IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Krishnanand S Pai
20th August 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 and Plotly
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result from Machine Learning Lab

Introduction

Space X is a revolutionary company that disrupted the space industry through its low-cost rocket launches specifically its Falcon 9. Most of these savings were achieved by reusing their rockets. Their achievement has seen many failures and successes. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch

- Through this project, we will find answers to :
  - Identifying all factors that influence the landing outcome
  - The relationship between each variable and how it is affecting the outcome.
  - The best condition needed to increase the probability of a successful landing

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - The data was collected using the SpaceX REST API and web scraping from Wikipedia
- Perform data wrangling
  - Data was processed using OneHotEncoding 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

- Describe how data sets were collected
  - REST API – the data set was collected using the SpaceX REST API using the GET request for rocket launch data. As the result was a json file, we then normalized the data and converted into a DataFrame to do our analysis
  - Web Scraping – this data set was collected by web scraping using the library BeautifulSoup, parsing the html table and then converting it to a DataFrame

Data Collection - SpaceX API

```
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 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 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.head()
```

Initiate GET request from the SpaceX REST API

Normalize the JSON response and create a DataFrame

Perform data cleansing and fill the missing values

[Github - Data Collection API notebook]

Data Collection - Scraping

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

Create a `BeautifulSoup` object from the HTML `response`

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

```python
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

```python
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if (name != None and len(name) > 0):
        column_names.append(name)
```

Initiate GET request from the URL

Use BeautifulSoup & html parser to parse the data

Use <table> and <tr> elements to iterate through the html table and create a DataFrame

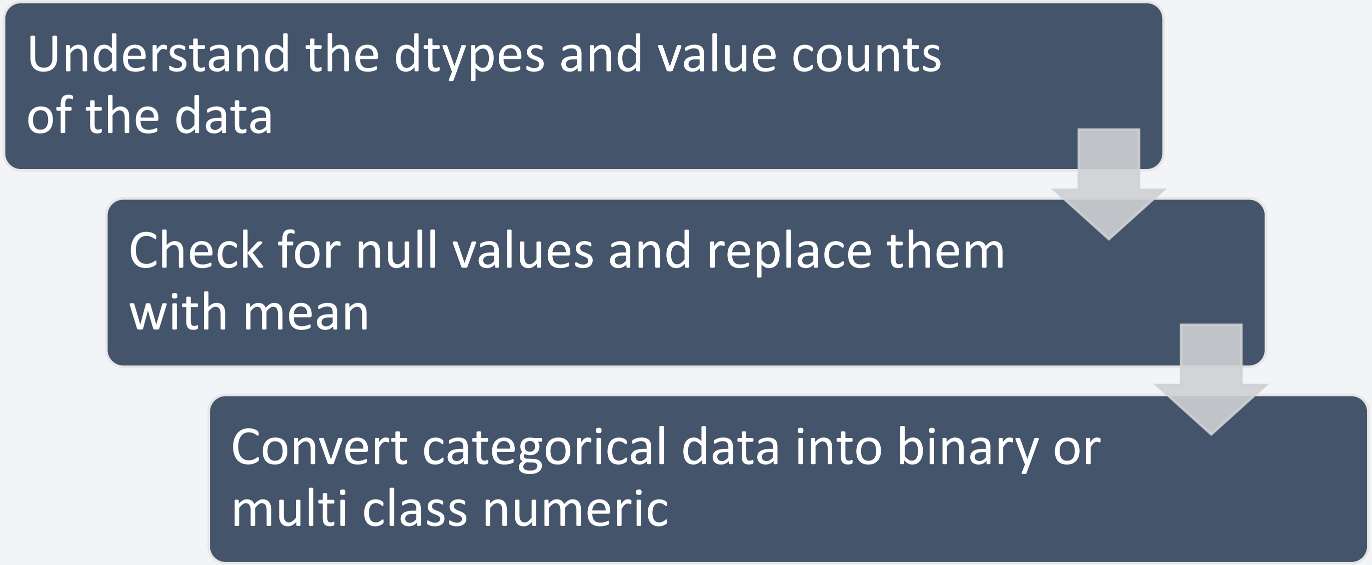
[GITHUB – Data Collection - Scraping]

Data Wrangling

The data was processed using multiple checks and transformations:

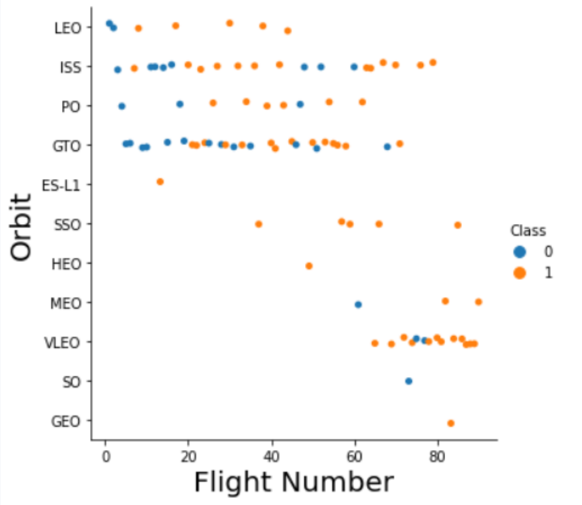
- If there are any null values present
- For any null values, they were replaced with mean values
- Classify multiple outcomes into good (1) or a bad(0) outcome

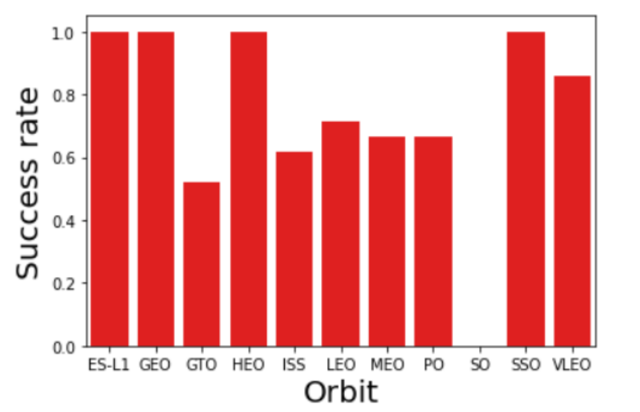
[GITHUB – Data Wrangling Notebook]

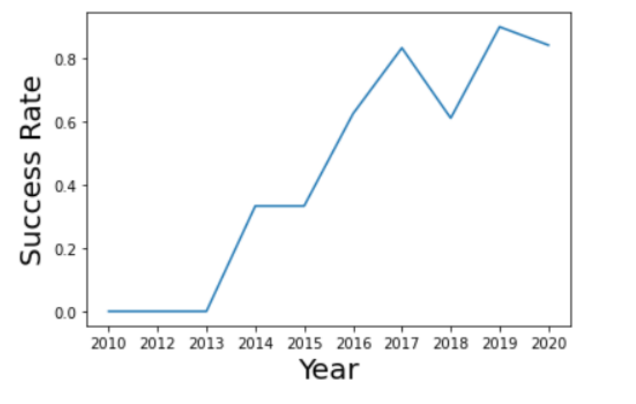


EDA with Data Visualization

- Multiple charts were used to plot the data:
  - Scatter plots: were utilized to understand the relationship between two continuous variables and if there was any correlation between the two
    - Payload vs Flight Number
    - Flight Number vs Launch Site
    - Payload vs Launch Site
    - Flight number vs Orbit
    - Payload Mass vs Orbit
  - Bar charts: were utilized to understand the relationship between a categorical variable vs a continuous variable
    - Orbit vs Class
  - Line plot: was utilized to understand if there was any trend in two continuous variables
    - Orbit vs success rate







[GITHUB – EDA with Data Viz Notebook]

EDA with SQL

Using SQL, queries were performed on the table to summarize and get insights from the datasets:

1. Display the names of the unique launch sites in the space mission
2. Display 5 records where launch sites begin with the string 'CCA
3. Display the total payload mass carried by boosters launched by NASA (CRS)
4. Display average payload mass carried by booster version F9 v1.1
5. List the date when the first succesful landing outcome in ground pad was acheived.¶
6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7. List the total number of successful and failure mission outcomes
8. List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
9. List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015
10. Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

GITHUB – EDA WITH SQL Notebook

Build an Interactive Map with Folium

- Multiple map objects were created using Folium and MarkerCluster
  - Circles - were created to showcase a location on the map
  - Lines - were created to showcase distances between two locations
  - Markers - were created to label the location on the map
  - MarkerCluster - was utilized to create multiple markers on the map

GITHUB – Interactive Map Notebook

Build a Dashboard with Plotly Dash

- Using Plotly Dash, an html web page was created with the following features:
  - Dropdowns – to select the various sites (all or individual) so that the user can view the charts for the particular site
  - Pie Chart – to view the success rate of the launches at these sites
  - Scatter Plot – to view the number of launches and their outcomes (success or failure)
- These plots and interactions were added to get visual insights from it

GITHUB – SPACE DASH APP - Notebook

Predictive Analysis (Classification)

Building the model

- Loading the data sets
- Convert the dataset into numeric using numpy
- Split the data into independent and target columns
- Normalize the data using StandardScaler()
- Split the data into train and test data sets
- Set the parameters and use GridSearch to iterate across the parameters to fit the model

Evaluate the model

- Check the accuracy of the model using the test sets
- Use confusion matrix to check for false positives
- Get the hyper parameters for each of the model

Improving the model

- Use feature engineering to add / subtract more features

Find the best model

- Use the accuracy score of all the models to find the best model

GITHUB – Predictive analysis Notebook

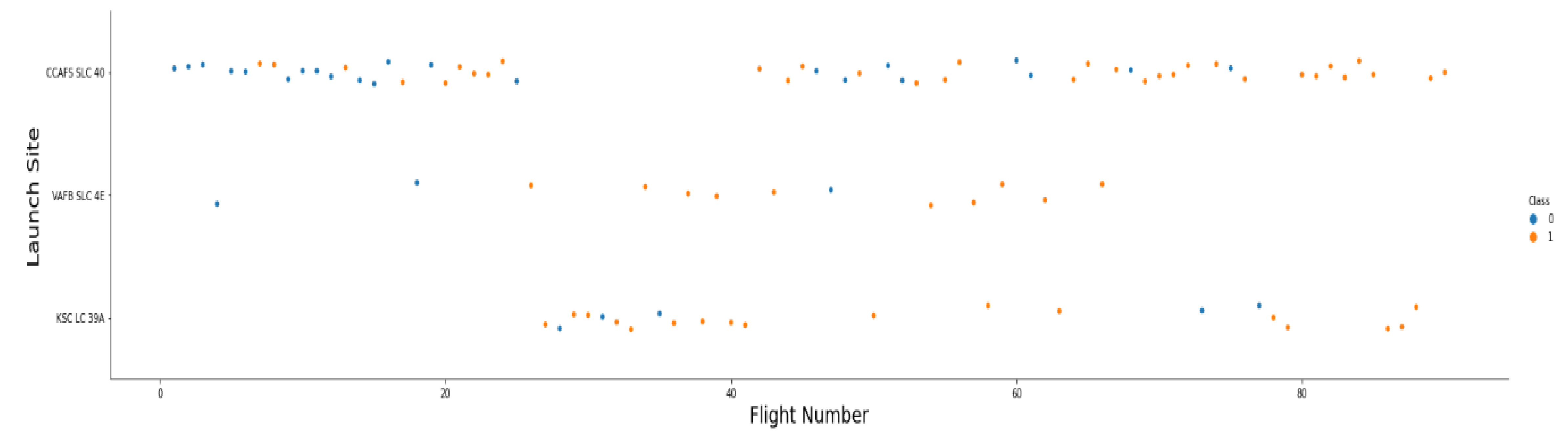
Results

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

Section 2

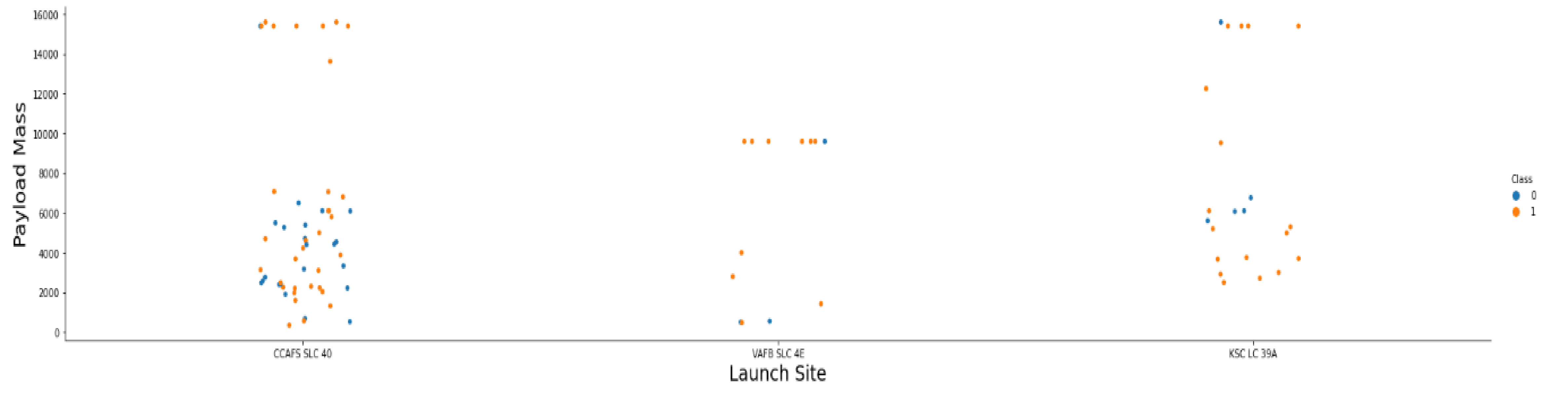
Insights drawn from EDA

Flight Number vs. Launch Site



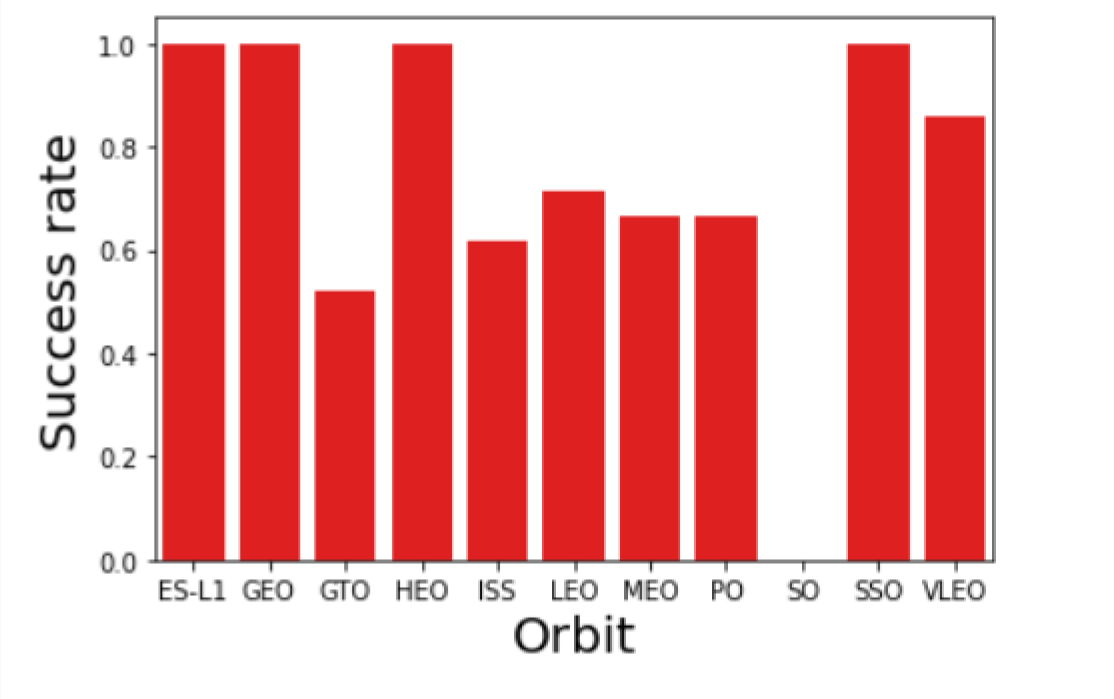
- As SpaceX conducted more flights their success rate improved
- There is a higher success rate in VAFB SLC 4E

Payload vs. Launch Site



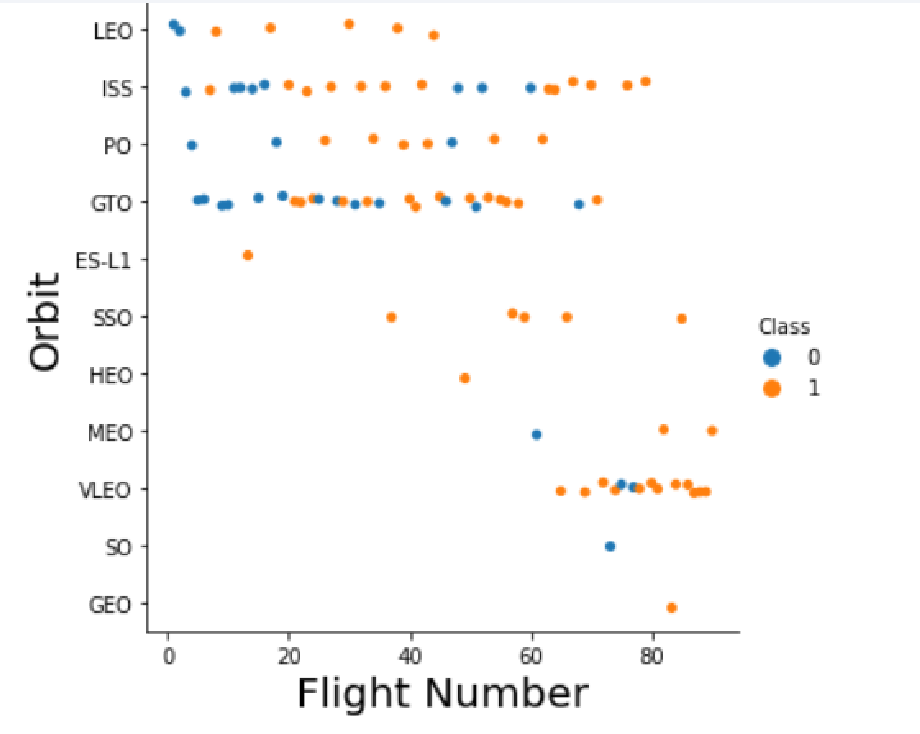
- There are no rockets with payload beyond 10,000kg in the launch site VABF SLC 4E
- Payloads of 10,000kg and higher have a higher success rate

Success Rate vs. Orbit Type



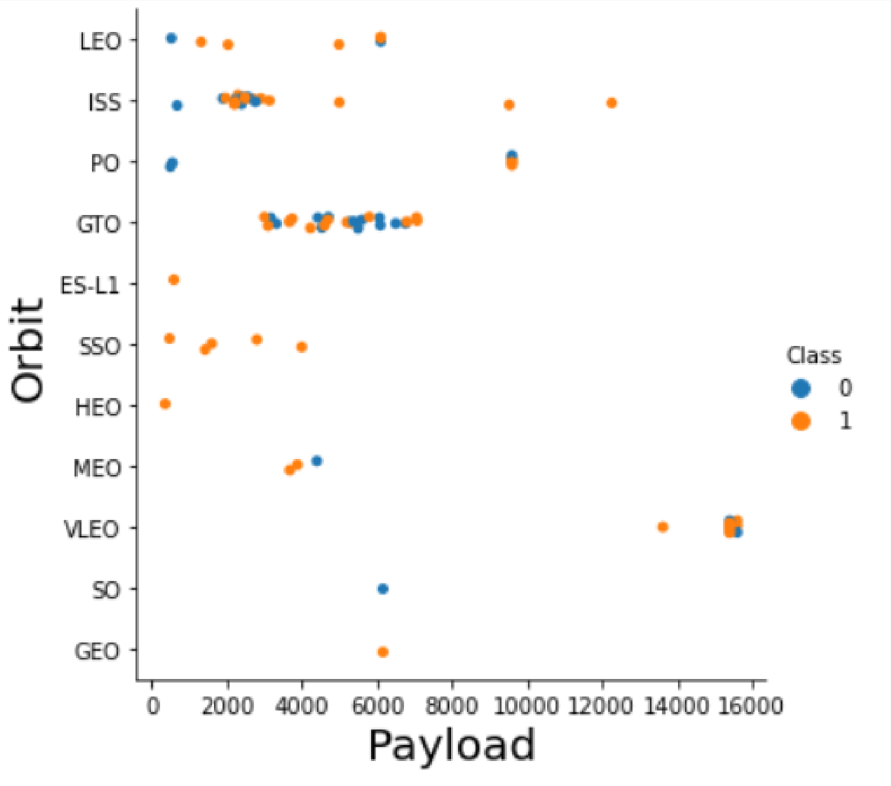
- Orbits that have a 100% success rate:
  - ES – L1
  - GEO
  - HEO
  - SSO

Flight Number vs. Orbit Type



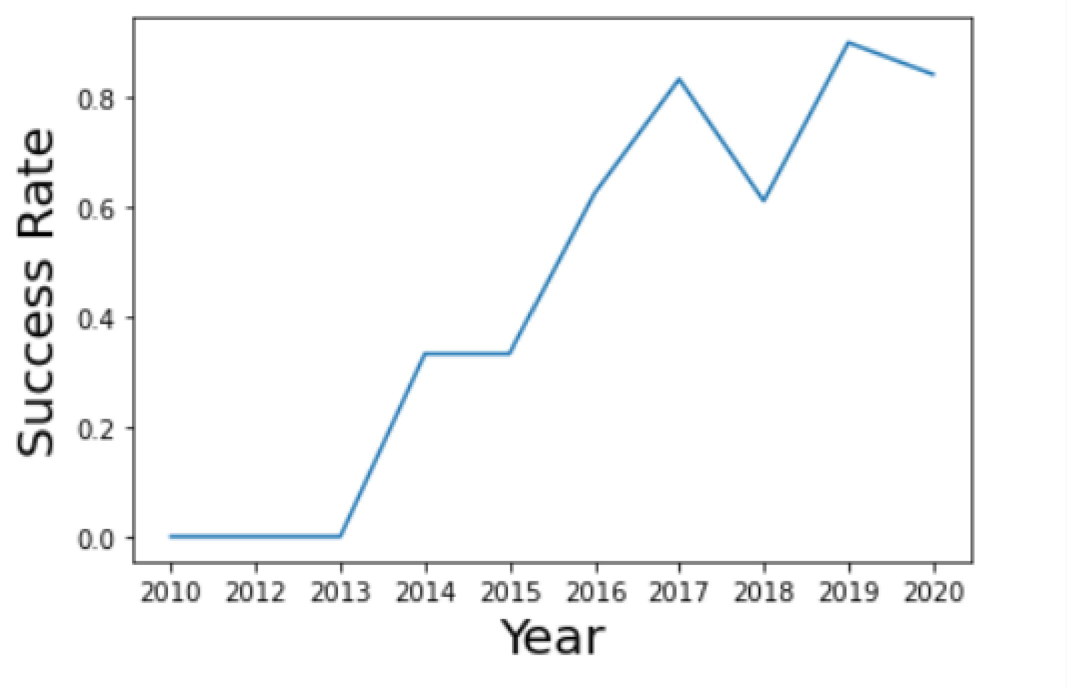
- After 80 flights, SpaceX found 100% success rate across all the orbits that they launched in

Payload vs. Orbit Type



- ESL1, SSO, HEO have a 100% success rate for low payloads
- For higher payloads >= 10,000 kg, ISS has the best chance of success

Launch Success Yearly Trend



- The year 2019 had the highest success rate among all the years
- There is a trend of increasing success rate as we move along the years

All Launch Site Names

Task 1

Display the names of the unique launch sites in the space mission

In [8]:
```
%sql SELECT DISTINCT (LAUNCH_SITE) from SPACEXTBL
```

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

Out[8]:

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

- SpaceX has 4 unique launch sites across the USA

Launch Site Names Begin with 'CCA'

Task 2

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

```
%sql SELECT * from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
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

- Since we have used LIKE %CCA & limit , it will select the first 5 records
- We will not be able to see the other Launch site with starting CCA

Total Payload Mass

Task 3

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER == 'NASA (CRS)';
```

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

SUM(PAYLOAD_MASS__KG_)
45596

- The total mass carried by boosters launched by NASA is 45,596kg
- SUM has been utilized to sum the total of the Payload columns where customer is NASA

Average Payload Mass by F9 v1.1

Task 4

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

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

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

AVG(PAYLOAD_MASS__KG_)
2928.4

- The average mass for booster F9 v1.1 is 2928.4kg
- The AVG method was used for the query

First Successful Ground Landing Date

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

Hint:Use min function

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

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

MIN(DATE)
01-05-2017

- The first successful ground pad launch was in May, 2017

Successful Drone Ship Landing with 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

```
%sql SELECT Booster_Version FROM SPACEXTBL WHERE "Landing _Outcome" == 'Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000
```

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

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

- There are 4 F9 booster versions which landed successfully on a drone ship with a payload between 4K & 6K tons

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
]: %sql SELECT COUNT(MISSION_OUTCOME) as missionoutcomes FROM SPACEXTBL GROUP BY MISSION_OUTCOME
```

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

]:

missionoutcomes
1
98
1
1

- There were 98 total successful mission outcomes

Boosters Carried Maximum Payload

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

```
%sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ == (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

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

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

- There are 12 booster versions that have carried that maximum payload

2015 Launch Records

```
%sql SELECT substr(DATE,4,2) as MONTH, MISSION_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE "Landing _Outcome" == 'Failure (drone ship
```

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

MONTH	Mission_Outcome	Booster_Version	Launch_Site
01	Success	F9 v1.1 B1012	CCAFS LC-40
04	Success	F9 v1.1 B1015	CCAFS LC-40

- There were 2 missions in 2015 where there was a failure to land on a drone ship in January and April

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

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT "Landing _Outcome", COUNT("Landing _Outcome") as Rank FROM SPACEXTBL WHERE DATE BETWEEN '04-06-2010' AND '20-03-2017' AND "Landing _Outc
```

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

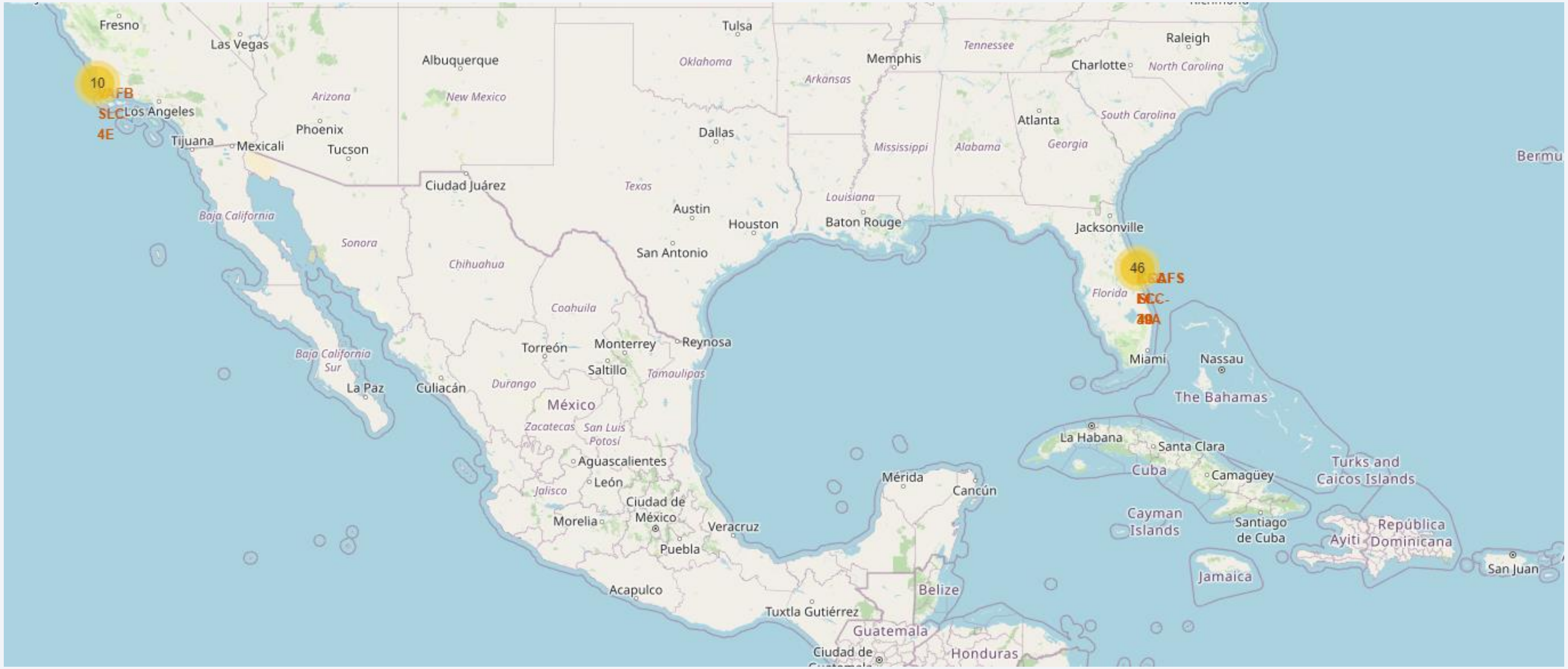
Landing _Outcome	Rank
Success	20
Success (drone ship)	8
Success (ground pad)	6

- The highest success is that of 20 followed by drone ship and ground pad landing

Section 3

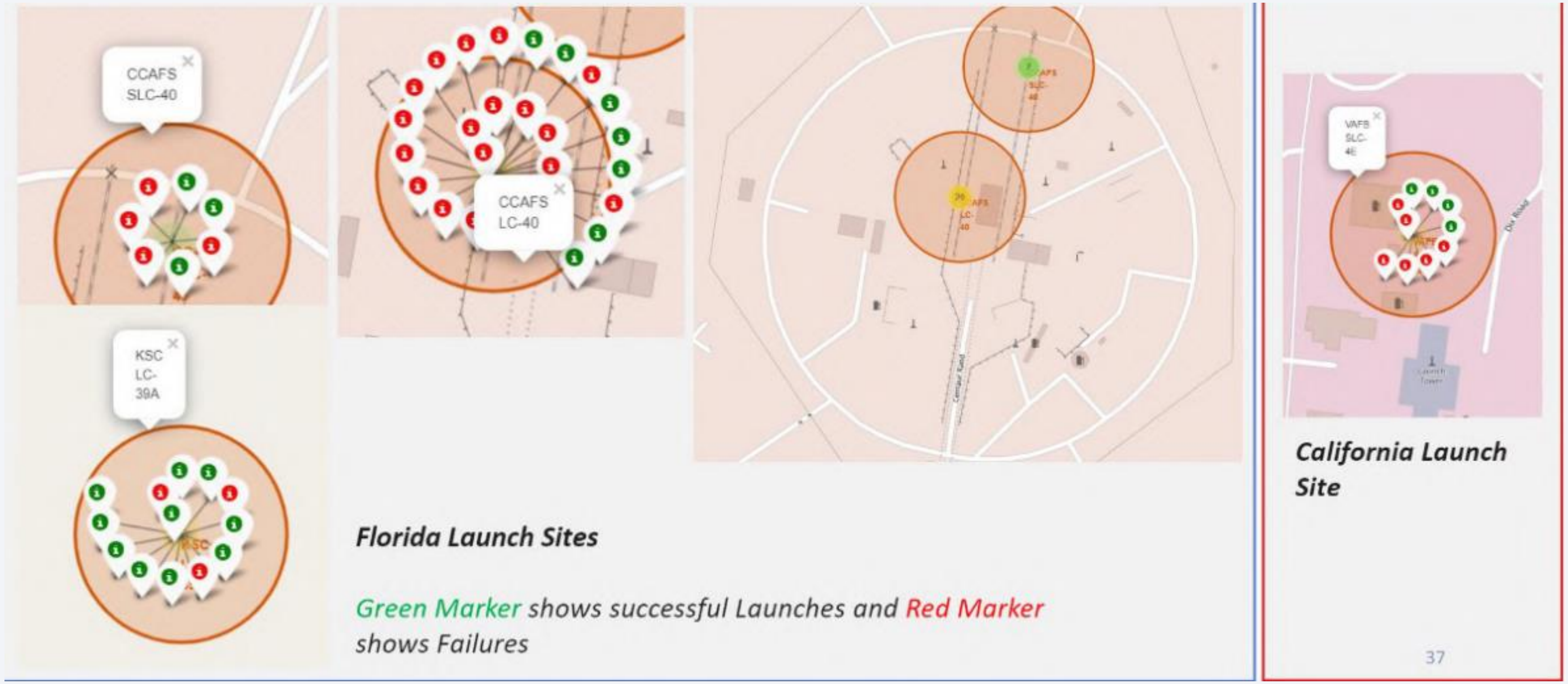
Launch Sites Proximities Analysis

Launch sites across the US



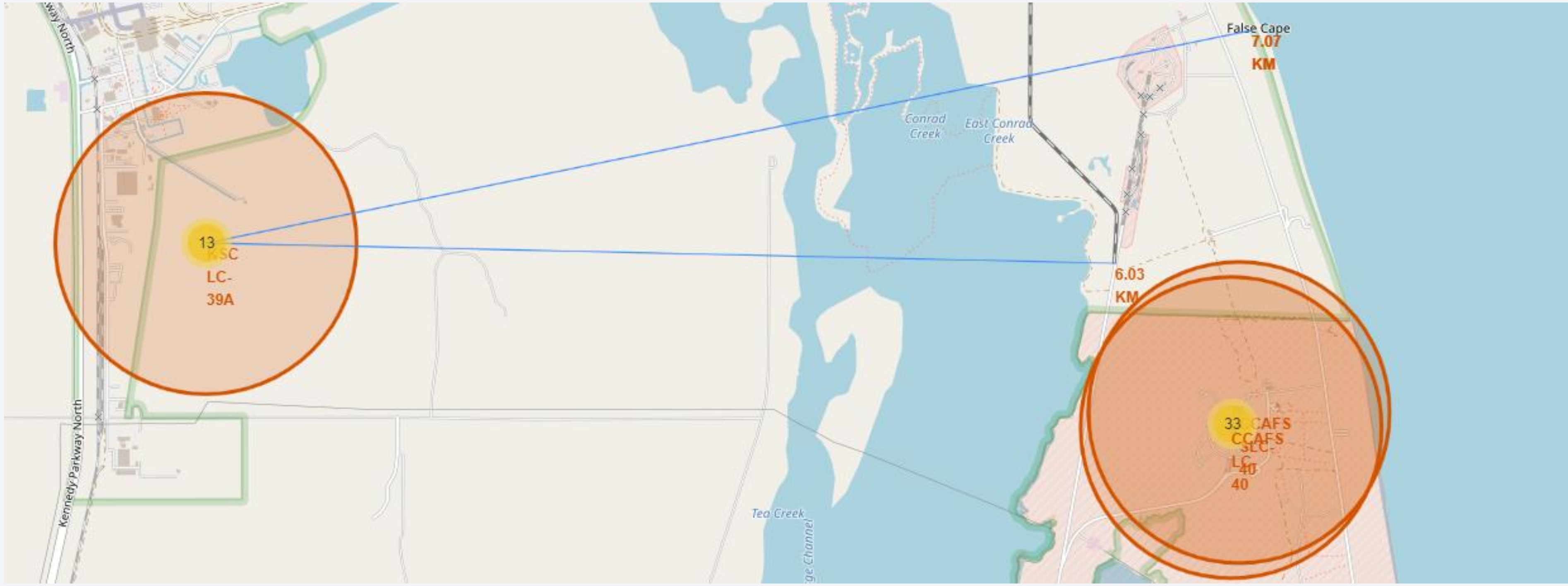
- We can see the markers with the name of the launch site and the total number of launches

Successful markers in the launch sites



- Here we can see the total number of successful / failure markers for the launch site CCAFS SLC 40

Proximity of coastline and railway line to the launch sites

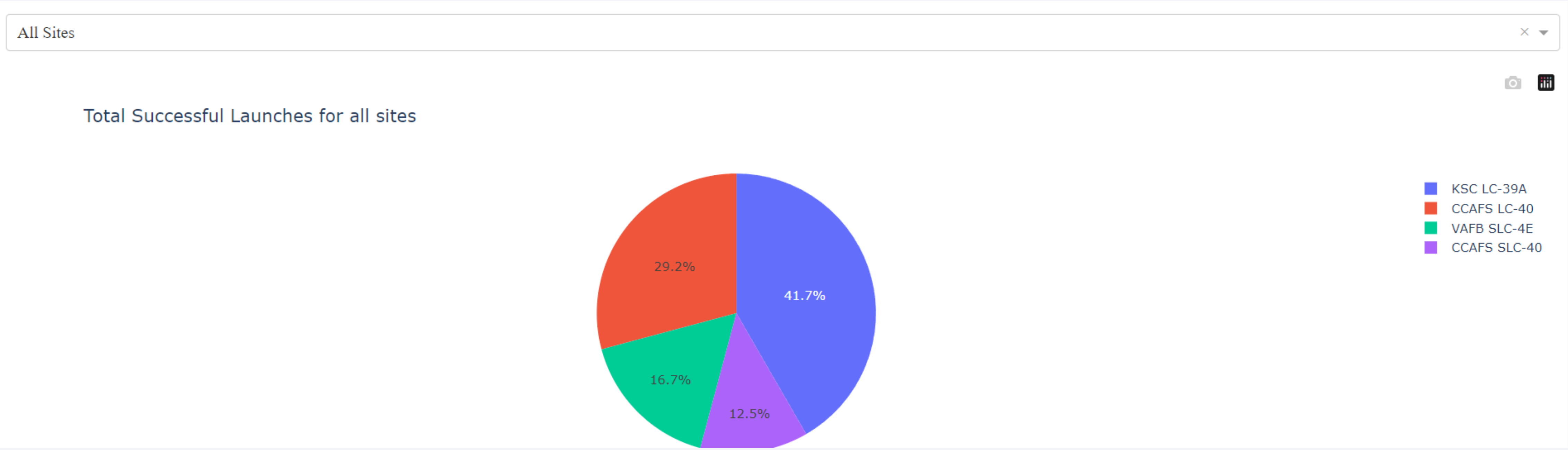


- Here we can see the distance between the launch site and the coastline as well as the railway line
- It is in close proximity to both these points

Section 4

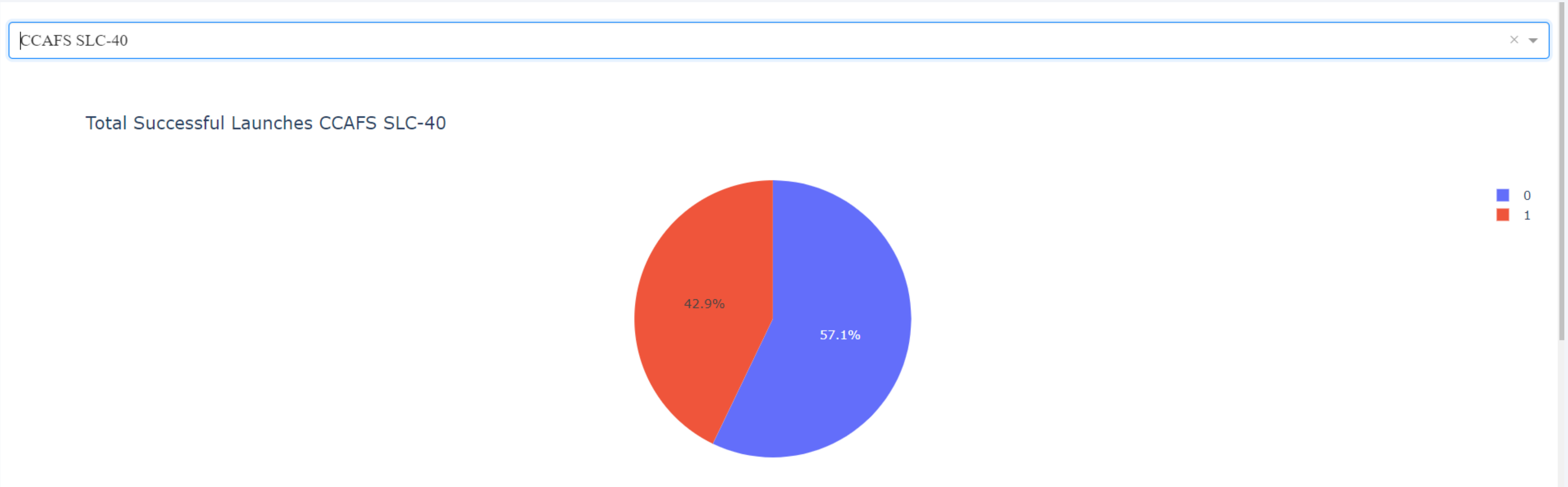
Build a Dashboard with Plotly Dash

Total Successful Launches - KSC LC - 39A



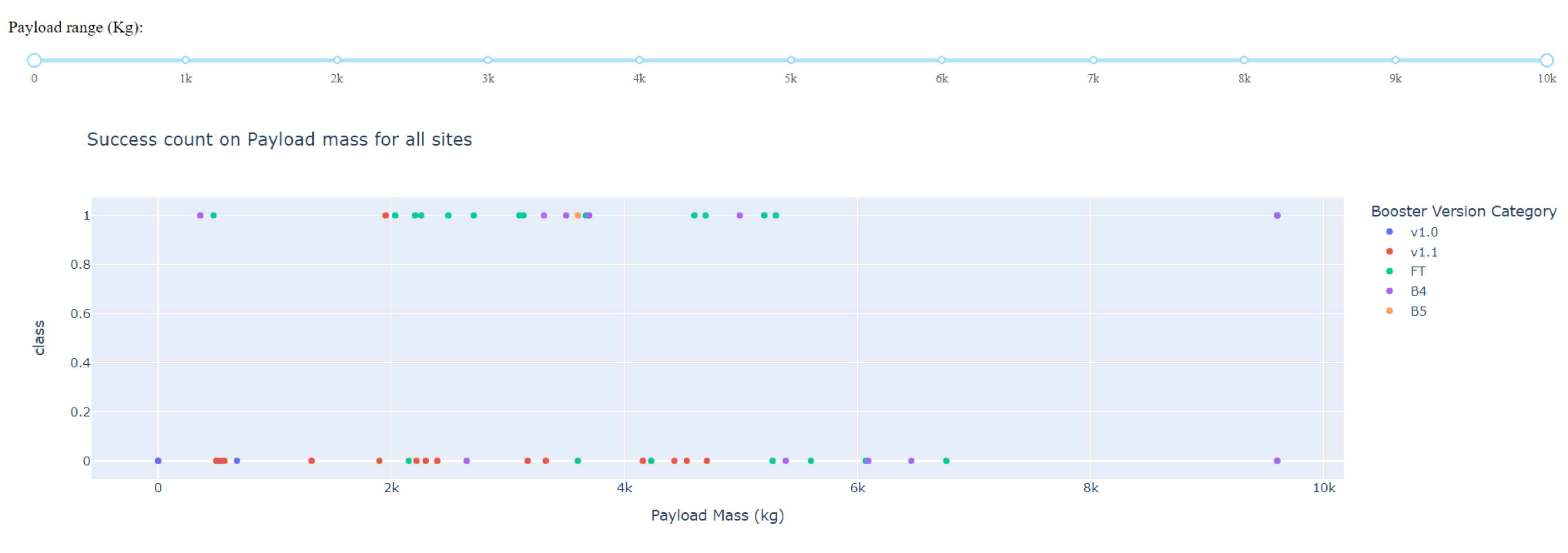
- Of the total launches, KSC LC – 39A has had the most successful launches

Highest success ratio CCAFS SLC - 40



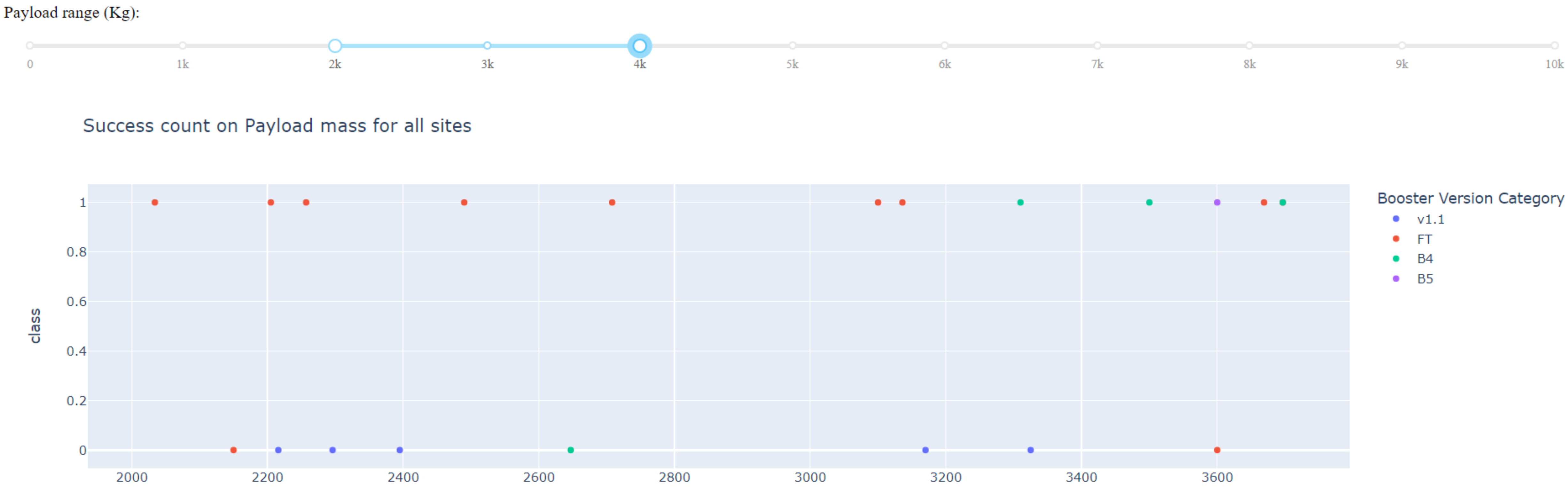
- The highest success ratio is for CCAFS SLC – 40 with a total success ratio of 42.9%

Least successful payloads between 0-2K kg



- As we can see the payload with least success ratio is that between 0-2K kg followed by 4-6k kg

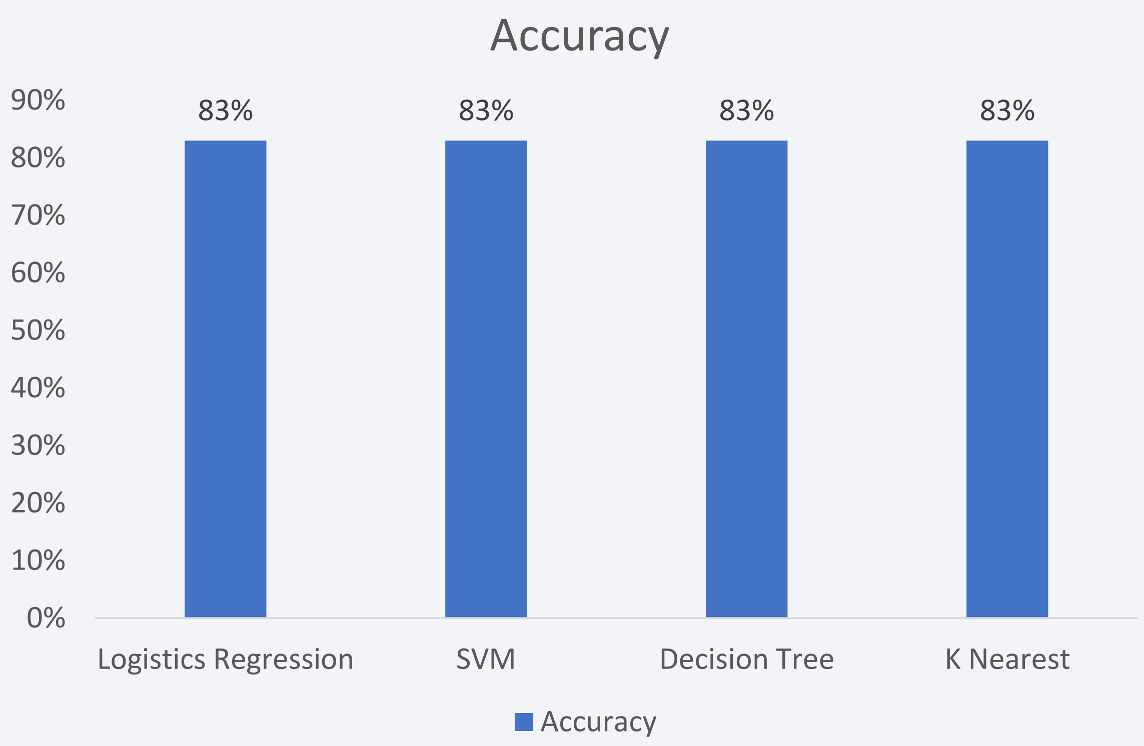
Most successful payloads between 2-4K kg



Section 5

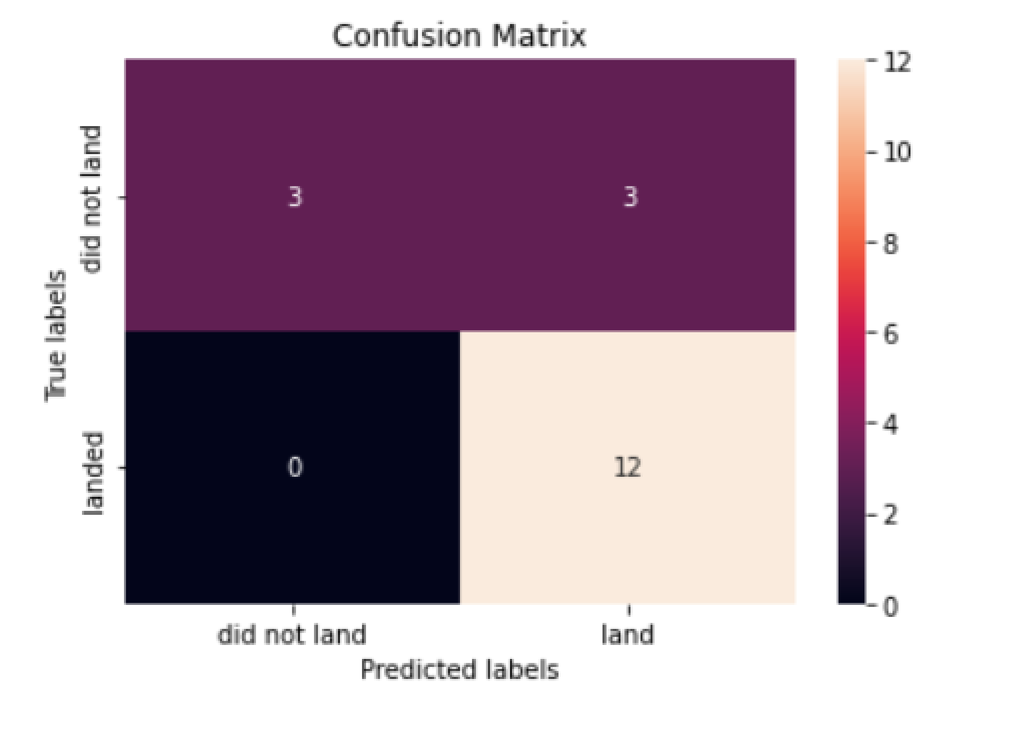
Predictive Analysis (Classification)

Classification Accuracy



All the models had an accuracy of 83%, hence we can choose any model for our prediction

Confusion Matrix



As you can see the confusion metrics we have about 3 false positives in the test set

Conclusions

- Most successful payload launches are between 2-4K
- We can use the SSO orbit to ensure 100% success for the Falcon 9 rocket
- We can predict the success of a landing with an 83% accuracy for futures launches

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!