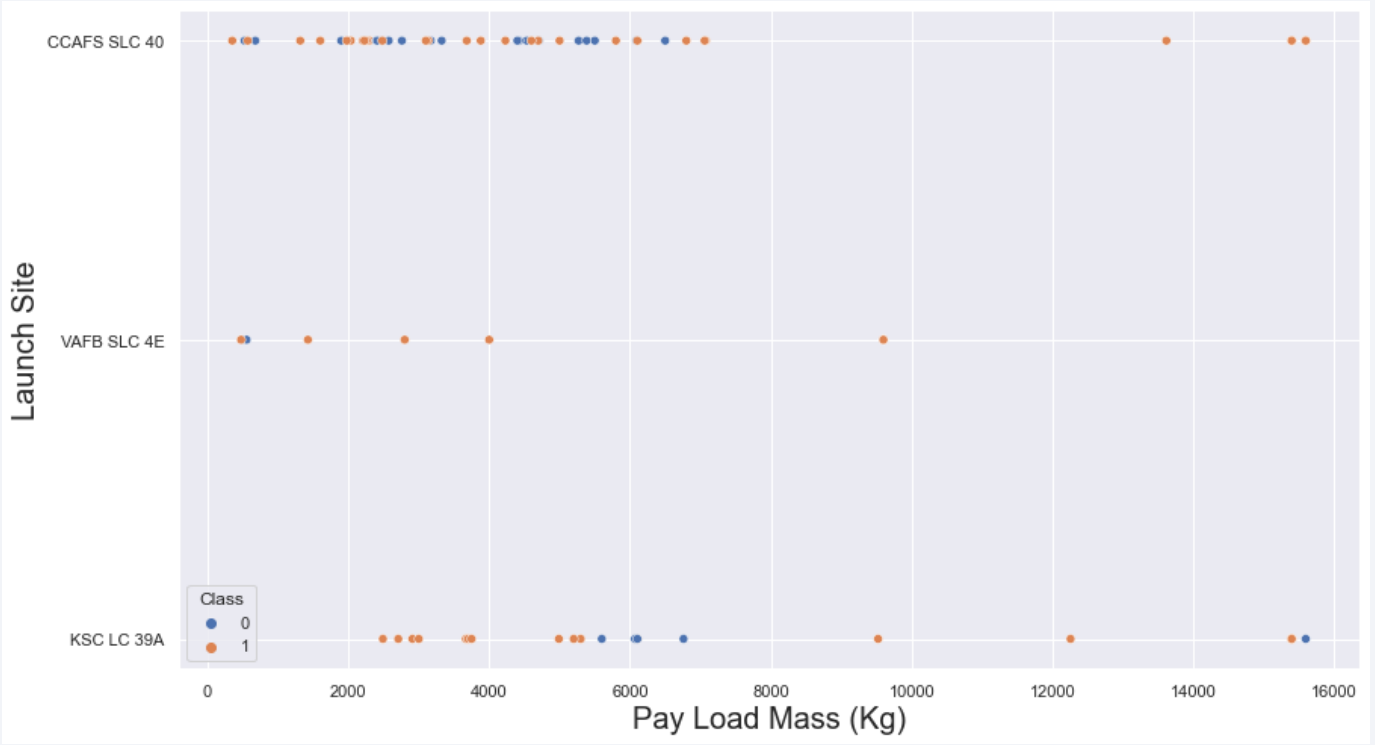
Flight Number vs. Launch Site

- This scatter plot shows 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.



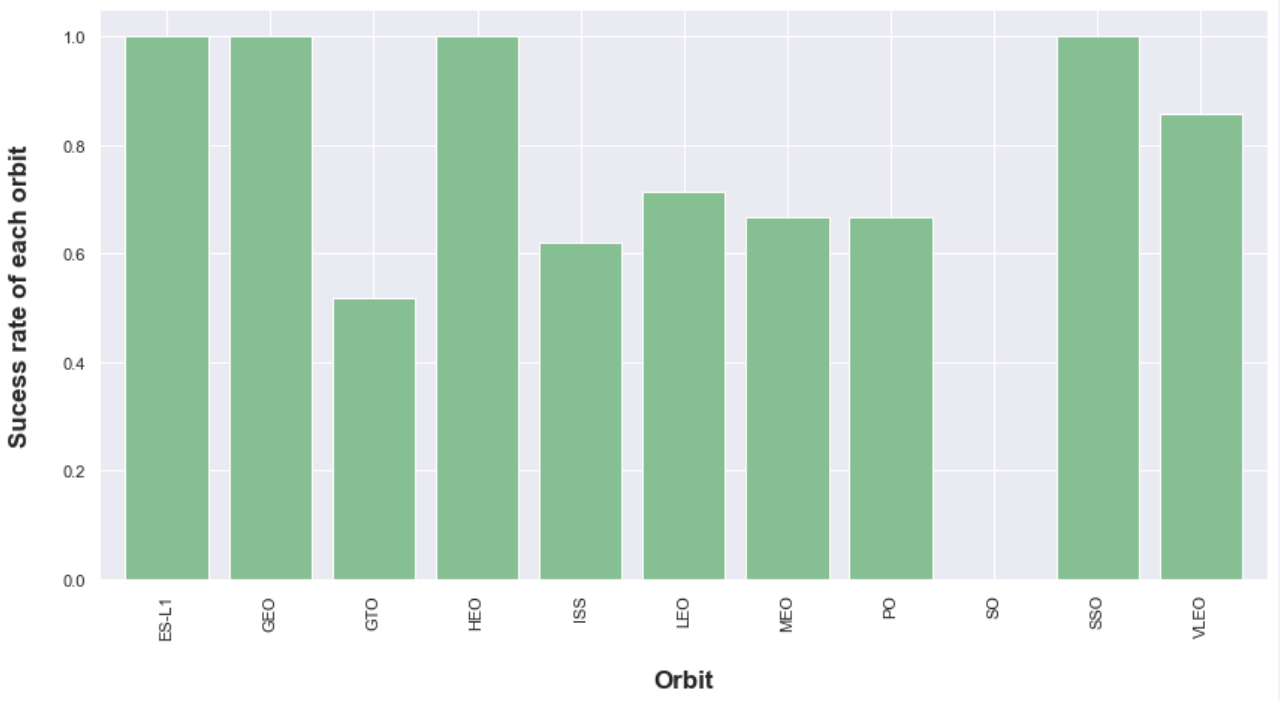
Payload vs. Launch Site

- The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket. There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependant on Pay Load Mass for a success launch.



Success Rate vs. Orbit Type

- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

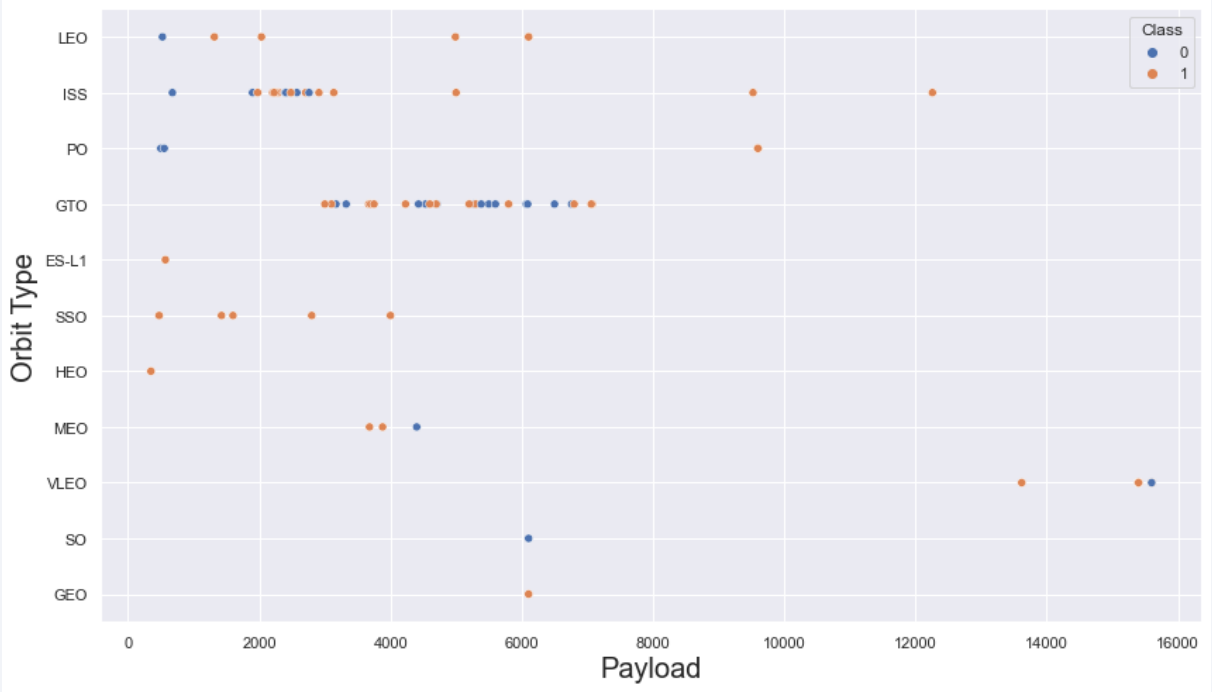


Flight Number vs. Orbit Type

- You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

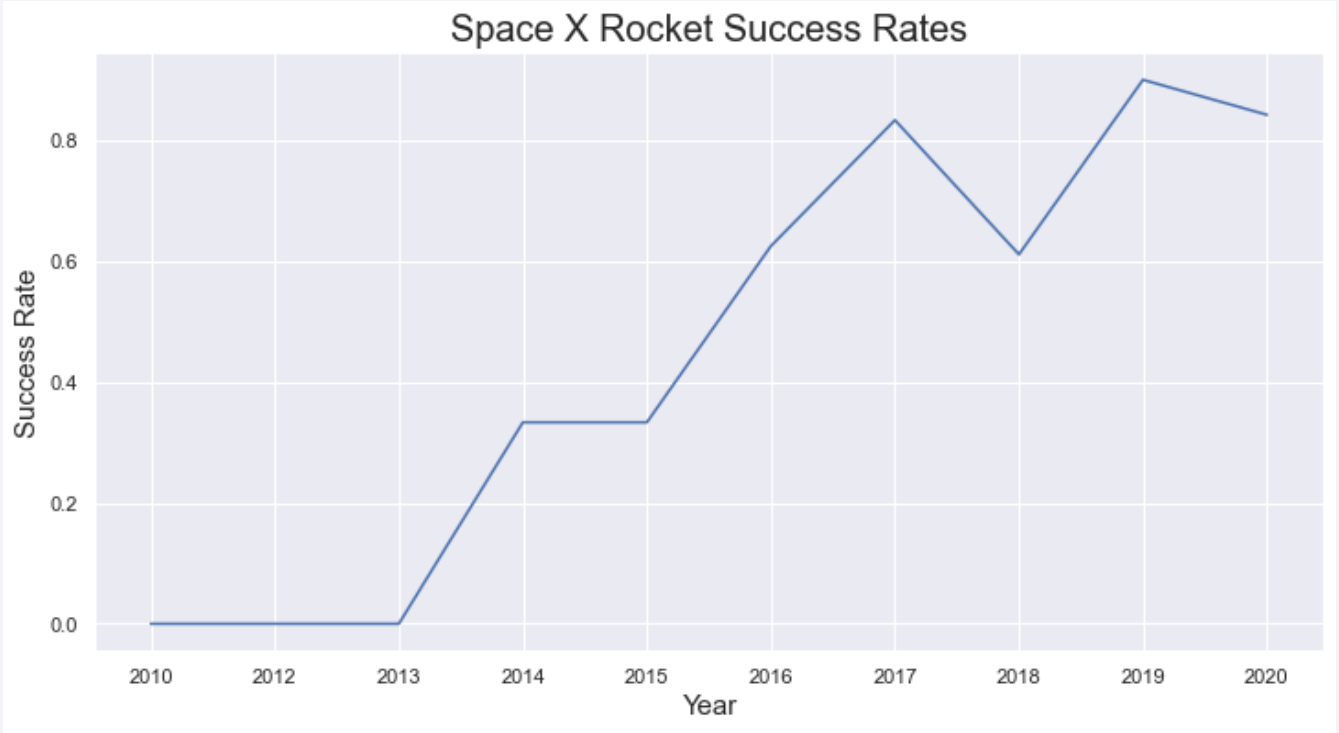
Payload vs. Orbit Type

- Heavier payload has positive impact on LEO, ISS and P0 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 figures clearly depicted and increasing trend from the year 2013 until 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 as "Launch_Sites" FROM SPACEXTBL ORDER BY Launch_site
✓ 0.1s
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

- 5 records where launch sites begin with ‘CCA’

```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

✓ 0.3s Python

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- the total payload carried by boosters from NASA

```
%sql SELECT SUM (PAYLOAD_MASS__kg_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)'
✓ 0.9s

* sqlite:///my_data1.db
Done.

SUM (PAYLOAD_MASS__kg_)
                  45596
```

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1'
✓ 0.2s

 * sqlite:///my_data1.db
Done.

 AVG(PAYLOAD_MASS__KG_)
                 2928.4
```

First Successful Ground Landing Date

- The dates of the first successful landing outcome on ground pad

```
%%sql
SELECT MIN(DATE) AS "First Landing"
FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Success (ground pad)'
```

✓ 0.1s

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

First Landing
22-12-2015

Successful Drone Ship Landing with Payload between 4000 and 6000

- The names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```
%%sql
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE "LANDING _OUTCOME" = 'Success (drone ship)'
AND PAYLOAD_MASS__KG_ > 4000
AND PAYLOAD_MASS__KG_ < 6000
```

✓ 0.1s

```
 * 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

- The total number of successful and failure mission outcomes

```
%%sql
SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "Successful Mission",
sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "Failure Mission"
FROM SPACEXTBL;
```

✓ 0.9s

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

Successful Mission	Failure Mission
100	1

Boosters Carried Maximum Payload

- The names of the booster which have carried the maximum payload mass

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

✓ 0.2s

```
* 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

- The failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
%%sql
SELECT BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE DATE LIKE '%2015'
AND "LANDING _OUTCOME" = 'Failure (drone ship)'
```

✓ 0.1s

```
 * 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

- 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

```
%%sql
SELECT "LANDING _OUTCOME" as "Landing Outcome", COUNT("LANDING _OUTCOME") AS "Total Count"
FROM SPACEXtBL
WHERE DATE BETWEEN '04/06/2010' AND '20/03/2017'
GROUP BY "Landing Outcome"
ORDER BY "Total Count"  DESC
```

✓ 0.8s

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

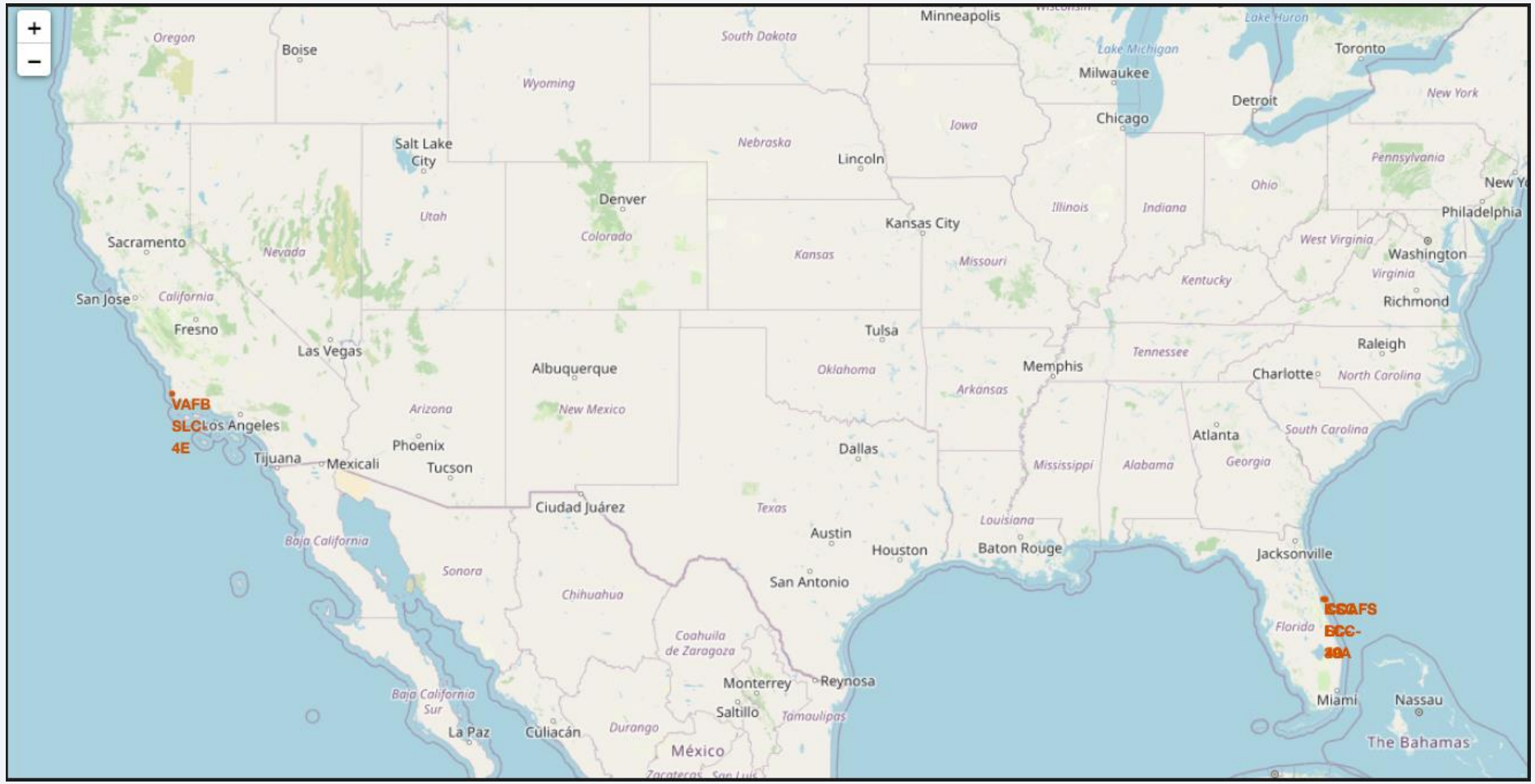
Landing Outcome	Total Count
Success	20
No attempt	9
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
No attempt	1
Failure (parachute)	1

Section 3

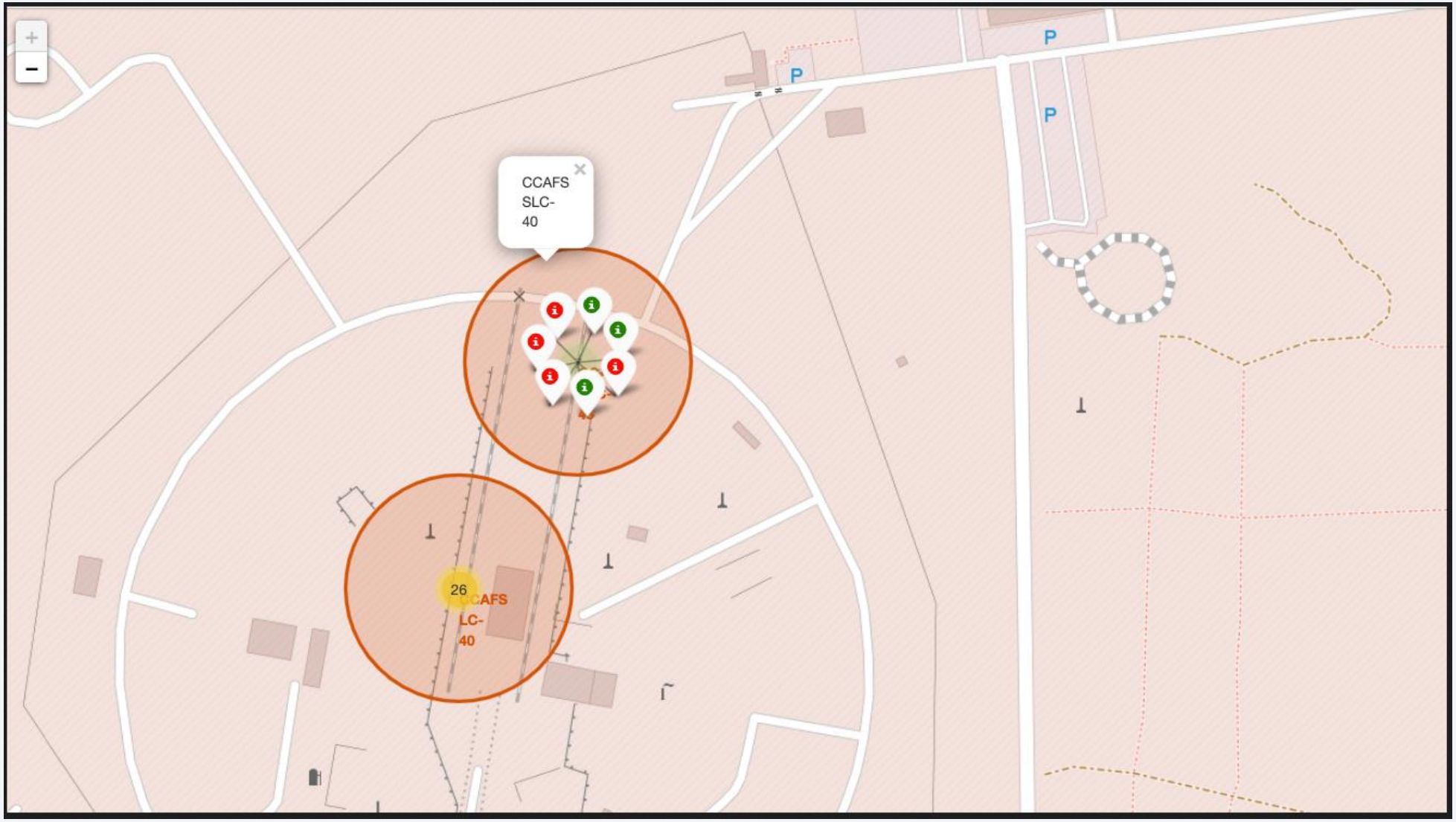
Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

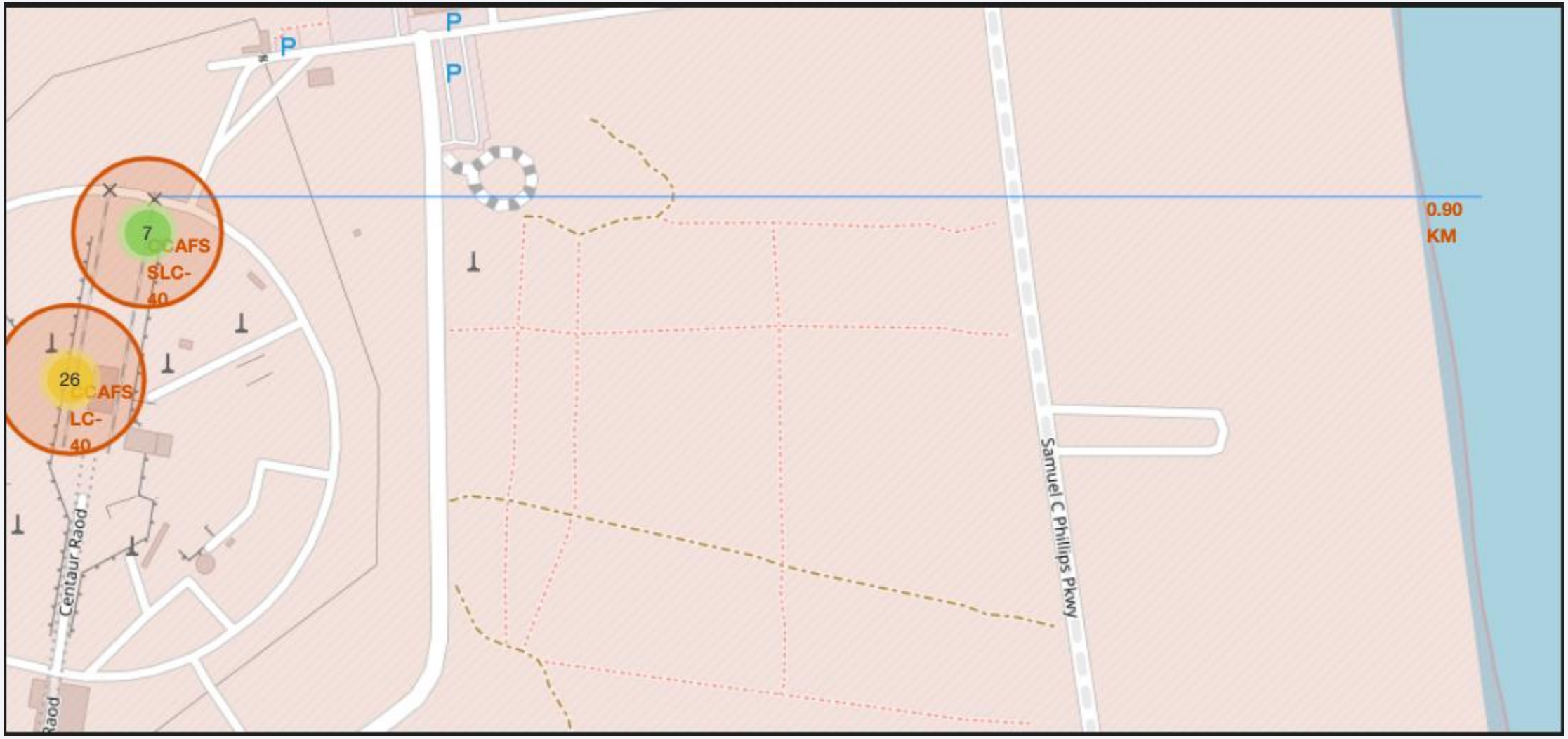
- Here, it shows the SpaceX launch sites that locate inside the United States

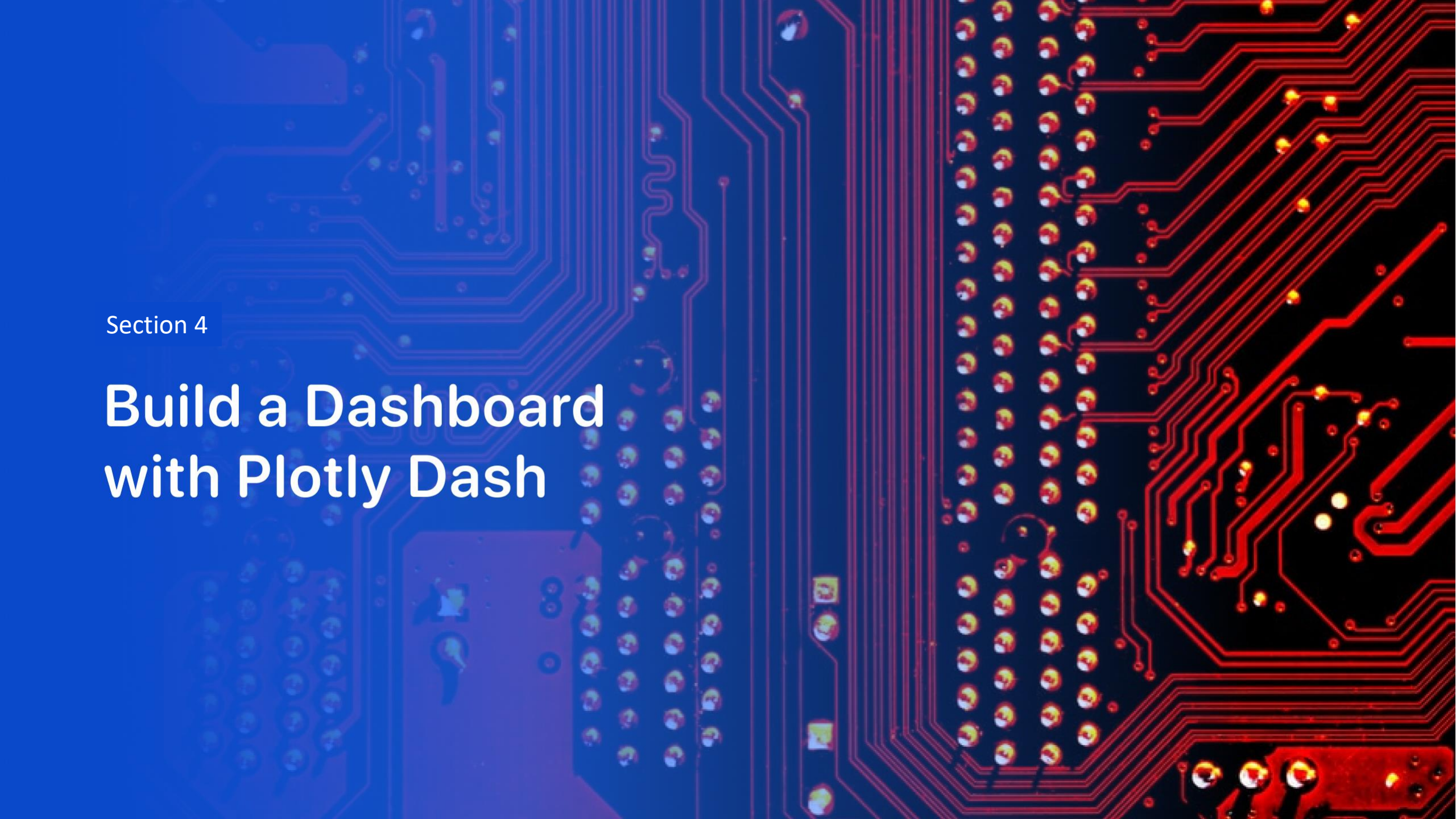


Colour Label Marker

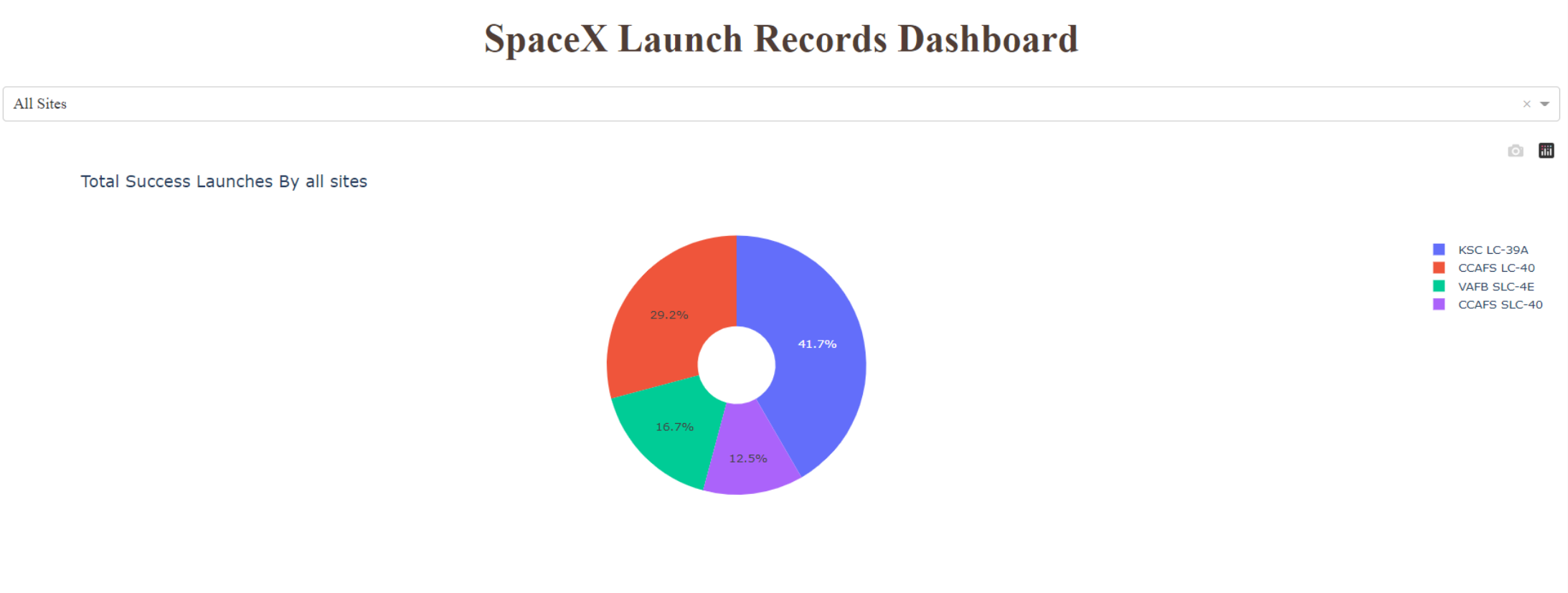


Launch sites Distances

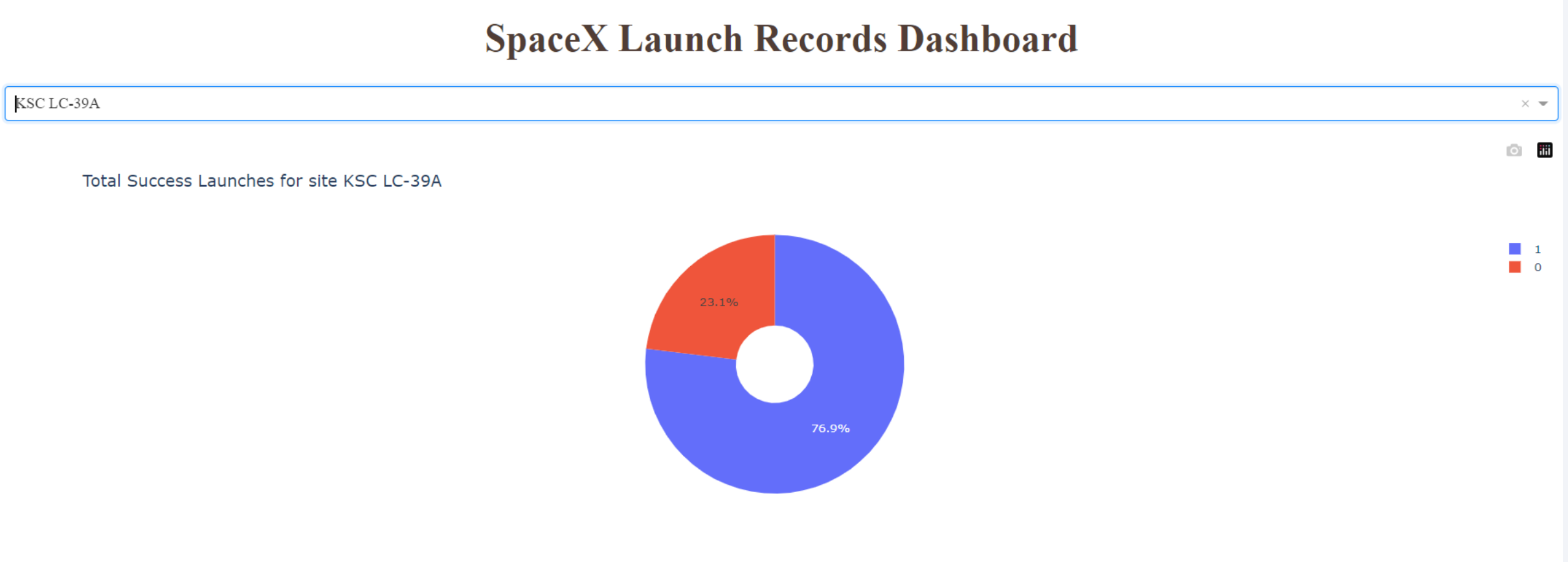



Section 4
Build a Dashboard with Plotly Dash

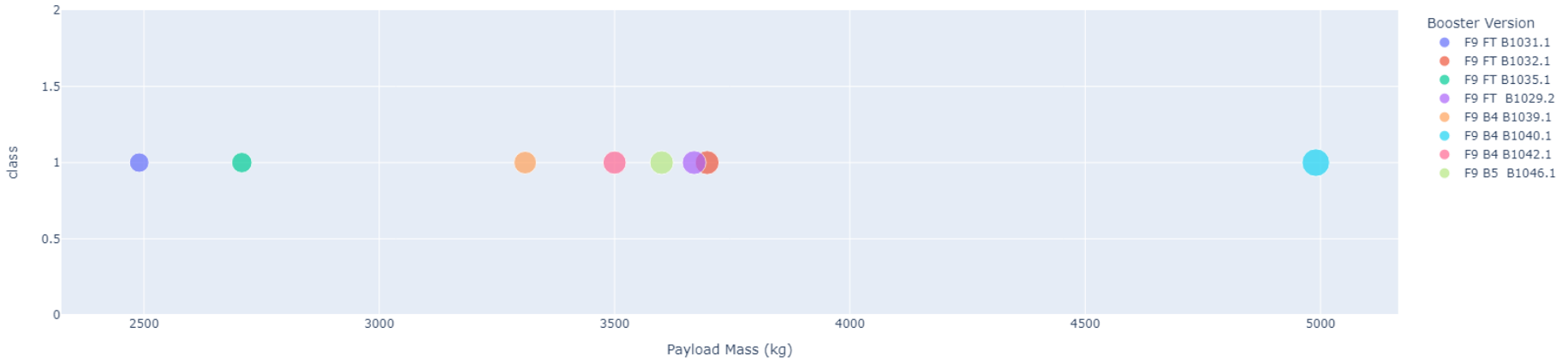
SpaceX Launch Records Dashboard

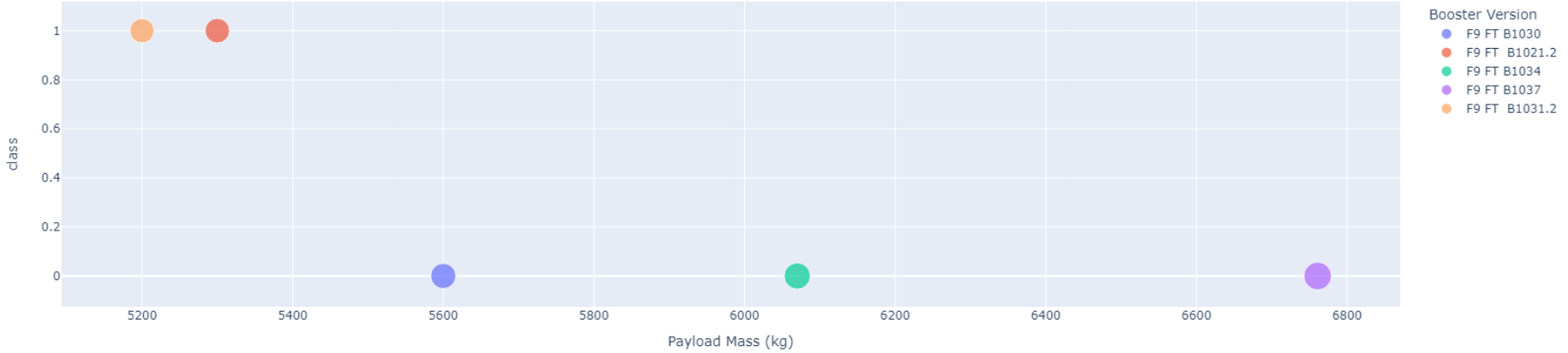


The highest launch success



Payload vs Launch Outcome





Section 5

Predictive Analysis (Classification)

Classification Accuracy

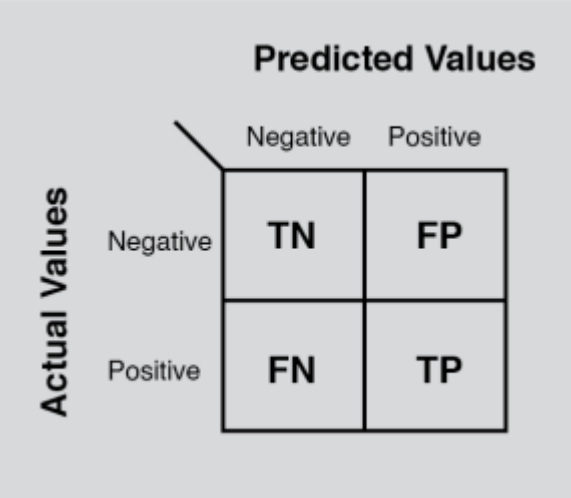
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params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

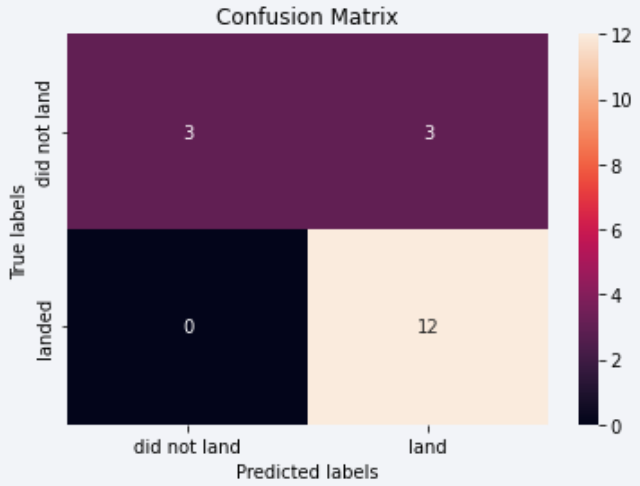
✓ 0.6s

```
Best Algorithm is Tree with a score of 0.8857142857142856
Best Params is : {'criterion': 'gini', 'max_depth': 2, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 5, 'splitter': 'best'}
```

Confusion Matrix

- The confusion matrix of the best performing model. The major problem is the false positives.





Conclusions

- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
- Low weighted payloads perform better than the heavier payloads
- We can see that KSC LC-39A had the most successful launches from all the sites
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate
- The Tree Classifier Algorithm is the best for Machine Learning for this dataset

Appendix

- fslima1980/courera: Applied Data Science Capstone (github.com)

Thank you!